

PDP11

DDCMP MODE LINE UNIT
MD-11-DZKCE-A

EP-DZKCE-A-DL-A

AUG 1977

COPYRIGHT © 1977

digital

FICHE 1 OF 1

MADE IN USA

801

END IDZKCEASE@
POP10 PAGE: 0001

00010000

770720

POP10 411

HDR1DZKCEASE@

00010000

770720

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZKCE-A-D
PRODUCT NAME: DDCHP MODE LINE UNIT TESTS
DATE: MAY 1977
MAINTAINER: DIAGNOSTICS
AUTHOR: DINESH GORADIA

The information in this document is subject to change without notice and should not be construed as a commitment by Digital Equipment Corporation. Digital Equipment Corporation assumes no responsibility for any errors that may appear in this document.

The software described in this document is furnished under a license and may only be used or copied in accordance with the terms of such license.

Digital Equipment Corporation assumes no responsibility for the use or reliability of its software on equipment that is not supplied by Digital.

Copyright (C) 1977 by Digital Equipment Corporation

1. ABSTRACT

The function of the KMC11 diagnostics is to verify that the option operates according to specifications. The diagnostics verify that there are no malfunctions and the all operations of the 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.

DZKCE 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 DDCMP mode. The modem signals are also checked. DZKCE 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 memory tests
3. DZKCE [REV] DDCMP Line unit tests
4. DZKCF [REV] BITSTUFF Line unit tests
5. DZKCA [REV] KMC11 CPU MICRO-DIAGNOSTICS

2. REQUIREMENTS

2.1 EQUIPMENT

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

2.2 STORAGE

Program will use all BK 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 PROCEDURE

- a. Set switch register to 000200
- b. Depress 'LOAD ADDRESS' key and release
- c. Set SWR to zero for 'AUTO SIZING' or SWR bit0=1 for manual input (questions) or SWR bit7=1 to use existing parameters set up by a previous start or a previously run 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 (DOCHP LINE#)?377
SWITCH PAC#2 (BMB73 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 carriage return to begin execution at the selected test.

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

SW 03 RESELECT 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(SW03) alters that location; therefore if four KMC11s are in the system ***DO NOT*** set switches greater than SW 03 in the up position. This would be a fatal error. do not select more active 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 enabled; and there is a HARD error (constant); SW08 is best. (SW14=1,0, SW10=0, SW09=0, SW08=1). for intermittent errors; SW14=1 will loop on test regardless of error or not error. (SW14=1, SW10=0, SW09=0, SW08=1,0)
3. SW11 Inhibit iterations.
4. SW14 Loop on current test.

4.2 STARTING ADDRESS

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

6. ERRORS

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

6.2 ERROR RECOVERY

If for some reason the 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 regardless of how program was started. Also it is important to use this listing along with the information printed on the TTY to completely isolate problems.

7.2 OPERATING RESTRICTIONS

The first time a KMC11 diagnostic is loaded into core and run the STATUS TABLE must be set up. This is done by manual input (SM00=1) or by autosizing (SM00=0 and SM07=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 SM07=1 on start up.

7.3 HARDWARE CONFIGURATION RESTRICTIONS

KMC11 IOP(MB204)- Jumper W1 must be in,

LINE UNIT(MB201)- Jumpers W1, W2, and W4 must be IN. Jumpers W3, and W5 must be OUT. SW8 of E26 must be in the ON position.

LINE UNIT (MB202)- Jumper W1 must be in. SW8 of E26 must be in the OFF position.

8. MISCELLANEOUS

8.1 EXECUTION TIME

All 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 DZKCE CSR: 175000 VEC: 0300 PASSES: 000001
ERRORS: 000000
```

NOTE: The pass count and error counts are cumulative 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 performed.

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

RUN (1500) The bit in 'RUN' always points to the KMC11 currently being tested. EXAMPLE: (RUN) 1500/000000001000000 Means that KMC11 no.06 is the KMC11 now running.

**KMCROO-KMCRI7
KMSTOO-KMSTI7
(2100)-(2300)** These locations contain the information needed to test up to 16 (decimal) KMC11s sequentially. 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 MB201
 BIT13=1 LINE UNIT IS AN MB202
 BIT12=1 NO LINE UNIT
 BITS 09-11 IS KMC11 BR PRIORITY LEVEL

STAT2: LOW BYTE IS SWITCH PAC#1 (DOCMP 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, if not, the address is updated by 10 and the search continues. A -1 indicates a KMC11 with no CRAM, and 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 'drop-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 interrupt occurs 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 interrupt occurred; the address to which the KMC11 interrupted to is picked up and reported as the vector. NOTE: if the vector reported is not the vector set up by you; there is a problem and AUTO SIZING should not be done.

8.5 SOFTWARE SWITCH REGISTER

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

Control:

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

SWR=XXXXXX NEW?

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

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

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

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

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

Note: for 1pg's line unit m8202-ye users.

CABLE DATA TEST: [TEST 56 TEST 57]

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)

MAINDEC-11-DZKCE-A

002

DECDOC VER 00.04 01-JUN-77 10:37 PAGE 01 PAGE: 0016

DOCUMENT

MAINDEC-11-DZKCE-A

COPYRIGHT 1977
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASS. 01754

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 BMB73 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

2687 ***** TEST 15 *****
DDCMP TEST OF RTS AND OUT ACTIVE
SET SOM AND LOAD OUT DATA SILO
SINGLE STEP 2 DATA CLOCKS, VERIFY
THAT RTS AND ACTIVE ARE SET

2732 ***** 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

2790 ***** TEST 17 *****
DDCMP TRANSMITTER TEST
SINGLE CLOCK THE CHARACTER 0
VERIFY EACH BIT POSITION AS IT
PASSES THE BIT WINDOW (SI BIT)
ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE

2843 ***** TEST 20 *****
DDCMP TRANSMITTER TEST
SINGLE CLOCK THE CHARACTER 125
VERIFY EACH BIT POSITION AS IT
PASSES THE BIT WINDOW (SI BIT)
ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE

2896 ***** TEST 21 *****
DDCMP TRANSMITTER TEST
SINGLE CLOCK THE CHARACTER 252
VERIFY EACH BIT POSITION AS IT
PASSES THE BIT WINDOW (SI BIT)
ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE

2949 ***** TEST 22 *****
DDCMP TRANSMITTER TEST
SINGLE CLOCK THE CHARACTER 377
VERIFY EACH BIT POSITION AS IT
PASSES THE BIT WINDOW (SI BIT)
ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE

3002 ***** TEST 23 *****
DDCMP TRANSMITTER TEST
SINGLE CLOCK A BINARY COUNT PATTERN
VERIFY EACH BIT POSITION AS IT
PASSES THE BIT WINDOW (SI BIT)
ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
AND R5 CONTAINS THE CHARACTER THAT FAILED

3065 ***** TEST 24 *****
DDCMP STRIP SYNC TEST
SET LU LOOP, SINGLE STEP 5 SYNCs
VERIFY THAT IN PACTIVE DOES NOT SET

3095 ***** TEST 25 *****
DDCMP IN ACTIVE TEST
SET LU LOOP, SINGLE STEP 5 SYNCs AND A NON-SYNC (301)
VERIFY THAT IN ACTIVE IS SET

3125 ***** TEST 26 *****
DDCMP IN ACTIVE TEST
SET LU LOOP, SINGLE STEP 1 SYNC AND A NON-SYNC (301)
VERIFY THAT IN ACTIVE DOES NOT SET

3155 ***** TEST 27 *****
DDCMP IN ACTIVE TEST
SET LU LOOP, SINGLE STEP 2 SYNCs AND A NON-SYNC (301)
VERIFY THAT IN ACTIVE IS SET

3185 ***** TEST 30 *****
IN CLEAR TEST
SYNC UP RECEIVER AND TRANSMIT A CHARACTER
WAIT FOR IN RDY, THEN SET IN CLEAR
VERIFY THAT IN ACTIVE AND IN RDY ARE CLEARED

3236 ***** TEST 31 *****
DDCMP BASIC RECEICER TEST
SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 0
VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED

3275 ***** TEST 32 *****
DDCMP BASIC RECEICER TEST
SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 125
VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED

3314 ***** TEST 33 *****
DDCMP BASIC RECEICER TEST
SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 252
VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED

3353 ***** TEST 34 *****
DDCMP BASIC RECEICER TEST
SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 377
VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED

3392 ***** TEST 35 *****
DDCMP DATA TEST
THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
CHECKING EACH CHARACTER AS IT IS RECEIVED

3433 ***** TEST 36 *****
DDCMP DATA TEST
THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
CHECKING EACH CHARACTER AS IT IS RECEIVED
THIS TEST IS EXACTLY THE SAME AS THE LAST TEST,
EXCEPT LINE UNIT LOOP IS SET IN LU REGISTER 12

3479 ***** TEST 37 *****
TRANSMITTER MARK TEST
SINGLE CLOCK 3 SYNCs AND A 301 AND 20 EXTRA
CLOCK TICKS, VERIFY THAT A 301, A 377 AND A 377
WERE RECEIVED INDICATING THAT THE TRANSMITTER WENT
TO A MARK STATE FOR 16 BITS WHEN OUT SILO WAS EMPTY

3526 ***** TEST 40 *****
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

3579 ***** TEST 41 *****
CABLE TURNAROUND TEST
CLEAR LINE UNIT LOOP, LOAD OUT DATA SILO
VERIFY THAT ALL MODEM SIGNALS ARE SET

3627 ***** TEST 42 *****
TEST OF CRC OPERATION
USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
0, VERIFY THE LSB OF THE BCC ON EACH SHIFT
TEST TRANSMITTER FIRST THEN THE RECEIVER BCC

3704 ***** TEST 43 *****
TEST OF CRC OPERATION
USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
377, VERIFY THE LSB OF THE BCC ON EACH SHIFT
TEST TRANSMITTER FIRST THEN THE RECEIVER BCC

3781 ***** TEST 44 *****
 TEST OF CRC OPERATION
 USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
 125, VERIFY THE LSB OF THE BCC ON EACH SHIFT
 TEST TRANSMITTER FIRST THEN THE RECEIVER BCC

3858 ***** TEST 45 *****
 TEST OF CRC OPERATION
 USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
 252, VERIFY THE LSB OF THE BCC ON EACH SHIFT
 TEST TRANSMITTER FIRST THEN THE RECEIVER BCC

3935 ***** TEST 46 *****
 TRANSMITTER CRC TEST
 USING THE CRC16 POLYNOMIAL, SINGLE CLOCK A BINARY
 COUNT PATTERN, VERIFY THE LSB OF THE TRANSMITTER BCC ON EACH SHIFT

4003 ***** TEST 47 *****
 RECEIVER CRC TEST
 USING THE CRC16 POLYNOMIAL, SINGLE CLOCK A BINARY
 COUNT PATTERN, VERIFY THE LSB OF THE RECEIVER BCC ON EACH SHIFT

4071 ***** TEST 50 *****
 TRANSMITTER DDCMP CRC TEST
 THIS TEST TRANSMITS A FOUR CHARACTER MESSAGE WITH CRC
 BOTH DATA AND THE BCC ARE VERIFIED IN THE BIT
 WINDOW. THE FOUR CHARACTERS ARE 0,125,252,377
 THE TRANSMITTER IS CHECKED FOR GOING TO A MARK STATE AFTER THE BCC

4174 ***** TEST 51 *****
 RECEIVER DDCMP CRC TEST
 THIS TEST CLOCKS A FOUR CHARACTER MESSAGE WITH BCC
 AND VERIFYS CORRECT DATA RECEPTION AND BCC MATCH
 THE FOUR CHARACTER MESSAGE IS 0,125,252,377

4233 ***** TEST 52 *****
 DDCMP EOM FUNCTION TEST
 THIS TEST LOADS OUT SILO WITH: 2 SYNCs, 4 CHAR MESSAGE, EOM
 4 CHARACTER MESS, EOM. THE DATA STREAM IS CHECKED TO BE
 4 CHAR, BCC, 4 CHAR, BCC, MARKS. THIS TEST VERIFYS THAT
 THE CHARACTERS LOADED WITH EOM SET ARE LOST
 ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
 THE FOUR CHARACTER MESSAGE IS 0,125,252,377
 RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED

4481 ***** TEST 53 *****
 DDCMP EOM FUNCTION TEST
 THIS TEST LOADS OUT SILO WITH: 2 SYNCs, 4 CHAR MESSAGE, EOM
 SOM, 4 CHAR MESS, EOM. THE DATA STREAM IS CHECKED TO BE
 4 CHAR, BCC, 4 CHAR, BCC, MARKS. THIS TEST VERIFYS THAT
 THE CHARACTERS LOADED WITH EOM SET ARE LOST
 ALSO THAT THE CHAR LOADED WITH SOM IS NOT IN THE BCC
 ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
 THE FOUR CHARACTER MESSAGE IS 0,125,252,377

RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED

4761 ***** TEST 54 *****
 EMPTY SILO TEST
 LOAD SILO WITH 2 SYNCs, 4 CHAR MESSAGE, SINGLE CLOCK
 UNTIL THE SILO IS EMPTY, LOAD 4 MORE CHARACTERS IN THE
 SILO. GIVE MORE TICKS, AND VERIFY THAT ONLY THE FIRST
 4 CHARACTER MESSAGE WAS RECEIVED AND THAT RTS IS CLEAR

4827 ***** TEST 55 *****
 HALF DUPLEX TEST
 SET LINE UNIT LOOP AND HALF DUPLEX, SEND SYNCs AND A
 MESSAGE. VERIFY THAT IN-ACTIVE AND IN-READY ARE CLEAR

4864 ***** TEST 56 *****
 DDCMP CABLE DATA TEST
 THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
 4 SYNCs, 16 CHAR, EOM, 16 CHAR, EOM, 16 CHAR, EOM
 THE 16 CHARACTERS INCLUDE A FLOATING ONE AND ZERO
 THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
 RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
 LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST

4961 ***** TEST 57 *****
 DDCMP CABLE DATA TEST
 THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
 4 SYNCs, 59 DATA CHARACTERS, EOM WITH GARBAGE CHARACTER
 THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
 RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
 LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST

```

1  .TITLE MAINDEC-11-DZKCE-A
2  .*COPYRIGHT (C) 1976
3  .*DIGITAL EQUIPMENT CORP.
4  .*MAYNARD, MASS. 01754
5  .*
6  .*PROGRAM BY DINESH GORADIA
7  .*
8  .*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
9  .*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
10 .*
11
12
13
14
15
16 .*MAINDEC-11-DZKCE-A KMC11 DDCMP LINE UNIT TESTS
17 .*COPYRIGHT 1976, DIGITAL EQUIPMENT CORP., MAYNARD, MASS. 01754
18 .*-----
19
20 .STARTING PROCEDURE
21 .LOAD PROGRAM
22 .LOAD ADDRESS, 000200
23 .SWR=0 AUTOSIZE KMC11
24 .SW07=1 USE CURRENT KMC11 PARAMETERS
25 .SW00=1 INPUT NEW KMC11 PARAMETERS
26 .PRESS START
27 .PROGRAM WILL TYPE "MAINDEC-11-DZKCE-A KMC11 DDCMP LINE UNIT TESTS"
28 .PROGRAM WILL TYPE STATUS MAP
29 .PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED
30 .AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
31 .AND THEN RESUME TESTING
32 .SUBSEQUENT RESTARTS WILL NOT TYPE PROGRAM TITLE
33
34 .SBTTL BASIC DEFINITIONS
35
36 .*INITIAL ADDRESS OF THE STACK POINTER *** 1200 ***
37 001200 STACK= 1200
38 .EQUIV EMT,ERROR ;;BASIC DEFINITION OF ERROR CALL
39 .EQUIV IOT,SCOPE ;;BASIC DEFINITION OF SCOPE CALL
40
41 .*MISCELLANEOUS DEFINITIONS
42 000011 HT= 11 ;;CODE FOR HORIZONTAL TAB
43 000012 LF= 12 ;;CODE FOR LINE FEED
44 000015 CR= 15 ;;CODE FOR CARRIAGE RETURN
45 000200 CRLF= 200 ;;CODE FOR CARRIAGE RETURN-LINE FEED
46 177776 PS= 177776 ;;PROCESSOR STATUS WORD
47 .EQUIV PS,PSW
48 177774 STKLM= 177774 ;;STACK LIMIT REGISTER
49 177772 PIRQ= 177772 ;;PROGRAM INTERRUPT REQUEST REGISTER
50 177570 DSWR= 177570 ;;HARDWARE SWITCH REGISTER
51 177570 DOISP= 177570 ;;HARDWARE DISPLAY REGISTER
52
53 .*GENERAL PURPOSE REGISTER DEFINITIONS
54 000000 R0= %0 ;;GENERAL REGISTER
55 000001 R1= %1 ;;GENERAL REGISTER
56 000002 R2= %2 ;;GENERAL REGISTER

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 3
 DZKCE.P11 12-MAY-77 12:23 BASIC DEFINITIONS

PAGE: 0024

```

57      000003      R3=      %3      ;; GENERAL REGISTER
58      000004      R4=      %4      ;; GENERAL REGISTER
59      000005      R5=      %5      ;; GENERAL REGISTER
60      000006      R6=      %6      ;; GENERAL REGISTER
61      000007      R7=      %7      ;; GENERAL REGISTER
62      000006      SP=      %6      ;; STACK POINTER
63      000007      PC=      %7      ;; PROGRAM COUNTER
64
65      ;*PRIORITY LEVEL DEFINITIONS
66      000000      PR0=      0      ;; PRIORITY LEVEL 0
67      000040      PR1=      40     ;; PRIORITY LEVEL 1
68      000100      PR2=      100    ;; PRIORITY LEVEL 2
69      000140      PR3=      140    ;; PRIORITY LEVEL 3
70      000200      PR4=      200    ;; PRIORITY LEVEL 4
71      000240      PR5=      240    ;; PRIORITY LEVEL 5
72      000300      PR6=      300    ;; PRIORITY LEVEL 6
73      000340      PR7=      340    ;; PRIORITY LEVEL 7
74
75      ;*"SWITCH REGISTER" SWITCH DEFINITIONS
76      100000      SW15= 100000
77      040000      SW14= 40000
78      020000      SW13= 20000
79      010000      SW12= 10000
80      004000      SW11= 4000
81      002000      SW10= 2000
82      001000      SW09= 1000
83      000400      SW08= 400
84      000200      SW07= 200
85      000100      SW06= 100
86      000040      SW05= 40
87      000020      SW04= 20
88      000010      SW03= 10
89      000004      SW02= 4
90      000002      SW01= 2
91      000001      SW00= 1
92      .EQUIV      SW09,SW9
93      .EQUIV      SW08,SW8
94      .EQUIV      SW07,SW7
95      .EQUIV      SW06,SW6
96      .EQUIV      SW05,SW5
97      .EQUIV      SW04,SW4
98      .EQUIV      SW03,SW3
99      .EQUIV      SW02,SW2
100     .EQUIV      SW01,SW1
101     .EQUIV      SW00,SW0
102
103     ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
104     100000      BIT15= 100000
105     040000      BIT14= 40000
106     020000      BIT13= 20000
107     010000      BIT12= 10000
108     004000      BIT11= 4000
109     002000      BIT10= 2000
110     001000      BIT09= 1000
111     000400      BIT08= 400
112     000200      BIT07= 200

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 4
 DZKCE.P11 12-MAY-77 12:23 BASIC DEFINITIONS

PAGE: 0025

```

113      000100      BIT06= 100
114      000040      BIT05= 40
115      000020      BIT04= 20
116      000010      BIT03= 10
117      000004      BIT02= 4
118      000002      BIT01= 2
119      000001      BIT00= 1
120      .EQUIV      BIT09,BIT9
121      .EQUIV      BIT08,BIT8
122      .EQUIV      BIT07,BIT7
123      .EQUIV      BIT06,BIT6
124      .EQUIV      BIT05,BIT5
125      .EQUIV      BIT04,BIT4
126      .EQUIV      BIT03,BIT3
127      .EQUIV      BIT02,BIT2
128      .EQUIV      BIT01,BIT1
129      .EQUIV      BIT00,BIT0
130
131      ;#BASIC "CPU" TRAP VECTOR ADDRESSES
132      ERRVEC= 4      ;; TIME OUT AND OTHER ERRORS
133      RESVEC= 10     ;; RESERVED AND ILLEGAL INSTRUCTIONS
134      TBITVEC= 14    ;; "T" BIT
135      TRTVEC= 14     ;; TRACE TRAP
136      BPTVEC= 14     ;; BREAKPOINT TRAP (BPT)
137      IOTVEC= 20     ;; INPUT/OUTPUT TRAP (IOT) **SCOPE**
138      PWRVEC= 24     ;; POWER FAIL
139      EMTVEC= 30     ;; EMULATOR TRAP (EMT) **ERROR**
140      TRAPVEC= 34    ;; "TRAP" TRAP
141      TKVEC= 60      ;; TTY KEYBOARD VECTOR
142      TPVEC= 64      ;; TTY PRINTER VECTOR
143      PIRQVEC= 240   ;; PROGRAM INTERRUPT REQUEST VECTOR
144
145
146
147
148      ;INSTRUCTION DEFINITIONS
149      ;-----
150
151      005746      PUSH1SP=5746      ;; DECREMENT PROCESSOR STACK 1 WORD
152      005726      POP1SP=5726      ;; INCREMENT PROCESSOR STACK 1 WORD
153      010046      PUSHRO=10046     ;; SAVE RO ON STACK
154      012600      POPRO=12600      ;; RESTORE RO FROM STACK
155      024646      PUSH2SP=24646    ;; DECREMENT STACK TWICE
156      022626      POP2SP=22626    ;; INCREMENT STACK TWICE
157      .EQUIV      EMT,HLT          ;; BASIC DEFINITION OF ERROR CALL
158
159
160

```

TRAPCATCHER FOR UNEXPECTED INTERRUPTS

```

161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207

```

```

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

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

.=20
$SCOPE          ; SCOPE LOOP HANDLER.
PR7              ; SERVICE AT LEVEL 7.
$PWRDN          ; POWER FAIL HANDLER
PR7              ; SERVICE AT LEVEL 7
$ERRR           ; ERROR HANDLER
PR7              ; SERVICE AT LEVEL 7
$STRAP          ; GENERAL HANDLER DISPATCH SERVICE
PR7              ; SERVICE AT LEVEL 7

.SBTTL ACT11 HOOKS

; *****
; HOOKS REQUIRED BY ACT11
$SVPC=          ; SAVE PC
.=46            ; 1) SET LOC. 46 TO ADDRESS OF SENDAD IN .SEOP
$SENDAD         ;
.=52            ; 2) SET LOC. 52 TO ZERO
WORD 0          ;
.= $SVPC        ; RESTORE PC

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

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

.=1000
MTITLE: .ASCII <200><12>/MAINDEC-11-DZKCE-2/<200>
        .ASCIIZ /KMC11 DDCMP LINE UNIT TESTS/<200>

DSWR = 177570
DISP = 177570

```

```

208      .SBTTL COMMON TAGS
209
210      ;*****
211      ;THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
212      ;USED IN THE PROGRAM.
213
214      001200      .=1200
215      001200      SCMTAG:      ;; START OF COMMON TAGS
216      001200      000000      .WORD      0      ;CONTAINS THE TEST NUMBER
217      001202      000      .BYTE      0      ;CONTAINS ERROR FLAG
218      001203      000      .BYTE      0      ;CONTAINS SUBTEST ITERATION COUNT
219      001204      000000      .WORD      0      ;CONTAINS SCOPE LOOP ADDRESS
220      001206      000000      .WORD      0      ;CONTAINS SCOPE RETURN FOR ERRORS
221      001210      000000      .WORD      0      ;CONTAINS TOTAL ERRORS DETECTED
222      001212      000000      .WORD      0      ;CONTAINS ITEM CONTROL BYTE
223      001214      000      .BYTE      0      ;CONTAINS MAX. ERRORS PER TEST
224      001215      001      .BYTE      1      ;CONTAINS PC OF LAST ERROR INSTRUCTION
225      001216      000000      .WORD      0      ;CONTAINS ADDRESS OF 'GOOD' DATA
226      001220      000000      .WORD      0      ;CONTAINS ADDRESS OF 'BAD' DATA
227      001222      000000      .WORD      0      ;CONTAINS 'GOOD' DATA
228      001224      000000      .WORD      0      ;CONTAINS 'BAD' DATA
229      001226      000000      .WORD      0      ;RESERVED--NOT TO BE USED
230      001230      000000      .WORD      0
231      001232      000000      .WORD      0
232      001234      000      .BYTE      0      ;AUTOMATIC MODE INDICATOR
233      001235      000      .BYTE      0      ;INTERRUPT MODE INDICATOR
234      001236      000000      .WORD      0
235      001240      177570      .WORD      DSWR      ;ADDRESS OF SWITCH REGISTER
236      001242      177570      .WORD      DOISP      ;ADDRESS OF DISPLAY REGISTER
237      001244      177560      .WORD      0      ;TTY KBD STATUS
238      001246      177562      .WORD      0      ;TTY KBD BUFFER
239      001250      177564      .WORD      0      ;TTY PRINTER STATUS REG. ADDRESS
240      001252      177566      .WORD      0      ;TTY PRINTER BUFFER REG. ADDRESS
241      001254      000      .BYTE      0      ;CONTAINS NULL CHARACTER FOR FILLS
242      001255      002      .BYTE      2      ;CONTAINS # OF FILLER CHARACTERS REQUIRED
243      001256      012      .BYTE      12      ;INSERT FILL CHARS. AFTER A "LINE FEED"
244      001257      000      .BYTE      0      ;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
245      001260      000000      .WORD      0      ;CONTAINS THE ADDRESS FROM
246      001262      000000      .WORD      0      ;WHICH ($REG0) WAS OBTAINED
247      001264      000000      .WORD      0      ;CONTAINS (($REG0)+0)
248      001266      000000      .WORD      0      ;CONTAINS (($REG0)+2)
249      001270      000000      .WORD      0      ;CONTAINS (($REG0)+4)
250      001272      000000      .WORD      0      ;CONTAINS (($REG0)+6)
251      001274      000000      .WORD      0      ;CONTAINS (($REG0)+10)
252      001276      000000      .WORD      0      ;CONTAINS (($REG0)+12)
253      001300      000000      .WORD      0      ;USER DEFINED
254      001302      000000      .WORD      0      ;USER DEFINED
255      001304      000000      .WORD      0      ;USER DEFINED
256      001306      000000      .WORD      0      ;USER DEFINED
257      001310      000000      .WORD      0      ;USER DEFINED
258      001312      077      .ASCII      '?'      ;MAX. NUMBER OF ITERATIONS
259      001313      015      .ASCII      '<15>'      ;QUESTION MARK
260      001314      000012      .ASCII      '<12>'      ;CARRIAGE RETURN
261      001315      000012      .ASCII      '<12>'      ;LINE FEED
262      ;*****
263      .SBTTL APT MAILBOX-ETABLE

```

264			*****		
265			..EVEN		
266			\$MAIL:		APT MAILBOX
267	001316		\$MSGTY: .WORD	AMSGTY	MESSAGE TYPE CODE
268	001316	000000	\$FATAL: .WORD	AFATAL	FATAL ERROR NUMBER
269	001320	000000	\$TESTN: .WORD	ATESTN	TEST NUMBER
270	001322	000000	\$PASS: .WORD	APASS	PASS COUNT
271	001324	000000	\$DEVCT: .WORD	ADEVCT	DEVICE COUNT
272	001326	000000	\$UNIT: .WORD	AUNIT	I/O UNIT NUMBER
273	001330	000000	\$MSGAD: .WORD	AMSGAD	MESSAGE ADDRESS
274	001332	000000	\$MSGLG: .WORD	AMSGLG	MESSAGE LENGTH
275	001334	000000	\$ETABLE:		APT ENVIRONMENT TABLE
276	001336		\$ENV: .BYTE	RENV	ENVIRONMENT BYTE
277	001336	002	\$ENVH: .BYTE	RENVH	ENVIRONMENT MODE BITS
278	001337	000	\$SWREG: .WORD	ASWREG	APT SWITCH REGISTER
279	001340	000000	\$USWR: .WORD	AUSWR	USER SWITCHES
280	001342	000000	\$CPUOP: .WORD	ACPUOP	CPU TYPE, OPTIONS
281	001344	000000			BITS 15-11=CPU TYPE
282					11/04=01, 11/05=02, 11/20=03, 11/40=04, 11/45=05
283					11/70=06, P00=07, Q=10
284					BIT 10=REAL TIME CLOCK
285					BIT 9=FLOATING POINT PROCESSOR
286					BIT 8=MEMORY MANAGEMENT
287					;; HIGH ADDRESS, M.S. BYTE
288	001346	000	\$MAMS1: .BYTE	AMAMS1	MEM. TYPE, BLK#1
289	001347	000	\$MTYP1: .BYTE	AMTYP1	MEM. TYPE BYTE -- (HIGH BYTE)
290					900 NSEC CORE=001
291					300 NSEC BIPOLAR=002
292					500 NSEC MOS=003
293					;; HIGH ADDRESS, BLK#1
294	001350	000000	\$MAOR1: .WORD	AMAOR1	MEM. LAST ADDR.=3 BYTES, THIS WORD AND LOW OF "TYPE" ABOVE
295					;; HIGH ADDRESS, M.S. BYTE
296	001352	000	\$MAMS2: .BYTE	AMAMS2	MEM. TYPE, BLK#2
297	001353	000	\$MTYP2: .BYTE	AMTYP2	MEM. LAST ADDRESS, BLK#2
298	001354	000000	\$MAOR2: .WORD	AMAOR2	HIGH ADDRESS, M.S. BYTE
299	001356	000	\$MAMS3: .BYTE	AMAMS3	MEM. TYPE, BLK#3
300	001357	000	\$MTYP3: .BYTE	AMTYP3	MEM. LAST ADDRESS, BLK#3
301	001360	000000	\$MAOR3: .WORD	AMAOR3	HIGH ADDRESS, M.S. BYTE
302	001362	000	\$MAMS4: .BYTE	AMAMS4	MEM. TYPE, BLK#4
303	001363	000	\$MTYP4: .BYTE	AMTYP4	MEM. LAST ADDRESS, BLK#4
304	001364	000000	\$MAOR4: .WORD	AMAOR4	INTERRUPT VECTOR#1, BUS PRIORITY#1
305	001366	000000	\$SVECT1: .WORD	AVECT1	INTERRUPT VECTOR#2, BUS PRIORITY#2
306	001370	000000	\$SVECT2: .WORD	AVECT2	BASE ADDRESS OF EQUIPMENT UNDER TEST
307	001372	000000	\$BASE: .WORD	ABASE	DEVICE MAP
308	001374	000000	\$DEVH: .WORD	PDEVH	CONTROLLER DESCRIPTION WORD#1
309	001376	000000	\$CDW1: .WORD	ACDW1	CONTROLLER DESCRIPTION WORD#2
310	001400	000000	\$CDW2: .WORD	ACDW2	DEVICE DESCRIPTOR WORD#0
311	001402	000000	\$DDW0: .WORD	ADDW0	DEVICE DESCRIPTOR WORD#1
312	001404	000000	\$DDW1: .WORD	ADDW1	DEVICE DESCRIPTOR WORD#2
313	001406	000000	\$DDW2: .WORD	ADDW2	DEVICE DESCRIPTOR WORD#3
314	001410	000000	\$DDW3: .WORD	ADDW3	DEVICE DESCRIPTOR WORD#4
315	001412	000000	\$DDW4: .WORD	ADDW4	DEVICE DESCRIPTOR WORD#5
316	001414	000000	\$DDW5: .WORD	ADDW5	DEVICE DESCRIPTOR WORD#6
317	001416	000000	\$DDW6: .WORD	ADDW6	DEVICE DESCRIPTOR WORD#7
318	001420	000000	\$DDW7: .WORD	ADDW7	DEVICE DESCRIPTOR WORD#8
319	001422	000000	\$DDW8: .WORD	ADDW8	

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 8
 DZKCE.P11 12-MAY-77 12:23 APT MAILBOX-ETABLE

PAGE: 0029

320 001424 000000
 321 001426 000000
 322 001430 000000
 323 001432 000000
 324 001434 000000
 325 001436 000000
 326 001440 000000

SDDW9: .WORD ADDW9 : DEVICE DESCRIPTOR WORD#9
 SDDW10: .WORD ADDW10 : DEVICE DESCRIPTOR WORD#10
 SDDW11: .WORD ADDW11 : DEVICE DESCRIPTOR WORD#11
 SDDW12: .WORD ADDW12 : DEVICE DESCRIPTOR WORD#12
 SDDW13: .WORD ADDW13 : DEVICE DESCRIPTOR WORD#13
 SDDW14: .WORD ADDW14 : DEVICE DESCRIPTOR WORD#14
 SDDW15: .WORD ADDW15 : DEVICE DESCRIPTOR WORD#15

327
 328
 329 001442

SETEND:

330
 331
 332
 333
 334 001442 000000
 335 001444 000000

PROGRAM CONTROL PARAMETERS

NEXT: .WORD 0 : ADDRESS OF NEXT TEST TO BE EXECUTED
 LOCK: .WORD 0 : ADDRESS FOR LOCK CURRENT DATA

PROGRAM VARIABLES

336
 337
 338
 339 001446 000000
 340 001450 000000
 341 001452 000000
 342 001454 000000
 343 001456 000000
 344 001460 000000
 345 001462 000000
 346 001464 000001
 347 001466 000000
 348 001470 000001
 349 001472 000001
 350 001474 000001
 351 001476 000001
 352 001500 000000

STATSW: .WORD 0 : SWITCHES AT START OF PROGRAM
 STAT: .WORD 0 : KM STATUS WORD STORAGE
 CLKX: .WORD 0 :
 MASKX: .WORD 0 :
 SAVSP: .WORD 0 : STACK POINTER STORAGE
 SAVPL: .WORD 0 : PROGRAM COUNTER STORAGE
 ZERO: .WORD 0 :
 ONE: .WORD 1 :
 MEMLIM: .WORD 0 : HIGHEST LOCATION FOR NPR'S
 KMACTV: .BLKW 1 : KMC11 SELECTED ACTIVE
 KMINUM: .BLKW 1 : OCTAL NUMBER OF KMC11'S
 SAVACT: .BLKW 1 : ORIGINAL ACTIVE DEVICES.
 SAVNUM: .BLKW 1 : WORKABLE NUMBER.
 RUN: .WORD 0 : POINTER TO RUNNING DEVICES
 .EVEN

353
 354 001502 002072
 355 001504 002276

CREAM: .WORD KM.MAP-6 : TABLE POINTER
 MILK: .WORD CNT.MAP-4 : TABLE POINTER

PROGRAM CONTROL FLAGS

356
 357
 358
 359 001506 000
 360 001510 000
 361 001511 000
 362 001511 000

INIFLG: .BYTE 0 : PROGRAM INITIALIZING FLAG
 .EVEN
 LOKFLG: .BYTE 0 : LOCK ON CURRENT TEST FLAG
 QV.FLG: .BYTE 0 : QUICK VERIFY FLAG
 .EVEN : ON FIRST PASS OF EACH KMC11 ITERATIONS WILL BE SUPPRES

363
 364

ERROR POINTER TABLE

.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 SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
 ; *NOTE1: IF SITEMB IS 0 THE ONLY PERTINENT DATA IS (SERRPC).
 ; *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

SERRTB:
.EVEN

; * DF ;; DOES NOT APPLY IN THIS DIAGNOSTIC.

365				
366				
367				
368				
369				
370				
371				
372				
373				
374				
375				
376				
377				
378				
379	001512			
380				
381			DF	
382	001512	000000	0	
383	001514	000000	0	
384	001516	000000	0	
385	001520	031610	EM1	
386	001522	032614	DH2	; ERROR 1
387	001524	033130	DT2	
388	001526	031646	EM2	
389	001530	032614	DH2	; ERROR 2
390	001532	033130	DT2	
391	001534	031711	EM3	
392	001536	032614	DH2	; ERROR 3
393	001540	033130	DT2	
394	001542	031755	EM4	
395	001544	000000	0	; ERROR 4
396	001546	000000	0	
397	001550	032017	EM5	
398	001552	032614	DH2	; ERROR 5
399	001554	033130	DT2	
400	001556	032017	EM5	
401	001560	032652	DH3	; ERROR 6
402	001562	033146	DT3	
403	001564	032047	EM6	
404	001566	032573	DH1	; ERROR 7
405	001570	033116	DT1	
406	001572	032066	EM7	
407	001574	032573	DH1	; ERROR 10
408	001576	033116	DT1	
409	001600	032113	EM10	
410	001602	032573	DH1	; ERROR 11
411	001604	033116	DT1	
412	001606	032137	EM11	
413	001610	032750	DH5	; ERROR 12
414	001612	033172	DT5	
415	001614	032166	EM12	
416	001616	032750	DH5	; ERROR 13
417	001620	033172	DT5	
418	001622	032137	EM11	
419	001624	032710	DH4	; ERROR 14
420	001626	033160	DT4	

F03

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 10
 DZKCE.P11 12-MAY-77 12:23 ERROR POINTER TABLE

PAGE: 0031

421	001630	032212	EM13	
422	001632	000000	0	; ERROR 15
423	001634	000000	0	
424	001636	032137	EM11	
425	001640	032750	DM5	; ERROR 16
426	001642	033210	DT6	
427	001644	032166	EM12	
428	001646	032750	DM5	; ERROR 17
429	001650	033210	DT6	
430	001652	032137	EM11	
431	001654	033002	DM6	; ERROR 20
432	001656	033226	DT7	
433	001660	032137	EM11	
434	001662	033002	DM6	; ERROR 21
435	001664	033250	DT10	
436	001666	032166	EM12	
437	001670	033002	DM6	; ERROR 22
438	001672	033226	DT7	
439	001674	032166	EM12	
440	001676	033002	DM6	; ERROR 23
441	001700	033250	DT10	
442	001702	032252	EM14	
443	001704	000000	0	; ERROR 24
444	001706	000000	0	
445	001710	032322	EM15	
446	001712	032573	DM1	; ERROR 25
447	001714	033116	DT1	
448	001716	032343	EM16	
449	001720	032652	DM3	; ERROR 16
450	001722	033272	DT11	
451	001724	032166	EM12	
452	001726	032573	DM1	; ERROR 27
453	001730	033304	DT12	
454	001732	032357	EM17	
455	001734	000000	0	; ERROR 30
456	001736	000000	0	
457	001740	032423	EM20	
458	001742	032573	DM1	; ERROR 31
459	001744	033116	DT1	
460	001746	032444	EM21	
461	001750	033050	DM7	; ERROR 32
462	001752	000000	0	
463	001754	032444	EM21	
464	001756	032652	DM3	; ERROR 33
465	001760	033146	DT3	
466	001762	032461	EM22	
467	001764	033073	DM10	; ERROR 34
468	001766	000000	0	
469	001770	032504	EM23	
470	001772	032614	DM2	; ERROR 35
471	001774	033130	DT2	
472	001776	032526	EM24	
473	002000	000000	0	; ERROR 36
474	002002	000000	0	
475	002004	032551	EM25	
476	002006	000000	0	; ERROR 37

G03

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 11
 DZKCE.P11 12-MAY-77 12:23 ERROR POINTER TABLE

PAGE: 0032

477	002010	000000	0	
478	002012	032047	EM6	
479	002014	032614	DM2	; ERROR 40
480	002016	033130	DT2	
481	002020	032017	EM5	
482	002022	032750	DM5	; ERROR 41
483	002024	033172	DT5	
484	002026	032212	EM13	
485	002030	032573	DM1	; ERROR 42
486	002032	033116	DT1	
487		002034		
488				
489				
490				
491				
492				
493		002034		
494		000024		
495	000024	000200		
496		000044		
497	000044	002034		
498		002034		
499				
500				
501				
502				
503	002034			
504	002034	000000		
505	002036	001316		
506	002040	000132		
507	002042	000137		
508	002044	000137		
509	002046	000052		
510				

. = 2034
 .SBTTL APT PARAMETER BLOCK
 ;*****
 ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
 ;*****
 . \$X = . ; SAVE CURRENT LOCATION
 . = 24 ; SET POWER FAIL TO POINT TO START OF PROGRAM
 200 ; FOR APT START UP
 . = 44 ; POINT TO APT INDIRECT ADDRESS PNTR.
 \$APTHDR ; POINT TO APT HEADER BLOCK
 . = \$X ; RESET LOCATION COUNTER
 ;*****
 ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
 ;INTERFACE SPEC.
 \$APTHD:
 \$HIBTS: .WORD 0 ; TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
 \$MADR: .WORD \$MAIL ; ADDRESS OF APT MAILBOX (BITS 0-15)
 \$TSTM: .WORD 90. ; RUN TIM OF LONGEST TEST
 \$PASTM: .WORD 95. ; RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
 \$UNITM: .WORD 95. ; ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
 .WORD \$ETEND-\$MAIL/2 ; LENGTH MAILBOX-ETABLE(WORDS)

H03

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 12
 DZKCE.P11 12-MAY-77 12:23 APT PARAMETER BLOCK

PAGE: 0033

```

511
512 ;KMC11 CONTROL INDICATORS FOR CURRENT KMC11 UNDER TEST
513 ;-----
514
515 002050 000000 STAT1: 0
516 002052 000000 STAT2: 0
517 002054 000000 STAT3: 0
518
519 ;KMC11 VECTOR AND REGISTER INDIRECT POINTERS
520 ;-----
521
522 002056 000000 KMRVEC: 0 ; POINTER TO KMC11 RECEIVER INTERRUPT VECTOR
523 002060 000000 KMRVLV: 0 ; POINTER TO KMC11 RECEIVER INTERRUPT SERVICE PS
524 002062 000000 KMTVEC: 0 ; POINTER TO KMC11 TRANSMITTER INTERRUPT VECTOR
525 002064 000000 KMTVLV: 0 ; POINTER TO KMC11 TRANSMITTER INTERRUPT SERVICE PS
526 002066 000000 KMCSR: 0 ; POINTER TO KMC11 CONTROL STATUS REGISTER
527 002070 000000 KMCSRH: 0 ; POINTER TO KMC11 CONTROL STATUS REGISTER HIGH BYTE.
528 002072 000000 KMCTL: 0 ; POINTER TO KMC11 CONTROL OUT REGISTER
529 002074 000000 KMP04: 0 ; POINTER TO KMC11 PORT REGISTER(SEL 4)
530 002076 000000 KMP06: 0 ; POINTER TO KMC11 PORT REGISTER(SEL 6)
531
532 ;TEMP STORAGE
533 ;-----
534
535 ;TEMP: 0
536 ;.=.+40
537
538 ;KMC11 STATUS TABLE AND ADDRESS ASSIGNMENTS
539 ;-----
540
541 . =2100
542 002100 002100 KM.MAP:
543 002100 000001 KMC00: .BLKW 1 ; CONTROL STATUS REGISTER FOR KMC11 NUMBER 00
544 002102 000001 KMS100: .BLKW 1 ; VECTOR FOR KMC11 NUMBER 00
545 002104 000001 KMS200: .BLKW 1 ; DDCMP LINE# FOR KMC11 NUMBER 00
546 002106 000001 KMS300: .BLKW 1 ; 3RD STATUS WORD
547
548 002110 000001 KMC01: .BLKW 1 ; CONTROL STATUS REGISTER FOR KMC11 NUMBER 01
549 002112 000001 KMS101: .BLKW 1 ; VECTOR FOR KMC11 NUMBER 01
550 002114 000001 KMS201: .BLKW 1 ; DDCMP LINE# FOR KMC11 NUMBER 01
551 002116 000001 KMS301: .BLKW 1 ; 3RD STATUS WORD
552
553 002120 000001 KMC02: .BLKW 1 ; CONTROL STATUS REGISTER FOR KMC11 NUMBER 02
554 002122 000001 KMS102: .BLKW 1 ; VECTOR FOR KMC11 NUMBER 02
555 002124 000001 KMS202: .BLKW 1 ; DDCMP LINE# FOR KMC11 NUMBER 02
556 002126 000001 KMS302: .BLKW 1 ; 3RD STATUS WORD
557
558 002130 000001 KMC03: .BLKW 1 ; CONTROL STATUS REGISTER FOR KMC11 NUMBER 03
559 002132 000001 KMS103: .BLKW 1 ; VECTOR FOR KMC11 NUMBER 03
560 002134 000001 KMS203: .BLKW 1 ; DDCMP LINE# FOR KMC11 NUMBER 03
561 002136 000001 KMS303: .BLKW 1 ; 3RD STATUS WORD
562
563 002140 000001 KMC04: .BLKW 1 ; CONTROL STATUS REGISTER FOR KMC11 NUMBER 04
564 002142 000001 KMS104: .BLKW 1 ; VECTOR FOR KMC11 NUMBER 04
565 002144 000001 KMS204: .BLKW 1 ; DDCMP LINE# FOR KMC11 NUMBER 04
566 002146 000001 KMS304: .BLKW 1 ; 3RD STATUS WORD

