PDP11

BASIC MICROPROCESSOR TST

MD-11-DZKCC-A

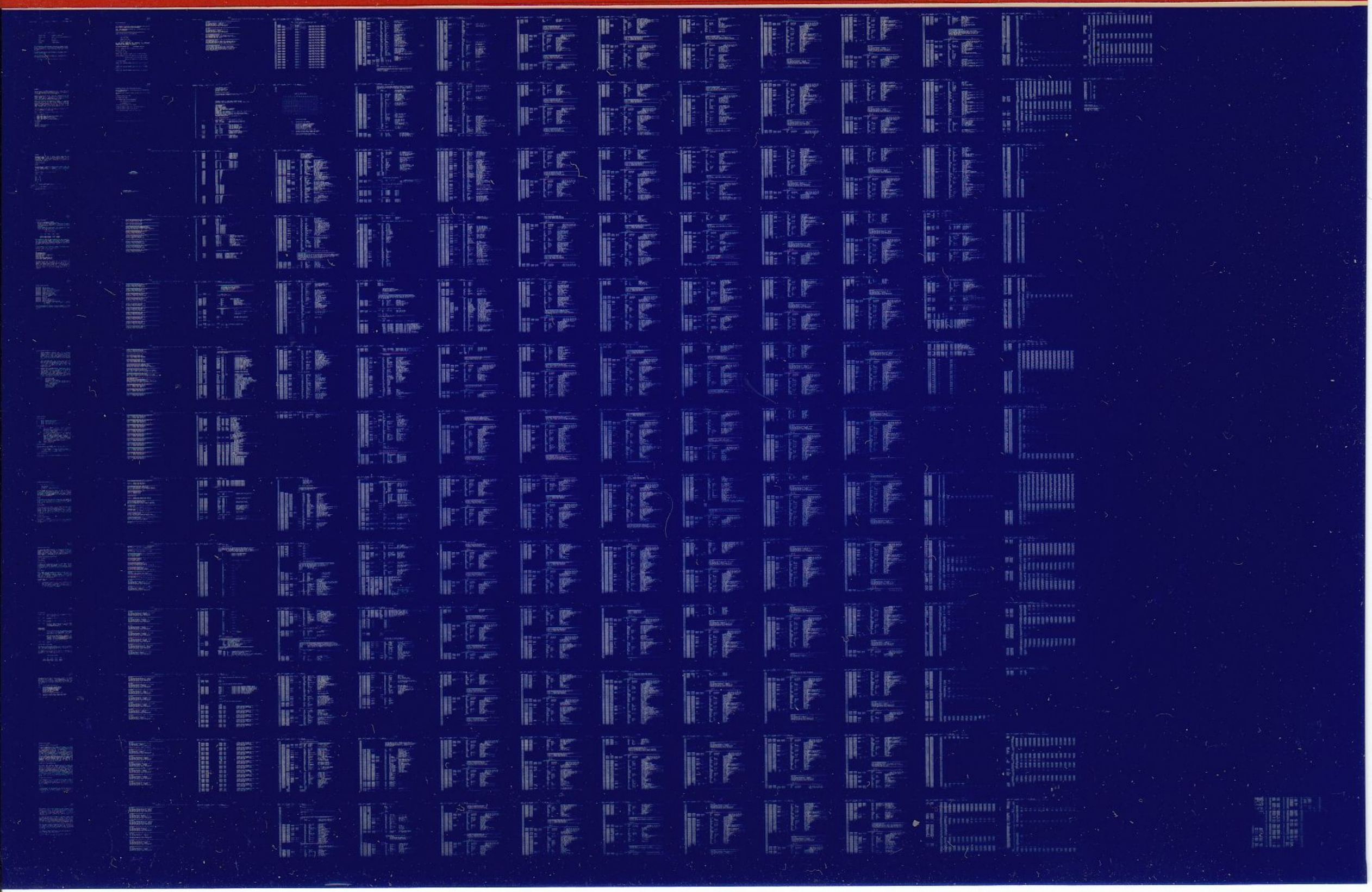
EP-DZKCC-A-DL-A

FICHE 1 OF 1

AUG 1977

digital

MADE IN USA



IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DZKCC-A-D

PRODUCT NAME: BASIC W/R AND MICRO-PROCESSOR TESTS

DATE: MAY 1977

MAINTAINER: DIAGNOSTICS

AUTHOR: DINESH GORADIA

1. ABSTRACT

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

DZKCC tests the KMC11 micro-processor (M8204). It performs write/read tests on the KMC unibus registers, checks the micro-processor operation, checks out Main Memory, scratch pad memory, the ALU functions as well as interrupts and NPR operation. DZKCC performs no tests on the line unit or any CRAM dependent tests. It will run on KMC11's containing CRAM (IOP). It does not require a line unit to run.

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 memory tests (Heat test tape)
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 equilivalent)
KMC11-AN IOP (M8204)

2.2 STORAGE

Program will use all 8K of memory except where ABL and BOOTSTRAP LOADER reside. Locations 2100 thru 2300; contain the "STATUS TABLE" information which is generated at start of diagnostics by manual input (questions) or automatically (auto-sizing). This area is an overlay area and should not be altered by the operator.

3. LOADING PROCEEDURE

3.1 METHOD

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

ABSOLUTE LOADER starting address *500

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

3.1.1 Place address of ABS loader into switch register.
(also place 'HALT' SW up)

3.1.2 Depress 'LOAD ADDRESS' key on console and release.

3.1.3 Depress 'START KEY' on console and release (program should now be loading into CPU)

4. STARTING PROCEEDURE

a. Set switch register to 000200
b. Depress 'LOAD ADDRESS' key and release
c. Set SWR to zero for 'AUTO SIZING' or SWR bit0=1 for manual input (questions) or SWR bit7=1 to use existing parameters set up by a previous start or a previously run KMC11 diagnostic.
d. Depress 'START KEY' and release. The program will type Maindec Name and program name (if this was the first start up of the program) and also the following:

MAP OF KMC11 STATUS

PC	CSR	STAT1	STAT2	STAT3
002100	160010	045310	177777	000000
002110	160020	045320	177777	000000

The program will type 'R' and proceed to run the diagnostic. The above is only an example. This would indicate the status table starting at add. 2100 in the program. In this example the table contains the information and status of two KMC11'S. THE STATUS TABLE MUST BE VERIFIED BY THE USER IF AUTO SIZING IS DONE. For information of status table see section 8.4 for help.

If the diagnostic was started with SW00=1 indicating manual parameter input then the following shows an example of the questions asked and some example answers:

```
HOW MANY KMC11'S TO BE TESTED?1

01
CSR ADDRESS?160010
VECTOR ADDRESS?310
BR PRIORITY LEVEL? (4,5,6,7)?5
WHICH LINE UNIT? IF NONE TYPE "N", IF M8201 TYPE "1", IF
M8202 TYPE "2"?1
IS THE LOOP BACK CONNECTOR ON?Y
SWITCH PAC#1 (DDCMP LINE#)?377
SWITCH PAC#2 (BM873 BOOT ADD)?377
```

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

4.1 CONTROL SWITCH SETTINGS

SW 15 Set: Halt on error
SW 14 Set: Loop on current test
SW 13 Set: Inhibit error print out
SW 12 Set: Inhibit type out abell on error.
SW 11 Set: Inhibit iterations. (quick pass)
SW 10 Set: Escape to next test on error
SW 09 Set: Loop with current data
SW 08 Set: Catch error and loop on it
SW 07 Set: Use previous status table.
SW 06 Set: Halt in ROMCLK routine before clocking micro-processor
SW 05 Set: Reserved
SW 04 Set: Reserved
SW 03 Set: Reselect KMC11's desired active
SW 02 Set: Lock on selected test
SW 01 Set: Restart program at selected test
SW 00 Set: Build new status table from questions. (If SW07=0 and SW00=0 a new status table is built by auto-sizing)

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

4.1.2 SWITCH REGISTER OPTIONS (at start up)

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

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

SW 03 RESELECT KMC11'S DESIRED ACTIVE. Please note that a message is typed out for setting the switch register equal to KMC11's active. this means if the system has four KMC11s; bits 00,01,02,03 will be set in loc 'KMACTV' from the switch register. Using this switch(SW00) alters that location;therefore if four KMC11s are in the system ***DO NOT*** set switchs greater than SW 03 in the up position. this would be a fatal error. do not select more active KMC11s than there is information on in the status table.

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

4.1.3 DYNAMIC SWITCHES

ERROR SWITCHES

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

SCOPE SWITCHES

1. SW06 Halt in ROMCLK routine before clocking micro-processor instruction. This allows the operator to scope a micro-processor instruction in the static state before it is clocked. Hit continue to resume running.
2. SW09 (if enabled by 'SCOP1') on an error; If an '*' is printed in front of the test no. (ex. *TEST NO. 10) SW09 is incorporated in that test and therefore SW09 is usually the best switch for the scope loop (SW14=0, SW10=0, SW09=1, SW08=0). If SW09 is not enabeled; and there is a HARD error (constant); SW08 is best. (SW14=1,0, SW10=0, SW09=0, SW08=1). for intermittemt errors; SW14=1 will loop on test reguardless of error or not error. (SW14=1, SW10=0, SW09=0, SW08=1,0)
3. SW11 Inhibit interations.
4. SW14 Loop on current test.

4.2 STARTING ADDRESS

Starting address is at 000200 there are no other starting addresses for the KMC11 diagnostics. (See Section 4.0)

NOTE: If address 000042 is non-zero the program assumes it is under ACT11 or XXDP control and will act accordingly after all available KMC11's are tested the program will return to 'XXDP' or 'ACT-11'.

5. OPERATING PROCEDURE

When program is initially started messages as described in section 4.0 will be printed, and program will begin running the diagnostic

5.2 PROGRAM AND/OR OPERATOR ACTION

The typical approach should be

1. Halt on error (via SW 15=1) when ever an error occurs.
2. Clear SW 15.
3. Set SW 14: (loop on this test)
4. Set SW 13: (inhibit error print out)

The TEST NUMBER and PC will be typed out and 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 reguardless of how program was started. Also it is important to use this listing along with the information printed on the TTY to completly isolate problems.

7.2 OPERATING RESTRICTIONS

The first time a KMC11 diagnostic is loaded into core and run the STATUS TABLE must be set up. This is done by manual input (SW00=1) or by autosizing (SW00=0 and SW07=0). Thereafter however the status table need not be setup by subsequent restarts or even loading the next KMC diagnostic because the STATUS TABLE is overlayed. The current parameters in the STATUS TABLE are used when SW07=1 on start up.

7.3 HARDWARE CONFIGURATION RESTRICTIONS

KMC11 IOP(M8204)- Jumper W1 must be in,

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 DZKCC CSR: 175000 VEC: 0300 PASSES: 000001
ERRORS: 000000

NOTE: The pass count and error counts are cummulitive for each KMC11 that is running, and are set to zero only when the diagnostic is started. Therefore after an overnight run for example, the total passes and errors for each KMC11 since the diagnostic was started are reflected in PASSES: and ERRORS:.

8.4 KEY LOCATIONS

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

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

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

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

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

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

KMACTV (1306) Each bit set in this location indicates that the associated KMC11 will be tested in turn. EXAMPLE: (KMACTV) 1470/0000000000011111 means that KMC11 no. 00,01,02,03,04 will be tested. EXAMPLE: (KMACTV) 1470/0000000000010001 Means that KMC11 no. 00,04 will be tested.

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

8.4A 'STATUS TABLE' (2100-2300)

The table is filled by AUTO SIZING or by the manual parameter input (questions) as described previously. Also if desired by user; the locations may be altered by hand (toggled in) to suit the specific configuration.

The example status map shown below contains information for two KMC11'S. the table can contain up to 16 KMC11'S. Following the map is a description of the bits for each map entry

MAP OF KMC11 STATUS

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

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

CSR: Contains KMC11 CSR address

STAT1: BITS 00-08 IS KMC11 VECTOR ADDRESS
BIT14=1 TURNAROUND CONNECTOR IS ON
BIT14=0 NO TURNAROUND CONNECTOR
BIT13=0 LINE UNIT IS AN M8201
BIT13=1 LINE UNIT IS AN M8202
BIT12=1 NO LINE UNIT
BITS 09-11 IS KMC11 BR PRIORITY LEVEL

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

STAT3: NOT USED

8.5 METHOD OF AUTO SIZING

8.5.1 FINDING THE CONTROL STATUS REGISTER.

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

8.5.2 FINDING THE VECTOR AND BR LEVEL

The vector area (address 300-776) is filled with the instruction IOT and '.+2' (next address). The processor status is started at 7 and the KMC is programmed to interrupt. The PS is lowered by 1 until the KMC interrupts, a delay is made and if no interupt occures at PS level 3 (because of a bad KMC11) the program assumes vector address 300 at BR level 5 and the problem should be fixed in the diagnostic. Once the problem is fixed; the program should be re-setup again to get correct vector. If an interupt occured; the address to which the KMC11 interupted to is picked up and reported as the vector. NOTE: if the vector reported is not the vector set up by you; there is a problem and AUTO SIZING should not be done.

8.6 SOFTWARE SWITCH REGISTER

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

Control:

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

SWR=XXXXXX NEW?

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

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

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

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

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

APT/ACT/XXDP/SLIDE

++++++++++++++++++++++++++

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

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

****

ETABLE SETTING FOR APT TO RUN UNDER APT

++

FIRST PASS TIME:

LONGEST TEST TIME:

ADDITIONAL TEST TIME:

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

SOFTWARE ENVIRONMENT:001 ENVIRONMENT MODE:200

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

SWITCH 2:-NOT USED.

CPU OPTIONS:-NOT USED.

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

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

BITS<2:4>:=BITS <12:14> OF STAT1 OF DEV:2

BITS<10:12>:=BITS<12:14> OF STAT1 OF DEV:3

IN THE SAME MANNER

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

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

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

INTERRUPT VECTOR 1:FIRST DEVICE RECEIVE VECTOR.

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

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

INTERRUPT VECTOR 2:NOT USED.

BUS PRIORITY:NOT USED.

BASE ADDRESS:FIRST DEVICE CSR ADDRESS.

REST SHOULD FOLLOW SEQUENTIALLY

IN INCREMENTS OF 10.

DEVICE MAP:AS DESCRIBED IN APT MANUAL.

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

CONTROLLER SPECIFIC CODE 2:-NOT USED.

DEVICE DESCRIPTOR WORD 0:STAT2 OF FIRST DEVICE.

. .

TO

. .

. .

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

DOCUMENT
****
MAINDEC-11-DZKCC-A
****

```
2225    ********** TEST 1 **********
        VERIFY THAT REFERENCING UNIBUS DEVICE REGISTERS
        DOES NOT CAUSE A TIME OUT TRAP

2256    ********** TEST 2 **********
        VERIFY THAT RUN CAN BE CLEARED

2275    ********** TEST 3 **********
        UNIBUS REGISTER WORD DUAL ADDRESSING TEST
        LOAD ALL REGISTERS WITH INCREMENTING PATTERN
        READ BACK ALL REGISTERS TO VERIFY CORRECT ADDRESSING

2318    ********** TEST 4 **********
        CONTROL STATUS REGISTER WRITE/READ TEST
        SET BIT0, VERIFY BIT0 WAS SET
        CLEAR BIT0, VERIFY BIT0 WAS CLEARED

2350    ********** TEST 5 **********
        CONTROL STATUS REGISTER WRITE/READ TEST
        SET BIT1, VERIFY BIT1 WAS SET
        CLEAR BIT1, VERIFY BIT1 WAS CLEARED

2382    ********** TEST 6 **********

2383    CONTROL STATUS REGISTER WRITE/READ TEST
        SET BIT2, VERIFY BIT2 WAS SET
        CLEAR BIT2, VERIFY BIT2 WAS CLEARED

2414    ********** TEST 7 **********
        CONTROL STATUS REGISTER WRITE/READ TEST
        SET BIT5, VERIFY BIT5 WAS SET
        CLEAR BIT5, VERIFY BIT5 WAS CLEARED

2446    ********** TEST 10 **********
        CONTROL STATUS REGISTER WRITE/READ TEST
        SET BIT6, VERIFY BIT6 WAS SET
        CLEAR BIT6, VERIFY BIT6 WAS CLEARED
```

```
2478    ************************** TEST 11 ****************************
        CONTROL STATUS REGISTER WRITE/READ TEST
        SET BIT7, VERIFY BIT7 WAS SET
        CLEAR BIT7, VERIFY BIT7 WAS CLEARED

2510    ************************** TEST 12 ****************************
        CONTROL STATUS REGISTER WRITE/READ TEST
        SET BIT9, VERIFY BIT9 WAS SET
        CLEAR BIT9, VERIFY BIT9 WAS CLEARED

2542    ************************** TEST 13 ****************************
        CONTROL STATUS REGISTER WRITE/READ TEST
        SET BIT11, VERIFY BIT11 WAS SET
        CLEAR BIT11, VERIFY BIT11 WAS CLEARED

2574    ************************** TEST 14 ****************************
        CONTROL STATUS REGISTER WRITE/READ TEST
        SET BIT12, VERIFY BIT12 WAS SET
        CLEAR BIT12, VERIFY BIT12 WAS CLEARED

2606    ************************** TEST 15 ****************************

2607    CONTROL OUT REGISTER WRITE/READ TEST
        SET BIT0, VERIFY BIT0 WAS SET
        CLEAR BIT0, VERIFY BIT0 WAS CLEARED

2638    ************************** TEST 16 ****************************
        CONTROL OUT REGISTER WRITE/READ TEST
        SET BIT1, VERIFY BIT1 WAS SET
        CLEAR BIT1, VERIFY BIT1 WAS CLEARED

2670    ************************** TEST 17 ****************************
        CONTROL OUT REGISTER WRITE/READ TEST
        SET BIT2, VERIFY BIT2 WAS SET
        CLEAR BIT2, VERIFY BIT2 WAS CLEARED

2702    ************************** TEST 20 ****************************
        CONTROL OUT REGISTER WRITE/READ TEST
        SET BIT6, VERIFY BIT6 WAS SET
        CLEAR BIT6, VERIFY BIT6 WAS CLEARED

2734    ************************** TEST 21 ****************************
        CONTROL OUT REGISTER WRITE/READ TEST
        SET BIT7, VERIFY BIT7 WAS SET
        CLEAR BIT7, VERIFY BIT7 WAS CLEARED

2766    ************************** TEST 22 ****************************
        CONTROL OUT REGISTER WRITE/READ TEST
        SET BIT12, VERIFY BIT12 WAS SET
        CLEAR BIT12, VERIFY BIT12 WAS CLEARED
```

```
2798    ********** TEST 23 **********
        CONTROL OUT REGISTER WRITE/READ TEST
        SET BIT13, VERIFY BIT13 WAS SET
        CLEAR BIT13, VERIFY BIT13 WAS CLEARED

2830    ********** TEST 24 **********

2831    PORT4 REGISTER WRITE/READ TEST
        FLOAT A ONE THROUGH PORT4 REGISTER
        FLOAT A ZERO THROUGH PORT4 REGISTER

2874    ********** TEST 25 **********
        PORT6 REGISTER WRITE/READ TEST
        FLOAT A ONE THROUGH PORT6 REGISTER
        FLOAT A ZERO THROUGH PORT6 REGISTER

2918    ********** TEST 26 **********
        UNIBUS REGISTER BYTE DUAL ADDRESSING TEST
        LOAD ALL REGISTERS WITH INCREMENTING PATTERN
        READ BACK ALL REGISTERS TO VERIFY CORRECT ADDRESSING

2961    ********** TEST 27 **********
        MAINTENANCE INSTRUCTION REGISTER TEST
        VERIFY THAT THE MAINT IR CAN BE WRITTEN TO ALL ZEROS'
        AND ALL ONES'. VERIFY THAT IT IS CLEARED ON A BUS RESET.

3003    ********** TEST 30 **********
        MAINTENANCE INSTRUCTION REGISTER TEST
        VERIFY THAT THE MAINT IR CAN BE WRITTEN TO ALL ZEROS'
        AND ALL ONES'. VERIFY THAT IT IS CLEARED ON A MASTER RESET.

3045    ********** TEST 31 **********
        MICRO PROCESSOR TEST
        LOAD KMP06 WITH A MICRO-PROCESSOR INSTRUCTION, CLOCK IT
        VERIFY INSTRUCTION EXECUTED PROPERLY
        INSTRUCTION SHOULD MOVE IBUS#4 TO IBUS#5, IBUS#4 IS ALL 1'S
        AND IBUS#5 IS ALL 0'S. RESULT SHOULD BE ALL 1'S IN SEL4

3074    ********** TEST 32 **********
        MICRO PROCESSOR IBUS# REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH IBUS# REGISTER 0
        FLOAT A 0 THROUGH IBUS# REGISTER 0

3131    ********** TEST 33 **********
        MICRO PROCESSOR IBUS# REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH IBUS# REGISTER 2
        FLOAT A 0 THROUGH IBUS# REGISTER 2

3188    ********** TEST 34 **********
        MICRO PROCESSOR IBUS# REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH IBUS# REGISTER 4
        FLOAT A 0 THROUGH IBUS# REGISTER 4
```

```
3241    ************************** TEST 35 ****************************
        MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH IBUS* REGISTER 5
        FLOAT A 0 THROUGH IBUS* REGISTER 5

3294    ************************** TEST 36 ****************************
        MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH IBUS* REGISTER 10
        FLOAT A 0 THROUGH IBUS* REGISTER 10

3351    ************************** TEST 37 ****************************
        MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH IBUS* REGISTER 11
        FLOAT A 0 THROUGH IBUS* REGISTER 11

3410    ************************** TEST 40 ****************************
        MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH IBUS REGISTER 0
        FLOAT A 0 THROUGH IBUS REGISTER 0

3463    ************************** TEST 41 ****************************
        MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH IBUS REGISTER 1
        FLOAT A 0 THROUGH IBUS REGISTER 1

3516    ************************** TEST 42 ****************************
        MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH IBUS REGISTER 2
        FLOAT A 0 THROUGH IBUS REGISTER 2

3569    ************************** TEST 43 ****************************
        MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH IBUS REGISTER 3
        FLOAT A 0 THROUGH IBUS REGISTER 3

3622    ************************** TEST 44 ****************************
        MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH IBUS REGISTER 4
        FLOAT A 0 THROUGH IBUS REGISTER 4

3675    ************************** TEST 45 ****************************
        MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH IBUS REGISTER 5
        FLOAT A 0 THROUGH IBUS REGISTER 5

3728    ************************** TEST 46 ****************************
        MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
        FLOAT A 1 THROUGH IBUS REGISTER 6
        FLOAT A 0 THROUGH IBUS REGISTER 6
```

```
3781    ************************** TEST 47 ****************************
        MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST

3783    FLOAT A 1 THROUGH IBUS REGISTER 7
        FLOAT A 0 THROUGH IBUS REGISTER 7

3834    ************************** TEST 50 ****************************
        MICRO PROCESSOR IBUS DUAL ADDRESS TEST
        WRITE ALL IBUS REGISTERS WITH INCREMENTING PATTERN
        READ ALL IBUS REGISTERS TO VERIFY CORRECT ADDRESSING

3897    ************************** TEST 51 ****************************
        MICRO PROCESSOR BR REGISTER TEST
        FLOAT A 1 THROUGH THE BR
        FLOAT A 0 THROUGH THE BR

3949    ************************** TEST 52 ****************************
        SCRATCH PAD TEST

3951    FLOAT A 1 THROUGH EACH SCRATCH PAD LOCATION
        FLOAT A 0 THROUGH EACH SCRATCH PAD LOCATION

4016    ************************** TEST 53 ****************************
        SCRATCH PAD DUAL ADDRESSING TEST
        WRITE AN INCREMENTING PATTERN IN ALL SP LOCATIONS
        READ ALL SP LOCATIONS TO VERIFY CORRECT ADDRESSING

4077    ************************** TEST 54 ****************************
        INTERRUPT TEST
        TEST THAT DEVICE CAN INTERRUPT TO VECTOR A

4108    ************************** TEST 55 ****************************
        INTERRUPT TEST
        TEST THAT DEVICE CAN INTERRUPT TO VECTOR B

4138    ************************** TEST 56 ****************************
        PRIORITY INTERRUPT TESTS
        SET PS TO ALL BR LEVELS EQUAL OR GREATER THAN
        THE KMC11 LEVEL,VERIFY THAT KMC11 DOES NOT INTERRUPT

4178    ************************** TEST 57 ****************************
        PRIORITY INTERRUPT TESTS
        SET PS TO ALL BR LEVELS LESS THAN THE KMC11 LEVEL
        VERIFY THAT THE KMC11 WILL INTERRUPT

4224    ************************** TEST 60 ****************************
        NPR TEST
        TEST OF DATO, 1 WORD FROM UPROC TO 11 MEMORY
```

```
4259    ********************** TEST 61 **********************
        NPR TEST
        TEST OF DATI, 1 WORD FROM 11 MEMORY TO UPROC

4297    ********************** TEST 62 **********************
        NPR TEST
        TEST OF DATOB, 1 BYTE FROM UPROC TO 11 MEMORY

4331    ********************** TEST 63 **********************
        TEST OF EA BITS 16 AND 17
        DO A DATO TO AN ADDRESS USING OUT BA BITS 16 AND 17
        VERIFY CORRECT RESULTS

4372    ********************** TEST 64 **********************
        TEST OF EA BITS 16 AND 17
        DO A DATI USING IN BA BITS 16 AND 17
        VERIFY CORRECT RESULTS

4410    ********************** TEST 65 **********************
        NPR NON-EXISTENT MEMORY TEST
        DO A DATO TO A NON-EXISTENT ADDRESS
        VERIFY THAT THE NON-EXISTENT BIT SET IN IBUS REG 11

4447    ********************** TEST 66 **********************
        NPR NON-EXISTENT MEMORY TEST
        DO A DATI FROM A NON-EXISTENT ADDRESS
        VERIFY THAT THE NON-EXISTENT BIT SET IN IBUS REG 11

4484    ********************** TEST 67 **********************
        NPR TEST
        USING DATO, NPR A BINARY COUNT (0-377 )
        FROM MICRO-PROCESSOR TO ALL AVAILABLE MEMORY

4546    ********************** TEST 70 **********************
        ALU C BIT TEST
        TEST THAT AN ADD OF 377 AND 377 WILL SET THE C BIT

4586    ********************** TEST 71 **********************
        ALU TEST
        TEST OF ALU FUNCTION SEL B WITH C BIT CLEARED
        ALU FUNCTION (B)    CODE=11
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

4637    ********************** TEST 72 **********************
        ALU TEST
        TEST OF ALU FUNCTION SEL A WITH C BIT CLEARED
        ALU FUNCTION (A)    CODE=10
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS
```

```
4688    ************************** TEST 73 ****************************
        ALU TEST
        TEST OF ALU FUNCTION A OR NOTB WITH C BIT CLEARED
        ALU FUNCTION (A OR NOTB)    CODE=12
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

4739    ************************** TEST 74 ****************************
        ALU TEST
        TEST OF ALU FUNCTION A AND B WITH C BIT CLEARED
        ALU FUNCTION (A AND B)    CODE=13
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

4790    ************************** TEST 75 ****************************

4791    ALU TEST
        TEST OF ALU FUNCTION A OR B WITH C BIT CLEARED
        ALU FUNCTION (A OR B)    CODE=14
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

4841    ************************** TEST 76 ****************************
        ALU TEST
        TEST OF ALU FUNCTION A XOR B WITH C BIT CLEARED
        ALU FUNCTION (A XOR B)    CODE=15
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

4892    ************************** TEST 77 ****************************
        ALU TEST
        TEST OF ALU FUNCTION ADD WITH C BIT CLEARED
        ALU FUNCTION (A PLUS B)    CODE=00
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

4943    ************************** TEST 100 ***************************
        ALU TEST
        TEST OF ALU FUNCTION 2A W/C WITH C BIT CLEARED
        ALU FUNCTION (A PLUS A PLUS C)    CODE=6
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

4994    ************************** TEST 101 ***************************
        ALU TEST
        TEST OF ALU FUNCTION SUB WITH C BIT CLEARED
        ALU FUNCTION (A-B)    CODE=16
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS
```

```
5045    ************************** TEST 102 ****************************
        ALU TEST
        TEST OF ALU FUNCTION ADD W/C WITH C BIT CLEARED
        ALU FUNCTION (A PLUS B PLUS C)    CODE=01
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5096    ************************** TEST 103 ****************************
        ALU TEST
        TEST OF ALU FUNCTION SUB W/C WITH C BIT CLEARED
        ALU FUNCTION (A-B-C)    CODE=2
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5147    ************************** TEST 104 ****************************
        ALU TEST
        TEST OF ALU FUNCTION INC A WITH C BIT CLEARED
        ALU FUNCTION (A PLUS 1)    CODE=3
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5198    ************************** TEST 105 ****************************
        ALU TEST
        TEST OF ALU FUNCTION 2A WITH C BIT CLEARED
        ALU FUNCTION (A PLUS A)    CODE=5
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5249    ************************** TEST 106 ****************************
        ALU TEST
        TEST OF ALU FUNCTION A PLUS C WITH C BIT CLEARED
        ALU FUNCTION (A PLUS C)    CODE=4
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5300    ************************** TEST 107 ****************************
        ALU TEST
        TEST OF ALU FUNCTION 2'S COMP SUB WITH C BIT CLEARED
        ALU FUNCTION (A-B-1)    CODE=17
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5351    ************************** TEST 110 ****************************
        ALU TEST
        TEST OF ALU FUNCTION DEC A WITH C BIT CLEARED
        ALU FUNCTION (A-1)    CODE=7
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS
```

```
5402    ************************** TEST 111 ****************************
        ALU TEST
        TEST OF ALU FUNCTION SEL B WITH C BIT SET
        ALU FUNCTION (B)    CODE=11
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA

5407    PERFORM THE FUNCTION, VERIFY THE RESULTS

5453    ************************** TEST 112 ****************************
        ALU TEST
        TEST OF ALU FUNCTION SEL A WITH C BIT SET
        ALU FUNCTION (A)    CODE=10
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5504    ************************** TEST 113 ****************************
        ALU TEST
        TEST OF ALU FUNCTION A OR NOTB WITH C BIT SET
        ALU FUNCTION (A OR NOTB)    CODE=12
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5555    ************************** TEST 114 ****************************
        ALU TEST
        TEST OF ALU FUNCTION A AND B WITH C BIT SET
        ALU FUNCTION (A AND B)    CODE=13
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5606    ************************** TEST 115 ****************************
        ALU TEST
        TEST OF ALU FUNCTION A OR B WITH C BIT SET
        ALU FUNCTION (A OR B)    CODE=14
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5657    ************************** TEST 116 ****************************
        ALU TEST
        TEST OF ALU FUNCTION A XOR B WITH C BIT SET
        ALU FUNCTION (A XOR B)    CODE=15
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5708    ************************** TEST 117 ****************************
        ALU TEST
        TEST OF ALU FUNCTION ADD WITH C BIT SET
        ALU FUNCTION (A PLUS B)    CODE=00
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS
```

```
5759    ************************** TEST 120 ****************************
        ALU TEST
        TEST OF ALU FUNCTION 2A W/C WITH C BIT SET
        ALU FUNCTION (A PLUS A PLUS C)    CODE=6
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5810    ************************** TEST 121 ****************************
        ALU TEST
        TEST OF ALU FUNCTION SUB WITH C BIT SET
        ALU FUNCTION (A-B)    CODE=16
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5861    ************************** TEST 122 ****************************
        ALU TEST
        TEST OF ALU FUNCTION ADD W/C WITH C BIT SET
        ALU FUNCTION (A PLUS B PLUS C)    CODE=01
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5912    ************************** TEST 123 ****************************
        ALU TEST
        TEST OF ALU FUNCTION SUB W/C WITH C BIT SET
        ALU FUNCTION (A-B-C)    CODE=2
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

5963    ************************** TEST 124 ****************************
        ALU TEST
        TEST OF ALU FUNCTION INC A WITH C BIT SET
        ALU FUNCTION (A PLUS 1)    CODE=3

5967    LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

6014    ************************** TEST 125 ****************************
        ALU TEST
        TEST OF ALU FUNCTION 2A WITH C BIT SET
        ALU FUNCTION (A PLUS A)    CODE=5
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

6065    ************************** TEST 126 ****************************
        ALU TEST
        TEST OF ALU FUNCTION A PLUS C WITH C BIT SET
        ALU FUNCTION (A PLUS C)    CODE=4
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS
```

```
6116    ********************** TEST 127 **********************
        ALU TEST
        TEST OF ALU FUNCTION 2'S COMP SUB WITH C BIT SET
        ALU FUNCTION (A-B-1)    CODE=17
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

6167    ********************** TEST 130 **********************
        ALU TEST
        TEST OF ALU FUNCTION DEC A WITH C BIT SET
        ALU FUNCTION (A-1)    CODE=7
        LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
        PERFORM THE FUNCTION, VERIFY THE RESULTS

6218    ********************** TEST 131 **********************
        TEST OF PROGRAM CLOCK BIT
        DO A MASTER CLEAR, VERIFY THAT PROGRAM CLOCK IS SET
        WRITE PROGRAM CLOCK BIT TO A ONE, VERIFY THAT IT CLEARS,
        AND THEN SETS SOME TIME LATER

6269    ********************** TEST 132 **********************
        FORCE POWER FAIL TEST
        SET FORCE POWER FAIL BIT VERIFY THAT PROCESSOR TRAPS TO 24
        GOING DOWN AND COMING UP. VERIFY ALSO THAT BUS INIT WAS
        BLOCKED FROM GETTING TO THE KMC DURING THE POWER FAIL

6317    ********************** TEST 133 **********************
        MICRO-PROCESSOR NOISE TEST
        WRITE ALL ZERO'S THEN ALL ONE'S THEN A DATA PATTERN
        TO THE IBUS* AND IBUS REGISTERS AND TO THE SP AND MAIN MEM
        THEN GO BACK AND READ THE DATA PATERNS  TO VERIFY THAT
        READING AND WRITING OF OTHER LOCATIONS AND REGISTERS
        DID NOT CHANGE THE DATA.
```

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

                                ;STARTING PROCEDURE
                                ;LOAD PROGRAM
                                ;LOAD ADDRESS 000200
                                ;SWR=0  AUTOSIZE KMC11
                                ;SW07=1  USE CURRENT KMC11 PARAMETERS
                                ;SW00=1  INPUT NEW KMC11 PARAMETERS
                                ;PRESS START
                                ;PROGRAM WILL TYPE "MAINDEC-11-DZKCC-A   BASIC KMC11 CONTROLLER TEST"
                                ;PROGRAM WILL TYPE STATUS MAP
                                ;PROGRAM WILL TYPE "R" TO INDICATE THAT TESTING HAS STARTED
                                ;AT THE END OF A PASS, PROGRAM WILL TYPE PASS COMPLETE MESSAGE
                                ;AND THEN RESUME TESTING
                                ;SUBSEQUENT RESTARTS WILL NOT TYPE PROGRAM TITLE

                                .SBTTL  BASIC DEFINITIONS

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

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

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

```
                                ;*GENERAL REGISTER DEFINITIONS
000003                  R3=     %3              ;;GENERAL REGISTER
000004                  R4=     %4              ;;GENERAL REGISTER
000005                  R5=     %5              ;;GENERAL REGISTER
000006                  R6=     %6              ;;GENERAL REGISTER
000007                  R7=     %7              ;;GENERAL REGISTER
000006                  SP=     %6              ;;STACK POINTER
000007                  PC=     %7              ;;PROGRAM COUNTER

                                ;*PRIORITY LEVEL DEFINITIONS
000000                  PR0=    0               ;;PRIORITY LEVEL 0
000040                  PR1=    40              ;;PRIORITY LEVEL 1
000100                  PR2=    100             ;;PRIORITY LEVEL 2
000140                  PR3=    140             ;;PRIORITY LEVEL 3
000200                  PR4=    200             ;;PRIORITY LEVEL 4
000240                  PR5=    240             ;;PRIORITY LEVEL 5
000300                  PR6=    300             ;;PRIORITY LEVEL 6
000340                  PR7=    340             ;;PRIORITY LEVEL 7

                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
100000                  SW15=   100000
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
000002                  SW01=   2
000001                  SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

                                ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
```

```
                        
000100                  BIT06=  100
000040                  BIT05=  40
000020                  BIT04=  20
000010                  BIT03=  10
000004                  BIT02=  4
000002                  BIT01=  2
000001                  BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0
                        
                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                        
                        
                        
                        ;INSTRUCTION DEFINITIONS
                        ;----------------------
                        
005746                  PUSH1SP=5746    ;DECREMENT PROCESSOR STACK 1 WORD
005726                  POP1SP=5726     ;INCREMENT PROCESSOR STACK 1 WORD
010046                  PUSHR0=10046    ;SAVE R0 ON STACK
012600                  POPR0=12600     ;RESTORE R0 FROM STACK
024646                  PUSH2SP=24646   ;DECREMENT STACK TWICE
022626                  POP2SP=22626    ;INCREMENT STACK TWICE
                        .EQUIV  EMT,HLT ;BASIC DEFINITION OF ERROR CALL
                        
                        
                        
```

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

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

            000020              .=20
000020  004134                          $SCOPE          ; SCOPE LOOP HANDLER.
000022  000340                          PR7             ; SERVICE AT LEVEL 7.
000024  007126                          $PWRDN                  ;POWER FAIL HANDLER
000026  000340                          PR7                     ;SERVICE AT LEVEL 7
000030  006512                          $ERROR                  ;ERROR HANDLER
000032  000340                          PR7                     ;SERVICE AT LEVEL 7
000034  006414                          $TRAP                   ;GENERAL HANDLER DISPATCH SERVICE
000036  000340                          PR7                     ;SERVICE AT LEVEL 7
                                .SBTTL  ACT11 HOOKS
                                ;;*****************************************************************
                                ;HOOKS REQUIRED BY ACT11
            000040                      $SVPC=.                 ;SAVE PC
            000046                      .=46
000046  004070                          $ENDAD                  ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
            000052                      .=52
000052  000000                          .WORD   0               ;;2)SET LOC.52 TO ZERO
            000040                      .=$SVPC                 ;; RESTORE PC

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

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


            001000              .=1000
001000  005200  040515  047111  MTITLE: .ASCII  <200><12>/MAINDEC-11-DZKCC-A/<200>
(2) 001025     102  051501  041511          .ASCIZ  /BASIC KMC11 CONTROLLER TEST/<200>
(2)
            177570              DSWR    =       177570
            177570              DDISP   =       177570
```

```
                                .SBTTL  COMMON TAGS

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

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

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

```
320  001424  000000                SDDW9:  .WORD   ADDW9   ;;DEVICE DESCRIPTOR WORD#9
321  001426  000000                SDDW10: .WORD   ADDW10  ;;DEVICE DESCRIPTOR WORD#10
322  001430  000000                SDDW11: .WORD   ADDW11  ;;DEVICE DESCRIPTOR WORD#11
323  001432  000000                SDDW12: .WORD   ADDW12  ;;DEVICE DESCRIPTOR WORD#12
324  001434  000000                SDDW13: .WORD   ADDW13  ;;DEVICE DESCRIPTOR WORD#13
325  001436  000000                SDDW14: .WORD   ADDW14  ;;DEVICE DESCRIPTOR WORD#14
326  001440  000000                SDDW15: .WORD   ADDW15  ;;DEVICE DESCRIPTOR WORD#15
327
328
329  001442                        SETEND:
330
331                                ;
332                                ;  PROGRAM CONTROL PARAMETERS
333                                ;-------------------------------
334  001442  000000                NEXT:   .WORD   0               ; ADDRSS OF NEXT TEST TO BE EXECUTED
335  001444  000000                LOCK:   .WORD   0               ; ADDRESS FOR LOCK CURRENT DATA
336                                ;
337                                ;   PROGRAM VARIABLES
338                                ;-----------------------
339  001446  000000                STRTSW: .WORD   0               ; SWITCHES AT START OF PROGRAM
340  001450  000000                STAT:   .WORD   0               ; KM STATUS WORD STORAGE
341  001452  000000                CLKX:   .WORD   0               ;
342  001454  000000                MASKX:  .WORD   0               ;
343  001456  000000                SAVSP:  .WORD   0               ; STACK POINTER STORAGE
344  001460  000000                SAVPC:  .WORD   0               ; PROGRAM COUNTER STORAGE
345  001462  000000                ZERO:   .WORD   0               ;
346  001464  000001                ONE:    .WORD   1               ;
347  001466  000000                MEMLIM: .WORD   0               ; HIGHEST LOCATION FOR NPR'S
348  001470  000001                KMACTV: .BLKW   1               ; KMC11 SELECTED ACTIVE
349  001472  000001                KMNUM:  .BLKW   1               ; OCTAL NUMBER OF KMC11'S
350  001474  000001                SAVACT: .BLKW   1               ; ORIGINAL ACTIVE DEVICES.
351  001476  000001                SAVNUM: .BLKW   1               ; WORKABLE NUMBER.
352  001500  000000                RUN:    .WORD   0               ; POINTER TO RUNNING DEVICES
353                                        .EVEN
354  001502  002072                CREAM:  .WORD   KM.MAP-6        ; TABLE POINTER
355  001504  002276                MILK:   .WORD   CNT.MAP-4       ; TABLE POINTER
356                                ;
357                                ;  PROGRAM CONTROL FLAGS
358                                ;-----------------------
359  001506     000                INIFLG: .BYTE   0               ; PROGRAM INITIALIZING FLAG
360          001510                        .EVEN
361  001510     000                LOKFLG: .BYTE   0               ; LOCK ON CURRENT TEST FLAG
362  001511     000                QV.FLG: .BYTE   0               ; QUICK VERIFY FLAG
363                                                               ; ON FIRST PASS OF EACH KMC11 ITERATIONS WILL BE SUPPRES
364                                        .EVEN
```

```
                                .SBTTL  ERROR POINTER TABLE

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

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


001512                          $ERRTB:
                                .EVEN
                                ;*      DF              ;; DOES NOT APPLY IN THIS DIAGNOSTIC.
001512  000000                          0
001514  000000                          0
001516  000000                          0
001520  035746                          EM1
001522  036632                          DH1             ; ERROR 1
001524  037112                          DT1
001526  036013                          EM2
001530  036673                          DH2             ; ERROR 2
001532  037124                          DT2
001534  036054                          EM3
001536  036731                          DH3             ; ERROR 3
001540  037142                          DT3
001542  036102                          EM4
001544  036752                          DH4             ; ERROR 4
001546  037154                          DT4
001550  036102                          EM4
001552  037014                          DH5             ; ERROR 5
001554  037172                          DT5
001556  036143                          EM5
001560  036731                          DH3             ; ERROR 6
001562  037210                          DT6
001564  036165                          EM6
001566  037055                          DH6             ; ERROR 7
001570  037172                          DT5
001572  036207                          EM7
001574  000000                          0               ; ERROR 10
001576  000000                          0
001600  036243                          EM10
001602  000000                          0               ; ERROR 11
001604  000000                          0
001606  036307                          EM11
001610  036731                          DH3             ; ERROR 12
001612  037142                          DT3
001614  036321                          EM12
001616  037055                          DH6             ; ERROR 13
001620  037172                          DT5
001622  036343                          EM13
001624  037055                          DH6             ; ERROR 14
001626  037172                          DT5
```

```
421  001630  036355                          EM14
422  001632  036731                          DH3          ; ERROR 15
423  001634  037210                          DT6
424  001636  036367                          EM15
425  001640  036752                          DH4          ; ERROR 16
426  001642  037154                          DT4
427  001644  036413                          EM16
428  001646  000000                          0            ; ERROR 17
429  001650  000000                          0
430  001652  036443                          EM17
431  001654  000000                          0            ; ERROR  20
432  001656  000000                          0
433  001660  036307                          EM11
434  001662  037055                          DH6          ; EPROR 21
435  001664  037172                          DT5
436  001666  036471                          EM20
437  001670  036731                          DH3          ; ERROR 22
438  001672  037222                          DT7
439  001674  036524                          EM21
440  001676  036731                          DH3          ; ERROR 23
441  001700  037142                          DT3
442  001702  036471                          EM20
443  001704  000000                          0            ; ERROR 24
444  001706  000000                          0
445  001710  036573                          EM22
446  001712  036731                          DH3          ; ERROR 25
447  001714  037142                          DT3
448          002034          .=2034
449                          .SBTTL  APT PARAMETER BLOCK
450
451                          ;;*****************************************************************
452                          ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
453                          ;;*****************************************************************
454          002034                  .$X=.       ;;SAVE CURRENT LOCATION
455          000024                  .=24        ;;SET POWER FAIL TO POINT TO STAPT OF PROGRAM
456  000024  000200                  200         ;;FOR APT START UP
457          000044                  .=44        ;;POINT TO APT INDIRECT ADDRESS PNTR.
458  000044  002034                  $APTHDR     ;;POINT TO APT HEADER BLOCK
459          002034                  .=.$X       ;;RESET LOCATION COUNTER
460                          ;;*****************************************************************
461                          ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
462                          ;INTERFACE SPEC.
463
464  002034                  $APTHD:
465  002034  000000          $HIBTS: .WORD   0           ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
466  002036  001316          $MBADR: .WORD   $MAIL       ;;ADDRESS OF APT MAILBOX (BITS 0-15)
467  002040  000132          $TSTM:  .WORD   90.         ;;RUN TIM OF LONGEST TEST
468  002042  000137          $PASTM: .WORD   95.         ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
469  002044  000137          $UNITM: .WORD   95.         ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
470  002046  000052                  .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
471
```

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

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

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

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

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

                                ;TEMP:  0
                                ;.=.+40

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

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

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

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

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

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

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

584 002300 000000 KM.END: 000000

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

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

```
638
639
640
641
642
643
```

FORMAT OF STATUS TABLE

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

DEFINITION OF FORMAT

CSR: CONTAINS KMC11 CSR ADDRESS

STAT1: BITS 00-08 IS KMC11 VECTOR ADDRESS
BIT14=1 ???? TURNAROUND CONNECTOR IS ON
BIT14=0 NO TURNAROUND CONNECTOR
BIT13=0 LINE UNIT IS AN M8201
BIT13=1 LINE UNIT IS AN M8202
BIT12=1 NO LINE UNIT
BITS 09-11 IS KMC11 BR PRIORITY LEVEL

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

STAT3: BIT0=1 DO FREE RUNNING TESTS ON KMC
(MUST BE SET TO A ONE MANUALLY [PROGRAMS G AND H ONLY])

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

002402  012737  000340  177776  .START: MOV     #340,PS         ;LOCK OUT INTERRUPTS
002410  012706  001200                  MOV     #STACK,SP       ;SET UP STACK
002414  012737  007126  000024          MOV     #$PWRDN,@#24    ;SET UP POWER FAIL VECTOR
002422  013737  001472  001476          MOV     KMNUM,SAVNUM    ;SAVE NUMBER OF DEVICES IN SYSTEM.
002430  005037  011416                  CLR     SWFLG           ;CLEAR SOFT TYPEOUT FLAG
002434  105037  001203                  CLRB    $ERFLG          ;CLEAR ERROR FLAG
002440  105037  001511                  CLRB    QV.FLG          ;ZERO QUICK VERIFY FLAG
002444  012737  002070  001502          MOV     #KM.MAP-10,CREAM;GET MAP POINTER.
002452  012737  002276  001504          MOV     #CNT.MAP-4,MILK ;GET PASS COUNT MAP POINTER
002460  012737  100000  001500          MOV     #BIT15,RUN      ;POINT POINTER TO FIRST DEVICE.
002466  012700  002302                  MOV     #CNT.MAP,R0     ;PASS COUNT POINTER TO R0
002472  005020                  23$:     CLR     (R0)+           ;CLEAR TABLE
002474  022700  002402                  CMP     #CNT.MAP+100,R0 ;DONE YET?
002500  001374                  BNE     23$             ;KEEP GOING
002502  005037  001216                  CLR     $ERRPC          ;CLEAR LAST ERROR POINTER
002506  012737  000001  001202          MOV     #1,$TSTNM       ;SET UP FOR TEST 1
002514  012737  002402  001206          MOV     #.$TAPT,$LPADR  ;SET UP FOR POWER FAIL BEFORE
                                                ;TESTING STARTS
002522  132737  000001  001336          BITB    #1,$ENV         ; IS IT RUNNING UNDER APT?
002530  001404                  BEQ     3$              ; IF NOT CHECK FOR TYPE OF SWITCH REGISTER.
002532  013737  001340  000176          MOV     $SWREG,SWREG    ; LOAD SOFTWARE SWITCH REG.
002540  000423                  BR      6$+2            ; GO SET UP SOFTWARE SWITCH REG.
002542  013746  000006          3$:     MOV     @#6,-(SP)       ;SAVE CURRENT VECTORS
002546  013746  000004                  MOV     @#4,-(SP)
002552  012737  002606  000004          MOV     #6$,@#4         ;SET UP FOR TIMEOUT
002560  012737  177570  001240          MOV     #177570,SWR     ;SET SWR TO HARD SWR ADDRESS
002566  012737  177570  001242          MOV     #177570,DISPLAY ;SET DISPLAY TO HARD SWR ADDRESS
002574  022777  177777  176436          CMP     #-1,@SWR        ;REFERENCE HARDWARE SWITCH REGISTER
002602  001402                  BEQ     6$+2            ;IF = -1 USE SOFT SWR ANYWAY
002604  000407                  BR      7$              ;IF IT EXISTS AND NOT = -1 USE HARD SWR
002606  022626                  6$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
002610  012737  000176  001240          MOV     #$SWREG,SWR     ;POINTER TO SOFT SWR
002616  012737  000174  001242          MOV     #$DISPREG,DISPLAY;POINTER TO SOFT DISPLAY REG
002624  012637  000004          7$:     MOV     (SP)+,@#4       ;RESTORE VECTORS
002630  012637  000006                  MOV     (SP)+,@#6
002634  105737  001506                  TSTB    INIFLG          ;HAS INITIALIZATION BEEN PERFORMED
002640  001006                  BNE     20$             ;BR IF YES
002642  022737  004070  000042          CMP     #$ENDAD,@#42    ;IF ACT-11 AUTOMATIC MODE, DON'T TYPE ID
002650  001402                  BEQ     20$
002652  104401  001000                  TYPE    ,MTITLE         ;TYPE TITLE MESSAGE
002656  004737  011212          20$:    JSR     PC,CKSWR                ;CHECK FOR SOFT SWR
002662  017737  176352  001446          MOV     @SWR,STRTSW     ;STORE STARTING SWITCHES
002670  005737  000042                  TST     @#42            ;IS IT RUNNING IN AUTO MODE?
002674  001402                  BEQ     .+6             ;BR IF NO
002676  005037  001446                  CLR     STRTSW          ;IF YES, CLEAR SWITCHES
002702  032737  000001  001446          BIT     #SW00,STRTSW    ;IF SW00=1, QUESTIONS ARE ASKED.
002710  001012                  BNE     17$             ;BR IF SW00=1
002712  105737  001446                  TSTB    STRTSW          ;BIT7=1??
```

```
                                BPL     17$             ;BR IF SW07=0
                                TST     KMACTV          ;ARE ANY DEVICES SELECTED?
                                BNE     16$             ;BR IF YES
                                TYPE,   NOACT           ;NO DEVICES SELECTED.
                                HALT                    ;STOP THE SHOW
                                BR      .-2             ;DISQUALIFY CONTINUE SWITCH
                        17$:    TSTB    $ENV            ; IS IT UNDER APT DUMP MODE?
                                BEQ     27$             ; YES, CHECK IF APT SIZED IT?
                                BITB    #1,$ENV         ; IS IT UNDER Q,V OR RUN MODE?
                                BNE     30$             ; YES, NEEDS ONLY APT SIZING.
                                BR      33$             ; NO, NEEDS REGULAR AUTO.SIZE.
                        27$:    TSTB    $ENVM           ; IS IT SIZED BY APT?
                                BMI     30$             ; YES, NEEDS ONLY APT SIZING.
                                BIC     #SW00,STRTSW    ; SIZE ONLY IN AUTO MODE.
                        33$:    JSR     PC,AUTO.SIZE    ; GO DO THE AUTO.SIZE.
                                BR      16$             ; GO PRINT THE MAP.
                        30$:    JSR     PC,APT.SIZE     ; GO DO THE APT SIZING.
                        16$:    TSTB    INIFLG          ;FIRST TIME?
                                BEQ     21$             ;BR IF YES
                                TSTB    STRTSW          ;IF USING SAME PARAMETERS DONT TYPE MAP
                                BMI     1$
                                BIT     #BIT1!BIT2,STRTSW;IS TEST NO. OR LOCK SELECTED
                                BEQ     24$             ;IF NO THEN TYPE STATUS
                                BR      1$              ;IF YES DO NOT TYPE STATUS
                        21$:    COMB    INIFLG          ;SET FLAG
                        24$:    TYPE    ,XHEAD          ;TYPE HEADER
                                MOV     #KM.MAP,R4      ;SET POINTER
                        5$:     MOV     R4,$TMP0        ;SET ADDRESS
                                MOV     (R4)+,$TMP1     ;SET CSR
                                BEQ     1$              ;ALL DONE IF ZERO
                                MOV     (R4)+,$TMP2     ;SET STAT1
                                MOV     (R4)+,$TMP3     ;SET STAT2
                                MOV     (R4)+,$TMP4     ;SET STAT3
                                CONVRT                  ;TYPE OUT STATUS MAP
                                XSTATQ                  ;
                                BR      5$
                        1$:     MOV     #KM.MAP,R0      ;R0 POINTS TO STATUS TABLE

                        ;;*************************************************************************
                        ;;*AUTO SIZE TEST
                        ;;*THIS TEST VERIFYS THAT THE KMC11S AND/OR KMC11S ARE AT THE CORRECT FLOATING
                        ;;*ADDRESSES FOR YOUR SYSTEM. IF THIS TEST FAILS, IT IS NOT A HARDWARE ERROR.
                        ;;*CHECK THE ADDRESSES OF ALL FLOATING DEVICES (DJ,DH,DQ,DU,DUP,LK,DMC,DZ,KMC).
                        ;;*IF THERE ARE NO OTHER FLOATING DEVICES BEFORE THE KMC11, THE FIRST
                        ;;* KMC11 IS 760110. NO DEVICE SHOULD EVER BE AT
                        ;;*ADDRESS 760000.
                        ;;*************************************************************************

                                MOV     @#4,-(SP)       ;SAVE LOC 4
                                MOV     @#6,-(SP)       ;SAVE LOC 6
                                CLR     @#6             ;CLEAR VEC+2
                                CLR     $TMP2           ;CLEAR FLAG
                        AUSTRT: MOV     (R0),KMCSR      ;GET NEXT KMC CSR
                                BEQ     AUDONE          ;BR IF DONE
                        2$:     MOV     #NODEV,@#4      ;SET UP FOR TIMEOUT
                        3$:     MOV     #10,R3          ;R3 IS COUNT OF DEVICES BEFORE KMC
```

```
 804  003146  012702  003342          4$:     MOV     #DEVTAB,R2      ;R2 IS DEVICE TABLE PONTER
 805  003152  012701  160010                  MOV     #160010,R1      ;START WITH ADDRESS 160010
 806  003156  005711                  FLOAT:  TST     (R1)            ;CHECK ADDRESS IN R1
 807  003160  111204                          MOVB    (R2),R4         ;IF NO TIMEOUT, GET NEXT ADDRESS
 808  003162  060401                          ADD     R4,R1           ;IN R1
 809  003164  005201                          INC     R1              ;
 810  003166  040401                          BIC     R4,R1           ;
 811  003170  005703                          TST     R3              ;ANY MORE DEVICES TO CHECK FOR?
 812  003172  001371                          BNE     FLOAT           ;BR IF YES
 813  003174  012737  003244  000004          MOV     #ERR,@#4        ;OK ONLY KMC'S ARE LEFT, SET UP FOR TIMEOUT
 814  003202  005711                  FY:     TST     (R1)            ;CHECK KMC ADDRESS
 815  003204  020137  002066                  CMP     R1,KMCSR        ;DOES IT MATCH
 816  003210  001403                          BEQ     OK              ;BR IF YES
 817  003212  062701  000010                  ADD     #10,R1          ;GET NEXT KMC ADDRESS
 818  003216  000771                          BR      FY              ;DO IT AGAIN
 819  003220  062700  00001C          Ok:     ADD     #10,R0          ;SKIP TO NEXT KMC CSR
 820  003224  062701  000010                  ADD     #10,R1          ; GET NEXT KMC ADDRESS
 821  003230  011037  002066                  MOV     (R0),KMCSR      ; GET NEXT KMC CSR
 822  003234  001447                          BEQ     AUDONE          ; BRANCH IF ALL DONE.
 823  003236  000761                          BR      FY              ; DO IT AGAIN.
 824  003240  122243                  NODEV:  CMPB    (R2)+,-(R3)     ;ON TIMEOUT, INC R2, DEC R3
 825  003242  000002                          RTI                     ;$LPADR
 826  003244  005737  001302          ERR:    TST     $TMP2           ;CHECK FLAG IF = 0 TYPE HEADER
 827  003250  001014                          BNE     1$              ;SKIP HEADER
 828  003252  104401                          TYPE                    ;TYPEOUT HEADER MESSAGE
 829  003254  010762                          CONERR                  ;CONFIGURATION ERROR!!!!
 830  003256  012737  003244  001460          MOV     #ERR,SAVPC      ;SAVE PC FOR TYPEOUT
 831  003264  104417                          CNVRT                   ;TYPE OUT ERROR PC
 832  003266  003322                          ERRPC                   ;
 833  003270  104401                          TYPE                    ;TYPE REST OF HEADER
 834  003272  011027                          CNERR                   ;
 835  003274  012737  177777  001302          MOV     #-1,$TMP2       ;SET FLAG SO IT ONLY GETS TYPED ONCE
 836  003302  010137  001264          1$:     MOV     R1,$REG1        ;SAVE R1 FOR TYPEOUT
 837  003306  104416                          CONVRT
 838  003310  003330                          CONTAB                  ;TYPE CSR VALUES
 839  003312  104401                  3$:     TYPE
 840  003314  011050                          KMCM
 841  003316  022626                  4$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
 842  003320  000737                          BR      OK              ;BR TO GET OUT
 843  003322  000001                  ERRPC:  1
 844  003324     006     002                  .BYTE   6,2
 845  003326  001460                          SAVPC
 846  003330  000002                  CONTAB: 2
 847  003332     006     004                  .BYTE   6,4
 848  003334  001264                          $REG1
 849  003336     006     002                  .BYTE   6,2
 850  003340  002066                          KMCSR
 851  003342     007                  DEVTAB: .BYTE   7               ;DJ
 852  003343     017                          .BYTE   17              ;DH
 853  003344     007                          .BYTE   7               ;DQ
 854  003345     007                          .BYTE   7               ;DU
 855  003346     007                          .BYTE   7               ;DUP
 856  003347     007                          .BYTE   7               ;LK
 857  003350     007                          .BYTE   7               ;DMC
 858  003351     007                          .BYTE   7               ;DZ
 859  003352     007                          .BYTE   7               ;KMC
```

```
                003354                          .EVEN
003354                          AUDONE:
003354  012637  000006          1$:     MOV     (SP)+,@#6       ;RESTORE LOC 6
003360  012637  000004                  MOV     (SP)+,@#4       ;RESTORE LOC 4
003364  032737  000010  001446          BIT     #SW03,STRTSW    ;SELECT SPECIFIC DEVICES??
003372  001422                          BEQ     3$              ;BR IF NO.
003374  104401  010017                  TYPE    ,MNEW           ;TYPE THE MESSAGE.
003400  005000                          CLR     R0              ;ZERO DATA LIGHTS
003402  000000                          HALT                    ;WAIT FOR USER TO TELL WHAT DEVICES TO RUN
003404  027737  175630  001474          CMP     @SWR,SAVACT     ;IS THE NUMBER VALID?
003412  101404                          BLOS    2$              ;BR IF NUMBER IS OK.
003414  104401  007672                  TYPE    ,MERR3          ;TELL USER OF INVALID NUMBER.
003420  000000                          HALT                    ;STOP EVERY THING.
003422  000776                          BR      .-2             ;& START THE PROGRAM AGAIN.
003424  017737  175610  001470  2$:     MOV     @SWR,KMACTV     ;GET NEW DEVICE PATTERN
003432  013700  001470                  MOV     KMACTV,R0       ;SHOW THE USER WHAT HE SELECTED.
003436  000000                          HALT                    ;CONTINUE DYNAMIC SWITCHES.
003440  012700  000300          3$:     MOV     #300,R0         ;PREPARE TO CLEAR THE FLOATING
003444  012701  000302                  MOV     #302,R1         ;VECTOR AREA. 300-776
003450  010120                  4$:     MOV     R1,(R0)+        ;START PUTTING "PC+2 - HALT"
003452  005021                          CLR     (R1)+           ;IN VECTOR AREA.
003454  022021                          CMP     (R0)+,(R1)+     ;POP POINTERS
003456  022700  0010C                   CMP     #1000,R0        ;ALL DONE??
003462  001372                          BNE     4$              ;BR IF NO.

                                ;TEST START AND RESTART
                                ;----------------------

003464  012706  001200          .BEGIN: MOV     #STACK,SP       ;SET UP STACK
003470  013746  000006                  MOV     @#6,-(SP)       ;SAVE LOC 6
003474  013746  000004                  MOV     @#4,-(SP)       ;SAVE LOC 4
003500  005000                          CLR     R0              ;START AT 0
003502  012737  003546  000004          MOV     #2$,@#4         ;SET UP FOR TIME OUT
003510  005037  000006                  CLR     @#6             ;TO AUTOSIZE MEMORY
003514  005720                  6$:     TST     (R0)+           ;CHECK ADDRESS IN R0
003516  022700  157776                  CMP     #157776,R0      ;IS IT AT LEAST 28K
003522  001374                          BNE     6$              ;BR IF NO
003524  162700  007776                  SUB     #7776,R0        ;SAVE 2K FOR MONITORS
003530  010037  001466          7$:     MOV     R0,MEMLIM       ;STORE MEMORY LIMIT
003534  012637  000004                  MOV     (SP)+,@#4       ;RESTORE LOC 4
003540  012637  000006                  MOV     (SP)+,@#6       ;RESTORE LOC 6
003544  000413                          BR      10$             ;CONTINUE
003546  022626                  2$:     CMP     (SP)+,(SP)+     ;ADJUST STACK
003550  162700  000004                  SUB     #4,R0           ;GET LAST GOOD ADDRESS
003554  162700  007776                  SUB     #7776,R0        ;SAVE 2K FOR MONITORS
003560  022700  030000                  CMP     #30000,R0       ;IS IT 8K?
003564  001361                          BNE     7$              ;BR IF NO
003566  012700  037400                  MOV     #37400,R0       ;IF 8K DON'T SAVE 2K
003572  000756                          BR      7$              ;
003574  012737  000340  177776  10$:    MOV     #340,PS         ;LOCK OUT INTERRUPTS
003602  032737  000004  001446          BIT     #BIT2,STRTSW    ;CHECK FOR LOCK ON TEST
003610  001406                          BEQ     1$              ;BR IF NO LOCK DESIRED.
003612  104401  007716                  TYPE    ,MLOCK          ;TYPE LOCK SELECTED.
003616  012737  000240  004146          MOV     #NOP,TTST       ;SET UP TO LOCK
003624  000403                          BR      3$              ;CONTINUE ALONG.
003626  013737  004360  004146  1$:     MOV     BRW,TTST        ;PREPARE NORMAL SCOPE ROUTINE
```