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 13
 DZKCE.P11 12-MAY-77 12:23 APT PARAMETER BLOCK

PAGE: 0034

567					
568	002150	000001	KMCRO5:	.BLKW	1
569	002152	000001	KMS105:	.BLKW	1
570	002154	000001	KMS205:	.BLKW	1
571	002156	000001	KMS305:	.BLKW	1
572					
573	002160	000001	KMCRO6:	.BLKW	1
574	002162	000001	KMS106:	.BLKW	1
575	002164	000001	KMS206:	.BLKW	1
576	002166	000001	KMS306:	.BLKW	1
577					
578	002170	000001	KMCRO7:	.BLKW	1
579	002172	000001	KMS107:	.BLKW	1
580	002174	000001	KMS207:	.BLKW	1
581	002176	000001	KMS307:	.BLKW	1
582					
583	002200	000001	KMCRI0:	.BLKW	1
584	002202	000001	KMS110:	.BLKW	1
585	002204	000001	KMS210:	.BLKW	1
586	002206	000001	KMS310:	.BLKW	1
587					
588	002210	000001	KMCRI1:	.BLKW	1
589	002212	000001	KMS111:	.BLKW	1
590	002214	000001	KMS211:	.BLKW	1
591	002216	000001	KMS311:	.BLKW	1
592					
593	002220	000001	KMCRI2:	.BLKW	1
594	002222	000001	KMS112:	.BLKW	1
595	002224	000001	KMS212:	.BLKW	1
596	002226	000001	KMS312:	.BLKW	1
597					
598	002230	000001	KMCRI3:	.BLKW	1
599	002232	000001	KMS113:	.BLKW	1
600	002234	000001	KMS213:	.BLKW	1
601	002236	000001	KMS313:	.BLKW	1
602					
603	002240	000001	KMCRI4:	.BLKW	1
604	002242	000001	KMS114:	.BLKW	1
605	002244	000001	KMS214:	.BLKW	1
606	002246	000001	KMS314:	.BLKW	1
607					
608	002250	000001	KMCRI5:	.BLKW	1
609	002252	000001	KMS115:	.BLKW	1
610	002254	000001	KMS215:	.BLKW	1
611	002256	000001	KMS315:	.BLKW	1
612					
613	002260	000001	KMCRI6:	.BLKW	1
614	002262	000001	KMS116:	.BLKW	1
615	002264	000001	KMS216:	.BLKW	1
616	002266	000001	KMS316:	.BLKW	1
617					
618	002270	000001	KMCRI7:	.BLKW	1
619	002272	000001	KMS117:	.BLKW	1
620	002274	000001	KMS217:	.BLKW	1
621	002276	000001	KMS317:	.BLKW	1
622					

;CONTROL STATUS REGISTER FOR KMC11 NUMBER 05
 ;VECTOR FOR KMC11 NUMBER 05
 ;DDCMP LINE# FOR KMC11 NUMBER 05
 ;3RD STATUS WORD

;CONTROL STATUS REGISTER FOR KMC11 NUMBER 06
 ;VECTOR FOR KMC11 NUMBER 06
 ;DDCMP LINE# FOR KMC11 NUMBER 06
 ;3RD STATUS WORD

;CONTROL STATUS REGISTER FOR KMC11 NUMBER 07
 ;VECTOR FOR KMC11 NUMBER 07
 ;DDCMP LINE# FOR KMC11 NUMBER 07
 ;3RD STATUS WORD

;CONTROL STATUS REGISTER FOR KMC11 NUMBER 10
 ;VECTOR FOR KMC11 NUMBER 10
 ;DDCMP LINE# FOR KMC11 NUMBER 10
 ;3RD STATUS WORD

;CONTROL STATUS REGISTER FOR KMC11 NUMBER 11
 ;VECTOR FOR KMC11 NUMBER 11
 ;DDCMP LINE# FOR KMC11 NUMBER 11
 ;3RD STATUS WORD

;CONTROL STATUS REGISTER FOR KMC11 NUMBER 12
 ;VECTOR FOR KMC11 NUMBER 12
 ;DDCMP LINE# FOR KMC11 NUMBER 12
 ;3RD STATUS WORD

;CONTROL STATUS REGISTER FOR KMC11 NUMBER 13
 ;VECTOR FOR KMC11 NUMBER 13
 ;DDCMP LINE# FOR KMC11 NUMBER 13
 ;3RD STATUS WORD

;CONTROL STATUS REGISTER FOR KMC11 NUMBER 14
 ;VECTOR FOR KMC11 NUMBER 14
 ;DDCMP LINE# FOR KMC11 NUMBER 14
 ;3RD STATUS WORD

;CONTROL STATUS REGISTER FOR KMC11 NUMBER 15
 ;VECTOR FOR KMC11 NUMBER 15
 ;DDCMP LINE# FOR KMC11 NUMBER 15
 ;3RD STATUS WORD

;CONTROL STATUS REGISTER FOR KMC11 NUMBER 16
 ;VECTOR FOR KMC11 NUMBER 16
 ;DDCMP LINE# FOR KMC11 NUMBER 16
 ;3RD STATUS WORD

;CONTROL STATUS REGISTER FOR KMC11 NUMBER 17
 ;VECTOR FOR KMC11 NUMBER 17
 ;DDCMP LINE# FOR KMC11 NUMBER 17
 ;3RD STATUS WORD

J03

PAGE: 0035

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 14
DZKCE.P11 12-MAY-77 12:23 APT PARAMETER BLOCK
623 002300 000000 KM.END: 000000

K03

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 15
 DZKCE.P11 12-MAY-77 12:23 APT PARAMETER BLOCK

PAGE: 0036

624				
625			; KMC11 PASS COUNT AND ERROR COUNT TABLE	
626			-----	
627				
628	002302		CNT MAP:	
629	002302	000000	PACT00: 0	; PASS COUNT FOR KMC11 NUMBER 00
630	002304	000000	ERCT00: 0	; ERROR COUNT FOR KMC11 NUMBER 00
631				
632	002306	000000	PACT01: 0	; PASS COUNT FOR KMC11 NUMBER 01
633	002310	000000	ERCT01: 0	; ERROR COUNT FOR KMC11 NUMBER 01
634				
635	002312	000000	PACT02: 0	; PASS COUNT FOR KMC11 NUMBER 02
636	002314	000000	ERCT02: 0	; ERROR COUNT FOR KMC11 NUMBER 02
637				
638	002316	000000	PACT03: 0	; PASS COUNT FOR KMC11 NUMBER 03
639	002320	000000	ERCT03: 0	; ERROR COUNT FOR KMC11 NUMBER 03
640				
641	002322	000000	PACT04: 0	; PASS COUNT FOR KMC11 NUMBER 04
642	002324	000000	ERCT04: 0	; ERROR COUNT FOR KMC11 NUMBER 04
643				
644	002326	000000	PACT05: 0	; PASS COUNT FOR KMC11 NUMBER 05
645	002330	000000	ERCT05: 0	; ERROR COUNT FOR KMC11 NUMBER 05
646				
647	002332	000000	PACT06: 0	; PASS COUNT FOR KMC11 NUMBER 06
648	002334	000000	ERCT06: 0	; ERROR COUNT FOR KMC11 NUMBER 06
649				
650	002336	000000	PACT07: 0	; PASS COUNT FOR KMC11 NUMBER 07
651	002340	000000	ERCT07: 0	; ERROR COUNT FOR KMC11 NUMBER 07
652				
653	002342	000000	PACT10: 0	; PASS COUNT FOR KMC11 NUMBER 10
654	002344	000000	ERCT10: 0	; ERROR COUNT FOR KMC11 NUMBER 10
655				
656	002346	000000	PACT11: 0	; PASS COUNT FOR KMC11 NUMBER 11
657	002350	000000	ERCT11: 0	; ERROR COUNT FOR KMC11 NUMBER 11
658				
659	002352	000000	PACT12: 0	; PASS COUNT FOR KMC11 NUMBER 12
660	002354	000000	ERCT12: 0	; ERROR COUNT FOR KMC11 NUMBER 12
661				
662	002356	000000	PACT13: 0	; PASS COUNT FOR KMC11 NUMBER 13
663	002360	000000	ERCT13: 0	; ERROR COUNT FOR KMC11 NUMBER 13
664				
665	002362	000000	PACT14: 0	; PASS COUNT FOR KMC11 NUMBER 14
666	002364	000000	ERCT14: 0	; ERROR COUNT FOR KMC11 NUMBER 14
667				
668	002366	000000	PACT15: 0	; PASS COUNT FOR KMC11 NUMBER 15
669	002370	000000	ERCT15: 0	; ERROR COUNT FOR KMC11 NUMBER 15
670				
671	002372	000000	PACT16: 0	; PASS COUNT FOR KMC11 NUMBER 16
672	002374	000000	ERCT16: 0	; ERROR COUNT FOR KMC11 NUMBER 16
673				
674	002376	000000	PACT17: 0	; PASS COUNT FOR KMC11 NUMBER 17
675	002400	000000	ERCT17: 0	; ERROR COUNT FOR KMC11 NUMBER 17
676				

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 16
 DZKCE.P11 12-MAY-77 12:23 APP PARAMETER BLOCK

PAGE: 0037

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	
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	CSR
I	C	O	N	T	R	O	L	R	E	G	I	S	T	E	R	
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	
I	*	I	*	I	*	I	*	I	*	I	*	I	V	E	C	STAT1
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	
I	*	I	B	M	I	A	D	D	*	I	*	I	L	I	N	STAT2
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	STAT3
I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	I	

DEFINITION OF FORMAT

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 (DOCMP 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])

```

731
732 ;PROGRAM INITIALIZATION
733 ;LOCK OUT INTERRUPTS
734 ;SET UP PROCESSOR STACK
735 ;SET UP POWER FAIL VECTOR
736 ;CLEAR PROGRAM CONTROL FLAGS AND COUNTS
737 ;TYPE TITLE MESSAGE
738
739 002402 012737 000340 177776 .START: MOV #340,PS ;LOCK OUT INTERRUPTS
740 002410 012706 001200 MOV #STACK,SP ;SET UP STACK
741 002414 012737 007126 000024 MOV #SPWRD,2#24 ;SET UP POWER FAIL VECTOR
742 002422 013737 001472 001476 MOV KMMUM,SAVNUM ;SAVE NUMBER OF DEVICES IN SYSTEM.
743 002430 005037 011415 CLR SWFLG ;CLEAR SOFT TYPEOUT FLAG
744 002434 105037 001203 CLR8 SERFLG ;CLEAR ERROR FLAG
745 002440 105037 001511 CLR8 QV.FLG ;ZERO QUICK VERIFY FLAG
746 002444 012737 002070 001502 MOV #KM.MAP-10,CREAM ;GET MAP POINTER.
747 002452 012737 002276 001504 MOV #CNT.MAP-4,MILK ;GET PASS COUNT MAP POINTER
748 002460 012737 100000 001500 MOV #BIT15,RUN ;POINT POINTER TO FIRST DEVICE.
749 002466 012700 002302 MOV #CNT.MAP,RO ;PASS COUNT POINTER TO RO
750 002472 005020 23$: CLR (RO)+ ;CLEAR TABLE
751 002474 022700 002402 CMP #CNT.MAP+100,RO ;DONE YET?
752 002500 001374 BNE 23$ ;KEEP GOING
753 002502 005037 001216 CLR $ERRPC ;CLEAR LAST ERROR POINTER
754 002506 012737 000001 001202 MOV #1,$STSTM ;SET UP FOR TEST 1
755 002514 012737 002402 001206 MOV #.START,$LPAOR ;SET UP FOR POWER FAIL BEFORE
756 ;TESTING STARTS
757 002522 132737 000001 001336 BITB #1,$ENV ;IS IT RUNNING UNDER APT?
758 002530 001404 BEQ 3$ ;IF NOT CHECK FOR TYPE OF SWITCH REGISTER.
759 002532 013737 001340 000176 MOV $SWREG,SWREG ;LOAD SOFTWARE SWITCH REG.
760 002540 000123 BR 6$+2 ;GO SET UP SOFTWARE SWITCH REG.
761 002542 013746 000006 3$: MOV 2#6,-(SP) ;SAVE CURRENT VECTORS
762 002546 013746 000004 MOV 2#4,-(SP)
763 002552 012737 002606 000004 MOV #6$,2#4 ;SET UP FOR TIMEOUT
764 002560 012737 177570 001240 MOV #177570,SWR ;SET SWR TO HARD SWR ADDRESS
765 002566 012737 177570 001242 MOV #177570,DISPLAY ;SET DISPLAY TO HARD SWR ADDRESS
766 002574 022777 177777 176436 CMP #-1,2SWR ;REFERENCE HARDWARE SWITCH REGISTER
767 002602 001402 BEQ 6$+2 ;IF = -1 USE SOFT SWR ANYWAY
768 002604 000407 BR 7$ ;IF IT EXISTS AND NOT = -1 USE HARD SWR
769 002606 022626 6$: CMP (SP)+(SP)+ ;ADJUST STACK
770 002610 012737 000176 001240 MOV #SWREG,SWR ;POINTER TO SOFT SWR
771 002616 012737 000174 001242 MOV #DISPREG,DISPLAY ;POINTER TO SOFT DISPLAY REG
772 002624 012637 000004 7$: MOV (SP)+,2#4 ;RESTORE VECTORS
773 002630 012637 000006 MOV (SP)+,2#6
774 002634 105737 001506 TSTB INIFLG ;HAS INITIALIZATION BEEN PERFORMED
775 002640 001006 BNE 20$ ;BR IF YES
776 002642 022737 004070 000042 CMP #SENDAD,2#42 ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
777 002650 001402 BEQ 20$
778 002652 104401 001000 TYPE #TITLE ;TYPE TITLE MESSAGE
779 002656 004737 011212 20$: JSR PC,CKSWR ;CHECK FOR SOFT SWR
780 002662 017737 176352 001446 MOV 2SWR,STRTSW ;STORE STARTING SWITCHES
781 002670 005737 000042 TST 2#42 ;IS IT RUNNING IN AUTO MODE?
782 002674 001402 BEQ .+6 ;BR IF NO
783 002676 005037 001446 CLR STRTSW ;IF YES, CLEAR SWITCHES
784 002702 032737 000001 001446 BIT #SW00,STRTSW ;IF SW00=1, QUESTIONS ARE ASKED.
785 002710 001012 BNE 17$ ;BR IF SW00=1
786 002712 105737 001446 TSTB STRTSW ;BIT7=1??

```

N03

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 18
 DZKCE.P11 12-MAY-77 12:23 PROGRAM INITIALIZATION AND START UP.

PAGE: 0039

787	002716	100007		BPL	17\$		;BR IF SW07=0
788	002720	005737	001470	TST	KMACTV		;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	\$ENVH		;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							
827							
828							
829							
830							
831							
832							
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	AUSTR: 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

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 19
 DZKCE.P11 12-MAY 77 12:23

PROGRAM INITIALIZATION AND START UP.

PAGE: 0040

843	003146	012702	003342	4\$:	MOV	#DEVTAB,R2	:R2 IS DEVICE TABLE POINTER
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,2#4	:OK ONLY KMC'S ARE LEFT, SET UP FOR TIMEOUT
853	003202	005711		FY:	TST	(R1)	:CHECK KMC ADDRESS
854	003204	020137	002066		CMF	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	AUDONE	:BRANCH IF ALL DONE.
862	003236	000761			BR	FY	:DO IT AGAIN.
863	003240	122243		MODEV:	CMFB	(R2)+,-(R3)	:ON TIMEOUT, INC R2, DEC R3
864	003242	000002			RTI		:SLPDR
865	003244	005737	001302	ERR:	TST	\$TMP2	:CHECK FLAG IF = 0 TYPE HEADER
866	003250	001014			BNE	IS	: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			CONVRT		:TYPE OUT ERROR PC
871	003266	003322			ERRPC		
872	003270	104401			TYPE		:TYPE REST OF HEADER
873	003272	011027			CONERR		
874	003274	012737	177777 001302	IS:	MOV	#-1,\$TMP2	:SET FLAG SO IT ONLY GETS TYPED ONCE
875	003302	010137	001264		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\$:	CMF	(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			.PYTE	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$: MOV      (SP)+,2#6      ;RESTORE LOC 6
902      003360      012637 000004      MOV      (SP)+,2#4      ;RESTORE LOC 4
903      003364      032737 000010      001446  BIT      #SW03,STATSW ;SELECT SPECIFIC DEVICES??
904      003372      001422      BEQ      3$      ;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 NUM. A 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 EVERYTHING.
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      2#6,-(SP)      ;SAVE LOC 6
929      003474      013746 000004      MOV      2#4,-(SP)      ;SAVE LOC 4
930      003500      005000      CLR      R0      ;START AT 0
931      003502      012737 003546      000004  MOV      #25,2#4      ;SET UP FOR TIME OUT
932      003510      005037 000006      CLR      2#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$: MOV      R0,MEALIM      ;STORE MEMORY LIMIT
938      003534      012637 000004      MOV      (SP)+,2#4      ;RESTORE LOC 4
939      003540      012637 000006      MOV      (SP)+,2#6      ;RESTORE LOC 6
940      003544      000413      BR       10$      ;CONTINUE
941      003546      022626      2$: CMP      (SP)+,(SP)+      ;ADJUST STACK
942      003550      162700 000004      SUB      #4,R0      ;GET LAST GOOD ADDRESS
943      003554      162700 007776      SUB      #7776,R0      ;SAVE 2K FOR MONITORS
944      003560      022700 030000      CMP      #30000,R0      ;IS IT BK?
945      003564      001361      BNE      7$      ;BR IF NO
946      003566      012700 037400      MOV      #37400,R0      ;IF BK DON'T SAVE 2K
947      003572      000756      BR       7$      ;
948      003574      012737 000340      10$: MOV      #340,PS      ;LOCK OUT INTERRUPTS
949      003602      032737 000004      001446  BIT      #BIT2,STATSW ;CHECK FOR LOCK ON TEST
950      003610      001406      BEQ      1$      ;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

```

D04

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 21

PAGE: 0042

DZKCE.P11 12-MAY-77 12:23 PROGRAM INITIALIZATION AND START UP.

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,STATSW	; IS TEST NO. SELECTED?
957	003650	001002				BNE	5\$	; BR IF YES
958	003652	107401	007642			TYPE	MR	; TYPE R
959	003656	000177	175324		5\$:	JMP	\$LPADR	; START TESTING

```

960      ;END OF PASS
961      ;TYPE NAME OF TEST
962      ;UPDATE PASS COUNT
963      ;CHECK FOR EXIT TO ACT-11
964      ;RESTART TEST
965
966      .SBTTL  END OF PASS ROUTINE
967
968      ;*****
969      ;INCREMENT THE PASS NUMBER ($PASS)
970      ;*IF THERES A MONITOR GO TO IT
971      ;*IF THERE ISN'T JUMP TO CYCLE
972
973      $EOP:
974      RESET
975      INC      $PASS
976      CLRB    $ERFLG
977      TYPE    ,MEPASS
978      TYPE    ,MCSR
979      CNVRT   ,XCSR
980      TYPE    ,MVECX
981      CNVRT   ,XVEC
982      TYPE    ,MPASSX
983      CNVRT   ,XPASS
984      TYPE    ,MERRX
985      CNVRT   ,XERR
986      MOV     $MILK,RO
987      MOV     $PASS,(RO)+
988      MOV     $ERTTL,(RO)+
989      MOV     $KMRVL,$KMRVEC
990      CLR     $KMRVL
991      MOV     $KMTVL,$KMTVEC
992      CLR     $KMTVL
993      DEC     $AVNUM
994      BNE     $DOAGN
995      MOV     $377,$QV.FLG
996      MOV     $KMIN,$AVNUM
997      CLR     $ERRPC
998      CLR     $TIMES
999      INC     $PASS
1000     BIC     $100000,$PASS
1001     DEC     (PC)+
1002     $EOPCT: .WORD 1
1003     BGT     $DOAGN
1004     MOV     (PC)+,$2(PC)+
1005     $ENDCT: .WORD 1
1006     $EOPCT
1007     $GET42: MOV     $42,RO
1008     BEQ     $DOAGN
1009     RESET
1010     $END40: JSR     PC,(RO)
1011     NOP
1012     NOP
1013     NOP
1014     $DOAGN:
1015     JMP     $2(PC)+

```

INCREMENT THE PASS COUNT
 CLEAR ERROR FLAG
 TYPE END PASS.
 TYPE "CSR"
 SHOW IT.
 TYPE VECTOR.
 SHOW IT.
 TYPE "PASSES"
 SHOW IT.
 TYPE "ERRORS"
 SHOW IT.
 SET POINTER TO PASSCNT.
 SAVE THE PASS COUNT.
 SAVE ERROR COUNT
 RESTORE THE RECEIVER INTERRUPT VECTOR.
 RESTORE RECEIVER LEVEL
 RESTORE THE TRANSMIT INTERRUPT VECTOR.
 RESTORE TRANSMITTER LEVEL
 ALL DEVICE TESTED?
 BRANCH IF NO.
 SET QUICK VERIFY FLAG.
 RESTORE DEVICE COUNT.
 CLEAR LAST ERROR PC
 ZERO THE NUMBER OF ITERATIONS
 INCREMENT THE PASS NUMBER
 DON'T ALLOW A NEG. NUMBER
 LOOP?

YES
 RESTORE COUNTER

GET MONITOR ADDRESS
 BRANCH IF NO MONITOR
 CLEAR THE WORLD
 GO TO MONITOR
 SAVE ROOM
 FOR
 ACT11

RETURN

```

SRTNAD: .WORD      CYCLE
XCSR:   1
        .BYTE      6,2
        KMCSR
XVEC:   1
        .BYTE      4,2
        KMVEC
XPASS:  1
        .BYTE      6,2
        $PASS
XERR:   1
        .BYTE      6,2
        $ERTTL

```

```

;SCOPE LOOP AND INTERATION HANDLER
;-----

```

```

.SBTL SCOPE HANDLER ROUTINE

```

```

:*****
:THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
:AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
:AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
:THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
:SW14=1          LOOP ON TEST
:SW11=1          INHIBIT ITERATIONS
:CALL
:*              SCOPE          ;;SCOPE=IOT

```

```

$SCOPE: CLR          $ERRPC          ; CLEAR LAST ERROR PC
        CMP          TST1+2,(SP)    ; IS THIS TEST #1 ?
        BEQ          $XTSTR         ; IF SO DON'T LOOP.
TTST:   BR           1$              ;
        TSTB         $STKS          ; KEYBOARD DONE ?
        BPL          $OVER          ; IF NO DONT WAIT.
        MOV          $STKB,-2(SP)   ;
1$:     BIT           $BIT14,$SWR    ; LOOP ON PRESENT TEST?
        BNE          $OVER          ; YES IF SW14=1
;*****START OF CODE FOR THE XOR TESTER*****
$XTSTR: BR           6$              ; IF RUNNING ON THE "XOR" TESTER CHANGE
; THIS INSTRUCTION TO A "NOP" (NOP=240)
        MOV          $ERRVEC,-(SP)  ; SAVE THE CONTENTS OF THE ERROR VECTOR
        MOV          $5,$ERRVEC    ; SET FOR TIMEOUT
        TST          $177060       ; TIME OUT ON XOR?
        MOV          (SP)+,$ERRVEC ; RESTORE THE ERROR VECTOR
        BR           $SVLAD        ; GO TO THE NEXT TEST
5$:     CMP           (SP)+,(SP)+   ; CLEAR THE STACK AFTER A TIME OUT
        MOV          (SP)+,$ERRVEC ; RESTORE THE ERROR VECTOR
        $OVER        ; LOOP ON THE PRESENT TEST
6$:;*****END OF CODE FOR THE XOR TESTER*****
2$:     TSTB         $ERFLG        ; HAS AN ERROR OCCURRED?
        BEQ          3$            ; BR IF NO
4$:     CLRB         $ERFLG        ; ZERO THE ERROR FLAG
        CLR          $TIMES        ; CLEAR THE NUMBER OF ITERATIONS TO MAKE
3$:     BIT           $BIT11,$SWR  ; INHIBIT ITERATIONS?

```

```

1072 004256 001011      BNE      IS          ;; BR IF YES
1073 004260 005737 001324  TST      $PASS          ;; IF FIRST PASS OF PROGRAM
1074 004264 001406      BEQ      IS          ;; 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  MOV      $TSTNM,$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      $LPADR,(SP)          ;; FUDGE RETURN ADDRESS
1085 004346 005037 001444      CLR      LOCK          ;; RESET LOCK ON DATA.
1086 004352 013701 002066      MOV      KMC$R,R1          ;; R1 CONTAINS BASE KMC ADDRESS.
1087 004356 000002      RTI
1088 004360 000406      BRW: .WORD      406
1089 004362 000020      $MXCNT: 20          ;; MAX. NUMBER OF ITERATIONS
1090
1091      ;CHECK FOR FREEZE ON CURRENT DATA
1092      ;-----
1093
1094 004364 004737 011212      .SCOPI: JSR      PC,CK$WR          ;;CHECK FOR SOFT SWR
1095 004370 032777 001000 174642  BIT      #SW09,$SWR          ;; IS SW09=1(SET)?
1096 004376 001405      BEQ      IS          ;; BR IF NOT SET.
1097 004400 005737 001444      TST      LOCK
1098 004404 001402      BEQ      IS
1099 004406 013716 001444      MOV      LOCK,(SP)          ;; GOTO THE ADDRESS IN LOCK.
1100 004412 000002      1$: 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      $TPFLG          ;; IS THERE A TERMINAL?
1123 004420 100002      BPL      IS          ;; BR IF YES
1124 004422 000000      HALT          ;; HALT HERE IF NO TERMINAL
1125 004424 000430      BR      3$          ;; 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,SENV ;: RUNNING IN APT MODE
1129 004442 001011 BNE 62$ ;: NO GO CHECK FOR APT .ONSOLE
1130 004444 132737 000100 001337 BITB #APTSPool,SENV ;: SPOOL MESSAGE TO APT
1131 004452 001405 BEQ 62$ ;: NO GO CHECK FOR CONSOLE
1132 004454 010037 004464 MOV RO,61$ ;: SETUP MESSAGE ADDRESS FOR APT
1133 004460 004737 004704 JSR PC,$ATY3 ;: SPOOL MESSAGE TO APT
1134 004464 000000 .WORD 0 ;: MESSAGE ADDRESS
1135 004466 132737 000040 001337 61$: BITB #APTC SUP,SENV ;: APT CONSOLE SUPPRESSED
1136 004474 001003 BNE 60$ ;: YES, SKIP TYPE OUT
1137 004476 112046 2$: MOVB (RO)+,-(SP) ;: PUSH CHARACTER TO BE TYPED ONTO STACK
1138 004500 001005 BNE 4$ ;: BR IF IT ISN'T THE TERMINATOR
1139 004502 005726 TST (SP)+ ;: IF TERMINATOR POP IT OFF THE STACK
1140 004504 012600 60$: MOV (SP)+,RO ;: RESTORE RO
1141 004506 062716 000002 3$: 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 $STPS ;: WAIT UNTIL PRINTER IS READY
1172 004632 100375 BPL $TYPEC ;:
1173 004634 116677 000002 174410 MOVB 2(SP),2$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

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 26
 DZKCE.P11 12-MAY-77 12:23 APT COMMUNICATIONS ROUTINE

PAGE: 0047

```

1184 .SBTTL APT COMMUNICATIONS ROUTINE
1185
1186 *****
1187 004676 112737 000001 005142 SATY1: MOVB #1,$FFLG ;;TO REPORT FATAL ERROR
1188 004704 112737 000001 005140 SATY3: MOVB #1,$MFLG ;;TO TYPE A MESSAGE
1189 004712 000403 BR SATYC
1190 004714 112737 000001 005142 SATY4: MOVB #1,$FFLG ;;TO ONLY REPORT FATAL ERROR
1191 004722 SATYC:
1192 004722 010046 MOV RO,-(SP) ;;PUSH RO ON STACK
1193 004724 010146 MOV R1,-(SP) ;;PUSH R1 ON STACK
1194 004726 105737 005140 TSTB $MFLG ;;SHOULD TYPE A MESSAGE?
1195 004732 001450 BEQ 5$ ;;IF NOT: BR
1196 004734 122737 000001 001336 CMPB $APTENV,$ENV ;;OPERATING UNDER APT?
1197 004742 001031 BNE 3$ ;;IF NOT: BR
1198 004744 132737 000100 001337 BITB $APTSPool,$ENVM ;;SHOULD SPOOL MESSAGES?
1199 004752 001425 BEQ 3$ ;;IF NOT: BR
1200 004754 017600 000004 MOV #4(SP),RO ;;GET MESSAGE ADDR.
1201 004760 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
1202 004766 005737 001316 1$: TST $MSGTYPE ;;SEE IF DONE W/ LAST XMISSION?
1203 004772 001375 BNE 1$ ;;IF NOT: WAIT
1204 004774 010037 001332 MOV RO,$MSGAD ;;PUT ADDR IN MAILBOX
1205 005000 105720 2$: TSTB (RO)+ ;;FIND END OF MESSAGE
1206 005002 001376 BNE 2$
1207 005004 163700 001332 SUB $MSGAD,RO ;;SUB START OF MESSAGE
1208 005010 006200 ASR RO ;;GET MESSAGE LGTH IN WORDS
1209 005012 010037 001334 MOV RO,$MSGLGTH ;;PUT LENGTH IN MAILBOX
1210 005016 012737 000004 001316 MOV #4,$MSGTYPE ;;TELL APT TO TAKE MSG.
1211 005024 000413 BR 5$
1212 005026 017637 000004 005052 3$: MOV #4(SP),4$ ;;PUT MSG ADDR IN JSR LINKAGE
1213 005034 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDRESS
1214 005042 013746 177776 MOV 177776,-(SP) ;;PUSH 177776 ON STACK
1215 005046 004737 004414 JSR PC,$TYPE ;;CALL TYPE MACRO
1216 005052 000000 4$: .WORD 0
1217 005054 5$:
1218 005054 105737 005142 10$: TSTB $FFLG ;;SHOULD REPORT FATAL ERROR?
1219 005060 001416 BEQ 12$ ;;IF NOT: BR
1220 005062 005737 001336 TST $ENV ;;RUNNING UNDER APT?
1221 005066 001413 BEQ 12$ ;;IF NOT: BR
1222 005070 005737 001316 11$: TST $MSGTYPE ;;FINISHED LAST MESSAGE?
1223 005074 001375 BNE 11$ ;;IF NOT: WAIT
1224 005076 017637 000004 001320 MOV #4(SP),$FATAL ;;GET ERROR #
1225 005104 062766 000002 000004 ADD #2,4(SP) ;;BUMP RETURN ADDR.
1226 005112 005237 001316 INC $MSGTYPE ;;TELL APT TO TAKE ERROR
1227 005116 105037 005142 12$: CLRB $FFLG ;;CLEAR FATAL FLAG
1228 005122 105037 005141 CLRB $LFLG ;;CLEAR LOG FLAG
1229 005126 105037 005140 CLRB $MFLG ;;CLEAR MESSAGE FLAG
1230 005132 012601 MOV (SP)+,R1 ;;POP STACK INTO R1
1231 005134 012600 MOV (SP)+,RO ;;POP STACK INTO RO
1232 005136 000207 RTS PC ;;RETURN
1233 005140 000 $MFLG: .BYTE 0 ;;MESSG. FLAG
1234 005141 000 $LFLG: .BYTE 0 ;;LOG FLAG
1235 005142 000 $FFLG: .BYTE 0 ;;FATAL FLAG
1236 005144 .EVEN
1237 000200 APTSIZE=200
1238 000001 AP*ENV=001
1239 000100 APTSPool=100

```

```

1240      00004C      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 000004 000002 SRDCHR: 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$:  TSTB      2$TKS          ;: WAIT FOR
1262      005160 100375      1$      BPL          1$          ;: A CHARACTER
1263      005162 117766 174060 000004      MOVB      2$TKB, 4(SP)      ;: READ THE TTY
1264      005170 042766 177600 000004      BIC      8(C177), 4(SP)      ;: GET RID OF JUNK IF ANY
1265      005176 026627 000004 000023      CMP      4(SP), #23      ;: IS IT A CONTROL-S?
1266      005204 001013      2$:  BNE      3$          ;: BRANCH IF NO
1267      005206 105777 174032      TSTB      2$TKS          ;: WAIT FOR A CHARACTER
1268      005212 100375      2$      BPL          2$          ;: LOOP UNTIL ITS THERE
1269      005214 117746 174026      MOVB      2$TKB, -(SP)      ;: GET CHARACTER
1270      005220 042716 177500      BIC      8(C177), (SP)      ;: MAKE IT 7-BIT ASCII
1271      005224 022627 000021      CMP      (SP)+, #21      ;: IS IT A CONTROL-Q?
1272      005230 001366      2$      BNE      2$          ;: IF NOT DISCARD IT
1273      005232 000750      1$      BR          1$          ;: YES, RESUME
1274      005234 026627 000004 000140 3$:  CMP      4(SP), #140      ;: IS IT UPPER CASE?
1275      005242 002407      4$      BLT          4$          ;: BRANCH IF YES
1276      005244 026627 000004 000175      CMP      4(SP), #175      ;: IS IT A SPECIAL CHAR?
1277      005252 003003      4$      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      ;      ROLIN          ;: 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
1287      005264 010346 005520      $ROLIN: MOV      R3, -(SP)      ;: SAVE R3
1288      005266 005046      1$:  CLR          -(SP)      ;: CLEAR THE RUBOUT KEY
1289      005270 012703 005527      1$:  MOV      #1$TTYIN, R3      ;: GET ADDRESS
1290      005274 022703 005527      2$:  CMP      #1$TTYIN+7, R3      ;: BUFFER FULL?
1291      005300 101456      4$      BLOS          4$          ;: BR IF YES
1292      005302 104402      RDCHR          ;: GO READ ONE CHARACTER FROM THE TTY
1293      005304 112613      10$: MOVB      (SP)+, (R3)      ;: GET CHARACTER
1294      005306 12---13 000177      10$: CMPB      #177, (R3)      ;: IS IT A RUBOUT
1295      005312 001022      5$      BNE          5$          ;: BR IF NO

```

```

1296 005314 005716          IST      (SP)          ;; IS THIS THE FIRST RUBOUT?
1297 005316 001007          BNE      6$          ;; BR IF NO
1298 005320 112737 000134 JO5516  MOVB      #' \, 9$      ;; TYPE A BACK SLASH
1299 005326 104401 005516          TYPE      9$
1300 005332 012716 177777          MOV      1-1, (SP)      ;; SET THE RUBOUT KEY
1301 005336 005303          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          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          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          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 CHARACTER
1322 005436 104401 001312          TYPE      $QUES      ;; TYPE A '?'
1323 005442 000712          BR      1$          ;; CLEAR THE BUFFER AND LOOP
1324 005444 111337 005516          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  /IU<15><12>      ;; CONTROL "U"
1340 005534 043536 005015 000 $CNTLG: .ASCIZ  /IG<15><12>      ;; CONTROL "G"
1341 005541 015 051412 051127 $MSWR: .ASCIZ  <15><12>/SWR = /
1342 005546 036440 000040          $MNEW: .ASCIZ  / NEW = /
1343 005552 020040 042516 020127 .EVEN
1344 005560 020075 000          .SBTTL READ AN OCTAL NUMBER FROM THE TTY
1345 005564
1346
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

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 29
 DZKCE.P11 12-MAY-77 12:23

PAGE: 0050

```

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
1357      ;*      RETURN HERE
1358      ;*
1359      ;: READ AN OCTAL NUMBER
1360      ;: LOW ORDER BITS ARE ON TOP OF THE STACK
1361      ;: HIGH ORDER BITS ARE IN $HIOCT
1362      $RDOCT: MOV      (SP), -(SP)      ;: PROVIDE SPACE FOR THE
1363      MOV      4(SP), 2(SP)      ;: INPUT NUMBER
1364      MOV      R0, -(SP)      ;: PUSH R0 ON STACK
1365      MOV      R1, -(SP)      ;: PUSH R1 ON STACK
1366      MOV      R2, -(SP)      ;: PUSH R2 ON STACK
1367      1$: ROLIN      ;: READ AN ASCII LINE
1368      MOV      (SP)+, R0      ;: GET ADDRESS OF 1ST CHARACTER
1369      MOV      R0, 5$      ;: AND SAVE IT
1370      CLR      R1      ;: CLEAR DATA WORD
1371      CLR      R2
1372      2$: MOVB      (R0)+, -(SP)      ;: PICKUP THIS CHARACTER
1373      BEQ      3$      ;: IF ZERO GET OUT
1374      CMPB      #'0, (SP)      ;: MAKE SURE THIS CHARACTER
1375      BGT      4$      ;: IS AN OCTAL DIGIT
1376      CMPB      #'7, (SP)
1377      BLT      4$
1378      ASL      R1      ;: *2
1379      ROL      R2      ;: *4
1380      ASL      R1      ;: *8
1381      ROL      R2
1382      BIC      #'C7, (SP)      ;: STRIP THE ASCII JUNK
1383      ADD      (SP)+, R1      ;: ADD IN THIS DIGIT
1384      BR      2$      ;: LOOP
1385      3$: TST      (SP)+      ;: CLEAN TERMINATOR FROM STACK
1386      MOV      R1, 12(SP)      ;: SAVE THE RESULT
1387      MOV      R2, $HIOCT
1388      MOV      (SP)+, R2      ;: POP STACK INTO R2
1389      MOV      (SP)+, R1      ;: POP STACK INTO R1
1390      MOV      (SP)+, R0      ;: POP STACK INTO R0
1391      RTI      ;: RETURN
1392      4$: TST      (SP)+      ;: CLEAN PARTIAL FROM STACK
1393      CLRB      (R0)      ;: SET A TERMINATOR
1394      TYPE      ;: TYPE UP THRU THE BAD CHAR.
1395      5$: .WORD      0      ;: "?" "CR" & "LF"
1396      TYPE      $QUES      ;: TRY AGAIN
1397      BR      1$
1398      $HIOCT: .WORD      0      ;: HIGH ORDER BITS GO HERE
1399
1400      ; INPUT OCTAL NUMBER ROUTINE
1401      -----
1402      $INPUT: MOV      R5, -(SP)      ; SAVE REGISTER R5.
1403      MOV      2(SP), R5      ; GET FIRST PARAMETER ADDRESS.
1404      MOV      (R5)+, WHAT      ; GET MESSAGE ADDRESS.
1405      MOV      (R5)+, LOLIM      ; GET LOW LIMIT FOR THE #
1406      MOV      (R5)+, HILIM      ; GET HIGH LIMIT FOR THE #.
1407
1408      005564 011646
1409      005566 016666 000004 000002
1410      005574 010046
1411      005576 010146
1412      005600 010246
1413      005602 104403
1414      005604 012600
1415      005606 010037 005712
1416      005612 005001
1417      005614 005002
1418      005616 112046
1419      005620 001420
1420      005622 122716 000060
1421      005626 003026
1422      005630 122716 000067
1423      005634 002423
1424      005636 006301
1425      005640 006102
1426      005642 006301
1427      005644 006102
1428      005646 006301
1429      005650 006102
1430      005652 042716 177770
1431      005656 062601
1432      005660 000756
1433      005662 005726
1434      005664 010166 000012
1435      005670 010237 005722
1436      005674 012602
1437      005676 012601
1438      005700 012600
1439      005702 000002
1440      005704 005726
1441      005706 105010
1442      005710 104401
1443      005712 000000
1444      005714 104401 001312
1445      005720 000730
1446      005722 000000
1447
1448      005724 010546
1449      005726 016605 000002
1450      005732 012537 005770
1451      005736 012537 006050
1452      005742 012537 006052

```

M04

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 30
 DZKCE.P11 12-MAY-77 12:23

PAGE: 0051

READ AN OCTAL NUMBER FROM THE TTY

```

1408 005746 012537 006054      MOV      (RS)+,WHERE      ; GET ADDRESS OF INBUFFER
1409 005752 112537 006056      MOV      (RS)+,LOBITS      ; GET LOWMASK BITS.
1410 005756 112537 006057      MOV      (RS)+,AORCNT      ; GET # OF #'S TO BE GENERATED.
1411 005762 010566 000002      MOV      RS,2(SP)      ; SAVE THE RETURN ADDRESS.
1412 005766 104401      INLP1: TYPE      ; TYPE THE MESSAGE.
1413 005770 000000      WHAT: .WORD      0
1414 005772 104404      RDOCT
1415 005774 021637 006052      CMP      (SP),HILIM      ; READ OCTAL # FROM KEYBOARD.
1416 006000 003003      BGT      2$      ; IS IT IN HIGH LIMIT?
1417 006002 021637 006050      CMP      (SP),LOLIM      ; BRANCH IF NO.
1418 006006 002005      BGE      3$      ; IS IT MORE THAN LOW LIMIT.
1419 006010 104401 001312      2$: TYPE      ,SQUES      ; BRANCH IF YES.
1420 006014 104401 001313      TYPE      ,SCLF      ; TYPE '<CR>','<LF>'
1421 006020 000762      BR      INLP1
1422 006022 013705 006054      3$: MOV      WHERE,RS      ; GET BUFFER ADDRESS.
1423 006026 011625      4$: MOV      (SP),(RS)+      ; SAVE THE # IN RIGHT PLACE.
1424 006030 062716 000002      ADD      #2,(SP)      ; NEXT SEQUENTIAL NUMBER.
1425 006034 105337 006057      DEC      AORCNT      ; COUNT BY 1.
1426 006040 001372      BNE      4$      ; BRANCH IF NOT DONE.
1427 006042 005726      TST      (SP)+      ; POP THE STACK POINTER.
1428 006044 012605      MOV      (SP)+,RS      ; 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      AORCNT: .BYTE      0
1435
1436      ; ADVANCE TO NEXT TEST HANDLER
1437      ;-----
1438
1439 006060 013716 001442      .ADVANCE: MOV      NEXT,(SP)      ; CRUNCH STACK WITH ADDRESS OF 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

```

N04

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 31

PAGE: 0052

DZKCE.P11 12-MAY-77 12:23

READ AN OCTAL NUMBER FROM THE TTY

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							
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\$:	CLPB	(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		

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 32
 DZKCE.P11 12-MAY-77 12:23 READ AN OCTAL NUMBER FROM THE TTY

PAGE: 0053

1520 006406 000000
 1521 006410 000000
 1522 006411 000000
 1523 006412 000000

WRCNT: 0
 CHRCNT: 0
 SPACNT=CHRCNT+1
 BINWRD: 0

; TRAP DISPATCH SERVICE
 ; ARGUMENT OF TRAP IS EXTRACTED
 ; AND USED AS OFFSET TO OBTAIN POINTER
 ; TO SELECTED SUBROUTINE

.SBTTL TRAP DECODER

 ; THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
 ; AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
 ; OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
 ; GO TO THAT ROUTINE.

1539 006414 010046
 1540 006416 016600 000002
 1541 006422 005740
 1542 006424 111000
 1543 006426 006300
 1544 006430 016000 006450
 1545 006434 000200

\$TRAP: MOV RO, -(SP) ; SAVE RO
 MOV 2(SP), RO ; GET TRAP ADDRESS
 TST -(RO) ; BACKUP BY 2
 MOVB (RO), RO ; GET RIGHT BYTE OF TRAP
 ASL RO ; POSITION FOR INDEXING
 MOV \$TRPAD(RO), RO ; INDEX TO TABLE
 RTS RO ; GO TO ROUTINE

; THIS IS USE TO HANDLE THE "GETPRI" MACRO

1550 006436 011646
 1551 006440 016666 000004 000002
 1552 006446 000002

\$TRAP2: MOV (SP), -(SP) ; MOVE THE PC DOWN
 MOV 4(SP), 2(SP) ; MOVE THE PSW DOWN
 RTI ; RESTORE THE PSW

.SBTTL TRAP TABLE

; THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
 ; BY THE "TRAP" INSTRUCTION.

1559
 1560
 1561 006450 006436
 1562 006452 004414

ROUTINE

 \$TRPAD: .WORD \$TRAP2
 .TYPE ; ; CALL=TYPE TRAP+1(104401) TTY TYPEOUT ROUTINE

1565 006454 005144
 1566 006456 005264
 1567 006460 005564
 1568 006462 004364
 1569 006464 006072
 1570 006466 006132
 1571 006470 007362
 1572 006472 007332
 1573 006474 007400
 1574 006476 007446
 1575 006500 007512

\$ROCHR ; ; CALL=ROCHR TRAP+2(104402) TTY TYPEIN CHARACTER ROUTINE
 \$ROLIN ; ; CALL=ROLIN TRAP+3(104403) TTY TYPEIN STRING ROUTINE
 \$RODOCT ; ; CALL=RODOCT TRAP+4(104404) READ AN OCTAL NUMBER FROM TTY
 .SCOPI ; ; CALL=SCOPI TRAP+5(104405) CALL TO LOOP ON CURRENT DATA HANDLER
 .SAVOS ; ; CALL=SAVOS TRAP+6(104406) CALL TO REGISTER SAVE ROUTINE
 .RESOS ; ; CALL=RESOS TRAP+7(104407) CALL TO REGISTER RESTORE ROUTINE
 .MSTCLR ; ; CALL=MSTCLR TRAP+10(104410) CALL TO ISSUE A MASTER CLEAR
 .DELAY ; ; CALL=DELAY TRAP+11(104411) CALL TO DELAY
 .ROMCLK ; ; CALL=ROMCLK TRAP+12(104412) CALL TO CLOCK ROM ONCE
 .DATACLK ; ; CALL=DATACLK TRAP+13(104413) CALL TO CLOCK DATA
 .TIMER ; ; CALL=TIMER TRAP+14(104414) CALL TO DELAY A CLOCK TICK