```
   916  003634  012737  011460  001206  3$:     MOV     #CYCLE,$LPADR   ;START AT "CYCLE" FIND WHICH DEVICE TO TEST
   917  003642  032737  000002  001476  4$:     BIT     #SW01,STRTSW    ;IS TEST NO. SELECTED?
   918  003650  001002                          BNE     5$              ;BR IF YES
   919  003652  104401  007642                  TYPE    ,MR             ;TYPE R
   920  003656  000177  175324          5$:     JMP     @$LPADR         ;START TESTING
```

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

                                .SBTTL  END OF PASS ROUTINE

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

003662                          $EOP:
003662  000005                          RESET
003664  005237  001324                  INC     $PASS           ; INCREMENT THE PASS COUNT
003670  105037  001203                  CLRB    $ERFLG          ; CLEAR ERROR FLAG
003674  104401  007620                  TYPE    ,MEPASS         ; TYPE END PASS.
003700  104401  007745                  TYPE    ,MCSRX          ; TYPE "CSR"
003704  104417  004104                  CNVRT   ,XCSR           ; SHOW IT.
003710  104401  007753                  TYPE    ,MVECX          ; TYPE VECTOR.
003714  104417  004112                  CNVRT   ,XVEC           ; SHOW IT.
003720  104401  007761                  TYPE    ,MPASSX         ; TYPE " PASSES "
003724  104417  004120                  CNVRT   ,XPASS          ; SHOW IT.
003730  104401  007772                  TYPE    ,MERRX          ; TYPE " ERRORS "
003734  104417  004126                  CNVRT   ,XERR           ; SHOW IT.
003740  013700  001504                  MOV     #ILK,R0         ; SET POINTER TO PASSCNT.
003744  013720  001324                  MOV     $PASS,(R0)+     ; SAVE THE PASS COUNT.
003750  013720  001212                  MOV     $ERTTL,(R0)+    ; SAVE ERROR COUNT
003754  013777  002060  176074          MOV     KMRLVL,@KMRVEC  ; RESTORE THE RECEIVER INTERRUPT VECTOR.
003762  005077  176072                  CLR     @KMRLVL         ; RESTORE RECEIVER LEVEL
003766  013777  002064  176066          MOV     KMTLVL,@KMTVEC  ; RESTORE THE TRANSMIT INTERRUPT VECTOR.
003774  005077  176064                  CLR     @KMTLVL         ; RESTORE TRANSMITTER LEVEL
004000  005337  001476                  DEC     SAVNUM          ; ALL DEVICE TESTED?
004004  001035                          BNE     $DOAGN          ; BRANCH IF NO.
004006  112737  000377  001511          MOVB    #377,QV.FLG     ; SET QUICK VERIFY FLAG.
004014  013737  001472  001476          MOV     KMNUM,SAVNUM    ; RESTORE DEVICE COUNT.
004022  005037  001215                  CLR     $ERRPC          ; CLEAR LAST ERROR PC
004026  005037  001310                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
004032  005237  001324                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
004036  042737  100000  001324          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
004044  005327                          DEC     (PC)+           ;;LOOP?
004046  000001                  $EOPCT: .WORD   1
004050  003013                          BGT     $DOAGN          ;;YES
004052  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
004054  000001                  $ENDCT: .WORD   1
004056  004046                          $EOPCT
004060  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
004064  001405                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
004066  000005                          RESET                   ;;CLEAR THE WORLD
004070  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
004072  000240                          NOP                     ;;SAVE ROOM
004074  000240                          NOP                     ;;FOR
004076  000240                          NOP                     ;;ACT11
004100                          $DOAGN:
004100  000137                          JMP     @(PC)+          ;;RETURN
```

```
                        END OF PASS ROUTINE

 004102  011460         $RTNAD: .WORD   CYCLE
 004104  000001         XCSR:   1
 004106     006  002             .BYTE   6,2
 004110  002066                 KMCSR
 004112  000001         XVEC:   1
 004114     004  002             .BYTE   4,2
 004116  002056                 KMRVEC
 004120  000001         XPASS:  1
 004122     006  002             .BYTE   6,2
 004124  001324                 $PASS
 004126  000001         XERR:   1
 004130     006  002             .BYTE   6,2
 004132  001212                 $ERTTL

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

                        .SBTTL  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

 004134         $SCOPE:
 004134  005037  001216         CLR     $ERRPC                  ; CLEAR LAST ERROR PC
 004140  023716  013734         CMP     TST1+2,(SP)             ; IS THIS TEST #1 ?
 004144  001413                 BEQ     $XTSTR                  ; IF SO DON'T LOOP.
 004146  000406         TTST:   BR      1$                      ;
 004150  105777  175070         TSTB    @$TKS                   ; KEYBOARD DONE ?
 004154  100067                 BPL     $OVER                   ; IF NO DONT WAIT.
 004156  017766  175064  177776   MOV     @$TKB,-2(SP)            ;
 004164  032777  040000  175046 1$:      BIT     #BIT14,@$SWR    ;;LOOP ON PRESENT TEST?
 004172  001060                 BNE     $OVER                   ;;YES IF SW14=1
                        ;#####START OF CODE FOR THE XOR TESTER#####
 004174  000416         $XTSTR: BR      6$                      ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
 004176  013746  000004         MOV     @#ERRVEC,-(SP)          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
 004202  012737  004222  000004 MOV     #5$,@#ERRVEC            ;;SET FOR TIMEOUT
 004210  005737  177060         TST     @#177060                ;;TIME OUT ON XOR?
 004214  012637  000004         MOV     (SP)+,@#ERRVEC          ;;RESTORE THE ERROR VECTOR
 004220  000436                 BR      $SVLAD                  ;;GO TO THE NEXT TEST
 004222  022626         5$:     CMP     (SP)+,(SP)+             ;;CLEAR THE STACK AFTER A TIME OUT
 004224  012637  000004         MOV     (SP)+,@#ERRVEC          ;;RESTORE THE ERROR VECTOR
 004230  000441                 BR      $OVER                   ;;LOOP ON THE PRESENT TEST
 004232         6$:;#####END OF CODE FOR THE XOR TESTER#####
 004232  105737  001203         TSTB    $ERFLG                  ;;HAS AN ERROR OCCURRED?
 004236  001404                 BEQ     3$                      ;;BR IF NO
 004240  105037  001203         CLRB    $ERFLG                  ;;ZERO THE ERROR FLAG
 004244  005037  001310         CLR     $TIMES                  ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
 004250  032777  004000  174762 3$:     BIT     #BIT11,@$SWR    ;;INHIBIT ITERATIONS?
```

```
1033  004256  001011                        BNE     1$              ;;BR IF YES
1034  004260  005737  001324                TST     $PASS           ;;IF FIRST PASS OF PROGRAM
1035  004264  001406                        BEQ     1$              ;;       INHIBIT ITERATIONS
1036  004266  005237  001204                INC     $ICNT           ;;INCREMENT ITERATION COUNT
1037  004272  023737  001310  001204        CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
1038  004300  002015                        BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
1039  004302  012737  000001  001204  1$:    MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
1040  004310  013737  004362  001310        MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
1041  004316  105237  001202          $SVLAD: INCB    $TSTNM          ;;COUNT TEST NUMBERS
1042  004322  113737  001202  001322        MOVB    $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
1043  004330  011637  001206                MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
1044  004334  013777  001202  174700  $OVER: MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
1045  004342  013716  001206                MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
1046  004346  005037  001444                CLR     LOCK                    ; RESET LOCK ON DATA.
1047  004352  013701  002066                MOV     KMCSR,R1                ; R1 CONTAINS BASE KMC ADDRESS.
1048  004356  000002                        RTI
1049  004360  000406                BRW:    .WORD   406
1050  004362  000020                $MXCNT: 20                      ;;MAX. NUMBER OF ITERATIONS
1051
1052                                        ;CHECK FOR FREEZE ON CURRENT DATA
1053                                        ;--------------------------------
1054
1055  004364  004737  011212        .SCOP1: JSR     PC,CKSWR                ;CHECK FOR SOFT SWR
1056  004370  032777  001000  174642        BIT     #SW09,@SWR      ;IS SW09=1(SET)?
1057  004376  001405                        BEQ     1$              ;BR IF NOT SET.
1058  004400  005737  001444                TST     LOCK
1059  004404  001402                        BEQ     1$
1060  004406  013716  001444                MOV     LOCK,(SP)       ;GOTO THE ADDRESS IN LOCK.
1061  004412  000002                1$:     RTI                     ;GO BACK.
1062
1063                                        ;TELETYPE OUTPUT ROUTINE
1064                                        ;-----------------------
1065
1066                                .SBTTL  TYPE ROUTINE
1067
1068                                ;;******************************************************************
1069                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
1070                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
1071                                ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
1072                                ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
1073                                ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
1074                                ;*
1075                                ;*CALL:
1076                                ;*1) USING A TRAP INSTRUCTION
1077                                ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
1078                                ;*OR
1079                                ;*      TYPE
1080                                ;*      MESADR
1081                                ;*
1082
1083  004414  105737  001257        $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
1084  004420  100002                        BPL     1$              ;;BR IF YES
1085  004422  000000                        HALT                    ;;HALT HERE IF NO TERMINAL
1086  004424  000430                        BR      3$              ;;LEAVE
1087  004426  010046                1$:     MOV     R0,-(SP)        ;;SAVE R0
1088  004430  017600  000002                MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
```

```
1089  004434  122737  000001  001336          CMPB    #APTENV,$ENV      ;;RUNNING IN APT MODE
1090  004442  001011                          BNE     62$               ;;NO,GO CHECK FOR APT CONSOLE
1091  004444  132737  000100  001337          BITB    #APTSPOOL,$ENVM   ;;SPOOL MESSAGE TO APT
1092  004452  001405                          BEQ     62$               ;;NO,GO CHECK FOR CONSOLE
1093  004454  010037  004464                  MOV     R0,61$            ;;SETUP MESSAGE ADDRESS FOR APT
1094  004460  004737  004704                  JSR     PC,$ATY3          ; SPOOL MESSAGE TO APT
1095  004464  000000                  61$:    .WORD   0                 ;;MESSAGE ADDRESS
1096  004466  132737  000040  001337  62$:    BITB    #APTCSUP,$ENVM    ;;APT CONSOLE SUPPRESSED
1097  004474  001003                          BNE     60$               ;;YES,SKIP TYPE OUT
1098  004476  112046                  2$:     MOVB    (R0)+,-(SP)       ;;PUSH CHARACTER TO BE TYPED ONTO STACK
1099  004500  001005                          BNE     4$                ;;BR IF IT ISN'T THE TERMINATOR
1100  004502  005726                          TST     (SP)+             ;;IF TERMINATOR POP IT OFF THE STACK
1101  004504  012600                  60$:    MOV     (SP)+,R0          ;;RESTORE R0
1102  004506  062716  000002          3$:     ADD     #2,(SP)           ;;ADJUST RETURN PC
1103  004512  000002                          RTI                       ;;RETURN
1104  004514  122716  000011          4$:     CMPB    #HT,(SP)          ;;BRANCH IF <HT>
1105  004520  001430                          BEQ     8$
1106  004522  122716  000200                  CMPB    #CRLF,(SP)        ;;BRANCH IF NOT <CRLF>
1107  004526  001006                          BNE     5$
1108  004530  005726                          TST     (SP)+             ;;POP  <CR><LF> EQUIV
1109  004532  104401                          TYPE                      ;;TYPE A CR AND LF
1110  004534  001313                          $CRLF
1111  004536  105037  004672                  CLRB    $CHARCNT          ;;CLEAR CHARACTER COUNT
1112  004542  000755                          BR      2$                ;;GET NEXT CHARACTER
1113  004544  004737  004626          5$:     JSR     PC,$TYPEC         ;;GO TYPE THIS CHARACTER
1114  004550  123726  001256          6$:     CMPB    $FILLC,(SP)+      ;;IS IT TIME FOR FILLER CHARS.?
1115  004554  001350                          BNE     2$                ;;IF NO GO GET NEXT CHAR.
1116  004556  013746  001254                  MOV     $NULL,-(SP)       ;;GET # OF FILLER CHARS. NEEDED
1117                                                                    ;;AND THE NULL CHAR.
1118  004562  105366  000001          7$:     DECB    1(SP)             ;;DOES A NULL NEED TO BE TYPED?
1119  004566  002770                          BLT     6$                ;;BR IF NO--GO POP THE NULL OFF OF STACK
1120  004570  004737  004626                  JSR     PC,$TYPEC         ;;GO TYPE A NULL
1121  004574  105337  004672                  DECB    $CHARCNT          ;;DO NOT COUNT AS A COUNT
1122  004600  000770                          BR      7$                ;;LOOP
1123
1124                                  ;HORIZONTAL TAB PROCESSOR
1125
1126  004602  112716  000040          8$:     MOVB    #' ,(SP)          ;;REPLACE TAB WITH SPACE
1127  004606  004737  004626          9$:     JSR     PC,$TYPEC         ;;TYPE A SPACE
1128  004612  132737  000007  004672          BITB    #7,$CHARCNT       ;;BRANCH IF NOT AT
1129  004620  001372                          BNE     9$                ;;TAB STOP
1130  004622  005726                          TST     (SP)+             ;;POP SPACE OFF STACK
1131  004624  000724                          BR      2$                ;;GET NEXT CHARACTER
1132  004626  105777  174416          $TYPEC: TSTB    @$TPS             ;;WAIT UNTIL PRINTER IS READY
1133  004632  100375                          BPL     $TYPEC
1134  004634  116677  000002  174410          MOVB    2(SP),@$TPB       ;;LOAD CHAR TO BE TYPED INTO DATA REG.
1135  004642  122766  000015  000002          CMPB    #CR,2(SP)         ;;IS CHARACTER A CARRIAGE RETURN?
1136  004650  001003                          BNE     1$                ;;BRANCH IF NO
1137  004652  105037  004672                  CLRB    $CHARCNT          ;;YES--CLEAR CHARACTER COUNT
1138  004656  000406                          BR      $TYPEX            ;;EXIT
1139  004660  122766  000012  000002  1$:     CMPB    #LF,2(SP)         ;;IS CHARACTER A LINE FEED?
1140  004666  001402                          BEQ     $TYPEX            ;;BRANCH IF YES
1141  004670  105227                          INCB    (PC)+             ;;COUNT THE CHARACTER
1142  004672  000000          $CHARCNT:.WORD  0                         ;;CHARACTER COUNT STORAGE
1143  004674  000207          $TYPEX: RTS     PC
1144
```

```
                                .SBTTL  APT COMMUNICATIONS ROUTINE
                                ;;*****************************************************************
004676  112737  000001  005142  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
004704  112737  000001  005140  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
004712  000403                  BR      $ATYC
004714  112737  000001  005142  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
004722                          $ATYC:
004722  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
004724  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
004726  105737  005140          TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
004732  001450                  BEQ     5$              ;;IF NOT:  BR
004734  122737  000001  001336  CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
004742  001031                  BNE     3$              ;;IF NOT:  BR
004744  132737  000100  001337  BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
004752  001425                  BEQ     3$              ;;IF NOT:  BR
004754  017600  000004          MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
004760  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
004766  005737  001316          1$:     TST     $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
004772  001375                  BNE     1$              ;;IF NOT:  WAIT
004774  010037  001332          MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
005000  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
005002  001376                  BNE     2$
005004  163700  001332          SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
005010  006200                  ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
005012  010037  001334          MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
005016  012737  000004  001316  MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
005024  000413                  BR      5$
005026  017637  000004  005052  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
005034  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
005042  013746  177776          MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
005046  004737  004414          JSR     PC,$TYPE        ;;CALL TYPE MACRO
005052  000000                  4$:     .WORD   0
005054                          5$:
005054  105737  005142          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
005060  001416                  BEQ     12$             ;;IF NOT:  BR
005062  005737  001336          TST     $ENV            ;;RUNNING UNDER APT?
005066  001413                  BEQ     12$             ;;IF NOT:  BR
005070  005737  001316          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
005074  001375                  BNE     11$             ;;IF NOT:  WAIT
005076  017637  000004  001320  MOV     @4(SP),$FATAL   ;;GET ERROR #
005104  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
005112  005237  001316          INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
005116  105037  005142          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
005122  105037  005141          CLRB    $LFLG           ;;CLEAR LOG FLAG
005126  105037  005140          CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
005132  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
005134  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
005136  000207                  RTS     PC              ;;RETURN
005140     000          $MFLG:  .BYTE   0               ;;MESSG. FLAG
005141     000          $LFLG:  .BYTE   0               ;;LOG FLAG
005142     000          $FFLG:  .BYTE   0               ;;FATAL FLAG
005144                          .EVEN
000200                  APTSIZE=200
000001                  APTENV=001
000100                  APTSPOOL=100
```

```
                        000040                  APTCSUP=040
                                                        ;-------------------------
                                                .SBTTL  TTY INPUT ROUTINE
                                                ;;******************************************************************
                                                .ENABL  LSB

                                                .DSABL  LSB


                                                ;;******************************************************************
                                                ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
                                                ;*CALL:
                                                ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
                                                ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
                                                ;*                              ;;WITH PARITY BIT STRIPPED OFF
                                                ;

005144  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
005146  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
005154  105777  174064          1$:     TSTB    @$TKS           ;;WAIT FOR
005160  100375                          BPL     1$              ;;A CHARACTER
005162  117766  174060  000004          MOVB    @$TKB,4(SP)     ;;READ THE TTY
005170  042766  177600  000004          BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
005176  026627  000004  000023          CMP     4(SP),#23       ;;IS IT A CONTROL-S?
005204  001013                          BNE     3$              ;;BRANCH IF NO
005206  105777  174032          2$:     TSTB    @$TKS           ;;WAIT FOR A CHARACTER
005212  100375                          BPL     2$              ;;LOOP UNTIL ITS THERE
005214  117746  174026                  MOVB    @$TKB,-(SP)     ;;GET CHARACTER
005220  042716  177600                  BIC     #^C177,(SP)     ;;MAKE IT 7-BIT ASCII
005224  022627  000021                  CMP     (SP)+,#21       ;;IS IT A CONTROL-Q?
005230  001366                          BNE     2$              ;;IF NOT DISCARD IT
005232  000750                          BR      1$              ;;YES, RESUME
005234  026627  000004  000140  3$:     CMP     4(SP),#140      ;;IS IT UPPER CASE?
005242  002407                          BLT     4$              ;;BRANCH IF YES
005244  026627  000004  000175          CMP     4(SP),#175      ;;IS IT A SPECIAL CHAR?
005252  003003                          BGT     4$              ;;BRANCH IF YES
005254  042766  000040  000004          BIC     #40,4(SP)       ;;MAKE IT UPPER CASE
005262  000002                  4$:     RTI                     ;;GO BACK TO USER
                                                ;;******************************************************************
                                                ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
                                                ;*CALL:
                                                ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
                                                ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
                                                ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S

005264  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
005266  005046                          CLR     -(SP)           ;;CLEAR THE RUBOUT KEY
005270  012703  005520          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
005274  022703  005527          2$:     CMP     #$TTYIN+7,R3    ;;BUFFER FULL?
005300  101456                          BLOS    4$              ;;BR IF YES
005302  104402                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
005304  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
005306  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
005312  001022                          BNE     5$              ;;BR IF NO
```

```
1257  005314  005716                          TST     (SP)            ;;IS THIS THE FIRST RUBOUT?
1258  005316  001007                          BNE     6$              ;;BR IF NO
1259  005320  112737  000134  005516          MOVB    #'\,@$          ;;TYPE A BACK SLASH
1260  005326  104401  005516                  TYPE    ,@$
1261  005332  012716  177777                  MOV     #-1,(SP)        ;;SET THE RUBOUT KEY
1262  005336  005303                  6$:     DEC     R3              ;;BACKUP BY ONE
1263  005340  020327  005520                  CMP     R3,#$TTYIN      ;;STACK EMPTY?
1264  005344  103404                          BLO     4$              ;;BR IF YES
1265  005346  111337  005516                  MOVB    (R3),@$         ;;SETUP TO TYPEOUT THE DELETED CHAR.
1266  005352  104401  005516                  TYPE    ,@$             ;;GO TYPE
1267  005356  000746                          BR      2$              ;;GO READ ANOTHER CHAR.
1268  005360  005716                  5$:     TST     (SP)            ;;RUBOUT KEY SET?
1269  005362  001406                          BEQ     7$              ;;BR IF NO
1270  005364  112737  000134  005516          MOVB    #'\,@$          ;;TYPE A BACK SLASH
1271  005372  104401  005516                  TYPE    ,@$
1272  005376  005016                          CLR     @(SP)           ;;CLEAR THE RUBOUT KEY
1273  005400  122713  000025          7$:     CMPB    #25,(R3)        ;;IS CHARACTER A CTRL U?
1274  005404  001003                          BNE     8$              ;;BR IF NO
1275  005406  104401  005527                  TYPE    ,$CNTLU         ;;TYPE A CONTROL "U"
1276  005412  000726                          BR      1$              ;;GO START OVER
1277  005414  122713  000022          8$:     CMPB    #22,(R3)        ;;IS CHARACTER A "^R"?
1278  005420  001011                          BNE     3$              ;;BRANCH IF NO
1279  005422  105013                          CLRB    (R3)            ;;CLEAR THE CHARACTER
1280  005424  104401  001313                  TYPE    ,$CRLF          ;;TYPE A "CR" & "LF"
1281  005430  104401  005520                  TYPE    ,$TTYIN         ;;TYPE THE INPUT STRING
1282  005434  000717                          BR      2$              ;;GO PICKUP ANOTHER CHACTER
1283  005436  104401  001312          4$:     TYPE    ,$QUES          ;;TYPE A '?'
1284  005442  000712                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
1285  005444  111337  005516          3$:     MOVB    (R3),@$         ;;ECHO THE CHARACTER
1286  005450  104401  005516                  TYPE    ,@$
1287  005454  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
1288  005460  001305                          BNE     2$              ;;LOOP IF NOT RETURN
1289  005462  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
1290  005466  104401  001314                  TYPE    ,$LF            ;;TYPE A LINE FEED
1291  005472  005726                          TST     (SP)+           ;;CLEAN RUBOUT KEY FROM THE STACK
1292  005474  012603                          MOV     (SP)+,R3        ;;RESTORE R3
1293  005476  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
1294  005500  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
1295  005506  012766  005520  000004          MOV     #$TTYIN,4(SP)
1296  005514  000002                          RTI                     ;;RETURN
1297  005516     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
1298  005517     000                          .BYTE   0               ;;TERMINATOR
1299  005520  000007                  $TTYIN: .BLKB   7               ;;RESERVE 7 BYTES FOR TTY INPUT
1300  005527     136  006525  000012  $CNTLU: .ASCIZ  /^U/<15><12>    ;;CONTROL "U"
1301  005534  043536  005015     000  $CNTLG: .ASCIZ  /^G/<15><12>    ;;CONTROL "G"
1302  005541     015  051412  051127  $MSWR:  .ASCIZ  <15><12>/SWR = /
1303  005546  036440  000040
1304  005552  020040  042516  020127  $MNEW:  .ASCIZ  /  NEW = /
1305  005560  020075     000
1306          005564                  .EVEN
1307                                  .SBTTL  READ AN OCTAL NUMBER FROM THE TTY
1308
1309                                  ;;*****************************************************************
1310                                  ;*THIS ROUTINE WILL READ AN OCTAL (ASCII) NUMBER FROM THE TTY AND
1311                                  ;*CHANGE IT TO BINARY.
1312                                  ;*THE INPUT CHARACTERS WILL BE CHECKED TO INSURED THEY ARE LEGAL
```

```
1313                                    ;*OCTAL DIGITS. IF AN ILLEGAL CHARACTER IS READ A "?" WILL BE TYPED
1314                                    ;*FOLLOWED BY A CARRIAGE RETURN-LINE FEED. THE COMPLETE NUMBER MUST
1315                                    ;*THEN BE RETYPED. THE INPUT IS TERMINATED BY TYPING A CARRIAGE RETURN.
1316                                    ;*CALL:
1317                                    ;*      RDOCT                   ;;READ AN OCTAL NUMBER
1318                                    ;*      RETURN HERE             ;;LOW ORDER BITS ARE ON TOP OF THE STACK
1319                                    ;*                              ;;HIGH ORDER BITS ARE IN $HIOCT
1320
1321  005564  011646                    $RDOCT: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR THE
1322  005566  016666  000004  000002            MOV     4(SP),2(SP)     ;;INPUT NUMBER
1323  005574  010046                            MOV     R0,-(SP)        ;;PUSH R0 ON STACK
1324  005576  010146                            MOV     R1,-(SP)        ;;PUSH R1 ON STACK
1325  005600  010246                            MOV     R2,-(SP)        ;;PUSH R2 ON STACK
1326  005602  104403                    1$:     RDLIN                   ;;READ AN ASCIZ LINE
1327  005604  012600                            MOV     (SP)+,R0        ;;GET ADDRESS OF 1ST CHARACTER
1328  005606  010037  005712                    MOV     R0,5$           ;;AND SAVE IT
1329  005612  005001                            CLR     R1              ;;CLEAR DATA WORD
1330  005614  005002                            CLR     R2
1331  005616  112046                    2$:     MOVB    (R0)+,-(SP)     ;;PICKUP THIS CHARACTER
1332  005620  001420                            BEQ     3$              ;;IF ZERO GET OUT
1333  005622  122716  000060                    CMPB    #'0,(SP)        ;;MAKE SURE THIS CHARACTER
1334  005626  003026                            BGT     4$              ;;IS AN OCTAL DIGIT
1335  005630  122716  000067                    CMPB    #'7,(SP)
1336  005634  002423                            BLT     4$
1337  005636  006301                            ASL     R1              ;;*2
1338  005640  006102                            ROL     R2
1339  005642  006301                            ASL     R1              ;;*4
1340  005644  006102                            ROL     R2
1341  005646  006301                            ASL     R1              ;;*8
1342  005650  006102                            ROL     R2
1343  005652  042716  177770                    BIC     #^C7,(SP)       ;;STRIP THE ASCII JUNK
1344  005656  062601                            ADD     (SP)+,R1        ;;ADD IN THIS DIGIT
1345  005660  000756                            BR      2$              ;;LOOP
1346  005662  005726                    3$:     TST     (SP)+           ;;CLEAN TERMINATOR FROM STACK
1347  005664  010166  000012                    MOV     R1,12(SP)       ;;SAVE THE RESULT
1348  005670  010237  005722                    MOV     R2,$HIOCT
1349  005674  012602                            MOV     (SP)+,R2        ;;POP STACK INTO R2
1350  005676  012601                            MOV     (SP)+,R1        ;;POP STACK INTO R1
1351  005700  012600                            MOV     (SP)+,R0        ;;POP STACK INTO R0
1352  005702  000002                            RTI                     ;;RETURN
1353  005704  005726                    4$:     TST     (SP)+           ;;CLEAN PARTIAL FROM STACK
1354  005706  105010                            CLRB    (R0)            ;;SET A TERMINATOR
1355  005710  104401                            TYPE                    ;;TYPE UP THRU THE BAD CHAR.
1356  005712  000000                    5$:     .WORD   0
1357  005714  104401  001312                    TYPE    ,$QUES          ;;"?" "CR" & "LF"
1358  005720  000730                            BR      1$              ;;TRY AGAIN
1359  005722  000000                    $HIOCT: .WORD   0               ;;HIGH ORDER BITS GO HERE
1360                                    ;
1361                                    ;    INPUT OCTAL NUMBER ROUTINE
1362                                    ;-----------------------------------
1363                                    ;
1364  005724  010546                    $INPUT: MOV     R5,-(SP)                ; SAVE REGISTER R5.
1365  005726  016605  000002                    MOV     2(SP),R5                ; GET FIRST PARAMETER ADDRESS.
1366  005732  012537  005770                    MOV     (R5)+,WHAT              ; GET MESSAGE ADDRESS.
1367  005736  012537  006050                    MOV     (R5)+,LOLIM             ; GET LOW LIMIT FOR THE  #
1368  005742  012537  006052                    MOV     (R5)+,HILIM             ; GET HIGH LIMIT FOR THE #.
```

```
1369  005746  012537  006054            MOV     (R5)+,WHERE     ; GET ADDRESS OF INBUFFEP
1370  005752  112537  006056            MOVB    (R5)+,LOBITS    ; GET LOWMASK BITS.
1371  005756  112537  006057            MOVB    (R5)+,ADRCNT    ; GET # OF #'S TO BE GENERATED.
1372  005762  010566  000002            MOV     R5,2(SP)        ; SAVE THE RETURN ADDRESS.
1373  005766  104401              INLP1: TYPE                    ; TYPE THE MESSAGE.
1374  005770  000000              WHAT:  .WORD   0
1375  005772  104404                     RDOCT                  ; READ OCTAL # FROM KEYBOARD.
1376  005774  021637  006052            CMP     (SP),HILIM      ; IS IT IN HIGH LIMIT?
1377  006000  003003                     BGT     2$              ; BRANCH IF NO.
1378  006002  021637  006050            CMP     (SP),LOLIM      ; IS IT MORE THAN LOW LIMIT.
1379  006006  002005                     BGE     3$              ; BRANCH IF YES.
1380  006010  104401  001312      2$:    TYPE    ,$QUES          ; TYPE " ? "
1381  006014  104401  001313            TYPE    ,$CRLF          ; TYPE (CR),(LF)
1382  006020  000762                     BR      INLP1           ;
1383  006022  013705  006054      3$:    MOV     WHERE,R5        ; GET BUFFER ADDRESS.
1384  006026  011625              4$:    MOV     (SP),(R5)+      ; SAVE THE # IN RIGHT PLACE.
1385  006030  062716  000002            ADD     #2,(SP)         ; NEXT SEQUENTIAL NUMBER.
1386  006034  105337  006057            DECB    ADRCNT          ; COUNT BY 1.
1387  006040  001372                     BNE     4$              ; BRANCH IF NOT DONE.
1388  006042  005726                     TST     (SP)+           ; POP THE STACK POINTER.
1389  006044  012605                     MOV     (SP)+,R5        ; POP THE REG.5
1390  006046  000002                     RTI
1391  006050  000000              LOLIM: .WORD   0
1392  006052  000000              HILIM: .WORD   0
1393  006054  000000              WHERE: .WORD   0
1394  006056     000              LOBITS:.BYTE   0
1395  006057     000              ADRCNT:.BYTE   0
1396
1397                               ;  ADVANCE TO NEXT TEST HANDLER
1398                               ;---------------------------------
1399                               ;
1400  006060  013716  001442  .ADVANCE:      MOV     NEXT,(SP)              ; CRUNCH STACK WITH ADDRESSOF SCOPE CALL
1401  006064  005037  001444              CLR     LOCK                   ; RESET TIGHT LOOP ADDRESS
1402  006070  000002                      RTI                            ; CHECK TO SEE IF OLD TEST GETS REPEATED
1403
1404                               ;SAVE PC OF TEST THAT FAILED AND R0-R5
1405                               ;-------------------------------------
1406
1407  006072  016637  000004  001460  .SAVOS: MOV     4(SP),SAVPC     ;SAVE R7 (PC)
1408
1409                               ;SAVE R0-R5
1410
1411  006100  010537  001274        SVOS:    MOV     R5,$REG5        ;SAVE R5
1412  006104  010437  001272                 MOV     R4,$REG4        ;SAVE R4
1413  006110  010337  001270                 MOV     R3,$REG3        ;SAVE R3
1414  006114  010237  001266                 MOV     R2,$REG2        ;SAVE R2
1415  006120  010137  001264                 MOV     R1,$REG1        ;SAVE R1
1416  006124  010037  001262                 MOV     R0,$REG0        ;SAVE R0
1417  006130  000002                         RTI                     ;LEAVE.
1418
1419                               ;RESTORE R0-R5
1420
1421  006132  013700  001262        .RESOS:  MOV     $REG0,R0        ;RESTORE R0
1422  006136  013701  001264                 MOV     $REG1,R1        ;RESTORE R1
1423  006142  013702  001266                 MOV     $REG2,R2        ;RESTORE R2
1424  006146  013703  001270                 MOV     $REG3,R3        ;RESTORE R3
```

```
1425  006152  013704  001272                  MOV     $REG4,R4        ;RESTORE R4
1426  006156  013705  001274                  MOV     $REG5,R5        ;RESTORE R5
1427  006162  000002                           RTI                     ;LEAVE
1428
1429                                  ;       ;CONVERT OCTAL NUMBER TO ASCII AND OUTPUT TO TELEPRINTER
1430                                  ;       ;-----------------------------------------------------
1431                                  ;
1432  006164  104401  001313          .CONVR: TYPE    ,$CRLF
1433  006170  010046                  .CNVRT: MOV     R0,-(SP)
1434  006172  010146                          MOV     R1,-(SP)
1435  006174  010346                          MOV     R3,-(SP)
1436  006176  010446                          MOV     R4,-(SP)
1437  006200  010546                          MOV     R5,-(SP)
1438  006202  017601  000012                  MOV     @12(SP),R1
1439  006206  062766  000002  000012          ADD     #2,12(SP)
1440  006214  012137  006406                  MOV     (R1)+,WRDCNT
1441  006220  112137  006410          1$:     MOVB    (R1)+,CHRCNT
1442  006224  112137  006411                  MOVB    (R1)+,SPACNT
1443  006230  013137  006412                  MOV     @(R1)+,BINWRD
1444  006234  122737  000003  006410          CMPB    #3,CHRCNT
1445  006242  001003                          BNE     2$
1446  006244  042737  177400  006412          BIC     #177400,BINWRD
1447  006252  013704  006412          2$:     MOV     BINWRD,R4
1448  006256  113705  006410                  MOVB    CHRCNT,R5
1449  006262  012700  011106                  MOV     #TEMP,R0
1450  006266  010403                  3$:     MOV     R4,R3
1451  006270  042703  177770                  BIC     #177770,R3
1452  006274  062703  000060                  ADD     #060,R3
1453  006300  110320                          MOVB    R3,(R0)+
1454  006302  000241                          CLC
1455  006304  006004                          ROR     R4
1456  006306  000241                          CLC
1457  006310  006004                          ROR     R4
1458  006312  000241                          CLC
1459  006314  006004                          ROR     R4
1460  006316  005305                          DEC     R5
1461  006320  001362                          BNE     3$
1462  006322  012703  011150                  MOV     #MDATA,R3
1463  006326  114023                  4$:     MOVB    -(R0),(R3)+
1464  006330  105337  006410                  DECB    CHRCNT
1465  006334  001374                          BNE     4$
1466  006336  105737  006411                  TSTB    SPACNT
1467  006342  001405                          BEQ     6$
1468  006344  112723  000040          5$:     MOVB    #040,(R3)+
1469  006350  105337  006411                  DECB    SPACNT
1470  006354  001373                          BNE     5$
1471  006356  105013                  6$:     CLRB    (R3)
1472  006360  104401  011150                  TYPE    ,MDATA
1473  006364  005337  006406                  DEC     WRDCNT
1474  006370  001313                          BNE     1$
1475  006372  012605                          MOV     (SP)+,R5
1476  006374  012604                          MOV     (SP)+,R4
1477  006376  012603                          MOV     (SP)+,R3
1478  006400  012601                          MOV     (SP)+,R1
1479  006402  012600                          MOV     (SP)+,R0
1480  006404  000002                          RTI
```

```
1481  006406  000000                  WRDCNT: 0
1482  006410  000000                  CHRCNT: 0
1483          006411                  SPACNT=CHRCNT+1
1484  006412  000000                  BINWRD: 0
1485
1486
1487                                          ;TRAP DISPATCH SERVICE
1488                                          ;ARGUMENT OF TRAP IS EXTRACTED
1489                                          ;AND USED AS OFFSET TO OBTAIN POINTER
1490                                          ;TO SELECTED SUBROUTINE
1491
1492                                  .SBTTL  TRAP DECODER
1493
1494                                  ;;*****************************************************************
1495                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
1496                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
1497                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
1498                                  ;*GO TO THAT ROUTINE.
1499
1500  006414  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
1501  006416  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
1502  006422  005740                          TST     -(R0)           ;;BACKUP BY 2
1503  006424  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
1504  006426  006300                          ASL     R0              ;;POSITION FOR INDEXING
1505  006430  016000  006450                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
1506  006434  000200                          RTS     R0              ;;GO TO ROUTINE
1507
1508
1509                                  ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO
1510
1511  006436  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
1512  006440  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
1513  006446  000002                          RTI                     ;;RESTORE THE PSW
1514
1515                                  .SBTTL  TRAP TABLE
1516
1517                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
1518                                  ;*BY THE "TRAP" INSTRUCTION.
1519
1520                                  ;       ROUTINE
1521                                  ;       -------
1522  006450  006436                  $TRPAD: .WORD   $TRAP2
1523  006452  004414                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
1524
1525
1526  006454  005144                          $RDCHR  ;;CALL=RDCHR    TRAP+2(104402)  TTY TYPEIN CHARACTER ROUTINE
1527  006456  005264                          $RDLIN  ;;CALL=RDLIN    TRAP+3(104403)  TTY TYPEIN STRING ROUTINE
1528  006460  005564                          $RDOCT  ;;CALL=RDOCT    TRAP+4(104404)  READ AN OCTAL NUMBER FROM TTY
1529  006462  004364                          .SCOPI  ;;CALL=SCOPI    TRAP+5(104405)  CALL TO LOOP ON CURRENT DATA HANDLER
1530  006464  006072                          .SAVDS  ;;CALL=SAVDS    TRAP+6(104406)  CALL TO REGISTER SAVE ROUTINE
1531  006466  006132                          .RESDS  ;;CALL=RESDS    TRAP+7(104407)  CALL TO REGISTER RESTORE ROUTINE
1532  006470  007362                          .MSTCLR ;;CALL=MSTCLR   TRAP+10(104410) CALL TO ISSUE A MASTER CLEAR
1533  006472  007332                          .DELAY  ;;CALL=DELAY    TRAP+11(104411) CALL TO DELAY
1534  006474  007400                          .ROMCLK ;;CALL=ROMCLK   TRAP+12(104412) CALL TO CLOCK ROM ONCE
1535  006476  007446                          .DATACLK        ;;CALL=DATACLK  TRAP+13(104413) CALL TO CLOCK DATA
1536  006500  007512                          .TIMER  ;;CALL=TIMER    TRAP+14(104414) CALL TO DELAY A CLOCK TICK
```

```
1537  006502  005724                  $INPUT  ;;CALL=INPUT    TRAP+15(104415) CALL TO OCTAL # INPUT ROUTINE
1538  006504  006164                  .CONVRT ;;CALL=CONVRT   TRAP+16(104416) CALL TO     .....
1539  006506  006170                  .CNVRT  ;;CALL=CNVRT    TRAP+17(104417) CALL TO     .....
1540  006510  006060                  .ADVANCE        ;;CALL=ADVANCE  TRAP+20(104420) CALL TO ADVANCE TO NEXT TEST
1541                            ;
1542                            ;-----------------------------------------------------------------
1543                            ;:*****************************************************************
1544                                    ;ERROR HANDLER
1545                                    ;-------------
1546
1547  006512  004737  011212    $ERROR: JSR     PC,CKSWR        ;CHECK FOR SOFT SWR
1548  006516  032777  010000  172514    BIT     #SW12,@SWR      ;BELL ON ERROR?
1549  006524  001406                    BEQ     XBX             ;BR IF NO BELL
1550  006526  105777  172516            TSTB    @$TPS           ;TTY READY.
1551  006532  100003                    BPL     XBX             ;DON'T WAIT IF TTY NOT READY.
1552  006534  112777  000207  172510    MOVB    #207,@$TPB      ;PUSH A BELL AT THE TTY.
1553  006542  032777  020000  172470  XBX:    BIT     #SW13,@SWR      ;DELETE ERROR PRINT OUT?
1554  006550  001107                    BNE     HALTS           ;BR IF NO PRINT OUT WANTED.
1555  006552  021637  001216            CMP     (SP),$ERRPC     ;WAS THIS ERROR FOUND LAST TIME?
1556  006556  001404                    BEQ     1$              ;BR IF YES
1557  006560  011637  001216            MOV     (SP),$ERRPC     ;RECORD BEING HERE
1558  006564  105037  001203            CLRB    $ERFLG          ;PREPARE HEADER
1559  006570  104406                1$:     SAVOS                   ;SAVE ALL PROC REGISTERS
1560  006572  011605                    MOV     (SP),R5         ;GET THE PC OF ERROR
1561  006574  162705  000002            SUB     #2,R5           ;GET ADDRESS OF TRAP CALL
1562  006600  011504                    MOV     (R5),R4         ;GET ERROR INSTRUCTION
1563  006602  110437  001214            MOVB    R4,$ITEMB       ; COPY ERROR # FOR APT HANDLING
1564  006606  006304                    ASL     R4              ;MULT BY TWO
1565  006610  061504                    ADD     (R5),R4         ;DOUBLE IT
1566  006612  006304                    ASL     R4              ;MULT AGAIN
1567  006614  042704  177001            BIC     #177001,R4      ;CLEAR JUNK
1568  006620  062704  001512            ADD     #$ERRTB,R4      ;GET POINTER
1569  006624  012437  006740            MOV     (R4)+,ERRMSG    ;GET ERROR MESSAGE
1570  006630  012437  006752            MOV     (R4)+,DATAHD    ;GET DATA HEADRER
1571  006634  011437  006764            MOV     (R4),DATABP     ;GET DATA TABLE
1572  006640  105737  001203            TSTB    $ERFLG          ;TYPE HEADREER
1573  006644  001403                    BEQ     TYPMSG          ;BR IF YES
1574  006646  005737  006764            TST     DATABP          ;DOES DATA TABLE EXIST?
1575  006652  001040                    BNE     TYPDAT          ;BR IF YES.
1576  006654  104401  001313    TYPMSG: TYPE    ,$CRLF
1577  006660  104401  001313            TYPE    ,$CRLF
1578  006664  005737  001444            TST     LOCK
1579  006670  001402                    BEQ     1$
1580  006672  104401  010015            TYPE    ,MASTEK
1581  006676  104401  010003    1$:     TYPE    ,MTSTN
1582  006702  104417  007120            CNVRT   ,XTSTN          ;SHOW IT
1583  006706  104401  010072            TYPE    ,MERRPC         ;TYPE PC.
1584  006712  104417  007112            CNVRT   ,ERTABO         ;SHOW IT
1585  006716  104401  001313            TYPE    ,$CRLF          ;GIVE A CR/LF
1586  006722  112737  177777  001203    MOVB    #-1,$ERFLG      ;NO MORE HEADER UNLESS NO DATA TABLE.
1587  006730  005737  006740            TST     ERRMSG          ;IS THERE AN ERROR MESSAGE?
1588  006734  001402                    BEQ     WRKO.FM         ;BR IF NO.
1589  006736  104401                    TYPE                    ;TYPE
1590  006740  000000            ERRMSG: 0                       ;    ERROR MESSAGE
1591  006742                    WRKO.FM:                        ;
1592  006742  005737  005752            TST     DATAHD          ;DATA HEADER?
```

```
1593  006746  001402                        BEQ     TYPDAT          ;BR IF NO
1594  006750  104401                        TYPE                    ;TYPE
1595  006752  000000              DATAHD:   0                       ;    DATA HEADER
1596  006754  005737  006764      TYPDAT:   TST     DATABP          ;DATA TABLE?
1597  006760  001402                        BEQ     RESREG          ;BR IF NO.
1598  006762  104416                        CONVRT                  ;SHOW
1599  006764  000000              DATABP:   0                       ;    DATA TABLE
1600  006766  104407              RESREG:   RESOS                   ;RESTORE PROC REGISTERS
1601  006770  122737  000001  001336 HALTS:  CMPB    #APTENV,$ENV    ; IS APT RUNNING ?
1602  006776  001007                        BNE     3$              ; SKIP APT CALL IF NOT.
1603  007000  113737  001214  007012        MOVB    $ITEMB,6$       ; COPY ERROR #.
1604  007006  004737  004714                JSR     PC,$ATY4        ; CALL APT SERVICES.
1605  007012  000000              6$:       .WORD   0               ; ERROR # GOES HERE.
1606  007014  000777              9$:       BR      9$              ; LOCK HERE.
1607  007016  022737  004070  000042 3$:     CMP     #$ENDAD,@#42    ;IF ACT-11 AUTOMATIC MODE, HALT!!
1608  007024  001403                        BEQ     1$
1609  007026  005777  172206                TST     @SWR            ;HALT ON ERROR?
1610  007032  100005                        BPL     EXITER          ;BR IF NO HALT ON ERROR
1611  007034  010046              1$:       PUSHRO                  ;SAVE R0
1612  007036  016600  000002                MOV     2(SP),R0        ;SHOW ERROR PC IN DATA LIGHTS
1613  007042  000000                        HALT                    ;HALT
1614  007044  012600                        POPRO                   ;GET R0
1615  007046  005237  001212      EXITER:   INC     $ERTTL          ;UPDATE ERROR COUNT
1616  007052  032777  000400  172160        BIT     #SW08,@SWR      ;GOTO TOP OF TEST?
1617  007060  001007                        BNE     1$              ;BR IF YES
1618  007062  032777  002000  172150        BIT     #SW10 @SWR      ;GOTO NEXT TEST?
1619  007070  001407                        BEQ     2$              ;BR IF NO
1620  007072  013737  001442  001206        MOV     NEXT,$LPADR     ;SET FOR NEXT TEST
1621  007100  012706  001200      1$:       MOV     #STACK,SP       ;RESET SP
1622  007104  000177  172076                JMP     @$LPADR         ;GOTO SPECIFIED TEST
1623  007110  000002              2$:       RTI                     ;$LPADR
1624  007112  000001              ERTABO:   1
1625  007114     006     002                .BYTE   6,2
1626  007116  001460                        $AVPC
1627  007120  000001              XTSTN:    1
1628  007122     003     002                .BYTE   3,2
1629  007124  001202                        $TSTNM
1630                                        ;ENTER HERE ON POWER FAILURE
1631                                        ;--------------------------
1632
1633                              .SBTTL  POWER DOWN AND UP ROUTINES
1634
1635                              ;;**********************************************************************
1636                              ;POWER DOWN ROUTINE
1637  007126  012737  007316  000024 $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
1638  007134  012737  000340  000026        MOV     #340,@#PWRVEC+2 ;;PRIO:7
1639  007142  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
1640  007144  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
1641  007146  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
1642  007150  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
1643  007152  010446                        MOV     R4,-(SP)        ;;PUSH R4 ON STACK
1644  007154  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
1645  007156  017746  172056                MOV     @SWR,-(SP)      ;;PUSH @SWR ON STACK
1646  007162  010637  007322                MOV     SP,$SAVR6       ;;SAVE SP
1647  007166  012737  007200  000024        MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
1648  007174  000000                        HALT
```

```
 1649  007176  000776                    BR      .-2             ;;HANG UP
 1650
 1651                                    ;;*****************************************************************
 1652                                    ;POWER UP ROUTINE
 1653  007200  012737  007316  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
 1654  007206  013706  007322            MOV     $SAVR6,SP       ;;GET SP
 1655  007212  005037  007322            CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
 1656  007216  005237  007322      1$:   INC     $SAVR6          ;;WAIT FOR THE INC
 1657  007222  001375                    BNE     1$              ;;OF  WORD
 1658  007224  104401  007562            TYPE    ,MPFAIL         ;
 1659  007230  104417  007324            CNVRT   ,PFTAB          ;
 1660  007234  105037  001203            CLRB    $ERFLG          ;CLEAR ERROR FLAG.
 1661  007240  005037  001216            CLR     $ERRPC          ; CLEAR LAST ERROR PC
 1662  007244  013701  002066            MOV     KMCSR,R1        ; RESTORE DEVICE ADDRESS.
 1663  007250  005011                    CLR     (R1)            ; CLEAR THE CSR.
 1664  007252  104410                    MSTCLR                  ;
 1665  007254  012677  171760            MOV     (SP)+,@SWR      ;;POP STACK INTO @SWR
 1666  007260  012605                    MOV     (SP)+,R5        ;;POP STACK INTO R5
 1667  007262  012604                    MOV     (SP)+,R4        ;;POP STACK INTO R4
 1668  007264  012603                    MOV     (SP)+,R3        ;;POP STACK INTO R3
 1669  007266  012602                    MOV     (SP)+,R2        ;;POP STACK INTO R2
 1670  007270  012601                    MOV     (SP)+,R1        ;;POP STACK INTO R1
 1671  007272  012600                    MOV     (SP)+,R0        ;;POP STACK INTO R0
 1672  007274  012737  007126  000024    MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
 1673  007302  012737  000340  000026    MOV     #340,@#PWRVEC+2 ;;PRIO:7
 1674  007310  104401                    TYPE                    ;;REPORT THE POWER FAILURE
 1675  007312  007562            $PWRMG: .WORD   MPFAIL          ;;POWER FAIL MESSAGE POINTER
 1676  007314  000002                    RTI
 1677  007316  000000            $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
 1678  007320  000776                    BR      .-2             ;;  BEFORE THE POWER DOWN WAS COMPLETE
 1679  007322  000000            $SAVR6: 0                       ;;PUT THE SP HERE
 1680
 1681  007324  000001            PFTAB:  1
 1682  007326      003      002   .BYTE   3,2
 1683  007330  001202                    $TSTNM
 1684
 1685  007332                    .DELAY:
 1686  007332  012777  000020  172534    MOV     #20,@KMP04
 1687  007340  104412                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1688  007342  121111                    121111                  ;POKE CLOCK DELAY BIT
 1689  007344                      1$:
 1690  007344  104412                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 1691  007346  121224                    121224                  ;PORT4=IBUS#11
 1692  007350  032777  000020  172516    BIT     #BIT4,@KMP04    ;IS CLOCK BIT SET?
 1693  007356  001772                    BEQ     1$              ;BR IF NO
 1694  007360  000002                    RTI
 1695
 1696  007362                    .MSTCLR:
 1697  007362  152777  000100  172500    BISB    #BIT6,@KMCSRH   ;SET MASTER CLEAR
 1698  007370  142777  000300  172472    BICB    #BIT6!BIT7,@KMCSRH  ;CLEAR MASTER CLEAR AND RUN
 1699  007376  000002                    RTI                     ;RETURN
 1700
 1701  007400                    .ROMCLK:
 1702  007400  152777  000002  172462    BISB    #BIT1,@KMCSRH   ;SET ROMI
 1703  007406  013677  172464            MOV     @(SP)+,@KMP06   ;LOAD INSTRUCTION IN SEL6
 1704  007412  062746  000002            ADD     #2,-(SP)        ;ADJUST STACK
```

```
1705  007416  032777  000100  171614          BIT     #SW06,@SWR      ;HALT IF SW06 =1
1706  007424  001401                          BEQ     1$              ;BR IF SW06 =0
1707  007426  000000                          HALT                    ;HALT BEFORE CLOCKING INSTRUCTION
1708  007430  152777  000003  172432  1$:     BISB    #BIT1!BIT0,@KMCSRH ;CLOCK INSTRUCTION
1709  007436  142777  000007  172424          BICB    #BIT2!BIT1!BIT0,@KMCSRH  ;CLEAR ROM0, ROMI, STEP
1710  007444  000002                          RTI
1711
1712  007446                          .DATACLK:
1713  007446  013637  011106                  MOV     @(SP)+,TEMP     ;PUT TICK COUNT IN TEMP
1714  007452  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
1715  007456  152777  000020  172404  1$:     BISB    #BIT4,@KMCSRH   ;SET STEP LU
1716  007464  027777  172376  172374          CMP     @KMCSR,@KMCSR   ;WASTE TIME
1717  007472  142777  000020  172370          BICB    #BIT4,@KMCSRH   ;CLEAR STEP LU
1718  007500  005337  011106                  DEC     TEMP            ;DEC TICK COUNT
1719  007504  001364                          BNE     1$              ;BR IF NOT DONE
1720  007506  000002                          RTI                     ;RETURN
1721  007510  000001                  3$:     .BLKW 1
1722
1723  007512                          .TIMER:
1724  007512  013637  011106                  MOV     @(SP)+,TEMP     ;MOVE COUNT TO TEMP
1725  007516  062746  000002                  ADD     #2,-(SP)        ;ADJUST STACK
1726  007522                          1$:
1727  007522  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1728  007524  021364                          021364                  ;PORT4=IBUS* REG11
1729  007526  032777  000002  172340          BIT     #2,@KMP04       ;IS PGM CLOCK BIT CLEAR?
1730  007534  001772                          BEQ     1$              ;BR IF YES
1731  007536                          2$:
1732  007536  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
1733  007540  021364                          021364                  ;PORT4=IBUS* REG11
1734  007542  032777  000002  172324          BIT     #2,@KMP04       ;IS PGM CLOCK BIT SET?
1735  007550  001372                          BNE     2$              ;BR IF YES
1736  007552  005337  011106                  DEC     TEMP            ;DEC COUNT
1737  007556  001361                          BNE     1$              ;BR IF NOT DONE
1738  007560  000002                          RTI                     ;RETURN
1739
1740  007562  050200  051127  043040  MPFAIL: .ASCIZ  <200>/PWR FAILED. RESTART AT TEST /
 (2)  007620  042600  042116  050040  MEPASS: .ASCIZ  <200>/END PASS DZKCC  /
 (2)  007642  051200     000  MR:     .ASCIZ  <200>/R/
 (2)  007645     200  047516  042040  MERR2:  .ASCIZ  <200>/NO DEVICES PRESENT./
 (2)  007672  044600  051516  043125  MERR3:  .ASCIZ  <200>/INSUFFICIENT DATA!/
 (2)  007716  046200  041517  020113  MLOCK:  .ASCIZ  <200>/LOCK ON SELECTED TEST/
 (2)  007745     103  051123  020072  MCSRX:  .ASCIZ  /CSR: /
 (2)  007753     126  041505  020072  MVECX:  .ASCIZ  /VEC: /
 (2)  007761     120  051501  042523  MPASSX: .ASCIZ  /PASSES: /
 (2)  007772  051105  047522  051522  MERRX:  .ASCIZ  /ERRORS: /
 (2)  010003     124  051505  020124  MTSTN:  .ASCIZ  /TEST NO: /
 (2)  010015     052     000  MASTEK: .ASCIZ  /*/
 (2)  010017     200  042523  020124  MNEW:   .ASCIZ  <200>/SET SWITCH REG TO KMC11'S DESIRED ACTIVE./
 (2)  010072  041520  020072     000  MERRPC: .ASCIZ  /PC: /
 (2)  010077     200  020040  020040  XHEAD:  .ASCII  <200>/            MAP OF KMC11 STATUS/
 (2)  010136  020200  020040  020040          .ASCII  <200>/            -------------------/
 (2)  010175     200  020040  041520          .ASCII  <200>/  PC      CSR     STAT1    STAT2    STAT3/
 (2)  010247     200  026455  026455          .ASCIZ  <200>/------   ------   ------   ------   ------/
 (2)  010323     200  047510  020127  NUM:    .ASCIZ  <200>/HOW MANY KMC11'S TO BE TESTED?/
 (2)  010363     200  051503  020122  CSR:    .ASCIZ  <200>/CSR ADDRESS?/
 (2)  010401     200  042526  052103  VEC:    .ASCIZ  <200>/VECTOR ADDRESS?/
```

```
   (2)  010422  041200  020122  051120  PRIO:    .ASCIZ  <200>/BR PRIORITY LEVEL? (4,5,6,7)?/
   (2)  010461     200  044127  041511  MODU:    .ASCIZ  <200>/WHICH LINE UNIT? IF NONE TYPE "N", IF M8201 TYPE "1", IF M8202 TYP
   (2)  010573     200  053523  052111  LINE:    .ASCIZ  <200>/SWITCH PAC#1 (DDCMP LINE #)?/
   (2)  010631     200  053523  052111  BM:      .ASCIZ  <200>/SWITCH PAC#2 (BM873 BOOT ADD)?/
   (2)  010671     200  051511  052040  CONN:    .ASCIZ  <200>/IS THE LOOP BACK CONNECTOR ON?/
   (2)  010731     200  047516  042040  NOACT:   .ASCIZ  <200>/NO DEVICES ARE SELECTED/
   (2)  010762  100200  046513  030503  CONERR:  .ASCIZ  <200><200>/KMC11 AT NONSTANDARD ADDRESS  PC: /
   (2)  011027     200  054105  042520  CNERR:   .ASCIZ  <200>/EXPECTED  FOUND/
   (2)  011050  024040  046513  024503  KMCM:    .ASCIZ  / (KMC) /
   (2)                                  .EVEN
   (2)  011060  000005                  XSTATQ:  5
1741    011062     006     003                   .BYTE   6,3
1742    011064  001276                           $TMP0
1743    011066     006     003                   .BYTE   6,3
1744    011070  001300                           $TMP1
1745    011072     006     003                   .BYTE   6,3
1746    011074  001302                           $TMP2
1747    011076     006     003                   .BYTE   6,3
1748    011100  001304                           $TMP3
1749    011102     006     002                   .BYTE   6,2
1750    011104  001306                           $TMP4
1751                                 .EVEN
1752
1753                                             ;BUFFERS FOR INPUT-OUTPUT
1754
1755    011106  000000                  TEMP:    0
1756            011150                  .=.+40
1757    011150  000000                  MDATA:   0
1758            011212                  .=.+40
1759
1760
1761                                             ;ROUTINE USED TO CHANGE SOFTWARE SWITCH
1762                                             ;REGISTER USING THE CONSOLE TERMINAL
1763                                             ;-----------------------------------
1764
1765    011212  022737  000176  001240  CKSWR:   CMP     #SWREG,SWR      ;IS THE SOFT SWR BEING USED?
1766    011220  001075                           BNE     CKSWRS          ;BR IF NO
1767    011222  132737  000001  001336           BITB    #1,$ENV         ; IS IT RUNNING UNDER APT?
1768    011230  001071                           BNE     CKSWRS          ; EXIT IF YES.
1769    011232  022777  000007  170006           CMP     #7,@$TKB        ;WAS CTRL G TYPED? (7 BIT ASCII)
1770    011240  001404                           BEQ     1$              ;BR IF YES
1771    011242  022777  000207  167776           CMP     #207,@$TKB      ;WAS CTRL G TYPED? (8 BIT ASCII)
1772    011250  001061                           BNE     CKSWRS          ;BR IF NO
1773    011252  010246                  1$:      MOV     R2,-(SP)        ;STORE R2
1774    011254  010346                           MOV     R3,-(SP)        ;STORE R3
1775    011256  010446                           MOV     R4,-(SP)        ;STORE R4
1776    011260  012737  177777  011416           MOV     #-1,SWFLG       ;SET SOFT TYPE OUT FLAG
1777    011266  005002                  CKSWR1:  CLR     R2              ;CLEAR NEW SWR CONTENTS
1778    011270  012704  177777                   MOV     #-1,R4          ;SET FLAG TO ALL ONES
1779    011274  104401  005541                   TYPE    ,$MSWR          ;TYPE "SWR= "
1780    011300  104417                  CKSWR2:  CNVRT                   ;TYPE OUT PRESENT CONTENTS
1781    011302  011452                           SOFTSW                  ;OF SOFT SWITCH REGISTER
1782    011304  104401  005552          CKSWR3:  TYPE    ,$MNEW          ;TYPE "NEW? "
1783    011310  004737  011420          CKSWR4:  JSR     PC,INCHAR       ;GET RESPONSE
1784    011314  022703  000015                   CMP     #15,R3          ;WAS IT A CR?
1785    011320  001424                           BEQ     5$              ;BR IF YES
```

```
1786  011322  022703  000012            CMP     #12,R3          ;WAS IT A LF?
1787  011326  001416                    BEQ     4$              ;BR IF YES
1788  011330  022703  000025            CMP     #25,R3          ;WAS IT CTRL U?
1789  011334  001754                    BEQ     CKSWR1          ;BR IF YES(START OVER)
1790  011336  022703  000007            CMP     #7,R3           ;IF CNTL G GET NEXT CHAR
1791  011342  001762                    BEQ     CKSWR4
1792  011344  005004                    CLR     R4              ;IT MUST BE A DIGIT SO CLR FLAG
1793  011346  042703  177770            BIC     #177770,R3      ;ONLY 0-7 ARE LEGAL SO MASK #OFF BITS
1794  011352  006302                    ASL     R2              ;SHIFT R2 3 TIMES
1795  011354  006302                    ASL     R2
1796  011356  006302                    ASL     R2
1797  011360  050302                    BIS     R3,R2           ;ADD LAST DIGIT
1798  011362  000752                    BR      CKSWR4          ;GET NEXT CHARACTER
1799  011364  012766  002402  000006  4$:     MOV     #.START,6(SP)   ;LF WAS TYPED SO GO TO START
1800  011372  005704                    5$:     TST     R4              ;IS FLAG CLEAR?
1801  011374  001002                            BNE     6$              ;IF NOT DON'T CHANGE SOFT SWR
1802  011376  010277  167636                    MOV     R2,@SWR         ;IF YES THEN WRITE NEW CONTENTS TO SOFT SWR
1803  011402  005037  011416            6$:     CLR     SWFLG           ;CLEAR TYPEOUT FLAG
1804  011406  012604                            MOV     (SP)+,R4        ;RESTORE R4
1805  011410  012603                            MOV     (SP)+,R3        ;RESTORE R3
1806  011412  012602                            MOV     (SP)+,R2        ;RESTORE R2
1807  011414  000207                    CKSWR5: RTS     PC              ;RETURN
1808
1809  011416  000000                    SWFLG:  0
1810
1811  011420  105777  167620            INCHAR: TSTB    @$TKS
1812  011424  100375                            BPL     .-4
1813  011426  017703  167614                    MOV     @$TKB,R3
1814  011432  105777  167612                    TSTB    @$TPS
1815  011436  100375                            BPL     .-4
1816  011440  010377  167606                    MOV     R3,@$TPB
1817  011444  042703  000200                    BIC     #BIT7,R3
1818  011450  000207                            RTS     PC
1819
1820  011452  000001                    SOFTSW: 1
1821  011454     006      002                    .BYTE   6,2
1822  011456  000176                            SWREG
```

```
1823
1824                                        ;
1825                                        ;ROUTINE USED TO "CYCLE" THROUGH UP TO 16 KMC11'S
1826                                        ;THIS ROUTINE SETS UP THE CONTROL ADDRESS FOR THE DIAGNOSTIC
1827                                        ;AND RUNS THE SPECIFIED KMC11'S.  THIS ROUTINE *MUST*
1828                                        ;BE RUN FIRST BEFORE ENTERING THE DIAGNOSTIC FOR THE
1829                                        ;SETUP NECESSARY.
1830                                        ;
1831
1832  011460  005737  001470        CYCLE:  TST     KMACTV          ;ARE ANY KMC11'S TO BE TESTED?
1833  011464  001004                        BNE     1$              ;BR IF OK,
1834  011466  104401  010731                TYPE    ,NOACT          ;NO KMC11'S SELECTED!!
1835  011472  000000                        HALT                    ;STOP THE SHOW.
1836  011474  000776                        BR      .-2             ;DISQUALIFY CONT. SW.
1837  011476  000241                1$:     CLC                     ;CLEAR PROC. CARRY BIT.
1838  011500  006137  001500                ROL     RUN             ;UPDATE POINTER
1839  011504  005537  001500                ADC     RUN             ;CATCH CARRY FROM RUN
1840  011510  062737  000004  001504        ADD     #4,MILK         ;UPDATE POINTER
1841  011516  062737  000010  001502        ADD     #10,CREAM       ;UPDATE ADDRESS POINTER.
1842  011524  022737  002300  001502        CMP     #KM.MAP+200,CREAM
1843  011532  001006                        BNE     2$              ;KEEP GOING; NOT ALL TESTED FOR.
1844  011534  012737  002100  001502        MOV     #KM.MAP,CREAM   ;RESET ADDRESS POINTER.
1845  011542  012737  002302  001504        MOV     #CNT.MAP,MILK   ;RESET PASS COUNT POINTER
1846  011550  033737  001500  001470  2$:   BIT     RUN,KMACTV      ;IS THIS ONE ACTIVE?
1847  011556  001747                        BEQ     1$              ;BR IF NO
1848  011560  013700  001502                MOV     CREAM,R0        ;GET ADDRESS POINTER
1849  011564  013702  001504                MOV     MILK,R2         ;GET PASS COUNT POINTER
1850  011570  012037  002066                MOV     (R0)+,KMCSR     ;LOAD SYSTEM CTRL. REG
1851  011574  011037  002056                MOV     (R0),KMRVEC     ;LOAD VECTOR
1852  011600  042737  177000  002056        BIC     #177000,KMRVEC  ;CLEAR UNWANTED BITS
1853  011606  012037  002050                MOV     (R0)+,STAT1     ;LOAD STAT1
1854  011612  012037  002052                MOV     (R0)+,STAT2     ;LOAD STAT2
1855  011616  012037  002054                MOV     (R0)+,STAT3     ;LOAD STAT3
1856  011622  012237  001324                MOV     (R2)+,$PASS     ;LOAD PASS COUNT
1857  011626  012237  001212                MOV     (R2)+,$ERTTL    ;LOAD ERROR COUNT
1858  011632  012700  000002                MOV     #2,R0           ;SAVE CORE THIS WAY!
1859  011636  013737  002066  002070        MOV     KMCSR,KMCSRH
1860  011644  005237  002070                INC     KMCSRH
1861  011650  013737  002070  002072        MOV     KMCSRH,KMCTL
1862  011656  005237  002072                INC     KMCTL
1863  011662  013737  002072  002074        MOV     KMCTL,KMP04
1864  011670  060037  002074                ADD     R0,KMP04
1865  011674  013737  002074  002076        MOV     KMP04,KMP06
1866  011702  060037  002076                ADD     R0,KMP06
1867
1868  011706  013737  002056  002060        MOV     KMRVEC,KMRLVL   ;PTY LVL
1869  011714  060037  002060                ADD     R0,KMRLVL       ;
1870  011720  013737  002060  002062        MOV     KMRLVL,KMTVEC   ;TX VEC
1871  011726  060037  002062                ADD     R0,KMTVEC       ;
1872  011732  013737  002062  002064        MOV     KMTVEC,KMTLVL   ;TX LVL
1873  011740  060037  002064                ADD     R0,KMTLVL
1874
1875  011744  032737  000002  001446        BIT     #SW01,STRTSW    ;IS TEST NO. SELECTED
1876  011752  001447                        BEQ     7$              ;BR IF NO
1877  011754                        4$:
1878  011754  005737  000042                TST     @#42            ;RUNNING IN AUTO MODE?
```

```
1879  011760  001044                        BNE     7$              ;BR IF YES
1880  011762  104401  001313                TYPE    ,$CRLF
1881  011766  104415                        INPUT
1882  011770  010003                        MTSTN
1883  011772  000001                        1
1884  011774  001000                        1000
1885  011776  001202                        $TSTNM
1886  012000     000                .BYTE   0
1887  012001     001                .BYTE   1
1888  012002  012700  013732                MOV     #TST1,R0
1889  012006  022710                5$:     CMP     (PC)+,(R0)      ;CMP FIRST WORD TO 12737
1890  012010  012737                        MOV     (PC)+,@(PC)+
1891  012012  001020                        BNE     6$              ;BR IF NOT SAME
1892  012014  023760  001202  000002        CMP     $TSTNM,2(R0)    ;DOES $TSTNM MATCH?
1893  012022  001014                        BNE     6$              ;BR IF NO
1894  012024  022760  001202  000004        CMP     #$TSTNM,4(R0)   ;IS LAST WORD OK?
1895  012032  001010                        BNE     6$              ;BR IF NO
1896  012034  010037  001206                MOV     R0,$LPADR       ;IT IS A LEGAL TEST SO DO IT
1897  012040  104401  007642                TYPE    ,MR
1898  012044  042737  000002  001446        BIC     #SW01,STRTSW
1899  012052  000412                        BR      8$
1900  012054  005720                6$:     TST     (R0)+           ;POP R0
1901  012056  020027  034670                CMP     R0,#TLAST+10    ;AT END YET?
1902  012062  001351                        BNE     5$              ;BR IF NO
1903  012064  104401  001312                TYPE    ,$QUES          ;YES ILLEGAL TEST NO.
1904  012070  000731                        BR      4$              ;TRY AGAIN
1905
1906  012072  012737  013732  001206  7$:     MOV     #TST1,$LPADR    ;PREPARE $LPADR ADDRESS
1907  012100  013701  002066          8$:     MOV     KMCSR,R1        ;R1 = BASE KMC11 ADDRESS
1908  012104  000177  167076                JMP     @$LPADR         ;GO START TESTING.
1909
1910
1911                                        ;ROUTINE USED TO "AUTO SIZE" THE KMC11
1912                                        ;CSR AND VECTOR.
1913                                        ;NOTE:  THE CSR MAY BE ANY WHERE IN THE FLOATING
1914                                        ;       ADDRESS RANGE (160000:164000)
1915                                        ;       AND THE VECTOR MAY BE ANY WHERE IN THE
1916                                        ;       FLOATING VECTOR RANGE (300:770)
1917                                        ;
1918
1919  012110                        AUTO.SIZE:
1920  012110  000005                        RESET                   ;INSURE A BUS INIT.
1921  012112  012702  002100        CSRMAP: MOV     #KM.MAP,R2      ;LOAD MAP POINTER.
1922  012116  005022                1$:     CLR     (R2)+           ;ZERO ENTIRE MAP
1923  012120  022702  002300                CMP     #KM.END,R2      ;ALL DONE?
1924  012124  001374                        BNE     1$              ;BR IF NO
1925  012126  005037  001472                CLR     KMNUM           ;SET OCTAL NUMBER OF KMC11'S TO 0
1926  012132  012702  002100                MOV     #KM.MAP,R2      ;R2 POINTS TO KMC MAP
1927  012136  005037  001470                CLR     KMACTV          ;CLEAR ACTIVE
1928  012142  032737  000001  001446        BIT     #SW00,STRTSW    ;QUESTIONS?
1929  012150  001002                        BNE     .+6             ;BR IF YES
1930  012152  000137  012532                JMP     7$              ;IF NO SKIP QUESTIONS
1931  012156  012737  000001  001306        MOV     #1,$TMP4        ;START WITH 1
1932  012164  104415                        INPUT
1933  012166  010323                        NUM
1934  012170  000001                        1
```

```
 1935  012172  000020                          16.
 1936  012174  001302                          $TMP2
 1937  012176     000                          .BYTE   0
 1938  012177     001                          .BYTE   1
 1939  012200  013737  001302  001472          MOV     $TMP2,KMNUM     ;KMNUM = HOW MANY
 1940  012206  104401  001313          12$:    TYPE    ,$CRLF
 1941  012212  104416                          CONVRT                  ;TYPE WHICH KMC IS BEING DONE
 1942  012214  013164                          WHICH                   ;$TMP4 IS WHICH KMC
 1943  012216  005237  001306                  INC     $TMP4
 1944  012222  104415                          INPUT
 1945  012224  010363                          CSR
 1946  012226  160000                          160000
 1947  012230  164000                          164000
 1948  012232  001304                          $TMP3
 1949  012234     000                          .BYTE   0
 1950  012235     001                          .BYTE   1
 1951  012236  013722  001304                  MOV     $TMP3,(R2)+     ;STORE CSR IN MAP
 1952  012242  104415                          INPUT
 1953  012244  010401                          VEC
 1954  012246  000000                          0
 1955  012250  000776                          776
 1956  012252  001304                          $TMP3
 1957  012254     000                          .BYTE   0
 1958  012255     001                          .BYTE   1
 1959  012256  013712  001304                  MOV     $TMP3,(R2)      ;STORE VECTOR IN MAP
 1960  012262  104401                  10$:    TYPE
 1961  012264  010422                          PRIO                    ;ASK WHAT BR LEVEL
 1962  012266  004737  013456                  JSR     PC,INTTY        ;GET RESPONSE
 1963  012272  022703  000024                  CMP     #24,R3          ;
 1964  012276  101014                          BHI     50$             ;BR IF LESS THAN 4
 1965  012300  022703  000027                  CMP     #27,R3          ;
 1966  012304  103411                          BLO     50$             ;BR IF GREATER THAN 7
 1967  012306  012704  000011                  MOV     #11,R4          ;R4 = NUMBER OF SHIFTS
 1968  012312  006303                          ASL     R3              ;SHIFT R3 LEFT
 1969  012314  005304                          DEC     R4              ;DEC SHIFT COUNT
 1970  012316  001375                          BNE     .-4             ;BR IF NOT DONE
 1971  012320  042703  170777                  BIC     #170777,R3      ;BIC UNWANTED BITS
 1972  012324  050312                          BIS     R3,(R2)         ;PUT BR LEVEL IN STATUS MAP
 1973  012326  000403                          BR      8$              ;CONTINUE
 1974  012330  104401                  50$:    TYPE
 1975  012332  001312                          $QUES                   ;RESPONSE IS OUT OF LIMITS
 1976  012334  000752                          BR      10$             ;TRY AGAIN
 1977  012336                          8$:
 1978  012336                          9$:
 1979  012336  104401                  16$:    TYPE
 1980  012340  010461                          MODU                    ;ASK WHICH LINE UNIT
 1981  012342  004737  013456                  JSR     PC,INTTY        ;GET REPLY
 1982  012346  022703  000021                  CMP     #21,R3          ;"1"
 1983  012352  001417                          BEQ     30$
 1984  012354  022703  000022                  CMP     #22,R3          ;"2"
 1985  012360  001412                          BEQ     31$
 1986  012362  022703  000116                  CMP     #116,R3         ;"N"
 1987  012366  001403                          BEQ     32$
 1988  012370  104401                          TYPE
 1989  012372  001312                          $QUES                   ;IF NOT A 1,2 OR N TYPE "?"
 1990  012374  000760                          BR      16$             ;TRY AGIAN
```

```
1991  012376  052722  010000          32$:    BIS     #BIT12,(R2)+    ;SET BIT 12 IN STAT2 IF NO LU
1992  012402  022222                          CMP     (R2)+,(R2)+     ;POP OVER STAT2 AND STAT3
1993  012404  000445                          BR      33$
1994  012406  052712  020000          31$:    BIS     #BIT13,(R2)     ;SET BIT 13 IN STAT2 IF M8202
1995  012412  104401                  30$:    TYPE
1996  012414  010671                          CONN                    ;ASK IF LOOP-BACK IS ON
1997  012416  004737  013456                  JSR     PC,INTTY        ;GET REPLY
1998  012422  022703  000131                  CMP     #131,R3         ;Y
1999  012426  001406                          BEQ     17$
2000  012430  022703  000116                  CMP     #116,R3         ;N
2001  012434  001406                          BEQ     18$
2002  012436  104401                          TYPE
2003  012440  001312                          $QUES                   ;IF NOT Y OR N TYPE "?"
2004  012442  000763                          BR      30$             ;TRY AGAIN
2005  012444  052722  040000          17$:    BIS     #BIT14,(R2)+    ;TURNAROUND IS CONNECTED
2006  012450  000402                          BR      19$
2007  012452  042722  040000          18$:    BIC     #BIT14,(R2)+    ;NO TURNAROUND
2008  012456                          19$:
2009  012456  104415                          INPUT
2010  012460  010573                          LINE
2011  012462  000000                          0
2012  012464  000377                          377
2013  012466  001304                          $TMP3
2014  012470     000                          .BYTE   0
2015  012471     001                          .BYTE   1
2016  012472  113722  001304                  MOVB    $TMP3,(R2)+     ;STORE SWITCH PAC IN MAP
2017  012476  104415                          INPUT
2018  012500  010631                          BM
2019  012502  000000                          0
2020  012504  000377                          377
2021  012506  001304                          $TMP3
2022  012510     000                          .BYTE   0
2023  012511     001                          .BYTE   1
2024  012512  113722  001304                  MOVB    $TMP3,(R2)+     ;STORE SWITCH PAC IN MAP
2025  012516  005722                          TST     (R2)+           ;POP OVER STAT3
2026  012520  005337  001302          33$:    DEC     $TMP2           ;DEC KMC COUNT
2027  012524  001230                          BNE     12$             ;BR IF MORE TO DO
2028  012526  000137  013064                  JMP     13$             ;CONTINUE
2029  012532  012701  160000          7$:     MOV     #160000,R1      ;SET FOR FIRST ADDRESS TO BE TESTED
2030  012536  012737  013156  000004          MOV     #6$,@#4         ;SET FOR NON-EXISTANT DEVICE TIME OUT
2031  012544  005011                  2$:     CLR     (R1)            ;CLEAR SEL0
2032  012546  005711                          TST     (R1)            ;IF KMC11 KMCSR S/B 0
2033  012550  001135                          BNE     3$              ;IF NO DEV ; TRAP TO 4. IF NO BIT 8 THEN NO KMC11
2034  012552  005061  000006                  CLR     6(R1)           ;CLEAR SEL6
2035  012556  005761  000006                  TST     6(R1)           ;IF KMC11 THEN KMRIC S/B =0!
2036  012562  001130                          BNE     3$              ;BR IF NOT KMC11
2037  012564  012711  002000                  MOV     #BIT10,(R1)     ;SET ROMO
2038  012570  005061  000004                  CLR     4(R1)           ;CLEAR SEL4
2039  012574  012761  125252  000006          MOV     #125252,6(R1)   ;WRITE THIS TO SEL6
2040  012602  052711  020000                  BIS     #BIT13,(R1)     ;WRITE IT!
2041  012606  022761  125252  000004          CMP     #125252,4(R1)   ;WAS IT WRITTEN?
2042  012614  001113                          BNE     3$              ;IF NO IT IS NOT CRAM
2043                                  ;AT THIS POINT IT IS ASSUMED THAT R1 HOLDS A KMC11 CSR ADDRESS
2044  012616                          21$:
2045  012616  010122                  22$:    MOV     R1,(R2)+        ;STORE CSR IN CORE TABLE.
2046  012620  012711  001000          15$:    MOV     #BIT9,(R1)      ;CLEAR LINE UNIT LOOP
```

```
2047  012624  005061  000004          CLR     4(R1)           ;CLEAR PORT4
2048  012630  012761  122113  000006  MOV     #122113,6(R1)   ;LOAD INSTRUCTION (CLR DTR)
2049  012636  052711  000400          BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION
2050  012642  012761  021264  000006  MOV     #021264,6(R1)   ;LOAD INSTRUCTION
2051  012650  052711  000400          BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION
2052  012654  122761  000377  000004  CMPB    #377,4(R1)      ;IS IT ALL ONES?
2053  012662  001003                  BNE     .+10            ;BR IF NO
2054  012664  052712  010000          BIS     #BIT12,(R2)     ;IF YES, NO LINE UNIT, SET STATUS BIT
2055  012670  000436                  BR      20$
2056  012672  032761  000002  000004  BIT     #BIT1,4(R1)     ;IS SWITCH A ONE?
2057  012700  001403                  BEQ     .+10            ;BR IF M8201
2058  012702  052712  060000          BIS     #BIT13!BIT14,(R2) ;M8202 ASSUME CONNECTOR
2059  012706  000427                  BR      20$             ;CONNECTOR ON)
2060  012710  032761  000010  000004  BIT     #BIT3,4(R1)     ;IS MRDY SET
2061  012716  001023                  BNE     20$             ;BR IF M8201 NO CONNECTOR (ON LINE)
2062  012720  012761  000100  000004  MOV     #BIT6,4(R1)     ;LOAD PORT4
2063  012726  012761  122113  000006  MOV     #122113,6(R1)   ;LOAD INSTRUCTION
2064  012734  052711  000400          BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION(SET DTR)
2065  012740  012761  021264  000006  MOV     #021264,6(R1)   ;LOAD INSTRUCTION
2066  012746  052711  000400          BIS     #BIT8,(R1)      ;CLOCK INSTRUCTION(READ MODEM REG)
2067  012752  032761  000010  000004  BIT     #BIT3,4(R1)     ;IS MRDY SET NOW?
2068  012760  001402                  BEQ     20$             ;BR IF NO CONNECTOR
2069  012762  052712  040000          BIS     #BIT14,(R2)     ;SET STATUS BIT FOR CONNECTOR
2070  012766  005722          20$:    TST     (R2)+           ;POP POINTER
2071  012770  012761  021324  000006  MOV     #021324,6(R1)   ;PUT INSTRUCTION IN PORT6
2072  012776  012711  001400          MOV     #BIT9!BIT8,(R1) ;PORT4=LU 15
2073  013002  156122  000004          BISB    4(R1),(R2)+     ;STORE DOCMP LINE # IN TABLE
2074  013006  012761  021344  000006  MOV     #021344,6(R1)   ;PORT6=INSTRUCTION
2075  013014  012711  001400          MOV     #BIT8!BIT9,(R1) ;CLOCK INSTR.
2076  013020  156122  000004          BISB    4(R1),(R2)+     ;STORE BM873 ADD IN TABLE
2077  013024  005722                  TST     (R2)+           ;POP OVER STAT3
2078  013026  005011                  CLR     (R1)            ;CLEAR ROMI
2079  013030  005237  001472          INC     KMNUM           ;UPDATE DEVICE COUNTER
2080  013034  022737  000020  001472  CMP     #20,K'NUM       ;ARE MAX. NO. OF DEV FOUND?
2081  013042  001410                  BEQ     13$             ;YES DON'T LOOK FOR ANY MORE.
2082  013044  005011          3$:     CLR     (R1)            ;CLEAR BIT 10
2083  013046  005061  000006          CLR     6(R1)           ;CLEAR SEL 6
2084  013052  062701  000010  14$:    ADD     #10,R1          ;UPDATE CSR POINTER ADDRESS
2085  013056  022701  164000          CMP     #164000,R1
2086  013062  001230                  BNE     2$              ;BR IF MORE ADDRESS TO CHECK.
2087  013064  005037  001470  13$:    CLR     KMACTV
2088  013070  005737  001472          TST     KMNUM           ;WERE ANY KMC11'S FOUND AT ALL?
2089  013074  001423                  BEQ     5$              ;ERROR AUTO SIZER FOUND NO KMC11'S IN THIS SYS.
2090  013076  013701  001472          MOV     KMNUM,R1
2091  013102  010137  001476          MOV     R1,SAVNUM       ;SAVE NUMBER OF DEVICES
2092  013106  000241          4$:     CLC
2093  013110  006137  001470          ROL     KMACTV          ;GENERATE ACTIVE REGISTER OF DEVICES.
2094  013114  005237  001470          INC     KMACTV          ;SET THE BIT
2095  013120  005301                  DEC     R1
2096  013122  001371                  BNE     4$              ;BR IF MORE TO GENERATE
2097  013124  012737  000006  000004  MOV     #6,@#4          ;RESTORE TRAP VECTOR
2098  013132  013737  001470  001474  MOV     KMACTV,SAVACT   ;SAVE ACTIVE REGISTER
2099  013140  000137  013172          JMP     VECMAP          ;GO FIND THE VECTOR NOW.
2100  013144  104401  007645  5$:     TYPE    ,MERR2          ;NOTIFY OPR THAT NO KMC11'S FOUND.
2101  013150  005000                  CLR     R0              ;MAKE DATA LIGHTS ZERO
2102  013152  000000                  HALT                    ;STOP THE SHOW
```

```
2103  013154  000776                    BR      .-2             ;DISABLE CONT. SW.
2104  013156  012716  013052      6$:   MOV     #14$,(SP)       ;ENTERED BY NON-EXISTANT TIME-OUT.
2105  013162  000002                    RTI                     ;RETURN TO MAINSTREAM
2106
2107  013164  000001              WHICH: 1
2108  013166      002     002           .BYTE   2,2
2109  013170  001306                    $TMP4
2110
2111  013172  032737  000001  001446 VECMAP: BIT     #SW00,STRTSW
2112  013200  001114                    BNE     5$
2113  013202  012737  000340  000022        MOV     #340,@#22       ;SET IOT TRAP PRIO TO 7
2114  013210  012737  013364  000020        MOV     #4$,@#20        ;SET IOT TRAP VECTOR
2115  013216  012702  002100              MOV     #KM.MAP,R2      ;SET SOFTWARE POINTER
2116  013222  012700  000300              MOV     #300,R0         ;FLOATING VECTORS START HERE.
2117  013226  012701  000302              MOV     #302,R1         ;PC OF IOT INSTR.
2118  013232  010120              1$:     MOV     R1,(R0)+        ;START FILLING VECTOR AREA
2119  013234  012721  000004              MOV     #4,(R1)+        ;WITH .+2; IOT
2120  013240  022021                    CMP     (R0)+,(R1)+     ;ADD 2 TO R0 +R1
2121  013242  020127  001000              CMP     R1,#1000
2122  013246  101771                    BLOS    1$              ;BR IF MORE TO FILL
2123  013250  013737  001470  001276        MOV     KMACTV,$TMP0    ;STORE TEMPORALLY
2124  013256  006037  001276      2$:     ROR     $TMP0           ;BRING OUT A BIT
2125  013262  103063                    BCC     5$              ;BR IF ALL DONE
2126  013264  012704  000012              MOV     #12,R4          ;R4 IS INDEX REGISTER
2127  013270  016437  013442  177776        MOV     BRLVL(R4),PS    ;SET PS TO 7
2128  013276  011201                    MOV     (R2),R1
2129  013300  012761  000200  000004        MOV     #200,4(R1)
2130  013306  012711  001000              MOV     #BIT9,(R1)      ;SET ROMI
2131  013312  012761  121111  000006        MOV     #121111,6(R1)   ;PUT INSTRUCTION IN PORT6
2132  013320  012711  001400              MOV     #BIT9!BIT8,(R1) ;FORCE AN INTERRUPT
2133  013324  105200              7$:     INCB    R0              ;STALL
2134  013326  001376                    BNE     .-2             ;FOR TIME TO INTERUPT
2135  013330  162704  000002              SUB     #2,R4           ;GET NEXT LOWEST PS LEVEL
2136  013334  001404                    BEQ     6$              ;BR IF R4 = 0
2137  013336  016437  013442  177776        MOV     BRLVL(R4),PS    ;MOVE NEXT LOWER LEVEL IN PS
2138  013344  000767                    BR      7$              ;BR TO DELAY
2139  013346  052762  005300  000002  6$:   BIS     #5300,2(R2)     ;NO INTERUPT ASSUME 300 AT LEVEL 5 AND FIX KMC1! LATER
2140  013354  005011              3$:     CLR     (R1)            ;CLEAR ROMI
2141  013356  062702  000010              ADD     #10,R2          ;POP SOFTWARE POINTER
2142  013362  000735                    BR      2$              ;KEEP GOING
2143  013364  051662  000002      4$:     BIS     (SP),2(R2)      ;GET VECTOR ADDRESS
2144  013370  042762  000007  000002        BIC     #7,2(R2)        ;CLEAR JUNK
2145  013376  016405  013444              MOV     BRLVL+2(R4),R5  ;GET BR LEVEL OF KMC11
2146  013402  006305                    ASL     R5              ;SHIFT LEVEL 4 PLACES
2147  013404  006305                    ASL     R5              ;TO THE LEFT FOR THE
2148  013406  006305                    ASL     R5              ;STATUS TABLE
2149  013410  006305                    ASL     R5
2150  013412  042705  170777              BIC     #170777,R5      ;CLEAR UNWANTED BITS
2151  013416  050562  000002              BIS     R5,2(R2)        ;PUT BR LEVEL IN STATUS TABLE
2152  013422  022626                    CMP     (SP)+,(SP)+     ;POP IOT JUNK OFF STACK
2153  013424  012716  013354              MOV     #3$,(SP)        ;SET FOR RETURN
2154  013430  000002                    RTI
2155  013432  012737  004134  000020  5$:   MOV     #$SCOPE,@#20    ; RESTORE SCOPE VECTOR
2156  013440  000207                    RTS     PC              ;ALL DONE WITH "AUTO SIZING"
2157
2158  013442  000000              BRLVL:  PRO     ;LEVEL 0
```

```
2159  013444  000000                          PR0     ;LEVEL 0
2160  013446  000200                          PR4     ;LEVEL 4
2161  013450  000240                          PR5     ;LEVEL 5
2162  013452  000300                          PR6     ;LEVEL 6
2163  013454  000340                          PR7     ;LEVEL 7
2164
2165
2166  013456  105777  165562        INTTY:    TSTB    @#TKS           ;WAIT FOR DONE
2167  013462  100375                          BPL     .-4
2168  013464  017703  165556                  MOV     @#TKB,R3        ;PUT CHAR IN R3
2169  013470  105777  165554                  TSTB    @#TPS           ;WAIT UNTIL PRINTER IS READY
2170  013474  100375                          BPL     .-4
2171  013476  010377  165550                  MOV     R3,@#TPB        ;ECHO CHAR
2172  013502  042703  000240                  BIC     #BIT7!BIT5,R3   ;MASK OFF LOWER CASE
2173  013506  000207                          RTS     PC              ;RETURN
2174
2175  013510                        APT.SIZE:
2176  013510  000005                          RESET
2177  013512  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
2178  013514  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
2179  013516  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
2180  013520  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
2181  013522  005037  013724                  CLR     VECTR           ; CLEAR THE LOCAL VARIABLE
2182  013526  005037  013730                  CLR     PRIRTY          ; CLEAN UP LOCAL VARIABLE
2183  013532  013700  001376                  MOV     $CDW1,R0        ; GET THE DEVICE COUNT
2184  013536  010037  001476                  MOV     R0,SAVNUM       ; SAVE THE NO. OF DEVICES
2185  013542  012701  001346                  MOV     #$MAMS1,R1      ; GET EXTRA INFO, BITS POINTER
2186  013546  013737  001372  013726          MOV     $BASE,BASE      ; GET BASE CSR ADDRESS
2187  013554  113737  001366  013724          MOVB    $VECT1,VECTR    ; GET THE VECTOR
2188  013562  113737  001367  013730          MOVB    $VECT1+1,PRIRTY ; GET THE PRIORITY
2189  013570  013737  001374  001470          MOV     $DEVM,KMACTV    ; SAVE THE KMC'S SELECTED ACTIVE
2190  013576  013737  001470  001474          MOV     KMACTV,SAVACT   ; SAVE THE ACTIVE REGISTER
2191  013604  012702  001402                  MOV     #$DDWO,R2       ; GET ADDRESS OF FIRST DEVICE DESCRIPTOR WORD
2192  013610  012703  002100                  MOV     #KM.MAP,R3      ; GET POINTER TO DEVICE MAP
2193  013614  005023                3$:       CLR     (R3)+           ; CLEAR DEVICE MAP
2194  013616  022703  002300                  CMP     #KM.END,R3      ; IS WHOLE DEV.MAP CLEARED?
2195  013622  003374                          BGT     3$              ; NO, THEN GO ON.
2196  013624  012703  002100                  MOV     #KM.MAP,R3      ; RESTORE DEV.MAP POINTER.
2197  013630  013723  013726        1$:       MOV     BASE,(R3)+      ; LOAD CSR ADDRESS
2198  013634  112163  000001                  MOVB    (R1)+,1(R3)     ; GET EXTRA INFO. BITS
2199  013640  006213                          ASR     (R3)            ; SET IT IN RIGHT POSITION.
2200  013642  006213                          ASR     (R3)            ; SET IT IN RIGHT POSITION.
2201  013644  053713  013730                  BIS     PRIRTY,(R3)     ; GET PRIORITY IN STAT1
2202  013650  006313                          ASL     (R3)            ; SET THEM IN RIGHT POSITION
2203  013652  006313                          ASL     (R3)            ;  "   "   "   "   "
2204  013654  006313                          ASL     (R3)            ;  "   "   "   "   "
2205  013656  006313                          ASL     (R3)            ;  "   "   "   "   "
2206  013660  053723  013724                  BIS     VECTR,(R3)+     ; GET THE VECTOR IN STAT1.
2207  013664  012223                          MOV     (R2)+,(R3)+     ; GET THE STAT2 FROM DDWXX
2208  013666  005723                          TST     (R3)+           ; SKIP OVER STAT3
2209  013670  005300                          DEC     R0              ; COUNT BY 1
2210  013672  001407                          BEQ     2$              ; ALL DONE?
2211  013674  062737  000010  013726          ADD     #10,BASE        ; INCREMENT BASE CSR ADDRESS BY 10
2212  013702  062737  000010  013724          ADD     #10,VECTR       ; INCREMENT VECTOR ADDRESS BY 10
2213  013710  000747                          BR      1$              ; SET THE NEXT MAP ENTRY
2214  013712                        2$:
```

```
2215  013712  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
2216  013714  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
2217  013716  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
2218  013720  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
2219  013722  000207                        RTS     PC              ; RETURN
2220  013724  000000                VECTR:  .WORD   0
2221  013726  000000                BASE:   .WORD   0
2222  013730  000000                PRIRTY: .WORD   0
2223
2224
2225                                        ;*************************** TEST 1 ****************************
2226                                        ;*VERIFY THAT REFERENCING UNIBUS DEVICE REGISTERS
2227                                        ;*DOES NOT CAUSE A TIME OUT TRAP
2228                                        ;;***********************************************************
2229
2230                                        ;  TEST 1
2231                                        ;--------------
2232                                ;;**********************************************************
2233  013732  000004                TST1:   SCOPE
2234  013734  012737  000001  001202        MOV     #1,$TSTNM               ; LOAD THE NO. OF THIS TEST
2235  013742  012737  014042  001442        MOV     #TST2,NEXT              ; POINT TO THE START OF NEXT TEST.
2236  013750  012737  014002  001444        MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
2237                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
2238  013756  013701  002066                MOV     KMCSR,R1        ;R1 CONTAINS BASE KMC11 ADDRESS
2239  013762  012700  000004                MOV     #4,R0           ;4 REGISTERS TO BE TESTED
2240  013766  012737  014034  000004        MOV     #2$,4           ;SET UP TIMEOUT TRAP
2241  013774  012737  000340  000006        MOV     #340,6          ;LEVEL 7
2242  014002  005711                1$:     TST     (R1)            ;REFERENCE DEVICE REGISTER
2243  014004  000240                        NOP
2244  014006  104405                        SCOP1                   ;SW09=1?
2245  014010  062701  000002                ADD     #2,R1           ;NEXT REGISTER
2246  014014  005300                        DEC     R0              ;DEC REGISTER COUNT
2247  014016  001371                        BNE     1$              ;BR IF NOT LAST REGISTER
2248  014020  012737  000006  000004        MOV     #6,4            ;RESTORE LOC 4
2249  014026  005037  000006                CLR     6               ;RESTORE LOC 6
2250  014032  104420                        ADVANCE                 ; ADVANCE TO NEXT TEST
2251  014034  011602                2$:     MOV     (SP),R2         ;GET PC OF TRAP
2252  014036  104001                        ERROR   1               ;TIME-OUT ERROR
2253  014040  000002                        RTI
2254
2255
2256                                        ;*************************** TEST 2 ****************************
2257                                        ;*VERIFY THAT RUN CAN BE CLEARED
2258                                        ;;***********************************************************
2259
2260                                        ;  TEST 2
2261                                        ;--------------
2262                                ;;**********************************************************
2263  014042  000004                TST2:   SCOPE
2264  014044  012737  000002  001202        MOV     #2,$TSTNM               ; LOAD THE NO. OF THIS TEST
2265  014052  012737  014072  001442        MOV     #TST3,NEXT              ; POINT TO THE START OF NEXT TEST.
2266                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
2267  014060  005011                        CLR     (R1)            ;CLEAR KMCSR
2268  014062  005005                        CLR     R5              ;CLEAR "EXPECTED"
2269  014064  011104                        MOV     (R1),R4         ;PUT KMCSR IN "FOUND"
2270  014066  001401                        BEQ     1$              ;BR IF CLEARED
```

```
2271  014070  104002                          ERROR   2               ;ERROR KMCSR NOT CLEARED
2272  014072                          1$:
2273
2274
2275                                          ;*************************** TEST 3 ***************************
2276                                          ;*UNIBUS REGISTER WORD DUAL ADDRESSING TEST
2277                                          ;*LOAD ALL REGISTERS WITH INCREMENTING PATTERN
2278                                          ;*READ BACK ALL REGISTERS TO VERIFY CORRECT ADDRESSING
2279                                          ;;*************************************************************
2280
2281                                          ;  TEST 3
2282                                          ;-------------
2283                                          ;;*************************************************************
2284  014072  000004                  TST3:   SCOPE
2285  014074  012737  000003  001202          MOV     #3,$TSTNM               ; LOAD THE NO. OF THIS TEST
2286  014102  012737  014222  001442          MOV     #TST4,NEXT              ; POINT TO THE START OF NEXT TEST.
2287  014110  012737  014124  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
2288                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
2289  014116  104410                          MSTCLR                  ;MASTER CLEAR KMC11
2290  014120  012700  000001                  MOV     #1,R0           ;START PATTERN AT 1
2291  014124  005011                  1$:     CLR     (R1)            ;CLEAR REGISTER
2292  014126  010005                          MOV     R0,R5           ;PUT DATA IN "EXPECTED"
2293  014130  010011                          MOV     R0,(R1)         ;WRITE KMC REGISTER WITH PATTERN
2294  014132  011104                          MOV     (R1),R4         ;READ KMC REGISTER INTO "FOUND"
2295  014134  020504                          CMP     R5,R4           ;IS DATA CORRECT
2296  014136  001401                          BEQ     2$              ;BR IF YES
2297  014140  104002                          ERROR   2               ;DATA ERROR
2298  014142  104405                  2$:     SCOP1                   ;SW09=1?
2299  014144  005721                          TST     (R1)+           ;NEXT REGISTER
2300  014146  005200                          INC     R0              ;INCREMENT DATA PATTERN
2301  014150  022700  000005                  CMP     #5,R0           ;LAST REGISTER?
2302  014154  001363                          BNE     1$              ;BR IF NO
2303  014156  013701  002066                  MOV     KMCSR,R1        ;BASE KMC11  ADDRESS TO R1
2304  014162  012700  000001                  MOV     #1,R0           ;RESTART PATTERN AT 1
2305  014166  012737  014174  001444          MOV     #3$,LOCK        ;NEW SCOP1
2306  014174  010005                  3$:     MOV     R0,R5           ;PUT DATA IN "EXPECTED"
2307  014176  011104                          MOV     (R1),R4         ;READ KMC REGISTER INTO "FOUND"
2308  014200  020504                          CMP     R5,R4           ;IS DATA CORRECT
2309  014202  001401                          BEQ     4$              ;BR IF YES
2310  014204  104002                          ERROR   2               ;DUAL ADDRESSING ERROR
2311  014206  104405                  4$:     SCOP1                   ;SW09=1?
2312  014210  005721                          TST     (R1)+           ;NEXT REGISTER
2313  014212  005200                          INC     R0              ;INCREMENT PATTERN
2314  014214  022700  000005                  CMP     #5,R0           ;LAST REGISTER?
2315  014220  001365                          BNE     3$              ;BR IF NO
2316
2317
2318                                          ;*************************** TEST 4 ***************************
2319                                          ;*CONTROL STATUS REGISTER WRITE/READ TEST
2320                                          ;*SET BIT0, VERIFY BIT0 WAS SET
2321                                          ;*CLEAR BIT0, VERIFY BIT0 WAS CLEARED
2322                                          ;;*************************************************************
2323
2324                                          ;  TEST 4
2325                                          ;-------------
2326                                          ;;*************************************************************
```

```
2327  014222  000004                    TST4:   SCOPE
2328  014224  012737  000004  001202            MOV     #4,$ISTNM               ; LOAD THE NO. OF THIS TEST
2329  014232  012737  014320  001442            MOV     #TST5,NEXT              ; POINT TO THE START OF NEXT TEST.
2330  014240  012737  014250  001444            MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA
2331  014246  104410                            MSTCLR                  ;MASTER CLEAR KMC11
2332  014250  013701  002066            1$:     MOV     KMCSR,R1        ;PUT REGISTER ADDRESS IN R1
2333  014254  012705  000001                    MOV     #BIT0,R5        ;PUT DATA IN "EXPECTED"
2334  014260  010511                            MOV     R5,(R1)         ;WRITE BIT 0
2335  014262  011104                            MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
2336  014264  020504                            CMP     R5,R4           ;IS DATA CORRECT
2337  014266  001401                            BEQ     2$              ;BR IF YES
2338  014270  104002                            ERROR   2               ;DATA ERROR
2339  014272  104405                    2$:     SCOP1                   ;SW09 UP?
2340  014274  012737  014302  001444            MOV     #3$,LOCK        ;NEW SCOP1
2341  014302  042711  000001            3$:     BIC     #BIT0,(R1)      ;CLEAR BIT 0
2342  014306  005005                            CLR     R5              ;CLEAR "EXPECTED"
2343  014310  011104                            MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
2344  014312  001402                            BEQ     4$              ;BR IF ZERO
2345  014314  104002                            ERROR   2               ;DATA ERROR BIT0 NOT CLEARED
2346  014316  104405                            SCOP1                   ;SW09 UP?
2347  014320                            4$:
2348
2349
2350                                    ;*************************** TEST 5 ****************************
2351                                    ;*CONTROL STATUS REGISTER WRITE/READ TEST
2352                                    ;*SET BIT1, VERIFY BIT1 WAS SET
2353                                    ;*CLEAR BIT1, VERIFY BIT1 WAS CLEARED
2354                                    ;;*************************************************************
2355
2356                                    ;  TEST 5
2357                                    ;-------------
2358                                    ;;*************************************************************
2359  014320  000004                    TST5:   SCOPE
2360  014322  012737  000005  001202            MOV     #5,$TSTNM               ; LOAD THE NO. OF THIS TEST
2361  014330  012737  014416  001442            MOV     #TST6,NEXT              ; POINT TO THE START OF NEXT TEST.
2362  014336  012737  014346  001444            MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
2363  014344  104410                            MSTCLR                  ;MASTER CLEAR KMC11
2364  014346  013701  002066            1$:     MOV     KMCSR,R1        ;PUT REGISTER ADDRESS IN R1
2365  014352  012705  000002                    MOV     #BIT1,R5        ;PUT DATA IN "EXPECTED"
2366  014356  010511                            MOV     R5,(R1)         ;WRITE BIT 1
2367  014360  011104                            MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
2368  014362  020504                            CMP     R5,R4           ;IS DATA CORRECT
2369  014364  001401                            BEQ     2$              ;BR IF YES
2370  014366  104002                            ERROR   2               ;DATA ERROR
2371  014370  104405                    2$:     SCOP1                   ;SW09 UP?
2372  014372  012737  014400  001444            MOV     #3$,LOCK        ;NEW SCOP1
2373  014400  042711  000002            3$:     BIC     #BIT1,(R1)      ;CLEAR BIT 1
2374  014404  005005                            CLR     R5              ;CLEAR "EXPECTED"
2375  014406  011104                            MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
2376  014410  001402                            BEQ     4$              ;BR IF ZERO
2377  014412  104002                            ERROR   2               ;DATA ERROR BIT1 NOT CLEARED
2378  014414  104405                            SCOP1                   ;SW09 UP?
2379  014416                            4$:
2380
2381
2382                                    ;*************************** TEST 6 ****************************
```

```
                                        ;*CONTROL STATUS REGISTER WRITE/READ TEST
                                        ;*SET BIT2, VERIFY BIT2 WAS SET
                                        ;*CLEAR BIT2, VERIFY BIT2 WAS CLEARED
                                        ;;*****************************************************************

                                        ;  TEST 6
                                        ;-------------
                                    ;;*****************************************************************
014416  000004                  TST6:   SCOPE
014420  012737  000006  001202          MOV     #6,$TSTNM               ; LOAD THE NO. OF THIS TEST
014426  012737  014514  001442          MOV     #TST7,NEXT              ; POINT TO THE START OF NEXT TEST.
014434  012737  014444  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
014442  104410                          MSTCLR                  ;MASTER CLEAR KMC11
014444  013701  002066          1$:     MOV     KMCSR,R1        ;PUT REGISTER ADDRESS IN R1
014450  012705  000004                  MOV     #BIT2,R5        ;PUT DATA IN "EXPECTED"
014454  010511                          MOV     R5,(R1)         ;WRITE BIT 2
014456  011104                          MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
014460  020504                          CMP     R5,R4           ;IS DATA CORRECT
014462  001401                          BEQ     2$              ;BR IF YES
014464  104002                          ERROR   2               ;DATA ERROR
014466  104405                  2$:     SCOP1                   ;SW09 UP?
014470  012737  014476  001444          MOV     #3$,LOCK        ;NEW SCOP1
014476  042711  000004          3$:     BIC     #BIT2,(R1)      ;CLEAR BIT 2
014502  005005                          CLR     R5              ;CLEAR "EXPECTED"
014504  011104                          MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
014506  001402                          BEQ     4$              ;BR IF ZERO
014510  104002                          ERROR   2               ;DATA ERROR BIT2 NOT CLEARED
014512  104405                          SCOP1                   ;SW09 UP?
014514                          4$:


                                        ;*************************** TEST 7 ****************************
                                        ;*CONTROL STATUS REGISTER WRITE/READ TEST
                                        ;*SET BIT5, VERIFY BIT5 WAS SET
                                        ;*CLEAR BIT5, VERIFY BIT5 WAS CLEARED
                                        ;;*****************************************************************

                                        ;  TEST 7
                                        ;-------------
                                    ;;*****************************************************************
014514  000004                  TST7:   SCOPE
014516  012737  000007  001202          MOV     #7,$TSTNM               ; LOAD THE NO. OF THIS TEST
014524  012737  014612  001442          MOV     #TST10,NEXT             ; POINT TO THE START OF NEXT TEST.
014532  012737  014542  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
014540  104410                          MSTCLR                  ;MASTER CLEAR KMC11
014542  013701  002066          1$:     MOV     KMCSR,R1        ;PUT REGISTER ADDRESS IN R1
014546  012705  000040                  MOV     #BIT5,R5        ;PUT DATA IN "EXPECTED"
014552  010511                          MOV     R5,(R1)         ;WRITE BIT 5
014554  011104                          MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
014556  020504                          CMP     R5,R4           ;IS DATA CORRECT
014560  001401                          BEQ     2$              ;BR IF YES
014562  104002                          ERROR   2               ;DATA ERROR
014564  104405                  2$:     SCOP1                   ;SW09 UP?
014566  012737  014574  001444          MOV     #3$,LOCK        ;NEW SCOP1
014574  042711  000040          3$:     BIC     #BIT5,(R1)      ;CLEAR BIT 5
014600  005005                          CLR     R5              ;CLEAR "EXPECTED"
```

```
2439  014602  011104                          MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
2440  014604  001402                          BEQ     4$              ;BR IF ZERO
2441  014606  104002                          ERROR   2               ;DATA ERROR BITS NOT CLEARED
2442  014610  104405                          SCOPE                   ;SW09 UP?
2443  014612                          4$:
2444
2445
2446                                          ;************************** TEST 10 **************************
2447                                          ;*CONTROL STATUS REGISTER WRITE/READ TEST
2448                                          ;*SET BIT6, VERIFY BIT6 WAS SET
2449                                          ;*CLEAR BIT6, VERIFY BIT6 WAS CLEARED
2450                                          ;;*********************************************************
2451
2452                                          ;  TEST 10
2453                                          ;-------------
2454                                          ;;*********************************************************
2455  014612  000004                  TST10:  SCOPE
2456  014614  012737  000010  001202          MOV     #10,$TSTNM              ; LOAD THE NO. OF THIS TEST
2457  014622  012737  014710  001442          MOV     #TST11,NEXT             ; POINT TO THE START OF NEXT TEST.
2458  014630  012737  014640  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
2459  014636  104410                          MSTCLR                  ;MASTER CLEAR KMC11
2460  014640  013701  002066          1$:     MOV     KMCSR,R1        ;PUT REGISTER ADDRESS IN R1
2461  014644  012705  000100                  MOV     #BIT6,R5        ;PUT DATA IN "EXPECTED"
2462  014650  010511                          MOV     R5,(R1)         ;WRITE BIT 6
2463  014652  011104                          MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
2464  014654  020504                          CMP     R5,R4           ;IS DATA CORRECT
2465  014656  001401                          BEQ     2$              ;BR IF YES
2466  014660  104002                          ERROR   2               ;DATA ERROR
2467  014662  104405                  2$:     SCOPE                   ;SW09 UP?
2468  014664  012737  014672  001444          MOV     #3$,LOCK        ;NEW SCOPE
2469  014672  042711  000100          3$:     BIC     #BIT6,(R1)      ;CLEAR BIT 6
2470  014676  005005                          CLR     R5              ;CLEAR "EXPECTED"
2471  014700  011104                          MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
2472  014702  001402                          BEQ     4$              ;BR IF ZERO
2473  014704  104002                          ERROR   2               ;DATA ERROR BIT6 NOT CLEARED
2474  014706  104405                          SCOPE                   ;SW09 UP?
2475  014710                          4$:
2476
2477
2478                                          ;************************** TEST 11 **************************
2479                                          ;*CONTROL STATUS REGISTER WRITE/READ TEST
2480                                          ;*SET BIT7, VERIFY BIT7 WAS SET
2481                                          ;*CLEAR BIT7, VERIFY BIT7 WAS CLEARED
2482                                          ;;*********************************************************
2483
2484                                          ;  TEST 11
2485                                          ;-------------
2486                                          ;;*********************************************************
2487  014710  000004                  TST11:  SCOPE
2488  014712  012737  000011  001202          MOV     #11,$TSTNM              ; LOAD THE NO. OF THIS TEST
2489  014720  012737  015006  001442          MOV     #TST12,NEXT             ; POINT TO THE START OF NEXT TEST.
2490  014726  012737  014736  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
2491  014734  104410                          MSTCLR                  ;MASTER CLEAR KMC11
2492  014736  013701  002066          1$:     MOV     KMCSR,R1        ;PUT REGISTER ADDRESS IN R1
2493  014742  012705  000200                  MOV     #BIT7,R5        ;PUT DATA IN "EXPECTED"
2494  014746  010511                          MOV     R5,(R1)         ;WRITE BIT 7
```

```
                                        MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
                                        CMP     R5,R4           ;IS DATA CORRECT
                                        BEQ     2$              ;BR IF YES
                                        ERROR   2               ;DATA ERROR
                                2$:     SCOPE                   ;SW09 UP?
015000  012737  014770  001444          MOV     #3$,LOCK        ;NEW SCOPE
015006  042711  000200          3$:     BIC     #BIT7,(R1)      ;CLEAR BIT 7
015012  005005                          CLR     R5              ;CLEAR "EXPECTED"
015014  011104                          MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