```

1576 006502 005724 $INPUT ;;CALL=INPUT TRAP+15(104415) CALL TO OCTAL # INPUT ROUTINE
1577 006504 006164 .CONVRT ;;CALL=CC*VRT TRAP+16(104416) CALL TO .....
1578 006506 006170 .CNVRT ;;CALL=CN 'T TRAP+17(104417) CALL TO .....
1579 006510 006060 .ADVANCE ;; ALL=ADVANCE TRAP+20(104420) CALL TO ADVANCE TO NEXT TEST

```

```

-----
: *****
: ERROR HANDLER
: -----

```

```

1586 006512 004737 011212 172514 $ERROR: JSR PC,CKSWR ;CHECK FOR SOFT SWR
1587 006516 032777 010000 BIT #SW12,2SWR ;BELL ON ERROR?
1588 006524 001406 BEQ XBX ;BR IF NO BELL
1589 006526 105777 172516 TSTB 2STPS ;TTY READY.
1590 006532 100003 BPL XBX ;DON'T WAIT IF TTY NOT READY.
1591 006534 112777 000207 172510 MOVB #207,2STPB ;PUSH A BELL AT THE TTY.
1592 006542 032777 020000 172470 XBX: BIT #SW13,2SWR ;DELETE ERROR PRINT OUT?
1593 006550 001107 BNE HALTS ;BR IF NO PRINT OUT WANTED.
1594 006552 021637 001216 CMP (SP),SERRPC ;WAS THIS ERROR FOUND LAST TIME?
1595 006556 001404 BEQ IS ;BR IF YES
1596 006560 011637 001216 MOV (SP),SERRPC ;RECORD BEING HERE
1597 006564 105037 001203 CLRB SERFLG ;PREPARE HEADER
1598 006570 104406 IS: SAVOS ;SAVE ALL PROC REGISTERS
1599 006572 011605 MOV (SP),R5 ;GET THE PC OF ERROR
1600 006574 162705 000002 SUB #2,R5 ;GET ADDRESS OF TRAP CALL
1601 006600 011504 MOV (R5),R4 ;GET ERROR INSTRUCTION
1602 006602 110437 001214 MOVB R4,$ITEMB ;COPY ERROR # FOR APT HANDLING
1603 006606 006304 ASL R4 ;MULT BY TWO
1604 006610 061504 ADD (R5),R4 ;DOUBLE IT
1605 006612 006304 ASL R4 ;MULT AGAIN
1606 006614 042704 177001 BIC #177001,R4 ;CLEAR JUNK
1607 006620 062704 001512 ADD #SERRTB,R4 ;GET POINTER
1608 006624 012437 006740 MOV (R4)+,ERRMSG ;GET ERROR MESSAGE
1609 006630 012437 006752 MOV (R4)+,DATAHD ;GET DATA HEADER
1610 006634 011437 006764 MOV (R4),DATABP ;GET DATA TABLE
1611 006640 105737 001203 TSTB SERFLG ;TYPE HEADREER
1612 006644 001403 BEQ TYPMSG ;BR IF YES
1613 006646 005737 006764 TST DATABP ;DOES DATA TABLE EXIST?
1614 006652 001040 BNE TYPDAT ;BR IF YES.
1615 006654 104401 001313 TYPMSG: TYPE ,SCRLF
1616 006660 104401 001313 TYPE ,SCRLF
1617 006664 005737 001444 TST LOCK
1618 006670 001402 BEQ IS
1619 006672 104401 010015 TYPE ,MASTEK
1620 006676 104401 010003 IS: TYPE ,MTSTN
1621 006702 104417 007120 CNVRT ,XTSTN ;SHOW IT
1622 006706 104401 010072 TYPE ,MERRPC ;TYPE PC.
1623 006712 104417 007112 CNVRT ,ERTABO ;SHOW IT
1624 006716 104401 001313 TYPE ,SCRLF ;GIVE A CR/LF
1625 006722 112737 177777 001203 MOVB #-1,SERFLG ;NO MORE HEADER UNLESS NO DATA TABLE.
1626 006730 005737 006740 TST ERRMSG ;IS THERE AN ERROR MESSAGE?
1627 006734 001402 BEQ WRKO.FM ;BR IF NO.
1628 006736 104401 TYPE ;ERROR MESSAGE
1629 006740 000000 ERRMSG: 0
1630 006742 WRKO.FM: TST DATAHD ;DATA HEADER?
1631 006742 005737 006752

```

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: RESOS		RESTORE PROC REGISTERS
1640	006770	122737	000001	001336	HALTS: CMPB	8APTENV, 8ENV	IS APT RUNNING?
1641	006776	001007			BNE	3\$	SKIP APT CALL IF NOT.
1642	007000	113737	001214	007012	MOVB	8ITEMB, 6\$	COPY ERROR #.
1643	007006	004737	004714		JSR	PC, 8ATY4	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	8SENDAD, 2#42	IF ACT-11 AUTOMATIC MODE, HALT!!
1647	007024	001403			BEQ	1\$	
1648	007026	005777	172206		TST	2SWR	HALT ON ERROR?
1649	007032	100005			BPL	EXITER	BR IF NO HALT ON ERROR
1650	007034	010046			1\$: PUSHRO		SAVE RO
1651	007036	016600	000002		MOV	2(SP), RO	SHOW ERROR PC IN DATA LIGHTS
1652	007042	000000			HALT		HALT
1653	007044	012577			POPRO		GET RO
1654	007046	005237	001212		EXITER: INC	8ERTTL	UPDATE ERROR COUNT
1655	007052	032777	000400	172160	BIT	8SW08, 2SWR	GOTO TOP OF TEST?
1656	007060	001007			BNE	1\$	BR IF YES
1657	007062	032777	002000	172150	BIT	8SW10, 2SWR	GOTO NEXT TEST?
1658	007070	001407			BEQ	2\$	BR IF NO
1659	007072	013737	001442	001206	MOV	NEXT, 8LPADR	SET FOR NEXT TEST
1660	007100	012706	001200		1\$: MOV	8STACK, SP	RESET SP
1661	007104	000177	172076		JMP	2\$LPADR	GOTO SPECIFIED TEST
1662	007110	000002			2\$: RTI		8LPADR
1663	007112	000001			ERTAB0: 1		
1664	007114	006	002		.BYTE	6,2	
1665	007116	001460			SAVPC		
1666	007120	000001			XTSTN: 1		
1667	007122	003	002		.BYTE	3,2	
1668	007124	001202			8TSTNM		
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	8PWRDN: MOV	8SILLUP, 28PWRVEC	SET FOR FAST UP
1677	007134	012737	000340	000026	MOV	8340, 28PWRVEC+2	PRI0:7
1678	007142	010046			MOV	RO, -(SP)	PUSH RO 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	2SWR, -(SP)	PUSH 2SWR ON STACK
1685	007162	010637	007322		MOV	SP, 8SAVR6	SAVE SP
1686	007166	012737	007200	000024	MOV	88PWRUP, 28PWRVEC	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      $SILLUP, 2*$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 15:      INC      $SAVR6 ;; WAIT FOR THE INC
1696 007222 001375          BNE      15 ;; OF WORD
1697 007224 104401 007562          TYPE     ,MPFAIL
1698 007230 104417 007324          CNVRT    ,PFTAB
1699 007234 105037 001203          CLR      $ERFLG ;; CLEAR ERROR FLAG.
1700 007240 005037 001216          CLR      $ERRPC ;; CLEAR LAST ERROR PC
1701 007244 013701 002066          MOV      KMCSR, R1 ;; RESTORE DEVICE ADDRESS.
1702 007250 005011          CLR      (R1) ;; CLEAR THE CSR.
1703 007252 104410          MSTCLR
1704 007254 012677 171760          MOV      (SP)+, 2$SWR ;; POP STACK INTO 2$SWR
1705 007260 012605          MOV      (SP)+, R5 ;; POP STACK INTO R5
1706 007262 012604          MOV      (SP)+, R4 ;; POP STACK INTO R4
1707 007264 012603          MOV      (SP)+, R3 ;; POP STACK INTO R3
1708 007266 012602          MOV      (SP)+, R2 ;; POP STACK INTO R2
1709 007270 012601          MOV      (SP)+, R1 ;; POP STACK INTO R1
1710 007272 012600          MOV      (SP)+, R0 ;; POP STACK INTO R0
1711 007274 012737 007126 000024  MOV      2*$PWRON, 2*$PWRVEC ;; SET UP THE POWER DOWN VECTOR
1712 007302 012737 000340 000026  MOV      2$40, 2*$PWRVEC+2 ;; PRI0:7
1713 007310 104401          TYPE     ,MPFAIL ;; REPORT THE POWER FAILURE
1714 007312 007562          $PWRMG: .WORD    MPFAIL ;; POWER FAIL MESSAGE POINTER
1715 007314 000002          RTI
1716 007316 000000          $SILLUP: 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          .STNM
1723
1724 007332          .DELAY:
1725 007332 012777 000020 172534  MOV      2$20, 2$KMP04
1726 007340 104412          ROMCLK    ;; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1727 007342 121111          121111 ;; POKE CLOCK DELAY BIT
1728 007344          15:
1729 007344 104412          ROMCLK    ;; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1730 007346 121224          121224 ;; FORT4+IBUS*11
1731 007350 032777 000020 172516  BIT      2$BIT4, 2$KMP04 ;; IS CLOCK BIT SET?
1732 007356 001772          BEQ      15 ;; IF NO
1733 007360 000002          RTI
1734
1735 007362          .MSTCLR:
1736 007362 152777 000100 172500  BISB     2$BIT6, 2$KMCSRH ;; SET MASTER CLEAR
1737 007370 142777 000300 172472  BICB     2$BIT6!BIT7, 2$KMCSRH ;; CLEAR MASTER CLEAR AND RUN
1738 007376 000002          RTI ;; RETURN
1739
1740 007400          .ROMCLK:
1741 007400 152777 000002 172462  BISB     2$BIT1, 2$KMCSRH ;; SET ROMI
1742 007406 013677 172464          MOV      2(SP)+, 2$KMP06 ;; LOAD INSTRUCTION IN SEL6
1743 007412 062746 000002          ADD      2, -(SP) ;; ADJUST STACK

```

```

1744 007416 032777 000100 171614 BIT #SW06,2SWR ;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$: BISB #BIT1:BIT0,2KMCSRH ;CLOCK INSTRUCTION
1748 007436 142777 000007 172424 BICB #BIT2:BIT1:BIT0,2KMCSRH ;CLEAR ROM0, ROM1, STEP
1749 007444 000002 RTI
1750
1751 007446 .DATACLK:
1752 007446 013637 011106 MOV 2(SP)+,TEMP ;PUT TICK COUNT IN TEMP
1753 007452 062746 000002 ADD #2,-(SP) ;ADJUST STACK
1754 007456 152777 000020 172404 1$: BISB #BIT4,2KMCSRH ;SET STEP LU
1755 007464 027777 172376 172374 CMP 2KMCSR,2KMCSR ;WASTE TIME
1756 007472 142777 000020 172370 BICB #BIT4,2KMCSRH ;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$: .BLKW 1
1761
1762 007512 .TIMER:
1763 007512 013637 011106 MOV 2(SP)+,TEMP ;MOVE COUNT TO TEMP
1764 007516 062746 000002 ADD #2,-(SP) ;ADJUST STACK
1765 007522 1$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1766 007522 104412 021364 ;PORT4+IBUS* REG11
1767 007524 021364 BIT #2,2KMP04 ;IS PGM CLOCK BIT CLEAR?
1768 007526 032777 000002 172340 BEQ 1$ ;BR IF YES
1769 007534 001772 2$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1770 007536 104412 021364 ;PORT4+IBUS* REG11
1771 007536 021364 BIT #2,2KMP04 ;IS PGM CLOCK BIT SET?
1772 007540 021364 BNE 2$ ;BR IF YES
1773 007542 032777 000002 172324 DEC TEMP ;DEC COUNT
1774 007550 001372 011106 BNE 1$ ;BR IF NOT DONE
1775 007552 005337 011106 RTI ;RETURN
1776 007556 001361
1777 007560 000002
1778
1779 007562 050200 051127 043040 MPFAIL: ASCIZ <200>/PWR FAILED. RESTART AT TEST
(2) 007620 042600 042116 050040 MEPASS: ASCIZ <200>/END PASS DZKCE /
(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: ASCIZ <200>/
(2) 010136 020200 020040 020040 ASCIZ <200>/
(2) 010175 200 020040 041520 ASCIZ <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 PRI0: .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 (DOCMP 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 KMC1: .ASCIZ / (KMC) /
(2) .EVEN
(2) 011060 000005 XSTAT0: 5
1780 011062 006 003 .BYTE 6,3
1781 011064 001276 STMP0
1782 011066 006 003 .BYTE 6,3
1783 011070 001300 STMP1
1784 011072 006 003 .BYTE 6,3
1785 011074 001302 STMP2
1786 011076 006 003 .BYTE 6,3
1787 011100 001304 STMP3
1788 011102 006 002 .BYTE 6,2
1789 011104 001306 STMP4
1790 .EVEN
1791 ;BUFFERS FOR INPUT-OUTPUT
1792
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
1804 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,SENV ;IS IT RUNNING UNDER APT?
1807 011230 001071 BNE CKSWR5 ;EXIT IF YES.
1808 011232 022777 000007 170006 CMP #7,STKB ;WAS CTRL G TYPED? (7 BIT ASCII)
1809 011240 001404 BEQ 1$ ;BR IF YES
1810 011242 022777 000207 167776 CMP #207,STKB ;WAS CTRL G TYPED? (8 BIT ASCII)
1811 011250 001061 BNE CKSWR5 ;BR IF NO
1812 011252 010246 1$: MOV R2,-(SP) ;STORE R2
1813 011254 010346 MOV R3,-(SP) ;STORE R3
1814 011256 010446 MOV R4,-(SP) ;STORE R4
1815 011260 012737 177777 011416 MOV #-1,SWFLG ;SET SOFT TYPE OUT FLAG
1816 011266 005002 CKSWR1: CLR R2 ;CLEAR NEW SWR CONTENTS
1817 011270 012704 177777 MOV #-1,R4 ;SET FLAG TO ALL ONES
1818 011274 104401 005541 TYPE ,SMSWR ;TYPE "SWR="
1819 011300 104417 CKSWR2: CNVRT ;TYPE OUT PRESENT CONTENTS
1820 011302 011452 SOFTSW ;OF SOFT SWITCH REGISTER
1821 011304 104401 005552 CKSWR3: TYPE ,SMNEW ;TYPE "NEW"
1822 011310 004737 011420 CKSWR4: JSR PC,INCHAR ;GET RESPONSE
1823 011314 022703 000015 CMP #15,R3 ;WAS IT A CR?
1824 011320 001424 BEQ 5$ ;BR IF YES

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 38
 DZKCE.P11 12-MAY-77 12:23 POWER DOWN AND UP ROUTINES

PAGE: 0059

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	R2,2SWR	; 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	2\$TK5	
1851	011424	100375				BPL	.-4	
1852	011426	017703	167614			MOV	2\$TK8,R3	
1853	011432	105777	167612			TSTB	2\$TFS	
1854	011436	100375				BPL	.-4	
1855	011440	010377	167636			MOV	R3,2\$TP8	
1856	011444	042703	000200			BIC	#817,R3	
1857	011450	000207				RTS	PC	
1858								
1859	011452	000001			SOFTSW:	1		
1860	011454	006	002			.BYTE	6.2	
1861	011456	000176				SWREG		

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 39
 DZKCE.P11 12-MAY-77 12:23 POWER DOWN AND UP ROUTINES

PAGE: 0060

```

1862
1863
1864
1865
1866
1867
1868
1869
1870
1871 011460 005737 001470
1872 011464 001004
1873 011466 104401 010731
1874 011472 000000
1875 011474 000776
1876 011476 000241
1877 011500 006137 001500
1878 011504 005537 001500
1879 011510 062737 000004 001504
1880 011516 062737 000010 001502
1881 011524 022737 002300 001502
1882 011532 001006
1883 011534 012737 002100 001502
1884 011542 012737 002302 001504
1885 011550 033737 001500 001470
1886 011556 001747
1887 011560 013700 001502
1888 011564 013702 001504
1889 011570 012037 002056
1890 011574 011037 002056
1891 011600 042737 177000 002056
1892 011606 012037 002050
1893 011612 012037 002052
1894 011616 012037 002054
1895 011622 012237 001324
1896 011626 012237 001212
1897 011632 012700 000002
1898 011636 013737 002066 002070
1899 011644 005237 002070
1900 011650 013737 002070 002072
1901 011656 005237 002072
1902 011662 013737 002072 002074
1903 011670 060037 002074
1904 011674 013737 002074 002076
1905 011702 060037 002076
1906
1907 011706 013737 002056 002060
1908 011714 060037 002060
1909 011720 013737 002060 002062
1910 011726 060037 002062
1911 011732 013737 002062 002064
1912 011740 060037 002064
1913
1914 011744 032737 000002 001446
1915 011752 001447
1916 011754
1917 011754 005737 000042

      CYCLE: TST      KMACTV      ;ARE ANY KMC11'S TO BE TESTED?
              BNE      1$          ;BR IF OK.
              TYPE     ,NOACT      ;NO KMC11'S SELECTED!!
              HALT                     ;STOP THE SHOW.
              BR        .-2         ;DISQUALIFY CONT. SW.
1$:          CLC                    ;CLEAR PROC. CARRY BIT.
              ROL      RUN          ;UPDATE POINTER
              ADC      RUN          ;CATCH CARRY FROM RUN
              ADD      #4,MILK      ;UPDATE POINTER
              ADD      #10,CREAM    ;UPDATE ADDRESS POINTER.
              CMP      #KM.MAP+200,CREAM
              BNE      2$          ;KEEP GOING, NOT ALL TESTED FOR.
              MOV      #KM.MAP,CREAM ;RESET ADDRESS POINTER.
              MOV      #CNT.MAP,MILK ;RESET PASS COUNT POINTER
2$:          BIT      RUN,KMACTV    ;IS THIS ONE ACTIVE?
              BEQ      1$          ;BR IF NO
              MOV      CREAM,R0     ;GET ADDRESS POINTER
              MOV      MILK,R2      ;GET PASS COUNT POINTER
              MOV      (R0)+,KMCSR  ;LOAD SYSTEM CTRL. REG
              MOV      (R0),KMRVEC  ;LOAD VECTOR
              BIC      #177000,KMRVEC ;CLEAR UNWANTED BITS
              MOV      (R0)+,STAT1  ;LOAD STAT1
              MOV      (R0)+,STAT2  ;LOAD STAT2
              MOV      (R0)+,STAT3  ;LOAD STAT3
              MOV      (R2)+,SPASS  ;LOAD PASS COUNT
              MOV      (R2)+,SERTTL ;LOAD ERROR COUNT
              MOV      #2,R0        ;SAVE CORE THIS WAY!
              MOV      KMCSR,KMCSRH
              INC      KMCSRH
              MOV      KMCSRH,KMCTL
              INC      KMCTL
              MOV      KMCTL,KMP04
              ADD      R0,KMP04
              MOV      KMP04,KMP06
              ADD      R0,KMP06
              MOV      KMRVEC,KMRLVL ;PTY LVL
              ADD      R0,KMRLVL
              MOV      KMRLVL,KMTVEC ;TX VEC
              ADD      R0,KMTVEC
              MOV      KMTVEC,KMTLVL ;TX LVL
              ADD      R0,KMTLVL
              BIT      #SW01,STATSW ;IS TEST NO. SELECTED
              BEQ      7$          ;BR IF NO
4$:          TST      J#42         ;RUNNING IN AUTO MODE?

```

1918	011760	001044			BNE	7\$		;BR IF YES
1919	011762	104401	001313		TYPE	,SCRLF		
1920	011766	104415			INPUT			
1921	011770	010003			MTSTN			
1922	011772	000001			1			
1923	011774	001000			1000			
1924	011776	001202			\$TSTNM			
1925	012000	000			0			
1926	012001	001			.BYTE			
1927	012002	012700	013732		.BYTE			
1928	012006	022710			SS:	MOV	\$TST1,R0	
1929	012010	012737				CMP	(PC)+,(R0)	;CMP FIRST WORD TO 12737
1930	012012	001020				MOV	(PC)+,2(PC)+	
1931	012014	023760	001202	000002		BNE	6\$	;BR IF NOT SAME
1932	012022	001014				CMP	\$TSTNM,2(R0)	;DOES \$TSTNM MATCH?
1933	012024	022760	001202	000004		BNE	6\$	;BR IF NO
1934	012032	001010				CMP	\$TSTNM,4(R0)	;IS LAST WORD OK?
1935	012034	010037	001206			BNE	6\$	;BR IF NO
1936	012040	104401	007642			MOV	R0,\$LPADR	;IT IS A LEGAL TEST SO DO IT
1937	012044	042737	000002	001446		TYPE	MR	
1938	012052	000412				BIC	\$SW01,STRISW	
1939	012054	005720				BR	8\$	
1940	012056	020027	027444		6\$:	TST	(R0)+	;POP R0
1941	012062	001351				CMP	R0,\$TLAST+10	;AT END YET?
1942	012064	104401	001312			BNE	5\$	;BR IF NO
1943	012070	000731				TYPE	\$QUES	;YES ILLEGAL TEST NO.
1944						BR	4\$	;TRY AGAIN
1945	012072	012737	013732	001206	7\$:	MOV	\$TST1,\$LPADR	;PREPARE \$LPADR ADDRESS
1946	012100	013701	002066		8\$:	MOV	KMC11,R1	;R1 = BASE KMC11 ADDRESS
1947	012104	000177	167076			JMP	2\$LPADR	;GO START TESTING.
1948								
1949								
1950								
1951								
1952								
1953								
1954								
1955								
1956								
1957								
1958	012110							
1959	012110	000005						
1960	012112	012702	002100					
1961	012116	005022						
1962	012120	022702	002300					
1963	012124	001374						
1964	012126	005037	001472					
1965	012132	012702	002100					
1966	012136	005037	001470					
1967	012142	032737	000001	001446				
1968	012150	001002						
1969	012152	000137	012532					
1970	012156	012737	000001	001306				
1971	012164	104415						
1972	012166	010223						
1973	012170	000001						

;ROUTINE USED TO "AUTO SIZE" THE KMC11
 ;CSR AND VECTOR.
 ;NOTE: THE CSR MAY BE ANY WHERE IN THE FLOATING
 ;ADDRESS RANGE (160000:164000)
 ;AND THE VECTOR MAY BE ANY WHERE IN THE
 ;FLOATING VECTOR RANGE (300:770)

AUTO.SIZE:
 CSRMAP: MOV \$KM.MAP,R2 ;INSURE A BUS INIT.
 1\$: CLR (R2)+ ;LOAD MAP POINTER.
 CMP \$KM.END,R2 ;ZERO ENTIRE MAP
 BNE 1\$;ALL DONE?
 CLR KMMUM ;BR IF NO
 MOV \$KM.MAP,R2 ;SET OCTAL NUMBER OF KMC11'S TO 0
 CLR KMACTV ;R2 POINTS TO KMC MAP
 BIT \$SW00,STRISW ;CLEAR ACTIVE
 BNE .+6 ;QUESTIONS?
 JMP 7\$;BR IF YES
 MOV \$1,\$TMP4 ;IF NO SKIP QUESTIONS
 INPUT ;START WITH 1
 NUM
 1

1974	012172	000020		16			
1975	012174	001302		STMP2			
1976	012176	000		.BYTE	0		
1977	012177	001		.BYTE	1		
1978	012200	013737	001302 001472	MOV	STMP2,KMNUM	;KMNUM = HOW MANY	
1979	012206	104401	001313	TYPE	,SCRLF		
1980	012212	104416		CONVRT		;TYPE WHICH KMC IS BEING DONE	
1981	012214	013164		WHICH		;STMP4 IS WHICH KMC	
1982	012216	005237	001306	INC	STMP4		
1983	012222	104415		INPUT			
1984	012224	010363		CSR			
1985	012226	160000			160000		
1986	012230	164000			164000		
1987	012232	001304		STMP3			
1988	012234	000		.BYTE	0		
1989	012235	001		.BYTE	1		
1990	012236	013722	001304	MOV	STMP3,(R2)+	;STORE CSR IN MAP	
1991	012242	104415		INPUT			
1992	012244	010401		VEC			
1993	012246	000000		0			
1994	012250	000776		776			
1995	012252	001304		STMP3			
1996	012254	000		.BYTE	0		
1997	012255	001		.BYTE	1		
1998	012256	013712	001304	MOV	STMP3,(R2)	;STORE VECTOR IN MAP	
1999	012262	104401		TYPE			
2000	012264	010422		PRI0		;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		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\$:			
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	

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 42
 DZKCE.P11 12-MAY-77 12:23 POWER DOWN AND UP ROUTINES

PAGE: 0063

```

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 ;ASK IF LOOP-BACK IS ON
2035 012414 010671 CONN ;GET REPLY
2036 012416 004737 013456 JSR PC,INTTY ;Y
2037 012422 022703 000131 CMP #131,R3 ;Y
2038 012426 001406 BEQ 17$ ;N
2039 012430 022703 000116 CMP #116,R3
2040 012434 001406 BEQ 18$
2041 012436 104401 TYPE
2042 012440 001312 SQUES ;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 STMP3
2053 012470 000 .BYTE 0
2054 012471 001 .BYTE 1
2055 012472 113722 001304 MOVB STMP3,(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 STMP3
2061 012510 000 .BYTE 0
2062 012511 001 .BYTE 1
2063 012512 113722 001304 MOVB STMP3,(R2)+ ;STORE SWITCH PAC IN MAP
2064 012516 005722 TST (R2)+ ;POP OVER STAT3
2065 012520 005337 001302 33$: DEC STMP2 ;DEC KMC COUNT
2066 012524 001230 BNE 12$ ;BR IF MORE TO DO
2067 012526 000137 013064 JMP 13$ ;CONTINUE
2068 012532 012701 160000 7$: MOV #160000,R1 ;SET FOR FIRST ADDRESS TO BE TESTED
2069 012536 012737 013156 000004 MOV #65,2#4 ;SET FOR NON-EXISTANT DEVICE TIME OUT
2070 012544 005011 2$: 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 ROM0
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 21$:
2084 012616 010122 22$: MOV R1,(R2)+ ;STORE CSR IN CORE TABLE.
2085 012620 012711 001300 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 M80Y 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 M80Y 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			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 IS
2112	013002	156122	000004		BISB	4(R1),(R2)+	; STORE DDCMP LINE # IN TABLE
2113	013006	012761	021344	000006	MOV	#021344,6(R1)	; PCAT6+INSTRUCTION
2114	013014	012711	001400		MOV	#BIT8!BIT9,(R1)	; CLOCK INSTR.
2115	013020	156122	000004		BISB	4(R1),(R2)+	; STORE BMB73 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			CLR	(R1)	; CLEAR BIT 10
2122	013046	005061	000006		CLR	6(R1)	; CLEAR SEL 6
2123	013052	062701	000010		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		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			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	#6,2#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		TYPE	MERR2	; NOTIFY OPR THAT NO KMC11'S FOUND.
2140	013150	005000			CLR	AO	; MAKE DATA LIGHTS ZERO
2141	013152	000000			HALT		; STOP THE SHOW

```

2143 013154 000776 013052 6S: BR -2 ;DISABLE CONT. SW
2143 013156 012716 013052 6S: MOV #14S,(SP) ;ENTERED BY NON-EXISTANT TIME-OUT.
2144 013162 000002 013052 6S: RTI ;RETURN TO MAINSTREAM
2145
2146 013164 000001 002 WHICH: 1 ;
2147 013166 002 002 1 BYTE 2,2 ;
2148 013170 001306 002 1 $TMP4 ;
2149
2150 013172 032737 000001 001446 VECMAP: BIT #SW00,STATSW
2151 013200 001114 000001 001446 5S ;
2152 013202 012737 000340 000022 MOV #340,2#22 ;SET IOT TRAP PRIO TO 7
2153 013210 012737 013364 000020 MOV #4S,2#20 ;SET IOT TRAP VECTOR
2154 013216 012702 002100 MOV #KM,MAP,R2 ;SET SOFTWARE POINTER
2155 013222 012700 000300 MOV #300,R0 ;FLOATING VECTORS START HERE.
2156 013226 012701 000302 MOV #302,R1 ;PC OF IOT INSTR.
2157 013232 010120 000004 1S: MOV R1,(R0)+ ;START FILLING VECTOR AREA
2158 013234 012721 000004 1S: MOV #4,(R1)+ ;WITH .+2; IOT
2159 013240 022021 000004 1S: CMP (R0)+,(R1)+ ;ADD 2 TO R0 +R1
2160 013242 020127 001000 1S: CMP R1,#1000 ;
2161 013246 101771 001470 001276 1S: BLOS 1S ;BR IF MORE TO FILL
2162 013250 013737 001470 001276 2S: MOV KMACTV,$TMP0 ;STORE TEMPORALLY
2163 013256 006037 001276 2S: ROR $TMP0 ;BRING OUT A BIT
2164 013262 103063 000012 177776 BCC 5S ;BR IF ALL DONE
2165 013264 012704 000012 177776 MOV #12,R4 ;R4 IS INDEX REGISTER
2166 013270 016437 013442 177776 MOV BRLVL(R4),PS ;SET PS TO 7
2167 013276 011201 000200 000004 MOV (R2),R1 ;
2168 013300 012761 000200 000004 MOV #200,4(R1) ;
2169 013306 012711 001000 000006 MOV #BIT9,(R1) ;SET ROMI
2170 013312 012761 121111 000006 MOV #121111,6(R1) ;PUT INSTRUCTION IN PORT6
2171 013320 012711 001400 7S: MOV #BIT9!BIT8,(R1) ;FORCE AN INTERRUPT
2172 013324 105200 7S: INCB R0 ;STALL
2173 013326 001376 000002 7S: BNE -2 ;FOR TIME TO INTERRUPT
2174 013330 162704 000002 7S: SUB #2,R4 ;GET NEXT LOWEST PS LEVEL
2175 013334 001404 000002 7S: BEQ 6S ;BR IF R4 = 0
2176 013336 016437 013442 177776 MOV BRLVL(R4),PS ;MOVE NEXT LOWER LEVEL IN PS
2177 013344 000767 000002 6S: BR 7S ;BR TO DELAY
2178 013346 052762 005300 000002 6S: BIS #5300,2(R2) ;NO INTERRUPT ASSUME 300 AT LEVEL 5 AND FIX KMC11 LATER
2179 013354 005011 000010 3S: CLR (R1) ;CLEAR ROMI
2180 013356 062702 000010 3S: ADD #10,R2 ;POP SOFTWARE POINTER
2181 013362 000735 000002 4S: BR 2S ;KEEP GOING
2182 013364 051662 000002 4S: 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 R5 ;SHIFT LEVEL 4 PLACES
2186 013404 006305 R5 ;TO THE LEFT FOR THE
2187 013406 006305 R5 ;STATUS TABLE
2188 013410 006305 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 013354 CMP (SP)+,(SP)+ ;POP IOT JUNK OFF STACK
2192 013424 012716 013354 MOV #3S,(SP) ;SET FOR RETURN
2193 013430 000002 RTI ;
2194 013432 012737 004134 000020 5S: MOV #SCOPE,2#20 ;RESTORE SCOPE VECTOR
2195 013440 000207 5S: RTS PC ;ALL DONE WITH "AUTO SIZING"
2196
2197 013442 000000 BRLVL: PRO ;LEVEL C

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 45
 DZKCE.P11 12-MAY-77 12:23 POWER DOWN AND UP ROUTINES

PAGE: 0066

2198	013444	000000		PRO	: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	2STKS	;WAIT FOR DONE
2206	013462	100375		BPL	.-4	
2207	013464	017703	165556	MOV	2STKB,R3	;PUT CHAR IN R3
2208	013470	105777	165554	TSTB	2STPS	;WAIT UNTIL PRINTER IS READY
2209	013474	100375		BPL	.-4	
2210	013476	010377	165550	MOV	R3,2STPB	;ECHO CHAR
2211	013502	042703	000240	BIC	#8117!BITS,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	PRIORITY	CLEAN UP LOCAL VARIABLE
2222	013532	013700	001376	MOV	\$CDW1,R0	GET THE DEVICE COUNT
2223	013536	010037	001476	MOV	R0,\$AVNUM	SAVE THE NO. OF DEVICES
2224	013542	012701	001346	MOV	#8117!R1	GET EXTRA INFO, BITS POINTER
2225	013546	013737	001372	MOV	\$BASE,BASE	GET BASE CSR ADDRESS
2226	013554	113737	001366	MOVB	\$VECT1,VECTR	GET THE VECTOR
2227	013562	113737	001367	MOVB	\$VECT1+1,PRIORITY	GET THE PRIORITY
2228	013570	013737	001474	MOV	\$DEVH,KMACTV	SAVE THE KMC'S SELECTED ACTIVE
2229	013576	013737	001470	MOV	KMACTV,\$AVACT	SAVE THE ACTIVE REGISTER
2230	013604	012702	001402	MOV	#8000,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	PRIORITY,(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	ADD	#10,BASE	INCREMENT BASE CSR ADDRESS BY 10
2251	013702	062737	000010	ADD	#10,VECTR	INCREMENT VECTOR ADDRESS BY 10
2252	013710	000747		BR	1\$	SET THE NEXT MAP ENTRY
2253	013712			2\$:		

DZYCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 46
 DZYCE.P11 12-MAY-77 12:23 POWER DOWN AND UP ROUTINES

PAGE: 0067

2254 013712 012603
 2255 013714 012602
 2256 013716 012601
 2257 013720 012600
 2258 013722 000207
 2259 013724 000000
 2260 013726 000000
 2261 013730 000000

MOV (SP)+,R3 ; POP STACK INTO R3
 MOV (SP)+,R2 ; POP STACK INTO R2
 MOV (SP)+,R1 ; POP STACK INTO R1
 MOV (SP)+,R0 ; POP STACK INTO R0
 RTS PC ; RETURN
 VECTR: .WORD 0
 BASE: .WORD 0
 PRIATY: .WORD 0

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

TEST 1

 TEST1: SCOPE ; LOAD THE NO. OF THIS TEST
 MOV #1,STSTNM ; POINT TO THE START OF NEXT TEST.
 MOV #ST2,NEXT ; R1 CONTAINS BASE KMC11 ADDRESS
 CLR #KMC11 ; CLEAR SELD
 MOV #11,R2 ; SAVE R2 FOR TYPEOUT
 ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 021004! (20*11) ; PORT4+LINE UNIT REG 11
 MOV 4(R1),R4 ; PUT "FOUND" IN R4
 BIC #54,R4 ; CLEAR UNKNOWN BITS
 MOV #20,R5 ; PUT "EXPECTED" IN R5
 CMPB R5,R4 ; IS OUT READY SET?
 BEQ 15 ; BR IF YES
 ERROR 2 ; ERROR IN LU 11

15:

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

TEST 2

 TEST2: SCOPE ; LOAD THE NO. OF THIS TEST
 MOV #2,STSTNM ; POINT TO THE START OF NEXT TEST.
 MOV #ST3,NEXT ; R1 CONTAINS BASE KMC11 ADDRESS
 MOV #12,R2 ; SAVE R2 FOR TYPEOUT
 ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 021004! (20*12) ; PORT4+LINE UNIT REG 12
 MOV 4(R1),R4 ; PUT "FOUND" IN R4
 BIC #17,R4 ; CLEAR UNKNOWN BITS
 CLR R5 ; PUT "EXPECTED" IN R5

2262
 2263
 2264
 2265
 2266
 2267
 2268
 2269
 2270
 2271
 2272
 2273
 2274 013732 000004
 2275 013734 012737 000001 001202
 2276 013742 012737 014006 001442
 2277
 2278 013750 005077 166112
 2279 013754 012702 000011
 2280 013760 104412
 2281 013762 021224
 2282 013764 016104 000004
 2283 013770 042704 000054
 2284 013774 012705 000020
 2285 014000 120504
 2286 014002 001401
 2287 014004 104002
 2288 014006
 2289
 2290
 2291
 2292
 2293
 2294
 2295
 2296
 2297
 2298
 2299
 2300 014006 000004
 2301 014010 012737 000002 001202
 2302 014016 012737 014054 001442
 2303
 2304 014024 012702 000012
 2305 014030 104412
 2306 014032 021244
 2307 014034 016104 000004
 2308 014040 042704 000017
 2309 014044 005005

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 47
 DZKCE.P11 12-MAY-77 12:23 LINE UNIT READ/ONLY TESTS

PAGE: 0068

```

2310 014046 120504      CMPB   R5,R4      ;ARE ALL BITS CLEARED?
2311 014050 001401      BEQ     15        ;BR IF YES
2312 014052 104002      ERROR   2        ;ERROR IN LU 12
2313 014054
2314
2315
2316
2317
2318
2319
2320
2321
2322
2323
2324
2325 014054 000004      ;***** TEST 3 *****
2326 014056 012737 000003 001202      ;*MODEM CONTROL REGISTER READ/ONLY TEST
2327 014064 012737 014126 001442      ;*DO A MASTER CLEAR, VERIFY THAT ALL READ/ONLY
2328                                     ;*BITS ARE IN THE CORRECT STATE
2329                                     ;*****
2330                                     ; TEST 3
2331                                     ;-----
2332                                     ;*****
2333 014072 104410      ;*****
2334 014074 012702 000013      ;*SCOPE
2335 014100 104412      ;*MOV #3,STSTNM ; LOAD THE NO. OF THIS TEST
2336 014102 021264      ;*MOV #ST4,NEXT ; POINT TO THE START OF NEXT TEST.
2337 014104 016104 000004      ;*R1 CONTAINS BASE KMC11 ADDRESS
2338 014110 042704 000213      ;*MSTCLR ; MASTER CLEAR KMC11
2339 014114 012705 000100      ;*MOV #13,R2 ; SAVE R2 FOR TYPEOUT
2340 014120 120504      ;*ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2341 014122 001401      ;*021004! <20*13> ; PORT4+LINE UNIT REG 13
2342 014124 104002      ;*MOV 4(R1),R4 ; PUT "FOUND" IN R4
2343                                     ;*BIC #213,R4 ; CLEAR UNKNOWN BITS
2344                                     ;*MOV #100,R5 ; PUT "EXPECTED" IN R5
2345                                     ;*CMPB R5,R4 ; ARE RING, DTR, AND MODEM READY SET?
2346                                     ;*BEQ 15 ; BR IF YES
2347                                     ;*ERROR 2 ; ERROR IN LU 13
2348
2349
2350
2351
2352
2353
2354
2355
2356
2357
2358
2359
2360
2361
2362
2363
2364
2365

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 48
 DZKCE.P11 12-MAY-77 12:23 LINE UNIT READ/ONLY TESTS

PAGE: 0069

```

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

```

***** 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

 : TEST 5

 :*****
 :SCOPE
 :MOV #5,\$TSTNM ; LOAD THE NO. OF THIS TEST
 :MOV #TST6,NEXT ; POINT TO THE START OF NEXT TEST.
 :MOV #1\$,\$LOCK ; ADDRESS FOR LOCK ON DATA.
 :R1 CONTAINS BASE KMC11 ADDRESS
 :MASTER CLEAR KMC11
 :SAVE REGISTER ADDRESS FOR TYPEOUT
 :LOAD PORT4
 :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 :SET BITS IN LU-12
 :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 :READ LU-12
 :PUT "EXPECTED" IN R5
 :PUT "FOUND" IN R4
 :CLEAR UNWANTED BITS
 :IS BITS SET?
 :BR IF YES
 :ERROR, BIT 5 IS NOT SET
 :SCOPE SUBTEST (SW09=1)
 :NEW SCOPI
 :LOAD PORT4
 :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 :CLEAR BIT 5 IN LU-12
 :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 :READ LU-12
 :PUT "EXPECTED" IN R5
 :PUT "FOUND" IN R4
 :CLEAR UNWANTED BITS
 :IS BITS CLEAR?
 :BR IF YES
 :ERROR, BITS IS NOT CLEAR
 :SCOPE SUBTEST (SW09=1)

 : 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

```

2422
2423
2424 014360 000004
2425 014362 012737 000006 001202
2426 014370 012737 014520 001442
2427 014376 012737 014412 001444
2428
2429 014404 104410
2430 014406 012702 000017
2431 014412 012761 000001 000004 15:
2432 014420 104412
2433 014422 122117
2434 014424 104412
2435 014426 021365
2436 014430 012705 000001
2437 014434 116104 000005
2438 014440 042704 000376
2439 014444 120504
2440 014446 001401
2441 014450 104003
2442 014452 104405
2443 014454 012737 014462 001444
2444 014462 005061 000004 35:
2445 014466 104412
2446 014470 122117
2447 014472 104412
2448 014474 021365
2449 014476 005005
2450 014500 116104 000005
2451 014504 042704 000376
2452 014510 120504
2453 014512 001401
2454 014514 104003
2455 014516 104405 45:
2456
2457
2458
2459
2460
2461
2462
2463
2464
2465
2466
2467 014520 000004
2468 014522 012737 000007 001202
2469 014530 012737 014730 001442
2470 014536 012737 014556 001444
2471
2472 014544 104410
2473 014546 012702 000013
2474 014552 012700 000001
2475 014556
2476 014556 010061 000004
2477 014562 042761 000257 000004

*****
15: SCOPE
MOV #6, $TSTNM ; LOAD THE NO. OF THIS TEST
MOV #TST7, NEXT ; POINT TO THE START OF NEXT TEST.
MOV #15, LOCK ; ADDRESS FOR LOCK ON DATA.
; R1 CONTAINS BASE KMC11 ADDRESS
MSTCLR ; MASTER CLEAR KMC11
MOV #17, R2 ; SAVE REGISTER ADDRESS FOR TYPEOUT
MOV #1, 4(R1) ; LOAD PORT4
ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122117 ; SET BIT1 IN LU-17
ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021365 ; READ LU-17
MOV #1, R5 ; PUT "EXPECTED" IN R5
MOVB 5(R1), R4 ; PUT "FOUND" IN R4
BIC #376, R4 ; CLEAR UNWANTED BITS
CMPB R5, R4 ; IS BIT1 SET?
BEQ 25 ; BR IF YES
ERROR 3 ; ERROR, BIT 1 IS NOT SET
SCOPI ; SCOPE SUBTEST (SW09=1)
MOV #35, LOCK ; NEW SCOPI
CLR 4(R1) ; LOAD PORT4
ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122117 ; CLEAR BIT 1 IN LU-17
ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021365 ; READ LU-17
CLR R5 ; PUT "EXPECTED" IN R5
MOVB 5(R1), R4 ; PUT "FOUND" IN R4
BIC #376, R4 ; CLEAR UNWANTED BITS
CMPB R5, R4 ; IS BIT1 CLEAR?
BEQ 45 ; BR IF YES
ERROR 3 ; ERROR, BIT1 IS NOT CLEAR
SCOPI ; SCOPE SUBTEST (SW09=1)
45:

***** 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
*****
15: SCOPE
MOV #7, $TSTNM ; LOAD THE NO. OF THIS TEST
MOV #TST10, NEXT ; POINT TO THE START OF NEXT TEST.
MOV #645, LOCK ; ADDRESS FOR LOCK ON DATA.
; R1 CONTAINS BASE KMC11 ADDRESS
MSTCLR ; MASTER CLEAR KMC11
MOV #13, R2 ; SAVE REGISTER ADDRESS FOR TYPEOUT
MOV #1, R0 ; START WITH BIT 0
645: MOV R0, 4(R1) ; PUT PATTERN INTO PORT4
BIC #257, 4(R1) ; CLEAR UNWANTED BITS

```


2534	014766			64\$:	MOV	RO,4(R1)	PUT PATTERN INTO PORT4
2535	014766	010061	000004		ROMCLK		NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2536	014772	104412			122100!14		MOV DATA TO IBUS REGISTER 14
2537	014774	122114			ROMCLK		NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2538	014776	104412			21005!<14*20>		READ FROM IBUS REGISTER 14
2539	015000	021305			MOV	RO,R5	PUT EXPECTED IN R5
2540	015002	010005			MOVB	5(R1),R4	PUT "FOUND" INTO R4
2541	015004	116104	000005		CMPB	R5,R4	DATA CORRECT?
2542	015010	120504			BEQ	65\$	BR IF YES
2543	015012	001401			ERROR	3	ERROR
2544	015014	104003			65\$:	SCOPI	SW09=1?
2545	015016	104405			CLC		CLEAR CARRY
2546	015020	000241			ROLB	RO	SHIFT BIT IN RO
2547	015022	106100			BNE	64\$	IF RO=0 THEN DONE
2548	015024	001360			MOV	#67\$,LOCK	NEW SCOPI
2549	015026	012737	015042 001444		MOV	#1,RO	START WITH BIT 0
2550	015034	012700	000001		COM	RO	CHANGE TO FLOATING ZERO
2551	015040	005100		69\$:			
2552	015042			67\$:	MOV	RO,4(R1)	PUT PATTERN INTO PORT4
2553	015042	010061	000004		ROMCLK		NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2554	015046	104412			122100!14		MOV DATA TO IBUS REGISTER 14
2555	015050	122114			ROMCLK		NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2556	015052	104412			21005!<14*20>		READ FROM IBUS REGISTER 14
2557	015054	021305			MOV	RO,R5	PUT EXPECTED IN R5
2558	015056	010005			MOVB	5(R1),R4	PUT "FOUND" INTO R4
2559	015060	116104	000005		CMPB	R5,R4	DATA CORRECT?
2560	015064	120504			BEQ	68\$	BR IF YES
2561	015066	001401			ERROR	3	ERROR
2562	015070	104003			68\$:	SCOPI	SW09=1?
2563	015072	104405			COM	RO	CHANGE TO FLOATING 1
2564	015074	005100			CLC		CLEAR CARRY
2565	015076	000241			ROLB	RO	SHIFT BIT IN RO
2566	015100	106100			BNE	69\$	IF RO=0 THEN DONE
2567	015102	001356					
2568							
2569							
2570							
2571							
2572							
2573							
2574							
2575							
2576							
2577							
2578							
2579	015104	000004			*****		
2580	015106	012737	000011 001202		↑ST11: SCOPE		
2581	015114	012737	015146 001442		MOV	#11,\$STNM	; LOAD THE NO. OF THIS TEST
2582					MOV	#ST12,NEXT	; POINT TO THE START OF NEXT TEST.
2583	015122	104410					R1 CONTAINS BASE KMC11 ADDRESS
2584	015124	104412			MSTCLR		MASTER CLEAR KMC11
2585	015126	021324			ROMCLK		NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2586	015130	016104	000004		021324		PORT4+LUI5
2587	015134	113705	002052		MOV	4(R1),R4	PUT "FOUND" IN R4
2588	015140	120504			MOVB	STAT2,R5	PUT "EXPECTED" IN R5
2589	015142	001401			CMPB	R5,R4	SW OK?
					BEQ	1\$	BR IF YES

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

TEST 11

 ↑ST11: SCOPE
 MOV #11,\$STNM ; LOAD THE NO. OF THIS TEST
 MOV #ST12,NEXT ; POINT TO THE START OF NEXT TEST.
 R1 CONTAINS BASE KMC11 ADDRESS
 MASTER CLEAR KMC11
 NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 PORT4+LUI5
 PUT "FOUND" IN R4
 PUT "EXPECTED" IN R5
 SW OK?
 BR IF YES

```

2590 015144 104031          ERROR 31          ;ERROR, SWITCH PAC READ ERROR
2591 015146
2592
2593
2594          ;***** TEST 12 *****
2595          ;SWITCH PAC TEST
2596          ;THIS TEST READS SWITCH PAC#2
2597          ;THIS SWITCH PAC CONTAINS THE BM873 BOOT ADD
2598          ;*****
2599
2600          ; TEST 12
2601          ;-----
2602          ;*****
2603 015146 000004          ;ST12: SCOPE
2604 015150 012737 000012 001202          MOV #12,$STNM          ; LOAD THE NO. OF THIS TEST
2605 015156 012737 015210 001442          MOV #ST13,NEXT          ; POINT TO THE START OF NEXT TEST.
2606                                     ;R1 CONTAINS BASE KMC11 ADDRESS
2607 015164 104410          MSTCLR          ;MASTER CLEAR KMC11
2608 015166 104412          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2609 015170 021344          021344          ;PORT4+LU16
2610 015172 016104 000004          MOV 4(R1),R4          ;PUT "FOUND" IN R4
2611 015176 113705 002053          MOV# STAT2+1,R5          ;PUT "EXPECTED" IN R5
2612 015202 120504          CMPB R5,R4          ;SW OK?
2613 015204 001401          BEQ 1$          ;BR IF YES
2614 015206 104031          ERROR 31          ;ERROR, SWITCH PAC READ ERROR
2615 015210
2616
2617          ;***** TEST 13 *****
2618          ;LINE UNIT CLOCK TEST
2619          ;THIS TEST VERIFYS THAT THE LU INTERNAL CLOCK
2620          ;*(BIT 1 IN LU-17) IS WORKING
2621          ;*****
2622
2623          ; TEST 13
2624          ;-----
2625          ;*****
2626          ;ST13: SCOPE
2627 015210 000004          MOV #13,$STNM          ; LOAD THE NO. OF THIS TEST
2628 015212 012737 000013 001202          MOV #ST14,NEXT          ; POINT TO THE START OF NEXT TEST.
2629 015220 012737 015310 001442          ;R1 CONTAINS BASE KMC11 ADDRESS
2630                                     ;MASTER CLEAR KMC11
2631 015226 104410          MSTCLR          ;PREPARE FOR DELAY
2632 015230 005037 011106          CLR TEMP
2633 015234
2634 015234 104412          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2635 015236 021364          021364          ;PORT4+LU-17
2636 015240 032761 000002 000004          BIT #2,4(R1)          ;IS CLOCK BIT SET?
2637 015246 001004          BNE 2$          ;BR IF YES
2638 015250 005237 011106          INC TEMP          ;DELAY
2639 015254 001367          BNE 1$          ;DELAY FINISHED?
2640 015256 104004          ERROR 4          ;ERROR BIT IS STUCK CLEAR
2641 015260 005037 011106          CLR TEMP          ;PREPARE FOR DELAY
2642 015264
2643 015264 104412          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2644 015266 021364          021364          ;PORT4+LU-17
2645 015270 032761 000002 000004          BIT #2,4(R1)          ;IS CLOCK BIT CLEAR?

```

```

2646 015276 001404          BEQ 4$          ;BR IF YES
2647 015300 005237 011106   INC 4$          ;DELAY
2648 015304 001367          BNE 3$          ;BR IF DELAY NOT DONE
2649 015306 104004          ERROR 4          ;ERROR BIT IS STUCK SET
2650 015310
2651
2652
2653
2654
2655
2656
2657
2658
2659
2660
2661
2662
2663 015310 000004          ;***** TEST 14 *****
2664 015312 012737 000014 001202 ;*OUT DATA SILO TEST
2665 015320 012737 015410 001442 ;*SET SOM AND LOAD OUT DATA SILO
2666
2667 015326 104410          ;*VERIFY THAT OCOR SET, INDICATING THAT THE
2668 015330 012711 004000          ;*CHARACTER IS AT THE BOTTOM OF THE OUT SILO
2669 015334 012761 000001 000004 ;*****
2670 015342 104412          ; TEST 14
2671 015344 122111          ;-----
2672 015346 104412          ;*****
2673 015350 122110          ;*****
2674 015352 104414 000002          ;*****
2675 015356 012702 000017          ;*****
2676 015362 104412          ;*****
2677 015364 021364          ;*****
2678 015366 016104 000004          ;*****
2679 015372 042704 000357          ;*****
2680 015376 012705 000020          ;*****
2681 015402 120504          ;*****
2682 015404 001401          ;*****
2683 015406 104005          ;*****
2684 015410          ;*****
2685
2686
2687
2688
2689
2690
2691
2692
2693
2694
2695
2696
2697 015410 000004          ;*****
2698 015412 012737 000015 001202 ;*****
2699 015420 012737 015546 001442 ;*****
2700
2701 015426 104410          ;*****

```

4\$: BEQ 4\$;BR IF YES
 INC 4\$;DELAY
 BNE 3\$;BR IF DELAY NOT DONE
 ERROR 4 ;ERROR BIT IS STUCK SET

***** 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

TEST 14

15\$: SCOPE ; LOAD THE NO. OF THIS TEST
 MOV #14,\$STNM ; POINT TO THE START OF NEXT TEST.
 MOV \$ST15,NEXT ;R1 CONTAINS BASE KMC11 ADDRESS
 ;MASTER CLEAR KMC11
 ;SET LINE UNIT LOOP
 ;LOAD PORT4 WITH BIT0
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;SET SOM
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;LOAD OUT DATA SILO
 ;WAIT FOR OCOR
 ;SAVE ADDRESS FOR TYPEOUT
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;PORT4+LU 1?
 ;PUT "FOUND" IN R4
 ;CLEAR UNWANTED BITS
 ;PUT "EXPECTED" IN R5
 ;IS OCOR SET?
 ;BR IF YES

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

TEST 15

15\$: SCOPE ; LOAD THE NO. OF THIS TEST
 MOV #15,\$STNM ; POINT TO THE START OF NEXT TEST.
 MOV \$ST16,NEXT ;R1 CONTAINS BASE KMC11 ADDRESS
 ;MASTER CLEAR KMC11

2702	015430	012711	004000		MOV	#BIT11,(R1)	;SET LINE UNIT LOOP
2703	015434	012761	000001	000004	MOV	#1,4(R1)	;LOAD PORT4 WITH BIT0
2704	015442	104412			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2705	015444	122111			122111		;SET SOM
2706	015446	104412			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2707	015450	122110			122110		;LOAD OUT DATA SILO
2708	015452	004737	030006		JSR	PC,OCOR	;WAIT FOR OCOR
2709	015456	104413	000002		DATACLK,	2	;CLOCK DATA FOUR TIMES
2710	015462	012702	000011		MOV	#11,R2	;SAVE ADDRESS FOR TYPEOUT
2711	015466	104412			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2712	015470	021224			021224		;PORT4+LU 11
2713	015472	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
2714	015476	042704	000257		BIC	#257,R4	;CLEAR UNWANTED BITS
2715	015502	012705	000120		MOV	#120,R5	;PUT "EXPECTED" IN R5
2716	015506	120504			CMPB	R5,R4	;IS ACTIVE SET?
2717	015510	001401			BEQ	15	;BR IF YES
2718	015512	104005			ERROR	5	
2719	015514			15:			
2720	015514	012702	000013		MOV	#13,R2	;SAVE ADDRESS FOR TYPEOUT
2721	015520	104412			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2722	015522	021264			021264		;PORT4+LU 13
2723	015524	016104	000004		MOV	4(R1),R4	;PUT EXPECTED IN R4
2724	015530	042704	000337		BIC	#337,R4	;CLEAR UNWANTED BITS
2725	015534	012705	000040		MOV	#BIT5,R5	;PUT "EXPECTED" IN R5, RTS SHOULD BE SET
2726	015540	120504			CMPB	R5,R4	;IS RTS OK?
2727	015542	001401			BEQ	25	;BR IF YES
2728	015544	104005			ERROR	5	;RTS ERROR
2729	015546			25:			
2730							
2731							
2732							
2733							
2734							
2735							
2736							
2737							
2738							
2739							
2740							
2741							
2742	015546	000004					
2743	015550	012737	000016	001202	ST16:	SCOPE	
2744	015556	012737	015744	001442	MOV	#16,STSTNM	;LOAD THE NO. OF THIS TEST
2745					MOV	#ST17,NEXT	;POINT TO THE START OF NEXT TEST.
2746	015564	104410			MSTCLR		;R1 CONTAINS BASE KMC11 ADDRESS
2747	015566	012711	004000		MOV	#BIT11,(R1)	;MASTER CLEAR KMC11
2748	015572	012761	000001	000004	MOV	#1,4(R1)	;SET LINE UNIT LOOP
2749	015600	104412			ROMCLK		;LOAD PORT4 WITH BIT0
2750	015602	122111			122111		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2751	015604	104412			ROMCLK		;SET SOM
2752	015606	122110			122110		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2753	015610	004737	030006		JSR	PC,OCOR	;LOAD OUT DATA SILO
2754	015614	104413	000002		DATACLK,	2	;WAIT FOR OCOR
2755	015620	012761	000200	000004	MOV	#BIT7,4(R1)	;CLOCK DATA FOUR TIMES
2756	015626	104412			ROMCLK		;SET BIT7 IN PORT4
2757	015630	122111			122111		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
							;SET OUT CLEAR

```

***** 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
*****

```

```

; TEST 16
;-----

```

```

*****
;ST16: SCOPE
MOV #16,STSTNM ; LOAD THE NO. OF THIS TEST
MOV #ST17,NEXT ; POINT TO THE START OF NEXT TEST.
;R1 CONTAINS BASE KMC11 ADDRESS
MASTER CLEAR KMC11
SET LINE UNIT LOOP
LOAD PORT4 WITH BIT0
NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
SET SOM
NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
LOAD OUT DATA SILO
WAIT FOR OCOR
CLOCK DATA FOUR TIMES
SET BIT7 IN PORT4
NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
SET OUT CLEAR

```

2758	015632	104413	000001		DATACLK, 1	: GIVE A TICK TO CLEAR RTS
2759	015636	012702	000017		MOV #17,R2	: SAVE ADDRESS FOR TYPEOUT
2760	015642	104412			ROMCLK	: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2761	015644	021364			021364	: PORT4+LU 17
2762	015646	016104	000004		MOV 4(R1),R4	: PUT "FOUND" IN R4
2763	015652	042704	000357		BIC #357,R4	: CLEAR UNWANTED BITS
2764	015656	005005			CLR R5	: PUT "EXPECTED" IN R5
2765	015660	120504			CMPB R5,R4	: IS OCOR CLEARED?
2766	015662	001401			BEQ 15	: BR IF YES
2767	015664	104005			ERROR 5	
2768	015666			15:		
2769	015666	012702	000013		MOV #13,R2	: SAVE ADDRESS FOR TYPEOUT
2770	015672	104412			ROMCLK	: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2771	015674	021264			021264	: PORT4+LU 13
2772	015676	016104	000004		MOV 4(R1),R4	: PUT EXPECTED IN R4
2773	015702	042704	000337		BIC #337,R4	: CLEAR UNWANTED BITS
2774	015706	005005			CLR R5	: PUT "EXPECTED" IN R5, RTS SHOULD BE CLEARED
2775	015710	120504			CMPB R5,R4	: IS RTS OK?
2776	015712	001401			BEQ 25	: BR IF YES
2777	015714	104005			ERROR 5	: RTS ERROR
2778	015716			25:		
2779	015716	012702	000011		MOV #11,R2	: SAVE ADDRESS FOR TYPEOUT
2780	015722	104412			ROMCLK	: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2781	015724	021224			021224	: PORT4+LU 11
2782	015726	016104	000004		MOV 4(R1),R4	: PUT "FOUND" IN R4
2783	015732	012705	000020		MOV #BIT4,R5	: ONLY OUT READY SHOULD BE SET
2784	015736	120504			CMPB R5,R4	: IS ACTIVE CLEAR?
2785	015740	001401			BEQ 35	: BR IF YES
2786	015742	104005			ERROR 5	: ERROR ACTIVE NOT CLEARED
2787	015744			35:		
2788						
2789						
2790						: ***** TEST 17 *****
2791						: *DDCMP TRANSMITTER TEST
2792						: *SINGLE CLOCK THE CHARACTER 0
2793						: *VERIFY EACH BIT POSITION AS IT
2794						: *PASSES THE BIT WINDOW (SI BIT)
2795						: *ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2796						: *****
2797						
2798						: TEST 17
2799						: -----
2800						: *****
2801	015744	000004			↑ST17: SCOPE	
2802	015746	012737	000017	001202	MOV #17,STSTNM	: LOAD THE NO. OF THIS TEST
2803	015754	012737	016126	001442	MOV #17,ST20,NEXT	: POINT TO THE START OF NEXT TEST.
2804						: R1 CONTAINS BASE KMC11 ADDRESS
2805	015762	104410			MSTCLR	: MASTER CLEAR KMC11
2806	015764	012711	004000		MOV #BIT11,(R1)	: SET LINE UNIT LOOP
2807	015770	004737	030140		JSR PC,OUTRDY	: WAIT FOR OUT-READY
2808	015774	012761	000001	000004	MOV #1,4(R1)	: SET BIT0 IN PORT4
2809	016002	104412			ROMCLK	: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2810	016004	122111			122111	: SET SOM!
2811	016006	012705	000000		MOV #0,R5	: LOAD CHARACTER IN R5 FOR TYPEOUT
2812	016012	004737	030140		JSR PC,OUTRDY	: WAIT FOR OUT-READY
2813	016016	010561	000004		MOV R5,4(R1)	: LOAD PORT4 WITH CHARACTER

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

```

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

```

: TEST 20

```

:*****
1ST20:  SCOPE
        MOV      #20,$TSTNM      ; LOAD THE NO. OF THIS TEST
        MOV      #TST21,NEXT     ; POINT TO THE START OF NEXT TEST.

                                ; R1 CONTAINS BASE KMC11 ADDRESS
MSTCLR                                ; MASTER CLEAR KMC11
MOV      #BIT11,(R1)              ; SET LINE UNIT LOOP
JSR      PC,OUTRDY               ; WAIT FOR OUT-READY
MOV      #1,4(R1)                ; SET BIT0 IN PORT4
ROMCLK                                ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122111                            ; SET SOM!
MOV      #125,R5 ;LOAD CHARACTER IN R5 FOR TYPEOUT
JSR      PC,OUTRDY               ; WAIT FOR OUT-READY
MOV      R5,4(R1)                ; LOAD PORT4 WITH CHARACTER
ROMCLK                                ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122110                            ; LOAD OUT DATA
JSR      PC,OCOR                 ; WAIT FOR OCOR TO SET

```

2870	016214	005003		CLR	R3		; CLEAR BIT COUNTER
2871	016216	010502		MOV	R5,R2		; LOAD CHARACTER IN R2
2872	016220	104413	000002	DATACLK,		2	; 2 TICKS TO SET UP TRANSMITTER
2873	016224	104413	000001	1\$: DATACLK,		1	; SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2874	016230	106002		RORB	R2		; SHIFT NEXT SOFTWARE BIT IN TO CARRY
2875	016232	103005		BCC	2\$		; BR IF CARRY CLEAR
2876	016234	004737	027754	JSR	PC,GETSI		; GET THE WINDOW
2877	016240	103406		BCS	3\$		; BR IF BIT IS A MARK
2878	016242	104006		ERROR	6		; ERROR BIT WAS A SPACE
2879	016244	000404		BR	3\$		; CONTINUE WITH TEST
2880	016246	004737	027754	2\$: JSR	PC,GETSI		; GET THE WINDOW
2881	016252	103001		BCC	3\$		; BR IF BIT IS A SPACE
2882	016254	104006		ERROR	6		; ERROR BIT WAS A MARK
2883	016256			3\$: INC	R3		; NEXT BIT
2884	016256	005203		CMP	#10,R3		; DONE YET?
2885	016260	022703	000010	BNE	1\$		; BR IF NO
2886	016264	001357		DATACLK,		14	; CLOCK TRANSMITTER 14 MORE TICKS
2887	016266	104413	000014	ROMCLK			; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2888	016272	104412		021264			; PORT4+LU-13
2889	016274	021264		BIT	#BITS,4(R1)		; RTS SHOULD BE CLEAR NOW
2890	016276	032761	000040 000004	BEQ	4\$		; BR IF YES
2891	016304	001401		ERROR	34		; ERROR, RTS NOT CLEAR
2892	016306	104034					
2893	016310			4\$:			

***** TEST 21 *****
 *DDCMP TRANSMITTER TEST
 *SINGLE CLOCK THE CHARACTER 252
 *VERIFY EACH BIT POSITION AS IT
 *PASSES THE BIT WINDOW (SI BIT)
 *ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE

TEST 21

2906				*****			
2907	016310	000004		†ST21: SCOPE			
2908	016312	012737	000021 001202	MOV	#21,STSTNM		; LOAD THE NO. OF THIS TEST
2909	016320	012737	016472 001442	MOV	#ST22,NEXT		; POINT TO THE START OF NEXT TEST.
2910							; R1 CONTAINS BASE KMC11 ADDRESS
2911	016326	104410		MSTCLR			; MASTER CLEAR KMC11
2912	016330	012711	004000	MOV	#BIT11,(R1)		; SET LINE UNIT LOOP
2913	016334	004737	030140	JSR	PC,OUTRDY		; WAIT FOR OUT-READY
2914	016340	012761	000001 000004	MOV	#1,4(R1)		; SET BIT0 IN PORT4
2915	016346	104412		ROMCLK			; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2916	016350	122111		122111			; SET SOM!
2917	016352	012705	000252	MOV	#252,R5		; LOAD CHARACTER IN R5 FOR TYPEOUT
2918	016356	004737	030140	JSR	PC,OUTRDY		; WAIT FOR OUT-READY
2919	016362	010561	000004	MOV	R5,4(R1)		; LOAD PORT4 WITH CHARACTER
2920	016366	104412		ROMCLK			; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2921	016370	122110		122110			; LOAD OUT DATA
2922	016372	004737	030006	JSR	PC,OCOR		; WAIT FOR OCOR TO SET
2923	016376	005003		CLR	R3		; CLEAR BIT COUNTER
2924	016400	010502		MOV	R5,R2		; LOAD CHARACTER IN R2
2925	016402	104413	000002	DATACLK,		2	; 2 TICKS TO SET UP TRANSMITTER

BASIC TRANSMITTER TESTS

```

2926 016406 104413 000001 1S: DATACLK, 1 ; SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2927 016412 106002 RORB R2 ; SHIFT NEXT SOFTWARE BIT IN TO CARRY
2928 016414 103005 BCC 2S ; BR IF CARRY CLEAR
2929 016416 004737 027754 JSR PC,GETSI ; GET THE WINDOW
2930 016422 103406 BCS 3S ; BR IF BIT IS A MARK
2931 016424 104006 ERROR 6 ; ERROR BIT WAS A SPACE
2932 016426 000404 BR 3S ; CONTINUE WITH TEST
2933 016430 004737 027754 2S: JSR PC,GETSI ; GET THE WINDOW
2934 016434 103001 BCC 3S ; BR IF BIT IS A SPACE
2935 016436 104006 ERROR 6 ; ERROR BIT WAS A MARK
2936 016440 3S:
2937 016440 005203 INC R3 ; NEXT BIT
2938 016442 022703 000010 CMP #10,R3 ; DONE YET?
2939 016446 001357 BNE 1S ; BR IF NO
2940 016450 104413 000014 DATACLK, 14 ; CLOCK TRANSMITTER 14 MORE TICKS
2941 016454 104412 ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2942 016456 021264 021264 PORT4+LU-13
2943 016460 032761 000040 000004 BIT #BITS,4(R1) ; RTS SHOULD BE CLEAR NOW
2944 016466 001401 BEQ 4S ; BR IF YES
2945 016470 104034 ERROR 3H ; ERROR, RTS NOT CLEAR
2946 016472 4S:

```

```

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

```

TEST 22

```

2950 *****
2951 *DOCMP TRANSMITTER TEST
2952 *SINGLE CLOCK THE CHARACTER 377
2953 *VERIFY EACH BIT POSITION AS IT
2954 *PASSES THE BIT WINDOW (SI BIT)
2955 *ON AN ERROR, R3 CONTAINS BIT POSITION OF FAILURE
2956 *****
2957 ; TEST 22
2958 ; -----
2959 ; *****
2960 016472 000004 1S22: SCOPE
2961 016474 012737 000022 001202 MOV #22,$TSTNM ; LOAD THE NO. OF THIS TEST
2962 016502 012737 016654 001442 MOV $TST23,NEXT ; POINT TO THE START OF NEXT TEST.
2963 ; R1 CONTAINS BASE KMC11 ADDRESS
2964 016510 104410 MSTCLR ; MASTER CLEAR KMC11
2965 016512 012711 004000 MOV #BIT11,(R1) ; SET LINE UNIT LOOP
2966 016516 004737 030140 JSR PC,OUTRDY ; WAIT FOR OUT-READY
2967 016522 012761 000001 000004 MOV #1,4(R1) ; SET BIT0 IN PORT4
2968 016530 104412 ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2969 016532 122111 122111 SET SOM!
2970 016534 012705 000377 MOV #377,R5 ; LOAD CHARACTER IN R5 FOR TYPEOUT
2971 016540 004737 030140 JSR PC,OUTRDY ; WAIT FOR OUT-READY
2972 016544 010561 000004 MOV R5,4(R1) ; LOAD PORT4 WITH CHARACTER
2973 016550 104412 ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2974 016552 122110 122110 LOAD OUT DATA
2975 016554 004737 030006 JSR PC,OCOR ; WAIT FOR OCOR TO SET
2976 016560 005003 CLR R3 ; CLEAR BIT COUNTER
2977 016562 010502 MOV R5,R2 ; LOAD CHARACTER IN R2
2978 016564 104413 000002 DATACLK, 2 ; 2 TICKS TO SET UP TRANSMITTER
2979 016570 104413 000001 1S: DATACLK, 1 ; SHIFT NEXT BIT IN THE WINDOW (SI BIT)
2980 016574 106002 RORB R2 ; SHIFT NEXT SOFTWARE BIT IN TO CARRY
2981 016576 103005 BCC 2S ; BR IF CARRY CLEAR

```

BASIC TRANSMITTER TESTS

2992	016600	004737	027754		JSR	PC,GETSI	:GET THE WINDOW
2993	016604	103406			BCS	3\$	:BR IF BIT IS A MARK
2994	016606	104006			ERROR	6	:ERROR BIT WAS A SPACE
2995	016610	000404			BR	3\$	:CONTINUE WITH TEST
2996	016612	004737	027754	2\$:	JSR	PC,GETSI	:GET THE WINDOW
2997	016616	103001			BCC	3\$	:BR IF BIT IS A SPACE
2998	016620	104006			ERROR	6	:ERROR BIT WAS A MARK
2999	016622			3\$:			
2990	016622	005203			INC	R3	:NEXT BIT
2991	016624	022703	000010		CMP	#10,R3	:DONE YET?
2992	016630	001357			BNE	1\$	:BR IF NO
2993	016632	104413	000014		DATACLK,	14	:CLOCK TRANSMITTER 14 MORE TICKS
2994	016636	104412			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
2995	016640	021264			021264		:PORT4=LU-13
2996	016642	032761	000040	000004	BIT	#BITS,4(R1)	:RTS SHOULD BE CLEAR NOW
2997	016650	001401			BEQ	4\$	:BR IF YES
2998	016652	104034			ERROR	34	:ERROR, RTS NOT CLEAR
2999	016654			4\$:			

```

***** TEST 23 *****
:DOCMP 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
*****

```

: TEST 23

:-----

3013					*****		
3014	016654	000004			†ST23: SCOPE		
3015	016656	012737	000023	001202	MOV	#23,ST23NM	: LOAD THE NO. OF THIS TEST
3016	016664	012737	017062	001442	MOV	#ST24,NEXT	: POINT TO THE START OF NEXT TEST.
3017							:R1 CONTAINS BASE KMC11 ADDRESS
3018	016672	104410			MSTCLR		:MASTER CLEAR KMC11
3019	016674	012711	004000		MOV	#BIT11,(R1)	:SET LINE UNIT LOOP
3020	016700	005003			CLR	R3	:R3 CONTAINS BIT COUNT
3021	016702	005004			CLR	R4	:R4 CONTAINS CHAR TO BE LOADED IN SILO
3022	016704	005005			CLR	R5	:R5 CONTAINS CHARACTER CURRENTLY BEING SHIFTED OUT
3023	016706	004737	030140		JSR	PC,OUTRDY	:WAIT FOR OUT-READY
3024	016712	012761	000001	000004	MOV	#1,4(R1)	:SET BIT0 IN PORT4
3025	016720	104412			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3026	016722	122111			122111		:SET SOM!
3027	016724	004737	030140		JSR	PC,OUTRDY	:WAIT FOR OUT-READY
3028	016730	010461	000004		MOV	R4,4(R1)	:LOAD PORT4 WITH CHARACTER
3029	016734	104412			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3030	016736	122110			122110		:LOAD OUT DATA
3031	016740	005204			INC	R4	:INCREMENT TO NEXT CHARACTER
3032	016742	004737	030140		JSR	PC,OUTRDY	:WAIT FOR OUT-READY
3033	016746	010461	000004		MOV	R4,4(R1)	:LOAD PORT4 WITH CHARACTER
3034	016752	104412			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3035	016754	122110			122110		:LOAD OUT DATA
3036	016756	004737	030006		JSR	PC,OCOR	:WAIT FOR OCOR TO SET
3037	016762	104413	000002		DATACLK,	2	:2 TICKS TO SET UP TRANSMITTER

```

3038 016766 005203 45: CLR R3 ;CLEAR BIT COUNTER
3039 016770 010502 MOV R5,R2 ;LOAD CHARACTER IN R2
3040 016772 104413 000001 15: DATACLK, 1 ;SHIFT NEXT BIT IN THE WINDOW (SI BIT)
3041 016776 106002 RORB R2 ;SHIFT NEXT SOFTWARE BIT IN TO CARRY
3042 017000 103005 BCC 25 ;BR IF CARRY CLEAR
3043 017002 004737 027754 JSR PC,GETSI ;GET THE WINDOW
3044 017006 103406 BCS 35 ;BR IF BIT IS A MARK
3045 017010 104006 ERROR 6 ;ERROR BIT WAS A SPACE
3046 017012 000404 BR 35 ;CONTINUE WITH TEST
3047 017014 004737 027754 25: JSR PC,GETSI ;GET THE WINDOW
3048 017020 103001 BCC 35 ;BR IF BIT IS A SPACE
3049 017022 104006 ERROR 6 ;ERROR BIT WAS A MARK
3050 017024 35: INC R3 ;NEXT BIT
3051 017024 005203 CMP #10,R3 ;DONE YET?
3052 017026 022703 000010 BNE 15 ;BR IF NO
3053 017032 001357 INC R4 ;NEXT CHARACTER
3054 017034 005204 JSR PC,OUTRDY ;WAIT FOR OUT-READY
3055 017036 004737 030140 MOV R4,4(R1) ;LOAD PORT4 WITH CHARACTER
3056 017042 010461 000004 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3057 017046 104412 122110 ;LOAD OUT DATA
3058 017047 122110 INC R5 ;NEXT CHARACTER
3059 017052 005205 CMP #400,R5 ;DONE YET?
3060 017054 022705 000400 BNE 45 ;BR IF NO
3061 017060 001342 55:
3062 017062
3063
3064
3065 ;***** TEST 24 *****
3066 ;*DDCMP STRIP SYNC TEST
3067 ;*SET LU LOOP, SINGLE STEP 5 SYNCs
3068 ;*VERIFY THAT IN ACTIVE DOES NOT SET
3069 ;*****
3070
3071 ; TEST 24
3072 ;-----
3073 ;*****
3074 017062 000004 15: ST24: SCOPE ;*****
3075 017064 012737 000024 001202 MOV #24,STSTNM ; LOAD THE NO. OF THIS TEST
3076 017072 012737 017150 001442 MOV #TS125,NEXT ; POINT TO THE START OF NEXT TEST.
3077 ;R1 CONTAINS BASE KMC11 ADDRESS
3078 017100 104410 MSTCLR ;MASTER CLEAR KMC11
3079 017102 012711 004000 MOV #BIT11,(R1) ;SET LU LOOP
3080 017106 012702 000012 MOV #12,R2 ;SAVE LU REG FOR TYPEOUT
3081 017112 004737 030024 JSR PC,SYNC ;SINGLE CLOCK 5 SYNC CHARACTERS
3082 017116 000005 5
3083 017120 104413 000054 DATACLK, 54 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3084 017124 104412 ROMCLK 021244 ;PORT4+LU12
3085 017126 021244 MOV 4(R1),R4 ;PUT "FOUND" IN R4
3086 017130 016104 000004 BIC #277,R4 ;CLEAR UNWANTED BITS
3087 017134 042704 000277 CLR R5 ;PUT "EXPECTED" IN R5
3088 017140 005005 CMPB R5,R4 ;IS ACTIVE CLEAR?
3089 017142 120504 BEQ 15 ;BR IF YES
3090 017144 001401 ERROR 40 ;ERROR ACTIVE IS NOT CLEAR
3091 017146 104040
3092 017150
3093

```

3094
 3095
 3096
 3097
 3098
 3099
 3100
 3101
 3102
 3103
 3104
 3105
 3106
 3107
 3108
 3109
 3110
 3111
 3112
 3113
 3114
 3115
 3116
 3117
 3118
 3119
 3120
 3121
 3122
 3123
 3124
 3125
 3126
 3127
 3128
 3129
 3130
 3131
 3132
 3133
 3134
 3135
 3136
 3137
 3138
 3139
 3140
 3141
 3142
 3143
 3144
 3145
 3146
 3147
 3148
 3149

 017150 000004
 017152 012737 000025 001202
 017160 012737 017240 001442
 017166 104410
 017170 012711 004000
 017174 012702 000012
 017200 004737 030024
 017204 000005
 017206 104413 000064
 017212 104412
 017214 021244
 017216 016104 000004
 017222 042704 000277
 017226 012705 000100
 017232 120504
 017234 001401
 017236 104040
 017240

```

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

; TEST 25
-----
;*****
;ST25: SCOPE
;MOV      #25,STSTNM          ; LOAD THE NO. OF THIS TEST
;MOV      #ST26,NEXT         ; POINT TO THE START OF NEXT TEST.
;R1 CONTAINS BASE KMC11 ADDRESS
;MASTER CLEAR KMC11
;SET LU LOOP
;SAVE LU REG FOR TYPEOUT
;SINGLE CLOCK 5 SYNC CHARACTERS
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4+LUI2
;PUT "FOUND" IN R4
;CLEAR UNWANTED BITS
;PUT "EXPECTED" IN R5
;IS ACTIVE SET?
;BR IF YES
;ERROR ACTIVE IS NOT SET
;*****
;S:

```

```

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

; TEST 26
-----
;*****
;ST26: SCOPE
;MOV      #26,STSTNM          ; LOAD THE NO. OF THIS TEST
;MOV      #ST27,NEXT         ; POINT TO THE START OF NEXT TEST.
;R1 CONTAINS BASE KMC11 ADDRESS
;MASTER CLEAR KMC11
;SET LU LOOP
;SAVE LU REG FOR TYPEOUT
;SINGLE CLOCK 1 SYNC CHARACTERS
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4+LUI2
;PUT "FOUND" IN R4
;CLEAR UNWANTED BITS
;PUT "EXPECTED" IN R5
;IS ACTIVE CLEAR?
;*****

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 62
 DZKCE.P11 12-MAY-77 12:23 BASIC RECEIVER TESTS

PAGE: 0003

3150 017322 001401
 3151 017324 104040
 3152 017326
 3153
 3154
 3155
 3156
 3157
 3158
 3159
 3160
 3161
 3162
 3163

15: BEQ 15 ;BR IF YES
 ERROR 40 ;ERROR ACTIVE IS NOT CLEAR

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

TEST 27

 ST27: SCOPE ;
 MOV #27,STSTNM ; LOAD THE NO. OF THIS TEST
 MOV #TST30,NEXT ; POINT TO THE START OF NEXT TEST.
 ; R1 CONTAINS BASE KMC11 ADDRESS
 MSTCLR ; MASTER CLEAR KMC11
 MOV #BIT11,(R1) ; SET LU LOOP
 MOV #12,R2 ; SAVE LU REG FOR TYPEOUT
 JSR PC,SYNC ; SINGLE CLOCK 2 SYNC CHARACTERS
 2
 DATACLK, 34 ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ROMCLK 021244 ; PORT4+LUI2
 MOV 4(R1),R4 ; PUT "FOUND" IN R4
 BIC #277,R4 ; CLEAR UNWANTED BITS
 MOV #BIT6,R5 ; PUT "EXPECTED" IN R5
 CMPB R5,R4 ; IS ACTIVE SET?
 BEQ 15 ; BR IF YES
 ERROR 40 ; ERROR ACTIVE IS NOT SET

15:

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

TEST 30

 ST30: SCOPE ;
 MOV #30,STSTNM ; LOAD THE NO. OF THIS TEST
 MOV #TST31,NEXT ; POINT TO THE START OF NEXT TEST.
 ; R1 CONTAINS BASE KMC11 ADDRESS
 MSTCLR ; MASTER CLEAR KMC11
 MOV #12,R2 ; SAVE REG ADDRESS IN R2 FOR TYPEOUT
 MOV #BIT11,(R1) ; SET LINE UNIT LOOP
 JSR PC,CHAR ; LOAD SILO WITH 3 SYNCs
 301 ; AND A NON-SYNC (301)
 DATACLK, 53 ; SINGLE CLOCK THE DATA
 TIMER, 2 ; WAIT FOR INRDY

3183
 3184
 3185
 3186
 3187
 3188
 3189
 3190
 3191
 3192
 3193
 3194
 3195 017416 000004
 3196 017420 012737 000030 001202
 3197 017426 012737 017570 001442
 3198
 3199 017434 104410
 3200 017436 012702 000012
 3201 017442 012711 004000
 3202 017446 004737 030172
 3203 017452 000301
 3204 017454 104413 000053
 3205 017460 104414 000002

3206	017464	104412			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3207	017466	021244			021244		;PORT4+LU 12
3208	017470	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
3209	017474	042704	000357		BIC	#357,R4	;CLEAR UNWANTED BITS
3210	017500	012705	000020		MOV	#BIT4,R5	;PUT "EXPECTED" IN R5
3211	017504	120504			CMPB	R5,R4	;IS INRDY SET?
3212	017506	001401			BEQ	1\$	
3213	017510	104040			ERROR	40	;ERROR, INRDY IS NOT SET
3214	017512						
3215	017512	012761	000200	000004	1\$:	MOV	#BIT7,4(R1)
3216	017520	104412			ROMCLK		;LOAD PORT4
3217	017522	122112			122112		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3218	017524	104412			ROMCLK		;SET IN CLEAR
3219	017526	021244			021244		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3220	017530	016104	000004		MOV	4(R1),R4	;PORT4+LU 12
3221	017534	042704	000277		BIC	#277,R4	;PUT "FOUND" IN R4
3222	017540	005005			CLR	R5	;CLEAR UNWANTED BITS
3223	017542	120504			CMPB	R5,R4	;PUT "EXPECTED" IN R5
3224	017544	001401			BEQ	2\$	;IS IN ACTIVE CLEAR?
3225	017546	104040			ERROR	40	;ERROR, IN ACTIVE IS NOT CLEAR
3226	017550						
3227	017550	016104	000004		2\$:	MOV	4(R1),R4
3228	017554	042704	000357		BIC	#357,R4	;PUT "FOUND" IN R4
3229	017560	005005			CLR	R5	;CLEAR UNWANTED BITS
3230	017562	120504			CMPB	R5,R4	;PUT "EXPECTED" IN R5
3231	017564	001401			BEQ	3\$	;IS INRDY CLEARED?
3232	017566	104040			ERROR	40	;ERROR, INRDY IS NOT CLEARED
3233	017570				3\$:		
3234							
3235							
3236							
3237							
3238							
3239							
3240							
3241							
3242							
3243							
3244							
3245	017570	000004					
3246	017572	012737	000031	001202	1\$T31:	SCOPE	
3247	017600	012737	017704	001442	MOV	#31,\$TSTNM	;LOAD THE NO. OF THIS TEST
3248					MOV	#TST32,NEXT	;POINT TO THE START OF NEXT TEST.
3249	017606	104410					;R1 CONTAINS BASE KMC11 ADDRESS
3250	017610	012702	000012		MSTCLR		;MASTER CLEAR KMC11
3251	017614	012711	004000		MOV	#12,R2	;SAVE REG ADDRESS IN R2 FOR TYPEOUT
3252	017620	004737	030172		MOV	#BIT11,(R1)	;SET LINE UNIT LOOP
3253	017624	000000			JSR	PC,CHAR	;LOAD SILO WITH 3 SYNC
3254	017626	104413	000053		0		;AND THE CHARACTER 0
3255	017632	104414	000002		DATACLK,	53	;SINGLE CLOCK THE DATA
3256	017636	104412			TIMER,	2	;WAIT FOR INRDY
3257	017640	021244			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3258	017642	016104	000004		021244		;PORT4+LU 12
3259	017646	042704	000357		MOV	4(R1),R4	;PUT "FOUND" IN R4
3260	017652	012705	000020		BIC	#357,R4	;CLEAR UNWANTED BITS
3261	017656	120504			MOV	#BIT4,R5	;PUT "EXPECTED" IN R5
					CMPB	R5,R4	;IS INRDY SET?

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

; TEST 31

 ;TST31: SCOPE

MOV #31,\$TSTNM ; LOAD THE NO. OF THIS TEST
 MOV #TST32,NEXT ; POINT TO THE START OF NEXT TEST.
 ;R1 CONTAINS BASE KMC11 ADDRESS
 ;MASTER CLEAR KMC11
 ;SAVE REG ADDRESS IN R2 FOR TYPEOUT
 ;SET LINE UNIT LOOP
 ;LOAD SILO WITH 3 SYNC
 ;AND THE CHARACTER 0
 ;SINGLE CLOCK THE DATA
 ;WAIT FOR INRDY
 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 ;PORT4+LU 12
 ;PUT "FOUND" IN R4
 ;CLEAR UNWANTED BITS
 ;PUT "EXPECTED" IN R5
 ;IS INRDY SET?

H07

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 64
 DZKCE.P11 12-MAY-77 12:23 BASIC RECEIVER TESTS

PAGE: 0085

```

3262 017660 001401      BEQ      15
3263 017662 104040      ERROR    40      ;ERROR, INRDY IS NOT SET
3264 017664      15:
3265 017664 104412      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3266 017666 021204      021204      ;PORT4+IN DATA
3267 017670 016104 000004 MOV      4(R1),R4      ;PUT "FOUND" IN R4
3268 017674 005005      CLR      R5      ;PUT "EXPECTED" IN R5
3269 017676 120504      CMPB     R5,R4      ;WAS A 0 RECEIVED?
3270 017700 001401      BEQ      25
3271 017702 104010      ERROR    10      ;ERROR, RECEIVED DATA IS WRONG
3272 017704
3273
3274
3275      ;***** TEST 32 *****
3276      ;DDCMP BASIC RECEICER TEST
3277      ;SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 125
3278      ;VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3279      ;*****
3280
3281      TEST 32
3282      -----
3283      ;*****
3284 017704 000004 1ST32: SCOPE
3285 017706 012737 000032 001202 MOV      #32,$TSTNM      ; LOAD THE NO. OF THIS TEST
3286 017714 012737 020022 001442 MOV      #TST33,NEXT      ; POINT TO THE START OF NEXT TEST.
3287
3288 017722 104410      M1TCLR      ; R1 CONTAINS BASE KMC11 ADDRESS
3289 017724 012702 000012      MOV      #12,R2      ; MASTER CLEAR KMC11
3290 017730 012711 004000      MOV      #BIT11,(R1)      ; SAVE REG ADDRESS IN R2 FOR TYPEOUT
3291 017734 004737 030172      JSR      PC,CHAR      ; SET LINE UNIT LOOP
3292 017740 000125      125      ; LOAD SILO WITH 3 SYNCs
3293 017742 104413 000053      DATACLK, 53      ; AND THE CHARACTER 125
3294 017746 104414 000002      TIMER, 2      ; SINGLE CLOCK THE DATA
3295 017752 104412      ROMCLK      ; WAIT FOR INRDY
3296 017754 021244      021244      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3297 017756 016104 000004      MOV      4(R1),R4      ; PORT4+LU 12
3298 017762 042704 000357      BIC      #357,R4      ; PUT "FOUND" IN R4
3299 017766 012705 000020      MOV      #BIT4,R5      ; CLEAR UNWANTED BITS
3300 017772 120504      CMPB     R5,R4      ; PUT "EXPECTED" IN R5
3301 017774 001401      BEQ      15      ; IS INRDY SET?
3302 017776 104040      ERROR    40      ;ERROR, INRDY IS NOT SET
3303 020000      15:
3304 020000 104412      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3305 020002 021204      021204      ;PORT4+IN DATA
3306 020004 016104 000004      MOV      4(R1),R4      ;PUT "FOUND" IN R4
3307 020010 012705 000125      MOV      #125,R5      ;PUT "EXPECTED" IN R5
3308 020014 120504      CMPB     R5,R4      ;WAS A 125 RECEIVED?
3309 020016 001401      BEQ      25
3310 020020 104010      ERROR    10      ;ERROR, RECEIVED DATA IS WRONG
3311 020022
3312
3313
3314      ;***** TEST 33 *****
3315      ;DDCMP BASIC RECEICER TEST
3316      ;SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 252
3317      ;VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 65
 DZKCE.P11 12-MAY-77 12:23 BASIC RECEIVER TESTS