015016  001402                          BEQ     4$              ;BR IF ZERO
015020  104002                          ERROR   2               ;DATA ERROR BIT7 NOT CLEARED
015022  104405                          SCOPE                   ;SW09 UP?
015024                          4$:

                                ;*************************** TEST 12 ****************************
                                ;*CONTROL STATUS REGISTER WRITE/READ TEST
                                ;*SET BIT9, VERIFY BIT9 WAS SET
                                ;*CLEAR BIT9, VERIFY BIT9 WAS CLEARED
                                ;;*************************************************************

                                ;  TEST 12
                                ;-------------
                                ;;*************************************************************
015024  000004                  TST12:  SCOPE
015026  012737  000012  001202          MOV     #12,$TSTNM              ; LOAD THE NO. OF THIS TEST
015034  012737  015104  001442          MOV     #TST13,NEXT             ; POINT TO THE START OF NEXT TEST.
015042  012737  015052  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
015050  104410                          MSTCLR                  ;MASTER CLEAR KMC11
015052  013701  002066          1$:     MOV     KMCSR,R1        ;PUT REGISTER ADDRESS IN R1
015056  012705  001000                  MOV     #BIT9,R5        ;PUT DATA IN "EXPECTED"
015062  010511                          MOV     R5,(R1)         ;WRITE BIT 9
015064  011104                          MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
015066  020504                          CMP     R5,R4           ;IS DATA CORRECT
015070  001401                          BEQ     2$              ;BR IF YES
015072  104002                          ERROR   2               ;DATA ERROR
015074  104405                  2$:     SCOPE                   ;SW09 UP?
015076  012737  015104  001444          MOV     #3$,LOCK        ;NEW SCOPE
015104  042711  001000          3$:     BIC     #BIT9,(R1)      ;CLEAR BIT 9
015110  005005                          CLR     R5              ;CLEAR "EXPECTED"
015112  011104                          MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
015114  001402                          BEQ     4$              ;BR IF ZERO
015116  104002                          ERROR   2               ;DATA ERROR BIT9 NOT CLEARED
015120  104405                          SCOPE                   ;SW09 UP?
015122                          4$:

                                ;*************************** TEST 13 ****************************
                                ;*CONTROL STATUS REGISTER WRITE/READ TEST
                                ;*SET BIT11, VERIFY BIT11 WAS SET
                                ;*CLEAR BIT11, VERIFY BIT11 WAS CLEARED
                                ;;*************************************************************

                                ;  TEST 13
                                ;-------------
                                ;;*************************************************************
```

```
015104  000004                          TST13:  SCOPE
015106  012737  000013  001202                  MOV     #13,$TSTNM              ; LOAD THE NO. OF THIS TEST
015114  012737  015202  001442                  MOV     #TST14,NEXT             ; POINT TO THE START OF NEXT TEST.
015122  012737  015132  001444                  MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
015130  104410                                  MSTCLR                  ;MASTER CLEAR KMC11
015132  013701  002066                  1$:     MOV     KMCSR,R1        ;PUT REGISTER ADDRESS IN R1
015136  012705  004000                          MOV     #BIT11,R5       ;PUT DATA IN "EXPECTED"
015142  010511                                  MOV     R5,(R1)         ;WRITE BIT 11
015144  011104                                  MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
015146  020504                                  CMP     R5,R4           ;IS DATA CORRECT
015150  001401                                  BEQ     2$              ;BR IF YES
015152  104002                                  ERROR   2               ;DATA ERROR
015154  104405                          2$:     SCOPE1                  ;SW09 UP?
015156  012737  015164  001444                  MOV     #3$,LOCK        ;NEW SCOP1
015164  042711  004000                  3$:     BIC     #BIT11,(R1)     ;CLEAR BIT 11
015170  005005                                  CLR     R5              ;CLEAR "EXPECTED"
015172  011104                                  MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
015174  001402                                  BEQ     4$              ;BR IF ZERO
015176  104002                                  ERROR   2               ;DATA ERROR BIT11 NOT CLEARED
015200  104405                                  SCOP1                   ;SW09 UP?
015202                                  4$:


                                        ;*************************** TEST 14 ****************************
                                        ;*CONTROL STATUS REGISTER WRITE/READ TEST
                                        ;*SET BIT12, VERIFY BIT12 WAS SET
                                        ;*CLEAR BIT12, VERIFY BIT12 WAS CLEARED
                                        ;;*************************************************************

                                        ;  TEST 14
                                        ;--------------
                                        ;;*************************************************************
015202  000004                          TST14:  SCOPE
015204  012737  000014  001202                  MOV     #14,$TSTNM              ; LOAD THE NO. OF THIS TEST
015212  012737  015300  001442                  MOV     #TST15,NEXT             ; POINT TO THE START OF NEXT TEST.
015220  012737  015230  001444                  MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
015226  104410                                  MSTCLR                  ;MASTER CLEAR KMC11
015230  013701  002066                  1$:     MOV     KMCSR,R1        ;PUT REGISTER ADDRESS IN R1
015234  012705  010000                          MOV     #BIT12,R5       ;PUT DATA IN "EXPECTED"
015240  010511                                  MOV     R5,(R1)         ;WRITE BIT 12
015242  011104                                  MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
015244  020504                                  CMP     R5,R4           ;IS DATA CORRECT
015246  001401                                  BEQ     2$              ;BR IF YES
015250  104002                                  ERROR   2               ;DATA ERROR
015252  104405                          2$:     SCOP1                   ;SW09 UP?
015254  012737  015262  001444                  MOV     #3$,LOCK        ;NEW SCOP1
015262  042711  010000                  3$:     BIC     #BIT12,(R1)     ;CLEAR BIT 12
015266  005005                                  CLR     R5              ;CLEAR "EXPECTED"
015270  011104                                  MOV     (R1),R4         ;READ CONTROL STATUS REGISTER
015272  001402                                  BEQ     4$              ;BR IF ZERO
015274  104002                                  ERROR   2               ;DATA ERROR BIT12 NOT CLEARED
015276  104405                                  SCOP1                   ;SW09 UP?
015300                                  4$:


                                        ;*************************** TEST 15 ****************************
```

```
                                        ;*CONTROL OUT REGISTER WRITE/READ TEST
                                        ;*SET BIT0, VERIFY BIT0 WAS SET
                                        ;*CLEAR BIT0, VERIFY BIT0 WAS CLEARED
                                        ;;*****************************************************************

                                        ;  TEST 15
                                        ;-------------
                                        ;;*****************************************************************
015300  000004                  TST15:  SCOPE
015302  012737  000015  001202          MOV     #15,$TSTNM              ; LOAD THE NO. OF THIS TEST
015310  012737  015376  001442          MOV     #TST16,NEXT             ; POINT TO THE START OF NEXT TEST.
015316  012737  015326  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
015324  104410                          MSTCLR                  ;MASTER CLEAR KMC11
015326  013701  002072          1$:     MOV     KMCTL,R1        ;PUT REGISTER ADDRESS IN R1
015332  012705  000001                  MOV     #BIT0,R5        ;PUT DATA IN "EXPECTED"
015336  010511                          MOV     R5,(R1)         ;WRITE BIT 0
015340  011104                          MOV     (R1),R4         ;READ CONTROL OUT REGISTER
015342  020504                          CMP     R5,R4           ;IS DATA CORRECT
015344  001401                          BEQ     2$              ;BR IF YES
015346  104002                          ERROR   2               ;DATA ERROR
015350  104405                  2$:     SCOP1                   ;SW09 UP?
015352  012737  015360  001444          MOV     #3$,LOCK        ;NEW SCOP1
015360  042711  000001          3$:     BIC     #BIT0,(R1)      ;CLEAR BIT 0
015364  005005                          CLR     R5              ;CLEAR "EXPECTED"
015366  011104                          MOV     (R1),R4         ;READ CONTROL OUT REGISTER
015370  001402                          BEQ     4$              ;BR IF ZERO
015372  104002                          ERROR   2               ;DATA ERROR BIT0 NOT CLEARED
015374  104405                          SCOP1                   ;SW09 UP?
015376                          4$:
                                        ;;*****************************************************************
                                        ;*************************** TEST 16 ****************************
                                        ;*CONTROL OUT REGISTER WRITE/READ TEST
                                        ;*SET BIT1, VERIFY BIT1 WAS SET
                                        ;*CLEAR BIT1, VERIFY BIT1 WAS CLEARED
                                        ;;*****************************************************************

                                        ;  TEST 16
                                        ;-------------
                                        ;;*****************************************************************
015376  000004                  TST16:  SCOPE
015400  012737  000016  001202          MOV     #16.,$TSTNM             ; LOAD THE NO. OF THIS TEST
015406  012737  015474  001442          MOV     #TST17,NEXT             ; POINT TO THE START OF NEXT TEST.
015414  012737  015424  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
015422  104410                          MSTCLR                  ;MASTER CLEAR KMC11
015424  013701  002072          1$:     MOV     KMCTL,R1        ;PUT REGISTER ADDRESS IN R1
015430  012705  000002                  MOV     #BIT1,R5        ;PUT DATA IN "EXPECTED"
015434  010511                          MOV     R5,(R1)         ;WRITE BIT 1
015436  011104                          MOV     (R1),R4         ;READ CONTROL OUT REGISTER
015440  020504                          CMP     R5,R4           ;IS DATA CORRECT
015442  001401                          BEQ     2$              ;BR IF YES
015444  104002                          ERROR   2               ;DATA ERROR
015446  104405                  2$:     SCOP1                   ;SW09 UP?
015450  012737  015456  001444          MOV     #3$,LOCK        ;NEW SCOP1
015456  042711  000002          3$:     BIC     #BIT1,(R1)      ;CLEAR BIT 1
015462  005005                          CLR     R5              ;CLEAR "EXPECTED"
```

```
2663  015464  011104                          MOV     (R1),R4         ;READ CONTROL OUT REGISTER
2664  015466  001402                          BEQ     4$              ;BR IF ZERO
2665  015470  104002                          ERROR   2               ;DATA ERROR BIT1 NOT CLEARED
2666  015472  104405                          SCOPE                   ;SW09 UP?
2667  015474                          4$:
2668
2669
2670                                          ;*************************** TEST 17 ****************************
2671                                          ;*CONTROL OUT REGISTER WRITE/READ TEST
2672                                          ;*SET BIT2, VERIFY BIT2 WAS SET
2673                                          ;*CLEAR BIT2, VERIFY BIT2 WAS CLEARED
2674                                          ;;************************************************************
2675
2676                                          ;  TEST 17
2677                                          ;-------------
2678                                          ;;************************************************************
2679  015474  000004                  TST17:  SCOPE
2680  015476  012737  000017  001202          MOV     #17,$TSTNM              ; LOAD THE NO. OF THIS TEST
2681  015504  012737  015572  001442          MOV     #TST20,NEXT             ; POINT TO THE START OF NEXT TEST.
2682  015512  012737  015522  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
2683  015520  104410                          MSTCLR                  ;MASTER CLEAR KMC11
2684  015522  013701  002072          1$:     MOV     KMCTL,R1        ;PUT REGISTER ADDRESS IN R1
2685  015526  012705  000004                  MOV     #BIT2,R5        ;PUT DATA IN "EXPECTED"
2686  015532  010511                          MOV     R5,(R1)         ;WRITE BIT 2
2687  015534  011104                          MOV     (R1),R4         ;READ CONTROL OUT REGISTER
2688  015536  020504                          CMP     R5,R4           ;IS DATA CORRECT
2689  015540  001401                          BEQ     2$              ;BR IF YES
2690  015542  104002                          ERROR   2               ;DATA ERROR
2691  015544  104405                  2$:     SCOPE                   ;SW09 UP?
2692  015546  012737  015554  001444          MOV     #3$,LOCK                ;NEW SCOPE
2693  015554  042711  000004          3$:     BIC     #BIT2,(R1)      ;CLEAR BIT 2
2694  015560  005005                          CLR     R5              ;CLEAR "EXPECTED"
2695  015562  011104                          MOV     (R1),R4         ;READ CONTROL OUT REGISTER
2696  015564  001402                          BEQ     4$              ;BR IF ZERO
2697  015566  104002                          ERROR   2               ;DATA ERROR BIT2 NOT CLEARED
2698  015570  104405                          SCOPE                   ;SW09 UP?
2699  015572                          4$:
2700
2701
2702                                          ;*************************** TEST 20 ****************************
2703                                          ;*CONTROL OUT REGISTER WRITE/READ TEST
2704                                          ;*SET BIT6, VERIFY BIT6 WAS SET
2705                                          ;*CLEAR BIT6, VERIFY BIT6 WAS CLEARED
2706                                          ;;************************************************************
2707
2708                                          ;  TEST 20
2709                                          ;-------------
2710                                          ;;************************************************************
2711  015572  000004                  TST20:  SCOPE
2712  015574  012737  000020  001202          MOV     #20,$TSTNM              ; LOAD THE NO. OF THIS TEST
2713  015602  012737  015670  001442          MOV     #TST21,NEXT             ; POINT TO THE START OF NEXT TEST.
2714  015610  012737  015620  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
2715  015616  104410                          MSTCLR                  ;MASTER CLEAR KMC11
2716  015620  013701  002072          1$:     MOV     KMCTL,R1        ;PUT REGISTER ADDRESS IN R1
2717  015624  012705  000100                  MOV     #BIT6,R5        ;PUT DATA IN "EXPECTED"
2718  015630  010511                          MOV     R5,(R1)         ;WRITE BIT 6
```

```
2719  015632  011104                          MOV     (R1),R4         ;READ CONTROL OUT REGISTER
2720  015634  020504                          CMP     R5,R4           ;IS DATA CORRECT
2721  015636  001401                          BEQ     2$              ;BR IF YES
2722  015640  104002                          ERROR   2               ;DATA ERROR
2723  015642  104405                  2$:     SCOPE1                  ;SWR9 UP?
2724  015644  012737  015652  001444          MOV     #3$,LOCK        ;NEW SCOPE1
2725  015652  042711  000100          3$:     BIC     #BIT6,(R1)      ;CLEAR BIT 6
2726  015656  005005                          CLR     R5              ;CLEAR "EXPECTED"
2727  015660  011104                          MOV     (R1),R4         ;READ CONTROL OUT REGISTER
2728  015662  001402                          BEQ     4$              ;BR IF ZERO
2729  015664  104002                          ERROR   2               ;DATA ERROR BIT6 NOT CLEARED
2730  015666  104405                          SCOPE1                  ;SWR9 UP?
2731  015670                          4$:
2732
2733
2734                                          ;************************** TEST 21 **************************
2735                                          ;*CONTROL OUT REGISTER WRITE/READ TEST
2736                                          ;*SET BIT7, VERIFY BIT7 WAS SET
2737                                          ;*CLEAR BIT7, VERIFY BIT7 WAS CLEARED
2738                                          ;;*********************************************************
2739
2740                                          ;  TEST 21
2741                                          ;-------------
2742                                          ;;*********************************************************
2743  015670  000004                  TST21:  SCOPE
2744  015672  012737  000021  001202          MOV     #21,$TSTNM              ; LOAD THE NO. OF THIS TEST
2745  015700  012737  015766  001442          MOV     #TST22,NEXT             ; POINT TO THE START OF NEXT TEST.
2746  015706  012737  015716  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
2747  015714  104410                          MSTCLR                  ;MASTER CLEAR KMC11
2748  015716  013701  002072          1$:     MOV     KMCTL,R1        ;PUT REGISTER ADDRESS IN R1
2749  015722  012705  000200                  MOV     #BIT7,R5        ;PUT DATA IN "EXPECTED"
2750  015726  010511                          MOV     R5,(R1)         ;WRITE BIT 7
2751  015730  011104                          MOV     (R1),R4         ;READ CONTROL OUT REGISTER
2752  015732  020504                          CMP     R5,R4           ;IS DATA CORRECT
2753  015734  001401                          BEQ     2$              ;BR IF YES
2754  015736  104002                          ERROR   2               ;DATA ERROR
2755  015740  104405                  2$:     SCOPE1                  ;SWR9 UP?
2756  015742  012737  015750  001444          MOV     #3$,LOCK        ;NEW SCOPE1
2757  015750  042711  000200          3$:     BIC     #BIT7,(R1)      ;CLEAR BIT 7
2758  015754  005005                          CLR     R5              ;CLEAR "EXPECTED"
2759  015756  011104                          MOV     (R1),R4         ;READ CONTROL OUT REGISTER
2760  015760  001402                          BEQ     4$              ;BR IF ZERO
2761  015762  104002                          ERROR   2               ;DATA ERROR BIT7 NOT CLEARED
2762  015764  104405                          SCOPE1                  ;SWR9 UP?
2763  015766                          4$:
2764
2765
2766                                          ;************************** TEST 22 **************************
2767                                          ;*CONTROL OUT REGISTER WRITE/READ TEST
2768                                          ;*SET BIT12, VERIFY BIT12 WAS SET
2769                                          ;*CLEAR BIT12, VERIFY BIT12 WAS CLEARED
2770                                          ;;*********************************************************
2771
2772                                          ;  TEST 22
2773                                          ;-------------
2774                                          ;;*********************************************************
```

```
2775  015766  000004                    TST22:  SCOPE
2776  015770  012737  000022  001202            MOV     #22,$TSTNM              ; LOAD THE NO. OF THIS TEST
2777  015776  012737  016064  001442            MOV     #TST23,NEXT             ; POINT TO THE START OF NEXT TEST.
2778  016004  012737  016014  001444            MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
2779  016012  104410                            MSTCLR                  ;MASTER CLEAR KMC11
2780  016014  013701  002072            1$:     MOV     KMCTL,R1        ;PUT REGISTER ADDRESS IN R1
2781  016020  012705  010000                    MOV     #BIT12,R5       ;PUT DATA IN "EXPECTED"
2782  016024  010511                            MOV     R5,(R1)         ;WRITE BIT 12
2783  016026  011104                            MOV     (R1),R4         ;READ CONTROL OUT REGISTER
2784  016030  020504                            CMP     R5,R4           ;IS DATA CORRECT
2785  016032  001401                            BEQ     2$              ;BR IF YES
2786  016034  104002                            ERROR   2               ;DATA ERROR
2787  016036  104405                    2$:     SCOP1                   ;SWO9 UP?
2788  016040  012737  016046  001444            MOV     #3$,LOCK        ;NEW SCOP1
2789  016046  042711  010000            3$:     BIC     #BIT12,(R1)     ;CLEAR BIT 12
2790  016052  005005                            CLR     R5              ;CLEAR "EXPECTED"
2791  016054  011104                            MOV     (R1),R4         ;READ CONTROL OUT REGISTER
2792  016056  001402                            BEQ     4$              ;BR IF ZERO
2793  016060  104002                            ERROR   2               ;DATA ERROR BIT12 NOT CLEARED
2794  016062  104405                            SCOP1                   ;SWO9 UP?
2795  016064                            4$:
2796
2797
2798                                    ;*************************** TEST 23 ***************************
2799                                    ;*CONTROL OUT REGISTER WRITE/READ TEST
2800                                    ;*SET BIT13, VERIFY BIT13 WAS SET
2801                                    ;*CLEAR BIT13, VERIFY BIT13 WAS CLEARED
2802                                    ;:***************************************************************
2803
2804                                    ;  TEST 23
2805                                    ;-------------
2806                                    ;:***************************************************************
2807  016064  000004                    TST23:  SCOPE
2808  016066  012737  000023  001202            MOV     #23,$TSTNM              ; LOAD THE NO. OF THIS TEST
2809  016074  012737  016162  001442            MOV     #TST24,NEXT             ; POINT TO THE START OF NEXT TEST.
2810  016102  012737  016112  001444            MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
2811  016110  104410                            MSTCLR                  ;MASTER CLEAR KMC11
2812  016112  013701  002072            1$:     MOV     KMCTL,R1        ;PUT REGISTER ADDRESS IN R1
2813  016116  012705  020000                    MOV     #BIT13,R5       ;PUT DATA IN "EXPECTED"
2814  016122  010511                            MOV     R5,(R1)         ;WRITE BIT 13
2815  016124  011104                            MOV     (R1),R4         ;READ CONTROL OUT REGISTER
2816  016126  020504                            CMP     R5,R4           ;IS DATA CORRECT
2817  016130  001401                            BEQ     2$              ;BR IF YES
2818  016132  104002                            ERROR   2               ;DATA ERROR
2819  016134  104405                    2$:     SCOP1                   ;SWO9 UP?
2820  016136  012737  016144  001444            MOV     #3$,LOCK        ;NEW SCOP1
2821  016144  042711  020000            3$:     BIC     #BIT13,(R1)     ;CLEAR BIT 13
2822  016150  005005                            CLR     R5              ;CLEAR "EXPECTED"
2823  016152  011104                            MOV     (R1),R4         ;READ CONTROL OUT REGISTER
2824  016154  001402                            BEQ     4$              ;BR IF ZERO
2825  016156  104002                            ERROR   2               ;DATA ERROR BIT13 NOT CLEARED
2826  016160  104405                            SCOP1                   ;SWO9 UP?
2827  016162                            4$:
2828
2829
2830                                    ;*************************** TEST 24 ***************************
```

```
                                        ;*PORT4 REGISTER WRITE/READ TEST
                                        ;*FLOAT A ONE THROUGH PORT4 REGISTER
                                        ;*FLOAT A ZERO THROUGH PORT4 REGISTER
                                        ;;*****************************************************************

                                        ;  TEST 24
                                        ;-------------
                                        ;;*****************************************************************
016162  000004                  TST24:  SCOPE
016164  012737  000024  001202          MOV     #24,$TSTNM              ; LOAD THE NO. OF THIS TEST
016172  012737  016306  001442          MOV     #TST25,NEXT             ; POINT TO THE START OF NEXT TEST.
016200  012737  016220  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
016206  104410                          MSTCLR                  ;MASTER CLEAR KMC11
016210  013701  002074                  MOV     KMP04,R1                ;PUT REGISTER ADDRESS IN R1
016214  012700  000001                  MOV     #1,R0           ;START WITH BIT0
016220                          64$:
016220  010005                          MOV     R0,R5           ;PUT "EXPECTED" IN R5
016222  010511                          MOV     R5,(R1)         ;WRITE PORT4 REGISTER
016224  011104                          MOV     (R1),R4         ;READ PORT4 REGISTER
016226  020504                          CMP     R5,R4           ;COMPARE EXPECTED AND FOUND
016230  001401                          BEQ     65$             ;BR IF OK
016232  104002                          ERROR   2               ;WRITE/READ ERROR
016234  104405                  65$:    SCOP1                   ;LOOP TO 64$ IF SW09=1
016236  000241                          CLC                     ;CLEAR CARRY
016240  006100                          ROL     R0              ;SHIFT TO NEXT BIT
016242  001366                          BNE     64$             ;BR IF NOT DONE YET?
016244  012737  016256  001444          MOV     #66$,LOCK               ;NEW SCOP1
016252  012700  000001                  MOV     #1,R0           ;START WITH BIT0
016256                          66$:
016256  005100                          COM     R0              ;CHANGE TO A FLOATING ZERO
016260  010005                          MOV     R0,R5           ;PUT "EXPECTED" IN R5
016262  010511                          MOV     R5,(R1)         ;WRITE PORT4 REGISTER
016264  011104                          MOV     (R1),R4         ;READ PORT4 REGISTER
016266  020504                          CMP     R5,R4           ;COMPARE EXPECTED AND FOUND
016270  001401                          BEQ     67$             ;BR IF OK
016272  104002                          ERROR   2               ;WRITE/READ ERROR
016274  104405                  67$:    SCOP1                   ;LOOP TO 66$ IF SW09=1
016276  005100                          COM     R0              ;CHANGE BACK TO A FLOATING ONE
016300  000241                          CLC                     ;CLEAR CARRY
016302  006100                          ROL     R0              ;SHIFT TO NEXT BIT
016304  001364                          BNE     66$             ;BR IF NOT DONE YET?


                                        ;*************************** TEST 25 ****************************
                                        ;*PORT6 REGISTER WRITE/READ TEST
                                        ;*FLOAT A ONE THROUGH PORT6 REGISTER
                                        ;*FLOAT A ZERO THROUGH PORT6 REGISTER
                                        ;;*****************************************************************

                                        ;  TEST 25
                                        ;-------------
                                        ;;*****************************************************************
016306  000004                  TST25:  SCOPE
016310  012737  000025  001202          MOV     #25,$TSTNM              ; LOAD THE NO. OF THIS TEST
016316  012737  016432  001442          MOV     #TST26,NEXT             ; POINT TO THE START OF NEXT TEST.
016324  012737  016344  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
```

```
2887  016332  104410                          MSTCLR                  ;MASTER CLEAR KMC11
2888  016334  013701  002076                  MOV     KMP06,R1                ;PUT REGISTER ADDRESS IN R1
2889  016340  012700  000001                  MOV     #1,R0           ;START WITH BIT0
2890  016344                          64$:
2891  016344  010005                          MOV     R0,R5           ;PUT "EXPECTED" IN R5
2892  016346  010511                          MOV     R5,(R1)         ;WRITE PORT6 REGISTER
2893  016350  011104                          MOV     (R1),R4         ;READ PORT6 REGISTER
2894  016352  020504                          CMP     R5,R4           ;COMPARE EXPECTED AND FOUND
2895  016354  001401                          BEQ     65$             ;BR IF OK
2896  016356  104002                          ERROR   2               ;WRITE/READ ERROR
2897  016360  104405                  65$:    SCOP1                   ;LOOP TO 64$ IF SW09=1
2898  016362  000241                          CLC                     ;CLEAR CARRY
2899  016364  006100                          ROL     R0              ;SHIFT TO NEXT BIT
2900  016366  001366                          BNE     64$             ;BR IF NOT DONE YET?
2901  016370  012737  016402  001444          MOV     #66$,LOCK       ;NEW SCOP1
2902  016376  012700  000001                  MOV     #1,R0           ;START WITH BIT0
2903  016402                          66$:
2904  016402  005100                          COM     R0              ;CHANGE TO A FLOATING ZERO
2905  016404  010005                          MOV     R0,R5           ;PUT "EXPECTED" IN R5
2906  016406  010511                          MOV     R5,(R1)         ;WRITE PORT6 REGISTER
2907  016410  011104                          MOV     (R1),R4         ;READ PORT6 REGISTER
2908  016412  020504                          CMP     R5,R4           ;COMPARE EXPECTED AND FOUND
2909  016414  001401                          BEQ     67$             ;BR IF OK
2910  016416  104002                          ERROR   2               ;WRITE/READ ERROR
2911  016420  104405                  67$:    SCOP1                   ;LOOP TO 66$ IF SW09=1
2912  016422  005100                          COM     R0              ;CHANGE BACK TO A FLOATING ONE
2913  016424  000241                          CLC                     ;CLEAR CARRY
2914  016426  006100                          ROL     R0              ;SHIFT TO NEXT BIT
2915  016430  001364                          BNE     66$             ;BR IF NOT DONE YET?
2916
2917
2918                                  ;************************** TEST 26 **************************
2919                                  ;*UNIBUS REGISTER BYTE DUAL ADDRESSING TEST
2920                                  ;*LOAD ALL REGISTERS WITH INCREMENTING PATTERN
2921                                  ;*READ BACK ALL REGISTERS TO VERIFY CORRECT ADDRESSING
2922                                  ;;*********************************************************************
2923
2924                                  ;  TEST 26
2925                                  ;--------------
2926                          ;;*********************************************************************
2927  016432  000004          TST26:  SCOPE
2928  016434  012737  000026  001202          MOV     #26,$TSTNM              ; LOAD THE NO. OF THIS TEST
2929  016442  012737  016562  001442          MOV     #TST27,NEXT             ; POINT TO THE START OF NEXT TEST.
2930  016450  012737  016464  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
2931                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
2932  016456  104410                          MSTCLR                  ;MASTER CLEAR KMC11
2933  016460  012700  000001                  MOV     #1,R0           ;START PATTERN AT 1
2934  016464  105011          1$:     CLRB    (R1)                    ;CLEAR REGISTER
2935  016466  110005                          MOVB    R0,R5           ;PUT DATA IN "EXPECTED"
2936  016470  110011                          MOVB    R0,(R1)         ;WRITE KMC REGISTER WITH PATTERN
2937  016472  111104                          MOVB    (R1),R4         ;READ KMC REGISTER INTO "FOUND"
2938  016474  020504                          CMP     R5,R4           ;IS DATA CORRECT
2939  016476  001401                          BEQ     2$              ;BR IF YES
2940  016500  104002                          ERROR   2               ;DATA ERROR
2941  016502  104405          2$:     SCOP1                   ;SW09=1?
2942  016504  105721                          TSTB    (R1)+           ;NEXT REGISTER
```

```
2943  016506  005200                          INC     R0              ;INCREMENT DATA PATTERN
2944  016510  022700  000011                  CMP     #11,R0          ;LAST REGISTER?
2945  016514  001363                          BNE     1$              ;BR IF NO
2946  016516  013701  002066                  MOV     KMCSR,R1        ;BASE KMC11  ADDRESS TO R1
2947  016522  012700  000001                  MOV     #1,R0           ;RESTART PATTERN AT 1
2948  016526  012737  016534  001444          MOV     #3$,LOCK        ;NEW SCOPE1
2949  016534  110005                  3$:     MOVB    R0,R5           ;PUT DATA IN "EXPECTED"
2950  016536  111104                          MOVB    (R1),R4         ;READ KMC REGISTER INTO "FOUND"
2951  016540  020504                          CMP     R5,R4           ;IS DATA CORRECT
2952  016542  001401                          BEQ     4$              ;BR IF YES
2953  016544  104002                          ERROR   2               ;DUAL ADDRESSING ERROR
2954  016546  104405                  4$:     SCOPE1                  ;SW09=1?
2955  016550  105721                          TSTB    (R1)+           ;NEXT REGISTER
2956  016552  005200                          INC     R0              ;INCREMENT PATTERN
2957  016554  022700  000011                  CMP     #11,R0          ;LAST REGISTER?
2958  016560  001365                          BNE     3$              ;BR IF NO
2959
2960
2961                                          ;************************** TEST 27 **************************
2962                                          ;*MAINTENANCE INSTRUCTION REGISTER TEST
2963                                          ;*VERIFY THAT THE MAINT IR CAN BE WRITTEN TO ALL ZEROS'
2964                                          ;*AND ALL ONES'. VERIFY THAT IT IS CLEARED ON A BUS RESET.
2965                                          ;;*************************************************************
2966
2967                                          ;  TEST 27
2968                                          ;-------------
2969                                  ;;*************************************************************
2970  016562  000004                  TST27:  SCOPE
2971  016564  012737  000027  001202          MOV     #27,$TSTNM              ; LOAD THE NO. OF THIS TEST
2972  016572  012737  016722  001442          MOV     #TST30,NEXT             ; POINT TO THE START OF NEXT TEST.
2973  016600  012737  016616  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
2974                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
2975  016606  104410                          MSTCLR                  ;MASTER CLEAR KMC11
2976  016610  012711  003000                  MOV     #BIT9!BIT10,(R1) ;SEL6 IS NOW THE IR
2977  016614  005005                          CLR     R5              ;PUT "EXPECTED" IN R5
2978  016616  010561  000006          1$:     MOV     R5,6(R1)        ;CLEAR THE IR
2979  016622  016104  000006                  MOV     6(R1),R4        ;READ THE IR
2980  016626  020504                          CMP     R5,R4           ;IS IT CLEARED?
2981  016630  001401                          BEQ     2$              ;BR IF YES
2982  016632  104023                          ERROR   23              ;ERROR IR IS NOT CLEAR
2983  016634  104405                  2$:     SCOPE1                  ;LOOP TO 1$ IF SW09=1
2984  016636  012737  016650  001444          MOV     #3$,LOCK        ;NEW SCOPE1
2985  016644  012705  177777                  MOV     #-1,R5          ;PUT "EXPECTED" IN R5
2986  016650  010561  000006          3$:     MOV     R5,6(R1)        ;WRITE ALL ONES TO THE IR
2987  016654  016104  000006                  MOV     6(R1),R4        ;READ THE IR
2988  016660  020504                          CMP     R5,R4           ;IS IT ALL ONES?
2989  016662  001401                          BEQ     4$              ;BR IF YES
2990  016664  104023                          ERROR   23              ;ERROR IR IS NOT = ALL ONES
2991  016666  104405                  4$:     SCOPE1                  ;LOOP TO 3$ IF SW09=1
2992  016670  012737  016700  001444          MOV     #5$,LOCK        ;NEW SCOPE1
2993  016676  005005                          CLR     R5              ;PUT "EXPECTED" IN R5
2994  016700  000005                  5$:     RESET                   ;BUS RESET
2995  016702  012711  003000                  MOV     #BIT9!BIT10,(R1) ;SEL6 IS IR
2996  016706  016104  000006                  MOV     6(R1),R4        ;READ THE IR
2997  016712  020504                          CMP     R5,R4           ;IS IT CLEARED?
2998  016714  001401                          BEQ     6$              ;BR IF YES
```

```
2999  016716  104023                        ERROR   23              ;ERROR, IR IS NOT CLEARED
3000  016720  104405                6$:     SCOP1                   ;LOOP TO 5$ IF SW09=1
3001
3002
3003                                        ;*************************** TEST 3 ****************************
3004                                        ;*MAINTENANCE INSTRUCTION REGISTER TEST
3005                                        ;*VERIFY THAT THE MAINT IR CAN BE WRITTEN TO ALL ZEROS'
3006                                        ;*AND ALL ONES'. VERIFY THAT IT IS CLEARED ON A MASTER RESET.
3007                                        ;:***********************************************************
3008
3009                                        ;  TEST 30
3010                                        ;-------------
3011                                        ;;***********************************************************
3012  016722  000004                TST30:  SCOPE
3013  016724  012737  000030  001202        MOV     #30,$TSTNM              ; LOAD THE NO. OF THIS TEST
3014  016732  012737  017064  001442        MOV     #TST31,NEXT             ; POINT TO THE START OF NEXT TEST.
3015  016740  012737  016756  001444        MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
3016                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
3017  016746  104410                        MSTCLR                  ;MASTER CLEAR KMC11
3018  016750  012711  003000                MOV     #BIT9!BIT10,(R1) ;SEL6 IS NOW THE IR
3019  016754  005005                        CLR     R5              ;PUT "EXPECTED" IN R5
3020  016756  010561  000006        1$:     MOV     R5,6(R1)        ;CLEAR THE IR
3021  016762  016104  000006                MOV     6(R1),R4        ;READ THE IR
3022  016766  020504                        CMP     R5,R4           ;IS IT CLEARED?
3023  016770  001401                        BEQ     2$              ;BR IF YES
3024  016772  104023                        ERROR   23              ;ERROR IR IS NOT CLEAR
3025  016774  104405                2$:     SCOP1                   ;LOOP TO 1$ IF SW09=1
3026  016776  012737  017010  001444        MOV     #3$,LOCK        ;NEW SCOP1
3027  017004  012705  177777                MOV     #-1,R5          ;PUT "EXPECTED" IN R5
3028  017010  010561  000006        3$:     MOV     R5,6(R1)        ;WRITE ALL ONES TO THE IR
3029  017014  016104  000006                MOV     6(R1),R4        ;READ THE IR
3030  017020  020504                        CMP     R5,R4           ;IS IT ALL ONES?
3031  017022  001401                        BEQ     4$              ;BR IF YES
3032  017024  104023                        ERROR   23              ;ERROR IR IS NOT = ALL ONES
3033  017026  104405                4$:     SCOP1                   ;LOOP TO 3$ IF SW09=1
3034  017030  012737  017040  001444        MOV     #5$,LOCK        ;NEW SCOP1
3035  017036  005005                        CLR     R5              ;PUT "EXPECTED" IN R5
3036  017040  052711  040000        5$:     BIS     #BIT14,(R1)     ;MASTER CLEAR
3037  017044  012711  003000                MOV     #BIT9!BIT10,(R1) ;SEL6 IS IR
3038  017050  016104  000006                MOV     6(R1),R4        ;READ THE IR
3039  017054  020504                        CMP     R5,R4           ;IS IT CLEARED?
3040  017056  001401                        BEQ     6$              ;BR IF YES
3041  017060  104023                        ERROR   23              ;ERROR, IR IS NOT CLEARED
3042  017062  104405                6$:     SCOP1                   ;LOOP TO 5$ IF SW09=1
3043
3044
3045                                        ;*************************** TEST 31 ***************************
3046                                        ;*MICRO PROCESSOR TEST
3047                                        ;*LOAD KMP06 WITH A MICRO-PROCESSOR INSTRUCTION, CLOCK IT
3048                                        ;*VERIFY INSTRUCTION EXECUTED PROPERLY
3049                                        ;*INSTRUCTION SHOULD MOVE IBUS#4 TO IBUS#5, IBUS#4 IS ALL 1'S
3050                                        ;*AND IBUS#5 IS ALL 0'S. RESULT SHOULD BE ALL 1'S IN SEL4
3051                                        ;:***********************************************************
3052
3053                                        ;  TEST 31
3054                                        ;-------------
```

```
3055                                    ;;*****************************************************************
3056  017064  000004                    TST31:  SCOPE
3057  017066  012737  000031  001202            MOV     #31,$TSTNM              ; LOAD THE NO. OF THIS TEST
3058  017074  012737  017150  001442            MOV     #TST32,NEXT             ; POINT TO THE START OF NEXT TEST.
3059                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
3060  017102  104410                            MSTCLR                  ;MASTER CLEAR KMC11
3061  017104  012761  000377  000004           MOV     #377,4(R1)      ;PORT4 HI-BYTE=0'S LO-BYTE=1'S
3062  017112  012711  001000                   MOV     #BIT9,(R1)      ;SET ROMI
3063  017116  012761  121105  000006           MOV     #121105,6(R1)   ;INSTRUCTION TO PORT6
3064  017124  052711  001400                   BIS     #BIT8!BIT9,(R1) ;CLK INTSRUCTION, MOVE IBUS*4 TO IBUS*5
3065  017130  000240                            NOP                     ;
3066  017132  012705  177777                   MOV     #-1,R5          ;PUT "EXPECTED" IN R5
3067  017136  016104  000004                   MOV     4(R1),R4        ;PUT "FOUND" INTO R4
3068  017142  020504                            CMP     R5,R4           ;IS DATA CORRECT
3069  017144  001401                            BEQ     1$              ;BR IF YES
3070  017146  104003                            ERROR   3               ;ERROR
3071  017150                          1$:
3072
3073
3074                                            ;*************************** TEST 32 ****************************
3075                                            ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
3076                                            ;*FLOAT A 1 THROUGH IBUS* REGISTER 0
3077                                            ;*FLOAT A 0 THROUGH IBUS* REGISTER 0
3078                                            ;;*************************************************************
3079
3080                                            ;  TEST 32
3081                                            ;-------------
3082                                    ;;*****************************************************************
3083  017150  000004                    TST32:  SCOPE
3084  017152  012737  000032  001202            MOV     #32,$TSTNM              ; LOAD THE NO. OF THIS TEST
3085  017160  012737  017350  001442            MOV     #TST33,NEXT             ; POINT TO THE START OF NEXT TEST.
3086  017166  012737  017206  001444            MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3087                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
3088  017174  104410                            MSTCLR                  ;MASTER CLEAR KMC11
3089  017176  012702  000000                   MOV     #0,R2           ;SAVE REGISTER ADDRESS FOR TYPEOUT
3090  017202  012700  000001                   MOV     #1,R0           ;START WITH BIT 0
3091  017206                          64$:
3092  017206  010061  000004                   MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3093  017212  042761  000030  000004           BIC     #30,4(R1)       ;CLEAR UNWANTED BITS
3094  017220  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3095  017222  121100                            121100!0                ;MOV DATA TO IBUS* REGISTER 0
3096  017224  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3097  017226  121005                            121005!<0*20>           ;READ FROM IBUS* REGISTER 0
3098  017230  010005                            MOV     R0,R5           ;PUT EXPECTED IN R5
3099  017232  042705  000030                   BIC     #30,R5          ;CLEAR UNWANTED BITS
3100  017236  116104  000005                   MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3101  017242  120504                            CMPB    R5,R4           ;DATA CORRECT?
3102  017244  001401                            BEQ     65$             ;BR IF YES
3103  017246  104004                            ERROR   4               ;ERROR
3104  017250  104405                  65$:      SCOP1                   ;SW09=1?
3105  017252  000241                            CLC                     ;CLEAR CARRY
3106  017254  106100                            ROLB    R0              ;SHIFT BIT IN R0
3107  017256  001353                            BNE     64$             ;IF R0=0 THEN DONE
3108  017260  012737  017274  001444            MOV     #67$,LOCK       ;NEW SCOP1
3109  017266  012700  000001                   MOV     #1,R0           ;START WITH BIT 0
3110  017272  005100                  69$:      COM     R0              ;CHANGE TO FLOATING ZERO
```

```
3111  017274                          67$:
3112  017274  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3113  017300  042761  000030  000004          BIC     #30,4(R1)       ;CLEAR UNWANTED BITS
3114  017306  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3115  017310  121100                          121100!0                ;MOV DATA TO IBUS* REGISTER 0
3116  017312  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3117  017314  121005                          121005!<0*20>           ;READ FROM IBUS* REGISTER 0
3118  017316  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
3119  017320  042705  000030                  BIC     #30,R5          ;CLEAR UNWANTED BITS
3120  017324  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3121  017330  120504                          CMPB    R5,R4           ;DATA CORRECT?
3122  017332  001401                          BEQ     68$             ;BR IF YES
3123  017334  104004                          ERROR   4               ;ERROR
3124  017336  104405                  68$:    SCOPE1                  ;SW09=1?
3125  017340  005100                          COM     R0              ;CHANGE TO FLOATING 1
3126  017342  000241                          CLC                     ;CLEAR CARRY
3127  017344  106100                          ROLB    R0              ;SHIFT BIT IN R0
3128  017346  001351                          BNE     69$             ;IF R0=0 THEN DONE
3129
3130
3131                                          ;************************** TEST 33 ***************************
3132                                          ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
3133                                          ;*FLOAT A 1 THROUGH IBUS* REGISTER 2
3134                                          ;*FLOAT A 0 THROUGH IBUS* REGISTER 2
3135                                          ;:**********************************************************
3136
3137                                          ;  TEST 33
3138                                          ;--------------
3139                                  ;;**********************************************************
3140  017350  000004                  TST33:  SCOPE
3141  017352  012737  000033  001202          MOV     #33,$TSTNM              ; LOAD THE NO. OF THIS TEST
3142  017360  012737  017550  001442          MOV     #TST34,NEXT             ; POINT TO THE START OF NEXT TEST.
3143  017366  012737  017406  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3144                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3145  017374  104410                          MSTCLR                  ;MASTER CLEAR KMC11
3146  017376  012702  000002                  MOV     #2,R2           ;SAVE REGISTER ADDRESS FOR TYPEOUT
3147  017402  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
3148  017406                          64$:
3149  017406  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3150  017412  042761  000070  000004          BIC     #70,4(R1)       ;CLEAR UNWANTED BITS
3151  017420  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3152  017422  121102                          121100!2                ;MOV DATA TO IBUS* REGISTER 2
3153  017424  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3154  017426  121045                          121005!<2*20>           ;READ FROM IBUS* REGISTER 2
3155  017430  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
3156  017432  042705  000070                  BIC     #70,R5          ;CLEAR UNWANTED BITS
3157  017436  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3158  017442  120504                          CMPB    R5,R4           ;DATA CORRECT?
3159  017444  001401                          BEQ     65$             ;BR IF YES
3160  017446  104004                          ERROR   4               ;ERROR
3161  017450  104405                  65$:    SCOPE1                  ;SW09=1?
3162  017452  000241                          CLC                     ;CLEAR CARRY
3163  017454  106100                          ROLB    R0              ;SHIFT BIT IN R0
3164  017456  001353                          BNE     64$             ;IF R0=0 THEN DONE
3165  017460  012737  017474  001444          MOV     #67$,LOCK       ;NEW SCOP1
3166  017466  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
```

```
3167  017472  005100                    69$:    COM     R0              ;CHANGE TO FLOATING ZERO
3168  017474                            67$:
3169  017474  010061  000004                    MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3170  017500  042761  000070  000004            BIC     #70,4(R1)       ;CLEAR UNWANTED BITS
3171  017506  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3172  017510  121102                            121100!2                ;MOV DATA TO IBUS* REGISTER 2
3173  017512  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3174  017514  121045                            121005!<2*20>           ;READ FROM IBUS* REGISTER 2
3175  017516  010005                            MOV     R0,R5           ;PUT EXPECTED IN R5
3176  017520  042705  000070                    BIC     #70,R5          ;CLEAR UNWANTED BITS
3177  017524  116104  000005                    MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3178  017530  120504                            CMPB    R5,R4           ;DATA CORRECT?
3179  017532  001401                            BEQ     68$             ;BR IF YES
3180  017534  104004                            ERROR   4               ;ERROR
3181  017536  104405                    68$:    SCOP1                   ;SW09=1?
3182  017540  005100                            COM     R0              ;CHANGE TO FLOATING 1
3183  017542  000241                            CLC                     ;CLEAR CARRY
3184  017544  106100                            ROLB    R0              ;SHIFT BIT IN R0
3185  017546  001351                            BNE     69$             ;IF R0=0 THEN DONE
3186
3187
3188                                            ;************************** TEST 34 **************************
3189                                            ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
3190                                            ;*FLOAT A 1 THROUGH IBUS* REGISTER 4
3191                                            ;*FLOAT A 0 THROUGH IBUS* REGISTER 4
3192                                            ;;*************************************************************
3193
3194                                            ;  TEST 34
3195                                            ;-------------
3196                                            ;;*************************************************************
3197  017550  000004                    TST34:  SCOPE
3198  017552  012737  000034  001202            MOV     #34,$TSTNM              ; LOAD THE NO. OF THIS TEST
3199  017560  012737  017724  001442            MOV     #TST35,NEXT             ; POINT TO THE START OF NEXT TEST.
3200  017566  012737  017606  001444            MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3201                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
3202  017574  104410                            MSTCLR                  ;MASTER CLEAR KMC11
3203  017576  012702  000004                    MOV     #4,R2           ;SAVE REGISTER ADDRESS FOR TYPEOUT
3204  017602  012700  000001                    MOV     #1,R0           ;START WITH BIT 0
3205  017606                            64$:
3206  017606  010061  000004                    MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3207  017612  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3208  017614  121104                            121100!4                ;MOV DATA TO IBUS* REGISTER 4
3209  017616  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3210  017620  121105                            121005!<4*20>           ;READ FROM IBUS* REGISTER 4
3211  017622  010005                            MOV     R0,R5           ;PUT EXPECTED IN R5
3212  017624  116104  000005                    MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3213  017630  120504                            CMPB    R5,R4           ;DATA CORRECT?
3214  017632  001401                            BEQ     65$             ;BR IF YES
3215  017634  104004                            ERROR   4               ;ERROR
3216  017636  104405                    65$:    SCOP1                   ;SW09=1?
3217  017640  000241                            CLC                     ;CLEAR CARRY
3218  017642  106100                            ROLB    R0              ;SHIFT BIT IN R0
3219  017644  001360                            BNE     64$             ;IF R0=0 THEN DONE
3220  017646  012737  017662  001444            MOV     #67$,LOCK       ;NEW SCOP1
3221  017654  012700  000001                    MOV     #1,R0           ;START WITH BIT 0
3222  017660  005100                    69$:    COM     R0              ;CHANGE TO FLOATING ZERO
```

```
3223  017662                          67$:
3224  017662  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3225  017666  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3226  017670  121104                          121100!4                ;MOV DATA TO IBUS* REGISTER 4
3227  017672  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3228  017674  121105                          121005!(4*20)           ;READ FROM IBUS* REGISTER 4
3229  017676  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
3230  017700  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3231  017704  120504                          CMPB    R5,R4           ;DATA CORRECT?
3232  017706  001401                          BEQ     68$             ;BR IF YES
3233  017710  104004                          ERROR   4               ;ERROR
3234  017712  104405                  68$:    SCOP1                   ;SW09=1?
3235  017714  005100                          COM     R0              ;CHANGE TO FLOATING 1
3236  017716  000241                          CLC                     ;CLEAR CARRY
3237  017720  106100                          ROLB    R0              ;SHIFT BIT IN R0
3238  017722  001356                          BNE     69$             ;IF R0=0 THEN DONE
3239
3240
3241                                          ;*************************** TEST 35 ****************************
3242                                          ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
3243                                          ;*FLOAT A 1 THROUGH IBUS* REGISTER 5
3244                                          ;*FLOAT A 0 THROUGH IBUS* REGISTER 5
3245                                          ;;*************************************************************
3246
3247                                          ;  TEST 35
3248                                          ;--------------
3249                                          ;;*************************************************************
3250  017724  000004                  TST35:  SCOPE
3251  017726  012737  000035  001202          MOV     #35,$TSTNM              ; LOAD THE NO. OF THIS TEST
3252  017734  012737  020100  001442          MOV     #TST36,NEXT             ; POINT TO THE START OF NEXT TEST.
3253  017742  012737  017762  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3254                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3255  017750  104410                          MSTCLR                  ;MASTER CLEAR KMC11
3256  017752  012702  000005                  MOV     #5,R2           ;SAVE REGISTER ADDRESS FOR TYPEOUT
3257  017756  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
3258  017762                          64$:
3259  017762  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3260  017766  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3261  017770  121105                          121100!5                ;MOV DATA TO IBUS* REGISTER 5
3262  017772  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3263  017774  121125                          121005!(5*20)           ;READ FROM IBUS* REGISTER 5
3264  017776  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
3265  020000  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3266  020004  120504                          CMPB    R5,R4           ;DATA CORRECT?
3267  020006  001401                          BEQ     65$             ;BR IF YES
3268  020010  104004                          ERROR   4               ;ERROR
3269  020012  104405                  65$:    SCOP1                   ;SW09=1?
3270  020014  000241                          CLC                     ;CLEAR CARRY
3271  020016  106100                          ROLB    R0              ;SHIFT BIT IN R0
3272  020020  001360                          BNE     64$             ;IF R0=0 THEN DONE
3273  020022  012737  020036  001444          MOV     #67$,LOCK       ;NEW SCOP1
3274  020030  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
3275  020034  005100                  69$:    COM     R0              ;CHANGE TO FLOATING ZERO
3276  020036                          67$:
3277  020036  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3278  020042  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
3279  020044  121105                        121100!5                ;MOV DATA TO IBUS* REGISTER 5
3280  020046  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3281  020050  121125                        121005!<5*20>           ;READ FROM IBUS* REGISTER 5
3282  020052  010005                        MOV     R0,R5           ;PUT EXPECTED IN R5
3283  020054  116104  000005                MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3284  020060  120504                        CMPB    R5,R4           ;DATA CORRECT?
3285  020062  001401                        BEQ     68$             ;BR IF YES
3286  020064  104004                        ERROR   4               ;ERROR
3287  020066  104405                68$:    SCOP1                   ;SW09=1?
3288  020070  005100                        COM     R0              ;CHANGE TO FLOATING 1
3289  020072  000241                        CLC                     ;CLEAR CARRY
3290  020074  106100                        ROLB    R0              ;SHIFT BIT IN R0
3291  020076  001356                        BNE     69$             ;IF R0=0 THEN DONE
3292
3293
3294                                        ;************************** TEST 36 **************************
3295                                        ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
3296                                        ;*FLOAT A 1 THROUGH IBUS* REGISTER 10
3297                                        ;*FLOAT A 0 THROUGH IBUS* REGISTER 10
3298                                        ;:************************************************************
3299
3300                                        ;  TEST 36
3301                                        ;----------
3302                                   ;;***********************************************************
3303  020100  000004                TST36:  SCOPE
3304  020102  012737  000036  001202        MOV     #36,$TSTNM              ; LOAD THE NO. OF THIS TEST
3305  020110  012737  020300  001442        MOV     #TST37,NEXT             ; POINT TO THE START OF NEXT TEST.
3306  020116  012737  020136  001444        MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3307                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3308  020124  104410                        MSTCLR                  ;MASTER CLEAR KMC11
3309  020126  012702  OC2010                MOV     #10,R2          ;SAVE REGISTER ADDRESS FOR TYPEOUT
3310  020132  012700  000001                MOV     #1,R0           ;START WITH BIT 0
3311  020136                         64$:
3312  020136  010061  000004                MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3313  020142  042761  000141  000004        BIC     #141,4(R1)      ;CLEAR UNWANTED BITS
3314  020150  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3315  020152  121110                        121100!10               ;MOV DATA TO IBUS* REGISTER 10
3316  020154  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3317  020156  121205                        121005!<10*20>          ;READ FROM IBUS* REGISTER 10
3318  020160  010005                        MOV     R0,R5           ;PUT EXPECTED IN R5
3319  020162  042705  000141                BIC     #141,R5         ;CLEAR UNWANTED BITS
3320  020166  116104  000005                MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3321  020172  120504                        CMPB    R5,R4           ;DATA CORRECT?
3322  020174  001401                        BEQ     65$             ;BR IF YES
3323  020176  104004                        ERROR   4               ;ERROR
3324  020200  104405                65$:    SCOP1                   ;SW09=1?
3325  020202  000241                        CLC                     ;CLEAR CARRY
3326  020204  106100                        ROLB    R0              ;SHIFT BIT IN R0
3327  020206  001353                        BNE     64$             ;IF R0=0 THEN DONE
3328  020210  012737  020224  001444        MOV     #67$,LOCK       ;NEW SCOP1
3329  020216  012700  000001                MOV     #1,R0           ;START WITH BIT 0
3330  020222  005100                 69$:    COM     R0              ;CHANGE TO FLOATING ZERO
3331  020224                         67$:
3332  020224  010061  000004                MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3333  020230  042761  000141  000004        BIC     #141,4(R1)      ;CLEAR UNWANTED BITS
3334  020236  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
3335  020240  121110                        121100!10               ;MOV DATA TO IBUS* REGISTER 10
3336  020242  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3337  020244  121205                        121005!<10*20>          ;READ FROM IBUS* REGISTER 10
3338  020246  010005                        MOV     R0,R5           ;PUT EXPECTED IN R5
3339  020250  042705  000141                BIC     #141,R5         ;CLEAR UNWANTED BITS
3340  020254  116104  000005                MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3341  020260  120504                        CMPB    R5,R4           ;DATA CORRECT?
3342  020262  001401                        BEQ     68$             ;BR IF YES
3343  020264  104004                        ERROR   4               ;ERROR
3344  020266  104405                68$:    SCOP1                   ;SW09=1?
3345  020270  005100                        COM     R0              ;CHANGE TO FLOATING 1
3346  020272  000241                        CLC                     ;CLEAR CARRY
3347  020274  106100                        ROLB    R0              ;SHIFT BIT IN R0
3348  020276  001351                        BNE     69$             ;IF R0=0 THEN DONE
3349
3350
3351                                        ;************************** TEST 37 **************************
3352                                        ;*MICRO PROCESSOR IBUS* REGISTER WRITE/READ TEST
3353                                        ;*FLOAT A 1 THROUGH IBUS* REGISTER 11
3354                                        ;*FLOAT A 0 THROUGH IBUS* REGISTER 11
3355                                        ;:**********************************************************
3356
3357                                        ;  TEST 37
3358                                        ;-------------
3359                                   ;:***********************************************************
3360  020300  000004                TST37:  SCOPE
3361  020302  012737  000037  001202        MOV     #37,$TSTNM              ; LOAD THE NO. OF THIS TEST
3362  020310  012737  020510  001442        MOV     #TST40,NEXT             ; POINT TO THE START OF NEXT TEST.
3363  020316  012737  020336  001444        MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3364                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
3365  020324  104410                        MSTCLR                  ;MASTER CLEAR KMC11
3366  020326  012702  000011                MOV     #11,R2          ;SAVE REGISTER ADDRESS FOR TYPEOUT
3367  020332  012700  000001                MOV     #1,R0           ;START WITH BIT 0
3368  020336                         64$:
3369  020336  010061  000004                MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3370  020342  042761  000262  000004        BIC     #262,4(R1)      ;CLEAR UNWANTED BITS
3371  020350  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3372  020352  121111                        121100!11               ;MOV DATA TO IBUS* REGISTER 11
3373  020354  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3374  020356  121225                        121005!<11*20>          ;READ FROM IBUS* REGISTER 11
3375  020360  010005                        MOV     R0,R5           ;PUT EXPECTED IN R5
3376  020362  042705  000262                BIC     #262,R5         ;CLEAR UNWANTED BITS
3377  020366  052705  000020                BIS     #20,R5          ;ADD THESE BITS
3378  020372  116104  000005                MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3379  020376  120504                        CMPB    R5,R4           ;DATA CORRECT?
3380  020400  001401                        BEQ     65$             ;BR IF YES
3381  020402  104004                        ERROR   4               ;ERROR
3382  020404  104405                65$:    SCOP1                   ;SW09=1?
3383  020406  000241                        CLC                     ;CLEAR CARRY
3384  020410  106100                        ROLB    R0              ;SHIFT BIT IN R0
3385  020412  001351                        BNE     64$             ;IF R0=0 THEN DONE
3386  020414  012737  020430  001444        MOV     #67$,LOCK       ;NEW SCOP1
3387  020422  012700  000001                MOV     #1,R0           ;START WITH BIT 0
3388  020426  005100                 69$:    COM     R0              ;CHANGE TO FLOATING ZERO
3389  020430                         67$:
3390  020430  010061  000004                MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
```

```
3391  020434  042761  000262  000004          BIC     #262,4(R1)      ;CLEAR UNWANTED BITS
3392  020442  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3393  020444  121111                          121100!11               ;MOV DATA TO IBUS* REGISTER 11
3394  020446  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3395  020450  121225                          121005!(11*20)          ;READ FROM IBUS* REGISTER 11
3396  020452  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
3397  020454  042705  000262                  BIC     #262,R5         ;CLEAR UNWANTED BITS
3398  020460  052705  000020                  BIS     #20,R5          ;ADD THESE BITS
3399  020464  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3400  020470  120504                          CMPB    R5,R4           ;DATA CORRECT?
3401  020472  001401                          BEQ     68$             ;BR IF YES
3402  020474  104004                          ERROR   4               ;ERROR
3403  020476  104405                  68$:    SCOP1                   ;SW09=1?
3404  020500  005100                          COM     R0              ;CHANGE TO FLOATING 1
3405  020502  000241                          CLC                     ;CLEAR CARRY
3406  020504  106100                          ROLB    R0              ;SHIFT BIT IN R0
3407  020506  001347                          BNE     69$             ;IF R0=0 THEN DONE
3408
3409
3410                                          ;************************** TEST 40 **************************
3411                                          ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
3412                                          ;*FLOAT A 1 THROUGH IBUS REGISTER 0
3413                                          ;*FLOAT A 0 THROUGH IBUS REGISTER 0
3414                                          ;;*********************************************************************
3415
3416                                          ;  TEST 40
3417                                          ;-------------
3418                                  ;;*********************************************************************
3419  020510  000004                  TST40:  SCOPE
3420  020512  012737  000040  001202          MOV     #40,$TSTNM              ; LOAD THE NO. OF THIS TEST
3421  020520  012737  020664  001442          MOV     #TST41,NEXT             ; POINT TO THE START OF NEXT TEST.
3422  020526  012737  020546  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3423                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3424  020534  104410                          MSTCLR                  ;MASTER CLEAR KMC11
3425  020536  012702  000000                  MOV     #0,R2           ;SAVE REGISTER ADDRESS FOR TYPEOUT
3426  020542  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
3427  020546                          64$:
3428  020546  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3429  020552  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3430  020554  122100                          122100!0                ;MOV DATA TO IBUS REGISTER 0
3431  020556  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3432  020560  021005                          21005!(0*20)            ;READ FROM IBUS REGISTER 0
3433  020562  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
3434  020564  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3435  020570  120504                          CMPB    R5,R4           ;DATA CORRECT?
3436  020572  001401                          BEQ     65$             ;BR IF YES
3437  020574  104005                          ERROR   5               ;ERROR
3438  020576  104405                  65$:    SCOP1                   ;SW09=1?
3439  020600  000241                          CLC                     ;CLEAR CARRY
3440  020602  106100                          ROLB    R0              ;SHIFT BIT IN R0
3441  020604  001360                          BNE     64$             ;IF R0=0 THEN DONE
3442  020606  012737  020622  001444          MOV     #67$,LOCK       ;NEW SCOP1
3443  020614  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
3444  020620  005100                  69$:    COM     R0              ;CHANGE TO FLOATING ZERO
3445  020622                          67$:
3446  020622  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
```

```
3447  020626  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3448  020630  122100                          122100!0                ;MOV DATA TO IBUS REGISTER 0
3449  020632  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3450  020634  021005                          21005!<0*20>            ;READ FROM IBUS REGISTER 0
3451  020636  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
3452  020640  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3453  020644  120504                          CMPB    R5,R4           ;DATA CORRECT?
3454  020646  001401                          BEQ     68$             ;BR IF YES
3455  020650  104005                          ERROR   5               ;ERROR
3456  020652  104405                  68$:    SCOP1                   ;SW09=1?
3457  020654  005100                          COM     R0              ;CHANGE TO FLOATING 1
3458  020656  000241                          CLC                     ;CLEAR CARRY
3459  020660  106100                          ROLB    R0              ;SHIFT BIT IN R0
3460  020662  001356                          BNE     69$             ;IF R0=0 THEN DONE
3461
3462
3463                                          ;************************** TEST 41 **************************
3464                                          ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
3465                                          ;*FLOAT A 1 THROUGH IBUS REGISTER 1
3466                                          ;*FLOAT A 0 THROUGH IBUS REGISTER 1
3467                                          ;:**********************************************************************
3468
3469                                          ;  TEST 41
3470                                          ;--------------
3471                                  ;;******************************************************************
3472  020664  000004                  TST41:  SCOPE
3473  020666  012737  000041  001202          MOV     #41,$TSTNM              ; LOAD THE NO. OF THIS TEST
3474  020674  012737  021040  001442          MOV     #TST42,NEXT             ; POINT TO THE START OF NEXT TEST.
3475  020702  012737  020722  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3476                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3477  020710  104410                          MSTCLR                  ;MASTER CLEAR KMC11
3478  020712  012702  000001                  MOV     #1,R2           ;SAVE REGISTER ADDRESS FOR TYPEOUT
3479  020716  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
3480  020722                          64$:
3481  020722  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3482  020726  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3483  020730  122101                          122100!1                ;MOV DATA TO IBUS REGISTER 1
3484  020732  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3485  020734  021025                          21005!<1*20>            ;READ FROM IBUS REGISTER 1
3486  020736  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
3487  020740  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3488  020744  120504                          CMPB    R5,R4           ;DATA CORRECT?
3489  020746  001401                          BEQ     65$             ;BR IF YES
3490  020750  104005                          ERROR   5               ;ERROR
3491  020752  104405                  65$:    SCOP1                   ;SW09=1?
3492  020754  000241                          CLC                     ;CLEAR CARRY
3493  020756  106100                          ROLB    R0              ;SHIFT BIT IN R0
3494  020760  001360                          BNE     64$             ;IF R0=0 THEN DONE
3495  020762  012737  020776  001444          MOV     #67$,LOCK       ;NEW SCOP1
3496  020770  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
3497  020774  005100                  69$:    COM     R0              ;CHANGE TO FLOATING ZERO
3498  020776                          67$:
3499  020776  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3500  021002  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3501  021004  122101                          122100!1                ;MOV DATA TO IBUS REGISTER 1
3502  021006  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
3503  021010  021025                        21005!(1*20)            ;READ FROM IBUS REGISTER 1
3504  021012  010005                        MOV     R0,R5           ;PUT EXPECTED IN R5
3505  021014  116104  000005                MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3506  021020  120504                        CMPB    R5,R4           ;DATA CORRECT?
3507  021022  001401                        BEQ     68$             ;BR IF YES
3508  021024  104005                        ERROR   5               ;ERROR
3509  021026  104405              68$:      SCOP1                   ;SW09=1?
3510  021030  005100                        COM     R0              ;CHANGE TO FLOATING 1
3511  021032  000241                        CLC                     ;CLEAR CARRY
3512  021034  106100                        ROLB    R0              ;SHIFT BIT IN R0
3513  021036  001356                        BNE     69$             ;IF R0=0 THEN DONE
3514
3515
3516                                        ;************************** TEST 42 **************************
3517                                        ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
3518                                        ;*FLOAT A 1 THROUGH IBUS REGISTER 2
3519                                        ;*FLOAT A 0 THROUGH IBUS REGISTER 2
3520                                        ;:**********************************************************
3521
3522                                        ;  TEST 42
3523                                        ;-------------
3524                                    ;;:**********************************************************
3525  021040  000004              TST42:    SCOPE
3526  021042  012737  000042  001202        MOV     #42,$TSTNM              ; LOAD THE NO. OF THIS TEST
3527  021050  012737  021214  001442        MOV     #TST43,NEXT             ; POINT TO THE START OF NEXT TEST.
3528  021056  012737  021076  001444        MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3529                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
3530  021064  104410                        MSTCLR                  ;MASTER CLEAR KMC11
3531  021066  012702  000002                MOV     #2,R2           ;SAVE REGISTER ADDRESS FOR TYPEOUT
3532  021072  012700  000001                MOV     #1,R0           ;START WITH BIT 0
3533  021076                        64$:
3534  021076  010061  000004                MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3535  021102  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3536  021104  122102                        122100!2                ;MOV DATA TO IBUS REGISTER 2
3537  021106  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3538  021110  021045                        21005!(2*20)            ;READ FROM IBUS REGISTER 2
3539  021112  010005                        MOV     R0,R5           ;PUT EXPECTED IN R5
3540  021114  116104  000005                MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3541  021120  120504                        CMPB    R5,R4           ;DATA CORRECT?
3542  021122  001401                        BEQ     65$             ;BR IF YES
3543  021124  104005                        ERROR   5               ;ERROR
3544  021126  104405              65$:      SCOP1                   ;SW09=1?
3545  021130  000241                        CLC                     ;CLEAR CARRY
3546  021132  106100                        ROLB    R0              ;SHIFT BIT IN R0
3547  021134  001360                        BNE     64$             ;IF R0=0 THEN DONE
3548  021136  012737  021152  001444        MOV     #67$,LOCK       ;NEW SCOP1
3549  021144  012700  000001                MOV     #1,R0           ;START WITH BIT 0
3550  021150  005100                69$:    COM     R0              ;CHANGE TO FLOATING ZERO
3551  021152                        67$:
3552  021152  010061  000004                MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3553  021156  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3554  021160  122102                        122100!2                ;MOV DATA TO IBUS REGISTER 2
3555  021162  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3556  021164  021045                        21005!(2*20)            ;READ FROM IBUS REGISTER 2
3557  021166  010005                        MOV     R0,R5           ;PUT EXPECTED IN R5
3558  021170  116104  000005                MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
```

```
3559  021174  120504                          CMPB    R5,R4           ;DATA CORRECT?
3560  021176  001401                          BEQ     68$             ;BR IF YES
3561  021200  104005                          ERROR   5               ;ERROR
3562  021202  104405                  68$:    SCOP1                   ;SW09=1?
3563  021204  005100                          COM     R0              ;CHANGE TO FLOATING 1
3564  021206  000241                          CLC                     ;CLEAR CARRY
3565  021210  106100                          ROLB    R0              ;SHIFT BIT IN R0
3566  021212  001356                          BNE     69$             ;IF R0=0 THEN DONE
3567
3568
3569                                          ;*************************** TEST 43 ****************************
3570                                          ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
3571                                          ;*FLOAT A 1 THROUGH IBUS REGISTER 3
3572                                          ;*FLOAT A 0 THROUGH IBUS REGISTER 3
3573                                          ;:************************************************************
3574
3575                                          ;  TEST 43
3576                                          ;--------------
3577                                  ;;******************************************************************
3578  021214  000004                  TST43:  SCOPE
3579  021216  012737  000043  001202          MOV     #43,$TSTNM              ; LOAD THE NO. OF THIS TEST
3580  021224  012737  021370  001442          MOV     #TST44,NEXT             ; POINT TO THE START OF NEXT TEST.
3581  021232  012737  021252  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3582                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3583  021240  104410                          MSTCLR                  ;MASTER CLEAR KMC11
3584  021242  012702  000003                  MOV     #3,R2           ;SAVE REGISTER ADDRESS FOR TYPEOUT
3585  021246  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
3586  021252                          64$:
3587  021252  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3588  021256  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3589  021260  122103                          122100!3                ;MOV DATA TO IBUS REGISTER 3
3590  021262  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3591  021264  021065                          21005!<3*20>            ;READ FROM IBUS REGISTER 3
3592  021266  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
3593  021270  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3594  021274  120504                          CMPB    R5,R4           ;DATA CORRECT?
3595  021276  001401                          BEQ     65$             ;BR IF YES
3596  021300  104005                          ERROR   5               ;ERROR
3597  021302  104405                  65$:    SCOP1                   ;SW09=1?
3598  021304  000241                          CLC                     ;CLEAR CARRY
3599  021306  106100                          ROLB    R0              ;SHIFT BIT IN R0
3600  021310  001360                          BNE     64$             ;IF R0=0 THEN DONE
3601  021312  012737  021326  001444          MOV     #67$,LOCK       ;NEW SCOP1
3602  021320  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
3603  021324  005100                  69$:    COM     R0              ;CHANGE TO FLOATING ZERO
3604  021326                          67$:
3605  021326  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3606  021332  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3607  021334  122103                          122100!3                ;MOV DATA TO IBUS REGISTER 3
3608  021336  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3609  021340  021065                          21005!<3*20>            ;READ FROM IBUS REGISTER 3
3610  021342  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
3611  021344  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3612  021350  120504                          CMPB    R5,R4           ;DATA CORRECT?
3613  021352  001401                          BEQ     68$             ;BR IF YES
3614  021354  104005                          ERROR   5               ;ERROR
```

```
3615  021356  104405                  68$:    SCOP1                   ;SW09=1?
3616  021360  005100                          COM     R0              ;CHANGE TO FLOATING 1
3617  021362  000241                          CLC                     ;CLEAR CARRY
3618  021364  106100                          ROLB    R0              ;SHIFT BIT IN R0
3619  021366  001356                          BNE     69$             ;IF R0=0 THEN DONE
3620
3621
3622                                          ;*************************** TEST 44 ***************************
3623                                          ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
3624                                          ;*FLOAT A 1 THROUGH IBUS REGISTER 4
3625                                          ;*FLOAT A 0 THROUGH IBUS REGISTER 4
3626                                          ;;*********************************************************************
3627
3628                                          ;  TEST 44
3629                                          ;--------------
3630                                          ;;*********************************************************************
3631  021370  000004                  TST44:  SCOPE
3632  021372  012737  000044  001202          MOV     #44,$TSTNM              ; LOAD THE NO. OF THIS TEST
3633  021400  012737  021544  001442          MOV     #TST45,NEXT             ; POINT TO THE START OF NEXT TEST.
3634  021406  012737  021426  001444          MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3635                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3636  021414  104410                          MSTCLR                  ;MASTER CLEAR KMC11
3637  021416  012702  000004                  MOV     #4,R2           ;SAVE REGISTER ADDRESS FOR TYPEOUT
3638  021422  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
3639  021426                          64$:
3640  021426  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3641  021432  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3642  021434  122104                          122100!4                ;MOV DATA TO IBUS REGISTER 4
3643  021436  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3644  021440  021105                          21005!<4*20>            ;READ FROM IBUS REGISTER 4
3645  021442  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
3646  021444  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3647  021450  120504                          CMPB    R5,R4           ;DATA CORRECT?
3648  021452  001401                          BEQ     65$             ;BR IF YES
3649  021454  104005                          ERROR   5               ;ERROR
3650  021456  104405                  65$:    SCOP1                   ;SW09=1?
3651  021460  000241                          CLC                     ;CLEAR CARRY
3652  021462  106100                          ROLB    R0              ;SHIFT BIT IN R0
3653  021464  001360                          BNE     64$             ;IF R0=0 THEN DONE
3654  021466  012737  021502  001444          MOV     #67$,LOCK       ;NEW SCOP1
3655  021474  012700  000001                  MOV     #1,R0           ;START WITH BIT 0
3656  021500  005100                  69$:    COM     R0              ;CHANGE TO FLOATING ZERO
3657  021502                          67$:
3658  021502  010061  000004                  MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3659  021506  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3660  021510  122104                          122100!4                ;MOV DATA TO IBUS REGISTER 4
3661  021512  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3662  021514  021105                          21005!<4*20>            ;READ FROM IBUS REGISTER 4
3663  021516  010005                          MOV     R0,R5           ;PUT EXPECTED IN R5
3664  021520  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3665  021524  120504                          CMPB    R5,R4           ;DATA CORRECT?
3666  021526  001401                          BEQ     68$             ;BR IF YES
3667  021530  104005                          ERROR   5               ;ERROR
3668  021532  104405                  68$:    SCOP1                   ;SW09=1?
3669  021534  005100                          COM     R0              ;CHANGE TO FLOATING 1
3670  021536  000241                          CLC                     ;CLEAR CARRY
```

```
3671  021540  106100                  ROLB    R0              ;SHIFT BIT IN R0
3672  021542  001356                  BNE     69$             ;IF R0=0 THEN DONE
3673
3674
3675                                  ;*************************** TEST 45 ****************************
3676                                  ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
3677                                  ;*FLOAT A 1 THROUGH IBUS REGISTER 5
3678                                  ;*FLOAT A 0 THROUGH IBUS REGISTER 5
3679                                  ;:*****************************************************************
3680
3681                                  ;  TEST 45
3682                                  ;-------------
3683                                  ;;*****************************************************************
3684  021544  000004          TST45:  SCOPE
3685  021546  012737  000045  001202  MOV     #45,$TSTNM              ; LOAD THE NO. OF THIS TEST
3686  021554  012737  021720  001442  MOV     #TST46,NEXT             ; POINT TO THE START OF NEXT TEST.
3687  021562  012737  021602  001444  MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3688                                                          ;R1 CONTAINS BASE KMC11 ADDRESS
3689  021570  104410                  MSTCLR                  ;MASTER CLEAR KMC11
3690  021572  012702  000005          MOV     #5,R2           ;SAVE REGISTER ADDRESS FOR TYPEOUT
3691  021576  012700  000001          MOV     #1,R0           ;START WITH BIT 0
3692  021602                  64$:
3693  021602  010061  000004          MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3694  021606  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3695  021610  122105                  122100!5                ;MOV DATA TO IBUS REGISTER 5
3696  021612  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3697  021614  021125                  21005!(5*20)            ;READ FROM IBUS REGISTER 5
3698  021616  010005                  MOV     R0,R5           ;PUT EXPECTED IN R5
3699  021620  116104  000005          MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3700  021624  120504                  CMPB    R5,R4           ;DATA CORRECT?
3701  021626  001401                  BEQ     65$             ;BR IF YES
3702  021630  104005                  ERROR   5               ;ERROR
3703  021632  104405          65$:    SCOP1                   ;SW09=1?
3704  021634  000241                  CLC                     ;CLEAR CARRY
3705  021636  106100                  ROLB    R0              ;SHIFT BIT IN R0
3706  021640  001360                  BNE     64$             ;IF R0=0 THEN DONE
3707  021642  012737  021656  001444  MOV     #67$,LOCK       ;NEW SCOP1
3708  021650  012700  000001          MOV     #1,R0           ;START WITH BIT 0
3709  021654  005100          69$:    COM     R0              ;CHANGE TO FLOATING ZERO
3710  021656                  67$:
3711  021656  010061  000004          MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3712  021662  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3713  021664  122105                  122100!5                ;MOV DATA TO IBUS REGISTER 5
3714  021666  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3715  021670  021125                  21005!(5*20)            ;READ FROM IBUS REGISTER 5
3716  021672  010005                  MOV     R0,R5           ;PUT EXPECTED IN R5
3717  021674  116104  000005          MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3718  021700  120504                  CMPB    R5,R4           ;DATA CORRECT?
3719  021702  001401                  BEQ     68$             ;BR IF YES
3720  021704  104005                  ERROR   5               ;ERROR
3721  021706  104405          68$:    SCOP1                   ;SW09=1?
3722  021710  005100                  COM     R0              ;CHANGE TO FLOATING 1
3723  021712  000241                  CLC                     ;CLEAR CARRY
3724  021714  106100                  ROLB    R0              ;SHIFT BIT IN R0
3725  021716  001356                  BNE     69$             ;IF R0=0 THEN DONE
3726
```

```
3727
3728                                    ;************************** TEST 46 **************************
3729                                    ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
3730                                    ;*FLOAT A 1 THROUGH IBUS REGISTER 6
3731                                    ;*FLOAT A 0 THROUGH IBUS REGISTER 6
3732                                    ;;*********************************************************************
3733
3734                                    ;  TEST 46
3735                                    ;-------------
3736                                    ;;*********************************************************************
3737  021720  000004                TST46:  SCOPE
3738  021722  012737  000046  001202       MOV     #46,$TSTNM              ; LOAD THE NO. OF THIS TEST
3739  021730  012737  022074  001442       MOV     #TST47,NEXT             ; POINT TO THE START OF NEXT TEST.
3740  021736  012737  021756  001444       MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3741                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3742  021744  104410                       MSTCLR                  ;MASTER CLEAR KMC11
3743  021746  012702  000006               MOV     #6,R2           ;SAVE REGISTER ADDRESS FOR TYPEOUT
3744  021752  012700  000001               MOV     #1,R0           ;START WITH BIT 0
3745  021756                        64$:
3746  021756  010061  000004               MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3747  021762  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3748  021764  122106                       122100!6                ;MOV DATA TO IBUS REGISTER 6
3749  021766  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3750  021770  021145                       21005!<6*20>            ;READ FROM IBUS REGISTER 6
3751  021772  010005                       MOV     R0,R5           ;PUT EXPECTED IN R5
3752  021774  116104  000005               MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3753  022000  120504                       CMPB    R5,R4           ;DATA CORRECT?
3754  022002  001401                       BEQ     65$             ;BR IF YES
3755  022004  104005                       ERROR   5               ;ERROR
3756  022006  104405                65$:   SCOP1                   ;SW09=1?
3757  022010  000241                       CLC                     ;CLEAR CARRY
3758  022012  106100                       ROLB    R0              ;SHIFT BIT IN R0
3759  022014  001360                       BNE     64$             ;IF R0=0 THEN DONE
3760  022016  012737  022032  001444       MOV     #67$,LOCK       ;NEW SCOP1
3761  022024  012700  000001               MOV     #1,R0           ;START WITH BIT 0
3762  022030  005100                69$:   COM     R0              ;CHANGE TO FLOATING ZERO
3763  022032                        67$:
3764  022032  010061  000004               MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3765  022036  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3766  022040  122106                       122100!6                ;MOV DATA TO IBUS REGISTER 6
3767  022042  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3768  022044  021145                       21005!<6*20>            ;READ FROM IBUS REGISTER 6
3769  022046  010005                       MOV     R0,R5           ;PUT EXPECTED IN R5
3770  022050  116104  000005               MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3771  022054  120504                       CMPB    R5,R4           ;DATA CORRECT?
3772  022056  001401                       BEQ     68$             ;BR IF YES
3773  022060  104005                       ERROR   5               ;ERROR
3774  022062  104405                68$:   SCOP1                   ;SW09=1?
3775  022064  005100                       COM     R0              ;CHANGE TO FLOATING 1
3776  022066  000241                       CLC                     ;CLEAR CARRY
3777  022070  106100                       ROLB    R0              ;SHIFT BIT IN R0
3778  022072  001356                       BNE     69$             ;IF R0=0 THEN DONE
3779
3780
3781                                    ;************************** TEST 47 **************************
3782                                    ;*MICRO PROCESSOR IBUS REGISTER WRITE/READ TEST
```

```
3783                                    ;*FLOAT A 1 THROUGH IBUS REGISTER 7
3784                                    ;*FLOAT A 0 THROUGH IBUS REGISTER 7
3785                                    ;:*****************************************************************
3786
3787                                    ;  TEST 47
3788                                    ;-------------
3789                                    ;:*****************************************************************
3790  022074  000004                TST47:  SCOPE
3791  022076  012737  000047  001202        MOV     #47,$TSTNM              ; LOAD THE NO. OF THIS TEST
3792  022104  012737  022250  001442        MOV     #TST50,NEXT             ; POINT TO THE START OF NEXT TEST.
3793  022112  012737  022132  001444        MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3794                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3795  022120  104410                        MSTCLR                  ;MASTER CLEAR KMC11
3796  022122  012702  000007                MOV     #7,R2           ;SAVE REGISTER ADDRESS FOR TYPEOUT
3797  022126  012700  000001                MOV     #1,R0           ;START WITH BIT 0
3798  022132                        64$:
3799  022132  010061  000004                MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3800  022136  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3801  022140  122107                        122100!7                ;MOV DATA TO IBUS REGISTER 7
3802  022142  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3803  022144  021165                        21005!(7*20)            ;READ FROM IBUS REGISTER 7
3804  022146  010005                        MOV     R0,R5           ;PUT EXPECTED IN R5
3805  022150  116104  000005                MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3806  022154  120504                        CMPB    R5,R4           ;DATA CORRECT?
3807  022156  001401                        BEQ     65$             ;BR IF YES
3808  022160  104005                        ERROR   5               ;ERROR
3809  022162  104405                65$:    SCOP1                   ;SW09=1?
3810  022164  000241                        CLC                     ;CLEAR CARRY
3811  022166  106100                        ROLB    R0              ;SHIFT BIT IN R0
3812  022170  001360                        BNE     64$             ;IF R0=0 THEN DONE
3813  022172  012737  022206  001444        MOV     #67$,LOCK       ;NEW SCOP1
3814  022200  012700  000001                MOV     #1,R0           ;START WITH BIT 0
3815  022204  005100                69$:    COM     R0              ;CHANGE TO FLOATING ZERO
3816  022206                        67$:
3817  022206  010061  000004                MOV     R0,4(R1)        ;PUT PATTERN INTO PORT4
3818  022212  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3819  022214  122107                        122100!7                ;MOV DATA TO IBUS REGISTER 7
3820  022216  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3821  022220  021165                        21005!(7*20)            ;READ FROM IBUS REGISTER 7
3822  022222  010005                        MOV     R0,R5           ;PUT EXPECTED IN R5
3823  022224  116104  000005                MOVB    5(R1),R4        ;PUT "FOUND" INTO R4
3824  022230  120504                        CMPB    R5,R4           ;DATA CORRECT?
3825  022232  001401                        BEQ     68$             ;BR IF YES
3826  022234  104005                        ERROR   5               ;ERROR
3827  022236  104405                68$:    SCOP1                   ;SW09=1?
3828  022240  005100                        COM     R0              ;CHANGE TO FLOATING 1
3829  022242  000241                        CLC                     ;CLEAR CARRY
3830  022244  106100                        ROLB    R0              ;SHIFT BIT IN R0
3831  022246  001356                        BNE     69$             ;IF R0=0 THEN DONE
3832
3833
3834                                    ;*************************** TEST 50 ****************************
3835                                    ;*MICRO PROCESSOR IBUS DUAL ADDRESS TEST
3836                                    ;*WRITE ALL IBUS REGISTERS WITH INCREMENTING PATTERN
3837                                    ;*READ ALL IBUS REGISTERS TO VERIFY CORRECT ADDRESSING
3838                                    ;:*****************************************************************
```

```
                                        ;  TEST 50
                                        ;-------------
                                        ;;*****************************************************************
022250 000004                           TST50:  SCOPE
022252 012737 000050 001202                     MOV     #50,$TSTNM              ; LOAD THE NO. OF THIS TEST
022260 012737 022476 001442                     MOV     #TST51,NEXT             ; POINT TO THE START OF NEXT TEST.
022266 012737 022304 001444                     MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
                                                                        ;R1 CONTAINS BASE KMC11 ADDRESS
022274 104410                                   MSTCLR                  ;MASTER CLEAR KMC11
022276 012700 000001                            MOV     #1,R0           ;START WITH A ONE
022302 005002                                   CLR     R2              ;R2 CONTAINS ADDRESS OF REGISTER
022304 010203                           1$:     MOV     R2,R3           ;R3=REGISTER ADDRESS
022306 010061 000004                            MOV     R0,4(R1)        ;WRITE DATA TO PORT4
022312 042737 000017 022326                     BIC     #17,5$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
022320 050337 022326                            BIS     R3,5$           ;ADD ADDRESS TO INSTRUCTION
022324 104412                                   ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
022326 122100                           5$:     122100                  ;MOVE DATA TO IBUS REGISTER
022330 006303                                   ASL     R3              ;SHIFT ADDRESS
022332 006303                                   ASL     R3              ;4 TIMES TO GET
022334 006303                                   ASL     R3              ;IT TO BITS 4-7
022336 006303                                   ASL     R3              ;OF NEXT INSTRUCTION
022340 042737 000360 022354                     BIC     #360,6$         ;CLEAR ADDRESS FIELD
022346 050337 022354                            BIS     R3,6$           ;ADD ADDRESS TO INSTRUCTION
022352 104412                                   ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
022354 021005                           6$:     21005                   ;READ FROM IBUS REGISTER
022356 010005                                   MOV     R0,R5           ;PUT "EXPECTED" IN R5
022360 116104 000005                            MOVB    5(R1),R4        ;PUT "FOUND" IN R4
022364 120504                                   CMPB    R5,R4           ;IS DATA CORRECT?
022366 001401                                   BEQ     2$              ;BR IF YES
022370 104005                                   ERROR   5               ;DATA ERROR
022372 104405                           2$:     SCOP1                   ;SW09=1?
022374 005200                                   INC     R0              ;INCREMENT PATTERN
022376 005202                                   INC     R2              ;INCREMENT REGISTER ADDRESS
022400 022702 000010                            CMP     #7+1,R2 ;LAST ADDRESS DONE?
022404 001337                                   BNE     1$              ;BR IF NO
022406 012737 022424 001444                     MOV     #3$,LOCK        ;NEW SCOP1
022414 012700 000001                            MOV     #1,R0           ;RESTART PATTERN TO 1
022420 005002                                   CLR     R2              ;RESTART AT ADDRESS 0
022422 005003                                   CLR     R3              ;RESTART AT ADDRESS 0
022424 042737 000360 022440             3$:     BIC     #360,7$         ;CLEAR ADDRESS FIELD OF INSTRUCTION
022432 050337 022440                            BIS     R3,7$           ;ADD ADDRESS TO INSTRUCTION
022436 104412                                   ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
022440 021005                           7$:     21005                   ;READ FROM IBUS REGISTER
022442 010005                                   MOV     R0,R5           ;PUT "EXPECTED" IN R5
022444 116104 000005                            MOVB    5(R1),R4        ;PUT "FOUND" IN R5
022450 120504                                   CMPB    R5,R4           ;DATA CORRECT?
022452 001401                                   BEQ     4$              ;BR IF YES
022454 104005                                   ERROR   5               ;DUAL ADDRESSING ERROR
022456 104405                           4$:     SCOP1                   ;SW09=1?
022460 005200                                   INC     R0              ;INCREMENT PATTERN
022462 005202                                   INC     R2              ;NEXT ADDRESS
022464 062703 000020                            ADD     #20,R3          ;ADD 1 TO ADDRESS IN R3(SHIFTED 4 TIMES)
022470 022702 000010                            CMP     #7+1,R2 ;LAST ADDRESS DONE?
022474 001353                                   BNE     3$              ;BR IF NO
```

```
3895
3896
3897                                    ;*************************** TEST 51 ****************************
3898                                    ;*MICRO PROCESSOR BR REGISTER TEST
3899                                    ;*FLOAT A 1 THROUGH THE BR
3900                                    ;*FLOAT A 0 THROUGH THE BR
3901                                    ;:************************************************************
3902
3903                                    ;  TEST 51
3904                                    ;--------------
3905                                    ;:************************************************************
3906  022476  000004                TST51:  SCOPE
3907  022500  012737  000051  001202        MOV     #51,$TSTNM              ; LOAD THE NO. OF THIS TEST
3908  022506  012737  022646  001442        MOV     #TST52,NEXT             ; POINT TO THE START OF NEXT TEST.
3909  022514  012737  022530  001444        MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3910                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3911  022522  104410                       MSTCLR                  ;MASTER CLEAR KMC11
3912  022524  012700  000001               MOV     #1,R0           ;START PATTERN WITH BIT0
3913  022530                        64$:
3914  022530  010061  000004               MOV     R0,4(R1)        ;WRITE PATTERN IN PORT4
3915  022534  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3916  022536  120500                       120500                  ;MOVE DATA TO THE BR REGISTER
3917  022540  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3918  022542  061225                       061225                  ;MOVE BR TO PORT 5
3919  022544  010005                       MOV     R0,R5           ;PUT "EXPECTED" IN R5
3920  022546  116104  000005               MOVB    5(R1),R4        ;PUT "FOUND" IN R4
3921  022552  120504                       CMPB    R5,R4           ;DATA CORRECT?
3922  022554  001401                       BEQ     65$             ;BR IF YES
3923  022556  104006                       ERROR   6               ;DATA ERROR
3924  022560  104405                65$:   SCOP1
3925  022562  000241                       CLC                     ;CLEAR CARRY
3926  022564  106100                       ROLB    R0              ;SHIFT BIT IN R0
3927  022566  001360                       BNE     64$             ;DONE IF R0=0
3928  022570  012737  022604  001444        MOV     #67$,LOCK       ;NEW SCOP1
3929  022576  012700  000001               MOV     #1,R0           ;START PATTERN WITH BIT0
3930  022602  005100                69$:   COM     R0              ;CHANGE TO FLOATING ZERO
3931  022604                        67$:
3932  022604  010061  000004               MOV     R0,4(R1)        ;WRITE PATTERN IN PORT4
3933  022610  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3934  022612  120500                       120500                  ;MOVE DATA TO THE BR REGISTER
3935  022614  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3936  022616  061225                       061225                  ;MOVE BR TO PORT 5
3937  022620  010005                       MOV     R0,R5           ;PUT "EXPECTED" IN R5
3938  022622  116104  000005               MOVB    5(R1),R4        ;PUT "FOUND" IN R4
3939  022626  120504                       CMPB    R5,R4           ;DATA CORRECT?
3940  022630  001401                       BEQ     68$             ;BR IF YES
3941  022632  104006                       ERROR   6               ;DATA ERROR
3942  022634  104405                68$:   SCOP1
3943  022636  005100                       COM     R0              ;CHANGE BACK TO A ONE
3944  022640  000241                       CLC                     ;CLEAR CARRY
3945  022642  106100                       ROLB    R0              ;SHIFT BIT IN R0
3946  022644  001356                       BNE     69$             ;DONE IF R0=0
3947
3948
3949                                    ;*************************** TEST 52 ****************************
3950                                    ;*SCRATCH PAD TEST
```

```
3951                                    ;*FLOAT A 1 THROUGH EACH SCRATCH PAD LOCATION
3952                                    ;*FLOAT A 0 THROUGH EACH SCRATCH PAD LOCATION
3953                                    ;;*****************************************************************
3954
3955                                    ;  TEST 52
3956                                    ;--------------
3957                                    ;;*****************************************************************
3958  022646  000004                TST52:  SCOPE
3959  022650  012737  000052  001202        MOV     #52,$TSTNM              ; LOAD THE NO. OF THIS TEST
3960  022656  012737  023114  001442        MOV     #TST53,NEXT             ; POINT TO THE START OF NEXT TEST.
3961  022664  012737  022702  001444        MOV     #64$,LOCK               ; ADDRESS FOR LOCK ON DATA.
3962                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
3963  022672  104410                       MSTCLR                  ;MASTER CLEAR KMC11
3964  022674  005002                       CLR     R2              ;START AT ADDRESS ZERO
3965  022676  012700  000001              MOV     #1,R0           ;START WITH BIT0
3966  022702  042737  000017  022722  64$:  BIC     #17,65$         ;CLEAR ADDRESS FIELD OF INSTRUCTION
3967  022710  050237  022722              BIS     R2,65$          ;ADD ADDRESS TO INSTRUCTION
3968  022714  010061  000004              MOV     R0,4(R1)        ;WRITE PATTERN TO PORT4
3969  022720  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3970  022722  123100                65$:    123100                  ;WRITE SCRATCH PAD(ADDRESS IN R2)
3971  022724  042737  000017  022740        BIC     #17,66$         ;CLEAR ADDRESS FIELD OF INSTRUCTION
3972  022732  050237  022740              BIS     R2,66$          ;ADD ADDRESS TO INSTRUCTION
3973  022736  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3974  022740  040600                66$:    040600                  ;MOV SP TO BR
3975  022742  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3976  022744  061225                       061225                  ;MOVE BR TO PORT5
3977  022746  010005                       MOV     R0,R5           ;PUT "EXPECTED" IN R5
3978  022750  116104  000005              MOVB    5(R1),R4        ;PUT "FOUND" IN R4
3979  022754  120504                       CMPB    R5,R4           ;DATA CORRECT
3980  022756  001401                       BEQ     67$             ;BR IF YES
3981  022760  104007                       ERROR   7               ;DATA ERROR
3982  022762  104405                67$:    SCOP1                   ;SW09=1?
3983  022764  000241                       CLC                     ;CLEAR CARRY
3984  022766  106100                       ROLB    R0              ;SHIFT BIT IN R0
3985  022770  001344                       BNE     64$             ;DONE IF R0=0
3986  022772  012737  023006  001444        MOV     #69$,LOCK               ;NEW SCOP1
3987  023000  012700  000001              MOV     #1,R0           ;START WITH BIT0
3988  023004  005100                73$:    COM     R0              ;CHANGE TO FLOATING ZERO
3989  023006  042737  000017  023026  69$:  BIC     #17,70$         ;CLEAR ADDRESS FIELD OF INSTRUCTION
3990  023014  050237  023026              BIS     R2,70$          ;ADD ADDRESS TO INSTRUCTION
3991  023020  010061  000004              MOV     R0,4(R1)        ;WRITE PATTERN TO PORT4
3992  023024  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3993  023026  123100                70$:    123100                  ;WRITE SCRATCH PAD(ADDRESS IN R2)
3994  023030  042737  000017  023044        BIC     #17,71$         ;CLEAR ADDRESS FIELD OF INSTRUCTION
3995  023036  050237  023044              BIS     R2,71$          ;ADD ADDRESS TO INSTRUCTION
3996  023042  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3997  023044  040600                71$:    040600                  ;MOV SP TO BR
3998  023046  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
3999  023050  061225                       061225                  ;MOVE BR TO PORT5
4000  023052  010005                       MOV     R0,R5           ;PUT "EXPECTED" IN R5
4001  023054  116104  000005              MOVB    5(R1),R4        ;PUT "FOUND" IN R4
4002  023060  120504                       CMPB    R5,R4           ;DATA CORRECT
4003  023062  001401                       BEQ     72$             ;BR IF YES
4004  023064  104007                       ERROR   7               ;DATA ERROR
4005  023066  104405                72$:    SCOP1                   ;SW09=1?
4006  023070  005100                       COM     R0              ;CHANGE BACK TO A ONE
```

```
4007  023072  000241                    CLC                     ;CLEAR CARRY
4008  023074  106100                    ROLB    R0              ;SHIFT BIT IN R0
4009  023076  001342                    BNE     73$             ;DONE IF R0=0
4010  023100  012700  000001            MOV     #1,R0           ;RESTART AT BIT 0