PAGE: 0086

```

3318                                     ;:*****
3319                                     ;
3320                                     ;   TEST 33
3321                                     ;-----
3322                                     ;:*****
3323 020022 000004          1ST33: SCOPE
3324 020024 012737 000033 001202      MOV      #33,$STSTM          ; LOAD THE NO. OF THIS TEST
3325 020032 012737 020140 001442      MOV      #1ST34,NEXT        ; POINT TO THE START OF NEXT TEST.
3326                                     ;
3327 020040 104410          MSTCLR          ; R1 CONTAINS BASE KMC11 ADDRESS
3328 020042 012702 000012      MOV      #12,R2          ; MASTER CLEAR KMC11
3329 020046 012711 004000      MOV      #BIT11,(R1)      ; SAVE REG ADDRESS IN R2 FOR TYPEOUT
3330 020052 004737 030172      JSR      PC,CHAR          ; SET LINE UNIT LOOP
3331 020056 000252          252          ; LOAD SILO WITH 3 SYNCs
3332 020060 104413 000053      DATACLK, 53          ; AND THE CHARACTER 252
3333 020064 104414 000002      TIMER, 2          ; SINGLE CLOCK THE DATA
3334 020070 104412          ROMCLK          ; WAIT FOR INRDY
3335 020072 021244          021244          ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3336 020074 016104 000004      MOV      4(R1),R4          ; PORT4+LU 12
3337 020100 042704 000357      BIC      #357,R4          ; PUT "FOUND" IN R4
3338 020104 012705 000020      MOV      #BIT4,R5          ; CLEAR UNWANTED BITS
3339 020110 120504          CMPB      R5,R4          ; PUT "EXPECTED" IN R5
3340 020112 001401          BEQ      1$          ; IS INRDY SET?
3341 020114 104040          1$: ERROR 40          ; ERROR, INRDY IS NOT SET
3342 020116          ROMCLK          ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3343 020116 104412          021204          ; PORT4+IN DATA
3344 020120 021204          MOV      4(R1),R4          ; PUT "FOUND" IN R4
3345 020122 016104 000004      MOV      #252,R5          ; PUT "EXPECTED" IN R5
3346 020126 012705 000252      CMPB      R5,R4          ; WAS A 252 RECEIVED?
3347 020132 120504          BEQ      2$          ;
3348 020134 001401          ERROR 10          ; ERROR, RECEIVED DATA IS WRONG
3349 020136 104010          2$:
3350 020140
3351
3352                                     ;:***** TEST 34 *****
3353                                     ;
3354                                     ;*DOCMP BASIC RECEICER TEST
3355                                     ;*SYNC UP RECEIVER AND SINGLE CLOCK THE CHARACTER 377
3356                                     ;*VERIFY THAT IN RDY IS SET, AND THAT THE CHARACTER WAS RECEIVED
3357                                     ;:*****
3358                                     ;
3359                                     ;   TEST 34
3360                                     ;-----
3361                                     ;:*****
3362 020140 000004          1ST34: SCOPE
3363 020142 012737 000034 001202      MOV      #34,$STSTM          ; LOAD THE NO. OF THIS TEST
3364 020150 012737 020256 001442      MOV      #1ST35,NEXT        ; POINT TO THE START OF NEXT TEST.
3365                                     ;
3366 020156 104410          MSTCLR          ; R1 CONTAINS BASE KMC11 ADDRESS
3367 020160 012702 000012      MOV      #12,R2          ; MASTER CLEAR KMC11
3368 020164 012711 004000      MOV      #BIT11,(R1)      ; SAVE REG ADDRESS IN R2 FOR TYPEOUT
3369 020170 004737 030172      JSR      PC,CHAR          ; SET LINE UNIT LOOP
3370 020174 000377          377          ; LOAD SILO WITH 3 SYNCs
3371 020176 104413 000053      DATACLK, 53          ; AND THE CHARACTER 377
3372 020202 104414 000002      TIMER, 2          ; SINGLE CLOCK THE DATA
3373 020206 104412          ROMCLK          ; WAIT FOR INRDY
                                     ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304

```

```

3374 020210 021244          021244          :PORT4+LU 12
3375 020212 016104 000004      MOV      4(R1),R4      :PUT "FOUND" IN R4
3376 020216 042704 000357      BIC      #357,R4      :CLEAR UNWANTED BITS
3377 020222 012705 000020      MOV      #BIT4,R5      :PUT "EXPECTED" IN R5
3378 020226 120504          CMPB      R5,R4      :IS INRDY SET?
3379 020230 001401          BEQ      1$
3380 020232 104040          ERROR      40      :ERROR, INRDY IS NOT SET
3381 020234          1$:
3382 020234 104412          ROMCLK          :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3383 020236 021204          021204          :PORT4+IN DATA
3384 020240 016104 000004      MOV      4(R1),R4      :PUT "FOUND" IN R4
3385 020244 012705 000377      MOV      #377,R5      :PUT "EXPECTED" IN R5
3386 020250 120504          CMPB      R5,R4      :WAS A 377 RECEIVED?
3387 020252 001401          BEQ      2$
3388 020254 104010          ERROR      10      :ERROR, RECEIVED DATA IS WRONG
3389 020256          2$:
3390
3391
3392          :***** TEST 35 *****
3393          :*DDCMP DATA TEST
3394          :*THIS TEST SINGLE STEPS A BINARY COUNT PATTERN
3395          :*CHECKING EACH CHARACTER AS IT IS RECEIVED
3396          :*****
3397
3398          : TEST 35
3399          :-----
3400          :*****
3401 020256 000004          1$T35: SCOPE          :*****
3402 020260 012737 000035 001202      MOV      #35,$TSTNM      : LOAD THE NO. OF THIS TEST
3403 020266 012737 020406 001442      MOV      #T36,NEXT      : POINT TO THE START OF NEXT TEST.
3404          :R1 CONTAINS BASE KMC11 ADDRESS
3405 020274 104410          MSTCLR          :MASTER CLEAR KMC11
3406 020276 005037 030610      CLR      SCHAR          :START BINARY COUNT AT ZERO
3407 020302 005037 030612      CLR      STUFLG          :CLEAR BITSTUFF FLAG
3408 020306 005002          CLR      R2              :R2 IS "EXPECTED" DATA
3409 020310 012703 000073      MOV      #73,R3          :R3 IS CHARACTER COUNT
3410 020314 012711 004000      MOV      #BIT11,(R1)      :SET LINE UNIT LOOP
3411 020320 004737 030350      JSR      PC,SILOLD          :LOAD SILO WITH COUNT PATTERN
3412 020324 104413 000043          DATACLK, 43          :SYNC RECEIVER AND GET IT ACTIVE
3413 020330 104413 000730          DATACLK, 730          :CLOCK IN 73 CHARACTERS
3414 020334 004737 030614          1$: JSR      PC,INRDY      :WAIT FOR INRDY
3415 020340 104412          4$: ROMCLK          :NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3416 020342 021204          021204          :PORT4+IN DATA
3417 020344 016104 000004      MOV      4(R1),R4      :PUT "FOUND" IN R4
3418 020350 010205          MOV      R2,R5          :PUT "EXPECTED" IN R5
3419 020352 120504          CMPB      R5,R4          :IS DATA CORRECT?
3420 020354 001401          BEQ      2$          :BR IF YES
3421 020356 104010          ERROR      10          :DATA ERROR
3422 020360 005202          2$: INC      R2          :NEXT CHARACTER
3423 020362 022702 000400      CMP      #400,R2          :ALL DONE?
3424 020366 001407          BEQ      3$          :BR IF YES
3425 020370 005303          DEC      R3          :DECREMENT CHARACTER COUNT
3426 020372 001360          BNE      4$          :BR IF SILO NOT EMPTY
3427 020374 004737 030350      JSR      PC,SILOLD          :LOAD SILO WITH MORE OF COUNT PATTERN
3428 020400 012703 000073      MOV      #73,R3          :RELOAD CHARACTER COUNT
3429 020404 000751          BR      1$          :CONTINUE

```

K07

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 67
 DZKCE.P11 12-MAY-77 12:23 BASIC RECEIVER TESTS

PAGE: 0069

3430 020406

35:

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

; TEST 36

```
*****
1ST36: SCOPE
MOV #36,STSTNM ; LOAD THE NO. OF THIS TEST
MOV #TST37,NEXT ; POINT TO THE START OF NEXT TEST.
; R1 CONTAINS BASE KMC11 ADDRESS
MSTCLR ; MASTER CLEAR KMC11
CLR SCHAR ; START BINARY COUNT AT ZERO
CLR STUFLG ; CLEAR BITSTUFF FLAG
CLR R2 ; R2 IS "EXPECTED" DATA
MOV #73,R3 ; R3 IS CHARACTER COUNT
CLR (R1) ; CLEAR LU LOOP IN MAINT REG
MOV #BITS,4(R1) ; LOAD PORT4
ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
122112 ; SET LU LOOP IN LU REG 12
JSR PC,SILOLD ; LOAD SILO WITH COUNT PATTERN
DATACLK, 43 ; SYNC RECEIVER AND GET IT ACTIVE
15: DATACLK, 730 ; CLOCK IN 73 CHARACTERS
45: JSR PC,INRDY ; WAIT FOR INRDY
ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
021204 ; PORT4+IN DATA
MOV 4(R1),R4 ; PUT "FOUND" IN R4
MOV R2,R5 ; PUT "EXPECTED" IN R5
CMPB R5,R4 ; IS DATA CORRECT?
BEQ 25 ; BR IF YES
ERROR 10 ; DATA ERROR
25: INC R2 ; NEXT CHARACTER
CMP #400,R2 ; ALL DONE?
BEQ 35 ; BR IF YES
DEC R3 ; DECREMENT CHARACTER COUNT
BNE 45 ; BR IF SILO NOT EMPTY
JSR PC,SILOLD ; LOAD SILO WITH MORE OF COUNT PATTERN
MOV #73,R3 ; RELOAD CHARACTER COUNT
35: BR 15 ; CONTINUE
```

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

3431
3432
3433
3434
3435
3436
3437
3438
3439
3440
3441
3442
3443
3444 020406 000004
3445 020410 012737 000036 001202
3446 020416 012737 020546 001442
3447
3448 020424 104410
3449 020426 005037 03061J
3450 020432 005037 030612
3451 020436 005002
3452 020440 012703 000073
3453 020444 005011
3454 020446 012761 000040 000004
3455 020454 104412
3456 020456 122112
3457 020460 004737 030350
3458 020464 104413 000043
3459 020470 104413 000730
3460 020474 004737 030614
3461 020500 104412
3462 020502 021204
3463 020504 016104 000004
3464 020510 010205
3465 020512 120504
3466 020514 001401
3467 020516 104010
3468 020520 005202
3469 020522 022702 000400
3470 020526 001407
3471 020530 005303
3472 020532 001360
3473 020534 004737 030350
3474 020540 012703 000073
3475 020544 000751
3476 020546
3477
3478
3479
3480
3481
3482
3483
3484
3485

```

3486
3487
3488
3489
3490 020546 000004
3491 020550 012737 000037 001202
3492 020556 012737 020706 001442
3493
3494 020564 104410
3495 020566 012711 004000
3496 020572 004737 030172
3497 020576 000301
3498 020600 104413 000073
3499 020604 004737 030614
3500 020610 104412
3501 020612 021204
3502 020614 016104 000004
3503 020620 012705 000301
3504 020624 120504
3505 020626 001401
3506 020630 104010
3507 020632 004737 030614
3508 020636 104412
3509 020640 021204
3510 020642 016104 000004
3511 020646 012705 000377
3512 020652 120504
3513 020654 001401
3514 020656 104010
3515 020660 004737 030614
3516 020664 104412
3517 020666 021204
3518 020670 016104 000004
3519 020674 012705 000377
3520 020700 120504
3521 020702 001401
3522 020704 104010
3523 020706
3524
3525
3526
3527
3528
3529
3530
3531
3532
3533
3534
3535
3536 020706 000004
3537 020710 012737 000040 001202
3538 020716 012737 021104 001442
3539
3540 020724 104410
3541 020726 032737 020000 002050

; TEST 37
;-----
;*****
;TST37: SCOPE
;MOV #37,$TSTNM ; LOAD THE NO. OF THIS TEST
;MOV #TST40,NEXT ; POINT TO THE START OF NEXT TEST.
;R1 CONTAINS BASE KMC11 ADDRESS
;MASTER CLEAR KMC11
;SET LINE UNIT LOOP
;LOAD SILO WITH 3 SYNCs
;AND A 301
;CLOCK THE 301 IN AND 20 EXTRA TICKS
;WAIT FOR INRDY
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4+IN DATA
;PUT "FOUND" IN R4
;PUT "EXPECTED" IN R5
;WAS A 301 RECEIVED?
;
;ERROR FIRST CHARACTER INCORRECT
;WAIT FOR INRDY
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4+IN DATA
;PUT "FOUND" IN R4
;PUT "EXPECTED" IN R5
;WAS A 377 RECEIVED?
;
;ERROR, 377 WAS NOT RECEIVED
;WAIT FOR INRDY
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;PORT4+IN DATA
;PUT "FOUND" IN R4
;PUT "EXPECTED" IN R5
;WAS A 377 RECEIVED?
;
;ERROR, 177 WAS NOT RECEIVED

;***** TEST 40 *****
;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
;*****

; TEST 40
;-----
;*****
;TST40: SCOPE
;MOV #40,$TSTNM ; LOAD THE NO. OF THIS TEST
;MOV #TST41,NEXT ; POINT TO THE START OF NEXT TEST.
;R1 CONTAINS BASE KMC11 ADDRESS
;MASTER CLEAR KMC11
;IS LINE UNIT MRDY?

```

3542	020734	001004			BNE	.+12		: BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
3543	020736	032737	040000	002050	BIT	#BIT14,STAT1		: IS TURNAROUND CONNECTOR ON?
3544	020744	001457			BEQ	2\$		: SKIP TEST IF NO
3545	020746	005011			CLR	(R1)		: CLEAR LINE UNIT LOOP
3546	020750	012761	000100	000004	MOV	#100,4(R1)		: LOAD PORT4
3547	020756	104412			ROMCLK			: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3548	020760	122113			122113			: SET DTR
3549	020762	104414	000002		TIMER	2		: WAIT
3550	020766	104412			ROMCLK			: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3551	020770	021264			021264			: PORT4+LUI3
3552	020772	016104	000004		MOV	4(R1),R4		: PUT "FOUND" IN R4
3553	020776	042704	000023		BIC	#23,R4		: CLEAR UNWANTED BITS
3554	021002	012705	000310		MOV	#310,R5		: PUT "EXPECTED" IN R5
3555	021006	032737	020000	002050	BIT	#BIT13,STAT1		: IS LINE UNIT M8202?
3556	021014	001402			BEQ	.+6		: BR IF NO
3557	021016	042705	000200		BIC	#BIT7,R5		: NO RING ON M8202
3558	021022	120504			CMPB	R5,R4		: ARE RING AND MRDY SET?
3559	021024	001401			BEQ	1\$		
3560	021026	104011			ERROR	11		: ERROR, RING OR MRDY NOT SET
3561	021030	005061	000004		CLR	4(R1)		: CLEAR PORT4
3562	021034	104412			ROMCLK			: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3563	021036	122113			122113			: CLEAR DTR
3564	021040	104414	000002		TIMER	2		
3565	021044	104412			ROMCLK			: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3566	021046	021264			021264			: PORT4+LUI3
3567	021050	016104	000004		MOV	4(R1),R4		: PUT "FOUND" IN R4
3568	021054	042704	000023		BIC	#23,R4		: CLEAR UNWANTED BITS
3569	021060	005005			CLR	R5		: PUT "EXPECTED" IN R5
3570	021062	032737	020000	002050	BIT	#BIT13,STAT1		: IS LINE UNIT M8202?
3571	021070	001402			BEQ	.+6		: BR IF NO
3572	021072	052705	000010		BIS	#BIT3,R5		: MRDY SET ON M8202
3573	021076	120504			CMPB	R5,R4		: ARE RING AND MRDY CLEAR?
3574	021100	001401			BEQ	2\$		
3575	021102	104011			ERROR	11		: ERROR, RING OR MRDY NOT CLEAR
3576	021104							
3577								
3578								
3579								
3580								
3581								
3582								
3583								
3584								
3585								
3586								
3587								
3588	021104	000004						
3589	021106	012737	000041	001202	↑ST41: SCOPE			
3590	021114	012737	021264	001442	MOV	#41,STSTNM		: LOAD THE NO. OF THIS TEST
3591					MOV	#TST42,NEXT		: POINT TO THE START OF NEXT TEST.
3592	021122	104410			MSTCLR			: R1 CONTAINS BASE KMC11 ADDRESS
3593	021124	032737	020000	002050	BIT	#BIT13,STAT1		: MASTER CLEAR KMC11
3594	021132	001004			BNE	.+12		: IS LINE UNIT M8202?
3595	021134	032737	040000	002050	BIT	#BIT14,STAT1		: BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
3596	021142	001450			BEQ	1\$		: IS TURNAROUND CONNECTOR ON?
3597	021144	012711	004000		MOV	#BIT11,(R1)		: SKIP TEST IF NO
								: SET LINE UNIT LOOP

***** TEST 41 *****

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

; TEST 41

;-----

;*****

;ST41:

SCOPE

 MOV #41,STSTNM
 MOV #TST42,NEXT

 ; LOAD THE NO. OF THIS TEST
 ; POINT TO THE START OF NEXT TEST.
 ; R1 CONTAINS BASE KMC11 ADDRESS
 ; MASTER CLEAR KMC11
 ; IS LINE UNIT M8202?
 ; BR IF YES (DO TEST EVEN IF NO LOOP-BACK CONN)
 ; IS TURNAROUND CONNECTOR ON?
 ; SKIP TEST IF NO
 ; SET LINE UNIT LOOP

```

3598 021150 012761 000100 000004      MOV      #100, 4(R1)      ; LOAD PORT4
3599 021156 104412      ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3600 021160 122113      122113      ; CLEAR ALL MODEM SIGNALS, EXCEPT DTR
3601 021162 104414 000002      TIMER, 2      ; WAIT
3602 021166 012761 000001 000004      MOV      #1, 4(R1)      ; LOAD PORT4
3603 021174 104412      ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3604 021176 122111      122111      ; SET SOM
3605 021200 004537 031256      JSR      RS, MESLD      ; FILL OUT DATA SILO
3606 021204 031540      MESDAT      ; WITH 64 CHARACTERS
3607 021206 000100      64.
3608 021210 012700 000050      MOV      #50, R0      ; PREPARE FOR DELAY
3609 021214 005011      CLR      (R1)      ; CLEAR LINE UNIT LOOP
3610 021216
3611 021216 104412      ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3612 021220 021264      021264      ; PORT4+LUI3
3613 021222 016104 000004      MOV      4(R1), R4      ; PUT "FOUND" IN R4
3614 021226 042704 000023      BIC      #23, R4      ; CLEAR UNWANTED BITS
3615 021232 012705 000354      MOV      #354, R5      ; PUT "EXPECTED" IN R5
3616 021236 032737 020000 002050      BIT      #BIT13, STAT1      ; IS LINE UNIT MB202?
3617 021244 001402      BEQ      .+6      ; BR IF NO
3618 021246 042705 000200      BIC      #BIT7, R5      ; NO RING ON MB202
3619 021252 120504      CMPB      R5, R4      ; COMPARE EXPECTED AND FOUND
3620 021254 001403      BEQ      1$      ; BR IF OK
3621 021256 005300      DEC      R0      ; DEC DELAY COUNT
3622 021260 001356      BNE      2$      ; BR IF NOT ZERO
3623 021262 104011      ERROR      11      ; ERROR, ALL SIGNALS ARE NOT SET
3624 021264
3625
3626
3627 ;***** TEST 42 *****
3628 ;*TEST OF CRC OPERATION
3629 ;*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
3630 ;*0, VERIFY THE LSB OF THE BCC ON EACH SHIFT
3631 ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
3632 ;*****
3633
3634 ; TEST 42
3635 ;-----
3636 ;*****
3637 021264 000004      1$T42: SCOPE
3638 021266 012737 000042 001202      MOV      #42, $TSTNM      ; LOAD THE NO. OF THIS TEST
3639 021274 012737 021600 001442      MOV      #TST43, NEXT      ; POINT TO THE START OF NEXT TEST.
3640 021302 012737 021316 001444      MOV      #64$, LOCK      ; ADDRESS FOR LOCK ON DATA.
3641
3642 021310 104410      MSTCLR      ; R1 CONTAINS BASE KMC11 ADDRESS
3643 021312 012711 004000      MOV      #BIT11, (R1)      ; MASTER CLEAR KMC11
3644 021316 004737 031320      JSR      PC, CLR10      ; SET LU LOOP
3645 021322 005000      CLR      R0      ; CLEAR BCC REGISTERS
3646 021324 012737 120001 030754      MOV      #CRC16, XPOLY      ; START SHIFT COUNTER AT ZERO
3647 021332 012737 000000 021372      MOV      #0, 66$      ; LOAD POLYNOMIAL FOR SOFTWARE BCC
3648 021340 005037 021374      CLR      67$      ; LOAD CHAR FOR SOFTWARE BCC
3649 021344 004737 030760      JSR      PC, BCCLD      ; CLEAR OLD SOFTWARE BCC
3650 021350 000000      0      ; LOAD OUT SILO WITH 2 SYNCs
3651 021352 104413 000021      DATACLK, 21      ; AND THE CHARACTER 0
3652 021356 104413 000001      DATACLK, 1      ; GET TRANSMITTER ACTIVE
3653 021362 005200      INC      R0      ; SHIFT BCC ONCE
; BUMP SHIFT COUNT

```

3654	021364	004537	030650		JSR	RS, SIMBCC	: CALCULATE SOFTWARE BCC LSB
3655	021370	000001			1		: ONE SHIFT
3656	021372	000000		66\$:	0		: DATA CHARACTER
3657	021374	000000		67\$:	0		: OLD BCC
3658	021376	103405			BCS	68\$	: BR IF SOFT BCC LSB IS SET
3659	021400	004737	031072		JSR	PC, GET00	: GET HARDWARE TRANSMITTER BCC LSB
3660	021404	103006			BCC	69\$	: BR IF HARD BCC LSB IS CLEAR
3661	021406	104012			ERROR	12	: ERROR, BCC LSB IS SET
3662	021410	000404			BR	69\$	: CONTINUE
3663	021412	004737	031072	68\$:	JSR	PC, GET00	: GET HARDWARE TRANSMITTER BCC LSB
3664	021416	103401			BCS	69\$	: BR IF HARD BCC LSB IS SET
3665	021420	104016			ERROR	16	: ERROR, HARD BCC LSB IS CLEAR
3666	021422			69\$:			
3667	021422	006037	021372		ROR	66\$	: SHIFT SOFT DATA
3668	021426	013737	030756	021374	MOV	CALBCC, 67\$	: LOAD OLD SOFT BCC
3669	021434	022700	000010		CMP	#10, R0	: DONE YET?
3670	021440	001346			BNE	65\$	: BR IF NOT DONE
3671	021442	104405			SCOPI		: SCOPE SUBTEST (SW09=1)
3672	021444	012737	021452	001444	MOV	#71\$, LOCK	: NEW SCOPE1
3673	021452	004737	031320	71\$:	JSR	PC, CLR10	: CLEAR BCC REGISTERS
3674	021456	005000			CLR	R0	: START SHIFT COUNTER AT ZERO
3675	021460	012737	120001	030754	MOV	#CRC16, XPOLY	: LOAD POLYNOMIAL FOR SOFTWARE BCC
3676	021466	012737	000000	021526	MOV	#0, 73\$	: LOAD CHAR FOR SOFTWARE BCC
3677	021474	005037	021530		CLR	74\$	: CLEAR OLD SOFTWARE BCC
3678	021500	004737	030760		JSR	PC, BCCLD	: LOAD OUT SILO WITH 2 SYNCs
3679	021504	000000			0		: AND THE CHARACTER 0
3680	021506	104413	000032		DATACLK,	32	: GET RECEIVER ACTIVE
3681	021512	104413	000001	72\$:	DATACLK,	1	: SHIFT BCC ONCE
3682	021516	005200			INC	R0	: BUMP SHIFT COUNT
3683	021520	004537	030650		JSR	RS, SIMBCC	: CALCULATE SOFTWARE BCC LSB
3684	021524	000001			1		: ONE SHIFT
3685	021526	000000		73\$:	0		: DATA CHARACTER
3686	021530	000000		74\$:	0		: OLD BCC
3687	021532	103405			BCS	75\$	: BR IF SOFT BCC LSB IS SET
3688	021534	004737	031104		JSR	PC, GET01	: GET HARDWARE RECEIVER BCC LSB
3689	021540	103006			BCC	76\$	: BR IF HARD BCC LSB IS CLEAR
3690	021542	104013			ERROR	13	: ERROR, BCC LSB IS SET
3691	021544	000404			BR	76\$	: CONTINUE
3692	021546	004737	031104	75\$:	JSR	PC, GET01	: GET HARDWARE RECEIVER BCC LSB
3693	021552	103401			BCS	76\$	: BR IF HARD BCC LSB IS SET
3694	021554	104017			ERROR	17	: ERROR, BCC LSB IS CLEAR
3695	021556			76\$:			
3696	021556	006037	021526		ROR	73\$	: SHIFT SOFT DATA
3697	021562	013737	030756	021530	MOV	CALBCC, 74\$	: LOAD OLD SOFT BCC
3698	021570	022700	000010		CMP	#10, R0	: DONE YET?
3699	021574	001346			BNE	72\$	: BR IF NOT DONE
3700	021576	104405			SCOPI		: SCOPE SUBTEST (SW09=1)
3701	021600			77\$:			

```

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

```

 3702
 3703
 3704
 3705
 3706
 3707
 3708
 3709

```

3710
3711
3712
3713
3714 021600 000004
3715 021602 012737 000043 001202
3716 021610 012737 022114 001442
3717 021616 012737 021632 001444
3718
3719 021624 104410
3720 021626 012711 004000
3721 021632 004737 031320
3722 021636 005000
3723 021640 012737 120001 030754
3724 021646 012737 000377 021706
3725 021654 005037 021710
3726 021660 004737 030760
3727 021664 000377
3728 021666 104413 000021
3729 021672 104413 000701
3730 021676 005200
3731 021700 004537 030650
3732 021704 000001
3733 021706 000000
3734 021710 000000
3735 021712 103405
3736 021714 004737 031072
3737 021720 103006
3738 021722 104012
3739 021724 000404
3740 021726 004737 031072
3741 021732 103401
3742 021734 104016
3743 021736
3744 021736 006037 021706
3745 021742 013737 030756 021710
3746 021750 022700 000010
3747 021754 001346
3748 021756 104405
3749 021760 012737 021766 001444
3750 021766 004737 031320
3751 021772 005000
3752 021774 012737 120001 030754
3753 022002 012737 000377 022042
3754 022010 005037 022044
3755 022014 004737 030760
3756 022020 000377
3757 022022 104413 000032
3758 022026 104413 000001
3759 022032 005200
3760 022034 004537 030650
3761 022040 000001
3762 022042 000000
3763 022044 000000
3764 022046 103405
3765 022050 004737 031104

; TEST 43
;*****
;ST43: SCOPE
;MOV #43,STSTNM ; LOAD THE NO. OF THIS TEST
;MOV #ST44,NEXT ; POINT TO THE START OF NEXT TEST.
;MOV #64$,LOCK ; ADDRESS FOR LOCK ON DATA.
;R1 CONTAINS BASE KMC11 ADDRESS
;MASTER CLEAR KMC11
;SET LU LOOP
;CLEAR BCC REGISTERS
;START SHIFT COUNTER AT ZERO
;LOAD POLYNOMIAL FOR SOFTWARE BCC
;LOAD CHAR FOR SOFTWARE BCC
;CLEAR OLD SOFTWARE BCC
;LOAD OUT SILO WITH 2 SYNCs
;AND THE CHARACTER 377
;GET TRANSMITTER ACTIVE
;SHIFT BCC ONCE
;BUMP SHIFT COUNT
;CALCULATE SOFTWARE BCC LSB
;ONE SHIFT
;DATA CHARACTER
;OLD BCC
;BR IF SOFT BCC LSB IS SET
;GET HARDWARE TRANSMITTER BCC LSB
;BR IF HARD BCC LSB IS CLEAR
;ERROR, BCC LSB IS SET
;CONTINUE
;GET HARDWARE TRANSMITTER BCC LSB
;BR IF HARD BCC LSB IS SET
;ERROR, HARD BCC LSB IS CLEAR

;SHIFT SOFT DATA
;LOAD OLD SOFT BCC
;DONE YET?
;BR IF NOT DONE
;SCOPE SUBTEST (SW09=1)
;NEW SCOPE1
;CLEAR BCC REGISTERS
;START SHIFT COUNTER AT ZERO
;LOAD POLYNOMIAL FOR SOFTWARE BCC
;LOAD CHAR FOR SOFTWARE BCC
;CLEAR OLD SOFTWARE BCC
;LOAD OUT SILO WITH 2 SYNCs
;AND THE CHARACTER 377
;GET RECEIVER ACTIVE
;SHIFT BCC ONCE
;BUMP SHIFT COUNT
;CALCULATE SOFTWARE BCC LSB
;ONE SHIFT
;DATA CHARACTER
;OLD BCC
;BR IF SOFT BCC LSB IS SET
;GET HARDWARE RECEIVER BCC LSB

64$: JSR PC,CLR10
CLR RO
MOV #CRC16,XPOLY
MOV #377,66$;
CLR 67$
JSR PC,BCCLD
377
DATACLK, 21
65$: DATACLK, 1
INC RO
JSR RS,SIMBCC
1
66$: 0
67$: 0
BCS 68$
JSR PC,GETQ0
BCC 69$
ERROR 12
BR 69$
68$: JSR PC,GETQ0
BCS 69$
ERROR 16
69$: ROR 66$
MOV CALBCC,67$
CMP #10,RO
BNE 65$
SCOPE1
MOV #71$,LOCK
71$: JSR PC,CLR10
CLR RO
MOV #CRC16,XPOLY
MOV #377,73$;
CLR 74$
JSR PC,BCCLD
377
DATACLK, 32
72$: DATACLK, 1
INC RO
JSR RS,SIMBCC
1
73$: 0
74$: 0
BCS 75$
JSR PC,GETQ1

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 73
 DZKCE.P11 12-MAY-77 12:23 BASIC RECEIVER TESTS

PAGE: 0094

```

3766 022054 103006          BCC 76$          ;BR IF HARD BCC LSB IS CLEAR
3767 022056 104013          ERROR 13          ;ERROR, BCC LSB IS SET
3768 022060 000404          BR 76$          ;CONTINUE
3769 022062 004737 031104 75$: JSR PC,GETQI    ;GET HARDWARE RECEIVER BCC LSB
3770 022066 103401          BCS 76$          ;BR IF HARD BCC LSB IS SET
3771 022070 104017          ERROR 17          ;ERROR, BCC LSB IS CLEAR
3772 022072          76$:          ;
3773 022072 006037 022042 76$: ROR 73$          ;SHIFT SOFT DATA
3774 022076 013737 030756 022044 MOV CALBCC,74$ ;LOAD OLD SOFT BCC
3775 022104 022700 000010 CMP #10,R0 ;DONE YET?
3776 022110 001346 BNE 72$          ;BR IF NOT DONE
3777 022112 104405 SCOP1          ;SCOPE SUBTEST (SW09=1)
3778 022114 77$:          ;
3779          ;
3780          ;
3781          ;***** TEST 44 *****
3782          ;*TEST OF CRC OPERATION
3783          ;*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK THE CHARACTER
3784          ;*125, VERIFY THE LSB OF THE BCC ON EACH SHIFT
3785          ;*TEST TRANSMITTER FIRST THEN THE RECEIVER BCC
3786          ;*****
3787          ;
3788          ; TEST 44
3789          ;-----
3790          ;*****
3791 022114 000004          ;ST44: SCOPE
3792 022116 012737 000044 001202 MOV #44,STSTNM ; LOAD THE NO. OF THIS TEST
3793 022124 012737 022430 001442 MOV #ST45,NEXT ; POINT TO THE START OF NEXT TEST.
3794 022132 012737 022146 001444 MOV #64$,LOCK ; ADDRESS FOR LOCK ON DATA.
3795          ;
3796 022140 104410          ;R1 CONTAINS BASE KMC11 ADDRESS
3797 022142 012711 004000 MSTCLR ;MASTER CLEAR KMC11
3798 022146 004737 031320 64$: JSR PC,CLR10 ;SET LU LOOP
3799 022152 005000 CLR R0 ;CLEAR BCC REGISTERS
3800 022154 012737 120001 030754 MOV #CRC16,XPOLY ;START SHIFT COUNTER AT ZERO
3801 022162 012737 000125 022222 MOV #125,66$ ;LOAD POLYNOMIAL FOR SOFTWARE BCC
3802 022170 005037 022224 CLR 67$ ;LOAD CHAR FOR SOFTWARE BCC
3803 022174 004737 030760 JSR PC,BCCLD ;CLEAR OLD SOFTWARE BCC
3804 022200 000125 125 ;LOAD OUT SILO WITH 2 SYNC'S
3805 022202 104413 000021 DATACLK, 21 ;AND THE CHARACTER 125
3806 022206 104413 000001 65$: DATACLK, 1 ;GET TRANSMITTER ACTIVE
3807 022212 005200 INC R0 ;SHIFT BCC ONCE
3808 022214 004537 030650 JSR RS,SIMBCC ;BUMP SHIFT COUNT
3809 022220 000001 1 ;CALCULATE SOFTWARE BCC LSB
3810 022222 000000 66$: 0 ;ONE SHIFT
3811 022224 000000 67$: 0 ;DATA CHARACTER
3812 022226 103405 BCS 68$ ;OLD BCC
3813 022230 004737 031072 JSR PC,GETQ0 ;BR IF SOFT BCC LSB IS SET
3814 022234 103006 BCS 69$ ;GET HARDWARE TRANSMITTER BCC LSB
3815 022236 104012 ERPOR 12 ;BR IF HARD BCC LSB IS CLEAR
3816 022240 000404 BR 69$ ;ERROR, BCC LSB IS SET
3817 022242 004737 031072 68$: JSR PC,GETQ0 ;CONTINUE
3818 022246 103401 BCS 69$ ;GET HARDWARE TRANSMITTER BCC LSB
3819 022250 104016 ERROR 16 ;BR IF HARD BCC LSB IS SET
3820 022252 69$:          ;ERROR, HARD BCC LSB IS CLEAR
3821 022252 006037 022222 ROR 66$ ;SHIFT SOFT DATA

```

3822	022256	013737	030756	022224	MOV	CALBCC,67\$	:LOAD OLD SOFT BCC
3823	022264	022700	000010		CMP	#10,R0	:DONE YET?
3824	022270	001346			BNE	65\$	:BR IF NOT DONE
3825	022272	104405			SCOPI		:SCOPE SUBTEST (SM09=1)
3826	022274	012737	022302	001444	MOV	#71\$,LOCK	:NEW SCOPE1
3827	022302	004737	031320	71\$:	JSR	PC,CLRI0	:CLEAR BCC REGISTERS
3828	022306	005000			CLR	R0	:START SHIFT COUNTER AT ZERO
3829	022310	012737	120001	030754	MOV	#CRC16,XPOLY	:LOAD POLYNOMIAL FOR SOFTWARE BCC
3830	022316	012737	000125	022356	MOV	#125,73\$;	:LOAD CHAR FOR SOFTWARE BCC
3831	022324	005037	022360		CLR	74\$	:CLEAR OLD SOFTWARE BCC
3832	022330	004737	030760		JSR	PC,BCCLD	:LOAD OUT SILO WITH 2 SYNC
3833	022334	000125			125		:AND THE CHARACTER 125
3834	022336	104413	000032		DATACLK,	32	:GET RECEIVER ACTIVE
3835	022342	104413	000001	72\$:	DATACLK,	1	:SHIFT BCC ONCE
3836	022346	005200			INC	R0	:BUMP SHIFT COUNT
3837	022350	004537	030650		JSR	RS,SIMBCC	:CALCULATE SOFTWARE BCC LSB
3838	022354	000001			1		:ONE SHIFT
3839	022356	000000		73\$:	0		:DATA CHARACTER
3840	022360	000000		74\$:	0		:OLD BCC
3841	022362	103405			BCS	75\$	:BR IF SOFT BCC LSB IS SET
3842	022364	004737	031104		JSR	PC,GETQI	:GET HARDWARE RECEIVER BCC LSB
3843	022370	103006			BCC	76\$	:BR IF HARD BCC LSB IS CLEAR
3844	022372	104013			ERROR	13	:ERROR, BCC LSB IS SET
3845	022374	000404			BR	76\$	:CONTINUE
3846	022376	004737	031104	75\$:	JSR	PC,GETQI	:GET HARDWARE RECEIVER BCC LSB
3847	022402	103401			BCS	76\$	:BR IF HARD BCC LSB IS SET
3848	022404	104017			ERROR	17	:ERROR, BCC LSB IS CLEAR
3849	022406			76\$:			
3850	022406	006037	022356		ROR	73\$	:SHIFT SOFT DATA
3851	022412	013737	030756	022360	MOV	CALBCC,74\$	:LOAD OLD SOFT BCC
3852	022420	022700	000010		CMP	#10,R0	:DONE YET?
3853	022424	001346			BNE	72\$	:BR IF NOT DONE
3854	022426	104405			SCOPI		:SCOPE SUBTEST (SM09=1)
3855	022430			77\$:			

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

: TEST 45

 :-----
 :*****
 :TEST45: SCOPE
 :MOV #45,\$TSTNM ; LOAD THE NO. OF THIS TEST
 :MOV #TST46,NEXT ; POINT TO THE START OF NEXT TEST.
 :MOV #64\$,LOCK ; ADDRESS FOR LOCK ON DATA.
 :R1 CONTAINS BASE KMC11 ADDRESS
 :MASTER CLEAR KMC11
 :SET LU LOOP
 :CLEAR BCC REGISTERS
 :START SHIFT COUNTER AT ZERO
 :LOAD POLYNOMIAL FOR SOFTWARE BCC

3878	022476	012737	000252	022536	MOV	#252,66\$;	: LOAD CHAR FOR SOFTWARE BCC
3879	022504	005037	022540		CLR	67\$	: CLEAR OLD SOFTWARE BCC
3880	022510	004737	030760		JSR	PC,BCCLD	: LOAD OUT SILO WITH 2 SYNC
3881	022514	000252			252		: AND THE CHARACTER 252
3882	022516	104413	000021		DATACLK,	21	: GET TRANSMITTER ACTIVE
3883	022522	104413	000001	65\$:	DATACLK,	1	: SHIFT BCC ONCE
3884	022526	005200			INC	RO	: BUMP SHIFT COUNT
3885	022530	004537	030650		JSR	RS,SIMBCC	: CALCULATE SOFTWARE BCC LSB
3886	022534	000001			1		: ONE SHIFT
3887	022536	000000		66\$:	0		: DATA CHARACTER
3888	022540	000000		67\$:	0		: OLD BCC
3889	022542	103405			BCS	68\$	: BR IF SOFT BCC LSB IS SET
3890	022544	004737	031072		JSR	PC,GETQ0	: GET HARDWARE TRANSMITTER BCC LSB
3891	022550	103006			BCC	69\$	: BR IF HARD BCC LSB IS CLEAR
3892	022552	104012			ERROR	12	: ERROR, BCC LSB IS SET
3893	022554	000404			BR	69\$	: CONTINUE
3894	022556	004737	031072	68\$:	JSR	PC,GETQ0	: GET HARDWARE TRANSMITTER BCC LSB
3895	022562	103401			BCS	69\$	: BR IF HARD BCC LSB IS SET
3896	022564	104016			ERROR	16	: ERROR, HARD BCC LSB IS CLEAR
3897	022566			69\$:			
3898	022566	006037	022536		ROR	66\$	: SHIFT SOFT DATA
3899	022572	013737	030756	022540	MOV	CALBCC,67\$	: LOAD OLD SOFT BCC
3900	022600	022700	000010		CMP	#10,RO	: DONE YET?
3901	022604	001346			BNE	65\$	: BR IF NOT DONE
3902	022606	104405			SCOPI		: SCOPE SUBTEST (SW09=1)
3903	022610	012737	022616	001444	MOV	#71\$,LOCK	: NEW SCOPE1
3904	022616	004737	031320	71\$:	JSR	PC,CLRIO	: CLEAR BCC REGISTERS
3905	022622	005000			CLR	RO	: START SHIFT COUNTER AT ZERO
3906	022624	012737	120001	030754	MOV	#CRC16,XPOLY	: LOAD POLYNOMIAL FOR SOFTWARE BCC
3907	022632	012737	000252	022672	MOV	#252,73\$;	: LOAD CHAR FOR SOFTWARE BCC
3908	022640	005037	022674		CLR	74\$	: CLEAR OLD SOFTWARE BCC
3909	022644	004737	030760		JSR	PC,BCCLD	: LOAD OUT SILO WITH 2 SYNC
3910	022650	000252			252		: AND THE CHARACTER 252
3911	022652	104413	000032		DATACLK,	32	: GET RECEIVER ACTIVE
3912	022656	104413	000001	72\$:	DATACLK,	1	: SHIFT BCC ONCE
3913	022662	005200			INC	RO	: BUMP SHIFT COUNT
3914	022664	004537	030650		JSR	RS,SIMBCC	: CALCULATE SOFTWARE BCC LSB
3915	022670	000001			1		: ONE SHIFT
3916	022672	000000		73\$:	0		: DATA CHARACTER
3917	022674	000000		74\$:	0		: OLD BCC
3918	022676	103405			BCS	75\$	: BR IF SOFT BCC LSB IS SET
3919	022700	004737	031104		JSR	PC,GETQ1	: GET HARDWARE RECEIVER BCC LSB
3920	022704	103006			BCC	76\$	: BR IF HARD BCC LSB IS CLEAR
3921	022706	104013			ERROR	13	: ERROR, BCC LSB IS SET
3922	022710	000404			BR	76\$	: CONTINUE
3923	022712	004737	031104	75\$:	JSR	PC,GETQ1	: GET HARDWARE RECEIVER BCC LSB
3924	022716	103401			BCS	76\$	: BR IF HARD BCC LSB IS SET
3925	022720	104017			ERROR	17	: ERROR, BCC LSB IS CLEAR
3926	022722			76\$:			
3927	022722	006037	022672		ROR	73\$	: SHIFT SOFT DATA
3928	022726	013737	030756	022674	MOV	CALBCC,74\$	: LOAD OLD SOFT BCC
3929	022734	022700	000010		CMP	#10,RO	: DONE YET?
3930	022740	001346			BNE	72\$	: BR IF NOT DONE
3931	022742	104405			SCOPI		: SCOPE SUBTEST (SW09=1)
3932	022744			77\$:			
3933							

```

3934
3935
3936
3937
3938
3939
3940
3941
3942
3943
3944 022744 000004
3945 022746 012737 000046 001202
3946 022754 012737 023202 001442
3947
3948 022762 104410
3949 022764 012711 004000
3950 022770 005003
3951 022772 005004
3952 022774 005005
3953 022776 005037 023100
3954 023002 012737 120001 030754
3955 023010 004737 031122
3956 023014 010461 000004
3957 023020 104412
3958 023022 122110
3959 023024 005204
3960 023026 010461 000004
3961 023032 104412
3962 023034 122110
3963 023036 005204
3964 023040 010461 000004
3965 023044 104412
3966 023046 122110
3967 023050 004737 030006
3968 023054 104413 000021
3969 023060 010537 023076
3970 023064 104413 000001
3971 023070 004537 030650
3972 023074 000001
3973 023076 000000
3974 023100 000000
3975 023102 103405
3976 023104 004737 031072
3977 023110 103006
3978 023112 104020
3979 023114 000404
3980 023116 004737 031072
3981 023122 103401
3982 023124 104021
3983
3984 023126
3985 023126 006037 023076
3986 023132 013737 030756 023100
3987 023140 005203
3988 023142 022703 000010
3989 023146 001346

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

; TEST 46
;-----
;*****
;ST46: SCOPE
;MOV #46,STSTNM ; LOAD THE NO. OF THIS TEST
;MOV #ST47,NEXT ; POINT TO THE START OF NEXT TEST.
;R1 CONTAINS BASE KMC11 ADDRESS
;MASTER CLEAR KMC11
;SET LINE UNIT LOOP
;ZERO BIT COUNT
;R4 CONTAINS CHAR TO BE LOADED IN SILO
;R5 CONTAINS CHAR CURRENTLY BEING SHIFTED OUT
;CLEAR SOFT BCC
;LOAD POLYNOMIAL
;LOAD SILO WITH 2 SYNCs, SOM SET
;PORT4+CHAR
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;LOAD OUT DATA
;INCREMENT TO NEXT CHARACTER
;PORT4+CHAR
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;LOAD OUT DATA
;INCREMENT TO NEXT CHARACTER
;PORT4+CHAR
;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
;LOAD OUT DATA
;WAIT FOR OCOR
;CLOCK DATA
;LOAD CHAR FOR SOFT CRC
;SHIFT BCC ONCE
;CALCULATE SOFT BCC
;SOFT SHIFT COUNT
;SOFT CHARACTER
;OLD SOFT BCC
;BR IF SOFT BCC LSB IS SET
;GET HARDWARE TRANSMITTER BCC LSB
;BR IF OK (CLEARED)
;ERROR, BCC LSB WAS SET
;CONTINUE WITH TEST
;GET HARDWARE TRANSMITTER BCC LSB
;BR IF OK (SET)
;ERROR, BCC LSB WAS CLEAR

1$: MOV R5,3$
2$: DATACLK,1
JSR R5,SIMBCC
1
3$: 0
4$: 0
BCS 5$
JSR PC,GET00
BCC 5$
ERROR 20
BR 6$
5$: JSR PC,GET00
BCS 6$
ERROR 21
6$: ROR 3$
MOV CALBCC,4$
INC R3
CMP #10,R3
BNE 2$

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 77
 DZKCE.P11 12-MAY-77 12:23 BASIC RECEIVER TESTS

PAGE: 0099

```

3990 023150 005003 CLR R3 ;RESTART BIT COUNTER
3991 023152 005204 INC R4 ;INCREMENT DATA FOR SILO
3992 023154 022704 000400 CMP #400,R4 ;DONE BINARY COUNT YET?
3993 023160 003404 BLE 9$ ;BR IF YES
3994 023162 010461 000004 MOV R4,4(R1) ;PORT4+DATA
3995 023166 104412 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3996 023170 122110 ;LOAD OUT DATA
3997 023172 005205 9$: INC R5 ;INCREMENT DATA
3998 023174 022705 000400 CMP #400,R5 ;DONE BINARY PATTERN YET?
3999 023200 001327 BNE 1$ ;BR IF NO
4000 023202 7$:
4001
4002
4003 ;***** TEST 47 *****
4004 ;*RECEIVER CRC TEST
4005 ;*USING THE CRC16 POLYNOMIAL, SINGLE CLOCK A BINARY
4006 ;*COUNT PATTERN, VERIFY THE LSB OF THE RECEIVER BCC ON EACH SHIFT
4007 ;*****
4008
4009 ; TEST 47
4010 ;-----
4011 ;*****
4012 023202 000004 ST47: SCOPE ;*****
4013 023204 012737 000047 001202 MOV #47,STSTNM ; LOAD THE NO. OF THIS TEST
4014 023212 012737 023440 001442 MOV #ST47,ST47 ; POINT TO THE START OF NEXT TEST.
4015
4016 023220 104410 MSTCLR ;R1 CONTAINS BASE KMC11 ADDRESS
4017 023222 012711 004000 MOV #BIT11,(R1) ;MASTER CLEAR KMC11
4018 023226 005003 CLR R3 ;SET LINE UNIT LOOP
4019 023230 005004 CLR R4 ;ZERO BIT COUNT
4020 023232 005005 CLR R5 ;R4 CONTAINS CHAR TO BE LOADED IN SILO
4021 023234 005037 023336 CLR 4$ ;R5 CONTAINS CHAR CURRENTLY BEING SHIFTED OUT
4022 023240 012737 120001 030754 MOV #CRC16,XPOLY ;CLEAR SOFT BCC
4023 023246 004737 031122 JSR PC,SYNCD ;LOAD POLYNOMIAL
4024 023252 010461 000004 MOV R4,4(R1) ;LOAD SILO WITH 2 SYNCS, SOM SET
4025 023256 104412 ROMCLK ;PORT4+CHAR
4026 023260 122110 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4027 023262 005204 INC R4 ;LOAD OUT DATA
4028 023264 010461 000004 MOV R4,4(R1) ;INCREMENT TO NEXT CHARACTER
4029 023270 104412 ROMCLK ;PORT4+CHAR
4030 023272 122110 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4031 023274 005204 INC R4 ;LOAD OUT DATA
4032 023276 010461 000004 MOV R4,4(R1) ;INCREMENT TO NEXT CHARACTER
4033 023302 104412 ROMCLK ;PORT4+CHAR
4034 023304 122110 ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4035 023306 004737 030006 JSR PC,OCOR ;LOAD OUT DATA
4036 023312 104413 000032 DATACLK,32 ;WAIT FOR OCOR
4037 023316 010537 023334 1$: MOV R5,3$ ;CLOCK DATA
4038 023322 104413 000001 2$: DATACLK,1 ;LOAD CHAR FOR SOFT CRC
4039 023326 004537 030650 JSR R5,SIMBCC ;SHIFT BCC ONCE
4040 023332 000001 1 ;CALCULATE SOFT BCC
4041 023334 000000 3$: 0 ;SOFT SHIFT COUNT
4042 023336 000000 4$: 0 ;SOFT CHARACTER
4043 023340 103405 BCS 5$ ;OLD SOFT BCC
4044 023342 004737 031104 JSR PC,GETQI ;BR IF SOFT BCC LSB IS SET
4045 023346 103006 BCC 6$ ;GET HARDWARE RECEIVER BCC LSB
;BR IF OK (CLEARED)

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 78
 DZKCE.P11 12-MAY-77 12:23 BASIC RECEIVER TESTS

PAGE: 0099

```

4046 023350 104022          ERROR 22          ;ERROR, BCC LSB WAS SET
4047 023352 000404          BR 6$           ;CONTINUE WITH TEST
4048 023354 004737 031104    5$: JSR PC,GETQI ;GET HARDWARE RECEIVER BCC LSB
4049 023360 103401          BCS 6$          ;BR IF OK (SET)
4050 023362 104023          ERROR 23         ;ERROR, BCC LSB WAS CLEAR
4051
4052 023364          6$:
4053 023364 006037 023334    ROR 3$          ;SHIFT SOFT DATA
4054 023370 013737 030756 023336    MOV CALBCC,4$ ;LOAD OLD SOFT BCC
4055 023376 005203          INC R3          ;INCREMENT BIT COUNTER
4056 023400 022703 000010    CMP #10,R3     ;DONE A FULL CHARACTER YET?
4057 023404 001346          BNE 2$          ;BR IF NO
4058 023406 005003          CLR R3          ;RESTART BIT COUNTER
4059 023410 005204          INC R4          ;INCREMENT DATA FOR SILO
4060 023412 022704 000400    CMP #400,R4    ;DONE BINARY COUNT YET?
4061 023416 003404          BLE 9$          ;BR IF YES
4062 023420 010461 000004    MOV R4,4(R1)   ;PORT4+DATA
4063 023424 104412          ROMCLK          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4064 023426 122110          122110         ;LOAD OUT DATA
4065 023430 005205          9$: INC R5       ;INCREMENT DATA
4066 023432 022705 000400    CMP #400,R5    ;DONE BINARY PATTERN YET?
4067 023436 001327          BNE 1$          ;BR IF NO
4068 023440          7$:
4069
4070
4071          ;***** TEST 50 *****
4072          ;*TRANSMITTER DOCMP CRC TEST
4073          ;*THIS TEST TRANSMITS A FOUR CHARACTER MESSAGE WITH CRC
4074          ;*BOTH DATA AND THE BCC ARE VERIFIED IN THE BIT
4075          ;*WINDOW. THE FOUR CHARACTERS ARE 0,125,252,377
4076          ;*THE TRANSMITTER IS CHECKED FOR GOING TO A MARK STATE AFTER THE BCC
4077          ;*****
4078
4079          ; TEST 50
4080          ;-----
4081          ;*****
4082 023440 000004          ;*ST50: SCOPE
4083 023442 012737 000050 001202    MOV #50,$ST5NM ; LOAD THE NO. OF THIS TEST
4084 023450 012737 023772 001442    MOV $ST51,NEXT ; POINT TO THE START OF NEXT TEST.
4085
4086 023456 104410          ;R1 CONTAINS BASE KMC11 ADDRESS
4087          MSTCLR ;MASTER CLEAR KMC11
4088
4089          ;LOAD OUT DATA SILO
4090 023460 012711 004000    MOV #BIT11,(R1) ;SET LINE UNIT LOOP
4091 023464 012704 031540    MOV #MESDAT,R4 ;LOAD POINTER TO DATA
4092 023470 005037 023564    CLR 10$       ;CLEAR SOFT BCC
4093 023474 012700 000004    MOV #4,R0      ;LOAD CHARACTER COUNT
4094 023500 004737 031122    JSR PC,SYNLD  ;LOAD 2 SYNCs IN OUT SILO
4095 023504 004737 030140    JSR PC,OUTRDY ;WAIT FOR OUTRDY
4096 023510 004537 031256    JSR R5,MESLD  ;LOAD SILO WITH 4 CHAR MESS
4097 023514 031540          MESDAT          ;ADDRESS OF MESSAGE
4098 023516 000004          4              ;NUMBER OF CHARACTERS
4099 023520 004737 031232    JSR PC,EOM    ;LOAD GARBAGE CHARACTER, WITH EOM SET
4100 023524 004737 030006    JSR PC,OCOR  ;WAIT FOR OCOR
4101 023530 005003          CLR R3          ;CLEAR BIT COUNTER

```

```

4102 023532 104413 000022          DATA CLK, 22          ;CLOCK DATA
4103 023536 112405          12$: MOV (R4)+, R5          ;LOAD R5 WITH CHAR
4104 023540 010502          MOV R5, R2          ;LOAD R2 WITH CHAR
4105
4106          ;CHECK FIRST FOUR CHARACTER MESSAGE
4107          ;IN THE BIT WINDOW (0,125,252,377)
4108
4109 023542 012737 120001 030754      MOV #CRC16, XPOLY      ;LOAD POLYNOMIAL
4110 023550 010537 023562          MOV R5, 67$          ;LOAD SOFT CHAR FOR BCC
4111 023554 004537 030650          JSR R5, SIMBCC          ;CALCULATE SOFT BCC
4112 023560 000010          10          ;SHIFT COUNT
4113 023562 000000          67$: 0          ;CHARACTER
4114 023564 000000          10$: 0          ;OLD BCC
4115 023566 013737 030756 023564      MOV CALBCC, 10$      ;LOAD SOFT BCC FOR NEXT SHIFT
4116 023574 104413 000001          64$: DATA CLK, 1          ;SHIFT DATA IN TO BIT WINDOW
4117 023600 106002          ROR R2          ;SHIFT SOFT DATA
4118 023602 103005          BCC 65$          ;BR IF A SPACE
4119 023604 004737 027754          JSR PC, GETSI          ;LOOK AT BIT WINDOW
4120 023610 103406          BCS 66$          ;BR IF OK (MARK)
4121 023612 104006          ERROR 6          ;ERROR, BIT WINDOW WAS A SPACE
4122 023614 000404          BR 66$          ;CONTINUE
4123 023616 004737 027754          65$: JSR PC, GETSI          ;LOOK AT BIT WINDOW
4124 023622 103001          BCC 66$          ;BR IF OK (SPACE)
4125 023624 104006          ERROR 6          ;ERROR, BIT WINDOW WAS A MARK
4126 023626          66$:
4127 023628 005203          INC R3          ;BUMP BIT COUNTER
4128 023630 022703 000010          CMP #10, R3          ;DONE FULL 8 BITS YET
4129 023634 001357          BNE 64$          ;BR IF N)
4130 023636 005003          CLR R3          ;CLEAR B'T COUNTER
4131 023640 005300          DEC R0          ;DEC CHARACTER COUNT
4132 023642 001335          BNE 12$          ;BR IF NOT DONE YET
4133
4134          ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4135
4136 023644 013700 030756          MOV CALBCC, R0          ;PUT BCC IN R0
4137 023650 104413 000001          68$: DATA CLK, 1          ;SHIFT HARDWARE BCC
4138 023654 006000          ROR R0          ;SHIFT SOFT BCC
4139 023656 103005          BCC 69$          ;BR IF CARRY CLEAR
4140 023660 004737 027754          JSR PC, GETSI          ;LOOK AT BIT WINDOW
4141 023664 103406          BCS 70$          ;BR IF OK (MARK)
4142 023666 104014          ERROR 14          ;ERROR, CRC WRONG (SPACE)
4143 023670 000404          BR 70$          ;CONTINUE
4144 023672 004737 027754          63$: JSR PC, GETSI          ;LOOK AT BIT WINDOW
4145 023676 103001          BCC 70$          ;BR IF OK (SPACE)
4146 023700 104014          ERROR 14          ;ERROR, CRC WRONG (MARK)
4147 023702          70$:
4148 023702 005203          INC R3          ;BUMP BIT COUNTER
4149 023704 022703 000020          CMP #20, R3          ;FINISHED BCC YET?
4150 023710 001357          BNE 68$          ;BR IF NO
4151 023712 005003          CLR R3          ;CLEAR BIT COUNTER
4152
4153          ;CHECK TO SEE IF TRANSMITTER IS MARKING
4154
4155 023714 104413 000001          2$: DATA CLK, 1          ;CLOCK TRANSMITTER
4156 023720 004737 027754          JSR PC, GETSI          ;LOOK AT WINDOW
4157 023724 103401          BCS 3$          ;IT SHOULD BE MARKING

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 80
 DZKCE.P11 12-MAY-77 12:23

PAGE: 0101

BASIC RECEIVER TESTS

4158 023726 104024
 4159 023730 005203
 4160 023732 022703 000007
 4161 023736 001366
 4162 023740 104413 000010
 4163 023744 005003
 4164 023746 104413 000001
 4165 023752 004737 027754
 4166 023756 103401
 4167 023760 104024
 4168 023762 005203
 4169 023764 022703 000020
 4170 023770 001366
 4171 023772
 4172
 4173
 4174
 4175
 4176
 4177
 4178
 4179
 4180
 4181
 4182
 4183
 4184 023772 000004
 4185 023774 012737 000051 001202
 4186 024002 012737 024174 001442
 4187
 4188 024010 104410
 4189 024012 012711 004000
 4190 024016 012702 031540
 4191 024022 012700 000004
 4192 024026 004737 031122
 4193 024032 004737 030140
 4194 024036 004537 031256
 4195 024042 031540
 4196 024044 000004
 4197 024046 004737 031232
 4198 024052 004737 030006
 4199 024056 104413 000114
 4200 024062 004737 030614
 4201 024066 104412
 4202 024070 021204
 4203 024072 016104 000004
 4204 024076 112205
 4205 024100 120504
 4206 024102 001401
 4207 024104 104010
 4208 024106 005300
 4209 024110 001364
 4210
 4211
 4212
 4213 024112 004737 030614

3\$: ERROR 24 ; ERROR, BIT WAS A SPACE
 INC R3 ; BUMP BIT COUNTER
 CMP #7,R3 ; DONE YET
 BNE 25 ; BR IF NO
 DATACLK, 10 ; GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
 CLR R3 ; CLEAR BIT COUNTER
 4\$: DATACLK, 1 ; SHIFT OUT NEXT BIT
 JSR PC,GETSI ; LOOK AT BIT WINDOW
 BCS +4 ; BR IF IT IS A MARK
 ERROR 24 ; ERROR, TRANSMITTER IS NOT MARKING
 INC R3 ; INC BIT COUNT
 CMP #20,R3 ; DONE YET?
 BNE 45 ; BR IF NO
 5\$:

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

TEST 51

 1\$T51: SCOPE ;
 MOV #51,\$TSTNM ; LOAD THE NO. OF THIS TEST
 MOV #1\$T52,NEXT ; POINT TO THE START OF NEXT TEST.
 ; R1 CONTAINS BASE KMC11 ADDRESS
 MSTCLR ; MASTER CLEAR KMC11
 MOV #BIT11,(R1) ; SET LINE UNIT LOOP
 MOV #MESDAT,R2 ; LOAD POINTER TO DATA
 MOV #4,R0 ; LOAD CHARACTER COUNT
 JSR PC,SYNLD ; LOAD 2 SYNC'S IN OUT SILO
 JSR PC,OUTRDY ; WAIT FOR OUTRDY
 JSR R5,MESLD ; LOAD SILO WITH 4 CHAR MESS
 MESDAT ; ADDRESS OF MESSAGE
 4 ; NUMBER OF CHARACTERS
 JSR PC,EOM ; LOAD GARBAGE CHARACTER, WITH EOM SET
 JSR PC,OCOR ; WAIT FOR OCOR
 DATACLK,114 ; CLOCK DATA
 3\$: JSR PC,INRDY ; WAIT FOR INRDY
 ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 021204 ; GET IN DATA
 MOV 4(R1),R4 ; PUT "FOUND" IN R4
 MOVB (R2)+,R5 ; PUT "EXPECTED" IN R5
 CMPB R5,R4 ; COMPARE RECEIVED DATA
 BEQ 15 ; BR IF OK
 ERROR 10 ; DATA ERROR
 1\$: DEC R0 ; DEC CHARACTER COUNT
 BNE 3\$; BR IF NOT DONE YET

;CHECK TO SEE THAT IN BCC MATCH IS SET

JSR PC,INRDY ;WAIT FOR INRDY

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 81
 DZKCE.P11 12-MAY-77 12:23 BASIC RECEIVER TESTS

PAGE: 0102

4214	024116	104412			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4215	024120	021204			021204		;GET FIRST HALF OF CRC
4216	024122	116137	000004	001302	MOV8	4(R1),STMP2	;PUT IN STMP2
4217	024130	042737	177400	001302	BIC	#177400,STMP2	;CLEAR HI BYTE
4218	024136	004737	030614		JSR	PC,INRDY	;WAIT FOR INRDY
4219	024142	104412			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4220	024144	021244			021244		
4221	024146	016104	000004		MOV	4(R1),R4	;PUT "FOUND" IN R4
4222	024152	042704	000376		BIC	#376,R4	;CLEAR UNWANTED BITS
4223	024154	012705	000001		MOV	#1,R5	;PUT "EXPECTED" IN R5
4224	024162	120504			CMP8	R5,R4	;IS IN BCC MATCH SET?
4225	024164	001401			BEQ	25\$	
4226	024166	104015			ERROR	15	;IN BCC MATCH ERROR
4227	024170						
4228	024170	104412			ROMCLK		;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4229	024172	021204			021204		;GET LAST HALF
4230	024174						
4231							
4232							
4233							
4234							
4235							
4236							
4237							
4238							
4239							
4240							
4241							
4242							
4243							
4244							
4245							
4246							
4247	024174	000004					
4248	024176	012737	000052	001202	ST52:	SCOPE	
4249	024204	012737	025274	001442	MOV	#52,STSTNM	;LOAD THE NO. OF THIS TEST
4250					MOV	#ST53,NEXT	;POINT TO THE START OF NEXT TEST.
4251	024212	104410					;R1 CONTAINS BASE KMC11 ADDRESS
4252					MSTCLR		;MASTER CLEAR KMC11
4253							
4254							
4255	024214	012711	004000		MOV	#BIT1:(R1)	;SET LINE UNIT LOOP
4256	024220	012704	031540		MOV	#MESDAT,R4	;LOAD POINTER TO DATA
4257	024224	005037	024334		CLR	10\$	;CLEAR SFT BCC
4258	024230	012700	000004		MOV	#4,R0	;LOAD CHARACTER COUNT
4259	024234	004737	031122		JSR	PC,SYNLD	;LOAD 2 SYNC IN OUT SILO
4260	024240	004737	030140		JSR	PC,OUTRDY	;WAIT FOR OUTRDY
4261	024244	004537	031256		JSR	R5,MESLD	;LOAD SILO WITH 4 CHAR MESS
4262	024250	031540			MESDAT		;ADDRESS OF MESSAGE
4263	024252	000004					;NUMBER OF CHARACTERS
4264	024254	004737	031232		JSR	PC,EOM	;LOAD GARBAGE CHARACTER, WITH EOM SET
4265	024260	004537	031256		JSR	R5,MESLD	;LOAD FOUR MORE CHARACTERS
4266	024264	031540			MESDAT		;ADDRESS OF MESSAGE
4267	024266	000004					;NUMBER OF CHARACTERS
4268	024270	004737	031232		JSR	PC,EOM	;SET EOM
4269	024274	004737	030006		JSR	PC,OCOR	;WAIT FOR OCOR

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

TEST 52

 ;ST52: SCOPE
 MOV #52,STSTNM ;LOAD THE NO. OF THIS TEST
 MOV #ST53,NEXT ;POINT TO THE START OF NEXT TEST.
 ;R1 CONTAINS BASE KMC11 ADDRESS
 ;MASTER CLEAR KMC11
 ;LOAD OUT DATA SILO
 MOV #BIT1:(R1) ;SET LINE UNIT LOOP
 MOV #MESDAT,R4 ;LOAD POINTER TO DATA
 CLR 10\$;CLEAR SFT BCC
 MOV #4,R0 ;LOAD CHARACTER COUNT
 JSR PC,SYNLD ;LOAD 2 SYNC IN OUT SILO
 JSR PC,OUTRDY ;WAIT FOR OUTRDY
 JSR R5,MESLD ;LOAD SILO WITH 4 CHAR MESS
 MESDAT ;ADDRESS OF MESSAGE
 ;NUMBER OF CHARACTERS
 JSR PC,EOM ;LOAD GARBAGE CHARACTER, WITH EOM SET
 JSR R5,MESLD ;LOAD FOUR MORE CHARACTERS
 MESDAT ;ADDRESS OF MESSAGE
 ;NUMBER OF CHARACTERS
 JSR PC,EOM ;SET EOM
 JSR PC,OCOR ;WAIT FOR OCOR

4270	024300	005003			CLB	R3		:CLEAR BIT COUNTER
4271	024302	104413	000022		DATA CLK,	22		:CLOCK DATA
4272	024306	112405		12\$:	MOVB	(R4)+,R5		:LOAD R5 WITH CHAR
4273	024310	010502			MOV	R5,R2		:LOAD R2 WITH CHAR
4274								
4275								:CHECK FIRST FOUR CHARACTER MESSAGE
4276								:IN THE BIT WINDOW (0,125,252,377)
4277								
4278	024312	012737	120001	030754	MOV	#CRC16,XPOLY		:LOAD POLYNOMIAL
4279	024320	010537	024332		MOV	R5,67\$		:LOAD SOFT CHAR FOR BCC
4280	024324	004537	030650		JSR	R5,SIMBCC		:CALCULATE SOFT BCC
4281	024330	000010			LD			:SHIFT COUNT
4282	024332	000000		67\$:	0			:CHARACTER
4283	024334	000000		10\$:	0			:OLD BCC
4284	024336	013737	030756	024334	MOV	CALBCC,10\$		:LOAD SOFT BCC FOR NEXT SHIFT
4285	024344	104413	000001		DATA CLK,	1		:SHIFT DATA IN TO BIT WINDOW
4286	024350	106002		64\$:	RORB	R2		:SHIFT SOFT DATA
4287	024352	103005			BCC	65\$		:BR IF A SPACE
4288	024354	004737	027754		JSR	PC,GETSI		:LOOK AT BIT WINDOW
4289	024360	103406			BCS	66\$		:BR IF OK (MARK)
4290	024362	104006			ERROR	6		:ERROR, BIT WINDOW WAS A SPACE
4291	024364	000404			BR	66\$		:CONTINUE
4292	024366	004737	027754	65\$:	JSR	PC,GETSI		:LOOK AT BIT WINDOW
4293	024372	103001			BCC	66\$		:BR IF OK (SPACE)
4294	024374	104006			ERROR	6		:ERROR, BIT WINDOW WAS A MARK
4295	024376			66\$:				
4296	024376	005203			INC	R3		:BUMP BIT COUNTER
4297	024400	022703	000010		CMP	#10,R3		:DONE FULL 8 BITS YET
4298	024404	001357			BNE	64\$		:BR IF NO
4299	024406	005003			CLR	R3		:CLEAR BIT COUNTER
4300	024410	005300			DEC	R0		:DEC CHARACTER COUNT
4301	024412	001335			BNE	12\$		:BR IF NOT DONE YET
4302								
4303								:CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4304								
4305	024414	013700	030756		MOV	CALBCC,R0		:PUT BCC IN R0
4306	024420	104413	000001	68\$:	DATA CLK,	1		:SHIFT HARDWARE BCC
4307	024424	006000			ROR	R0		:SHIFT SOFT BCC
4308	024426	103005			BCC	69\$		:BR IF CARRY CLEAR
4309	024430	004737	027754		JSR	PC,GETSI		:LOOK AT BIT WINDOW
4310	024434	103406			BCS	70\$		:BR IF OK (MARK)
4311	024436	104014			ERROR	14		:ERROR, CRC WRONG (SPACE)
4312	024440	000404			BR	70\$		:CONTINUE
4313	024442	004737	027754	69\$:	JSR	PC,GETSI		:LOOK AT BIT WINDOW
4314	024446	103001			BCC	70\$		:BR IF OK (SPACE)
4315	024450	104014			ERROR	14		:ERROR, CRC WRONG (MARK)
4316	024452			70\$:				
4317	024452	005203			INC	R3		:BUMP BIT COUNTER
4318	024454	022703	000020		CMP	#20,R3		:FINISHED BCC YET?
4319	024460	001357			BNE	68\$		:BR IF NO
4320	024462	005003			CLR	R3		:CLEAR BIT COUNTER
4321	024464	012700	000004		MOV	#4,R0		:RESET CHARACTER COUNTER
4322	024470	012704	031540		MOV	#MESDAT,R4		:LOAD MESSAGE POINTER
4323	024474	005037	024526		CLR	11\$		:CLR SOFT BCC
4324	024500	112405		13\$:	MOVB	(R4)+,R5		:LOAD CHAR IN R5
4325	024502	010502			MOV	R5,R2		:LOAD CHAR IN R2

```

4326
4327 ;CHECK SECOND MESSAGE IN THE BIT WINDOW (0,125,252,377)
4328
4329 024504 012737 120001 030754 MOV #CRC16,XPOLY ;LOAD POLYNOMIAL
4330 024512 010537 024524 MOV R5,76$ ;LOAD SOFT CHAR FOR BCC
4331 024516 004537 030650 JSR R5,SIMBCC ;CALCULATE SOFT BCC
4332 024522 000010 10 ;SHIFT COUNT
4333 024524 000000 76$: 0 ;CHARACTER
4334 024526 000000 11$: 0 ;OLD BCC
4335 024530 013737 030756 024526 MOV CALBCC,11$ ;LOAD SOFT BCC FOR NEXT SHIFT
4336 024536 104413 000001 73$: DATACLK, 1 ;SHIFT DATA IN TO BIT WINDOW
4337 024542 106002 RORB R2 ;SHIFT SOFT DATA
4338 024544 103005 BCC 74$ ;BR IF A SPACE
4339 024546 004737 027754 JSR PC,GETSI ;LOOK AT BIT WINDOW
4340 024552 103406 BCS 75$ ;BR IF OK (MARK)
4341 024554 104006 ERROR 6 ;ERROR, BIT WINDOW WAS A SPACE
4342 024556 000404 BR 75$ ;CONTINUE
4343 024560 004737 027754 74$: JSR PC,GETSI ;LOOK AT BIT WINDOW
4344 024564 103001 BCC 75$ ;BR IF OK (SPACE)
4345 024566 104006 ERROR 6 ;ERROR, BIT WINDOW WAS A MARK
4346 024570 75$:
4347 024570 005203 INC R3 ;BUMP BIT COUNTER
4348 024572 022703 000010 CMP #10,R3 ;DONE FULL 8 BITS YET
4349 024576 001357 BNE 73$ ;BR IF NO
4350 024600 005003 CLR R3 ;CLEAR BIT COUNTER
4351 024602 005300 DEC R0 ;DEC CHARACTER COUNT
4352 024604 001335 BNE 13$ ;BR IF NOT DONE YET
4353
4354 ;CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4355
4356 024606 013700 030756 MOV CALBCC,R0 ;PUT BCC IN R0
4357 024612 104413 000001 77$: DATACLK, 1 ;SHIFT HARDWARE BCC
4358 024616 006000 ROR R0 ;SHIFT SOFT BCC
4359 024620 103005 BCC 78$ ;BR IF CARRY CLEAR
4360 024622 004737 027754 JSR PC,GETSI ;LOOK AT BIT WINDOW
4361 024626 103406 BCS 79$ ;BR IF OK (MARK)
4362 024630 104014 ERROR 14 ;ERROR, CRC WRONG (SPACE)
4363 024632 000404 BR 79$ ;CONTINUE
4364 024634 004737 027754 78$: JSR PC,GETSI ;LOOK AT BIT WINDOW
4365 024640 103001 BCC 79$ ;BR IF OK (SPACE)
4366 024642 104014 ERROR 14 ;ERROR, CRC WRONG (MARK)
4367 024644 79$:
4368 024644 005203 INC R3 ;BUMP BIT COUNTER
4369 024646 022703 000020 CMP #20,R3 ;FINISHED BCC YET?
4370 024652 001357 BNE 77$ ;BR IF NO
4371 024654 005003 CLR R3 ;CLEAR BIT COUNTER
4372
4373 ;CHECK TO SEE IF TRANSMITTER IS MARKING
4374
4375 024656 104413 000001 2$: DATACLK, 1 ;CLOCK TRANSMITTER
4376 024662 004737 027754 JSR PC,GETSI ;LOOK AT WINDOW
4377 024666 103401 BCS 3$ ;IT SHOULD BE MARKING
4378 024670 104024 ERROR 24 ;ERROR, BIT WAS A SPACE
4379 024672 005203 3$: INC R3 ;BUMP BIT COUNTER
4380 024674 022703 000007 CMP #7,R3 ;DONE YET
4381 024700 001366 BNE 2$ ;BR IF NO

```

4382	024702	104413	000010		DATACLK,	10		:GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
4383	024706	005003			CLR	R3		:CLEAR BIT COUNTER
4384	024710	104413	000001	4S:	DATACLK,	1		:SHIFT OUT NEXT BIT
4385	024714	004737	027754		JSR	PC,GETSI		:LOOK AT BIT WINDOW
4386	024720	103401			BCS	+4		:BR IF IT IS A MARK
4387	024722	104024			ERROR	24		:ERROR, TRANSMITTER IS NOT MARKING
4388	024724	005203			INC	R3		:INC BIT COUNT
4389	024726	022703	000020		CMP	#20,R3		:DONE YET?
4390	024732	001366			BNE	4S		:BR IF NO
4391								
4392								:CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE
4393								:WAS RECEIVED CORRECTLY (0,125,252,377)
4394								
4395	024734	104413	000001		DATACLK,	1		:GET LAST BIT IN RECEIVER
4396	024740	012703	000004		MOV	#4,R3		:R3=CHARACTER COUNT
4397	024744	012702	031540		MOV	#MESSAGE,R2		:LOAD MESSAGE POINTER IN R2
4398	024750	004737	030614	40S:	JSR	PC,INRDY		:WAIT FOR INRDY
4399	024754	104412			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4400	024756	021204			021204			
4401	024760	016104	000004		MOV	4(R1),R4		:PUT "FOUND" IN R4
4402	024764	112205			MOVB	(R2)+,R5		:PUT "EXPECTED" IN R5
4403	024766	120504			CMPB	R5,R4		:IS RECEIVED DATA CORRECT?
4404	024770	001401			BEQ	41S		:BR IF YES
4405	024772	104010			ERROR	10		:RECEIVE DATA ERROR
4406	024774	005303		41S:	DEC	R3		:DEC CHARACTER COUNT
4407	024776	001364			BNE	40S		:BR IF NOT DONE YET
4408								
4409								:CHECK TO SEE THAT IN BCC MATCH IS SET
4410								:AND THAT THE BCC WAS RECEIVED CORRECTLY
4411								
4412	025000	004737	030614		JSR	PC,INRDY		:WAIT FOR INRDY
4413	025004	104412			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4414	025006	021204			021204			:GET FIRST HALF OF CRC
4415	025010	116137	000004	001302	MOVB	4(R1),\$TMP2		:PUT IN \$TMP2
4416	025016	042737	177400	001302	BIC	#177400,\$TMP2		:CLEAR HI BYTE
4417	025024	004737	030614		JSR	PC,INRDY		:WAIT FOR INRDY
4418	025030	104412			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4419	025032	021244			021244			
4420	025034	016104	000004		MOV	4(R1),R4		:PUT "FOUND" IN R4
4421	025040	042704	000376		BIC	#376,R4		:CLEAR UNWANTED BITS
4422	025044	012705	000001		MOV	#1,R5		:PUT "EXPECTED" IN R5
4423	025050	120504			CMPB	R5,R4		:IS IN BCC MATCH SET?
4424	025052	001401			BEQ	50S		
4425	025054	104015			ERROR	15		:IN BCC MATCH ERROR
4426	025056			50S:				
4427	025056	104412			ROMCLK			:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4428	025060	021204			021204			:GET LAST HALF
4429	025062	116137	000004	001301	MOVB	4(R1),\$TMP1+1		:PUT IN \$TMP1
4430	025070	042737	000377	001300	BIC	#377,\$TMP1		:CLEAR LO BYTE
4431	025076	053737	001300	001302	BIS	\$TMP1,\$TMP2		:16 BIT BCC NOW IN \$TMP2
4432	025104	023737	030756	001302	CMP	CALBCC,\$TMP2		:IS IT CORRECT?
4433	025112	001401			BEQ	42S		:BR IF OK
4434	025114	104027			ERROR	27		
4435								
4436								:CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE
4437								:WAS RECEIVED CORRECTLY (0,125,252,377)

```

4438 025116 012703 000004 42$: MOV R3 ;R3=CHARACTER COUNT
4439 025122 012702 031540 MOV R2 ;LOAD MESSAGE POINTER IN R2
4440 025126 004737 030614 43$: JSR PC,INRDY ;WAIT FOR INRDY
4441 025132 104412 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4442 025134 021204 021204
4443 025136 016104 000004 MOV R4 ;PUT "FOUND" IN R4
4444 025142 112205 MOVE R5 ;PUT "EXPECTED" IN R5
4445 025144 120504 CMPB R5,R4 ;IS RECEIVED DATA CORRECT?
4446 025146 001401 BEQ 44$ ;BR IF YES
4447 025150 104010 ERROR 10 ;RECEIVE DATA ERROR
4448 025152 005303 44$: DEC R3 ;DEC CHARACTER COUNT
4449 025154 001364 BNE 43$ ;BR IF NOT DONE YET

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

4455 025156 004737 030614 JSR PC,INRDY ;WAIT FOR INRDY
4456 025162 104412 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4457 025164 021204 021204 ;GET FIRST HALF OF CRC
4458 025166 116137 000004 001302 MOVB 4(R1),STMP2 ;PUT IN STMP2
4459 025174 042737 177400 001302 BIC #177400,STMP2 ;CLEAR HI BYTE
4460 025202 004737 030614 JSR PC,INRDY ;WAIT FOR INRDY
4461 025206 104412 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4462 025210 021244 021244
4463 025212 016104 000004 MOV R4 ;PUT "FOUND" IN R4
4464 025216 042704 000376 BIC #376,R4 ;CLEAR UNWANTED BITS
4465 025222 012705 000001 MOV R5 ;PUT "EXPECTED" IN R5
4466 025226 120504 CMPB R5,R4 ;IS IN BCC MATCH SET?
4467 025230 001401 BEQ 51$
4468 025232 104015 ERROR 15 ;IN BCC MATCH ERROR
4469 025234 51$: ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4470 025234 104412 021204 ;GET LAST HALF
4471 025236 021204 000004 001301 MOVB 4(R1),STMP1+1 ;PUT IN STMP1
4472 025240 116137 000004 001301 BIC #377,STMP1 ;CLEAR LO BYTE
4473 025246 042737 000377 001300 BIS STMP1,STMP2 ;16 BIT BCC NOW IN STMP2
4474 025254 053737 001300 001302 CMP CALBCC,STMP2 ;IS IT CORRECT?
4475 025262 023737 030756 001302 BEQ 5$ ;BR IF OK
4476 025270 001401 ERROR 27
4477 025272 104027 5$:
4478 025274
4479
4480
4481 ;***** TEST 53 *****
4482 ;*DOCMP EOM FUNCTION TEST
4483 ;*THIS TEST LOADS OUT SILO WITH: 2 SYNCs, 4 CHAR MESSAGE, EOM
4484 ;*SOM, 4 CHAR MESS EOM. THE DATA STREAM IS CHECKED TO BE
4485 ;*4 CHAR, BCC 4 CHAR, BCC MARKS. THIS TEST VERIFYS THAT
4486 ;*THE CHARACTERS LOADED WITH EOM SET ARE LOST
4487 ;*ALSO THAT THE CHAR LOADED WITH SOM IS NOT IN THE BCC
4488 ;*ALL DATA AND BCC'S ARE CHECKED IN THE BIT WINDOW
4489 ;*THE FOUR CHARACTER MESSAGE IS 0,125,252,377
4490 ;*RECEIVED DATA IS VERIFIED, AND IN BCC MATCH IS CHECKED
4491 ;*****
4492
4493 ; TEST 53

```

```

4494
4495
4496 025274 000004
4497 025276 012737 000053 001202
4498 025304 012737 026474 001442
4499
4500 025312 104410
4501
4502
4503
4504 025314 012711 004000
4505 025320 012704 031540
4506 025324 005037 025440
4507 025330 012700 000004
4508 025334 004737 031122
4509 025340 004737 030140
4510 025344 004537 031256
4511 025350 031540
4512 025352 000004
4513 025354 004737 031232
4514 025360 004737 031202
4515 025364 004537 031256
4516 025370 031540
4517 025372 000004
4518 025374 004737 031232
4519 025400 004737 030006
4520 025404 005003
4521 025406 104413 000022
4522 025412 112405
4523 025414 010502
4524
4525
4526
4527
4528 025416 012737 120001 030754
4529 025424 010537 025436
4530 025430 004537 030650
4531 025434 000010
4532 025436 000000
4533 025440 000000
4534 025442 013737 030756 025440
4535 025450 104413 000001
4536 025454 106002
4537 025456 103005
4538 025460 004737 027754
4539 025464 103406
4540 025466 104006
4541 025470 000404
4542 025472 004737 027754
4543 025476 103001
4544 025500 104006
4545 025502
4546 025502 005203
4547 025504 022703 000010
4548 025510 001357
4549 025512 005003

*****
15153: SCOPE
MOV #53,STSTNM ; LOAD THE NO. OF THIS TEST
MOV #15154,NEXT ; POINT TO THE START OF NEXT TEST.
;R1 CONTAINS BASE KMC11 ADDRESS
;MASTER CLEAR KMC11
MSTCLR
;LOAD OUT DATA SILO
MOV #BIT11,(R1) ; SET LINE UNIT LOOP
MOV #MESDAT,R4 ; LOAD POINTER TO DATA
CLR 10$ ; CLEAR SOFT BCC
MOV #4,R0 ; LOAD CHARACTER COUNT
JSR PC,SYNLD ; LOAD 2 SYNCS IN OUT SILO
JSR PC,OUTRDY ; WAIT FOR OUTRDY
JSR R5,MESLD ; LOAD SILO WITH 4 CHAR MESS
MESDAT ; ADDRESS OF MESSAGE
4 ; NUMBER OF CHARACTERS
JSR PC,EOM ; LOAD GARBAGE CHARACTER, WITH EOM SET
JSR PC,SOM ; LOAD GARBAGE CHAR WITH SOM SET
JSP R5,MESLD ; LOAD FOUR MORE CHARACTERS
MESDAT ; ADDRESS OF MESSAGE
4 ; NUMBER OF CHARACTERS
JSR PC,EOM ; SET EOM
JSR PC,OCOR ; WAIT FOR OCOR
CLR R3 ; CLEAR BIT COUNTER
DATACLK,22 ; CLOCK DATA
12$: MOV# (R4)+,R5 ; LOAD R5 WITH CHAR
MOV R5,R2 ; LOAD R2 WITH CHAR
;CHECK FIRST FOUR CHARACTER MESSAGE
;IN THE BIT WINDOW (0,125,252,377)
MOV #CRC16,XPOLY ; LOAD POLYNOMIAL
MOV R5,67$ ; LOAD SOFT CHAR FOR BCC
JSR R5,SIMBCC ; CALCULATE SOFT BCC
10 ; SHIFT COUNT
67$: 0 ; CHARACTER
10$: 0 ; OLD BCC
MOV CALBCC,10$ ; LOAD SOFT BCC FOR NEXT SHIFT
64$: DATACLK,1 ; SHIFT DATA IN TO BIT WINDOW
RORB R2 ; SHIFT SOFT DATA
BCC 65$ ; BR IF A SPACE
JSR PC,GETSI ; LOOK AT BIT WINDOW
BCS 66$ ; BR IF OK (MARK)
ERROR 6 ; ERROR, BIT WINDOW WAS A SPACE
BR 66$ ; CONTINUE
65$: JSR PC,GETSI ; LOOK AT BIT WINDOW
BCC 66$ ; BR IF OK (SPACE)
ERROR 6 ; ERROR, BIT WINDOW WAS A MARK
66$: INC R3 ; BUMP BIT COUNTER
CMP #10,R3 ; DONE FULL 8 BITS YET
BNE 64$ ; BR IF NO
CLR R3 ; CLEAR BIT COUNTER