4011  023104  005202                    INC     R2              ;NEXT SP ADDRESS
4012  023106  022702  000020            CMP     #20,R2  ;LAST ADDRESS?
4013  023112  001273                    BNE     64$             ;BR IF NO
4014
4015
4016                                    ;************************** TEST 53 **************************
4017                                    ;*SCRATCH PAD DUAL ADDRESSING TEST
4018                                    ;*WRITE AN INCREMENTING PATTERN IN ALL SP LOCATIONS
4019                                    ;*READ ALL SP LOCATIONS TO VERIFY CORRECT ADDRESSING
4020                                    ;:**********************************************************
4021
4022                                    ;  TEST 53
4023                                    ;-------------
4024                                ;;**********************************************************
4025  023114  000004            TST53:  SCOPE
4026  023116  012737  000053  001202    MOV     #53,$TSTNM              ; LOAD THE NO. OF THIS TEST
4027  023124  012737  023336  001442    MOV     #TST54,NEXT             ; POINT TO THE START OF NEXT TEST.
4028  023132  012737  023150  001444    MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
4029                                                            ;R1 CONTAINS BASE KMC11 ADDRESS
4030  023140  104410                    MSTCLR                  ;MASTER CLEAR KMC11
4031  023142  012700  000001            MOV     #1,R0           ;START WITH A 1
4032  023146  005003                    CLR     R3              ;ADDRESS 0
4033  023150  010302            1$:     MOV     R3,R2           ;MOVE ADDRESS TO R2
4034  023152  042737  000017  023172    BIC     #17,2$          ;CLEAR ADDRESS FIELD
4035  023160  050237  023172            BIS     R2,2$           ;ADD ADDRESS TO INSTRUCTION
4036  023164  010061  000004            MOV     R0,4(R1)        ;WRITE PATTERN TO PORT4
4037  023170  104412                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4038  023172  123100            2$:     123100                  ;WRITE SP(ADDRESS IN R2)
4039  023174  042737  000017  023210    BIC     #17,3$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
4040  023202  050237  023210            BIS     R2,3$           ;ADD ADDRESS TO INSTRUCTION
4041  023206  104412                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4042  023210  060600            3$:     60600                   ;MOV SP TO BR
4043  023212  104412                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4044  023214  061225                    61225                   ;MOV BR TO PORT5
4045  023216  010005                    MOV     R0,R5           ;PUT "EXPECTED" IN R5
4046  023220  116104  000005            MOVB    5(R1),R4        ;PUT "FOUND" IN R4
4047  023224  120504                    CMPB    R5,R4           ;DATA CORRECT?
4048  023226  001401                    BEQ     4$              ;BR IF YES
4049  023230  104007                    ERROR   7               ;DATA ERROR
4050  023232  104405            4$:     SCOP1                   ;SW09=0
4051  023234  005200                    INC     R0              ;INCREMENT PATTERN
4052  023236  005203                    INC     R3              ;NEXT ADDRESS
4053  023240  022703  000020            CMP     #20,R3          ;LAST ADDRESS DONE?
4054  023244  001341                    BNE     1$              ;BR IF NO
4055  023246  012737  023262  001444    MOV     #5$,LOCK        ;NEW SCOP1
4056  023254  012700  000001            MOV     #1,R0           ;RESTART PATTERN AT 1
4057  023260  005003                    CLR     R3              ;RESTART AT ADDRESS ZERO
4058  023262  010302            5$:     MOV     R3,R2           ;PUT ADDRESS IN R2
4059  023264  042737  000017  023300    BIC     #17,6$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
4060  023272  050237  023300            BIS     R2,6$           ;ADD ADDRESS TO INSTRUCTION
4061  023276  104412                    ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4062  023300  060600            6$:     60600                   ;MOV SP TO BR
```

```
4063  023302  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4064  023304  061225                          61225                   ;MOV BR TO PORT5
4065  023306  010005                          MOV     R0,R5           ;PUT "EXPECTED" IN R5
4066  023310  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" IN R4
4067  023314  120504                          CMPB    R5,R4           ;DATA CORRECT?
4068  023316  001401                          BEQ     7$              ;BR IF YES
4069  023320  104007                          ERROR   7               ;SP ADDRESSING ERROR
4070  023322  104405                  7$:     SCOP1                   ;SW09=1?
4071  023324  005200                          INC     R0              ;INCREMENT PATTERN
4072  023326  005203                          INC     R3              ;NEXT ADDRESS
4073  023330  022703  000020                  CMP     #20,R3  ;LAST ADDRESS DONE?
4074  023334  001352                          BNE     5$              ;BR IF NO
4075
4076
4077                                          ;*************************** TEST 54 ****************************
4078                                          ;*INTERRUPT TEST
4079                                          ;*TEST THAT DEVICE CAN INTERRUPT TO VECTOR A
4080                                          ;:****************************************************************
4081
4082                                          ;  TEST 54
4083                                          ;--------------
4084                                  ;;******************************************************************
4085  023336  000004                  TST54:  SCOPE
4086  023340  012737  000054  001202          MOV     #54,$TSTNM              ; LOAD THE NO. OF THIS TEST
4087  023346  012737  023432  001442          MOV     #TST55,NEXT             ; POINT TO THE START OF NEXT TEST.
4088                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4089  023354  000005                          RESET                   ;BUS RESET
4090  023356  005011                          CLR     (R1)            ;CLEAR RUN
4091  023360  004537  035516                  JSR     R5,SETVEC       ;SET UP VECTORS
4092  023364  023426                          3$                      ;XX0
4093  023366  023424                          2$                      ;XX4
4094  023370     340     340                  .BYTE   340,340         ;LEVEL 7
4095  023372  012737  000340  177776  1$:     MOV     #340,PS         ;PS = LEVEL 7
4096  023400  012761  000200  000004          MOV     #200,4(R1)      ;WRITE PORT4
4097  023406  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4098  023410  121111                          121111                  ;SET BR RQ IN IBUS* REG 11
4099  023412  005037  177776                  CLR     PS              ;ALLOW INTERRUPT
4100  023416  000240                          NOP
4101  023420  104010                          ERROR   10              ;NO INTERRUPT
4102  023422  000403                          BR      4$
4103  023424  104011                  2$:     ERROR   11              ;WRONG VECTOR
4104  023426  012706  001200          3$:     MOV     #STACK,SP       ;RESET STACK
4105  023432                          4$:
4106
4107
4108                                          ;*************************** TEST 55 ****************************
4109                                          ;*INTERRUPT TEST
4110                                          ;*TEST THAT DEVICE CAN INTERRUPT TO VECTOR B
4111                                          ;:****************************************************************
4112
4113                                          ;  TEST 55
4114                                          ;--------------
4115                                  ;;******************************************************************
4116  023432  000004                  TST55:  SCOPE
4117  023434  012737  000055  001202          MOV     #55,$TSTNM              ; LOAD THE NO. OF THIS TEST
4118  023442  012737  023524  001442          MOV     #TST56,NEXT             ; POINT TO THE START OF NEXT TEST.
```

```
4119                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
4120  023450  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4121  023452  004537  035516                  JSR     R5,SETVEC       ;SET UP VECTORS
4122  023456  023516                          2$                      ;XXO
4123  023460  023520                          3$                      ;XX4
4124  023462     340     340                  .BYTE   340,340         ;LEVEL 7
4125  023464  012737  000340  177776  1$:     MOV     #340,PS         ;PS = LEVEL 7
4126  023472  012761  000300  000004          MOV     #300,4(R1)      ;WRITE PORT4
4127  023500  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4128  023502  121111                          121111                  ;SET BR RQ IN IBUS* REG 11
4129  023504  005037  177776                  CLR     PS              ;ALLOW INTERRUPT
4130  023510  000240                          NOP
4131  023512  104010                          ERROR   10              ;NO INTERRUPT
4132  023514  000403                          BR      4$
4133  023516  104011                  2$:     ERROR   11              ;WRONG VECTOR
4134  023520  012706  001200          3$:     MOV     #STACK,SP       ;RESET STACK
4135  023524                          4$:
4136
4137
4138                                          ;************************** TEST 56 **************************
4139                                          ;*PRIORITY INTERRUPT TESTS
4140                                          ;*SET PS TO ALL BR LEVELS EQUAL OR GREATER THAN
4141                                          ;*THE KMC11 LEVEL,VERIFY THAT KMC11 DOES NOT INTERRUPT
4142                                          ;;*********************************************************
4143
4144                                          ;  TEST 56
4145                                          ;-------------
4146                                          ;;*********************************************************
4147  023524  000004                  TST56:  SCOPE
4148  023526  012737  000056  001202          MOV     #56,$TSTNM              ; LOAD THE NO. OF THIS TEST
4149  023534  012737  023646  001442          MOV     #TST57,NEXT             ; POINT TO THE START OF NEXT TEST.
4150                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4151  023542  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4152  023544  012702  000340                  MOV     #340,R2         ;PUT LEVEL 7 IN R2
4153  023550  010237  177776                  MOV     R2,PS           ;SET PRIORITY TO 7
4154  023554  013700  002050                  MOV     STAT1,R0        ;GET BR LEVEL OF KMC11
4155  023560  006200                          ASR     R0              ;SHIFT R0 4 TIMES
4156  023562  006200                          ASR     R0              ;TO GET PROPER LEVEL
4157  023564  006200                          ASR     R0
4158  023566  006200                          ASR     R0
4159  023570  042700  177437                  BIC     #177437,R0      ;CLEAR UNWANTED BITS
4160  023574  004537  035516                  JSR     R5,SETVEC       ;SET UP VECTORS
4161  023600  023642                          2$                      ;A VECTOR
4162  023602  023642                          2$                      ;B VECTOR
4163  023604     340     340                  .BYTE   340,340         ;PRIORITY 7
4164  023606  012761  000200  000004  4$:     MOV     #200,4(R1)      ;LOAD PORT4
4165  023614  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4166  023616  121111                          121111                  ;SET BR REQUEST
4167  023620  010237  177776          5$:     MOV     R2,PS           ;PUT LEVEL IN R2 IN PS
4168  023624  000240                          NOP
4169  023626  020002                          CMP     R0,R2           ;IS PRESENT PS LEVEL = TO KMC LEVEL
4170  023630  001403                          BEQ     1$              ;BR IF YES
4171  023632  162702  000040                  SUB     #40,R2          ;NO GET NEXT LOWER LEVEL IN R2
4172  023636  000770                          BR      5$              ;AND CONTINUE WITH TEST
4173  023640  104420                  1$:     ADVANCE                 ; ADVANCE LOOP
4174  023642  104020                  2$:     ERROR   20              ;ERROR UNEXPECTED INTERRUPT
```

```
4175  023644  000002                          RTI
4176
4177
4178                                          ;*************************** TEST 57 ****************************
4179                                          ;*PRIORITY INTERRUPT TESTS
4180                                          ;*SET PS TO ALL BR LEVELS LESS THAN THE KMC11 LEVEL
4181                                          ;*VERIFY THAT THE KMC11 WILL INTERRUPT
4182                                          ;;************************************************************************
4183
4184                                          ;  TEST 57
4185                                          ;-------------
4186                                      ;;**********************************************************************
4187  023646  000004                      ↑ST57:  SCOPE
4188  023650  012737  000057  001202              MOV     #57,$TSTNM              ; LOAD THE NO. OF THIS TEST
4189  023656  012737  024014  001442              MOV     #↑ST60,NEXT             ; POINT TO THE START OF NEXT TEST.
4190                                                                      ;R1 CONTAINS BASE KMC11 ADDRESS
4191  023664  104410                              MSTCLR                  ;MASTER CLEAR KMC11
4192  023666  012702  000340                      MOV     #340,R2         ;PUT LEVEL 7 IN R2
4193  023672  010237  177776                      MOV     R2,PS           ;SET PRIORITY TO 7
4194  023676  013700  002050                      MOV     STAT1,R0        ;GET BR LEVEL OF KMC11
4195  023702  006200                              ASR     R0              ;SHIFT R0 4 TIMES
4196  023704  006200                              ASR     R0              ;TO GET PROPER LEVEL
4197  023706  006200                              ASR     R0
4198  023710  006200                              ASR     R0
4199  023712  042700  177437                      BIC     #177437,R0      ;CLEAR UNWANTED BITS
4200  023716  010002                              MOV     R0,R2           ;PUT KMC LEVEL IN R2
4201  023720  162702  000040                      SUB     #40,R2          ;GET NEXT LOWER LEVEL IN R2
4202  023724  004537  035516                      JSR     R5,SETVEC       ;SET UP VECTORS
4203  023730  023776                              2$                      ;A VECTOR
4204  023732  024004                              3$                      ;B VECTOR
4205  023734     340     340                      .BYTE   340,340         ;PRIORITY 7
4206  023736  012761  000200  000004  4$:         MOV     #200,4(R1)      ;LOAD PORT4
4207  023744  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4208  023746  121111                              121111                  ;SET BR REQUEST
4209  023750  010237  177776           5$:        MOV     R2,PS           ;PUT LEVEL IN R2 IN PS
4210  023754  000240                              NOP
4211  023756  104010                              ERROR   10              ;ERROR, NO INTERRUPT
4212  023760  022702  000140           6$:        CMP     #140,R2         ;IS IT DOWN TO LEVEL 3 YET?
4213  023764  001403                              BEQ     1$              ;YES.KMC DID NOT INTERUPT, ERROR
4214  023766  162702  000040                      SUB     #40,R2          ;PUT NEXT LOWER LEVEL IN R2
4215  023772  000761                              BR      4$              ;CONTINUE TEST
4216  023774  104420                      1$:     ADVANCE                 ;  ADVANCE LOOP
4217  023776  012716  023760           2$:        MOV     #6$,(SP)        ;SET UP FOR RTI
4218  024002  000002                              RTI
4219  024004  104011                      3$:     ERROR   11              ;ERROR, WRONG VECTOR
4220  024006  012716  023760                      MOV     #6$,(SP)        ;SET UP FOR RTI
4221  024012  000002                              RTI
4222
4223
4224                                          ;*************************** TEST 60 ****************************
4225                                          ;*NPR TEST
4226                                          ;*TEST OF DPTO, 1 WORD FROM UPROC TO 11 MEMORY
4227                                          ;;************************************************************************
4228
4229                                          ;  TEST 60
4230                                          ;-------------
```

```
4231                                    ;;*****************************************************************
4232  024014  000004                    TST60:  SCOPE
4233  024016  012737  000060  001202            MOV     #60,$TSTNM              ; LOAD THE NO. OF THIS TEST
4234  024024  012737  024122  001442            MOV     #TST61,NEXT             ; POINT TO THE START OF NEXT TEST.
4235                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
4236  024032  000005                            RESET                   ;BUS RESET
4237  024034  005011                            CLR     (R1)            ;CLEAR RUN
4238  024036  005061  000004                    CLR     4(R1)           ;CLR PORT4
4239  024042  004537  035540                    JSR     R5,NPRSET       ;SET UP IBUS REG 0-7
4240  024046  000000                            0                       ;IN DATA
4241  024050  177777                            -1                      ;OUT DATA
4242  024052  024120                            3$                      ;IN BA
4243  024054  024116                            2$                      ;OUT BA
4244  024056  005037  024116                    CLR     2$              ;CLEAR 2$
4245  024062  012761  000021  000004            MOV     #21,4(R1)       ;WRITE PORT4
4246  024070  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4247  024072  121110                            121110                  ;SET NPR BITS IN IBUS* REG 11
4248  024074  000240                            NOP
4249  024076  012705  177777                    MOV     #-1,R5          ;PUT "EXPECTED" IN R5
4250  024102  013704  024116                    MOV     2$,R4           ;PUT "FOUND" IN R4
4251  024106  020504                            CMP     R5,R4           ;DATA CORRECT?
4252  024110  001401                            BEQ     4$              ;BR IF YES
4253  024112  104012                            ERROR   12              ;ERROR NPR FAILED
4254  024114  104420                    4$:     ADVANCE                 ; ADVANCE LOOP
4255  024116  000000                    2$:     0                       ;OUT BA
4256  024120  000000                    3$:     0                       ;IN BA
4257
4258
4259                                                                    ;************************** TEST 61 **************************
4260                                                                    ;*NPR TEST
4261                                                                    ;*TEST OF DATI, 1 WORD FROM 11 MEMORY TO UPROC
4262                                    ;;*****************************************************************
4263
4264                                    ;  TEST 61
4265                                    ;-------------
4266                                ;;*****************************************************************
4267  024122  000004                TST61:  SCOPE
4268  024124  012737  000061  001202            MOV     #61,$TSTNM              ; LOAD THE NO. OF THIS TEST
4269  024132  012737  024240  001442            MOV     #TST62,NEXT             ; POINT TO THE START OF NEXT TEST.
4270                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
4271  024140  104410                            MSTCLR                  ;MASTER CLEAR KMC11
4272  024142  005061  000004                    CLR     4(R1)           ;CLR PORT4
4273  024146  004537  035540                    JSR     R5,NPRSET       ;SET UP IBUS REG 0-7
4274  024152  000000                            0                       ;IN DATA
4275  024154  177777                            -1                      ;OUT DATA
4276  024156  024236                            3$                      ;IN BA
4277  024160  024234                            2$                      ;OUT BA
4278  024162  012737  177777  024236            MOV     #-1,3$          ;PUT DATA IN 3$
4279  024170  012761  000001  000004            MOV     #1,4(R1)        ;WRITE PORT4
4280  024176  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4281  024200  121110                            121110                  ;SET NPR BITS IN IBUS* REG 11
4282  024202  000240                            NOP
4283  024204  012705  177777                    MOV     #-1,R5          ;PUT "EXPECTED" IN R5
4284  024210  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4285  024212  021004                            021004                  ;MOVE IN DATA LOW BYTE TO PORT4
4286  024214  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
4287  024216  021025                          021025                  ;MOVE IN DATA HIGH BYTE TO PORTS
4288  024220  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
4289  024224  020504                          CMP     R5,R4           ;DATA CORRECT?
4290  024226  001401                          BEQ     4$              ;BR IF YES
4291  024230  104012                          ERROR   12              ;ERROR NPR FAILED
4292  024232  104420                  4$:     ADVANCE                 ; ADVANCE LOOP
4293  024234  000000                  2$:     0                       ;OUT BA
4294  024236  000000                  3$:     0                       ;IN BA
4295
4296
4297                                          ;*************************** TEST 62 ***********************
4298                                          ;*NPR TEST
4299                                          ;*TEST OF DATOB, 1 BYTE FROM UPROC TO 11 MEMORY
4300                                          ;:*******************************************************************
4301
4302                                          ;  TEST 62
4303                                          ;--------------
4304                                  ;;*******************************************************************
4305  024240  000004                  TST62:  SCOPE
4306  024242  012737  000062  001202          MOV     #62,$TSTNM              ; LOAD THE NO. OF THIS TEST
4307  024250  012737  024344  001442          MOV     #TST63,NEXT             ; POINT TO THE START OF NEXT TEST.
4308                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4309  024256  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4310  024260  005061  000004                  CLR     4(R1)           ;CLR PORT4
4311  024264  004537  035540                  JSR     R5,NPRSET       ;SET UP IBUS REG 0-7
4312  024270  000000                          0                       ;IN DATA
4313  024272  177777                          -1                      ;OUT DATA
4314  024274  024342                          3$                      ;IN BA
4315  024276  024341                          2$+1            ;OUT BA
4316  024300  005037  024340                  CLR     2$              ;CLEAR 2$
4317  024304  012761  000221  000004          MOV     #221,4(R1)      ;WRITE PORT4
4318  024312  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4319  024314  121110                          121110                  ;SET NPR BITS IN IBUS* REG 11
4320  024316  000240                          NOP
4321  024320  012705  177400                  MOV     #177400,R5              ;PUT "EXPECTED" IN R5
4322  024324  013704  024340                  MOV     2$,R4           ;PUT "FOUND" IN R4
4323  024330  020504                          CMP     R5,R4           ;DATA CORRECT?
4324  024332  001401                          BEQ     4$              ;BR IF YES
4325  024334  104012                          ERROR   12              ;ERROR NPR FAILED
4326  024336  104420                  4$:     ADVANCE                 ; ADVANCE LOOP
4327  024340  000000                  2$:     0                       ;OUT BA
4328  024342  000000                  3$:     0                       ;IN BA
4329
4330
4331                                          ;*************************** TEST 63 ***************************
4332                                          ;*TEST OF EA BITS 16 AND 17
4333                                          ;*DO A DATO TO AN ADDRESS USING OUT BA BITS 16 AND 17
4334                                          ;*VERIFY CORRECT RESULTS
4335                                          ;:*******************************************************************
4336
4337                                          ;  TEST 63
4338                                          ;--------------
4339                                  ;;*******************************************************************
4340  024344  000004                  TST63:  SCOPE
4341  024346  012737  000063  001202          MOV     #63,$TSTNM              ; LOAD THE NO. OF THIS TEST
4342  024354  012737  024502  001442          MOV     #TST64,NEXT             ; POINT TO THE START OF NEXT TEST.
```

```
4343
4344  024362  104410                          MSTCLR          ;R1 CONTAINS BASE KMC11 ADDRESS
4345  024364  013737  002074  024412          MOV     KMP04,1$        ;MASTER CLEAR KMC11
4346  024372  013737  002074  024410          MOV     KMP04,2$        ;USE SEL4 FOR ADDRESS
4347  024400  004537  035540                  JSR     R5,NPRSET       ;USE SEL4 FOR ADDRESS
4348  024404  000000                          0                       ;LOAD BA AND DATA
4349  024406  125252                          125252                  ;IN DATA
4350  024410  000000                  2$:     0                       ;OUT DATA
4351  024412  000000                  1$:     0                       ;IN BA
4352  024414  012761  000014  000004          MOV     #14,4(R1)       ;OUT BA
4353  024422  104412                          ROMCLK                  ;LOAD SEL 4 WITH OUT BA16 AND 17
4354  024424  121111                          121111                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4355  024426  012761  000021  000004          MOV     #21,4(R1)       ;SET OUTBA 16 AND 17
4356  024434  012761  121110  000006          MOV     #121110,6(R1)   ;LOAD SEL4
4357  024442  012711  003000                  MOV     #BIT9!BIT10,(R1);PUT INSTRUCTION IN SEL6
4358  024446  052711  000400                  BIS     #BIT8,(R1)      ;SET CROMI AND CROMO!!
4359  024452  000240                          NOP                     ;CLOCK IT!
4360  024454  012705  121110                  MOV     #121110,R5      ;WAIT FOR NPR
4361  024460  104412                          ROMCLK                  ;PUT "EXPECTED" IN R5
4362  024462  021044                          021044                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4363  024464  104412                          ROMCLK                  ;MOVE OUT DATA LB TO SEL4
4364  024466  021065                          021065                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4365  024470  016104  000004                  MOV     4(R1),R4        ;MOVE OUT DATA HB TO SEL5
4366  024474  020504                          CMP     R5,R4           ;PUT "FOUND" IN R4
4367  024476  001401                          BEQ     3$              ;CORRECT RESULTS ?
4368  024500  104012                          ERROR   12              ;BR IF YES
4369  024502                          3$:                             ;ERROR BA 16 AND 17 FAILED
4370
4371
4372                                          ;************************** TEST 64 **************************
4373                                          ;*TEST OF EA BITS 16 AND 17
4374                                          ;*DO A DATI USING IN BA BITS 16 AND 17
4375                                          ;*VERIFY CORRECT RESULTS
4376                                          ;:*************************************************************
4377
4378                                          ;  TEST 64
4379                                          ;--------------
4380                                  ;;**********************************************************
4381  024502  000004                  TST64:  SCOPE
4382  024504  012737  000064  001202          MOV     #64,$TSTNM              ; LOAD THE NO. OF THIS TEST
4383  024512  012737  024626  001442          MOV     #TST65,NEXT             ; POINT TO THE START OF NEXT TEST.
4384                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4385  024520  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4386  024522  013737  002074  024550          MOV     KMP04,1$        ;USE SEL4 FOR ADDRESS
4387  024530  013737  002074  024546          MOV     KMP04,2$        ;USE SEL4 FOR ADDRESS
4388  024536  004537  035540                  JSR     R5,NPRSET       ;LOAD BA AND DATA
4389  024542  000000                          0                       ;IN DATA
4390  024544  125252                          125252                  ;OUT DATA
4391  024546  000000                  2$:     0                       ;IN BA
4392  024550  000000                  1$:     0                       ;OUT BA
4393  024552  012761  000015  000004          MOV     #15,4(R1)       ;LOAD SEL4
4394  024560  012761  121110  000006          MOV     #121110,6(R1)   ;PUT INSTRUCTION IN SEL6
4395  024566  012711  003000                  MOV     #BIT9!BIT10,(R1);SET CROMI AND CROMO!!
4396  024572  052711  000400                  BIS     #BIT8,(R1)      ;CLOCK IT!
4397  024576  000240                          NOP                     ;WAIT FOR NPR
4398  024600  012705  121110                  MOV     #121110,R5      ;PUT "EXPECTED" IN R5
```

```
4399  024604  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4400  024606  021004                          021004                  ;MOVE IN DATA LB TO SEL4
4401  024610  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4402  024612  021025                          021025                  ;MOVE IN DATA HB TO SEL5
4403  024614  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
4404  024620  020504                          CMP     R5,R4           ;CORRECT RESULTS ?
4405  024622  001401                          BEQ     3$              ;BR IF YES
4406  024624  104012                          ERROR   12              ;ERROR BA 16 AND 17 FAILED
4407  024626                          3$:
4408
4409
4410                                          ;*************************** TEST 65 ****************************
4411                                          ;*NPR NON-EXISTENT MEMORY TEST
4412                                          ;*DO A DATO TO A NON-EXISTENT ADDRESS
4413                                          ;*VERIFY THAT THE NON-EXISTENT BIT SET IN IBUS REG 11
4414                                          ;:************************************************************
4415
4416                                          ;  TEST 65
4417                                          ;-------------
4418                                  ;:******************************************************************
4419  024626  000004                  TST65:  SCOPE
4420  024630  012737  000065  001202          MOV     #65,$TSTNM              ; LOAD THE NO. OF THIS TEST
4421  024636  012737  024736  001442          MOV     #TST66,NEXT             ; POINT TO THE START OF NEXT TEST.
4422                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4423  024644  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4424  024646  004537  035540                  JSR     R5,NPRSET       ;LOAD IBUS REGISTERS 0-7
4425  024652  000000                          0                       ;IN DATA
4426  024654  000000                          0                       ;OUT DATA
4427  024656  177320                          177320                  ;IN BA
4428  024660  177320                          177320                  ;OUT BA
4429  024662  012761  000014  000004          MOV     #14,4(R1)       ;SET OUT BA BITS 16+17 IN PORT4
4430  024670  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4431  024672  121111                          121111                  ;SET OUTBA 16 AND 17
4432  024674  012761  000021  000004          MOV     #21,4(R1)       ;SET NPR REQUEST BITS IN PORT4
4433  024702  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4434  024704  121110                          121110                  ;MOV IBUS* 4 TO IBUS* 10
4435  024706  000240                          NOP
4436  024710  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4437  024712  121225                          121225                  ;MOV IBUS*11 TO IBUS*5
4438  024714  012705  000001                  MOV     #1,R5           ;PUT "EXPECTED" IN R5
4439  024720  116104  000005                  MOVB    5(R1),R4        ;PUT "FOUND" IN R4
4440  024724  042704  177776                  BIC     #177776,R4      ;CLEAR UNWANTED BITS
4441  024730  020504                          CMP     R5,R4           ;DATA CORRECT?
4442  024732  001401                          BEQ     1$              ;BR IF YES
4443  024734  104012                          ERROR   12              ;ERROR NON-EXISTENT MEM BIT FAILED TO SET
4444  024736                          1$:
4445
4446
4447                                          ;*************************** TEST 66 ****************************
4448                                          ;*NPR NON-EXISTENT MEMORY TEST
4449                                          ;*DO A DATI FROM A NON-EXISTENT ADDRESS
4450                                          ;*VERIFY THAT THE NON-EXISTENT BIT SET IN IBUS REG 11
4451                                          ;:************************************************************
4452
4453                                          ;  TEST 66
4454                                          ;-------------
```

```
                                        ;;*****************************************************************
024736 000004                           TST66:  SCOPE
024740 012737 000066 001202                     MOV     #66,$TSTNM              ; LOAD THE NO. OF THIS TEST
024746 012737 025044 001442                     MOV     #TST67,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                ;R1 CONTAINS BASE KMC11 ADDRESS
024754 104410                                   MSTCLR                          ;MASTER CLEAR KMC11
024756 004537 035540                            JSR     R5,NPRSET               ;LOAD IBUS REGISTERS 0-7
024762 000000                                   0                               ;IN DATA
024764 000000                                   0                               ;OUT DATA
024766 177320                                   177320                          ;IN BA
024770 177320                                   177320                          ;OUT BA
024772 005061 000004                            CLR     4(R1)
024776 104412                                   ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025000 121111                                   121111                          ;CLEAR NON-EXISTENT BIT
025002 012761 000015 000004                     MOV     #15,4(R1)               ;SET NPR REQUEST BITS IN PORT4
025010 104412                                   ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025012 121110                                   121110                          ;MOV IBUS* 4 TO IBUS* 10
025014 000240                                   NOP
025016 104412                                   ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
025020 121225                                   121225                          ;MOV IBUS*11 TO IBUS*5
025022 012705 000001                            MOV     #1,R5                   ;PUT "EXPECTED" IN R5
025026 116104 000005                            MOVB    5(R1),R4                ;PUT "FOUND" IN R4
025032 042704 177776                            BIC     #177776,R4              ;CLEAR UNWANTED BITS
025036 020504                                   CMP     R5,R4                   ;DATA CORRECT?
025040 001401                                   BEQ     1$                      ;BR IF YES
025042 104012                                   ERROR   12                      ;ERROR NON-EXISTENT MEM BIT FAILED TO SET
025044                                  1$:
                                        ;
                                        ;
                                        ;
                                        ;*************************** TEST 67 ****************************
                                        ;*NPR TEST
                                        ;*USING DATO, NPR A BINARY COUNT (0-377 )
                                        ;*FROM MICRO-PROCESSOR TO ALL AVAILABLE MEMORY
                                        ;*****************************************************************
                                        ;
                                        ;  TEST 67
                                        ;-------------
                                        ;*****************************************************************
025044 000004                           TST67:  SCOPE
025046 012737 000067 001202                     MOV     #67,$TSTNM              ; LOAD THE NO. OF THIS TEST
025054 012737 000003 001310                     MOV     #3,$TIMES               ; LOAD ITERATION COUNT
025062 012737 025244 001442                     MOV     #TST70,NEXT             ; POINT TO THE START OF NEXT TEST.
                                                ;R1 CONTAINS BASE KMC11 ADDRESS
025070 104410                                   MSTCLR                          ;MASTER CLEAR KMC11
025072 005037 025242                            CLR     5$                      ;START FLAG AT 0
025076 005000                                   CLR     R0                      ;DATA
025100 012702 037234                            MOV     #CORMAX,R2              ;ADDRESS
025104                                  1$:
025104 010037 025134                            MOV     R0,2$                   ;LOAD DATA
025110 010237 025140                            MOV     R2,4$                   ;LOAD BA
025114 032702 000001                            BIT     #BIT0,R2                ;IS BA ODD?
025120 001402                                   BEQ     .+6                     ;BR IF NO
025122 000337 025134                            SWAB    2$                      ;IF ODD PUT DATA IN HI-BYTE
025126 004537 035540                            JSR     R5,NPRSET               ;LOAD NPR REGISTERS
025132 000000                                   0                               ;IN DATA
025134 000000                           2$:     0                               ;OUT DATA
```

```
4511  025136  000000                          0                       ;IN BA
4512  025140  000000                  4$:     0                       ;OUT BA
4513  025142  105012                          CLRB    (R2)            ;CLEAR MEMORY LOCATION
4514  025144  012761  000221  000004          MOV     #221,4(R1)      ;LOAD PORT4
4515  025152  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4516  025154  121110                          121110                  ;DO THE NPR
4517  025156  000240                          NOP
4518  025160  010005                          MOV     R0,R5           ;PUT "EXPECTED" IN R5
4519  025162  111204                          MOVB    (R2),R4         ;PUT "FOUND" IN R4
4520  025164  120504                          CMPB    R5,R4           ;IS DATA CORRECT?
4521  025166  001401                          BEQ     3$              ;BR IF YES
4522  025170  104021                          ERROR   21              ;ERROR, DATA INCORRECT
4523  025172  104405                  3$:     SCOP1
4524  025174  005200                          INC     R0              ;NEXT CHARACTER
4525  025176  042700  177400                  BIC     #177400,R0      ;USE ONLY LOW BYTE
4526  025202  005737  025242                  TST     5$              ;HAS MAX MEMORY BEEN REACHED YET?
4527  025206  001402                          BEQ     6$              ;BR IF NO
4528  025210  005700                          TST     R0              ;DONE PATTERN?
4529  025212  001412                          BEQ     7$              ;BR IF YES
4530  025214  005202                  6$:     INC     R2              ;INC BA
4531  025216  023702  001466                  CMP     MEMLIM,R2       ;REACHED MEMORY LIMIT YET?
4532  025222  001330                          BNE     1$              ;BR IF NOT
4533  025224  012702  037234                  MOV     #CORMAX,R2      ;RESTART BA AT FIRST ADDRESS
4534  025230  012737  177777  025242          MOV     #-1,5$          ;SET FLAG TO END TEST AT END OF DATA PATTERN
4535  025236  000722                          BR 1$                   ;CONTINUE
4536  025240  104420                  7$:     ADVANCE                 ; ADVANCE LOOP
4537  025242  000000                  5$:     0                       ;THIS LOCATION IS A FLAG,  IT STARTS AT 0,
4538                                                                  ;AND IS SET TO -1 WHEN LAST MEMORY ADDRESS
4539                                                                  ;IS USED, TEST IS THEN ENDED WHEN PATTERN IS FINISHED
4540                                  ;$MEM1
4541                                  ;$MEM0
4542                                  ;$MEM2  1K
4543                                  ;$MEM3  1K
4544
4545
4546                                          ;************************** TEST 70 ****************************
4547                                          ;*ALU C BIT TEST
4548                                          ;*TEST THAT AN ADD OF 377 AND 377 WILL SET THE C BIT
4549                                          ;:************************************************************
4550
4551                                          ;  TEST 70
4552                                          ;-------------
4553                                  ;:************************************************************
4554  025244  000004                  TST70:  SCOPE
4555  025246  012737  000070  001202          MOV     #70,$TSTNM              ; LOAD THE NO. OF THIS TEST
4556  025254  012737  025360  001442          MOV     #TST71,NEXT             ; POINT TO THE START OF NEXT TEST.
4557  025262  012737  025306  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
4558                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4559  025270  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4560  025272  004737  035602                  JSR     PC,MEMLD        ;LOAD MAINMEM DATA
4561  025276  025350                          TDATA                   ;POINTER TO DATA
4562  025300  004737  035636                  JSR     PC,SPLD         ;LOAD SP DATA
4563  025304  025350                          TDATA                   ;POINTER TO DATA
4564  025306                          1$:
4565  025306  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4566  025310  010000                          010000                  ;MAR←0
```

```
4567  025312  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4568  025314  054400                          054400!<0*20>           ;ADD 377 AND 377, TO SET C BIT
4569  025316  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4570  025320  040421                          040401!<1*20>           ;ADD 0 AND 0 AND THE C BIT
4571  025322  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4572  025324  061224                          61224                   ;PUT RESULTS IN PORT4
4573  025326  012705  000001                  MOV     #1,R5           ;PUT "EXPECTED" IN R5
4574  025332  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
4575  025336  120504                          CMPB    R5,R4           ;DATA CORRECT?
4576  025340  001401                          BEQ     2$              ;BR IF YES
4577  025342  104015                          ERROR   15              ;ERROR C BIT NOT SET
4578  025344  104405                  2$:     SCOP1                   ;SW09=1?
4579  025346  104420                          ADVANCE                 ; ADVANCE LOOP
4580  025350     377     000     000  TDATA:  .BYTE   -1,0,0,0,0,0,0,0
4581  025353     000     000     000
4582  025356     000     000
4583                                  .EVEN
4584
4585
4586                                  ;************************** TEST 71 **************************
4587                                  ;*ALU TEST
4588                                  ;*TEST OF ALU FUNCTION SEL B WITH C BIT CLEARED
4589                                  ;*ALU FUNCTION (B)    ^ODE=11
4590                                  ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4591                                  ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
4592                                  ;:**********************************************************
4593
4594                                  ;  TEST 71
4595                                  ;--------------
4596                                  ;;**********************************************************
4597  025360  000004                  TST71:  SCOPE
4598  025362  012737  000071  001202          MOV     #71,$TSTNM              ; LOAD THE NO. OF THIS TEST
4599  025370  012737  025536  001442          MOV     #TST72,NEXT             ; POINT TO THE START OF NEXT TEST.
4600  025376  012737  025430  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
4601                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4602  025404  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4603  025406  005000                          CLR     R0              ;MEM + SP ADDRESS
4604  025410  012702  025526                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
4605  025414  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
4606  025420  035726                          MEMDAT                  ;POINTER TO DATA
4607  025422  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
4608  025426  035736                          SPDAT                   ;POINTER TO DATA
4609  025430  004737  035702          1$:     JSR     PC,CLRC         ;CLEAR C BIT!
4610  025434  042737  000017  025450          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
4611  025442  050037  025450                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
4612  025446  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4613  025450  010000                  2$:     010000                  ;LOAD MAR
4614  025452  042737  000017  025466          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
4615  025460  050037  025466                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
4616  025464  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4617  025466  040620                  3$:     040400!<11*20>          ;BR + SEL B
4618  025470  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4619  025472  061224                          61224                   ;MOVE BR TO PORT4
4620  025474  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
4621  025476  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
4622  025502  120504                          CMPB    R5,R4           ;DATA CORRECT?
```

```
4623  025504  001401                          BEQ     4$              ;BR IF YES
4624  025506  104015                          ERROR   15              ;ALU ERROR
4625  025510  104405                  4$:     SCOP1                   ;SW09=1?
4626  025512  005202                          INC     R2              ;NEXT DATA
4627  025514  005200                          INC     R0              ;NEXT ADDRESS
4628  025516  022700  000010                  CMP     #10,R0          ;DONE YET?
4629  025522  001342                          BNE     1$              ;BR IF NO
4630  025524  104420                          ADVANCE                 ; ADVANCE LOOP
4631  025526     000     377     000  5$:     .BYTE   0,-1,0,-1,125,252,125,252
4632  025531     377     125     252
4633  025534     125     252
4634                                  .EVEN
4635
4636
4637                                          ;*************************** TEST 72 ****************************
4638                                          ;*ALU TEST
4639                                          ;*TEST OF ALU FUNCTION SEL A WITH C BIT CLEARED
4640                                          ;*ALU FUNCTION (A)    CODE=10
4641                                          ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4642                                          ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
4643                                          ;;*************************************************************
4644
4645                                          ;  TEST 72
4646                                          ;-------------
4647                                  ;;*************************************************************
4648  025536  000004                  TST72:  SCOPE
4649  025540  012737  000072  001202          MOV     #72,$TSTNM              ; LOAD THE NO. OF THIS TEST
4650  025546  012737  025714  001442          MOV     #TST73,NEXT             ; POINT TO THE START OF NEXT TEST.
4651  025554  012737  025606  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
4652                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4653  025562  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4654  025564  005000                          CLR     R0              ;MEM + SP ADDRESS
4655  025566  012702  025704                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
4656  025572  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
4657  025576  035726                          MEMDAT                  ;POINTER TO DATA
4658  025600  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
4659  025604  035736                          SPDAT                   ;POINTER TO DATA
4660  025606  004737  035702          1$:     JSR     PC,CLRC         ;CLEAR C BIT!
4661  025612  042737  000017  025626          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
4662  025620  050037  025626                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
4663  025624  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4664  025626  010000                  2$:     010000                  ;LOAD MAR
4665  025630  042737  000017  025644          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
4666  025636  050037  025644                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
4667  025642  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4668  025644  040600                  3$:     040400!<10*20>          ;BR + SEL A
4669  025646  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4670  025650  061224                          61224                   ;MOVE BR TO PORT4
4671  025652  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
4672  025654  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
4673  025660  120504                          CMPB    R5,R4           ;DATA CORRECT?
4674  025662  001401                          BEQ     4$              ;BR IF YES
4675  025664  104015                          ERROR   15              ;ALU ERROR
4676  025666  104405                  4$:     SCOP1                   ;SW09=1?
4677  025670  005202                          INC     R2              ;NEXT DATA
4678  025672  005200                          INC     R0              ;NEXT ADDRESS
```

```
4679  025674  022700  000010                  CMP     #10,R0          ;DONE YET?
4680  025700  001342                          BNE     1$              ;BR IF NO
4681  025702  104420                          ADVANCE                 ; ADVANCE LOOP
4682  025704     000     000     377  5$:     .BYTE   0,0,-1,-1,125,125,252,252
4683  025707     377     125     125
4684  025712     252     252
4685                              .EVEN
4686
4687
4688                                          ;************************** TEST 73 ***********************.*******
4689                                          ;*ALU TEST
4690                                          ;*TEST OF ALU FUNCTION A OR NOTB WITH C BIT CLEARED
4691                                          ;*ALU FUNCTION (A OR NOTB)    CODE=12
4692                                          ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4693                                          ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
4694                                          ;:*****************************************************************
4695
4696                                          ;  TEST 73
4697                                          ;-------------
4698                              .*****************************************************************
4699  025714  000004                  TST73:  SCOPE
4700  025716  012737  000073  001202          MOV     #73,$TSTNM              ; LOAD THE NO. OF THIS TEST
4701  025724  012737  026072  001442          MOV     #TST74,NEXT             ; POINT TO THE START OF NEXT TEST.
4702  025732  012737  025764  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
4703                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4704  025740  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4705  025742  005000                          CLR     R0              ;MEM + SP ADDRESS
4706  025744  012702  026062                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
4707  025750  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
4708  025754  035726                          MEMDAT                  ;POINTER TO DATA
4709  025756  004737  035636                  JSP     PC,SPLD         ;LOAD 8 WORDS OF SP
4710  025762  035736                          SPDAT                   ;POINTER TO DATA
4711  025764  004737  035702          1$:     JSR     PC,CLRC         ;CLEAR C BIT!
4712  025770  042737  000017  026004          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
4713  025776  050037  026004                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
4714  026002  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4715  026004  010000                  2$:     010000                  ;LOAD MAR
4716  026006  042737  000017  026022          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
4717  026014  050037  026022                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
4718  026020  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4719  026022  040640                  3$:     040400!<12*20>          ;BR = A OR NOTB
4720  026024  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4721  026026  061224                          61224                   ;MOVE BR TO PORT4
4722  026030  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
4723  026032  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
4724  026036  120504                          CMPB    R5,R4           ;DATA CORRECT?
4725  026040  001401                          BEQ     4$              ;BR IF YES
4726  026042  104015                          ERROR   15              ;ALU ERROR
4727  026044  104405                  4$:     SCOP1                   ;SW09=1?
4728  026046  005202                          INC     R2              ;NEXT DATA
4729  026050  005200                          INC     R0              ;NEXT ADDRESS
4730  026052  022700  000010                  CMP     #10,R0          ;DONE YET?
4731  026056  001342                          BNE     1$              ;BR IF NO
4732  026060  104420                          ADVANCE                 ; ADVANCE LOOP
4733  026062     377     000     377  5$:     .BYTE   -1,0,-1,-1,-1,125.252,-1
4734  026065     377     377     125
```

```
 4735  026070  252  377
 4736                          .EVEN
 4737
 4738
 4739                          ;*************************** TEST 74 ****************************
 4740                          ;*ALU TEST
 4741                          ;*TEST OF ALU FUNCTION A AND B WITH C BIT CLEARED
 4742                          ;*ALU FUNCTION (A AND B)    CODE=13
 4743                          ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
 4744                          ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
 4745                          ;;************************************************************
 4746
 4747                          ;  TEST 74
 4748                          ;-------------
 4749                          ;;************************************************************
 4750  026072  000004          TST74:  SCOPE
 4751  026074  012737  000074  001202          MOV     #74,$TSTNM              ; LOAD THE NO. OF THIS TEST
 4752  026102  012737  026250  001442          MOV     #TST75,NEXT             ; POINT TO THE START OF NEXT TEST.
 4753  026110  012737  026142  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
 4754                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
 4755  026116  104410                         MSTCLR                  ;MASTER CLEAR KMC11
 4756  026120  005000                         CLR     R0              ;MEM + SP ADDRESS
 4757  026122  012702  026240                 MOV     #5$,R2          ;POINTER TO CORRECT DATA
 4758  026126  004737  035602                 JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
 4759  026132  035726                         MEMDAT                  ;POINTER TO DATA
 4760  026134  004737  035636                 JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
 4761  026140  035736                         SPDAT                   ;POINTER TO DATA
 4762  026142  004737  035702          1$:    JSR     PC,CLRC         ;CLEAR C BIT!
 4763  026146  042737  000017  026162         BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
 4764  026154  050037  026162                 BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
 4765  026160  104412                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4766  026162  010000          2$:    010000                          ;LOAD MAR
 4767  026164  042737  000017  026200         BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
 4768  026172  050037  026200                 BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
 4769  026176  104412                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4770  026200  040660          3$:    040400!<13*20>                  ;BR = A AND B
 4771  026202  104412                         ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
 4772  026204  061224                         61224                   ;MOVE BR TO PORT4
 4773  026206  111205                         MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
 4774  026210  116104  000004                 MOVB    4(R1),R4        ;PUT "FOUND" IN R4
 4775  026214  120504                         CMPB    R5,R4           ;DATA CORRECT?
 4776  026216  001401                         BEQ     4$              ;BR IF YES
 4777  026220  104015                         ERROR   15              ;ALU ERROR
 4778  026222  104405          4$:    SCOP1                           ;SW09=1?
 4779  026224  005202                         INC     R2              ;NEXT DATA
 4780  026226  005200                         INC     R0              ;NEXT ADDRESS
 4781  026230  022700  000010                 CMP     #10,R0          ;DONE YET?
 4782  026234  001342                         BNE     1$              ;BR IF NO
 4783  026236  104420                         ADVANCE                 ; ADVANCE LOOP
 4784  026240     000     000     000  5$:    .BYTE   0,0,0,-1,125,0,0,252
 4785  026243     377     125     000
 4786  026246     000     252
 4787                          .EVEN
 4788
 4789
 4790                          ;*************************** TEST 75 ****************************
```

```
                                        ;*ALU TEST
                                        ;*TEST OF ALU FUNCTION A OR B WITH C BIT CLEARED
                                        ;*ALU FUNCTION (A OR B)    CODE=14
                                        ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
                                        ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
                                        ;;*****************************************************************

                                        ;  TEST 75
                                        ;-------------
                                    ;;*****************************************************************
026250  000004                  TST75:  SCOPE
026252  012737  000075  001202          MOV     #75,$TSTNM              ; LOAD THE NO. OF THIS TEST
026260  012737  026426  001442          MOV     #TST76,NEXT             ; POINT TO THE START OF NEXT TEST.
026266  012737  026320  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.

                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
026274  104410                          MSTCLR                  ;MASTER CLEAR KMC11
026276  005000                          CLR     R0              ;MEM + SP ADDRESS
026300  012702  026416                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
026304  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
026310  035726                          MEMDAT                  ;POINTER TO DATA
026312  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
026316  035736                          SPDAT                   ;POINTER TO DATA
026320  004737  035702          1$:     JSR     PC,CLRC         ;CLEAR C BIT!
026324  042737  000017  026340          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
026332  050037  026340                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
026336  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026340  010000                  2$:     010000                  ;LOAD MAR
026342  042737  000017  026356          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
026350  050037  026356                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
026354  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026356  040700                  3$:     040400!<14*20>          ;BR = A OR B
026360  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
026362  061224                          61224                   ;MOVE BR TO PORT4
026364  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
026366  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
026372  120504                          CMPB    R5,R4           ;DATA CORRECT?
026374  001401                          BEQ     4$              ;BR IF YES
026376  104015                          ERROR   15              ;ALU ERROR
026400  104405                  4$:     SCOPE                   ;SW09=1?
026402  005202                          INC     R2              ;NEXT DATA
026404  005200                          INC     R0              ;NEXT ADDRESS
026406  022700  000010                  CMP     #10,R0          ;DONE YET?
026412  001342                          BNE     1$              ;BR IF NO
026414  104420                          ADVANCE                 ; ADVANCE LOOP
026416     000     377     377  5$:     .BYTE   0,-1,-1,-1,125,-1,-1,252
026421     377     125     377
026424     377     252
                                .EVEN


                                    ;***************************** TEST 76 *****************************
                                    ;*ALU TEST
                                    ;*TEST OF ALU FUNCTION A XOR B WITH C BIT CLEARED
                                    ;*ALU FUNCTION (A XOR B)    CODE=15
                                    ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
                                    ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
```

```
4847                                    ;;*****************************************************************
4848
4849                                    ;  TEST 76
4850                                    ;-------------
4851                                    ;;*****************************************************************
4852  026426  000004                TST76:  SCOPE
4853  026430  012737  000076  001202        MOV     #76,$TSTNM              ; LOAD THE NO. OF THIS TEST
4854  026436  012737  026604  001442        MOV     #TST77,NEXT             ; POINT TO THE START OF NEXT TEST.
4855  026444  012737  026476  001444        MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
4856                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
4857  026452  104410                       MSTCLR                  ;MASTER CLEAR KMC11
4858  026454  005000                       CLR     R0              ;MEM + SP ADDRESS
4859  026456  012702  026574               MOV     #5$,R2          ;POINTER TO CORRECT DATA
4860  026462  004737  035602               JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
4861  026466  035726                       MEMDAT                  ;POINTER TO DATA
4862  026470  004737  035636               JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
4863  026474  035736                       SPDAT                   ;POINTER TO DATA
4864  026476  004737  035702        1$:    JSR     PC,CLRC         ;CLEAR C BIT!
4865  026502  042737  000017  026516       BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
4866  026510  050037  026516               BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
4867  026514  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4868  026516  010000                2$:    010000                  ;LOAD MAR
4869  026520  042737  000017  026534       BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
4870  026526  050037  026534               BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
4971  026532  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4872  026534  040720                3$:    040400!<15*20>          ;BR = A XOR B
4873  026536  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4874  026540  061224                       61224                   ;MOVE BR TO PORT4
4875  026542  111205                       MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
4876  026544  116104  000004               MOVB    4(R1),R4        ;PUT "FOUND" IN R4
4877  026550  120504                       CMPB    R5,R4           ;DATA CORRECT?
4878  026552  001401                       BEQ     4$              ;BR IF YES
4879  026554  104015                       ERROR   15              ;ALU ERROR
4880  026556  104405                4$:    SCOP1                   ;SW09=1?
4881  026560  005202                       INC     R2              ;NEXT DATA
4882  026562  005200                       INC     R0              ;NEXT ADDRESS
4883  026564  022700  000010               CMP     #10,R0          ;DONE YET?
4884  026570  001342                       BNE     1$              ;BR IF NO
4885  026572  104420                       ADVANCE                 ; ADVANCE LOOP
4886  026574     000     377     377  5$:    .BYTE   0,-1,-1,0,0,-1,-1,0
4887  026577     000     000     377
4888  026602     377     000
4889                                    .EVEN
4890
4891
4892                                    ;*************************** TEST 77 ****************************
4893                                    ;*ALU TEST
4894                                    ;*TEST OF ALU FUNCTION ADD WITH C BIT CLEARED
4895                                    ;*ALU FUNCTION (A PLUS B)    CODE=00
4896                                    ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4897                                    ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
4898                                    ;;*****************************************************************
4899
4900                                    ;  TEST 77
4901                                    ;-------------
4902                                    ;;*****************************************************************
```

```
4903  026604  000004                    TST77:  SCOPE
4904  026606  012737  000077  001202            MOV     #77,$TSTNM              ; LOAD THE NO. OF THIS TEST
4905  026614  012737  026762  001442            MOV     #TST100,NEXT            ; POINT TO THE START OF NEXT TEST.
4906  026622  012737  026654  001444            MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
4907                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
4908  026630  104410                            MSTCLR                  ;MASTER CLEAR KMC11
4909  026632  005000                            CLR     R0              ;MEM + SP ADDRESS
4910  026634  012702  026752                    MOV     #5$,R2          ;POINTER TO CORRECT DATA
4911  026640  004737  035602                    JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
4912  026644  035726                            MEMDAT                  ;POINTER TO DATA
4913  026646  004737  035636                    JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
4914  026652  035736                            SPDAT                   ;POINTER TO DATA
4915  026654  004737  035702            1$:     JSR     PC,CLRC         ;CLEAR C BIT!
4916  026660  042737  000017  026674            BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
4917  026666  050037  026674                    BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
4918  026672  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4919  026674  010000                    2$:     010000                  ;LOAD MAR
4920  026676  042737  000017  026712            BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
4921  026704  050037  026712                    BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
4922  026710  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4923  026712  040400                    3$:     040400!<00*20>          ;BR + ADD
4924  026714  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4925  026716  061224                            61224                   ;MOVE BR TO PORT4
4926  026720  111205                            MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
4927  026722  116104  000004                    MOVB    4(R1),R4        ;PUT "FOUND" IN R4
4928  026726  120504                            CMPB    R5,R4           ;DATA CORRECT?
4929  026730  001401                            BEQ     4$              ;BR IF YES
4930  026732  104015                            ERROR   15              ;ALU ERROR
4931  026734  104405                    4$:     SCOPE                   ;SW09=1?
4932  026736  005202                            INC     R2              ;NEXT DATA
4933  026740  005200                            INC     R0              ;NEXT ADDRESS
4934  026742  022700  000010                    CMP     #10,R0          ;DONE YET?
4935  026746  001342                            BNE     1$              ;BR IF NO
4936  026750  104420                            ADVANCE                 ; ADVANCE LOOP
4937  026752     000     377     377    5$:     .BYTE   0,-1,-1,376,252,-1,-1,124
4938  026755     376     252     377
4939  026760     377     124
4940                                    .EVEN
4941
4942
4943                                            ;*************************** TEST 100 ****************************
4944                                            ;*ALU TEST
4945                                            ;*TEST OF ALU FUNCTION 2A W/C WITH C BIT CLEARED
4946                                            ;*ALU FUNCTION (A PLUS A PLUS C)    CODE=6
4947                                            ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4948                                            ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
4949                                            ;;*************************************************************************
4950
4951                                            ;  TEST 100
4952                                            ;-------------
4953                                    ;;*************************************************************
4954  026762  000004                    TST100: SCOPE
4955  026764  012737  000100  001202            MOV     #100,$TSTNM             ; LOAD THE NO. OF THIS TEST
4956  026772  012737  027140  001442            MOV     #TST101,NEXT            ; POINT TO THE START OF NEXT TEST.
4957  027000  012737  027032  001444            MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
4958                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
```

```
4959  027006  104410                          MSTCLR                  ;MASTER CLEAR KMC11
4960  027010  005000                          CLR     R0              ;MEM + SP ADDRESS
4961  027012  012702  027130                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
4962  027016  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
4963  027022  035726                          MEMDAT                  ;POINTER TO DATA
4964  027024  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
4965  027030  035736                          SPDAT                   ;POINTER TO DATA
4966  027032  004737  035702          1$:     JSR     PC,CLRC         ;CLEAR C BIT!
4967  027036  042737  000017  027052          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
4968  027044  050037  027052                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
4969  027050  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4970  027052  010000                  2$:     010000                  ;LOAD MAR
4971  027054  042737  000017  027070          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
4972  027062  050037  027070                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
4973  027066  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4974  027070  040540                  3$:     040400!<6*20>           ;BR + 2A W/C
4975  027072  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
4976  027074  061224                          61224                   ;MOVE BR TO PORT4
4977  027076  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
4978  027100  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
4979  027104  120504                          CMPB    R5,R4           ;DATA CORRECT?
4980  027106  001401                          BEQ     4$              ;BR IF YES
4981  027110  104015                          ERROR   15              ;ALU ERROR
4982  027112  104405                  4$:     SCOPE                   ;SW09=1?
4983  027114  005202                          INC     R2              ;NEXT DATA
4984  027116  005200                          INC     R0              ;NEXT ADDRESS
4985  027120  022700  000010                  CMP     #10,R0          ;DONE YET?
4986  027124  001342                          BNE     1$              ;BR IF NO
4987  027126  104420                          ADVANCE                 ; ADVANCE LOOP
4988  027130     000     000     376  5$:     .BYTE   0,0,376,376,252,252,124,124
4989  027133     376     252     252
4990  027136     124     124
4991                                  .EVEN
4992
4993
4994                                          ;*************************** TEST 101 ****************************
4995                                          ;*ALU TEST
4996                                          ;*TEST OF ALU FUNCTION SUB WITH C BIT CLEARED
4997                                          ;*ALU FUNCTION (A-B)    CODE=16
4998                                          ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
4999                                          ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5000                                          ;;*************************************************************************
5001
5002                                          ;  TEST 101
5003                                          ;--------------
5004                                  ;;*****************************************************************
5005  027140  000004                  TST101: SCOPE
5006  027142  012737  000101  001202          MOV     #101,$TSTNM             ; LOAD THE NO. OF THIS TEST
5007  027150  012737  027316  001442          MOV     #TST102,NEXT            ; POINT TO THE START OF NEXT TEST.
5008  027156  012737  027210  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5009                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5010  027164  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5011  027166  005000                          CLR     R0              ;MEM + SP ADDRESS
5012  027170  012702  027306                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
5013  027174  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5014  027200  035726                          MEMDAT                  ;POINTER TO DATA
```

```
5015  027202  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5016  027206  035736                          SPDAT                   ;POINTER TO DATA
5017  027210  004737  035702          1$:     JSR     PC,CLRC         ;CLEAR C BIT!
5018  027214  042737  000017  027230          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5019  027222  050037  027230                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5020  027226  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5021  027230  010000                  2$:     010000                  ;LOAD MAR
5022  027232  042737  000017  027246          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5023  027240  050037  027246                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5024  027244  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5025  027246  040740                  3$:     040400!<16*20>          ;BR + SUB
5026  027250  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5027  027252  061224                          61224                   ;MOVE BR TO PORT4
5028  027254  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5029  027256  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5030  027262  120504                          CMPB    R5,R4           ;DATA CORRECT?
5031  027264  001401                          BEQ     4$              ;BR IF YES
5032  027266  104015                          ERROR   15              ;ALU ERROR
5033  027270  104405                  4$:     SCOPE                   ;SW09=1?
5034  027272  005202                          INC     R2              ;NEXT DATA
5035  027274  005200                          INC     R0              ;NEXT ADDRESS
5036  027276  022700  000010                  CMP     #10,R0          ;DONE YET?
5037  027302  001342                          BNE     1$              ;BR IF NO
5038  027304  104420                          ADVANCE                 ; ADVANCE LOOP
5039  027306     000     001     377  5$:     .BYTE   0,1,-1,0,0,253,125,0
5040  027311     000     000     253
5041  027314     125     000
5042                                  .EVEN
5043
5044
5045                                  ;*************************** TEST 102 ****************************
5046                                  ;*ALU TEST
5047                                  ;*TEST OF ALU FUNCTION ADD W/C WITH C BIT CLEARED
5048                                  ;*ALU FUNCTION (A PLUS B PLUS C)    CODE=01
5049                                  ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5050                                  ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5051                                  ;*****************************************************************
5052
5053                                  ;  TEST 102
5054                                  ;-------------
5055                              ;;**************************************************************
5056  027316  000004              TST102: SCOPE
5057  027320  012737  000102  001202         MOV     #102,STSTNM             ; LOAD THE NO. OF THIS TEST
5058  027326  012737  027474  001442         MOV     #TST103,NEXT            ; POINT TO THE START OF NEXT TEST.
5059  027334  012737  027366  001444         MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5060                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5061  027342  104410                         MSTCLR                  ;MASTER CLEAR KMC11
5062  027344  005000                         CLR     R0              ;MEM + SP ADDRESS
5063  027346  012702  027464                 MOV     #5$,R2          ;POINTER TO CORRECT DATA
5064  027352  004737  035602                 JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5065  027356  035726                         MEMDAT                  ;POINTER TO DATA
5066  027360  004737  035636                 JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5067  027364  035736                         SPDAT                   ;POINTER TO DATA
5068  027366  004737  035702          1$:     JSR     PC,CLRC         ;CLEAR C BIT!
5069  027372  042737  000017  027406          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5070  027400  050037  027406                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
```

```
5071  027404  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5072  027406  010000                  2$:     010000                  ;LOAD MAR
5073  027410  042737  000017  027424          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5074  027416  050037  027424                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5075  027422  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5076  027424  040420                  3$:     040400!<01*20>          ;BR + ADD W/C
5077  027426  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5078  027430  061224                          61224                   ;MOVE BR TO PORT4
5079  027432  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5080  027434  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5081  027440  120504                          CMPB    R5,R4           ;DATA CORRECT?
5082  027442  001401                          BEQ     4$              ;BR IF YES
5083  027444  104015                          ERROR   15              ;ALU ERROR
5084  027446  104405                  4$:     SCOP1                   ;SW09=1?
5085  027450  005202                          INC     R2              ;NEXT DATA
5086  027452  005200                          INC     R0              ;NEXT ADDRESS
5087  027454  022700  000010                  CMP     #10,R0          ;DONE YET?
5088  027460  001342                          BNE     1$              ;BR IF NO
5089  027462  104420                          ADVANCE                 ; ADVANCE LOOP
5090  027464     000     377     377  5$:     .BYTE   0,-1,-1,376,252,-1,-1,124
5091  027467     376     252     377
5092  027472     377     124
5093                                  .EVEN
5094
5095
5096                                  ;************************** TEST 103 **************************
5097                                  ;*ALU TEST
5098                                  ;*TEST OF ALU FUNCTION SUB W/C WITH C BIT CLEARED
5099                                  ;*ALU FUNCTION (A-B-C)    CODE=2
5100                                  ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5101                                  ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5102                                  ;:************************************************************
5103
5104                                  ;  TEST 103
5105                                  ;-------------
5106                                  ;:************************************************************
5107  027474  000004                  TST103: SCOPE
5108  027476  012737  000103  001202          MOV     #103,$TSTNM             ; LOAD THE NO. OF THIS TEST
5109  027504  012737  027652  001442          MOV     #TST104,NEXT            ; POINT TO THE START OF NEXT TEST.
5110  027512  012737  027544  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5111                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5112  027520  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5113  027522  005000                          CLR     R0              ;MEM + SP ADDRESS
5114  027524  012702  027642                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
5115  027530  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5116  027534  035726                          MEMDAT                  ;POINTER TO DATA
5117  027536  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5118  027542  035736                          SPDAT                   ;POINTER TO DATA
5119  027544  004737  035702          1$:     JSR     PC,CLRC         ;CLEAR C BIT!
5120  027550  042737  000017  027564          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5121  027556  050037  027564                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5122  027562  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5123  027564  010000                  2$:     010000                  ;LOAD MAR
5124  027566  042737  000017  027602          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5125  027574  050037  027602                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5126  027600  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
```

```
5127  027602  040440                  3$:     040400!<2*20>           ;BR + SUB W/C
5128  027604  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5129  027606  061224                          61224                   ;MOVE BR TO PORT4
5130  027610  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5131  027612  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5132  027616  120504                          CMPB    R5,R4           ;DATA CORRECT?
5133  027620  001401                          BEQ     4$              ;BR IF YES
5134  027622  104015                          ERROR   15              ;ALU ERROR
5135  027624  104405                  4$:     SCOPE                   ;SW09=1?
5136  027626  005202                          INC     R2              ;NEXT DATA
5137  027630  005200                          INC     R0              ;NEXT ADDRESS
5138  027632  022700  000010                  CMP     #10,R0          ;DONE YET?
5139  027636  001342                          BNE     1$              ;BR IF NO
5140  027640  104420                          ADVANCE                 ; ADVANCE LOOP
5141  027642     377     000     376  5$:     .BYTE   -1,0,376,-1,-1,252,124,-1
5142  027645     377     377     252
5143  027650     124     377
5144                                  .EVEN
5145
5146
5147                                  ;************************** TEST 104 **************************
5148                                  ;*ALU TEST
5149                                  ;*TEST OF ALU FUNCTION INC A WITH C BIT CLEARED
5150                                  ;*ALU FUNCTION (A PLUS 1)    CODE=3
5151                                  ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5152                                  ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5153                                  ;:**********************************************************
5154
5155                                  ;  TEST 104
5156                                  ;-------------
5157                                  ;:**********************************************************
5158  027652  000004                  TST104: SCOPE
5159  027654  012737  000104  001202          MOV     #104,$TSTNM             ; LOAD THE NO. OF THIS TEST
5160  027662  012737  030030  001442          MOV     #TST105,NEXT            ; POINT TO THE START OF NEXT TEST.
5161  027670  012737  027722  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5162                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5163  027676  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5164  027700  005000                          CLR     R0              ;MEM + SP ADDRESS
5165  027702  012702  030020                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
5166  027706  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5167  027712  035726                          MEMDAT                  ;POINTER TO DATA
5168  027714  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5169  027720  035736                          SPDAT                   ;POINTER TO DATA
5170  027722  004737  035702          1$:     JSR     PC,CLRC         ;CLEAR C BIT!
5171  027726  042737  000017  027742          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5172  027734  050037  027742                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5173  027740  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5174  027742  010000                  2$:     010000                  ;LOAD MAR
5175  027744  042737  000017  027760          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5176  027752  050037  027760                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5177  027756  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5178  027760  040460                  3$:     040400!<3*20>           ;BR + INC A
5179  027762  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5180  027764  061224                          61224                   ;MOVE BR TO PORT4
5181  027766  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5182  027770  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
```

```
5183  027774  120504                          CMPB    R5,R4           ;DATA CORRECT?
5184  027776  001401                          BEQ     4$              ;BR IF YES
5185  030000  104015                          ERROR   15              ;ALU ERROR
5186  030002  104405                  4$:     SCOP1                   ;SW09=1?
5187  030004  005202                          INC     R2              ;NEXT DATA
5188  030006  005200                          INC     R0              ;NEXT ADDRESS
5189  030010  022700  000010                  CMP     #10,R0          ;DONE YET?
5190  030014  001342                          BNE     1$              ;BR IF NO
5191  030016  104420                          ADVANCE                 ; ADVANCE LOOP
5192  030020     001     001     000  5$:     .BYTE   1,1,0,0,126,126,253,253
5193  030023     000     126     126
5194  030026     253     253
5195                                  .EVEN
5196
5197
5198                                          ;*************************** TEST 105 ****************************
5199                                          ;*ALU TEST
5200                                          ;*TEST OF ALU FUNCTION 2A WITH C BIT CLEARED
5201                                          ;*ALU FUNCTION (A PLUS A)    CODE=5
5202                                          ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5203                                          ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5204                                          ;;*************************************************************************
5205
5206                                          ;  TEST 105
5207                                          ;-------------
5208                                  ;;*****************************************************************
5209  030030  000004                  TST105: SCOPE
5210  030032  012737  000105  001202          MOV     #105,$TSTNM             ; LOAD THE NO. OF THIS TEST
5211  030040  012737  030206  001442          MOV     #TST106,NEXT            ; POINT TO THE START OF NEXT TEST.
5212  030046  012737  030100  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5213                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5214  030054  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5215  030056  005000                          CLR     R0              ;MEM + SP ADDRESS
5216  030060  012702  030176                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
5217  030064  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5218  030070  035726                          MEMDAT                  ;POINTER TO DATA
5219  030072  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5220  030076  035736                          SPDAT                   ;POINTER TO DATA
5221  030100  004737  035702          1$:     JSR     PC,CLRC         ;CLEAR C BIT!
5222  030104  042737  000017  030120          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5223  030112  050037  030120                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5224  030116  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5225  030120  010000                  2$:     010000                  ;LOAD MAR
5226  030122  042737  000017  030136          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5227  030130  050037  030136                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5228  030134  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5229  030136  040520                  3$:     040400!<5*20>           ;BR + 2A
5230  030140  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5231  030142  061224                          61224                   ;MOVE BR TO PORT4
5232  030144  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5233  030146  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5234  030152  120504                          CMPB    R5,R4           ;DATA CORRECT?
5235  030154  001401                          BEQ     4$              ;BR IF YES
5236  030156  104015                          ERROR   15              ;ALU ERROR
5237  030160  104405                  4$:     SCOP1                   ;SW09=1?
5238  030162  005202                          INC     R2              ;NEXT DATA
```

```
5239  030164  005200                          INC     R0              ;NEXT ADDRESS
5240  030166  022700  000010                  CMP     #10,R0          ;DONE YET?
5241  030172  001342                          BNE     1$              ;BR IF NO
5242  030174  104420                          ADVANCE                 ; ADVANCE LOOP
5243  030176     000     000     376  5$:     .BYTE   0,0,376,376,252,252,124,124
5244  030201     376     252     252
5245  030204     124     124
5246                              .EVEN
5247
5248
5249                                      ;*************************** TEST 106 ****************************
5250                                      ;*ALU TEST
5251                                      ;*TEST OF ALU FUNCTION A PLUS C WITH C BIT CLEARED
5252                                      ;*ALU FUNCTION (A PLUS C)    CODE=4
5253                                      ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5254                                      ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5255                                      ;:*************************************************************
5256
5257                                      ;  TEST 106
5258                                      ;-------------
5259                              ;:*****************************************************************
5260  030206  000004                  TST106: SCOPE
5261  030210  012737  000106  001202          MOV     #106,$TSTNM             ; LOAD THE NO. OF THIS TEST
5262  030216  012737  030364  001442          MOV     #TST107,NEXT            ; POINT TO THE START OF NEXT TEST.
5263  030224  012737  030256  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5264                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5265  030232  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5266  030234  005000                          CLR     R0              ;MEM + SP ADDRESS
5267  030236  012702  030354                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
5268  030242  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5269  030246  035726                          MEMDAT                  ;POINTER TO DATA
5270  030250  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5271  030254  035736                          SPDAT                   ;POINTER TO DATA
5272  030256  004737  035702          1$:     JSR     PC,CLRC         ;CLEAR C BIT!
5273  030262  042737  000017  030276          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5274  030270  050037  030276                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5275  030274  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5276  030276  010000                  2$:     010000                  ;LOAD MAR
5277  030300  042737  000017  030314          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5278  030306  050037  030314                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5279  030312  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5280  030314  040500                  3$:     040400!<4*20>           ;BR + A PLUS C
5281  030316  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5282  030320  061224                          61224                   ;MOVE BR TO PORT4
5283  030322  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5284  030324  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5285  030330  120504                          CMPB    R5,R4           ;DATA CORRECT?
5286  030332  001401                          BEQ     4$              ;BR IF YES
5287  030334  104015                          ERROR   15              ;ALU ERROR
5288  030336  104405                  4$:     SCOP1                   ;SW09=1?
5289  030340  005202                          INC     R2              ;NEXT DATA
5290  030342  005200                          INC     R0              ;NEXT ADDRESS
5291  030344  022700  000010                  CMP     #10,R0          ;DONE YET?
5292  030350  001342                          BNE     1$              ;BR IF NO
5293  030352  104420                          ADVANCE                 ; ADVANCE LOOP
5294  030354     000     000     377  5$:     .BYTE   0,0,-1,-1,125,125,252,252
```

```
5295  030357  377     125     125
5296  030362  252     252
5297                          .EVEN
5298
5299
5300                          ;*************************** TEST 107 ****************************
5301                          ;*ALU TEST
5302                          ;*TEST OF ALU FUNCTION 2'S COMP SUB WITH C BIT CLEARED
5303                          ;*ALU FUNCTION (A-B-1)    CODE=17
5304                          ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5305                          ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5306                          ;:*************************************************************
5307
5308                          ;  TEST 107
5309                          ;-------------
5310                          ;;*************************************************************
5311  030364  000004          TST107: SCOPE
5312  030366  012737  000107  001202          MOV     #107,$TSTNM             ; LOAD THE NO. OF THIS TEST
5313  030374  012737  030542  001442          MOV     #TST110,NEXT            ; POINT TO THE START OF NEXT TEST.
5314  030402  012737  030434  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5315                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5316  030410  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5317  030412  005000                          CLR     R0              ;MEM + SP ADDRESS
5318  030414  012702  030532                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
5319  030420  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5320  030424  035726                          MEMDAT                  ;POINTER TO DATA
5321  030426  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5322  030432  035736                          SPDAT                   ;POINTER TO DATA
5323  030434  004737  035702          1$:     JSR     PC,CLRC         ;CLEAR C BIT!
5324  030440  042737  000017  030454          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5325  030446  050037  030454                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5326  030452  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5327  030454  010000                  2$:     010000                  ;LOAD MAR
5328  030456  042737  000017  030472          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5329  030464  050037  030472                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5330  030470  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5331  030472  040760                  3$:     040400!<17*20>          ;BP + 2'S COMP SUB
5332  030474  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5333  030476  061224                          61224                   ;MOVE BR TO PORT4
5334  030500  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5335  030502  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5336  030506  120504                          CMPB    R5,R4           ;DATA CORRECT?
5337  030510  001401                          BEQ     4$              ;BR IF YES
5338  030512  104015                          ERROR   15              ;ALU ERROR
5339  030514  104405                  4$:     SCOP1                   ;SW09=1?
5340  030516  005202                          INC     R2              ;NEXT DATA
5341  030520  005200                          INC     R0              ;NEXT ADDRESS
5342  030522  022700  000010                  CMP     #10,R0          ;DONE YET?
5343  030526  001342                          BNE     1$              ;BR IF NO
5344  030530  104420                          ADVANCE                 ; ADVANCE LOOP
5345  030532    377     000     376   5$:     .BYTE   -1,0,376,-1,-1,252,124,-1
5346  030535    377     377     252
5347  030540    124     377
5348                          .EVEN
5349
5350
```

```
5351                                    ;*************************** TEST 110 ****************************
5352                                    ;*ALU TEST
5353                                    ;*TEST OF ALU FUNCTION DEC A WITH C BIT CLEARED
5354                                    ;*ALU FUNCTION (A-1)    CODE=7
5355                                    ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5356                                    ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5357                                    ;:*************************************************************************
5358
5359                                    ;  TEST 110
5360                                    ;-------------
5361                                    ;:*************************************************************************
5362  030542  000004                TST110: SCOPE
5363  030544  012737  000110  001202        MOV     #110,$TSTNM             ; LOAD THE NO. OF THIS TEST
5364  030552  012737  030720  001442        MOV     #TST111,NEXT            ; POINT TO THE START OF NEXT TEST.
5365  030560  012737  030612  001444        MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5366                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5367  030566  104410                        MSTCLR                  ;MASTER CLEAR KMC11
5368  030570  005000                        CLR     R0              ;MEM + SP ADDRESS
5369  030572  012702  030710                MOV     #5$,R2          ;POINTER TO CORRECT DATA
5370  030576  004737  035602                JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5371  030602  035726                        MEMDAT                  ;POINTER TO DATA
5372  030604  004737  035636                JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5373  030610  035736                        SPDAT                   ;POINTER TO DATA
5374  030612  004737  035702          1$:   JSR     PC,CLRC         ;CLEAR C BIT!
5375  030616  042737  000017  030632        BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5376  030624  050037  030632                BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5377  030630  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5378  030632  010000                2$:     010000                  ;LOAD MAR
5379  030634  042737  000017  030650        BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5380  030642  050037  030650                BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5381  030646  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5382  030650  040560                3$:     040400!(7*20)           ;BR = DEC A
5383  030652  104412                        ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5384  030654  061224                        61224                   ;MOVE BR TO PORT4
5385  030656  111205                        MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5386  030660  116104  000004                MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5387  030664  120504                        CMPB    R5,R4           ;DATA CORRECT?
5388  030666  001401                        BEQ     4$              ;BR IF YES
5389  030670  104015                        ERROR   15              ;ALU ERROR
5390  030672  104405                4$:     SCOPE1                  ;SW09=1?
5391  030674  005202                        INC     R2              ;NEXT DATA
5392  030676  005200                        INC     R0              ;NEXT ADDRESS
5393  030700  022700  000010                CMP     #10,R0          ;DONE YET?
5394  030704  001342                        BNE     1$              ;BR IF NO
5395  030706  104420                        ADVANCE                 ; ADVANCE LOOP
5396  030710     377     377     376  5$:   .BYTE   -1,-1,376,376,124,124,251,251
5397  030713     376     124     124
5398  030716     251     251
5399                                  .EVEN
5400
5401
5402                                    ;*************************** TEST 111 ****************************
5403                                    ;*ALU TEST
5404                                    ;*TEST OF ALU FUNCTION SEL B WITH C BIT SET
5405                                    ;*ALU FUNCTION (B)    CODE=11
5406                                    ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
```

```
5407                                    ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5408                                    ;:*****************************************************************
5409
5410                                    ;  TEST 111
5411                                    ;-------------
5412                                    ;:*****************************************************************
5413  030720  000004                    TST111: SCOPE
5414  030722  012737  000111  001202            MOV     #111,$TSTNM             ; LOAD THE NO. OF THIS TEST
5415  030730  012737  031076  001442            MOV     #TST!12,NEXT            ; POINT TO THE START OF NEXT TEST.
5416  030736  012737  030770  001444            MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5417                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
5418  030744  104410                            MSTCLR                  ;MASTER CLEAR KMC11
5419  030746  005000                            CLR     R0              ;MEM + SP ADDRESS
5420  030750  012702  031066                    MOV     #5$,R2          ;POINTER TO CORRECT DATA
5421  030754  004737  035602                    JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5422  030760  035726                            MEMDAT                  ;POINTER TO DATA
5423  030762  004737  035636                    JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5424  030766  035736                            SPDAT                   ;POINTER TO DATA
5425  030770  004737  035714            1$:     JSR     PC,SETC         ;SET C BIT!
5426  030774  042737  000017  031010            BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5427  031002  050037  031010                    BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5428  031006  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5429  031010  010000            2$:             010000                  ;LOAD MAR
5430  031012  042737  000017  031026            BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5431  031020  050037  031026                    BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5432  031024  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5433  031026  040620            3$:             040400!<11*20>          ;BR + SEL B
5434  031030  104412                            ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5435  031032  061224                            61224                   ;MOVE BR TO PORT4
5436  031034  111205                            MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5437  031036  116104  000004                    MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5438  031042  120504                            CMPB    R5,R4           ;DATA CORRECT?
5439  031044  001401                            BEQ     4$              ;BR IF YES
5440  031046  104015                            ERROR   15              ;ALU ERROR
5441  031050  104405            4$:             SCOPE1                  ;SW09=1?
5442  031052  005202                            INC     R2              ;NEXT DATA
5443  031054  005200                            INC     R0              ;NEXT ADDRESS
5444  031056  022700  000010                    CMP     #10,R0          ;DONE YET?
5445  031062  001342                            BNE     1$              ;BR IF NO
5446  031064  104420                            ADVANCE                 ; ADVANCE LOOP
5447  031066     000     377     000    5$:     .BYTE   0,-1,0,-1,125,252,125,252
5448  031071     377     125     252
5449  031074     125     252
5450                                    .EVEN
5451
5452
5453                                    ;*************************** TEST 112 ****************************
5454                                    ;*ALU TEST
5455                                    ;*TEST OF ALU FUNCTION SEL A WITH C BIT SET
5456                                    ;*ALU FUNCTION (A)    CODE=10
5457                                    ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5458                                    ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5459                                    ;:*****************************************************************
5460
5461                                    ;  TEST 112
5462                                    ;-------------
```

```
5463                                    ;;*****************************************************************
5464  031076  000004                    TST112: SCOPE
5465  031100  012737  000112  001202            MOV     #112,$TSTNM             ; LOAD THE NO. OF THIS TEST
5466  031106  012737  031254  001442            MOV     #TST113,NEXT            ; POINT TO THE START OF NEXT TEST.
5467  031114  012737  031146  001444            MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5468                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
5469  031122  104410                           MSTCLR                  ;MASTER CLEAR KMC11
5470  031124  005000                           CLR     R0              ;MEM -> SP ADDRESS
5471  031126  012702  031244                   MOV     #5$,R2          ;POINTER TO CORRECT DATA
5472  031132  004737  035602                   JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5473  031136  035726                           MEMDAT                  ;POINTER TO DATA
5474  031140  004737  035636                   JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5475  031144  035736                           SPDAT                   ;POINTER TO DATA
5476  031146  004737  035714           1$:     JSR     PC,SETC         ;SET C BIT!
5477  031152  042737  000017  031166           BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5478  031160  050037  031166                   BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5479  031164  104412                           ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5480  031166  010000                   2$:     010000                  ;LOAD MAR
5481  031170  042737  000017  031204           BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5482  031176  050037  031204                   BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5483  031202  104412                           ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5484  031204  040600                   3$:     040400!<10*20>          ;BR + SEL A
5485  031206  104412                           ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5486  031210  061224                           61224                   ;MOVE BR TO PORT4
5487  031212  111205                           MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5488  031214  116104  000004                   MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5489  031220  120504                           CMPB    R5,R4           ;DATA CORRECT?
5490  031222  001401                           BEQ     4$              ;BR IF YES
5491  031224  104015                           ERROR   15              ;ALU ERROR
5492  031226  104405                   4$:     SCOP1                   ;SW09=1?
5493  031230  005202                           INC     R2              ;NEXT DATA
5494  031232  005200                           INC     R0              ;NEXT ADDRESS
5495  031234  022700  000010                   CMP     #10,R0          ;DONE YET?
5496  031240  001342                           BNE     1$              ;BR IF NO
5497  031242  104420                           ADVANCE                 ; ADVANCE LOOP
5498  031244     000      000      377   5$:     .BYTE   0,0,-1,-1,125,125,252,252
5499  031247     377      125      125
5500  031252     252      252
5501                                    .EVEN
5502
5503
5504                                            ;************************** TEST 113 **************************
5505                                            ;*ALU TEST
5506                                            ;*TEST OF ALU FUNCTION A OR NOTB WITH C BIT SET
5507                                            ;*ALU FUNCTION (A OR NOTB)    CODE=12
5508                                            ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5509                                            ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5510                                            ;;*************************************************************
5511
5512                                            ;  TEST 113
5513                                            ;-------------
5514                                    ;;*****************************************************************
5515  031254  000004                    TST113: SCOPE
5516  031256  012737  000113  001202            MOV     #113,$TSTNM             ; LOAD THE NO. OF THIS TEST
5517  031264  012737  031432  001442            MOV     #TST114,NEXT            ; POINT TO THE START OF NEXT TEST.
5518  031272  012737  031324  001444            MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
```

```
5519
5520  031300  104410                          MSTCLR                  ;R1 CONTAINS BASE KMC11 ADDRESS
5521  031302  005000                          CLR     R0              ;MASTER CLEAR KMC11
5522  031304  012702  031422                  MOV     #5$,R2          ;MEM + SP ADDRESS
5523  031310  004737  035602                  JSR     PC MEMLD        ;POINTER TO CORRECT DATA
5524  031314  035726                          MEMDAT                  ;LOAD 8 WORDS OF MAIN MEMORY
5525  031316  004737  035636                  JSR     PC,SPLD         ;POINTER TO DATA
5526  031322  035736                          SPDAT                   ;LOAD 8 WORDS OF SP
5527  031324  004737  035714          1$:     JSR     PC,SETC         ;SET C BIT!
5528  031330  042737  000017  031344          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5529  031336  050037  031344                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5530  031342  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5531  031344  010000                  2$:     010000                  ;LOAD MAR
5532  031346  042737  000017  031362          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5533  031354  050037  031362                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5534  031360  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5535  031362  040640                  3$:     040400!<12*20>          ;BR + A OR NOTB
5536  031364  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5537  031366  061224                          61224                   ;MOVE BR TO PORT4
5538  031370  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5539  031372  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5540  031376  120504                          CMPB    R5,R4           ;DATA CORRECT?
5541  031400  001401                          BEQ     4$              ;BR IF YES
5542  031402  104015                          ERROR   15              ;ALU ERROR
5543  031404  104405                  4$:     SCOP1                   ;SW09=1?
5544  031406  005202                          INC     R2              ;NEXT DATA
5545  031410  005200                          INC     R0              ;NEXT ADDRESS
5546  031412  022700  000010                  CMP     #10,R0          ;DONE YET?
5547  031416  001342                          BNE     1$              ;BR IF NO