```

4550	025514	005300		DEC	R0		:DEC CHARACTER COUNT
4551	025516	001335		BNE	12\$		:BR IF NOT DONE YET
4552							
4553							:CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4554							
4555	025520	013700	030756	MOV	CALBCC,R0		:PUT BCC IN R0
4556	025524	104413	000001	68\$: DATACLK,	1		:SHIFT HARDWARE BCC
4557	025530	006000		ROR	R0		:SHIFT SOFT BCC
4558	025532	103005		BCC	69\$		:BR IF CARRY CLEAR
4559	025534	004737	027754	JSR	PC,GETSI		:LOOK AT BIT WINDOW
4560	025540	103406		BCS	70\$		:BR IF OK (MARK)
4561	025542	104014		ERROR	14		:ERROR, CRC WRONG (SPACE)
4562	025544	000404		BR	70\$		:CONTINUE
4563	025546	004737	027754	69\$: JSR	PC,GETSI		:LOOK AT BIT WINDOW
4564	025552	103001		BCC	70\$		:BR IF OK (SPACE)
4565	025554	104014		ERROR	14		:ERROR, CRC WRONG (MARK)
4566	025556			70\$: INC	R3		:BUMP BIT COUNTER
4567	025556	005203		CMP	#20,R3		:FINISHED BCC YET?
4568	025560	022703	000020	BNE	68\$		:BR IF NO
4569	025564	001357		CLR	R3		:CLEAR BIT COUNTER
4570	025566	005003					
4571							:CHECK CHARACTER LOADED WITH SOM (000), IN THE BIT WINDOW
4572							
4573							
4574	025570	005005		CLR	R5		:CHARACTER LOADED WITH SOM
4575	025572	010502		MOV	R5,R2		:LOAD R2 WITH CHAR
4576	025574	104413	000001	32\$: DATACLK,	1		:CLOCK TRANSMITTER
4577	025600	106002		RORB	R2		:SHIFT SOFT DATA
4578	025602	103005		BCC	30\$		:BR IF SPACE
4579	025604	004737	027754	JSR	PC,GETSI		:LOOK AT BIT WINDOW
4580	025610	103406		BCS	31\$		:BR IF OK (MARK)
4581	025612	104006		ERROR	6		:ERROR,BIT WINDOW WAS A SPACE
4582	025614	000404		BR	31\$		:CONTINUE
4583	025616	004737	027754	30\$: JSR	PC,GETSI		:LOOK AT BIT WINDOW
4584	025622	103001		BCC	31\$		:BR IF OK (SPACE)
4585	025624	104006		ERROR	6		:ERROR,BIT WINDOW WAS A MARK
4586	025626	005203		INC	R3		:BUMP BIT COUNTER
4587	025630	022703	000010	31\$: CMP	#10,R3		:DONE CHARACTER YET?
4588	025634	001357		BNE	32\$		:BR IF NO
4589	025636	005003		CLR	R3		:RESET BIT COUNTER
4590	025640	012700	000004	MOV	#4,R0		:RESET CHARACTER COUNTER
4591	025644	012704	031540	MOV	#MESSAGEPTR,R4		:LOAD MESSAGE POINTER
4592	025650	005037	025702	CLR	11\$		:CLR SOFT BCC
4593	025654	112405		13\$: MOV	(R4)+,R5		:LOAD CHAR IN R5
4594	025656	010502		MOV	R5,R2		:LOAD CHAR IN R2
4595							
4596							:CHECK SECOND MESSAGE IN THE BIT WINDOW (0,125,252,377)
4597							
4598	025660	012737	120001	030754	MOV	#CRC16,XPOLY	:LOAD POLYNOMIAL
4599	025666	010537	025700	MOV	R5,76\$		:LOAD SOFT CHAR FOR BCC
4600	025672	004537	030650	JSR	R5,SIMBCC		:CALCULATE SOFT BCC
4601	025676	000010		10			:SHIFT COUNT
4602	025700	000000		76\$: 0			:CHARACTER
4603	025702	000000		11\$: 0			:OLD BCC
4604	025704	013737	030756	025702	MOV	CALBCC,11\$	:LOAD SOFT BCC FOR NEXT SHIFT
4605	025712	104413	000001	73\$: DATACLK,	1		:SHIFT DATA IN TO BIT WINDOW

4606	025716	106002		RORB	R2	:SHIFT SOFT DATA
4607	025720	103005		BCC	74\$	:BR IF A SPACE
4608	025722	004737	027754	JSR	PC,GETSI	:LOOK AT BIT WINDOW
4609	025726	103406		BCS	75\$	:BR IF OK (MARK)
4610	025730	104006		ERROR	6	:ERROR, BIT WINDOW WAS A SPACE
4611	025732	000404		BR	75\$	:CONTINUE
4612	025734	004737	027754	JSR	PC,GETSI	:LOOK AT BIT WINDOW
4613	025740	103001		BCC	75\$	:BR IF OK (SPACE)
4614	025742	104006		ERROR	6	:ERROR, BIT WINDOW WAS A MARK
4615	025744					
4616	025744	005203		INC	R3	:BUMP BIT COUNTER
4617	025746	022703	000010	CMP	#10,P3	:DONE FULL 8 BITS YET
4618	025752	001357		BNE	73\$	:BR IF NO
4619	025754	005003		CLR	R3	:CLEAR BIT COUNTER
4620	025756	005300		DEC	R0	:DEC CHARACTER COUNT
4621	025760	001335		BNE	13\$	:BR IF NOT DONE YET
4622						
4623						:CHECK BCC FOR PRECEDING MESSAGE IN THE BIT WINDOW
4624						
4625	025762	013700	030756	MOV	CALBCC,R0	:PUT BCC IN R0
4626	025766	104413	000001	DATACLK,	1	:SHIFT HARDWARE BCC
4627	025772	006000		ROR	R0	:SHIFT SOFT BCC
4628	025774	103005		BCC	78\$	:BR IF CARRY CLEAR
4629	025776	004737	027754	JSR	PC,GETSI	:LOOK AT BIT WINDOW
4630	026002	103406		BCS	79\$	:BR IF OK (MARK)
4631	026004	104014		ERROR	14	:ERROR, CRC WRONG (SPACE)
4632	026006	000404		BR	79\$	:CONTINUE
4633	026010	004737	027754	JSR	PC,GETSI	:LOOK AT BIT WINDOW
4634	026014	103001		BCC	79\$	:BR IF OK (SPACE)
4635	026016	104014		ERROR	14	:ERROR, CRC WRONG (MARK)
4636	026020					
4637	026020	005203		INC	R3	:BUMP BIT COUNTER
4638	026022	022703	000020	CMP	#20,P3	:FINISHED BCC YET?
4639	026026	001357		BNE	77\$	:BR IF NO
4640	026030	005003		CLR	R3	:CLEAR BIT COUNTER
4641						
4642						:CHECK TO SEE IF TRANSMITTER IS MARKING
4643						
4644	026032	104413	000001	DATACLK,	1	:CLOCK TRANSMITTER
4645	026036	004737	027754	JSR	PC,GETSI	:LOOK AT WINDOW
4646	026042	103401		BCS	3\$	:IT SHOULD BE MARKING
4647	026044	104024		ERROR	24	:ERROR, BIT WAS A SPACE
4648	026046	005203		INC	R3	:BUMP BIT COUNTER
4649	026050	022703	000007	CMP	#7,R3	:DONE YET
4650	026054	001366		BNE	2\$	:BR IF NO
4651	026056	104413	000010	DATACLK,	10	:GIVE ENOUGH TICKS TO CLEAR OUT ACTIVE
4652	026062	005003		CLR	R3	:CLEAR BIT COUNTER
4653	026064	104413	000001	DATACLK,	1	:SHIFT OUT NEXT BIT
4654	026070	004737	027754	JSR	PC,GETSI	:LOOK AT BIT WINDOW
4655	026074	103401		BCS	+4	:BR IF IT IS A MARK
4656	026076	104024		ERROR	24	:ERROR, TRANSMITTER IS NOT MARKING
4657	026100	005203		INC	R3	:INC BIT COUNT
4658	026102	022703	000020	CMP	#20,R3	:DONE YET?
4659	026106	001366		BNE	4\$	:BR IF NO
4660						
4661						:CHECK TO SEE THAT FIRST FOUR CHARACTER MESSAGE

```

4662                                     ;WAS RECEIVED CORRECTLY (0,125,252,377)
4663
4664 026110 104413 000001          DATACLK, 1          ;GET LAST BIT IN RECEIVER
4665 026114 012703 000004          MOV  #4,R3          ;R3=CHARACTER COUNT
4666 026120 012702 03154C          MOV  #MESDAT,R2       ;LOAD MESSAGE POINTER IN R2
4667 026124 004737 030614          JSR   PC,INRDY        ;WAIT FOR INRDY
4668 026130 104412                                     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4669 026132 021204
4670 026134 016104 000004          MOV  4(R1),R4        ;PUT "FOUND" IN R4
4671 026140 112205          MOVB  (R2)+,R5          ;PUT "EXPECTED" IN R5
4672 026142 120504          CMPB  R5,R4              ;IS RECEIVED DATA CORRECT?
4673 026144 001401          BEQ   41$                ;BR IF YES
4674 026146 104010          ERROR  10                ;RECEIVE DATA ERROR
4675 026150 005303          41$: DEC  R3              ;DEC CHARACTER COUNT
4676 026152 001364          BNE   40$                ;BR IF NOT DONE YET
4677
4678                                     ;CHECK TO SEE THAT IN BCC MATCH IS SET
4679                                     ;AND THAT THE BCC WAS RECEIVED CORRECTLY
4680
4681 026154 004737 030614          JSR   PC,INRDY        ;WAIT FOR INRDY
4682 026160 104412          ROMCLK                                     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4683 026162 021204          021204                    ;GET FIRST HALF OF CRC
4684 026164 116137 000004 00.302  MOVB  4(R1),STMP2      ;PUT IN STMP2
4685 026172 042737 177400 001302  BIC   #177400,STMP2 ;CLEAR HI BYTE
4686 026200 004737 030614          JSR   PC,INRDY        ;WAIT FOR INRDY
4687 026204 104412          ROMCLK                                     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4688 026206 021244          021244
4689 026210 016104 000004          MOV  4(R1),R4        ;PUT "FOUND" IN R4
4690 026214 042704 000376          BIC   #376,R4        ;CLEAR UNWANTED BITS
4691 026220 012705 000001          MOV  #1,R5          ;PUT "EXPECTED" IN R5
4692 026224 120504          CMPB  R5,R4              ;IS IN BCC MATCH SET?
4693 026226 001401          BEQ   50$                ;IN BCC MATCH ERROR
4694 026230 104015          ERROR  15
4695 026232                                     50$:
4696 026232 104412          ROMCLK                                     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4697 026234 021204          021204                    ;GET LAST HALF
4698 026236 116137 000004 00:301  MOVB  4(R1),STMP1+1    ;PUT IN STMP1
4699 026244 042737 000377 00130C  BIC   #377,STMP1      ;CLEAR LO BYTE
4700 026252 053737 001300 001302  BIS   STMP1,STMP2     ;16 BIT BCC NOW IN STMP2
4701 026260 023737 030756 001302  CMP   CALBCC,STMP2   ;IS IT CORRECT?
4702 026266 001401          BEQ   45$                ;BR IF OK
4703 026270 104027          ERROR  27
4704
4705                                     ;CHECK THAT CHARACTER LOADED WITH SOM WAS RECEIVED (000)
4706
4707 026272 004737 030614          45$: JSR   PC,INRDY        ;WAIT FOR INRDY
4708 026276 104412          ROMCLK                                     ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4709 026300 021204          021204                    ;GET RECEIVE DATA
4710 026302 016104 000004          MOV  4(R1),R4        ;PUT "FOUND" IN R4
4711 026306 005005          CLR   R5          ;PUT "EXPECTED" IN R5
4712 026310 120504          CMPB  R5,R4              ;IS RECEIVED DATA CORRECT?
4713 026312 001401          BEQ   42$                ;BR IF YES
4714 026314 104010          ERROR  10                ;RECEIVE DATA ERROR
4715
4716                                     ;CHECK TO SEE THAT SECOND FOUR CHARACTER MESSAGE
4717                                     ;WAS RECEIVED CORRECTLY (0,125,252,377)

```

```

4718
4719 026316 012703 000004      42$: MOV      #4,R3      ;R3=CHARACTER COUNT
4720 026322 012702 031540      43$: MOV      #MESSAGE,R2    ;LOAD MESSAGE POINTER IN R2
4721 026326 004737 030614      JSR      PC,INRDY    ;WAIT FOR INRDY
4722 026332 104412              ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4723 026334 021204              021204
4724 026336 016104 000004      MOV      4(R1),R4    ;PUT "FOUND" IN R4
4725 026342 112205              MOV      (R2)+,R5    ;PUT "EXPECTED" IN R5
4726 026344 120504              CMP      R5,R4      ;IS RECEIVED DATA CORRECT?
4727 026346 001401              BEQ      44$      ;BR IF YES
4728 026350 104010              ERROR     10      ;RECEIVE DATA ERROR
4729 026352 005303              44$: DEC      R3      ;DEC CHARACTER COUNT
4730 026354 001364              BNE      43$      ;BR IF NOT DONE YET
4731
4732 ;CHECK TO SEE THAT IN BCC MATCH IS SET
4733 ;AND THAT THE BCC WAS RECEIVED CORRECTLY
4734
4735 026356 004737 030614      JSR      PC,INRDY    ;WAIT FOR INRDY
4736 026362 104412              ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4737 026364 021204              021204    ;GET FIRST HALF OF CRC
4738 026366 116137 000004 001302 MOV      4(R1),STMP2    ;PUT IN STMP2
4739 026374 042737 177400 001302 BIC      #177400,STMP2 ;CLEAR HI BYTE
4740 026402 004737 030614      JSR      PC,INRDY    ;WAIT FOR INRDY
4741 026406 104412              ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4742 026410 021244              021244
4743 026412 016104 000004      MOV      4(R1),R4    ;PUT "FOUND" IN R4
4744 026416 042704 000376      BIC      #376,R4      ;CLEAR UNWANTED BITS
4745 026422 012705 000001      MOV      #1,R5      ;PUT "EXPECTED" IN R5
4746 026426 120504              CMP      R5,R4      ;IS IN BCC MATCH SET?
4747 026430 001401              BEQ      51$      ;IN BCC MATCH ERROR
4748 026432 104015              ERROR     15
4749 026434              51$: ROMCLK    ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4750 026434 104412              021204    ;GET LAST HALF
4751 026436 021204              021204    ;PUT IN STMP1
4752 026440 116137 000004 001301 MOV      4(R1),STMP1+1 ;CLEAR LO BYTE
4753 026446 042737 000377 001300 BIC      #377,STMP1    ;16 BIT BCC NOW IN STMP2
4754 026454 053737 001300 001302 BIS      STMP1,STMP2    ;IS IT CORRECT?
4755 026462 023737 030756 001302 CMP      CALBCC,STMP2 ;BR IF OK
4756 026470 001401              BEQ      5$      ;
4757 026472 104027              ERROR     27
4758 026474              5$:
4759
4760
4761 ;***** TEST 54 *****
4762 ;*EMPTY SILO TEST
4763 ;*LOAD SILO WITH 2 SYNCS, 4 CHAR MESSAGE, SINGLE CLOCK
4764 ;*UNTIL THE SILO IS EMPTY, LOAD 4 MORE CHARACTERS IN THE
4765 ;*SILO. GIVE MORE TICKS, AND VERIFY THAT ONLY THE FIRST
4766 ;*4 CHARACTER MESSAGE WAS RECEIVED AND THAT RTS IS CLEAR
4767 ;*****
4768
4769 ; TEST 54
4770 ;-----
4771 ;*****
4772 026474 000004      1$T54: SCOPE
4773 026476 012737 000054 001202 MOV      #54,STSTNM    ; LOAD THE NO. OF THIS TEST

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 91
 DZKCE.P11 12-MAY-77 12:23 BASIC RECEIVER TESTS

PAGE: 0112

4774	026504	012737	026726	001442	MOV	#TST55,NEXT	: POINT TO THE START OF NEXT TEST.
4775							: R1 CONTAINS BASE KMC11 ADDRESS
4776	026512	104410			MSTCLR		: MASTER CLEAR KMC11
4777	026514	012711	004000		MOV	#BIT11,(R1)	: SET LINE UNIT LOOP
4778	026520	012702	031540		MOV	#MESDAT,R2	: R2 POINTS TO MESSAGE
4779	026524	012700	000004		MOV	#4,R0	: R0 = CHAR COUNT
4780	026530	004737	031122		JSR	PC,SYNLD	: LOAD SILO WITH TWO SYNCs
4781	026534	004737	030140		JSR	PC,OUTRDY	: WAIT FOR OUTRDY
4782	026540	004537	031256		JSR	RS,MESLD	: LOAD MESSAGE IN SILO
4783	026544	031540			MESDAT		: START OF MESSAGE
4784	026546	000004			4		: CHARACTER COUNT
4785	026550	004737	030006		JSR	PC,OCOR	: WAIT FOR OCOR
4786	026554	104413	000065		DATACLK,	65	: CLOCK DATA (EMPTY SILO)
4787	026560	004537	031256		JSR	RS,MESLD	: PUT MORE CHARACTERS IN SILO
4788	026564	031540			MESDAT		
4789	026566	000004			4		
4790	026570	004737	030006		JSR	PC,OCOR	
4791	026574	104413	000005		DATACLK,	5	: CLOCK UNTIL RTS IS CLEARED
4792	026600	104412			ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4793	026602	021264			021264		: GET RTS
4794	026604	032761	000040	000004	BIT	#BITS,4(R1)	: IS IT CLEAR?
4795	026612	001401			BEQ	5\$	: BR IF YES
4796	026614	104034			ERROR	34	: ERROR, RTS NOT CLEAR
4797	026616	104413	000041		DATACLK,	41	: CLOCK XMITTER SOME MORE
4798	026622	004737	030614		JSR	PC,INRDY	: OK LETS CHECK WHAT WAS RECEIVED
4799	026626	104412			ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4800	026630	021204			021204		: GET RECEIVE DATA
4801	026632	016104	000004		MOV	4(R1),R4	: PUT IT IN R4
4802	026636	112205			MOVB	(R2)+,R5	: R5 = "EXPECTED"
4803	026640	120504			CPMB	R5,R4	: IS DATA CORRECT?
4804	026642	001401			BEQ	2\$	: BR IF OK
4805	026644	104010			ERROR	10	: DATA ERROR
4806	026646	005300			DEC	R0	: DEC CHAR COUNT
4807	026650	001364			BNE	1\$	: BR IF NOT DONE YET
4808	026652	004737	030614		JSR	PC,INRDY	: WAIT FOR INRDY
4809	026656	104412			ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4810	026660	021204			021204		: GET RECEIVE DATA
4811	026662	016104	000004		MOV	4(R1),R4	: PUT IT IN "FOUND"
4812	026666	012705	000377		MOV	#377,R5	: R5 = "EXPECTED"
4813	026672	120504			CPMB	R5,R4	: SHOULD SEE 377
4814	026674	001401			BEQ	4\$	: BR IF OK
4815	026676	104010			ERROR	10	: ERROR, TRANSMITTER DID NOT ABORT
4816	026700	004737	030614		JSR	PC,INRDY	: WAIT FOR INRDY
4817	026704	104412			ROMCLK		: NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4818	026706	021204			021204		: GET RECEIVE DATA
4819	026710	016104	000004		MOV	4(R1),R4	: PUT IT IN "FOUND"
4820	026714	012705	000377		MOV	#377,R5	: R5 = "EXPECTED"
4821	026720	120504			CPMB	R5,R4	: SHOULD SEE 377
4822	026722	001401			BEQ	10\$	: BR IF OK
4823	026724	104010			ERROR	10	: ERROR, TRANSMITTER DID NOT ABORT
4824	026726						
4825							
4826							
4827							
4828							
4829							

5\$: DATACLK, 41
 1\$: JSR PC,INRDY
 2\$: DEC R0
 3\$: JSR PC,INRDY
 4\$: JSR PC,INRDY
 10\$: ERROR 10

***** TEST 55 *****
 *HALF DUPLEX TEST
 *SET LINE UNIT LOOP AND HALF DUPLEX, SEND SYNCs AND A

```

4830                                     ;:MESSAGE, VERIFY THAT IN-ACTIVE AND IN-READY ARE CLEAR
4831                                     ;:*****
4832                                     ;
4833                                     ; TEST 55
4834                                     ;-----
4835                                     ;:*****
4836 026726 000004 1ST55: SCOPE
4837 026730 012737 000055 001202 MOV #55,$TSTNM ; LOAD THE NO. OF THIS TEST
4838 026736 012737 027044 001442 MOV #TST56,NEXT ; POINT TO THE START OF NEXT TEST.
4839                                     ; R1 CONTAINS BASE KMC11 ADDRESS
4840 026744 104410 MSTCLR ; MASTER CLEAR KMC11
4841 026746 012702 000012 MOV #12,R2 ; SAVE R2 FOR TYPEOUT
4842 026752 012711 004000 MOV #BIT11,(R1) ; SET LINE UNIT LOOP
4843 026756 012761 000020 000004 MOV #BIT4,4(R1) ; LOAD PORT4
4844 026764 104412 ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4845 026766 122113 122113 ; SET H/D BIT
4846 026770 004737 031122 JSR PC,SYNLD ; LOAD 2 SYNCS
4847 026774 004737 030140 JSR PC,OUTRDY ; WAIT FOR OUTRDY
4848 027000 004537 031256 JSR RS,MESLD ; LOAD 4 CHAR MESSAGE
4849 027004 031540 MESDAT ; ADDRESS OF MESSAGE
4850 027006 000004 4 ; CHARACTER COUNT
4851 027010 004737 030006 JSR PC,OCOR ; WAIT FOR OCOR
4852 027014 104413 000073 DATACLK, 73 ; SEND MESSAGE
4853 027020 104412 ROMCLK ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4854 027022 021244 021244 ; READ LU-12
4855 027024 016104 000004 MOV 4(R1),R4 ; PUT "FOUND" IN R4
4856 027030 042704 000257 BIC #257,R4 ; CLEAR UNWANTED BITS
4857 027034 005005 CLR R5 ; R5 = "EXPECTED"
4858 027036 120504 CMPB R5,R4 ; IN-ACTIVE AND IN-RDY SHOULD BE CLEAR
4859 027040 001401 BEQ 15 ; BR IF OK
4860 027042 104035 ERROR 35 ; ERROR BOTH ARE NOT CLEAR
4861 027044 15:
4862
4863
4864                                     ;:***** TEST 56 *****
4865                                     ;:DOCMP CABLE DATA TEST
4866                                     ;:THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
4867                                     ;:4 SYNCS, 16 CHAR EOM, 16 CHAR, EOM, 16 CHAR, EOM
4868                                     ;:THE 16 CHARACTERS INCLUDE A FLOATING ONE AND ZERO
4869                                     ;:THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
4870                                     ;:RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
4871                                     ;:LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
4872                                     ;:*****
4873                                     ;
4874                                     ; TEST 56
4875                                     ;-----
4876                                     ;:*****
4877 027044 000004 1ST56: SCOPE
4878 027046 012737 000056 001202 MOV #56,$TSTNM ; LOAD THE NO. OF THIS TEST
4879 027054 012737 027434 001442 MOV #TST57,NEXT ; POINT TO THE START OF NEXT TEST.
4880                                     ; R1 CONTAINS BASE KMC11 ADDRESS
4881 027062 104410 MSTCLR ; MASTER CLEAR KMC11
4882 027064 032737 040000 002050 BIT #BIT14,STAT1 ; SKIP TEST IF NO
4883 027072 001557 BEQ 35 ; LOOPBACK CONNECTOR ON
4884 027074 012711 004000 MOV #BIT11,(R1) ; SET LINE UNIT LOOP
4885 027100 004737 031122 JSR PC,SYNLD ; LOAD 2 SYNCS

```

4886	027104	004737	031122		JSR	PC,SYMLD	:LOAD 2 MORE SYNCs
4887	027110	012737	120001	030754	MOV	#CRC16,XPOLY	:LOAD POLYNOMIAL FOR SOFT CRC CALC
4888	027116	005037	027146		CLR	6\$	:CLEAR OLD BCC
4889	027122	012703	000020		MOV	#16.,R3	:CHARACTER COUNT
4890	027126	012702	031544		MOV	#FLDAT,R2	:R2= POINTER
4891	027132	112237	027144	7\$:	MOVB	(R2)+,5\$	:LOAD CHAR FOR SOFT BCC CALC.
4892	027136	004537	030650		JSR	RS,SIMBCC	:CALC SOFT BCC
4893	027142	000010			LD		:SHIFT COUNT
4894	027144	000000		5\$:	0		:CHARACTER
4895	027146	000000		6\$:	0		:OLD BCC
4896	027150	013737	030756	027146	MOV	CALBCC,6\$	:LOAD OLD BCC
4897	027156	005303			DEC	R3	:DEC COUNT
4898	027160	001364			BNE	7\$	:BR IF NOT DONE YET
4899	027162	004537	031256		JSR	RS,MESLD	:LOAD SILO
4900	027166	031544			FLDAT		:MESSAGE ADDRESS
4901	027170	000020			16.		:CHARACTER COUNT
4902	027172	004737	031232		JSR	PC,EOM	:LOAD AN EOM
4903	027176	004537	031256		JSR	RS,MESLD	:LOAD SILO
4904	027202	031544			FLDAT		:MESSAGE ADDRESS
4905	027204	000020			16.		:CHARACTER COUNT
4906	027206	004737	031232		JSR	PC,EOM	:LOAD AN EOM
4907	027212	004537	031256		JSR	RS,MESLD	:LOAD SILO
4908	027216	031544			FLDAT		:MESSAGE ADDRESS
4909	027220	000020			16.		:CHARACTER COUNT
4910	027222	004737	031232		JSR	PC,EOM	:LOAD AN EOM
4911	027226	004737	030006		JSR	PC,OCOR	:WAIT FOR OCOR
4912	027232	005011			CLR	(R1)	:CLEAR LINE UNIT LOOP
4913	027234	012700	000003		MOV	#3,R0	:R0 = MESSAGE COUNT
4914	027240	012703	000020		MOV	#16.,R3	:R3= CHARACTER COUNT
4915	027244	012702	031544		MOV	#FLDAT,R2	:LOAD MESSAGE POINTER IN R2
4916	027250	004737	030614	1\$:	JSR	PC,INRDY	:WAIT FOR INRDY
4917	027254	104412			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4918	027256	021204			021204		:GET DATA FROM IN SILO
4919	027260	016104	000004		MOV	4(R1),R4	:PUT CHARACTER IN "FOUND"
4920	027264	112205			MOVB	(R2)+,R5	:PUT "EXPECTED" IN R5
4921	027266	120504			CMPB	R5,R4	:IS RECEIVED DATA CORRECT
4922	027270	001401			BEQ	2\$	:BR IF OK
4923	027272	104025			ERROR	25	:DATA ERROR
4924	027274			2\$:			
4925	027274	005303			DEC	R3	:DEC CHARACTER COUNT
4926	027276	001364			BNE	1\$	:BR IF NOT DONE THIS MESSAGE
4927	027300	012703	000020		MOV	#16.,R3	:RESET CHARACTER COUNT
4928							
4929							
4930							
4931							
4932	027304	004737	030614		JSR	PC,INRDY	:WAIT FOR INRDY
4933	027310	104412			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4934	027312	021204			021204		:GET FIRST HALF OF CRC
4935	027314	116137	000004	001302	MOVB	4(R1),\$TMP2	:PUT IN \$TMP2
4936	027322	042737	177400	001302	BIC	#177400,\$TMP2	:CLEAR HI BYTE
4937	027330	004737	030614		JSR	PC,INRDY	:WAIT FOR INRDY
4938	027334	104412			ROMCLK		:NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4939	027336	021244			021244		
4940	027340	016104	000604		MOV	4(R1),R4	:PUT "FOUND" IN R4
4941	027344	042704	000376		BIC	#376,R4	:CLEAR UNWANTED BITS

```

4942 027350 012705 000001      MOV      #1,R5      ;PUT "EXPECTED" IN R5
4943 027354 120504      CMPB     R5,R4      ;IS IN BCC MATCH SET?
4944 027356 001401      BEQ      25$      ;
4945 027360 104015      ERROR     15      ;IN BCC MATCH ERROR
4946 027362      25$:      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4947 027362 104412      021204      ;GET LAST HALF
4948 027364 021204      MOVB     4(R1),STMP1+1 ;PUT IN STMP1
4949 027366 116137 000004 001301      BIC      #377,STMP1 ;CLEAR LO BYTE
4950 027374 042737 000377 001300      BIS      STMP1,STMP2 ;16 BIT BCC NOW IN STMP2
4951 027402 053737 001300 001302      CMP      CALBCC,STMP2 ;IS IT CORRECT?
4952 027410 023737 030756 001302      BEQ      4$      ;BR IF OK
4953 027416 001401      ERROR     27
4954 027420 104027      4$:      MOV      #FLDAT,R2      ;RESET MESSAGE POINTER
4955 027422 012702 031544      DEC      R0      ;DECREMENT COUNTER
4956 027426 005300      BNE      1$      ;BR IF NOT DONE
4957 027430 001307      3$:      ADVANCE      ; ADVANCE TO NEXT TEST
4958 027432 104420
4959
4960
4961      ;***** TEST 57 *****
4962      ;DOCMP CABLE DATA TEST
4963      ;THIS TEST LOADS OUT SILO WITH THE FOLLOWING:
4964      ;4 SYNCs, 59 DATA CHARACTERS, EOM WITH GARBAGE CHARACTER
4965      ;THE DATA IS TRANSMITTED OVER THE CABLE USING THE INTERNAL CLOCK
4966      ;RECEIVED DATA IS VERIFIED AS IS IN BCC MATCH
4967      ;LOOP-BACK CONNECTOR MUST BE ON TO RUN THIS TEST
4968      ;*****
4969
4970      ; TEST 57
4971      ;-----
4972      ;*****
4973 027434 000004      ;ST57: SCOPE
4974 027436 012737 000057 001202      MOV      #57,STSTNM      ; LOAD THE NO. OF THIS TEST
4975 027444 012737 003662 001442      MOV      #SEOP,NEXT      ; POINT TO THE END OF PASS HANDLER.
4976
4977 027452 104410      MSTCLR      ;R1 CONTAINS BASE KMC11 ADDRESS
4978 027454 032737 040000 002050      BIT      #BIT14,STAT1 ;MASTER CLEAR KMC11
4979 027462 001533      BEQ      3$      ;SKIP TEST IF NO
4980 027464 012711 004000      MOV      #BIT11,(R1) ;LOOPBACK CONNECTOR ON
4981 027470 004737 031122      JSR      PC,SYNLD ;SET LINE UNIT LOOP
4982 027474 004737 031122      JSR      PC,SYNLD ;LOAD 2 SYNCs
4983 027500 012737 120001 030754      MOV      #CRC16,XPOLY ;LOAD 2 MORE SYNCs
4984 027506 005037 027536      CLR      6$      ;LOAD POLYNOMIAL FOR SOFT CRC CALC
4985 027512 012703 000073      MOV      #59,R3      ;CLEAR OLD BCC
4986 027516 012702 031540      MOV      #MESDAT,R2 ;CHARACTER COUNT
4987 027522 112237 027534      7$:      MOVB     (R2)+,5$ ;R2= POINTER
4988 027526 004537 030650      JSR      R5,SIMBCC ;LOAD CHAR FOR SOFT BCC CALC.
4989 027532 000010      10      ;CALC SOFT BCC
4990 027534 000000      5$:      0      ;SHIFT COUNT
4991 027536 000000      6$:      0      ;CHARACTER
4992 027540 013737 030756 027536      MOV      CALBCC,6$ ;OLD BCC
4993 027546 005303      DEC      R3      ;LOAD OLD BCC
4994 027550 001364      BNE      7$      ;DEC COUNT
4995 027552 004537 031256      JSR      R5,MESLD ;BR IF NOT DONE YET
4996 027556 031540      MESDAT      ;LOAD SILO
4997 027560 000073      59.      ;MESSAGE ADDRESS
                     ;CHARACTER COUNT

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 95
 DZKCE.P11 12-MAY-77 12:23

PAGE: 0116

BASIC RECEIVER TESTS

```

4998 027562 004737 031232      JSR    PC,EOM      ;LOAD AN EOM
4999 027562 004737 030006      JSR    PC,OCOR     ;WAIT FOR OCOR
5000 027572 005011              CLR    (R1)        ;CLEAR LINE UNIT LOOP
5001 027574 012700 000073      MOV     #59,R0      ;R0= CHARACTER COUNT
5002 027600 012702 031540      MOV     #MESSAGE,R2 ;LOAD MESSAGE POINTER IN R2
5003 027604 004737 030614      JSR    PC,INRDY    ;WAIT FOR INRDY
5004 027610 104412              ROMCLK  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5005 027612 021204 000004      021204 ;GET DATA FROM IN SILO
5006 027614 016104              MOV     4(R1),R4      ;PUT CHARACTER IN "FOUND"
5007 027620 112205              MOV     (R2)+,R5      ;PUT "EXPECTED" IN R5
5008 027622 120504              CMP     R5,R4        ;IS RECEIVED DATA CORRECT
5009 027624 001401              BEQ     25$          ;BR IF OK
5010 027626 104025              ERROR    25          ;DATA ERROR
5011 027630              25$:
5012 027630 005300              DEC     R0          ;DECREMENT COUNTER
5013 027632 001364              BNE     1$          ;BR IF NOT DONE
5014
5015                          ;CHECK TO SEE THAT IN BCC MATCH IS SET
5016                          ;AND THAT THE BCC WAS RECEIVED CORRECTLY
5017
5018 027634 004737 030614      JSR    PC,INRDY    ;WAIT FOR INRDY
5019 027640 104412              ROMCLK  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5020 027642 021204 000004      021204 ;GET FIRST HALF OF CRC
5021 027644 116137 001302      MOV     4(R1),STMP2 ;PUT IN STMP2
5022 027652 042737 001302      BIC     #177400,STMP2 ;CLEAR HI BYTE
5023 027660 004737 030614      JSR    PC,INRDY    ;WAIT FOR INRDY
5024 027664 104412              ROMCLK  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5025 027666 021244 000004      021244 ;PUT "FOUND" IN R4
5026 027670 016104 000376      MOV     4(R1),R4      ;CLEAR UNWANTED BITS
5027 027674 042704 000001      BIC     #376,R4      ;PUT "EXPECTED" IN R5
5028 027700 012705 000001      MOV     #1,R5      ;IS IN BCC MATCH SET?
5029 027704 120504              CMP     R5,R4
5030 027706 001401              BEQ     25$          ;IN BCC MATCH ERROR
5031 027710 104015              ERROR    15
5032 027712              25$:
5033 027712 104412              ROMCLK  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5034 027714 021204 000004      021204 ;GET LAST HALF
5035 027716 116137 001301      MOV     4(R1),STMP1+1 ;PUT IN STMP1
5036 027724 042737 001300      BIC     #377,STMP1 ;CLEAR LO BYTE
5037 027732 053737 001302      BIS     STMP1,STMP2 ;16 BIT BCC NOW IN STMP2
5038 027740 023737 030756      CMP     CALBCC,STMP2 ;IS IT CORRECT?
5039 027746 001401              BEQ     3$          ;BR IF OK
5040 027750 104027              ERROR    27
5041 027752 104420              3$: ADVANCE ; ADVANCE TO NEXT TEST
5042
5043
5044 ;SUBROUTINES
5045 ;-----
5046
5047 027754      GETSI:
5048              ;THIS SUBROUTINE READS LU 17, AND PUTS IT INTO NITCH.
5049              ;NITCH IS ROTATED LEFT UNTILL THE SI BIT IS IN CARRY
5050
5051 027754 104412              ROMCLK  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5052 027756 021364              021364 ;PORT4+LU 17
5053 027760 017737 152110 030004      MOV     #KMP04,NITCH ;STORE LU 17

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 96
 DZKCE.P11 12-MAY-77 12:23 SUBROUTINES

PAGE: 0117

```

5054 027766 106137 030004      ROLB  NITC
5055 027772 106137 030004      ROLB  NITC
5056 027776 106137 030004      ROLB  NITC      ;PUT SI IN THE CARRY BIT
5057 030002 000207          RTS    PC
5058 030004 000000      NITC: 0
5059
5060
5061 030006      OCOR:
5062          ;THIS SUBROUTINE SPINS ON OCOR
5063
5064 030006 104412      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5065 030010 021364      021364      ;PORT4+LU 17
5066 030012 032777 000020 152054  BIT    #BIT4,2KMP04      ;IS OCOR SET?
5067 030020 001772      BEQ    OCOR      ;BR IF NO
5068 030022 000207      RTS    PC      ;OK OCOR IS SET, GO BACK
5069
5070
5071 030024      SYNC:
5072          ;THIS SUBROUTINE LOADS THE SILO WITH THE NUMBER OF SYNC
5073          ;CHARACTERS PASSED TO IT IN THE WORD AFTER THE JSR CALL
5074          ;AND A NON-SYNC CHARACTER (301)
5075
5076 030024 013637 001276      MOV    2(SP),STMP0      ;GET COUNT
5077 030030 062746 000002      ADD    #2,-(SP)      ;ADJUST STACK
5078 030034 012761 000026 000004  MOV    #26,4(R1)      ;LOAD PORT4
5079 030042 104412      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5080 030044 122114      122114      ;LOAD SYNC REGISTER
5081 030046 004737 030140 15:  JSR    PC,OUTRDY      ;WAIT FOR OUTRDY
5082 030052 012761 000001 000004  MOV    #1,4(R1)      ;LOAD PORT4
5083 030060 104412      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5084 030062 122111      122111      ;SET SOM
5085 030064 012761 000026 000004  MOV    #26,4(R1)      ;LOAD PORT4
5086 030072 104412      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5087 030074 122110      122110      ;LOAD OUT DATA
5088 030076 005337 001276      DEC    STMP0      ;ALL DONE?
5089 030102 001361      BNE    15      ;BR IF NOT
5090 030104 004737 030140      JSR    PC,OUTRDY      ;WAIT FOR OUTRDY
5091 030110 005061 000004      CLR    4(R1)      ;LOAD PORT4
5092 030114 104412      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5093 030116 122111      122111      ;SET SOM
5094 030120 012761 000301 000004  MOV    #301,4(R1)      ;LOAD PORT4
5095 030126 104412      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5096 030130 122110      122110      ;LOAD OUT DATA
5097 030132 004737 030006      JSR    PC,OCOR      ;WAIT FOR OCOR
5098 030136 000207      RTS    PC
5099
5100
5101 030140      OUTRDY:
5102          ;THIS SUBROUTINE SPINS ON OUT READY
5103
5104 030140 005037 001306      CLR    STMP4      ;CLEAR TIMER
5105 030144
5106 030144 104412      15:  ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5107 030146 021224      021224      ;PORT4+LU11
5108 030150 032777 000020 151716  BIT    #BIT4,2KMP04      ;IS OUT RDY SET?
5109 030156 001004      BNE    25      ;BR IF YES

```

```

5110 030160 005237 001306      INC      STMP4      ;INC TIMER
5111 030164 001367              BNE      1$      ;KEEP CHECKING IF NOT DONE
5112 030166 104036              ERROR    36      ;ERROR, OUT READY NOT SET
5113 030170 000207      2$:      RTS      PC
5114
5115
5116 030172              CHAR:
5117              ;THIS SUBROUTINE LOADS THE SILO WITH 3 SYNCs
5118              ;AND THE CHARACTER PASSED TO IT.
5119
5120 030172 013637 001300      MOV      2(SP)+,STMP1      ;GET CHARACTER
5121 030176 062746 000002      ADD      2,-(SP)      ;ADJUST STACK
5122 030202 012737 000003 001276      MOV      2,STMP0      ;SET FOR 3 SYNCs
5123 030210 012761 000026 000004      MOV      26,4(R1)      ;LOAD PORT4
5124 030216 104412              ROMCLK              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5125 030220 122114              122114      ;LOAD SYNC REGISTER
5126 030222 004737 030140      JSR      PC,OUTRDY      ;WAIT FOR OUTRDY
5127 030226 012761 000001 000004      MOV      1,4(R1)      ;LOAD PORT4
5128 030234 104412              ROMCLK              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5129 030236 122111              122111      ;SET SOM
5130 030240 012761 000026 000004      MOV      26,4(R1)      ;LOAD PORT4
5131 030246 104412              ROMCLK              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5132 030250 122110              122110      ;LOAD OUT DATA
5133 030252 005337 001276      DEC      STMP0      ;ALL DONE?
5134 030256 001361              BNE      1$      ;BR IF NOT
5135 030260 004737 030140      JSR      PC,OUTRDY      ;WAIT FOR OUTRDY
5136 030264 013761 001300 000004      MOV      STMP1,4(R1)      ;LOAD PORT4
5137 030272 104412              ROMCLK              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5138 030274 122110              122110      ;LOAD OUT DATA
5139 030276 004737 030006      JSR      PC,OCOR      ;WAIT FOR OCOR
5140 030302 000207      RTS      PC
5141
5142
5143 030304              CHARSD:
5144              ;THIS SUBROUTINE LOADS THE SILO WITH THE CHARACTER PASSED TO IT.
5145
5146 030304 013637 001300      MOV      2(SP)+,STMP1      ;GET CHARACTER
5147 030310 062746 000002      ADD      2,-(SP)      ;ADJUST STACK
5148 030314 004737 030140      JSR      PC,OUTRDY      ;WAIT FOR OUTRDY
5149 030320 013761 001300 000004      MOV      STMP1,4(R1)      ;LOAD PORT4
5150 030326 104412              ROMCLK              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5151 030330 122110              122110      ;LOAD OUT DATA
5152 030332 004737 030140      JSR      PC,OUTRDY      ;WAIT FOR OUTRDY
5153 030336 104412              ROMCLK              ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5154 030340 122110              122110      ;LOAD GARBAGE CHAR
5155 030342 004737 030006      JSR      PC,OCOR      ;WAIT FOR OCOR
5156 030346 000207      RTS      PC
5157
5158
5159 030350              SILOLD:
5160              ;THIS SUBROUTINE FILLS THE OUT SILO
5161              ;WITH A BINARY COUNT PATTERN
5162
5163 030350 012737 000073 001300      MOV      73,STMP1      ;LOAD COUNT
5164 030356 005737 030610      TST      SCHAR      ;FIRST TIME HERE?
5165 030362 100470              BMI      4$      ;BR IF BITSTUFF

```

```

5166 030364 001032      BNE      2$      ;BR IF NO
5167 030366 062737 000002 001300      ADD      #2,$TMP1      ;ADD 2 TO CHARACTER COUNT
5168 030374 012737 000003 001276      MOV      #3,$TMP0      ;SET FOR 3 SYNCs
5169 030402 012761 000026 000004      MOV      #26,4(R1)      ;LOAD PORT4
5170 030410 104412      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5171 030412 122114      122114      ;LOAD SYNC REGISTER
5172 030414 004737 030140      1$:      JSR      PC,OUTRDY      ;WAIT FOR OUTRDY
5173 030420 012761 000001 000004      MOV      #1,4(R1)      ;LOAD PORT4
5174 030426 104412      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5175 030430 122111      122111      ;SET SOM
5176 030432 012761 000026 000004      MOV      #26,4(R1)      ;LOAD PORT4
5177 030440 104412      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5178 030442 122110      122110      ;LOAD OUT DATA
5179 030444 005337 001276      DEC      $TMP0      ;ALL DONE?
5180 030450 001361      BNE      1$      ;BR IF NOT
5181 030452 004737 030140      2$:      JSR      PC,OUTRDY      ;WAIT FOR OUTRDY
5182 030456 013761 030610 000004      MOV      $CHAR,4(R1)      ;LOAD PORT4
5183 030464 104412      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5184 030466 122110      122110      ;LOAD OUT DATA
5185 030470 005737 030612      TST      $TUFLG      ;BITSTUFF???
5186 030474 001407      BEQ      6$      ;BR IF NO
5187 030476 013737 030610 030510      MOV      $CHAR,$$      ;IT IS SOLD SO CHECK BITSTUFFING
5188 030504 004537 031340      JSR      $S,$$FCL      ;ADD ANY BIT STUFF CLOCK TICKS
5189 030510 000000      5$:      0      ;CHARACTER
5190 030512 000010      10      ;SHIFT COUNT
5191 030514 005237 030610      6$:      INC      $CHAR      ;NEXT CHARACTER
5192 030520 022737 000400 030610      CMP      #400,$CHAR      ;ALL DONE?
5193 030526 001403      BEQ      3$      ;DECREMENT COUNT
5194 030530 005337 001300      DEC      $TMP1      ;BR IF NOT DONE
5195 030534 001346      BNE      2$      ;WAIT FOR OCOR
5196 030536 004737 030006      3$:      JSR      PC,OCOR      ;START PATTERN AT ZERO
5197 030542 000207      RTS      PC      ;SET BITSTUFF FLAG
5198 030544 005037 030610      4$:      CLR      $CHAR      ;CLEAR STUFF COUNT
5199 030550 012737 177777 030612      MOV      #-1,$TUFLG      ;ADD 2 TO CHARACTER COUNT
5200 030556 005037 031536      CLR      $BITCON      ;SET BIT0 IN PORT4
5201 030562 062737 000002 001300      ADD      #2,$TMP1      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5202 030570 012761 000001 000004      MOV      #1,4(R1)      ;SET SOM!
5203 030576 104412      ROMCLK      ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5204 030600 122111      122111      ;LOAD GARBAGE CHAR
5205 030602 104412      ROMCLK      ;GO LOAD SILO
5206 030604 122110      122110
5207 030606 000721      BR      2$
5208 030610 000000      SCHAR: 0
5209 030612 000000      STUFLG: 0
5210
5211
5212 030614      INRDY:
5213      ;THIS SUBROUTINE SPINS ON INRDY
5214      ;IF INRDY FAILS TO SET THE DELAY TIMES OUT AND AN
5215      ;ERROR IS REPORTED. FOR BETTER SCOPE LOOPS THIS
5216      ;DELAY CAN BE MADE SHORTER BY ALTERING THE NUMBER
5217      ;INITIALLY LOADED INTO $TMP0, THE SMALLER THE NUMBER
5218      ;THE SHORTER THE DELAY. 0 IS THE LONGEST DELAY.
5219
5220 030614 012737 000000 001276      1$:      MOV      #0,$TMP0      ;SET UP DELAY COUNTER
5221 030622

```