5548  031420  104420                          ADVANCE                 ; ADVANCE LOOP
5549  031422     377     000     377  5$:     .BYTE   -1,0,-1,-1,-1,125,252,-1
5550  031425     377     377     125
5551  031430     252     377
5552                                  .EVEN
5553
5554
5555                                          ;*************************** TEST 114 ****************************
5556                                          ;*ALU TEST
5557                                          ;*TEST OF ALU FUNCTION A AND B WITH C BIT SET
5558                                          ;*ALU FUNCTION (A AND B)    CODE=13
5559                                          ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5560                                          ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5561                                          ;:*************************************************************
5562
5563                                          ;  TEST 114
5564                                          ;-------------
5565                                  ;:*************************************************************
5566  031432  000004                  TST114: SCOPE
5567  031434  012737  000114  001202          MOV     #114,$TSTNM             ; LOAD THE NO. OF THIS TEST
5568  031442  012737  031610  001442          MOV     #TST115,NEXT            ; POINT TO THE START OF NEXT TEST.
5569  031450  012737  031502  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5570                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5571  031456  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5572  031460  005000                          CLR     R0              ;MEM + SP ADDRESS
5573  031462  012702  031600                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
5574  031466  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
```

```
5575  031472  035726                          MEMDAT                  ;POINTER TO DATA
5576  031474  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5577  031500  035736                          SPDAT                   ;POINTER TO DATA
5578  031502  004737  035714          1$:     JSR     PC,SETC         ;SET C BIT!
5579  031506  042737  000017  031522          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5580  031514  050037  031522                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5581  031520  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5582  031522  010000                  2$:     010000                  ;LOAD MAR
5583  031524  042737  000017  031540          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5584  031532  050037  031540                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5585  031536  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5586  031540  040660                  3$:     040400!<13*20>          ;BR = A AND B
5587  031542  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5588  031544  061224                          61224                   ;MOVE BR TO PORT4
5589  031546  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5590  031550  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5591  031554  120504                          CMPB    R5,R4           ;DATA CORRECT?
5592  031556  001401                          BEQ     4$              ;BR IF YES
5593  031560  104015                          ERROR   15              ;ALU ERROR
5594  031562  104405                  4$:     SCOPE                   ;SW09=1?
5595  031564  005202                          INC     R2              ;NEXT DATA
5596  031566  005200                          INC     R0              ;NEXT ADDRESS
5597  031570  022700  000010                  CMP     #10,R0          ;DONE YET?
5598  031574  001342                          BNE     1$              ;BR IF NO
5599  031576  104420                          ADVANCE                 ; ADVANCE LOOP
5600  031600     000     000     000  5$:     .BYTE   0,0,0,-1,125,0,0,252
5601  031603     377     125     000
5602  031606     000     252
5603                                  .EVEN
5604
5605
5606                                          ;************************** TEST 115 **************************
5607                                          ;*ALU TEST
5608                                          ;*TEST OF ALU FUNCTION A OR B WITH C BIT SET
5609                                          ;*ALU FUNCTION (A OR B)    CODE=14
5610                                          ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5611                                          ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5612                                          ;;**********************************************************************
5613
5614                                          ;  TEST 115
5615                                          ;-------------
5616                                  ;;**********************************************************************
5617  031610  000004                  TST115: SCOPE
5618  031612  012737  000115  001202          MOV     #115,$TSTNM             ; LOAD THE NO. OF THIS TEST
5619  031620  012737  031766  001442          MOV     #TST116,NEXT            ; POINT TO THE START OF NEXT TEST.
5620  031626  012737  031660  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5621                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5622  031634  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5623  031636  005000                          CLR     R0              ;MEM + SP ADDRESS
5624  031640  012702  031756                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
5625  031644  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5626  031650  035726                          MEMDAT                  ;POINTER TO DATA
5627  031652  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5628  031656  035736                          SPDAT                   ;POINTER TO DATA
5629  031660  004737  035714          1$:     JSR     PC,SETC         ;SET C BIT!
5630  031664  042737  000017  031700          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
```

```
5631  031672  050037  031700                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5632  031676  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5633  031700  010000                  2$:     010000                  ;LOAD MAR
5634  031702  042737  000017  031716          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5635  031710  050037  031716                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5636  031714  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5637  031716  040700                  3$:     040400!<14*20>          ;BR = A OR B
5638  031720  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5639  031722  061224                          61224                   ;MOVE BR TO PORT4
5640  031724  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5641  031726  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5642  031732  120504                          CMPB    R5,R4           ;DATA CORRECT?
5643  031734  001401                          BEQ     4$              ;BR IF YES
5644  031736  104015                          ERROR   15              ;ALU ERROR
5645  031740  104405                  4$:     SCOP1                   ;SW09=1?
5646  031742  005202                          INC     R2              ;NEXT DATA
5647  031744  005200                          INC     R0              ;NEXT ADDRESS
5648  031746  022700  000010                  CMP     #10,R0          ;DONE YET?
5649  031752  001342                          BNE     1$              ;BR IF NO
5650  031754  104420                          ADVANCE                 ; ADVANCE LOOP
5651  031756     000     377     377  5$:     .BYTE   0,-1,-1,-1,125,-1,-1,252
5652  031761     377     125     377
5653  031764     377     252
5654                                  .EVEN
5655
5656
5657                                  ;*************************** TEST 116 ****************************
5658                                  ;*ALU TEST
5659                                  ;*TEST OF ALU FUNCTION A XOR B WITH C BIT SET
5660                                  ;*ALU FUNCTION (A XOR B)    CODE=15
5661                                  ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5662                                  ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5663                                  ;;*************************************************************
5664
5665                                  ;  TEST 116
5666                                  ;-------------
5667                                  ;;*************************************************************
5668  031766  000004                  TST116: SCOPE
5669  031770  012737  000116  001202          MOV     #116,$TSTNM             ; LOAD THE NO. OF THIS TEST
5670  031776  012737  032144  001442          MOV     #TST117,NEXT            ; POINT TO THE START OF NEXT TEST.
5671  032004  012737  032036  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5672                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5673  032012  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5674  032014  005000                          CLR     R0              ;MEM + SP ADDRESS
5675  032016  012702  032134                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
5676  032022  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5677  032026  035726                          MEMDAT                  ;POINTER TO DATA
5678  032030  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5679  032034  035736                          SPDAT                   ;POINTER TO DATA
5680  032036  004737  035714          1$:     JSR     PC,SETC         ;SET C BIT!
5681  032042  042737  000017  032056          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5682  032050  050037  032056                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5683  032054  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5684  032056  010000                  2$:     010000                  ;LOAD MAR
5685  032060  042737  000017  032074          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5686  032066  050037  032074                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
```

```
5687  032072  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5688  032074  040720                  3$:     040400!<15*20>          ;BR = A XOR B
5689  032076  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5690  032100  061224                          61224                   ;MOVE BR TO PORT4
5691  032102  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5692  032104  116104  000004                 MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5693  032110  120504                          CMPB    R5,R4           ;DATA CORRECT?
5694  032112  001401                          BEQ     4$              ;BR IF YES
5695  032114  104015                          ERROR   15              ;ALU ERROR
5696  032116  104405                  4$:     SCOPE                   ;SW09=1?
5697  032120  005202                          INC     R2              ;NEXT DATA
5698  032122  005200                          INC     R0              ;NEXT ADDRESS
5699  032124  022700  000010                 CMP     #10,R0          ;DONE YET?
5700  032130  001342                          BNE     1$              ;BR IF NO
5701  032132  104420                          ADVANCE                 ; ADVANCE LOOP
5702  032134     000     377     377  5$:     .BYTE   0,-1,-1,0,0,-1,-1,0
5703  032137     000     000     377
5704  032142     377     000
5705                                  .EVEN
5706
5707
5708                                  ;*************************** TEST 117 ****************************
5709                                  ;*ALU TEST
5710                                  ;*TEST OF ALU FUNCTION ADD WITH C BIT SET
5711                                  ;*ALU FUNCTION (A PLUS B)    CODE=00
5712                                  ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5713                                  ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5714                                  ;:*************************************************************
5715
5716                                  ;  TEST 117
5717                                  ;-------------
5718                                  ;:*************************************************************
5719  032144  000004          TST117: SCOPE
5720  032146  012737  000117  001202          MOV     #117,$TSTNM             ; LOAD THE NO. OF THIS TEST
5721  032154  012737  032322  001442          MOV     #TST120,NEXT            ; POINT TO THE START OF NEXT TEST.
5722  032162  012737  032214  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5723                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5724  032170  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5725  032172  005000                          CLR     R0              ;MEM + SP ADDRESS
5726  032174  012702  032312                 MOV     #5$,R2          ;POINTER TO CORRECT DATA
5727  032200  004737  035602                 JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5728  032204  035726                          MEMDAT                  ;POINTER TO DATA
5729  032206  004737  035636                 JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5730  032212  035736                          SPDAT                   ;POINTER TO DATA
5731  032214  004737  035714          1$:     JSR     PC,SETC         ;SET C BIT!
5732  032220  042737  000017  032234         BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5733  032226  050037  032234                 BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5734  032232  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5735  032234  010000                  2$:     010000                  ;LOAD MAR
5736  032236  042737  000017  032252         BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5737  032244  050037  032252                 BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5738  032250  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5739  032252  040400                  3$:     040400!<00*20>          ;BR = ADD
5740  032254  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5741  032256  061224                          61224                   ;MOVE BR TO PORT4
5742  032260  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
```

```
5743  032262  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5744  032266  120504                          CMPB    R5,R4           ;DATA CORRECT?
5745  032270  001401                          BEQ     4$              ;BR IF YES
5746  032272  104015                          ERROR   15              ;ALU ERROR
5747  032274  104405                  4$:     SCOPE                   ;SW09=1?
5748  032276  005202                          INC     R2              ;NEXT DATA
5749  032300  005200                          INC     R0              ;NEXT ADDRESS
5750  032302  022700  000010                  CMP     #10,R0          ;DONE YET?
5751  032306  001342                          BNE     1$              ;BR IF NO
5752  032310  104420                          ADVANCE                 ; ADVANCE LOOP
5753  032312    000     377     377   5$:     .BYTE   0,-1,-1,376,252,-1,-1,124
5754  032315    376     252     377
5755  032320    377     124
5756                                  .EVEN
5757
5758
5759                                  ;********************** TEST 120 ****************************
5760                                  ;*ALU TEST
5761                                  ;*TEST OF ALU FUNCTION 2A W/C WITH C BIT SET
5762                                  ;*ALU FUNCTION (A PLUS A PLUS C)    CODE=6
5763                                  ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5764                                  ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5765                                  ;:************************************************************
5766
5767                                  ;  TEST 120
5768                                  ;-------------
5769                                  ;:************************************************************
5770  032322  000004                  TST120: SCOPE
5771  032324  012737  000120  001202          MOV     #120,$TSTNM             ; LOAD THE NO. OF THIS TEST
5772  032332  012737  032500  001442          MOV     #TST121,NEXT            ; POINT TO THE START OF NEXT TEST.
5773  032340  012737  032372  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5774                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5775  032346  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5776  032350  005000                          CLR     R0              ;MEM + SP ADDRESS
5777  032352  012702  032470                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
5778  032356  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5779  032362  035726                          MEMDAT                  ;POINTER TO DATA
5780  032364  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5781  032370  035736                          SPDAT                   ;POINTER TO DATA
5782  032372  004737  035714          1$:     JSR     PC,SETC         ;SET C BIT!
5783  032376  042737  000017  032412          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5784  032404  050037  032412                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5785  032410  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5786  032412  010000                  2$:     010000                  ;LOAD MAR
5787  032414  042737  000017  032430          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5788  032422  050037  032430                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5789  032426  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5790  032430  040540                  3$:     040400!<6*20>           ;BR + 2A W/C
5791  032432  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5792  032434  061224                          61224                   ;MOVE BR TO PORT4
5793  032436  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5794  032440  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5795  032444  120504                          CMPB    R5,R4           ;DATA CORRECT?
5796  032446  001401                          BEQ     4$              ;BR IF YES
5797  032450  104015                          ERROR   15              ;ALU ERROR
5798  032452  104405                  4$:     SCOPE                   ;SW09=1?
```

```
5799  032454  005202                          INC     R2              ;NEXT DATA
5800  032456  005200                          INC     R0              ;NEXT ADDRESS
5801  032460  022700  000010                  CMP     #10,R0          ;DONE YET?
5802  032464  001342                          BNE     1$              ;BR IF NO
5803  032466  104420                          ADVANCE                 ; ADVANCE LOOP
5804  032470     001     001     377  5$:     .BYTE   1,1,-1,-1,253,253,125,125
5805  032473     377     253     253
5806  032476     125     125
5807                              .EVEN
5808
5809
5810                                          ;*************************** TEST 121 ****************************
5811                                          ;*ALU TEST
5812                                          ;*TEST OF ALU FUNCTION SUB WITH C BIT SET
5813                                          ;*ALU FUNCTION (A-B)    CODE=16
5814                                          ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5815                                          ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5816                                          ;;*************************************************************
5817
5818                                          ;  TEST 121
5819                                          ;-------------
5820                              ;;**********************************************************
5821  032500  000004              TST121: SCOPE
5822  032502  012737  000121  001202          MOV     #121,STSTNM             ; LOAD THE NO. OF THIS TEST
5823  032510  012737  032656  001442          MOV     #TST122,NEXT            ; POINT TO THE START OF NEXT TEST.
5824  032516  012737  032550  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5825                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5826  032524  104410                          MSTCLR                  ;MASTER CLEAR KMC11
5827  032526  005000                          CLR     R0              ;MEM + SP ADDRESS
5828  032530  012702  032646                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
5829  032534  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5830  032540  035726                          MEMDAT                  ;POINTER TO DATA
5831  032542  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5832  032546  035736                          SPDAT                   ;POINTER TO DATA
5833  032550  004737  035714          1$:     JSR     PC,SETC         ;SET C BIT!
5834  032554  042737  000017  032570          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5835  032562  050037  032570                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5836  032566  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5837  032570  010000                  2$:     010000                  ;LOAD MAR
5838  032572  042737  000017  032606          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5839  032600  050037  032606                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5840  032604  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5841  032606  040740                  3$:     040400!<16*20>          ;BR + SUB
5842  032610  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5843  032612  061224                          61224                   ;MOVE BR TO PORT4
5844  032614  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5845  032616  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5846  032622  120504                          CMPB    R5,R4           ;DATA CORRECT?
5847  032624  001401                          BEQ     4$              ;BR IF YES
5848  032626  104015                          ERROR   15              ;ALU ERROR
5849  032630  104405                  4$:     SCOPE                   ;SW09=1?
5850  032632  005202                          INC     R2              ;NEXT DATA
5851  032634  005200                          INC     R0              ;NEXT ADDRESS
5852  032636  022700  000010                  CMP     #10,R0          ;DONE YET?
5853  032642  001342                          BNE     1$              ;BR IF NO
5854  032644  104420                          ADVANCE                 ; ADVANCE LOOP
```

```
5855  032646  000  001  377  5$:    .BYTE   0.1,-1,0,0,253,125,0
5856  032651  000  000  253
5857  032654  125  000
5858                             .EVEN
5859
5860
5861                             ;***************************** TEST 122 *****************************
5862                             ;*ALU TEST
5863                             ;*TEST OF ALU FUNCTION ADD W/C WITH C BIT SET
5864                             ;*ALU FUNCTION (A PLUS B PLUS C)    CODE=01
5865                             ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5866                             ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5867                             ;:***********************************************************************
5868
5869                             ;  TEST 122
5870                             ;-------------
5871                             ;;***********************************************************************
5872  032656  000004             TST122: SCOPE
5873  032660  012737  000122  001202     MOV     #122,$TSTNM             ; LOAD THE NO. OF THIS TEST
5874  032666  012737  033034  001442     MOV     #TST123,NEXT            ; POINT TO THE START OF NEXT TEST.
5875  032674  012737  032726  001444     MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5876                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5877  032702  104410                     MSTCLR                          ;MASTER CLEAR KMC11
5878  032704  005000                     CLR     R0                      ;MEM + SP ADDRESS
5879  032706  012702  033024             MOV     #5$,R2                  ;POINTER TO CORRECT DATA
5880  032712  004737  035602             JSR     PC,MEMLD                ;LOAD 8 WORDS OF MAIN MEMORY
5881  032716  035726                     MEMDAT                          ;POINTER TO DATA
5882  032720  004737  035636             JSR     PC,SPLD                 ;LOAD 8 WORDS OF SP
5883  032724  035736                     SPDAT                           ;POINTER TO DATA
5884  032726  004737  035714      1$:    JSR     PC,SETC                 ;SET C BIT!
5885  032732  042737  000017  032746     BIC     #17,2$                  ;CLEAR ADDRESS FIELD OF INSTRUCTION
5886  032740  050037  032746             BIS     R0,2$                   ;ADD ADDRESS TO INSTRUCTION
5887  032744  104412                     ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5888  032746  010000              2$:    010000                          ;LOAD MAR
5889  032750  042737  000017  032764     BIC     #17,3$                  ;CLEAR ADDRESS OF INSTRUCTION
5890  032756  050037  032764             BIS     R0,3$                   ;ADD ADDRESS TO INSTRUCTION
5891  032762  104412                     ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5892  032764  040420              3$:    040400!<01*20>                  ;BR + ADD W/C
5893  032766  104412                     ROMCLK                          ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5894  032770  061224                     61224                           ;MOVE BR TO PORT4
5895  032772  111205                     MOVB    (R2),R5                 ;PUT "EXPECTED" IN R5
5896  032774  116104  000004             MOVB    4(R1),R4                ;PUT "FOUND" IN R4
5897  033000  120504                     CMPB    R5,R4                   ;DATA CORRECT?
5898  033002  001401                     BEQ     4$                      ;BR IF YES
5899  033004  104015                     ERROR   15                      ;ALU ERROR
5900  033006  104405              4$:    SCOP1                           ;SW09=1?
5901  033010  005202                     INC     R2                      ;NEXT DATA
5902  033012  005200                     INC     R0                      ;NEXT ADDRESS
5903  033014  022700  000010             CMP     #10,R0                  ;DONE YET?
5904  033020  001342                     BNE     1$                      ;BR IF NO
5905  033022  104420                     ADVANCE                         ; ADVANCE LOOP
5906  033024     001     000     000  5$:    .BYTE   1,0,0,-1,253,0,0,125
5907  033027     377     253     000
5908  033032     000     125
5909                             .EVEN
5910
```

```
5911
5912                                    ;*************************** TEST 123 ****************************
5913                                    ;*ALU TEST
5914                                    ;*TEST OF ALU FUNCTION SUB W/C WITH C BIT SET
5915                                    ;*ALU FUNCTION (A-B-C)    CODE=2
5916                                    ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
5917                                    ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
5918                                    ;:*************************************************************
5919
5920                                    ;  TEST 123
5921                                    ;-------------
5922                                ;:*************************************************************
5923  033034  000004                TST123: SCOPE
5924  033036  012737  000123 001202        MOV     #123,$TSTNM             ; LOAD THE NO. OF THIS TEST
5925  033044  012737  033212 001442        MOV     #TST124,NEXT            ; POINT TO THE START OF NEXT TEST.
5926  033052  012737  033104 001444        MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
5927                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
5928  033060  104410                       MSTCLR                  ;MASTER CLEAR KMC11
5929  033062  005000                       CLR     R0              ;MEM + SP ADDRESS
5930  033064  012702  033202               MOV     #5$,R2          ;POINTER TO CORRECT DATA
5931  033070  004737  035602               JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
5932  033074  035726                       MEMDAT                  ;POINTER TO DATA
5933  033076  004737  035636               JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
5934  033102  035736                       SPDAT                   ;POINTER TO DATA
5935  033104  004737  035714       1$:     JSR     PC,SETC         ;SET C BIT!
5936  033110  042737  000017 033124        BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
5937  033116  050037  033124               BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
5938  033122  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5939  033124  010000               2$:     010000                  ;LOAD MAR
5940  033126  042737  000017 033142        BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
5941  033134  050037  033142               BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
5942  033140  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5943  033142  040440               3$:     040400!<2*20>           ;BR + SUB W/C
5944  033144  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
5945  033146  061224                       61224                   ;MOVE BR TO PORT4
5946  033150  111205                       MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
5947  033152  116104  000004               MOVB    4(R1),R4        ;PUT "FOUND" IN R4
5948  033156  120504                       CMPB    R5,R4           ;DATA CORRECT?
5949  033160  001401                       BEQ     4$              ;BR IF YES
5950  033162  104015                       ERROR   15              ;ALU ERROR
5951  033164  104405               4$:     SCOP1                   ;SW09=1?
5952  033166  005202                       INC     R2              ;NEXT DATA
5953  033170  005200                       INC     R0              ;NEXT ADDRESS
5954  033172  022700  000010               CMP     #10,R0          ;DONE YET?
5955  033176  001342                       BNE     1$              ;BR IF NO
5956  033200  104420                       ADVANCE                 ; ADVANCE LOOP
5957  033202     000     001     377  5$:     .BYTE   0,1,-1,0,0,253,125,0
5958  033205     000     000     253
5959  033210     125     000
5960                                    .EVEN
5961
5962
5963                                    ;*************************** TEST 124 ****************************
5964                                    ;*ALU TEST
5965                                    ;*TEST OF ALU FUNCTION INC A WITH C BIT SET
5966                                    ;*ALU FUNCTION (A PLUS 1)    CODE=3
```

```
                                        ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
                                        ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
                                        ;;*****************************************************************

                                        ;  TEST 124
                                        ;-------------
                                    ;;*****************************************************************
033212  000004                  TST124: SCOPE
033214  012737  000124  001202          MOV     #124,$TSTNM             ; LOAD THE NO. OF THIS TEST
033222  012737  033370  001442          MOV     #TST125,NEXT            ; POINT TO THE START OF NEXT TEST.
033230  012737  033262  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
                                                                ;R1 CONTAINS BASE KMC11 ADDRESS
033236  104410                          MSTCLR                  ;MASTER CLEAR KMC11
033240  005000                          CLR     R0              ;MEM + SP ADDRESS
033242  012702  033360                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
033246  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
033252  035726                          MEMDAT                  ;POINTER TO DATA
033254  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
033260  035736                          SPDAT                   ;POINTER TO DATA
033262  004737  035714          1$:     JSR     PC,SETC         ;SET C BIT!
033266  042737  000017  033302          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
033274  050037  033302                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
033300  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
033302  010000                  2$:     010000                  ;LOAD MAR
033304  042737  000017  033320          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
033312  050037  033320                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
033316  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
033320  040460                  3$:     040400!<3*20>           ;BR + INC A
033322  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
033324  061224                          61224                   ;MOVE BR TO PORT4
033326  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
033330  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
033334  120504                          CMPB    R5,R4           ;DATA CORRECT?
033336  001401                          BEQ     4$              ;BR IF YES
033340  104015                          ERROR   15              ;ALU ERROR
033342  104405                  4$:     SCOPE                   ;SW09=1?
033344  005202                          INC     R2              ;NEXT DATA
033346  005200                          INC     R0              ;NEXT ADDRESS
033350  022700  000010                  CMP     #10,R0          ;DONE YET?
033354  001342                          BNE     1$              ;BR IF NO
033356  104420                          ADVANCE                 ; ADVANCE LOOP
033360     001     001     000  5$:     .BYTE   1,1,0,0,126,126,253,253
033363     000     126     126
033366     253     253
                                .EVEN


                                        ;*************************** TEST 125 ****************************
                                        ;*ALU TEST
                                        ;*TEST OF ALU FUNCTION 2A WITH C BIT SET
                                        ;*ALU FUNCTION (A PLUS A)    CODE=5
                                        ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
                                        ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
                                        ;;*****************************************************************

                                        ;  TEST 125
```

```
                                        ;----------
                                        ;;**********************************************************************
033370  000004                        TST125: SCOPE
033372  012737  000125  001202              MOV     #125,$TSTNM             ; LOAD THE NO. OF THIS TEST
033400  012737  033546  001442              MOV     #TST126,NEXT            ; POINT TO THE START OF NEXT TEST.
033406  012737  033440  001444              MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
033414  104410                              MSTCLR                  ;MASTER CLEAR KMC11
033416  005000                              CLR     R0              ;MEM + SP ADDRESS
033420  012702  033536                      MOV     #5$,R2          ;POINTER TO CORRECT DATA
033424  004737  035602                      JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
033430  035726                              MEMDAT                  ;POINTER TO DATA
033432  004737  035636                      JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
033436  035736                              SPDAT                   ;POINTER TO DATA
033440  004737  035714              1$:     JSR     PC,SETC         ;SET C BIT!
033444  042737  000017  033460              BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
033452  050037  033460                      BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
033456  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
033460  010000                      2$:     010000                  ;LOAD MAR
033462  042737  000017  033476              BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
033470  050037  033476                      BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
033474  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
033476  040520                      3$:     040400!<5*20>           ;BR + 2A
033500  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
033502  061224                              61224                   ;MOVE BR TO PORT4
033504  11;205                             MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
033506  116104  000004                      MOVB    4(R1),R4        ;PUT "FOUND" IN R4
033512  120504                              CMPB    R5,R4           ;DATA CORRECT?
033514  001401                              BEQ     4$              ;BR IF YES
033516  104015                              ERROR   15              ;ALU ERROR
033520  104405                      4$:     SCOPE                   ;SW09=1?
033522  005202                              INC     R2              ;NEXT DATA
033524  005200                              INC     R0              ;NEXT ADDRESS
033526  022700  000010                      CMP     #10,R0          ;DONE YET?
033532  001342                              BNE     1$              ;BR IF NO
033534  104420                              ADVANCE                 ; ADVANCE LOOP
033536     000     000     376  5$:         .BYTE   0,0,376,376,252,252,124,124
033541     376     252     252
033544     124     124
                                    .EVEN
                                        ;
                                        ;
                                        ;*************************** TEST 126 ****************************
                                        ;*ALU TEST
                                        ;*TEST OF ALU FUNCTION A PLUS C WITH C BIT SET
                                        ;*ALU FUNCTION (A PLUS C)    CODE=4
                                        ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
                                        ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
                                        ;;**********************************************************************
                                        ;
                                        ;  TEST 126
                                        ;----------
                                        ;;**********************************************************************
033546  000004                        TST126: SCOPE
033550  012737  000126  001202              MOV     #126,$TSTNM             ; LOAD THE NO. OF THIS TEST
033556  012737  033724  001442              MOV     #TST127,NEXT            ; POINT TO THE START OF NEXT TEST.
```

```
6079  033564  012737  033616  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
6080                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
6081  033572  104410                          MSTCLR                  ;MASTER CLEAR KMC11
6082  033574  005000                          CLR     R0              ;MEM + SP ADDRESS
6083  033576  012702  033714                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
6084  033602  004737  035602                  JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
6085  033606  035726                          MEMDAT                  ;POINTER TO DATA
6086  033610  004737  035636                  JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
6087  033614  035736                          SPDAT                   ;POINTER TO DATA
6088  033616  004737  035714          1$:     JSR     PC,SETC         ;SET C BIT!
6089  033622  042737  000017  033636          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
6090  033630  050037  033636                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
6091  033634  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6092  033636  010000                  2$:     010000                  ;LOAD MAR
6093  033640  042737  000017  033654          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
6094  033646  050037  033654                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
6095  033652  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6096  033654  040500                  3$:     040400!<4*20>           ;BR = A PLUS C
6097  033656  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6098  033660  061224                          61224                   ;MOVE BR TO PORT4
6099  033662  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
6100  033664  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
6101  033670  120504                          CMPB    R5,R4           ;DATA CORRECT?
6102  033672  001401                          BEQ     4$              ;BR IF YES
6103  033674  104015                          ERROR   15              ;ALU ERROR
6104  033676  104405                  4$:     SCOP1                   ;SW09=1?
6105  033700  005202                          INC     R2              ;NEXT DATA
6106  033702  005200                          INC     R0              ;NEXT ADDRESS
6107  033704  022700  000010                  CMP     #10,R0          ;DONE YET?
6108  033710  001342                          BNE     1$              ;BR IF NO
6109  033712  104420                          ADVANCE                 ; ADVANCE LOOP
6110  033714     001     001     000  5$:     .BYTE   1,1,0,0,126,126,253,253
6111  033717     000     126     126
6112  033722     253     253
6113                                  .EVEN
6114
6115
6116                                          ;*************************** TEST 127 ****************************
6117                                          ;*ALU TEST
6118                                          ;*TEST OF ALU FUNCTION 2'S COMP SUB WITH C BIT SET
6119                                          ;*ALU FUNCTION (A-B-1)    CODE=17
6120                                          ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
6121                                          ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
6122                                          ;:*************************************************************
6123
6124                                          ;  TEST 127
6125                                          ;-------------
6126                                  ;:*************************************************************
6127  033724  000004                  TST127: SCOPE
6128  033726  012737  000127  001202          MOV     #127,$TSTNM             ; LOAD THE NO. OF THIS TEST
6129  033734  012737  034102  001442          MOV     #TST130,NEXT            ; POINT TO THE START OF NEXT TEST.
6130  033742  012737  033774  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
6131                                                                    ;R1 CONTAINS BASE KMC11 ADDRESS
6132  033750  104410                          MSTCLR                  ;MASTER CLEAR KMC11
6133  033752  005000                          CLR     R0              ;MEM + SP ADDRESS
6134  033754  012702  034072                  MOV     #5$,R2          ;POINTER TO CORRECT DATA
```

```
6135  033760  004737  035602          JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
6136  033764  035726                  MEMDAT                  ;POINTER TO DATA
6137  033766  004737  035636          JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
6138  033772  035736                  SPDAT                   ;POINTER TO DATA
6139  033774  004737  035714  1$:     JSR     PC,SETC         ;SET C BIT!
6140  034000  042737  000017  034014  BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
6141  034006  050037  034014          BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
6142  034012  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6143  034014  010000          2$:     010000                  ;LOAD MAR
6144  034016  042737  000017  034032  BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
6145  034024  050037  034032          BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
6146  034030  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6147  034032  040760          3$:     040400!<17*20>          ;BR + 2'S COMP SUB
6148  034034  104412                  ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6149  034036  061224                  61224                   ;MOVE BR TO PORT4
6150  034040  111205                  MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
6151  034042  116104  000004          MOVB    4(R1),R4        ;PUT "FOUND" IN R4
6152  034046  120504                  CMPB    R5,R4           ;DATA CORRECT?
6153  034050  001401                  BEQ     4$              ;BR IF YES
6154  034052  104015                  ERROR   15              ;ALU ERROR
6155  034054  104405          4$:     SCOP1                   ;SW09=1?
6156  034056  005202                  INC     R2              ;NEXT DATA
6157  034060  005200                  INC     R0              ;NEXT ADDRESS
6158  034062  022700  000010          CMP     #10,R0          ;DONE YET?
6159  034066  001342                  BNE     1$              ;BR IF NO
6160  034070  104420                  ADVANCE                 ; ADVANCE LOOP
6161  034072     377     000     376  5$:     .BYTE   -1,0,376,-1,-1,252,124,-1
6162  034075     377     377     252
6163  034100     124     377
6164                                  .EVEN
6165
6166
6167                                  ;*************************** TEST 130 ****************************
6168                                  ;*ALU TEST
6169                                  ;*TEST OF ALU FUNCTION DEC A WITH C BIT SET
6170                                  ;*ALU FUNCTION (A-1)    CODE=7
6171                                  ;*LOAD MAIN MEM AND SP WITH 8 WORDS OF DATA
6172                                  ;*PERFORM THE FUNCTION, VERIFY THE RESULTS
6173                                  ;:*************************************************************
6174
6175                                  ;  TEST 130
6176                                  ;-------------
6177                          ;:*************************************************************
6178  034102  000004          TST130: SCOPE
6179  034104  012737  000130  001202          MOV     #130,$TSTNM             ; LOAD THE NO. OF THIS TEST
6180  034112  012737  034260  001442          MOV     #TST131,NEXT            ; POINT TO THE START OF NEXT TEST.
6181  034120  012737  034152  001444          MOV     #1$,LOCK                ; ADDRESS FOR LOCK ON DATA.
6182                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
6183  034126  104410                  MSTCLR                  ;MASTER CLEAR KMC11
6184  034130  005000                  CLR     R0              ;MEM + SP ADDRESS
6185  034132  012702  034250          MOV     #5$,R2          ;POINTER TO CORRECT DATA
6186  034136  004737  035602          JSR     PC,MEMLD        ;LOAD 8 WORDS OF MAIN MEMORY
6187  034142  035726                  MEMDAT                  ;POINTER TO DATA
6188  034144  004737  035636          JSR     PC,SPLD         ;LOAD 8 WORDS OF SP
6189  034150  035736                  SPDAT                   ;POINTER TO DATA
6190  034152  004737  035714  1$:     JSR     PC,SETC         ;SET C BIT!
```

```
6191  034156  042737  000017  034172          BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
6192  034164  050037  034172                  BIS     R0,2$           ;ADD ADDRESS TO INSTRUCTION
6193  034170  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6194  034172  010000                  2$:     010000                  ;LOAD MAR
6195  034174  042737  000017  034210          BIC     #17,3$          ;CLEAR ADDRESS OF INSTRUCTION
6196  034202  050037  034210                  BIS     R0,3$           ;ADD ADDRESS TO INSTRUCTION
6197  034206  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6198  034210  040560                  3$:     040400!<7*20>           ;BR ← DEC A
6199  034212  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6200  034214  061224                          61224                   ;MOVE BR TO PORT4
6201  034216  111205                          MOVB    (R2),R5         ;PUT "EXPECTED" IN R5
6202  034220  116104  000004                  MOVB    4(R1),R4        ;PUT "FOUND" IN R4
6203  034224  120504                          CMPB    R5,R4           ;DATA CORRECT?
6204  034226  001401                          BEQ     4$              ;BR IF YES
6205  034230  104015                          ERROR   15              ;ALU ERROR
6206  034232  104405                  4$:     SCOP1                   ;SW09=1?
6207  034234  005202                          INC     R2              ;NEXT DATA
6208  034236  005200                          INC     R0              ;NEXT ADDRESS
6209  034240  022700  000010                  CMP     #10,R0          ;DONE YET?
6210  034244  001342                          BNE     1$              ;BR IF NO
6211  034246  104420                          ADVANCE                 ;ADVANCE LOOP
6212  034250     377     377     376  5$:     .BYTE   -1,-1,376,376,124,124,251,251
6213  034253     376     124     124
6214  034256     251     251
6215                                  .EVEN
6216
6217
6218                                          ;************************** TEST 131 **************************
6219                                          ;*TEST OF PROGRAM CLOCK BIT
6220                                          ;*DO A MASTER CLEAR, VERIFY THAT PROGRAM CLOCK IS SET
6221                                          ;*WRITE PROGRAM CLOCK BIT TO A ONE, VERIFY THAT IT CLEARS,
6222                                          ;*AND THEN SETS SOME TIME LATER
6223                                          ;:**********************************************************************
6224
6225                                          ;  TEST 131
6226                                          ;-------------
6227                                  ;;**********************************************************************
6228  034260  000504                  TST131: SCOPE
6229  034262  012737  000131  001202          MOV     #131,$TSTNM             ; LOAD THE NO. OF THIS TEST
6230  034270  012737  034466  001442          MOV     #TST132,NEXT            ; POINT TO THE START OF NEXT TEST.
6231                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
6232  034276  104410                          MSTCLR                  ;MASTER CLEAR KMC11
6233  034300  005037  011106                  CLR     TEMP            ;PREPARE FOR
6234  034304  005037  001276                  CLR     $TMPO           ;DELAY
6235  034310  012702  000011                  MOV     #11,R2          ;SAVE FOR TYPEOUT
6236  034314  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6237  034316  121224                          121224                  ;PORT4←LU11
6238  034320  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
6239  034324  042704  000357                  BIC     #357,R4         ;CLEAR UNWANTED BITS
6240  034330  012705  000020                  MOV     #20,R5          ;PUT "EXPECTED" IN R5
6241  034334  120504                          CMPB    R5,R4           ;IS PGM CLOCK SET?
6242  034336  001401                          BEQ     1$
6243  034340  104016                          ERROR   16              ;ERROR, PGM CLOCK IS NOT SET
6244  034342  012761  000020  000004  1$:     MOV     #20,4(R1)       ;LOAD PORT 4
6245  034350  152761  000002  000001          BISB    #BIT1,1(R1)     ;SET ROMI
6246  034356  012761  121111  000006          MOV     #121111,6(R1)   ;SEL6 ← INSTRUCTION
```

```
6247  034356  152761  000003  000001          BISB    #BIT1!BIT0,1(R1);SET CLOCK BIT
6248  034364  012761  121224  000006          MOV     #121224,6(R1)   ;LOAD NEXT INSTRUCTION
6249  034372  152761  000003  000001          BISB    #BIT1!BIT0,1(R1);READ CLOCK BIT
6250  034400  142761  000003  000001          BICB    #BIT1!BIT0,1(R1);CLEAR MAINT BITS
6251  034406  016104  000004                  MOV     4(R1),R4        ;PUT "FOUND" IN R4
6252  034414  005005                          CLR     R5              ;PUT "EXPECTED" IN R5
6253  034416  120504                          CMPB    R5,R4           ;IS PGM CLOCK CLEAR?
6254  034420  001401                          BEQ     2$
6255  034422  104016                          ERROR   16              ;ERROR, PGM CLOCK IS NOT CLEAR
6256  034424                          2$:
6257  034424  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6258  034426  121224                          121224                  ;PORT4←LU11
6259  034430  122761  000020  000004          CMPB    #20,4(R1)       ;IS PGM CLOCK SET?
6260  034436  001411                          BEQ     3$              ;BR IF YES
6261  034440  005237  011106                  INC     TEMP            ;INCREMENT DELAY
6262  034444  005537  001276                  ADC     $TMP0           ;INCREMENT DELAY
6263  034450  022737  000006  001276          CMP     #6,$TMP0        ;IS DELAY DONE
6264  034456  001362                          BNE     2$              ;BR IF NO
6265  034460  104016                          ERROR   16              ;ERROR PGM CLOCK NOT SET
6266  034462                          3$:
6267
6268
6269                                          ;************************** TEST 132 **************************
6270                                          ;*FORCE POWER FAIL TEST
6271                                          ;*SET FORCE POWER FAIL BIT VERIFY THAT PROCESSOR TRAPS TO 24
6272                                          ;*GOING DOWN AND COMING UP. VERIFY ALSO THAT BUS INIT WAS
6273                                          ;*BLOCKED FROM GETTING TO THE KMC DURING THE POWER FAIL
6274                                          ;:************************************************************
6275
6276                                          ;  TEST 132
6277                                          ;-------------
6278                                          ;:************************************************************
6279  034462  000004                  TST132: SCOPE
6280  034464  012737  000132  001202          MOV     #132,$TSTNM             ; LOAD THE NO. OF THIS TEST
6281  034472  012737  034654  001442          MOV     #TST133,NEXT            ; POINT TO THE START OF NEXT TEST.
6282                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
6283  034500  104410                          MSTCLR                  ;MASTER CLEAR KMC11
6284  034502  005037  011106                  CLR     TEMP            ;PREPARE FOR DELAY
6285  034506  013746  000024                  MOV     @#24,-(SP)      ;STORE POWER FAIL ADDRESS
6286  034512  012737  034556  000024          MOV     #1$,@#24        ;SET UP FOR FORCE POWER FAIL
6287  034520  012761  000002  000004          MOV     #2,4(R1)        ;LOAD PORT4
6288  034526  012711  001000                  MOV     #BIT9,(R1)      ;SET ROMI
6289  034532  012761  121111  000006          MOV     #121111,6(R1)   ;LOAD INSTRUCTION
6290  034540  012711  001400                  MOV     #BIT9!BIT8,(R1) ;CLOCK INSTRUCTION
6291  034544  005237  011106          5$:     INC     TEMP            ;WAIT FOR POWER FAIL
6292  034550  001375                          BNE     5$              ;BR IF DELAY NOT DONE
6293  034552  104017                          ERROR   17              ;ERROR, NO POWER FAIL
6294  034554  000426                          BR      4$
6295  034556  012737  034574  000024  1$:     MOV     #3$,@#24        ;POWER UP ADDRESS
6296  034564  010637  034572                  MOV     SP,2$           ;STORE STACK
6297  034570  000000                          HALT                    ;WAIT FOR POWER UP SEQUENCE
6298  034572  000000                  2$:     0
6299  034574  013706  034572          3$:     MOV     2$,SP           ;RESTORE STACK
6300  034600  022626                          POP2SP                  ;POP STACK TWICE
6301  034602  012637  000024                  MOV     (SP)+,@#24      ;RESTORE TRUE POWER FAIL ADDRESS
6302  034606  022737  007126  000024          CMP     #$PWRDN,@#24    ;IS IT CORRECT?
```

```
6303  034620  001406                          BEQ     4$              ;BR IF YES
6304  034622  104017                          ERROR   17              ;ERROR, STACK IS INCORRECT
6305  034624  012737  007126  000024          MOV     #$PWRDN,@#24    ;RESTORE TRUE POWER FAIL ADDRESS
6306  034632  012706  001200                  MOV     #STACK,SP       ;RESTORE STACK
6307  034636  012711  003000          4$:     MOV     #BIT9!BIT10,(R1);SEL6 = MAINT IR
6308  034642  012705  121111                  MOV     #121111,R5      ;R5 = EXPECTED
6309  034646  016104  000004                  MOV     4(R1),R4        ;R4 = FOUND
6310  034652  020504                          CMP     R5,R4           ;MAINT IR SHOULD = 12111
6311  034654  001401                          BEQ     .+4             ;BR IF OK
6312  034656  104025                          ERROR   25              ;IF = 0 THEN BUS INIT WAS
6313                                                                  ;NOT BLOCKED FROM CLEARING
6314                                                                  ;THE KMC-11
6315
6316
6317                                          ;*************************** TEST 133 ****************************
6318                                          ;*MICRO-PROCESSOR NOISE TEST
6319                                          ;*WRITE ALL ZERO'S THEN ALL ONE'S THEN A DATA PATTERN
6320                                          ;*TO THE IBUS* AND IBUS REGISTERS AND TO THE SP AND MAIN MEM
6321                                          ;*THEN GO BACK AND READ THE DATA PATERNS  TO VERIFY THAT
6322                                          ;*READING AND WRITING OF OTHER LOCATIONS AND REGISTERS
6323                                          ;*DID NOT CHANGE THE DATA.
6324                                          ;;*************************************************************************
6325
6326                                          ;  TEST 133
6327                                          ;--------------
6328                                          ;;*************************************************************
6329  034660  000004          TST133: SCOPE
6330  034662  012737  000133  001202          MOV     #133,$TSTNM             ; LOAD THE NO. OF THIS TEST
6331  034670  012737  003662  001442          MOV     #SEOP,.NEXT             ; POINT TO THE END OF PASS HANDLER.
6332                                                                  ;R1 CONTAINS BASE KMC11 ADDRESS
6333  034676  104410                          MSTCLR                  ;MASTER CLEAR KMC11
6334  034700  005002                          CLR     R2              ;R2 IS INDEX REGISTER
6335  034702  042737  000017  034726  1$:     BIC     #17,2$          ;CLEAR ADDRESS FIELD
6336  034710  156237  035502  034726          BISB    30$(R2),2$      ;ADD IBUS* REG ADDRESS TO INSTRUCTION
6337  034716  116261  035510  000004          MOVB    31$(R2),4(R1)   ;LOAD PORT4
6338  034724  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6339  034726  121100                  2$:     121100                  ;WRITE IBUS* REGISTER
6340  034730  005202                          INC     R2              ;INC INDEX REGISTER
6341  034732  022702  000005                  CMP     #5,R2           ;DONE YET?
6342  034736  001361                          BNE     1$              ;BR IF NO
6343  034740  005002                          CLR     R2              ;R2 IS IBUS REGISTER ADDRESS
6344  034742  042737  000017  035006  3$:     BIC     #17,4$          ;CLEAR ADDRESS FIELD OF INSTRUCTIONS
6345  034750  042737  000017  035020          BIC     #17,5$
6346  034756  042737  000017  035030          BIC     #17,6$
6347  034764  050237  035006                  BIS     R2,4$           ;ADD IBUS REG ADDRESS TO INSTRUCTION
6348  034770  050237  035020                  BIS     R2,5$
6349  034774  050237  035030                  BIS     R2,6$
6350  035000  105061  000004                  CLRB    4(R1)           ;CLEAR PORT4
6351  035004  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6352  035006  122100                  4$:     122100                  ;WRITE 0 TO IBUS REG
6353  035010  112761  000377  000004          MOVB    #377,4(R1)      ;LOAD PORT4
6354  035016  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6355  035020  122100                  5$:     122100                  ;WRITE ALL ONES TO IBUS REG
6356  035022  110261  000004                  MOVB    R2,4(R1)        ;LOAD PORT4
6357  035026  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6358  035030  122100                  6$:     122100                  ;WRITE ITS OWN ADDRESS TO IBUS REG
```

```
6359  035032  005202                          INC     R2              ;NEXT ADDRESS
6360  035034  022702  000010                  CMP     #10,R2          ;DONE YET?
6361  035040  001340                          BNE     3$              ;BR IF NO
6362  035042  005002                          CLR     R2              ;START AT SP ADDRESS 0
6363  035044  042737  000017  035110  7$:     BIC     #17,8$          ;CLEAR ADDRESS FIELD
6364  035052  042737  000017  035122          BIC     #17,9$
6365  035060  042737  000017  035132          BIC     #17,10$
6366  035066  050237  035110                  BIS     R2,8$           ;ADD ADDRESS TO INSTRUCTION
6367  035072  050237  035122                  BIS     R2,9$
6368  035076  050237  035132                  BIS     R2,10$
6369  035102  105061  000004                  CLRB    4(R1)           ;CLEAR PORT4
6370  035106  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6371  035110  123100                  8$:     123100                  ;WRITE ZERO TO SP
6372  035112  112761  000377  000004          MOVB    #377,4(R1)      ;LOAD PORT4
6373  035120  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6374  035122  123100                  9$:     123100                  ;WRITE ALL ONES TO SP
6375  035124  110261  000004                  MOVB    R2,4(R1)        ;LOAD PORT4
6376  035130  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6377  035132  123100                  10$:    123100                  ;WRITE SP ADDRESS TO ITSELF
6378  035134  005202                          INC     R2              ;NEXT SP ADDRESS
6379  035136  022702  000020                  CMP     #20,R2          ;DONE YET?
6380  035142  001340                          BNE     7$              ;BR IF NO
6381  035144  005002                          CLR     R2              ;R2 = MAIN MEM ADDRESS
6382  035146  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6383  035150  010000                          010000                  ;MAR ← 0
6384  035152  105061  000004          11$:    CLRB    4(R1)           ;CLEAR PORT4
6385  035156  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6386  035160  122500                          122500                  ;WRITE ZEROS TO MEM
6387  035162  112761  000377  000004          MOVB    #377,4(R1)      ;LOAD PORT4
6388  035170  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6389  035172  122500                          122500                  ;WRITE ONES TO MEM
6390  035174  110261  000004                  MOVB    R2,4(R1)        ;LOAD PORT4
6391  035200  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6392  035202  136500                          136500                  ;WRITE TO MEM IT OWN ADDRESS
6393  035204  005202                          INC     R2              ;NEXT MEM ADDRESS
6394  035206  022702  001000          CMP     #1000,R2                ;DONE YET?
6395  035212  001357                          BNE     11$             ;BR IF NO
6396
6397                                          ;NOW GO BACK AND READ EVERYTHING
6398
6399  035214  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6400  035216  010000                          010000                  ;MAR←0
6401  035220  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6402  035222  004000                          4000                    ;MAR HI ← 0 (KMC ONLY)
6403  035224  005000                          CLR     R0              ;R0 IS INDEX REGISTER
6404  035226  042737  000360  035264  12$:    BIC     #360,13$        ;CLEAR ADDRESS FIELD
6405  035234  116002  035502                  MOVB    30$(R0),R2      ;R2 = IBUS# ADDRESS
6406  035240  010203                          MOV     R2,R3           ;PUT IBUS# ADDRESS IN R3
6407  035242  006303                          ASL     R3              ;SHIFT ADDRESS TO BITS 4-7
6408  035244  006303                          ASL     R3
6409  035246  006303                          ASL     R3
6410  035250  006303                          ASL     R3
6411  035252  050337  035264                  BIS     R3,13$          ;ADD ADDRESS TO INSTRUCTION
6412  035256  116005  035510                  MOVB    31$(R0),R5      ;R5 = "EXPECTED"
6413  035262  104412                          ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6414  035264  121004                  13$:    121004                  ;PORT4 ← IBUS# REGISTER
```

```
6415  035266  016104  000004                        MOV     4(R1),R4        ;R4 = "FOUND"
6416  035272  120504                                CMPB    R5,R4           ;IBUS* CONTENTS OK?
6417  035274  001401                                BEQ     .+4             ;BR IF YES
6418  035276  104004                                ERROR   4               ;IBUS* DATA ERROR
6419  035300  005200                                INC     R0              ;INC COUNTER
6420  035302  022700  000005                        CMP     #5,R0           ;DONE YET?
6421  035306  001347                                BNE     12$             ;BR IF NO
6422  035310  005002                                CLR     R2              ;R2 = IBUS REG ADDRESS
6423  035312  042737  000360  035342  14$:          BIC     #360,15$        ;CLEAR ADDRESS FIELD OF INSTRUCTION
6424  035320  010203                                MOV     R2,R3           ;R3 = IBUS ADDRESS
6425  035322  006303                                ASL     R3              ;SHIFT ADDRESS TO BITS 4-7
6426  035324  006303                                ASL     R3
6427  035326  006303                                ASL     R3
6428  035330  006303                                ASL     R3
6429  035332  050337  035342                        BIS     R3,15$          ;ADD ADDRESS TO INSTUCTION
6430  035336  010205                                MOV     R2,R5           ;R5 = "EXPECTED"
6431  035340  104412                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6432  035342  021004                        15$:     021004                  ;PORT4 ← IBUS REG
6433  035344  016104  000004                        MOV     4(R1),R4        ;R4 = "FOUND"
6434  035350  120504                                CMPB    R5,R4           ;IBUS CONTENTS OK?
6435  035352  001401                                BEQ     .+4             ;BR IF YES
6436  035354  104005                                ERROR   5               ;IBUS DATA ERROR
6437  035356  005202                                INC     R2              ;NEXT IBUS REGISTER
6438  035360  022702  000010                        CMP     #10,R2          ;DONE YET?
6439  035364  001352                                BNE     14$             ;BR IF NO
6440  035366  005002                                CLR     R2              ;R2 = SP ADDRESS
6441  035370  042737  000017  035404  16$:          BIC     #17,17$ ;CLEAR ADDRESS FIELD OF INSTRUCTION
6442  035376  050237  035404                        BIS     R2,17$          ;ADD ADDRESS TO INSTRUCTION
6443  035402  104412                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6444  035404  040600                        17$:     040600                  ;BR ← SP
6445  035406  010205                                MOV     R2,R5           ;R5 = "EXPECTED"
6446  035410  104412                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6447  035412  061224                                061224                  ;PORT4 ← BR
6448  035414  016104  000004                        MOV     4(R1),R4        ;R4 = "FOUND"
6449  035420  120504                                CMPB    R5,R4           ;SP CONTENTS OK?
6450  035422  001401                                BEQ     .+4             ;BR IF YES
6451  035424  104007                                ERROR   7               ;SP DATA ERROR
6452  035426  005202                                INC     R2              ;NEXT SP LOCATION
6453  035430  022702  000020                        CMP     #20,R2          ;DONE YET?
6454  035434  001355                                BNE     16$             ;BR IF NO
6455  035436  005002                                CLR     R2              ;R2 = MEMORY ADDRESS
6456  035440  104412                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6457  035442  010000                                010000                  ;MAR ← 0
6458  035444  104412                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6459  035446  004000                                4000                    ;MAR HI ← 0 (KMC ONLY)
6460  035450  010205                        18$:     MOV     R2,R5           ;R5 = "EXPECTED"
6461  035452  104412                                ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6462  035454  055224                                055224                  ;PORT4 ← MAIN MEM
6463  035456  016104  000004                        MOV     4(R1),R4        ;R4 = "FOUND"
6464  035462  120504                                CMPB    R5,R4           ;MAIN MEM CONTENTS OK?
6465  035464  001401                                BEQ     .+4             ;BR IF YES
6466  035466  104013                                ERROR   13              ;MAIN MEM DATA ERROR
6467  035470  005202                                INC     R2              ;NEXT MEM ADDRESS
6468  035472  022702  001000                CMP     #1000,R2                ;DONE YET?
6469  035476  001364                                BNE     18$             ;BR IF NO
6470  035500  104420                                ADVANCE                 ; ADVANCE LOOP
```

```
6471  035502  000  002  003  30$:   .BYTE   0,2,3,5,10
6472  035505  005  010
6473          035510               .EVEN
6474  035510  001  003  004  31$:   .BYTE   1,3,4,6,10
6475  035513  006  010
6476          035516               .EVEN
6477
6478
6479                               ;SUBROUTINES
6480                               ;-----------
6481
6482  035516                       SETVEC:
6483                                       ;THIS SUBROUTINE LOADS THE VECTORS AND VECTOR LEVELS
6484
6485  035516  012577  144334               MOV     (R5)+,@KMRVEC   ;LOAD BASE VECTOR
6486  035522  012577  144334               MOV     (R5)+,@KMTVEC   ;LOAD VECTOR + 2
6487  035526  112577  144326               MOVB    (R5)+,@KMRLVL   ;LOAD VECTOR + 4
6488  035532  112577  144326               MOVB    (R5)+,@KMTLVL   ;LOAD VECTOR + 6
6489  035536  000205                       RTS     R5              ;RETURN
6490
6491
6492  035540                       NPRSET:
6493                                       ;THIS SUBROUTINE LOADS IBUS REGISTERS 0-7
6494                                       ;WITH NPR INFORMATION (INBA, OUTBA, OUT DATA)
6495
6496  035540  010246                       MOV     R2,-(SP)        ;SAVE R2
6497  035542  005002                       CLR     R2              ;START AT IBUS REG 0
6498  035544  112561  000004       1$:     MOVB    (R5)+,4(R1)     ;LOAD PORT4
6499  035550  042737  000017  035564       BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
6500  035556  050237  035564               BIS     R2,2$           ;ADD ADDRESS TO INSTRUCTION
6501  035562  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6502  035564  122100               2$:     122100                  ;MOVE PORT4 TO IBUS REG
6503  035566  005202                       INC     R2              ;NEXT ADDRESS
6504  035570  022702  000010               CMP     #10,R2          ;ALL DONE?
6505  035574  001363                       BNE     1$              ;BR IF NO
6506  035576  012602                       MOV     (SP)+,R2        ;RESTORE R2
6507  035600  000205                       RTS     R5              ;RETURN
6508
6509
6510  035602                       MEMLD:
6511                                       ;THIS SUBROUTINE LOADS THE FIRST 8 LOCATIONS OF MAIN
6512                                       ;MEMORY WITH THIS DATA: 0,-1,,0,-1,125,252,125,252
6513
6514  035602  013605                       MOV     @(SP)+,R5       ;PUT POINTER TO DATA IN R5
6515  035604  062746  000002               ADD     #2,-(SP)        ;ADJUST STACK
6516  035610  012704  000010               MOV     #10,R4          ;DO 8 LOADS
6517  035614  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6518  035616  010000                       010000                  ;MAR < 0
6519  035620  112577  144250       1$:     MOVB    (R5)+,@KMP04    ;LOAD PORT4
6520  035624  104412                       ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6521  035626  136500                       136500                  ;MOV DATA TO MEM, AUTO INC MAR
6522  035630  005304                       DEC     R4              ;DECREMENT COUNT
6523  035632  001372                       BNE     1$              ;BR IF NOT DONE
6524  035634  000207                       RTS     PC              ;RETURN
6525
6526
```

```
6527  035636                                SPLD:
6528                                              ;THIS SUBROUTINE LOADS THE FIRST 8 SCRATCH PAD
6529                                              ;LOCATIONS WITH: 0,0,-1,-1,125,125,252,252
6530
6531  035636  013605                              MOV     @(SP)+,R5       ;PUT POINTER TO DATA IN R5
6532  035640  062746  000002                      ADD     #2,-(SP)        ;ADJUST STACK
6533  035644  005004                              CLR     R4              ;START AT SP ADDRESS 0
6534  035646  112577  144222          1$:         MOVB    (R5)+,@KMP04    ;LOAD PORT4 WITH DATA
6535  035652  042737  000017  035666              BIC     #17,2$          ;CLEAR ADDRESS FIELD OF INSTRUCTION
6536  035660  050437  035666                      BIS     R4,2$           ;ADD ADDRESS TO INSTRUCTION
6537  035664  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6538  035666  123100                    2$:       123100                  ;MOVE DATA TO SP
6539  035670  005204                              INC     R4              ;INCREMENT COUNT
6540  035672  022704  000010                      CMP     #10,R4          ;DONE YET?
6541  035676  001363                              BNE     1$              ;BR IF NO
6542  035700  000207                              RTS     PC              ;RETURN
6543
6544
6545  035702                                CLRC:
6546                                              ;THIS SUBROUTINE CLEARS THE MICRO PROCESSOR C BIT
6547
6548  035702  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6549  035704  010000                              010000                  ;MAR←0
6550  035706  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6551  035710  040400                              040400!<0*20>           ;CLEAR C BIT
6552  035712  000207                              RTS     PC              ;RETURN
6553
6554
6555  035714                                SETC:
6556                                              ;THIS SUBROUTINE SETS THE MICRO PROCESSOR C BIT
6557
6558  035714  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6559  035716  010003                              010003                  ;MAR←3
6560  035720  104412                              ROMCLK                  ;NEXT WORD IS INSTRUCTION, ROMCLK PC=5304
6561  035722  040403                              040403!<0*20>           ;SET C BIT
6562  035724  000207                              RTS     PC              ;RETURN
6563
6564
6565  035726     000     377     000  MEMDAT:  .BYTE   0,-1,0,-1,125,252,125,252
6566  035731     377     125     252
6567  035734     125     252
6568  035736     000     000     377  SPDAT:   .BYTE   0,0,-1,-1,125,125,252,252
6569  035741     377     125     125
6570  035744     252     252
6571                                        .EVEN
6572  035746  052600  044516  052502  EM1:     .ASCIZ  <200>/UNIBUS REGISTER ADDRESSING TIME-OUT/
      036013     200  047125  041111  EM2:     .ASCIZ  <200>↑UNIBUS REGISTER WRITE/READ TEST↑
      036054  046600  041511  047522  EM3:     .ASCIZ  <200>/MICRO PROCESSOR TEST/
      036102  046600  041511  047522  EM4:     .ASCIZ  <200>↑MICRO PROCESSOR WRITE/READ TEST↑
      036143     200  051102  051040  EM5:     .ASCIZ  <200>/BR REGISTER TEST/
      036165     200  041523  040522  EM6:     .ASCIZ  <200>/SCRATCH PAD TEST/
      036207     200  042504  044526  EM7:     .ASCIZ  <200>/DEVICE FAILED TO INTERRUPT/
      036243     200  042504  044526  EM10:    .ASCIZ  <200>/DEVICE INTERRUPTED TO WRONG VECTOR
      036307     200  050116  020122  EM11:    .ASCIZ  <200>/NPR TEST/
      036321     200  040515  047111  EM12:    .ASCIZ  <200>/MAIN MEMORY TEST/
      036343     200  040515  020122  EM13:    .ASCIZ  <200>/MAR TEST/
```

```
036355      200  046101  020125  EM14:   .ASCIZ  <200>/ALU TEST/
036367      200  051120  043517  EM15:   .ASCIZ  <200>/PROGRAM CLOCK TEST/
036413      200  047506  041522  EM16:   .ASCIZ  <200>/FORCE POWER FAIL ERROR/
036443      200  047125  054105  EM17:   .ASCIZ  <200>/UNEXPECTED INTERRUPT/
036471      200  046513  030503  EM20:   .ASCIZ  <200>/KMC11 CONFIGURATION ERROR/
036524  046600  044501  052116  EM21:   .ASCIZ  <200>/MAINTENANCE INSTRUCTION REGISTER TEST/
036573      200  047520  042527  EM22:   .ASCIZ  <200>/POWER FAIL INITIALIZE FAILURE/

036632  051200  043505  051511  DH1:    .ASCIZ  <200>/REGISTER          TRAPPED FROM/
036673      200  054105  042520  DH2:    .ASCIZ  <200>/EXPECTED  FOUND    REGISTER/
036731      200  054105  042520  DH3:    .ASCIZ  <200>/EXPECTED  FOUND/
036752  042600  050130  041505  DH4:    .ASCIZ  <200>/EXPECTED  FOUND  IBUS* REGISTER/
037014  042600  050130  041505  DH5:    .ASCIZ  <200>/EXPECTED  FOUND  IBUS REGISTER/
037055      200  054105  042520  DH6:    .ASCIZ  <200>/EXPECTED  FOUND    ADDRESS/
                                        .EVEN

037112  000002                  DT1:    2
037114      006      015                .BYTE   6,15
037116  001264                          $REG1
037120      006      002                .BYTE   6,2
037122  001266                          $REG2
037124  000003                  DT2:    3
037126      006      004                .BYTE   6,4
037130  001274                          $REG5
037132      006      004                .BYTE   6,4
037134  001272                          $REG4
037136      006      002                .BYTE   6,2
037140  001264                          $REG1
037142  000002                  DT3:    2
037144      006      004                .BYTE   6,4
037146  001274                          $REG5
037150      006      002                .BYTE   6,2
037152  001272                          $REG4
037154  000003                  DT4:    3
037156      003      007                .BYTE   3,7
037160  001274                          $REG5
037162      003      011                .BYTE   3,11
037164  001272                          $REG4
037166      002      002                .BYTE   2,2
037170  001266                          $REG2
037172  000003                  DT5:    3
037174      003      007                .BYTE   3,7
037176  001274                          $REG5
037200      003      007                .BYTE   3,7
037202  001272                          $REG4
037204      006      002                .BYTE   6,2
037206  001266                          $REG2
037210  000002                  DT6:    2
037212      003      007                .BYTE   3,7
037214  001274                          $REG5
037216      003      002                .BYTE   3,2
037220  001272                          $REG4
037222  000002                  DT7:    2
037224      006      004                .BYTE   6,4
037226  001264                          $REG1
037230      006      002                .BYTE   6,2
```

```
037232  002066                  KMCSR
037234                  CORMAX:
        000001          .END
```

```
ABASE = 000000            266     307
ACDW1 = 000000            266     309
ACDW2 = 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
ADEVM = 000000            266     308
ADRCNT  006057           1371*   1386*   1395*
ADVANC= 104420           1540*   2250    4173    4216    4254    4292    4326    4536    4579    4630    4681    4732    4783
                         4834    4885    4936    4987    5038    5089    5140    5191    5242    5293    5344    5395    5446
                         5497    5548    5599    5650    5701    5752    5803    5854    5905    5956    6007    6058    6109
                         6160    6211    6470
AENV  = 000002              1*    266     277
AENVM = 000000            266     278
AFATAL= 000000            266     269
AMADR1= 000000            266     294
AMADR2= 000000            266     298
AMADR3= 000000            266     301
AMADR4= 000000            266     304
AMAMS1= 000000            266     288
AMAMS2= 000000            266     296
AMAMS3= 000000            266     299
AMAMS4= 000000            266     302
AMSGAD= 000000            266     274
AMSGLG= 000000            266     275
AMSGTY= 000000            266     268
AMTYP1= 000000            266     289
AMTYP2= 000000            266     297
AMTYP3= 000000            266     300
AMTYP4= 000000            266     303
APASS = 000000            266     271
APRIOR= 000000            266
APTCSU= 000040           1096    1201*
APTENV= 000001           1089    1157    1199*   1601
APTSIZ= 000200           1198*
APTSP0= 000100           1091    1159    1200*
APT.SI  013510            764    2175*
ASWREG= 000000            266     279
ATESTN= 000000            266     270
AUDONE  003354            801     822     861*
AUNIT = 000000            266     273
```

```
AUSTRT  003126            800#
AUSWR = 000000            266     280
AUTO.S  012110            762    1919#
AVECT1= 000000            266     305
AVECT2= 000000            266     306
BASE    013726           2186#   2197    2211*   2221#
BINWRD  006412           1443#   1446*   1447    1484#
BIT0  = 000001            129#    1708    1709    2333    2341    2621    2629    4505    6247    6249    6250
BIT00 = 000001            119#     129
BIT01 = 000002            118#     128
BIT02 = 000004            117#     127
BIT03 = 000010            116#     126
BIT04 = 000020            115#     125
BIT05 = 000040            114#     124
BIT06 = 000100            113#     123
BIT07 = 000200            112#     122
BIT08 = 000400            111#     121
BIT09 = 001000            110#     120
BIT1  = 000002            128#     769    1702    1708    1709    2056    2365    2373    2653    2661    6245    6247    6249
                         6250
BIT10 = 002000            109#    2037    2976    2995    3018    3037    4357    4395    6307
BIT11 = 004000            108#    1032    2557    2565
BIT12 = 010000            107#    1991    2054    2589    2597    2781    2789
BIT13 = 020000            106#    1994    2040    2058    2813    2821
BIT14 = 040000            105#    1014    2005    2007    2058    2069    3036
BIT15 = 100000            104#     709
BIT2  = 000004            127#     769     910    1709    2397    2405    2685    2693
BIT3  = 000010            126#    2060    2067
BIT4  = 000020            125#    1692    1715    1717
BIT5  = 000040            124#    2172    2429    2437
BIT6  = 000100            123#    1697    1698    2062    2461    2469    2717    2725
BIT7  = 000200            122#    1698    1817    2172    2493    2501    2749    2757
BIT8  = 000400            121#    2049    2051    2064    2066    2072    2075    2132    3064    4358    4396    6290
BIT9  = 001000            120#    2046    2072    2075    2130    2132    2525    2533    2976    2995    3018    3037    3062
                         3064    4357    4395    6288    6290    6307
BM      010631           1740#    2018
BPTVEC= 000014            136#
BRLVL   013442           2127    2137    2145    2158#
BRW     004360            915    1049#
CHRCNT  006410           1441*   1444    1448    1464*   1482#   1483
CKSWR   011212            740    1055    1547    1765#
CKSWR1  011266           1777#   1789
CKSWR2  011300           1780#
CKSWR3  011304           1782#
CKSWR4  011310           1783#   1791    1798
CKSWR5  011414           1766    1768    1772    1807#
CLKX    001452            341#
CLRC    035702           4609    4660    4711    4762    4813    4864    4915    4966    5017    5068    5119    5170    5221
                         5272    5323    5374    6545#
CNERR   011027            834    1740#
CNT.MA  002302            355     589#    708     710     712    1845
CNVRT = 104417            831     940     942     944     946    1539#   1582    1584    1659    1780
CONERR  010762            829    1740#
CONN    010671           1740#   1996
CONTAB  003330            838     846#
CONVRT= 104416            781     837    1538#   1598    1941
```

```
CORMAX  037234            4501    4533    6572#
CR    = 000015              44#   1135    1145
CREAM   001502             354#    707#   1841#   1842    1844#   1848
CRLF  = 000200              45#   1106    1145
CSR     010363            1740#   1945
CSRMAP  012112            1921#
CYCLE   011460             916     977    1832#
DATABP  006764            1571#   1574    1596    1599#
DATACL= 104413            1535#
DATAHD  006752            1570#   1592    1595#
DDISP = 177570              51#    207#    236
DELAY = 104411            1533#
DEVTAB  003342             804     851#
DH1     036632             386    6572#
DH2     036673             389    6572#
DH3     036731             392     401     413     422     437     440     446    6572#
DH4     036752             395     425    6572#
DH5     037014             398    6572#
DH6     03                 404     416     419     434    6572#
DISPLA  001242             236#    726#    732#   1044#
DISPRE  000174             197#    732
DSWR  = 177570              50#    206#    235
DT1     037112             387    6572#
DT2     037124             390    6572#
DT3     037142             393     414     441     447    6572#
DT4     037154             396     426    6572#
DT5     037172             399     405     417     420     435    6572#
DT6     037210             402     423    6572#
DT7     037222             438    6572#
DZDME = ****** U           382
DZDMG = ****** U           382
EMTVEC= 000030             139#
EM1     035746             385    6572#
EM10    036243             409    6572#
EM11    036307             412     433    6572#
EM12    036321             415    6572#
EM13    036343             418    6572#
EM14    036355             421    6572#
EM15    036367             424    6572#
EM16    036413             427    6572#
EM17    036443             430    6572#
EM2     036013             388    6572#
EM20    036471             436     442    6572#
EM21    036524             439    6572#
EM22    036573             445    6572#
EM3     036054             391    6572#
EM4     036102             394     397    6572#
EM5     036143             400    6572#
EM6     03f165             403    6572#
EM7     036207             406    6572#
ERCT00  002304             591#
ERCT01  002310             594#
ERCT02  002314             597#
ERCT03  002320             600#
ERCT04  002324             603#
ERCT05  002330             606#
```

```
ERC106  002334            609#
ERCTU7  002340            612#
ERCT10  002344            615#
ERCT11  002350            618#
ERCT12  002354            621#
ERCT13  002360            624#
ERCT14  002364            627#
ERCT15  002370            630#
ERCT16  002374            633#
ERCT17  002400            636#
ERR     003244            813     826#    830
ERRMSG  006740           1569#   1587    1590#
ERRPC   003322            832     843#
ERRVEC= 000004            132#    1019    1020#   1022#   1025#
ERTABO  007112           1584    1624#
EXIT  = 000205            159#
EXITER  007046           1610    1615#
FLOAT   003156            806#    812
FY      003202            814#    818     823
GNS   = ****** U         1523    1526    1527    1528    1529    1530    1531    1532    1533    1534    1535    1536    1537
                         1538    1539    1540
HALTS   006770           1554    1601#
HILIM   006052           1368#   1376    1392#
HT    = 000011              42#   1104    1145
INCHAR  011420           1783    1811#
INIFLG  001506             359#    735     765     772#
INLP1   005766           1373#   1382
INPUT = 104415           1537#   1881    1932    1944    1952    2009    2017
INTTY   013456           1962    1981    1997    2166#
IOTVEC= 000020             137#
KMACTV  001470             348#    749     874#    875     1832    1846    1927#   2087#   2093#   2094#   2098    2123    2189#
                         2190
KMCM    011050             840   1740#
KMCR00  002100             504#
KMCR01  002110             509#
KMCR02  002120             514#
KMCR03  002130             519#
KMCR04  002140             524#
KMCR05  002150             529#
KMCR06  002160             534#
KMCR07  002170             539#
KMCR10  002200             544#
KMCR11  002210             549#
KMCR12  002220             554#
KMCR13  002230             559#
KMCR14  002240             564#
KMCR15  002250             569#
KMCR16  002260             574#
KMCR17  002270             579#
KMCSR   002066             487#    800#    815     821#    850     980    1047    1662    1716    1850#   1859    1907    2238
                          2303    2332    2364    2396    2428    2460    2492    2524    2556    2588    2946    6572
KMCSRH  002070             488#   1697#   1698#   1702#   1708#   1709#   1715#   1717#   1859#   1860#   1861
KMCTL   002072             489#   1861#   1862#   1863    2620    2652    2684    2716    2748    2780    2812
KMNUM   001472             349#    703     957    1925#   1939#   2079#   2080    2088    2090
KMP04   002074             490#   1686#   1692    1729    1734    1863#   1864#   1865    2844    4345    4346    4386    4387
                          6519#   6534#
```

```
KMP06   002076            491#   1703*   1865*   1866*   2888
KMRLVL  002060            484#    950     951*   1868*   1869*   1870    6487*
KMRVEC  002056            483#    950*    983    1851*   1852*   1868    6485*
KMS100  002102            505#
KMS101  002112            510#
KMS102  002122            515#
KMS103  002132            520#
KMS104  002142            525#
KMS105  002152            530#
KMS106  002162            535#
KMS107  002172            540#
KMS110  002202            545#
KMS111  002212            550#
KMS112  002222            555#
KMS113  002232            560#
KMS114  002242            565#
KMS115  002252            570#
KMS116  002262            575#
KMS117  002272            580#
KMS200  002104            506#
KMS201  002114            511#
KMS202  002124            516#
KMS203  002134            521#
KMS204  002144            526#
KMS205  002154            531#
KMS206  002164            536#
KMS207  002174            541#
KMS210  002204            546#
KMS211  002214            551#
KMS212  002224            556#
KMS213  002234            561#
KMS214  002244            566#
KMS215  002254            571#
KMS216  002264            576#
KMS217  002274            581#
KMS300  002106            507#
KMS301  002116            512#
KMS302  002126            517#
KMS303  002136            522#
KMS304  002146            527#
KMS305  002156            532#
KMS306  002166            537#
KMS307  002176            542#
KMS310  002206            547#
KMS311  002216            552#
KMS312  002226            557#
KMS313  002236            562#
KMS314  002246            567#
KMS315  002256            572#
KMS316  002266            577#
KMS317  002276            582#
KMTLVL  002064            486#    952     953*   1872*   1873*   6488*
KMTVEC  002062            485#    952*   1870*   1871*   1872    6486*
KM.END  002300            584#   1923    2194
KM.MAP  002100            354     503#    707     774     784    1842    1844    1921    1926    2115    2192    2196
LF    = 000012             43#   1139    1145
```

```
LINE    010573          1740#  2010
LOBITS  006056          1370*  1394#
LOCK    001444           335#  1046*  1058   1060   1401*  1578   2236*  2287*  2305*  2330*  2340*  2362*  2372*
                        2394*  2404*  2426*  2436*  2458*  2468*  2490*  2500*  2522*  2532*  2554*  2564*  2586*
                        2596*  2618*  2628*  2650*  2660*  2682*  2692*  2714*  2724*  2746*  2756*  2778*  2788*
                        2810*  2820*  2842*  2857*  2886*  2901*  2930*  2948*  2973*  2984*  2992*  3015*  3026*
                        3034*  3086*  3108*  3143*  3165*  3200*  3220*  3253*  3273*  3306*  3328*  3363*  3386*
                        3422*  3442*  3475*  3495*  3528*  3548*  3581*  3601*  3634*  3654*  3687*  3707*  3740*
                        3760*  3793*  3813*  3846*  3875*  3909*  3928*  3961*  3986*  4028*  4055*  4557*  4600*
                        4651*  4702*  4753*  4804*  4855*  4906*  4957*  5008*  5059*  5110*  5161*  5212*  5263*
                        5314*  5365*  5416*  5467*  5518*  5569*  5620*  5671*  5722*  5773*  5824*  5875*  5926*
                        5977*  6028*  6079*  6130*  6181*
LOKFLG  001510           361#
LOLIM   006050          1367*  1378   1391#
MASKX   001454           342#
MASTEK  010015          1580   1740#
MCSRX   007745           939   1740#
MDATA   011150          1462   1472   1757#
MEMDAT  035726          4606   4657   4708   4759   4810   4861   4912   4963   5014   5065   5116   5167   5218
                        5269   5320   5371   5422   5473   5524   5575   5626   5677   5728   5779   5830   5881
                        5932   5983   6034   6085   6136   6187   6565#
MEMLD   035602          4560   4605   4656   4707   4758   4809   4860   4911   4962   5013   5064   5115   5166
                        5217   5268   5319   5370   5421   5472   5523   5574   5625   5676   5727   5778   5829
                        5890   5931   5982   6033   6084   6135   6186   6510#
MEMLIM  001466           347#   898*  4531
MEPASS  007620           938   1740#
MERRPC  010072          1583   1740#
MERRX   007772           945   1740#
MERR2   007645          1740#  2100
MERR3   007672           871   1740#
MILK    001504           355#   708*   947   1840*  1845*  1849
MLOCK   007716           917   1740#
MNEW    010017           865   1740#
MODU    010461          1740#  1980
MPASSX  007761           943   1740#
MPFAIL  007562          1658   1675   1740#
MR      007642           919   1740#  1897
MRESET= 004000           159#
MSTCLR= 104410          1532#  1666*  2289   2331   2363   2395   2427   2459   2491   2523   2555   2587   2619
                        2651   2683   2715   2747   2779   2811   2843   2887   2932   2975   3017   3060   3088
                        3145   3202   3255   3308   3365   3424   3477   3530   3583   3636   3689   3742   3795
                        3848   3911   3963   4030   4120   4151   4191   4271   4309   4344   4385   4423   4460
                        4498   4559   4602   4653   4704   4755   4806   4857   4908   4959   5010   5061   5112
                        5163   5214   5265   5316   5367   5418   5469   5520   5571   5622   5673   5724   5775
                        5826   5877   5928   5979   6030   6081   6132   6183   6232   6283   6333
MTITLE  001000           205#   739
MTSTN   010003          1581   1740#  1882
MVECX   007753           941   1740#
NEXT    001442           334#  1400   1620   2235*  2265*  2286*  2329*  2361*  2393*  2425*  2457*  2489*  2521*
                        2553*  2585*  2617*  2649*  2681*  2713*  2745*  2777*  2809*  2841*  2885*  2929*  2972*
                        3014*  3058*  3085*  3142*  3199*  3252*  3305*  3362*  3421*  3474*  3527*  3580*  3633*
                        3686*  3739*  3792*  3845*  3908*  3960*  4027*  4087*  4118*  4149*  4189*  4234*  4269*
                        4307*  4342*  4383*  4421*  4458*  4496*  4556*  4599*  4650*  4701*  4752*  4803*  4854*
                        4905*  4956*  5007*  5058*  5109*  5160*  5211*  5262*  5313*  5364*  5415*  5466*  5517*
                        5568*  5619*  5670*  5721*  5772*  5823*  5874*  5925*  5976*  6027*  6078*  6129*  6180*
                        6230*  6281*  6331*
```

```
NOACT   010731            751    1740#   1834
NODEV   003240            802     824#
NPRSET  035540           4239    4273    4311    4347    4388    4424    4461    4508    6492#
NUM     010323           1740#   1933
OK      003220            816     819#    842
ONE     001464            346#
PACT00  002302            590#
PACT01  002306            593#
PACT02  002312            596#
PACT03  002316            599#
PACT04  002322            602#
PACT05  002326            605#
PACT06  002332            608#
PACT07  002336            611#
PACT10  002342            614#
PACT11  002346            617#
PACT12  002352            620#
PACT13  002356            623#
PACT14  002362            626#
PACT15  002366            629#
PACT16  002372            632#
PACT17  002376            635#
PARBIT= 040000            159#
PERFOR= 004537            159#
PFTAB   007324           1659    1681#
PIRQ    = 177772           49#
PIRQVE= 000240            143#
POPRO   = 012600           154#   1614
POP1SP= 005726            152#
POP2SP= 022626            156#   6300
PRIO    010422           1740#   1961
PRIRTY  013730           2182*   2188*   2201    2222#
PR0     = 000000            66#   2158    2159
PR1     = 000040            67#
PR2     = 000100            68#
PR3     = 000140            69#
PR4     = 000200            70#   2160
PR5     = 000240            71#   2161
PR6     = 000300            72#   2162
PR7     = 000340            73#    178     180     182     184    2163
PS      = 177776            46#     47     700*    909*   2127*   2137*   4095*   4099*   4125*   4129*   4153*   4167*   4193*
                           4209#
PSW     = 177776            47#
PUSHRO= 010046            153#   1611
PUSH1S= 005746            151#
PUSH2S= 024646            155#
PWRVEC= 000024            138#   1637*   1638*   1647*   1653*   1672*   1673*
QV.FLG  001511            362#    706*    956*
RDCHR   = 104402          1253    1526#
RDLIN   = 104403          1326    1527#
RDOCT   = 104404          1375    1528#
RESREG  006766            1597    1600#
RESVEC= 000010             133#
RESOS   = 104407          1531#   1600
ROMCLK= 104412            1534#   1687    1690    1727    1732    3094    3096    3114    3116    3151    3153    3171    3173
                          3207    3209    3225    3227    3260    3262    3278    3280    3314    3316    3334    3336    3371
```

```
                    3373    3392    3394    3429    3431    3447    3449    3482    3484    3500    3502    3535    3537
                    3553    3555    3588    3590    3606    3608    3641    3643    3659    3661    3694    3696    3712
                    3714    3747    3749    3765    3767    3800    3802    3818    3820    3855    3863    3881    3915
                    3917    3933    3935    3969    3973    3975    3992    3996    3998    4037    4041    4043    4061
                    4063    4097    4127    4165    4207    4246    4280    4284    4286    4318    4353    4361    4363
                    4399    4401    4430    4433    4436    4467    4470    4473    4515    4565    4567    4569    4571
                    4612    4616    4618    4663    4667    4669    4714    4718    4720    4765    4769    4771    4816
                    4820    4822    4867    4871    4873    4918    4922    4924    4969    4973    4975    5020    5024
                    5026    5071    5075    5077    5122    5126    5128    5173    5177    5179    5224    5228    5230
                    5275    5279    5281    5326    5330    5332    5377    5381    5383    5428    5432    5434    5479
                    5483    5485    5530    5534    5536    5581    5585    5587    5632    5636    5638    5683    5687
                    5689    5734    5738    5740    5785    5789    5791    5836    5840    5842    5887    5891    5893
                    5938    5942    5944    5989    5993    5995    6040    6044    6046    6091    6095    6097    6142
                    6146    6148    6193    6197    6199    6236    6257    6338    6351    6354    6357    6370    6373
                    6376    6382    6385    6388    6391    6399    6401    6413    6431    6443    6446    6456    6458
                    6461    6501    6517    6520    6537    6548    6550    6558    6560
RUN       001500     352#    709*   1838*   1839*   1846
SAVACT    001474     350#    869    2098*   2190*
SAVNUM    001476     351#    703*    954*    957*   2091*   2184*
SAVPC     001460     344#    830*    845    1407*   1626
SAVSP     001456     343#
SAVOS =   104406    1530#   1559
SCOP1 =   104405    1529#   2244    2298    2311    2339    2346    2371    2378    2403    2410    2435    2442    2467
                    2474    2499    2506    2531    2538    2563    2570    2595    2602    2627    2634    2659    2666
                    2691    2698    2723    2730    2755    2762    2787    2794    2819    2826    2853    2867    2897
                    2911    2941    2954    2983    2991    3000    3025    3033    3042    3104    3124    3161    3181
                    3216    3234    3269    3287    3324    3344    3382    3403    3438    3456    3491    3509    3544
                    3562    3597    3615    3650    3668    3703    3721    3756    3774    3809    3827    3870    3888
                    3924    3942    3982    4005    4050    4070    4523    4578    4625    4676    4727    4778    4829
                    4880    4931    4982    5033    5084    5135    5186    5237    5288    5339    5390    5441    5492
                    5543    5594    5645    5696    5747    5798    5849    5900    5951    6002    6053    6104    6155
                    6206
SETC      035714    5425    5476    5527    5578    5629    5680    5731    5782    5833    5884    5935    5986    6037
                    6088    6139    6190    6555#
SETVEC    035516    4091    4121    4160    4202    6482#
SOFTSW    011452    1781    1820#
SPACNT=   006411    1442*   1466    1469*   1483#
SPDAT     035736    4608    4659    4710    4761    4812    4863    4914    4965    5016    5067    5118    5169    5220
                    5271    5322    5373    5424    5475    5526    5577    5628    5679    5730    5781    5832    5883
                    5934    5985    6036    6087    6138    6189    6568#
SPLD      035636    4562    4607    4658    4709    4760    4811    4862    4913    4964    5015    5066    5117    5168
                    5219    5270    5321    5372    5423    5474    5525    5576    5627    5678    5729    5780    5831
                    5882    5933    5984    6035    6086    6137    6188    6527#
STACK =   001200      37#    701     888    1621    4104    4134    6306
STAT      001450     340#
STAT1     002050     476#   1853*   4154    4194
STAT2     002052     477#   1854*
STAT3     002054     478#   1855*
STKLMT=   177774      48#
STRTSW    001446     339#    741*    744*    745     747    761*    767     769     864     910     917    1875   1898*
                    1928    2111
SVOS      006100    1411#
SWFLG     011416     704*   1776*   1803*   1809#
SWR       001240     235#    725*    727     731*    741     869     874    1014    1032    1056    1548    1553    1609
                    1616    1618    1645    1665*   1705    1765   1802*
SWREG     000176     198#    720*    731    1765    1822
```

```
SW0    = 000001            101#
SW00   = 000001             91#   101     745     761    1928    2111
SW01   = 000002             90#   100     917    1875    1898
SW02   = 000004             89#    99
SW03   = 000010             98#    98     864
SW04   = 000020             87#    97
SW05   = 000040             86#    96
SW06   = 000100             85#    95    1705
SW07   = 000200             84#    94
SW08   = 000400             83#    93    1616
SW09   = 001000             82#    92    1056
SW1    = 000002            100#
SW10   = 002000             81#  1618
SW11   = 004000             80#
SW12   = 010000             79#  1548
SW13   = 020000             78#  1553
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#
TBITVE= 000014             134#
TDATA   025350            4561   4563   4580#
TEMP    011106            1449   1713#  1718#  1724#  1736#  1755#  6233#  6261#  6284#  6291#
TIMER = 104414            1536#
TKVEC = 000060             141#
TLAST = 034660            1901   6572#
TPVEC = 000064             142#
TRAPVE= 000034             140#
TRTVEC= 000014             135#
TST1    013732            1008   1888   1906   2233#
TST10   014612            2425   2455#
TST100  026762            4905   4954#
TST101  027140            4956   5005#
TST102  027316            5007   5056#
TST103  027474            5058   5107#
TST104  027652            5109   5158#
TST105  030040            5160   5209#
TST106  030206            5211   5260#
TST107  030364            5262   5311#
TST11   014710            2457   2487#
TST110  030542            5313   5362#
TST111  030720            5364   5413#
TST112  031076            5415   5464#
TST113  031254            5466   5515#
TST114  031432            5517   5566#
TST115  031610            5568   5617#
TST116  031766            5619   5668#
TST117  032144            5670   5719#
TST12   015006            2489   2519#
TST120  032322            5721   5770#
```

```
TST121  032500            5772    5821#
TST122  032656            5823    5872#
TST123  033034            5874    5923#
TST124  033212            5925    5974#
TST125  033370            5976    6025#
TST126  033546            6027    6076#
TST127  033724            6078    6127#
TST13   015104            2521    2551#
TST130  034102            6129    6178#
TST131  034260            6180    6228#
TST132  034466            6230    6279#
TST133  034660            6281    6329#   6572
TST134= ****** U          6331
TST14   015202            2553    2583#
TST15   015300            2585    2615#
TST16   015376            2617    2647#
TST17   015474            2649    2679#
TST2    014042            2235    2263#
TST20   015572            2681    2711#
TST21   015670            2713    2743#
TST22   015766            2745    2775#
TST23   016064            2777    2807#
TST24   016162            2809    2839#
TST25   016306            2841    2883#
TST26   016432            2885    2927#
TST27   016562            2929    2970#
TST3    014072            2265    2284#
TST30   016722            2972    3012#
TST31   017064            3014    3056#
TST32   017150            3058    3083#
TST33   017350            3085    3140#
TST34   017550            3142    3197#
TST35   017724            3199    3250#
TST36   020100            3252    3303#
TST37   020300            3305    3360#
TST4    014222            2286    2327#
TST40   020510            3362    3419#
TST41   020664            3421    3472#
TST42   021040            3474    3525#
TST43   021214            3527    3578#
TST44   021370            3580    3631#
TST45   021544            3633    3684#
TST46   021720            3686    3737#
TST47   022074            3739    3790#
TST5    014320            2329    2359#
TST50   022250            3792    3843#
TST51   022476            3845    3906#
TST52   022646            3908    3958#
TST53   023114            3960    4025#
TST54   023336            4027    4085#
TST55   023432            4087    4116#
TST56   023524            4118    4147#
TST57   023646            4149    4187#
TST6    014416            2361    2391#
TST60   024014            4189    4232#
TST61   024122            4234    4267#
```

```
TST62   024240          4269    4305#
TST63   024344          4307    4340#
TST64   024502          4342    4381#
TST65   024626          4383    4419#
TST66   024736          4421    4456#
TST67   025044          4458    4493#
TST7    014514          2393    2423#
TST70   025244          4496    4554#
TST71   025360          4556    4597#
TST72   025536          4599    4648#
TST73   025714          4650    4699#
TST74   026072          4701    4750#
TST75   026250          4752    4801#
TST76   026426          4803    4852#
TST77   026604          4854    4903#
TTST    004146           913*    915*   1010#
TWOSYN= 010000           159#
TYPDAT  006754          1575    1593    1596#
TYPE  = 104401           739     751     773     828     833     839     866     871     912     919     938     939     941
                         943     945    1109    1260    1266    1271    1275    1280    1281    1283    1286    1290    1355
                        1357    1373    1380    1381    1432    1472    1523#   1576    1577    1580    1581    1583    1585
                        1589    1594    1658    1674    1779    1782    1834    1880    1897    1903    1940    1960    1974
                        1979    1988    1995    2002    2100
TYPMSG  006654          1573    1576#
VEC     010401          1740#   1953
VECMAP  013172          2099    2111#
VECTR   013724          2181*   2187*   2206    2212*   2220#
WHAT    005770          1366*   1374#
WHERE   006054          1369*   1383    1393#
WHICH   013164          1942    2107#
WRDCNT  006406          1440*   1473*   1481#
WRKO.F  006742          1588    1591#
XBX     006542          1549    1551    1553#
XCSR    004104           940     978#
XERR    004126           946     987#
XHEAD   010077           773    1740#
XPASS   004120           944     984#
XSTATQ  011060           782    1740#
XTSTN   007120          1582    1627#
XVEC    004112           942     981#
ZERO    001462           345#
$APTHD  002034           458     464#
$ASTAT= ****** U        1179    1194
$ATYC   004722          1150    1152#
$ATY1   004676          1148#
$ATY3   004704          1094    1149#
$ATY4   004714          1151#   1604
$AUTOB  001234           232#
$BASE   001372           307#   2186
$BOADR  001222           227#
$BODAT  001226           229#
$CDW1   001376           309#   2183
$CDW2   001400           310#
$CHARC  004672          1111*   1121*   1128    1137*   1142#
$CKSWR= ****** U        1526
$CMTAG  001200           215#
```

```
$CM1  = 000006          247#   248#   249#   250#   251#   252#   253#
$CM2  = 000014          247#   248#   249#   250#   251#   252#   253#
$CM3  = 000006          245#   247
$CM4  = 000005          253#   254#   255#   256#   257#   258#
$CNTLG  005534         1301#
$CNTLU  005527         1275   1300#
$COD  = ****** U          1
$CPUOP  001344          281#
$CRAP = 177777            1#  2223#  2226   2228#  2254#  2257   2258#  2273#  2276   2279#  2316#  2319   2322#
                       2348#  2351   2354#  2380#  2383   2386#  2412#  2415   2418#  2444#  2447   2450#  2476#
                       2479   2482#  2508#  2511   2514#  2540#  2543   2546#  2572#  2575   2578#  2604#  2607
                       2610#  2636#  2639   2642#  2668#  2671   2674#  2700#  2703   2706#  2732#  2735   2738#
                       2764#  2767   2770#  2796#  2799   2802#  2828#  2831   2834#  2872#  2875   2878#  2916#
                       2919   2922#  2959#  2962   2965#  3001#  3004   3007#  3043#  3046   3051#  3072#  3075
                       3078#  3129#  3132   3135#  3186#  3189   3192#  3239#  3242   3245#  3292#  3295   3298#
                       3349#  3352   3355#  3408#  3411   3414#  3461#  3464   3467#  3514#  3517   3520#  3567#
                       3570   3573#  3620#  3623   3626#  3673#  3676   3679#  3726#  3729   3732#  3779#  3782
                       3785#  3832#  3835   3838#  3895#  3898   3901#  3947#  3950   3953#  4014#  4017   4020#
                       4075#  4078   4080#  4106#  4109   4111#  4136#  4139   4142#  4176#  4179   4182#  4222#
                       4225   4227#  4257#  4260   4262#  4295#  4298   4300#  4329#  4332   4335#  4370#  4373
                       4376#  4408#  4411   4414#  4445#  4448   4451#  4482#  4485   4488#  4544#  4547   4549#
                       4584#  4587   4592#  4635#  4638   4643#  4686#  4689   4694#  4737#  4740   4745#  4788#
                       4791   4796#  4839#  4842   4847#  4890#  4893   4898#  4941#  4944   4949#  4992#  4995
                       5000#  5043#  5046   5051#  5094#  5097   5102#  5145#  5148   5153#  5196#  5199   5204#
                       5247#  5250   5255#  5298#  5301   5306#  5349#  5352   5357#  5400#  5403   5408#  5451#
                       5454   5459#  5502#  5505   5510#  5553#  5556   5561#  5604#  5607   5612#  5655#  5658
                       5663#  5706#  5709   5714#  5757#  5760   5765#  5808#  5811   5816#  5859#  5862   5867#
                       5910#  5913   5918#  5961#  5964   5969#  6012#  6015   6020#  6063#  6066   6071#  6114#
                       6117   6122#  6165#  6168   6173#  6216#  6219   6223#  6267#  6270   6274#  6315#  6318
                       6324#
$CRLF   001313          260#  1110   1145   1280   1300   1360   1381   1432   1576   1577   1585   1880   1940
$DDW0   001402          311#  2191
$DDW1   001404          312#
$DDW10  001426          321#
$DDW11  001430          322#
$DDW12  001432          323#
$DDW13  001434          324#
$DDW14  001436          325#
$DDW15  001440          326#
$DDW2   001406          313#
$DDW3   001410          314#
$DDW4   001412          315#
$DDW5   001414          316#
$DDW6   001416          317#
$DDW7   001420          318#
$DDW8   001422          319#
$DDW9   001424          320#
$DEVCT  001326          272#
$DEVM   001374          308#  2189
$DOAGN  004100          955    964    969    975#
$ENDAD  004070          191    737    971#  1607
$ENDCT  004054          966#
$ENV    001336          277#   718    754    756   1089   1157   1181   1601   1767
$ENVM   001337          278#   759   1091   1096   1159
$EOP    003662          934#  6331
$EOPCT  004046          963#   967
```

```
$ERFLG  001203          218#   705*   937*   999    1028   1030*  1051   1558*  1572   1586*  1660*
$ERMAX  001215          224#  1051
$ERROR  006512          181   1547#
$ERRPC  001216          225#   714*   958*  1007*  1555   1557*  1661*
$ERRTB  001512          379#  1568
$ERTTL  001212          222#   949    989   1615*  1857*
$ETABL  001336          276#
$ETEND  001442          329#   470
$FATAL  001320          269#  1185*
$FFLG   005142         1148*  1151*  1179   1188*  1196#
$FILLC  001256          243#  1114   1145
$FILLS  001255          242#  1145
$GDADR  001220          226#
$GDDAT  001224          228#
$GET42  004060          968#
$GTSWR= ****** U       1525
$HD     = 000000         11
$HIBTS  002034          465#
$HIOCT  005722         1348*  1359#
$ICNT   001204          219#  1036*  1037   1039*  1050
$ILLUP  007316         1637   1653   1677#
$INPUT  005724         1364#  1537
$INTAG  001235          233#
$ITEMB  001214          223#  1563*  1603
$LF     001314          261#  1145   1290   1300   1360
$LFLG   005141         1189*  1195#
$LPADR  001206          220#   716*   916*   920   1043*  1045   1050   1620*  1622   1896*  1906*  1908
$LPERR  001210          221#
$MADR1  001350          294#
$MADR2  001354          298#
$MADR3  001360          301#
$MADR4  001364          304#
$MAIL   001316          267#   466    470    1042   1089
$MAMS1  001346          288#  2185
$MAMS2  001352          296#
$MAMS3  001356          299#
$MAMS4  001362          302#
$MBADR  002036          466#
$MFLG   005140         1149*  1155   1190*  1194#
$MNEW   005552         1304#  1782
$MSGAD  001332          274#  1165*  1168
$MSGLG  001334          275#  1170*
$MSGTY  001316          268#  1163   1171*  1183   1187*
$MSWR   005541         1302#  1779
$MTYP1  001347          289#
$MTYP2  001353          297#
$MTYP3  001357          300#
$MTYP4  001363          303#
$MXCNT  004362         1040   1050#
$N      = 000133          1#  2223   2228   2230   2238#  2254   2258   2260   2267#  2273   2279   2281   2289
                       2290#  2316   2322   2324   2331   2332#  2348   2354   2356   2363   2364#  2380   2386
                       2388   2395   2396#  2412   2418   2420   2427   2428#  2444   2450   2452   2459   2460#
                       2476   2482   2484   2491   2492#  2508   2514   2516   2523   2524#  2540   2546   2548
                       2555   2556#  2572   2578   2580   2587   2588#  2604   2610   2612   2619   2620#  2636
                       2642   2644   2651   2652#  2668   2674   2676   2683   2684#  2700   2706   2708   2715
                       2716#  2732   2738   2740   2747   2748#  2764   2770   2772   2779   2780#  2796   2802
```

2804 2811 2812# 2828 2834 2836 2843 2844# 2872 2878 2880 2887 2888#
2916 2922 2924 2932 2933# 2959 2965 2967 2975 2976# 3001 3007 3009
3017 3018# 3043 3051 3053 3060 3061# 3072 3078 3080 3088 3089# 3129
3135 3137 3145 3146# 3186 3192 3194 3202 3203# 3239 3245 3247 3255
3256# 3292 3298 3300 3308 3309# 3349 3355 3357 3365 3366# 3408 3414
3416 3424 3425# 3461 3467 3469 3477 3478# 3514 3520 3522 3530 3531#
3567 3573 3575 3583 3584# 3620 3626 3628 3636 3637# 3673 3679 3681
3689 3690# 3726 3732 3734 3742 3743# 3779 3785 3787 3795 3796# 3832
3838 3840 3848 3849# 3895 3901 3903 3911 3912# 3947 3953 3955 3963
3964# 4014 4020 4022 4030 4031# 4075 4080 4082 4089 4091# 4106 4111
4113 4120 4121# 4136 4142 4144 4151 4152# 4176 4182 4184 4191 4192#
4222 4227 4229 4236 4238# 4257 4262 4264 4271 4272# 4295 4300 4302
4309 4310# 4329 4335 4337 4344 4345# 4370 4376 4378 4385 4386# 4408
4414 4416 4423 4424# 4445 4451 4453 4460 4461# 4482 4488 4490 4498
4499# 4544 4549 4551 4559 4560# 4584 4592 4594 4602 4603# 4635 4643
4645 4653 4654# 4686 4694 4696 4704 4705# 4737 4745 4747 4755 4756#
4788 4796 4798 4806 4807# 4839 4847 4849 4857 4858# 4890 4898 4900
4908 4909# 4941 4949 4950 4951 4959 4960# 4992 5000 5001 5002 5010
5011# 5043 5051 5052 5053 5061 5062# 5094 5102 5103 5104 5112 5113#
5145 5153 5154 5155 5163 5164# 5196 5204 5205 5206 5214 5215# 5247
5255 5256 5257 5265 5266# 5298 5306 5307 5308 5316 5317# 5349 5357
5358 5359 5367 5368# 5400 5408 5409 5410 5418 5419# 5451 5459 5460
5461 5469 5470# 5502 5510 5511 5512 5520 5521# 5553 5561 5562 5563
5571 5572# 5604 5612 5613 5614 5622 5623# 5655 5663 5664 5665 5673
5674# 5706 5714 5715 5716 5724 5725# 5757 5765 5766 5767 5775 5776#
5808 5816 5817 5818 5826 5827# 5859 5867 5868 5869 5877 5878# 5910
5918 5919 5920 5928 5929# 5961 5969 5970 5971 5979 5980# 6012 6020
6021 6022 6030 6031# 6063 6071 6072 6073 6081 6082# 6114 6122 6123
6124 6132 6133# 6165 6173 6174 6175 6183 6184# 6216 6223 6224 6225
6232 6233# 6267 6274 6275 6276 6283 6284# 6315 6324 6325 6326 6333
6334# 6572#