```

5222 030622 104412 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5223 030624 021244 021244 ;PORT4+LUI2
5224 030626 032777 000020 151240 BIT #BIT4,2KMP04 ;IS INRDY SET?
5225 030634 001004 BNE 2$ ;BR IF YES
5226 030636 005237 001276 INC $TMP0 ;INC DELAY
5227 030642 001367 BNE 1$ ;TRY AGAIN
5228 030644 104037 ERROR 37 ;ERROR NO INRDY
5229 030646 000207 2$: RTS PC ;RETURN
5230
5231
5232 030650 SIMBCC: ;THIS SUBROUTINE CALCULATES THE CRC USING POLYNOMIAL GIVEN
5233 ;IN XPOLY. THE CORRECT CRC IS SLPADDED IN CALBCC, AND THE
5234 ;STATE OF THE LSB OF THE BCC IS SLPADDED IN THE C BIT.
5235
5236
5237 030650 010046 MOV RO,-(SP) ;SAVE RO ON STACK
5238 030652 012537 001276 MOV (R5)+,$TMP0 ;$TMP0 = SHIFT COUNT
5239 030656 012537 001300 MOV (R5)+,$TMP1 ;$TMP1 = CHARACTER
5240 030662 012537 030756 MOV (R5)+,CALBCC ;CALBCC = OLD BCC
5241 030666 013700 030756 1$: MOV CALBCC,RO ;PUT OLD BCC IN RO
5242 030672 000241 CLC ;
5243 030674 006037 030756 ROR CALBCC ;SHIFT OLD BCC
5244 030700 006037 001300 ROR $TMP1 ;SHIFT CHARACTER
5245 030704 005500 ADC RO ;ADD CHAR CARRY TO OLD BCC
5246 030706 006000 ROR RO ;PUT BIT0 TO CARRY BIT
5247 030710 103011 BCC 2$ ;CARRY IS FEEDBACK BIT
5248 030712 013700 030754 MOV XPOLY,RO ;IF FEEDBACK = 1
5249 030716 043700 030756 BIC CALBCC,RO ;EXCLUSIVLY OR XPOLY TO CALBCC
5250 030722 043737 030754 BIC XPOLY,CALBCC
5251 030730 050037 030756 BIS RO,CALBCC
5252 030734 005337 001276 2$: DEC $TMP0 ;DEC SHIFT COUNT
5253 030740 001352 BNE 1$ ;BR IF NOT DONE
5254 030742 013700 030756 MOV CALBCC,RO ;PUT RESULT IN RO
5255 030746 006000 ROR RO ;SHIFT BIT0 TO CARRY
5256 030750 012600 MOV (SP)+,RO ;RESTORE RO
5257 030752 000205 RTS R5 ;SLPADR
5258 030754 000000 XPOLY: 0
5259 030756 000000 CALBCC: 0
5260 000200 LRC8=200
5261 120001 CRC16=120001
5262 102010 CRC.CCITT=102010
5263
5264
5265 030760 BCCLD: ;THIS SUBROUTINE LOADS THE OUT SILO WITH 2 SYNCs
5266 ;WITH SOM SET, AND ONE CHARACTER PASSED TO IT
5267 ;WITH THE SOM BIT CLEAR (ENABLE CRC)
5268
5269
5270 030760 013637 001300 MOV @($P)+,$TMP1 ;GET CHARACTER
5271 030764 062746 000002 ADD #2,-($P) ;ADJUST STACK
5272 030770 012737 000002 001276 MOV #2,$TMP0 ;SET FOR 2 SYNCs
5273 030776 012761 000026 000004 MOV #26,4(R1) ;LOAD PORT4
5274 031004 104412 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5275 031006 122114 122114 ;LOAD SYNC REGISTER
5276 031010 004737 030140 1$: JSR PC,OUTRDY ;WAIT FOR OUTRDY
5277 031014 012761 000001 000004 MOV #1,4(R1) ;LOAD PORT4

```

```

5278 031022 104412 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5279 031024 122111 ;SET SOM
5280 031026 012761 000026 000004 MOV #26,4(R1) ;LOAD PORT4
5281 031034 104412 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5282 031036 122110 ;LOAD OUT DATA
5283 031040 005337 001276 DEC $TMP0 ;ALL DONE?
5284 031044 001361 BNE IS ;BR IF NOT
5285 031046 004737 030140 JSR PC,OUTRDY ;WAIT FOR OUTRDY
5286 031052 013761 001300 000004 MOV $TMP1,4(R1) ;LOAD PORT4
5287 031060 104412 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5288 031062 122110 ;LOAD OUT DATA
5289 031064 004737 030006 JSR PC,OCOR ;WAIT FOR OCOR
5290 031070 000297 RTS PC
5291
5292
5293 031072 GETQ0: ;THIS SUBROUTINE READS THE STATE OF THE TRANSMIT
5294 ;BCC LSB AND PUTS IT IN THE CARRY BIT
5295
5296
5297 031072 104412 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5298 031074 021364 ;PORT4+LU-17
5299 031076 106177 150772 ROLB @KMP04 ;PUT Q0 IN CARRY
5300 031102 000207 RTS PC ;RETURN
5301
5302
5303 031104 GETQ1: ;THIS SUBROUTINE READS THE STATE OF THE RECEIVE
5304 ;BCC LSB AND PUTS IT IN THE CARRY BIT
5305
5306
5307 031104 104412 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5308 031106 021364 ;PORT4+LU-17
5309 031110 106177 150760 ROLB @KMP04 ;PUT Q0 IN CARRY
5310 031114 106177 150754 ROLB @KMP04 ;PUT Q1 IN CARRY
5311 031120 000207 RTS PC ;RETURN
5312
5313
5314 031122 SYNLD: ;THIS SUBROUTINE LOADS OUT SILO WITH
5315 ;2 SYNC CHARACTERS WITH SOM SET
5316
5317
5318 031122 012737 000002 001276 MOV #2,$TMP0 ;LOAD COUNTER FOR 2 SYNCs
5319 031130 012761 000026 000004 MOV #26,4(R1) ;PORT4+26
5320 031136 104412 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5321 031140 122114 ;LOAD SYNC REG
5322 031142 004737 030140 JSR PC,OUTRDY ;WAIT FOR OUTRDY
5323 031146 012761 000001 000004 MOV #1,4(R1) ;LOAD PORT4
5324 031154 104412 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5325 031156 122111 ;SET SOM
5326 031160 012761 000026 000004 MOV #26,4(R1) ;PORT4+26
5327 031166 104412 ROMCLK ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5328 031170 122110 ;LOAD OUT DATA WITH SYNC
5329 031172 005337 001276 DEC $TMP0 ;DECREMENT COUNTER
5330 031176 001361 BNE IS ;BR IF NOT DONE
5331 031200 000207 RTS PC ;RETURN
5332
5333

```

```

5334 031202          SOM:
5335                ; THIS SUBROUTINE LOADS SOM AND OUT DATA WITH A
5336                ; GARBAGE CHARACTER (0)
5337
5338 031202 004737 030140      JSR      PC,OUTRDY      ; WAIT FOR OUTRDY
5339 031206 012761 000001 000004  MOV      #1,4(R1)      ; PORT4+1
5340 031214 104412          ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5341 031216 122111          122111      ; SET SOM
5342 031220 005061 000004      CLR      4(R1)      ; CLEAR DATA CHAR
5343 031224 104412          ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5344 031226 122110          122110      ; LOAD GARBAGE CHARACTER
5345 031230 000207          RTS      PC      ; RETURN
5346
5347
5348 031232          EOM:
5349                ; THIS SUBROUTINE LOADS EOM AND OUT DATA WITH A
5350                ; GARBAGE CHARACTER (2) TO ENABLE TRANSMISSION OF BCC
5351
5352 031232 004737 030140      JSR      PC,OUTRDY      ; WAIT FOR OUTRDY
5353 031236 012761 000002 000004  MOV      #2,4(R1)      ; PORT4+2
5354 031244 104412          ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5355 031246 122111          122111      ; SET EOM
5356 031250 104412          ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5357 031252 122110          122110      ; LOAD GARBAGE CHARACTER
5358 031254 000207          RTS      PC      ; RETURN
5359
5360
5361 031256          MESLD:
5362                ; THIS SUBROUTINE LOADS SILO WITH MESSAGE
5363                ; THE FIRST ARGUMENT IS THE ADDRESS OF THE MESSAGE
5364                ; THE SECOND ARGUMENT IS THE NUMBER OF CHARACTERS IN THE MESSAGE
5365
5366 031256 010046          MOV      RO,-(SP)      ; SAVE RO
5367 031260 012500          MOV      (R5)+,RO      ; RO=MESSAGE POINTER
5368 031262 012537 001276      MOV      (R5)+,$TMPD      ; $TMPD=CHARACTER COUNT
5369 031266 004737 030140      JSR      PC,OUTRDY      ; WAIT FOR OUT RDY
5370 031272 112061 000004      MOV      (R0)+,4(R1)      ; LOAD PORT4 WITH CHARACTER
5371 031276 104412          ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5372 031300 122110          122110      ; LOAD OUT DATA SILO
5373 031302 005337 001276      DEC      $TMPD      ; DEC CHAR COUNT
5374 031306 001367          BNE      IS      ; BR IF NOT DONE
5375 031310 004737 030006      JSR      PC,OCOR      ; WAIT FOR OCOR
5376 031314 012600          MOV      (SP)+,RO      ; RESTORE RO
5377 031316 000205          RTS      R5      ; RETURN
5378
5379
5380 031320          CLRIO:
5381                ; THIS SUBROUTINE SETS IN CLR AND OUT CLR TO
5382                ; CLEAR THE TRANSMIT AND RECEIVE BCC REGISTERS
5383
5384 031320 012761 000200 000004  MOV      #BIT7,4(R1)      ; LOAD PORT4
5385 031326 104412          ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5386 031330 122112          122112      ; SET IN CLR!
5387 031332 104412          ROMCLK      ; NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5388 031334 122111          122111      ; SET OUT CLR!
5389 031336 000207          RTS      PC      ; RETURN

```

```

5390
5391
5392 031340      STFFCL:
5393                ;THIS SUBROUTINE ADDS ANY NECESSARY BIT STUFF CLOCK TICKS
5394                ;FIRST ARGUMENT IS CHAR, SECOND ARGUMENT IS SHIFT COUNT.
5395
5396 031340 010046      MOV      RO, -(SP)      ;SAVE RO
5397 031342 012500      MOV      (R5)+, RO      ;PUT CHAR IN RO
5398 031344 012537 001302      MOV      (R5)+, STMP2      ;PUT SHIFT COUNT IN STMP2
5399 031350 106000      1$:      RORB      RO      ;LOOK AT NEXT BIT
5400 031352 103403      BCS      2$      ;BR IF A MARK
5401 031354 005037 031536      CLR      BITCON      ;IT WAS A SPACE, CLEAR 1'S COUNTER
5402 031360 000412      BR      3$      ;CONTINUE
5403 031362 005237 031536      2$:      INC      BITCON      ;INC CONSECUTIVE 1'S COUNTER
5404 031366 022737 000005 031536      CMP      #5, BITCON      ;IS IT 5 YET?
5405 031374 001004      BNE      3$      ;BR IF NO
5406 031376 005037 031536      CLR      BITCON      ;YES! SO START AGAIN
5407 031402 104413 000001      DATA CLK, 1      ;GIVE EXTRA TICK TO STUFF ZERO
5408 031406 005337 001302      3$:      DEC      STMP2      ;DEC SHIFT COUNT
5409 031412 001356      BNE      1$      ;BR IF NOT DONE
5410 031414 012600      MOV      (SP)+, RO      ;RESTORE RO
5411 031416 000205      RTS      R5      ;RETURN
5412
5413
5414 031420      STFFCK:
5415                ;THIS SUBROUTINE CHECKS TO SEE IF TRANSMITTER
5416                ;IS STUFFING ZEROS WHEN IT SHOULD. FIRST ARGUMENT
5417                ;IS THE CHARACTER, SECOND ARGUMENT IS SHIFT COUNT.
5418
5419 031420 010046      MOV      RO, -(SP)      ;SAVE RO
5420 031422 012500      MOV      (R5)+, RO      ;PUT CHAR IN RO
5421 031424 012537 001302      MOV      (R5)+, STMP2      ;PUT SHIFT COUNT IN STMP2
5422 031430 106000      1$:      RORB      RO      ;SHIFT OUT NEXT BIT
5423 031432 103403      BCS      2$      ;BR IF IT IS A MARK
5424 031434 005037 031536      CLR      BITCON      ;IT WAS A SPACE, CLEAR 1'S COUNTER
5425 031440 000416      BR      3$      ;CONTINUE
5426 031442 005237 031536      2$:      INC      BITCON      ;INC CONSECUTIVE 1'S COUNTER
5427 031446 022737 000005 031536      CMP      #5, BITCON      ;5 IN A ROW YET?
5428 031454 001010      BNE      3$      ;BR IF NO
5429 031456 005037 031536      CLR      BITCON      ;YES, SO START OVER
5430 031462 104413 000001      DATA CLK, 1      ;EXTRA TICK TO STUFF ZERO
5431 031466 004737 027754      JSR      PC, GETSI      ;LOOK AT WINDOW
5432 031472 103001      BCC      3$      ;IS IT A ZERO, BR IF YES
5433 031474 104030      ERROR      30      ;NO, ERROR ZERO WAS NOT STUFFED
5434 031476 005337 001302      3$:      DEC      STMP2      ;DEC SHIFT COUNT
5435 031502 001352      BNE      1$      ;BR IF NOT DONE
5436 031504 012600      MOV      (SP)+, RO      ;RESTORE RO
5437 031506 000205      RTS      R5      ;RETURN
5438
5439
5440 031510      CTSDLY:
5441                ;THIS SUBROUTINE WASTES TIME UNTIL CTS SETS,
5442                ;BUT HOPEFULLY NOT SO LONG THAT THE SILO RUNS OUT
5443
5444 031510 010046      MOV      RO, -(SP)      ;SAVE RO
5445 031512 012700 000032      MOV      #32, RO      ;LOAD RO WITH COUNT

```

```

5446 031516 027777 147522 147520 15: CMP 2STKS,2STKS ;WASTE TIME
5447 031524 005300 DEC RC ;DECREMENT COUNTER
5448 031526 001373 BNE 15 ;DO IT AGAIN IF NOT = 0
5449 031530 012600 MOV (SP)+,RO ;RESTORE RO
5450 031532 000207 RTS PC ;RETURN
5451
5452
5453 031534 000176 FLAG: 18<01111110> ;FLAG CHARACTER
5454 031536 000000 BITCON: 0
5455 031540 000 125 252 MESDAT: .BYTE 0,125,252,377
5456 031543 377
5457 031544 001 002 004 FLTDAT: .BYTE 1,2,4,10,20,40,100,200,376,375,373,367,357,337,277,177
5458 031547 010 020 040
5459 031552 100 200 376
5460 031555 375 373 367
5461 031560 357 337 277
5462 031563 177
5463 031564 100 140 160 STUFDT: .BYTE 100,140,160,170,3,300,174,176,177,1
5464 031567 170 003 300
5465 031572 174 176 177
5466 031575 001
5467 031576 363 347 317 .BYTE 363,347,317,200,0,377,377,377,200,37
5468 031601 200 000 377
5469 031604 377 377 200
5470 031607 037
5471
5472 031610 046200 047111 020105 .EVEN
031646 046200 047111 020105 EM1: .ASCIZ <200>/LINE UNIT INITIALIZATION TEST/
031711 200 044514 042516 EM2: .ASCIZ <200>/LINE UNIT REGISTER READ/ONLY TEST/
031755 200 044514 042516 EM3: .ASCIZ <200>/LINE UNIT REGISTER WRITE/READ TEST/
032017 200 051124 047101 EM4: .ASCIZ <200>/LINE UNIT INTERNAL CLOCK FAILURE/
032047 200 042522 042503 EM5: .ASCIZ <200>/TRANSMITTER DATA ERROR/
032066 051200 041505 044505 EM6: .ASCIZ <200>/RECEIVER TEST/
032113 200 047515 042504 EM7: .ASCIZ <200>/RECEIVER DATA ERROR/
032137 200 051124 047101 EM10: .ASCIZ <200>/MODEM SIGNAL ERROR/
032166 051200 041505 044505 EM11: .ASCIZ <200>/TRANSMITTER CRC ERROR/
032212 044600 020116 041502 EM12: .ASCIZ <200>/RECEIVER CRC ERROR/
032252 052200 040522 051516 EM13: .ASCIZ <200>/IN BCC MATCH ERROR (LU REG 12)/
032322 041600 041101 042514 EM14: .ASCIZ <200>/TRANSMITTER FAILED TO GO TO MARK STATE/
032343 200 046106 043501 EM15: .ASCIZ <200>/CABLE DATA TEST/
032357 200 051124 047101 EM16: .ASCIZ <200>/FLAG ERROR/
032423 200 053523 052111 EM17: .ASCIZ <200>/TRANSMITTER FAILED TO STUFF A ZERO/
032444 040600 047502 052122 EM20: .ASCIZ <200>/SWITCH PAC TEST/
032461 200 051124 047101 EM21: .ASCIZ <200>/ABORT ERROR/
032504 044200 046101 020106 EM22: .ASCIZ <200>/TRANSMITTER ERROR/
032526 047600 052125 051040 EM23: .ASCIZ <200>/HALF DUPLEX TEST/
032551 200 047111 051040 EM24: .ASCIZ <200>/OUT READY NOT SET/
EM25: .ASCIZ <200>/IN READY NOT SET/
032573 200 054105 042520 DH1: .ASCIZ <200>/EXPECTED FOUND/
032614 042600 050130 041505 DH2: .ASCIZ <200>/EXPECTED FOUND LU-REGISTER/
032652 041600 040510 040522 DH3: .ASCIZ <200>/CHARACTER BIT THAT FAILED/
032710 041600 051117 042522 DH4: .ASCIZ <200>/CORRECT CRC BIT THAT FAILED/
032750 042600 050130 041505 DH5: .ASCIZ <200>/EXPECTED FOUND SHIFT/
033002 042600 050130 041505 DH6: .ASCIZ <200>/EXPECTED FOUND CHARACTER SHIFT/
033050 041200 047514 045503 DH7: .ASCIZ <200>/BLOCK END NOT SET/
033073 200 052122 020123 DH10: .ASCIZ <200>/RTS DID NOT CLEAR/

```

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 104
 DZKCE.P11 12-MAY-77 12:23 SUBROUTINES

PAGE: 0125

.EVEN

033116	000002		DT1:	2	
033120	003	007		.BYTE	3,7
033122	001274			\$REG5	
033124	003	002		.BYTE	3,2
033126	001272			\$REG4	
033130	000003		DT2:	3	
033132	003	007		.BYTE	3,7
033134	001274			\$REG5	
033136	003	010		.BYTE	3,10
033140	001272			\$REG4	
033142	003	002		.BYTE	3,2
033144	001266			\$REG2	
033146	000002		DT3:	2	
033150	003	017		.BYTE	3,17
033152	001274			\$REG5	
033154	002	002		.BYTE	2,2
033156	001270			\$REG3	
033160	000002		DT4:	2	
033162	006	C-1		.BYTE	6,21
033164	030756			CALBCC	
033166	002	002		.BYTE	2,2
033170	001270			\$REG3	
033172	000003		DT5:	3	
033174	001	011		.BYTE	1,11
033176	001462			ZERO	
033200	001	011		.BYTE	1,11
033202	001464			ONE	
033204	002	002		.BYTE	2,2
033206	001262			\$REG0	
033210	000003		DT6:	3	
033212	001	011		.BYTE	1,11
033214	001464			ONE	
033216	001	011		.BYTE	1,11
033220	001462			ZERO	
033222	002	002		.BYTE	2,2
033224	001262			\$REG0	
033226	000004		DT7:	4	
033230	001	011		.BYTE	1,11
033232	001462			ZERO	
033234	001	011		.BYTE	1,11
033236	001464			ONE	
033240	003	007		.BYTE	3,7
033242	001274			\$REG5	
033244	002	001		.BYTE	2,1
033246	001270			\$REG3	
033250	000004		DT10:	4	
033252	001	011		.BYTE	1,11
033254	001464			ONE	
033256	001	011		.BYTE	1,11
033260	001462			ZERO	
033262	003	007		.BYTE	3,7
033264	001274			\$REG5	
033266	002	001		.BYTE	2,1
033270	001270			\$REG3	

J13

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 105
DZKCE.P11 12-MAY-77 12:23 SUBROUTINES

PAGE: 0126

033272	000002		DT11:	2	
033274	003	007		.BYTE	3,7
033276	031534			FLAG	
033300	002	002		.BYTE	2,2
033302	001270			\$REG3	
033304	000002		DT12:	2	
033306	006	004		.BYTE	6,4
033310	030756			CALBCC	
033312	006	002		.BYTE	6,2
033314	001302			\$TMP2	

033316	000001		CJRMX:	
			.END	

K10

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 107

PAGE: 0127

DZKCE.P11 12-MAY-77 12:23

CROSS REFERENCE TABLE -- USER SYMBOLS

ABASE =	000000	266	307		
ACDRI =	000000	266	309		
ACDRI2 =	000000	266	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		
ADEVN =	000000	266	308		
ADRCNT =	006057	1410*	1425*	1434*	
ADYANC =	104420	1579*	4958	5041	
RENV =	000002	1*	266	277	
RENVN =	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			
APTCU =	000040	1135	1240*		
APTEMV =	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		
AUSTRY =	003126	839*			
AUSWR =	000000	266	280		
AUTO.S	012110	801	1958*		

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 108
DZKCE.P11 12-MAY-77 12:23 CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

M10

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 109

PAGE: 0129

DZKCE.P11 12-MAY-77 12:23

CROSS REFERENCE TABLE -- USER SYMBOLS

CONERR	010762	868	1779#														
COMM	010671	1779#	2035														
CONTAB	003330	877	885#														
CONVRT=	104416	820	876	1577#	1637	1980											
CORMAX	033316	5472#															
CR	= 000015	44#	1174	1184													
CRC.CC=	102010	5262#															
CRC16 =	120001	3646	3675	3723	3752	3800	3829	3877	3906	3954	4022	4109	4278	4329			
		4528	4598	4887	4983	5261#											
CREAM	G01502	354#	746*	1880*	1881	1883*	1887										
CRLF	= 000200	45#	1145	1184													
CSR	010363	1779#	1984														
CSRMAP	012112	1960#															
CTSDLY	031510	5440#															
CYCLE	011460	955	1016	1871#													
DATABP	006764	1610*	1613	1635	1638#												
DATACL=	104413	1574#	2709	2754	2758	2819	2820	2834	2872	2873	2887	2925	2926	2940			
		2978	2979	2993	3037	3040	3083	3113	3143	3173	3204	3254	3293	3332			
		3371	3412	3413	3458	3459	3498	3651	3652	3680	3681	3728	3729	3757			
		3758	3805	3806	3834	3835	3882	3883	3911	3912	3968	3970	4036	4038			
		4102	4116	4137	4155	4162	4164	4199	4271	4285	4306	4336	4357	4375			
		4382	4384	4395	4521	4535	556	4576	4605	4626	4644	4651	4653	4664			
		4786	4791	4797	4852	5407	5430										
DATAHD	006752	1609*	1631	1634#													
DOISP =	177570	51#	207#	236													
DELAY =	104411	1572#															
DEVTAB	003342	843	890#														
DH1	032573	404	407	410	446	452	458	485	5472#								
DH10	033073	467	5472#														
DH2	032614	386	389	392	398	470	479	5472#									
DH3	032652	401	449	464	5472#												
DH4	032710	419	5472#														
DH5	032750	413	416	425	428	482	5472#										
DH6	033002	431	434	437	440	5472#											
DH7	033050	461	5472#														
DISPLA	001242	236#	765*	771*	1083*												
DISPRE	000174	197#	771														
DSMR =	177570	50#	206#	235													

N10

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 110
DZKCE.Pil 12-MAY-77 12:23 CROSS REFERENCE TABLE -- USER SYMBOLS

PAGE: 0130

[illegible]

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 111
DZKCE.P11 12-MAY-77 12:23 CROSS REFERENCE TABLE -- USER SYMBOLS

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 111
DZKCE.P11 12-MAY-77 12:23 CROSS REFER

INIFLG	001506	359#	774	804	811*									
INLPI	005766	1412#	1421											
INPUT =	104415	1576#	1920	1971	1983	1991	2048	2056						
INRDY	030614	3414	3460	3499	3507	3515	4200	4213	4218	4398	4412	4417	4441	4455
		4460	4667	4681	4686	4707	4721	4735	4740	4798	4808	4816	4916	4932
		4937	5003	5018	5023	5212#								
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												
KMICM	011050	879	1779#											
KMICR00	002100	543#												
KMICR01	002110	548#												
KMICR02	002120	553#												
KMICR03	002130	558#												
KMICR04	002140	563#												
KMICR05	002150	568#												
KMICR06	002160	573#												
KMICR07	002170	578#												
KMICR10	002200	583#												
KMICR11	002210	588#												
KMICR12	002220	593#												
KMICR13	002230	598#												
KMICR14	002240	603#												
KMICR15	002250	608#												
KMICR16	002260	613#												
KMICR17	002270	618#												
KMICSR	002066	526#	839*	854	860*	889	1019	1086	1701	1755	1889*	1898	1946	2278*
KMICSAH	002070	527#	1736*	1737*	1741*	1747*	1748*	1754*	1756*	1898*	1899*	1900		
KMICTL	002072	528#	1900*	1901*	1902									
KMINUM	001472	349#	742	996	1964*	1978*	2118*	2119	2127	2129				
KMP04	002074	529#	1725*	1731	1768	1773	1902*	1903*	1904	5053	5066	5108	5224	5299*
		5309*	5310*											
KMP06	002076	530#	1742*	1904*	1905*									
KMRVL	002060	523#	989	930*	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#												

[illegible]

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 113
DZKCE.P11 12-MAY-77 12:23 CROSS REFERENCE TABLE -- USER SYMBOLS

MLCK	007716	951	1779											
MMEM	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	2701
		2746	2805	2858	2911	2964	3018	3078	3108	3138	3168	3199	3249	3288
		3327	3366	3405	3448	3494	3540	3592	3642	3719	3796	3873	3948	4016
		4086	4188	4251	4500	4776	4840	4881	4977					
MTITLE	001000	205	778											
MTSTN	010003	1620	1779	1921										
MVECX	007753	980	1779											
NEXT	001442	334	1439	1659	2276	2302	2327	2353	2393	2426	2469	2528	2581	2605
		2629	2665	2699	2744	2803	2856	2909	2962	3016	3076	3106	3136	3166
		3197	3247	3286	3325	3364	3403	3446	3492	3538	3590	3639	3716	3793
		3870	3946	4014	4084	4186	4249	4498	4774	4838	4879	4975		
NITCH	030004	5053	5054	5055	5056	5058								
NOACT	010731	790	1779	1873										
NODEV	003240	841	863											
NUM	010323	1779	1972											
OCOR	030006	2708	2753	2816	2869	2922	2975	3036	3967	4035	4100	4198	4269	4519
		4785	4790	4851	4911	4999	5061	5067	5097	5139	5155	5196	5287	5375
OK	003220	855	858	881										
ONE	001464	346	5472											
OUTRDY	030140	2807	2812	2860	2865	2913	2918	2966	2971	3023	3027	3032	3055	4095
		4193	4260	4509	4781	4847	5081	5090	5101	5126	5135	5148	5152	5172
		5181	5276	5285	5322	5338	5352	5369						
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				</								

[illegible]

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 115
DZKCE.P11 12-MAY-77 12:23 CROSS REFERENCE TABLE -- USER SYMBOLS

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 115
DZKCE.P11 12-MAY-77 12:23 CROSS REFERENCE

STRISM	001446	339#	780#	783#	784	786	800#	806	808	903	949	956	1914	1937#
STUFDT	031564	1967	2150											
STUFLG	030612	5463#												
SVOS	006100	3407#	3450#	5185	5199#	5209#								
SWFLG	011416	1450#												
SWR	001240	743#	1815#	1842#	1848									
		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	030024	3081	3111	3141	3171	5071#								
SYNLD	031122	3955	4023	4094	4192	4259	4508	4780	4846	4885	4886	4981	4982	5314#
TBITVE=	000014	134#												
TEMP	011106	1488	1752#	1757#	1763#	1775#	1794#	2632#	2638#	2641#	2647#			
TIMER =	104414	1575#	2674	3205	3255	3294	3333	3372	3549	3564	3601			
TKVEC =	000060	141#												
TLAST =	027434	1940	5042#											
TPVEC =	000064	142#												
TRAPVE=	000034	140#												
TRTVEC=	000014	135#												
TST1	013732	1047	1927</											

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 116
DZKCE.P11 12-MAY-77 12:23 CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 117
DZKCE.P11 12-MAY-77 12:23 CROSS REFERENCE TABLE -- USER SYMBOLS

XERR	004126	985	1026#											
XHEAD	010077	812	1779#											
XPASS	004120	983	1023#											
XPOLY	030754	3646#	3675#	3723#	3752#	3800#	3829#	3877#	3906#	3954#	4022#	4109#	4278#	4329#
		4528#	4598#	4887#	4983#	5248	5250	5258#						
XSTAT0	011060	821	1779#											
XTSTN	007120	1621	1666#											
XVEC	004112	981	1020#											
ZERO	001462	345#	5472											
\$APTHD	002034	497	503#											
\$ASTAT=	***** U	1218	1233											
\$ATYC	004722	1189	1191#											
\$ATY1	004676	1187#												
\$ATY3	004704	1133	1188#											
\$ATY4	004714	1190#	1643											
\$AUT08	001234	232#												
\$BASE	001372	307#	2225											
\$BOARD	001222	227#												
\$BOOAT	001225	229#												
\$CDW1	001376	309#	2222											
\$CDW2	001400	310#												
\$CHARC	004672	1150#	1160#	1167	1176#	1181#								
\$CKSWR=	***** U	1565												
\$CNTAG	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#	2685#	2688
		2692#	2730#	2733	2737#	2788#	2791	2796#	2841#	2844	2849#	2894#	2897	2902#
		2947#	2950	2955#	3000#	3003	3009#	3063#	3066	3069#	3093#	3096	3099#	3123#
		3126	3129#	3153#	3156	3159#	3183#	3186	3190#	3234#	3237	3240#	3273#	3276
		3279#	3312#	3315	3318#	3351#	3354	3357#	3390#	3393	3396#	3431#	3434	3439#
		3477#	3480	3485#	3524#	3527	3531#	3577#	3580	3583#	3625#	3628	3632#	3702#
		3705	3709#	3779#	3782	3786#	3856#	3859	3863#	3933#	3936	3939#	4001#	4004
		4007#	4069#											

SODW6	001416	317#																		
SODW7	001420	318#																		
SODW8	001422	319#																		
SODW9	001424	320#																		
SOEVCT	001326	272#																		
SOEVM	001374	308#	2228																	
SODAGN	004100	994	1003	1008	1014#															
SENDAD	004070	191	776	1010#	1646															
SENDOCT	004054	1005#																		
SENV	001336	277#	757	793	795	1128	1196	1220	1640	1806										
SENYM	001337	278#	798	1130	1135	1198														
SEOP	003662	973#	4975																	
SEOPCT	004046	1002#	1006																	
SERFLG	001203	218#	744*	976*	1038	1067	1069*	1090	1597*	1611	1625*	1699*								
SERMAX	001215	224#	1090																	
SERROR	006512	181	1586#																	
SERRPC	001216	225#	753*	997*	1046*	1594	1596*	1700*												
SERRTB	001512	379#	1607																	
SERTTL	001212	222#	988	1028	1654*	1896*														
SETABL	001336	276#																		
SETEND	001442	329#	509																	
SFATAL	001320	269#	1224*																	
SFFLG	005142	1187*	1190*	1218	1227*	1235#														
SFILLC	001256	243#	1153	1184																
SFILLS	001255	242#	1184																	
\$GDADR	001220	226#																		
\$GDOAT	001224	228#																		
\$GET42	004060	1007#																		
\$GTSMR=	***** U	1564																		
\$HD =	000000	11																		
\$HIBTS	002034	504#																		
\$HI OCT	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																
\$LF	001314	261#	1184	1329	1339	1399														
\$LFLG	005141	1228*	1234#																	
\$LPADR	001206	220#	755*	955*	959	1082*	1084	1089	1659*	1661	1935*	1945*	1947							
\$LPERR	001210	221#																		
\$MAOR1	001350	294#																		
\$MAOR2	001354	298#																		
\$MAOR3	001360	301#																		
\$MAOR4	001364	304#																		
\$MAIL	001316	267#	505	509	1081	1128														
\$MAMS1	001346	288#	2224																	
\$MAMS2	001352	296#																		
\$MAMS3	001356	299#																		
\$MAMS4	001362	302#																		
\$MBAOR	002036	505#																		
\$MFLG	005140	1188*	1194	1229*	1233#															
\$MNEW	005552	1343#	1821																	
\$MSGAD	001332	274#	1204*	1207																
\$MSGLG	001334	275#	1209*																	
\$MSGTY	001316	268#	1202	1210*	1222	1226*														

\$MSMR	005541	1341#	1818											
\$MTYP1	001347	289#												
\$MTYP2	001353	297#												
\$MTYP3	001357	300#												
\$MTYP4	001363	303#												
\$MXCNT	004362	1079	1089#											
\$N	= 000057	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#	2685	2692	2694	2701	2702#	2730
		2737	2739	2746	2747#	2798	2796	5, 38	2805	2806#	2841	2849	2851	2858
		2859#	2894	2902	2904	29.1	2912#	2947	2955	2957	2964	2965#	3000	3009
		3011	3018	3019#	3063	3069	3071	3078	3079#	3093	3099	3101	3108	3109#
		3123	3129	3131	3138	3139#	3153	3159	3161	3168	3169#	3183	3190	3192
		3199	3200#	3234	3240	3242	3249	3250#	3273	3279	3281	3288	3289#	3312
		3318	3320	3327	3328#	3351	3357	3359	3366	3367#	3390	3396	3398	3405
		3406#	3431	3439	3441	3448	3449#	3477	3485	3487	3494	3495#	3524	3531
		3533	3540	3541#	3577	3583	3585	3592	3593#	3625	3632	3634	3642	3643#
		3702	3709	3711	3719	3720#	3779	3786	3788	3796	3797#	3856	3863	3865
		3873	3874#	3933	3939	3941	3948	3949#	4001	4007	4009	4016	4017#	4069
		4077	4079	4086	4087#	4172	4179	4181	4188	4189#	4231	4242	4244	4251
		4252#	4479	4491	4493	4500	4501#	4759	4767	4769	4776	4777#	4825	4831
		4833	4840	4841#	4862	4872	4874	4881	4882#	4959	4968	4970	4977	4978#
		5042#												
\$NULL	001254	241#	1155	1184										
\$NWTST=	000000	2273#	2299#	2324#	2350#	2380#	2423#	2466#	2525#	2578#	2602#	2626#	2662#	2696#
		2741#	2800#	2853#	2906#	2959#	3013#	3073#	3103#	3133#	3163#	3194#	3244#	3283#
		3322#	3361#	3400#	3443#	3489#	3535#	3587#	3636#	3713#	3790#	3867#	3943#	4011#
		4081#	4183#	4246#	4495#	4771#	4835#	4876#	4972#					
\$OVER	004334	1051	1054	1065	1077	1083#								
\$PASS	001324	271#	975#	987	999#	1000#	1017	1025	1073	1090	1895#			
\$PASTM	002042	507#												
\$PWRON	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		
\$ROCHR	005144	1259#	1565											
\$RODEC=	***** U	1568												
\$ROLIN	005264	1287#	1566											
\$RODOCT	005564	1360#	1567											
\$RODZ	= 000007	1280#												
\$REGAD	001260	245#												
\$REGO	001262	247#	1455#	1460	5472									
\$REG1	001264	248#	875#	887	1454#	1461								
\$REG2	001266	249#	1453#	1462	5472									
\$REG3	001270	250#	1452#	1463	5472									
\$REG4	001272	251#	1451#	1464	5472									
\$REG5	001274	252#	1450#	1465	5472									
\$RTNAD	004102	1016#												
\$R2A	= ***** U	1568												
\$S	= 000061	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#	2699
		2702#	2744	2747#	2803	2806#	2856	2859#	2909	2912#	2962	2965#	3016	3019#
		3076	3079#	3106	3109#	3136	3139#	3166	3169#	3197	3200#	3247	3250#	3286
		3289#	3325	3328#	3364	3367#	3403	3406#	3446	3449#	3492	3495#	3538	3541#

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 121
DZKCE.P11 12-MAY-77 12:23 CROSS REFERENCE TABLE -- USER SYMBOLS

[illegible]

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 123
DZKCE.P11 12-MAY-77 12:23 CROSS REFERENCE TABLE -- MACRO NAMES

COMMON	144														
ENDCOM	144														
ERROR	38	2287	2312	2338	2368	2398	2411	2441	2454	2488	2509	2544	2562	2590	2614
	2640	2649	2683	2718	2728	2767	2777	2786	2825	2829	2839	2878	2882	2892	2931
	2935	2945	2984	2988	2998	3045	3049	3091	3121	3151	3181	3213	3225	3232	3263
	3271	3302	3310	3341	3349	3380	3388	3421	3467	3506	3514	3522	3560	3575	3623
	3661	3665	3690	3694	3738	3742	3767	3771	3815	3819	3844	3848	3892	3896	3921
	3925	3978	3982	4046	4050	4121	4125	4142	4146	4158	4167	4207	4226	4290	4294
	4311	4315	4341	4345	4362	4366	4378	4387	4405	4425	4434	4448	4468	4477	4540
	4544	4561	4565	4581	4585	4610	4614	4631	4635	4647	4656	4674	4694	4703	4714
	4728	4748	4757	4796	4805	4815	4823	4860	4923	4945	4954	5010	5031	5040	5112
	5228	5433													
ESCAPE	144														
GETPRI	144														
GETSWR	144														
HLT	157														
KHEND	1	960													
KMFRT	1														
MULT	44														
NEWTST	44	2273	2299	2324	2350	2380	2423	2466	2525	2578	2602	2626	2662	2696	2741
	2800	2853	2906	2959	3013	3073	3103	3133	3163	3194	3244	3283	3322	3361	3400
	3443	3489	3535	3587	3636	3713	3790	3867	3943	4011	4081	4183	4246	4495	4771
	4835	4876	4972												
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	2697	2742
	2801	2854	2907	2960	3014	3074	3104	3134	3164	3195	3245	3284	3323	3362	3401
	3444	3490	3536	3588	3637	3714	3791	3868	3944	4012	4082	4184	4247	4496	4772
	4836	4877	4973												
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	49							

N11

DZKCE MAC'11 27(1006) 01-JUN-77 10:03 PAGE 124

PAGE: 0143

OZKCE.P11 12-MAY-77 12:23

CROSS REFERENCE TABLE -- MACRO NAMES

\$BUFFE	18	1791													
\$CDATA	18	4862	4959												
\$CLOCK	18	2616													
\$COMP	18	3208	3220	3227	3258	3267	3297	3306	3336	3345	3375	3384	3502	3510	3518
	3512	3567	4221	4420	4463	4689	4743	4940	5026						
\$CRC	18	3644	3673	3721	3750	3798	3827	3875	3904						
\$CRCSH	18	3949	4017												
\$CYCLE	18	1862													
\$EMPTY	18	4759													
\$EOP	18	960													
\$ERTBL	18	381													
\$FINI	18	5042													
\$FLAG	18														
\$FLOAT	18	2474	2494	2533	2550										
\$GETPA	18														
\$HALF	18	4825													
\$HADE	18	11													
\$INACT	18	3063	3093	3123	3153										
\$INIT	18														
\$LINE1	18	2456	2515												
\$LU1	18	2263	2289	2314	2340										
\$LU12	18	2370													
\$LU17	18	2413													
\$MARHI	18														
\$MARK	18	3477													
\$MATCH	18	4210	4408	4451	4677	4731	4928	5014							
\$MOCK	18														
\$MODEM	18	3524													
\$MSG	18	1779													
\$MULT	18														
\$PAEN	18	974													
\$PATTE	18	3390	3431												
\$PFAIL	18	1669													
\$QOQI	18	5293	5303												
\$QUEST	18	1971	1983	1991	2048	2056									
\$RANCL	18	1723													
\$RCLK	18	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	2670
	2672	2676	2704	2706	2711	2721	2749	2751	2756	2760	2770	2780	2809	2814	2835
	2862	2867	2888	2915	2920	2941	2968	2973	2994	3025	3029	3034	3057	3084	3114
	3144	3174	3206	3216	3218	3256	3265	3295	3304	3334	3343	3373	3382	3415	3455
	3461	3500	3508	3516	3547	3550	3562	3565	3599	3603	3610	3957	3961	3965	3995
	4025	4029	4033	4063	4201	4214	4219	4228	4399	4413	4418	4427	4442	4456	4461
	4470	4668	4682												

SSSTUFF	18														
SSMPAC	18	2568	2592												
STCHAR	18	4090	4189	4255	4504										
STCRC	18	4069	4231	4479											
STRANW	18	4109	4278	4329	4528	4598									
STRANI	18	2651	2685	2730											
STSTN	18	2271	2297	2322	2348	2378	2421	2464	2523	2576	2600	2624	2660	2694	2739
	2798	2851	2904	2957	3011	3071	3101	3131	3161	3192	3242	3281	3320	3359	3398
	3441	3487	3533	3575	3634	3711	3788	3865	3941	4009	4079	4181	4244	4493	4769
	4833	4874	4970												
SUPADD	18	1697													
SVARIA	18	203													
SWINDO	18	2788	2841	2894	2947										
SWZ	18	2263	2269	2289	2295	2314	2320	2340	2346	2370	2376	2413	2419	2456	2462
	2515	2521	2568	2574	2592	2598	2616	2622	2651	2658	2685	2692	2730	2737	2798
	2796	2841	2949	2894	2902	2947	2955	3000	3009	3063	3069	3093	3099	3123	3129
	3153	3159	3183	3190	3234	3240	3273	3279	3312	3318	3351	3357	3390	3396	3431
	3439	3477	3485	3524	3531	3577	3583	3625	3632	3702	3709	3779	3786	3856	3863
	3933	3939	4001	4007	4069	4077	4172	4179	4231	4242	4479	4491	4759	4767	4825
	4831	4862	4872	4959	4968										
SSZEROS	18														
SSCHRE	2088	247	248	249	250	251	252								
SSCHTH	2088	253	254	255	256	257									
SSESCA	1448														
SSNEWT	1448	2273	2299	2324	2350	2380	2423	2465	2525	2578	2602	2626	2662	2696	2741
	2800	2853	2906	2959	3013	3073	3103	3133	3163	3194	3244	3283	3322	3361	3400
	3443	3489	3535	3587	3636	3713	3790	3867	3943	4011	4081	4183	4246	4495	4771
	4835	4876	4972												
SSSCOP	18	1029													
SSSET	15548	1565	1566	1567	1568	1569	1570	1571	1572	1573	1574	1575	1576	1577	1578
	1579														
SSSKIP	1448														
.EQUAT	18	34													
.HEADE	18														
.SETUP	18														
.SACT1	18	185													
.SAPT8	18	2638													
.SAPTH	18	488													
.SAPTY	18	1184													
.SCATC	18														
.SCHTA	18	208													
.SEOP	18	966													
.SERRO	18														
.SERRT	18														
.SPOWE	18	1672													
.SRDOC	18	1346													
.SREAO	18	1243													
.SSCOP	18	1033													
.STRAP	18	1531													
.STYPE	18	1105													
.STYPO	18														

. ABS. 033316 000

C12

DZKCE MACY11 27(1006) 01-JUN-77 10:03 PAGE 126
DZKCE.P11 12-MAY-77 12:23 CROSS REFERENCE TABLE -- MACRO NAMES

PAGE: 0145

ERRORS DETECTED: 0
DEFAULT GLOBALS GENERATED: 0

DZKCE0,DZKCE/SOL/CRF+DZKCE.MAC,DZKCE.P11/EQ:DZDME
RUN-TIME: 30 27 2 SECONDS
RUN-TIME RATIO: 2064/60=34.3
CORE USED: 53K (106 PAGES)