$NULL 001254
$NWTST= 000000

241# 1116 1145
2232# 2262# 2283# 2326# 2358# 2390# 2422# 2454# 2486# 2518# 2550# 2582# 2614#
2646# 2678# 2710# 2742# 2774# 2806# 2838# 2882# 2926# 2969# 3011# 3055# 3082#
3139# 3196# 3249# 3302# 3359# 3418# 3471# 3524# 3577# 3630# 3683# 3736# 3789#
3842# 3905# 3957# 4024# 4084# 4115# 4146# 4186# 4231# 4266# 4304# 4339# 4380#
4418# 4455# 4492# 4553# 4596# 4647# 4698# 4749# 4800# 4851# 4902# 4953# 5004#
5055# 5106# 5157# 5208# 5259# 5310# 5361# 5412# 5463# 5514# 5565# 5616# 5667#
5718# 5769# 5820# 5871# 5922# 5973# 6024# 6075# 6126# 6177# 6227# 6278# 6328#

$OVER 004334
$PASS 001324
$PASTM 002042
$PWRDN 007126
$PWRMG 007312
$PWRUP 007200
$QUES 001312
$RDCHR 005144
$RDDEC= ****** U
$RDLIN 005264
$RDOCT 005564
$RDSZ = 000007
$REGAD 001260
$REG0 001262
$REG1 001264
$REG2 001266
$REG3 001270

1012 1015 1026 1038 1044#
271# 936# 948 960# 961# 978 986 1034 1051 1856#
468#
179 702 1637# 1672 6302 6305
1675#
1647 1653#
259# 1145 1283 1300 1357 1360 1380 1903 1975 1989 2003
1220# 1526
1529
1248# 1527
1321# 1528
1241#
245#
247# 1416# 1421
248# 336# 848 1415# 1422 6572
249# 1414# 1423 6572
250# 1413# 1424

```
$REG4   001272          251#  1412#  1425   6572
$REG5   001274          252#  1411#  1426   6572
$RTNAD  004102          977#
$R2A  = ****** U       1529
$S    = 000135            1#  2235   2238#  2265   2267#  2286   2290#  2329   2332#  2361   2364#  2393   2396#
                       2425   2428#  2457   2460#  2489   2492#  2521   2524#  2553   2556#  2585   2588#  2617
                       2620#  2649   2652#  2681   2684#  2713   2716#  2745   2748#  2777   2780#  2809   2812#
                       2841   2844#  2885   2888#  2929   2933#  2972   2976#  3014   3018#  3058   3061#  3085
                       3089#  3142   3146#  3199   3203#  3252   3256#  3305   3309#  3362   3366#  3421   3425#
                       3474   3478#  3527   3531#  3580   3584#  3633   3637#  3686   3690#  3739   3743#  3792
                       3796#  3845   3849#  3908   3912#  3960   3964#  4027   4031#  4087   4091#  4118   4121#
                       4149   4152#  4189   4192#  4234   4238#  4269   4272#  4307   4310#  4342   4345#  4383
                       4386#  4421   4424#  4458   4461#  4496   4499#  4556   4560#  4599   4603#  4650   4654#
                       4701   4705#  4752   4756#  4803   4807#  4854   4858#  4905   4909#  4956   4960#  5007
                       5011#  5058   5062#  5109   5113#  5160   5164#  5211   5215#  5262   5266#  5313   5317#
                       5364   5368#  5415   5419#  5466   5470#  5517   5521#  5568   5572#  5619   5623#  5670
                       5674#  5721   5725#  5772   5776#  5823   5827#  5874   5878#  5925   5929#  5976   5980#
                       6027   6031#  6078   6082#  6129   6133#  6180   6184#  6230   6233#  6281   6284#  6331
                       6334#
$SAVRE= ****** U       1529
$SAVR6  007322         1646#  1654   1655#  1656#  1679#
$SCOPE  004134          177   1006#  2155
$SETUP= 000000          959   1007   1209   1306
$SVLAD  004316         1023   1041#
$SVPC = 000040          189#   194
$SWR  = 164000            1#    11    258    259    931    959    970    976    978   1000   1001   1002   1003
                       1014   1026   1028   1029   1030   1031   1032   1044   1050   1676   2234   2264   2285
                       2328   2360   2392   2424   2456   2488   2520   2552   2584   2616   2648   2680   2712
                       2744   2776   2808   2840   2884   2928   2971   3013   3057   3084   3141   3198   3261
                       3304   3361   3420   3473   3526   3579   3632   3685   3738   3791   3844   3907   3959
                       4026   4086   4117   4148   4188   4233   4268   4306   4341   4382   4420   4457   4494
                       4555   4598   4649   4700   4751   4802   4853   4904   4955   5006   5057   5108   5159
                       5210   5261   5312   5363   5414   5465   5516   5567   5618   5669   5720   5771   5822
                       5873   5924   5975   6026   6077   6128   6179   6229   6280   6330
$SWREG  001340          279#   720
$SWRMK= 000000         1003
$TESTN  001322          270#  1042#
$TIMES  001310          258#   959#  1031#  1037   1040#  1050   4495#
$TKB    001246          238#  1013   1207   1224   1230   1769   1771   1813   2168
$TKS    001244          237#  1011   1207   1222   1228   1811   2166
$TMP0   001276          253#   775#  1742   2123#  2124#  6234#  6262#  6263
$TMP1   001300          254#   776#  1744
$TMP2   001302          255#   778#   799#   826    835#  1746   1936   1939   2026#
$TMP3   001304          256#   779#  1748   1948   1951   1956   1959   2013   2016   2021   2024
$TMP4   001306          257#   780#  1750   1931#  1943#  2109
$TN   = 000134            1#    11   2232   2234#  2262   2264#  2283   2285#  2326   2328#  2358   2360#  2390
                       2392#  2422   2424#  2454   2456#  2486   2488#  2518   2520#  2550   2552#  2582   2584#
                       2614   2616#  2646   2648#  2678   2680#  2710   2712#  2742   2744#  2774   2776#  2806
                       2808#  2838   2840#  2882   2884#  2926   2928#  2969   2971#  3011   3013#  3055   3057#
                       3082   3084#  3139   3141#  3196   3198#  3249   3251#  3302   3304#  3359   3361#  3418
                       3420#  3471   3473#  3524   3526#  3577   3579#  3630   3632#  3683   3685#  3736   3738#
                       3789   3791#  3842   3844#  3905   3907#  3957   3959#  4024   4026#  4084   4086#  4115
                       4117#  4146   4148#  4186   4188#  4231   4233#  4266   4268#  4304   4306#  4339   4341#
                       4380   4382#  4418   4420#  4455   4457#  4492   4494#  4553   4555#  4596   4598#  4647
                       4649#  4698   4700#  4749   4751#  4800   4802#  4851   4853#  4902   4904#  4953   4955#
                       5004   5006#  5055   5057#  5106   5108#  5157   5159#  5208   5210#  5259   5261#  5310
```

```
                    5312#  5361   5363#  5412   5414#  5463   5465#  5514   5516#  5565   5567#  5616   5618#
                    5667   5669#  5718   5720#  5769   5771#  5820   5822#  5871   5873#  5922   5924#  5973
                    5975#  6024   6026#  6075   6077#  6126   6128#  6177   6179#  6227   6229#  6278   6280#
                    6328   6330#
$TPB     001252      240#  1134*  1145   1552*  1816*  2171*
$TPFLG   001257      244#  1083   1145
$TPS     001250      239#  1132   1145   1550   1814   2169
$TRAP    006414      183   1500#
$TRAP2   006436     1511#  1522
$TRP   = 000021     1515#  1524#  1526   1527#  1528#  1529#  1530#  1531#  1532#  1533#  1534#  1535#  1536#
                    1537#  1538#  1539#  1540#  1541#
$TRPAD   006450     1505   1522#
$TSTM    002040      467#
$TSTNM   001202      217#   715*   999   1041*  1042   1044   1051   1629   1683   1885   1892   1894   2234*
                    2264*  2285*  2328*  2360*  2392*  2424*  2456*  2488*  2520*  2552*  2584*  2616*  2648*
                    2680*  2712*  2744*  2776*  2808*  2840*  2884*  2928*  2971*  3013*  3057*  3084*  3141*
                    3198*  3251*  3304*  3361*  3420*  3473*  3526*  3579*  3632*  3685*  3738*  3791*  3844*
                    3907*  3959*  4026*  4086*  4117*  4148*  4188*  4233*  4268*  4306*  4341*  4382*  4420*
                    4457*  4494*  4555*  4598*  4649*  4700*  4751*  4802*  4853*  4904*  4955*  5006*  5057*
                    5108*  5159*  5210*  5261*  5312*  5363*  5414*  5465*  5516*  5567*  5618*  5669*  5720*
                    5771*  5822*  5873*  5924*  5975*  6026*  6077*  6128*  6179*  6229*  6280*  6330*
$TTYIN   005520     1250   1251   1263   1281   1295   1299#
$TYPBN=  ******  U  1524
$TYPOS=  ******  U  1524
$TYPE    004414     1083#  1176   1515   1523
$TYPEC   004626     1113   1120   1127   1132#  1133
$TYPEX   004674     1138   1140   1143#
$TYPOC=  ******  U  1524
$UNIT    001330      273#
$UNITM   002044      469#
$USWR    001342      280#
$VECT1   001366      305#  2187   2188
$VECT2   001370      306#
$XTSTR   004174     1009   1017#
$Y     = 000000        1#   471#
$$GET4=  000000      970#
$40CAT=  ******  U  1014
.      = 037234      171#   173   176#  189   190#  192#  194#  196#  200#  204#  214#  262   348#
                     349#   350#  351#  360#  448#  454   455#  457#  459#  502#  504#  505#  506#
                     507#   509#  510#  511#  512#  514#  515#  516#  517#  519#  520#  521#  522#
                     524#   525#  526#  527#  529#  530#  531#  532#  534#  535#  536#  537#  539#
                     540#   541#  542#  544#  545#  546#  547#  549#  550#  551#  552#  554#  555#
                     556#   557#  559#  560#  561#  562#  564#  565#  566#  567#  569#  570#  571#
                     572#   574#  575#  576#  577#  579#  580#  581#  582#  743   753   860#  873
                     978   1050  1051  1145  1197#  1207  1299#  1300  1306#  1360  1649  1678  1721#
                    1756#  1758#  1812  1815  1836  1923  1970  2053  2057  2103  2134  2167  2170
                    4506   6311  6417  6435  6450  6465  6473#  6476#
.ADVAN   006060     1400#  1540
.BEGIN   003464      888#
.CNVRT   006170     1433#  1539
.CONVR   006164     1432#  1538
.DATAC   007446     1535   1712#
.DELAY   007332     1533   1685#
.MSTCL   007362     1532   1696#
.RESOS   006132     1421#  1531
.ROMCL   007400     1534   1701#
```

```
.SAVOS  006072            1407#   1530
.SCOP1  004364            1055#   1529
.START  002402             201     700#    716    1799
.TIMER  007512            1536    1723#
.$ASTA= ****** U          1149    1152
.$X    = 002034            454#    459
```

```
COMMEN   144#
ENDCOM   144#
ERROR     38#  2252  2271  2297  2310  2338  2345  2370  2377  2402  2409  2434  2441  2466  2473
             2498  2505  2530  2537  2562  2569  2594  2601  2626  2633  2658  2665  2690  2697  2722
             2729  2754  2761  2786  2793  2818  2825  2852  2866  2896  2910  2940  2953  2982  2990
             2999  3024  3032  3041  3070  3103  3123  3160  3180  3215  3233  3268  3286  3323  3343
             3381  3402  3437  3455  3490  3508  3543  3561  3596  3614  3649  3667  3702  3720  3755
             3773  3808  3826  3869  3887  3923  3941  3981  4004  4049  4069  4101  4103  4131  4133
             4174  4211  4219  4253  4291  4325  4368  4406  4443  4480  4522  4577  4624  4675  4726
             4777  4828  4879  4930  4981  5032  5083  5134  5185  5236  5287  5338  5389  5440  5491
             5542  5593  5644  5695  5746  5797  5848  5899  5950  6001  6052  6103  6154  6205  6243
             6255  6265  6293  6304  6312  6418  6436  6451  6466
ESCAPE   144#
GETPRI   144#
GETSWR   144#
HLT      157#
KMEND      1#   921
KMFRNT     1#
MULT     144#
NEWTST   144#  2232  2262  2283  2326  2358  2390  2422  2454  2486  2518  2550  2583  2614  2646
             2678  2710  2742  2774  2806  2838  2882  2926  2969  3011  3055  3082  3139  3196  3249
             3302  3359  3418  3471  3524  3577  3630  3683  3736  3789  3842  3905  3957  4024  4084
             4115  4146  4186  4231  4266  4304  4339  4380  4418  4455  4492  4553  4596  4647  4698
             4749  4800  4851  4902  4953  5004  5055  5106  5157  5208  5259  5310  5361  5412  5463
             5514  5565  5616  5667  5718  5769  5820  5871  5922  5973  6024  6075  6126  6177  6227
             6278  6328
POP      144#  1191  1192  1349  1665  1666  2214
PUSH     144#  1152  1154  1175  1323  1639  1645  2177
REPORT     1#  144#
SCOPE     39#  2233  2263  2284  2327  2359  2391  2423  2455  2487  2519  2551  2583  2615  2647
             2679  2711  2743  2775  2807  2839  2883  2927  2970  3012  3056  3083  3140  3197  3250
             3303  3360  3418  3472  3525  3578  3631  3684  3737  3790  3843  3906  3958  4025  4085
             4116  4147  4187  4232  4267  4305  4340  4381  4419  4456  4493  4554  4597  4648  4699
             4750  4801  4852  4903  4954  5005  5056  5107  5158  5209  5260  5311  5362  5413  5464
             5515  5566  5617  5668  5719  5770  5821  5872  5923  5974  6025  6076  6127  6178  6228
             6279  6329
SETPRI   144#
SETTRA  1515#  1526  1527  1528  1529  1530  1531  1532  1533  1534  1535  1536  1537  1538  1539
             1540
SETUP    144#
SKIP     144#
SLASH    144#
SPACE    144#
STARS    144#   187   210   262   265   451   453   460   929   996  1068  1147  1206  1212  1241
             1309  1494  1635  1651  2232  2262  2283  2326  2358  2390  2422  2454  2486  2518  2550
             2582  2614  2646  2678  2710  2742  2774  2806  2838  2882  2926  2969  3011  3055  3082
             3139  3196  3249  3302  3359  3418  3471  3524  3577  3630  3683  3736  3789  3842  3905
             3957  4024  4084  4115  4146  4186  4231  4266  4304  4339  4380  4418  4455  4492  4553
             4596  4647  4698  4749  4800  4851  4902  4953  5004  5055  5106  5157  5208  5259  5310
             5361  5412  5463  5514  5565  5616  5667  5718  5769  5820  5871  5922  5973  6024  6075
             6126  6177  6227  6278  6328
SWRSU    144#
TRMTRP  1515#
TYPBIN   144#
TYPDEC   144#
TYPNAM   144#
```

```
TYPNUM   144#
TYPOCS   144#
TYPOCT   144#
TYPTXT   144#
$ADD1      1#  2223
$ADD2      1#  2254
$ADD3      1#  2273  2916
$ADD4      1#  2316  2348  2380  2412  2444  2476  2508  2540  2572  2604  2636  2668  2700  2732
        2764   2796
$ALU       1#  4584  4635  4686  4737  4788  4839  4890  4941  4992  5043  5094  5145  5196  5247
        5298   5349  5400  5451  5502  5553  5604  5655  5706  5757  5808  5859  5910  5961  6012
        6063   6114  6165
$ALUO      1#  4544
$AUTO      1#   785
$BR        1#  3912  3929
$BUFFE     1#  1752
$COMP      1#  6238  6251
$CYCLE     1#  1823
$EOP       1#   921
$ERTBL     1#   381
$FINI      1#  6572
$FLOAT     1#  3090  3109  3147  3166  3204  3221  3257  3274  3310  3329  3367  3387  3426  3443
        3479   3496  3532  3549  3585  3602  3638  3655  3691  3708  3744  3761  3797  3814
$GETPA     1#
$HEADE     1#    11
$INTR      1#  4075  4106
$IR        1#  2959  3001
$MARHI     1#  6401  6458
$MEMFL     1#
$MEMO      1#
$MEM1      1#
$MEM2      1#
$MEM3      1#
$MOCK      1#
$MSG       1#  1740
$NOISE     1#  6315
$NPRBI     1#  4482
$NPR1      1#  4222  4257  4295
$NPR2      1#  4329  4370
$NPR3      1#  4408  4445
$PASEN     1#   935
$PFAIL     1#  1630
$POWER     1#  6267
$PRIO      1#  4136  4176
$PROC1     1#  3043
$PROC2     1#  3072  3129  3186  3239  3292  3349  3408  3461  3514  3567  3620  3673  3726  3779
$PROC3     1#  3832
$PROC4     1#  3895
$PROC5     1#
$QUEST     1#  1932  1944  1952  2009  2017
$RAMCL     1#  1684
$RCLK      1#  1687  1690  1727  1732  3094  3096  3114  3116  3151  3153  3171  3173  3207  3209
        3225   3227  3260  3262  3278  3280  3314  3316  3334  3336  3371  3373  3392  3394  3429
        3431   3447  3449  3482  3484  3500  3502  3535  3537  3553  3555  3588  3590  3606  3608
        3641   3643  3659  3661  3694  3696  3712  3714  3747  3749  3765  3767  3800  3802  3818
        3920   3855  3863  3881  3915  3917  3933  3935  3969  3973  3975  3992  3996  3998  4037
```

```
                4041    4043    4061    4063    4097    4127    4165    4207    4246    4280    4284    4286    4318    4353    4361
                4363    4399    4401    4430    4433    4436    4467    4470    4473    4515    4565    4567    4569    4571    4612
                4616    4618    4663    4667    4669    4714    4718    4720    4765    4769    4771    4816    4820    4822    4867
                4871    4873    4918    4922    4924    4969    4973    4975    5020    5024    5026    5071    5075    5077    5122
                5126    5128    5173    5177    5179    5224    5228    5230    5275    5279    5281    5326    5330    5332    5377
                5381    5383    5428    5432    5434    5479    5483    5485    5530    5534    5536    5581    5585    5587    5632
                5636    5638    5683    5687    5689    5734    5738    5740    5785    5789    5791    5836    5840    5842    5887
                5891    5893    5938    5942    5944    5989    5993    5995    6040    6044    6046    6091    6095    6097    6142
                6146    6148    6193    6197    6199    6236    6257    6338    6351    6354    6357    6370    6373    6376    6382
                6385    6388    6391    6399    6401    6413    6431    6443    6446    6456    6458    6461    6501    6517    6520
                6537    6548    6550    6558    6560
$ROVAR             1#    331
$SCROO             1#   1007
$SCRO1             1#   1046
$SIMBC             1#
$SOFTC             1#   1760
$SPF               1#   3965    3987
$SP1               1#   3947
$SP2               1#   4014
$TIMER             1#   6216
$TSTN              1#   2230    2260    2281    2324    2356    2388    2420    2452    2484    2516    2548    2580    2612    2644
                2676    2708    2740    2772    2804    2836    2880    2924    2967    3009    3053    3080    3137    3194    3247
                3300    3357    3416    3469    3522    3575    3628    3681    3734    3787    3840    3903    3955    4022    4082
                4113    4144    4184    4229    4264    4302    4337    4378    4416    4453    4490    4551    4594    4645    4696
                4747    4798    4849    4900    4951    5002    5053    5104    5155    5206    5257    5308    5359    5410    5461
                5512    5563    5614    5665    5716    5767    5818    5869    5920    5971    6022    6073    6124    6175    6225
                6276    6326
$UPROO             1#   1658
$VARIA             1#    203
$WRFLT             1#   2845    2858    2889    2902
$WR46              1#   2828    2872
$XZ                1#   2223    2228    2254    2258    2273    2279    2316    2322    2348    2354    2380    2386    2412    2418
                2444    2450    2476    2482    2508    2514    2540    2546    2572    2578    2604    2610    2636    2642    2668
                2674    2700    2706    2732    2738    2764    2770    2796    2802    2828    2834    2872    2878    2916    2922
                2959    2965    3001    3007    3043    3051    3072    3078    3129    3135    3186    3192    3239    3245    3292
                3298    3349    3355    3408    3414    3461    3467    3514    3520    3567    3573    3620    3626    3673    3679
                3726    3732    3779    3785    3832    3838    3895    3901    3947    3953    4014    4020    4075    4080    4106
                4111    4136    4142    4176    4182    4222    4227    4257    4262    4295    4300    4329    4335    4370    4376
                4408    4414    4445    4451    4482    4488    4544    4549    4584    4592    4635    4643    4686    4694    4737
                4745    4788    4796    4839    4847    4890    4898    4941    4949    4992    5000    5043    5051    5094    5102
                5145    5153    5196    5204    5247    5255    5298    5306    5349    5357    5400    5408    5451    5459    5502
                5510    5553    5561    5604    5612    5655    5663    5706    5714    5757    5765    5808    5816    5859    5867
                5910    5918    5961    5969    6012    6020    6063    6071    6114    6122    6165    6173    6216    6223    6267
                6274    6315    6324
$$CMRE           208#    247    248    249    250    251    252
$$CMTM           208#    253    254    255    256    257
$$ESCA           144#
$$NEWT           144#   2232    2262    2283    2326    2358    2390    2422    2454    2486    2518    2550    2582    2614    2646
                2678    2710    2742    2774    2806    2838    2882    2926    2969    3011    3055    3082    3139    3196    3249
                3302    3359    3418    3471    3524    3577    3630    3683    3736    3789    3842    3905    3957    4024    4084
                4115    4146    4186    4231    4266    4304    4339    4380    4418    4455    4492    4553    4596    4647    4698
                4749    4800    4851    4902    4953    5004    5055    5106    5157    5208    5259    5310    5361    5412    5463
                5514    5565    5616    5667    5718    5769    5820    5871    5922    5973    6024    6075    6126    6177    6227
                6278    6328
$$SCOP             1#    990
$$SET           1515#   1526    1527    1528    1529    1530    1531    1532    1533    1534    1535    1536    1537    1538    1539
```

```
          1540
$$SKIP     144#
.EQUAT       1#      34
.HEADE       1#
.SETUP       1#
.SACT1       1#     185
.SAPTB       1#     263#
.SAPTH       1#     449
.SAPTY       1#    1145
.SCATC       1#
.SCMTA       1#     208
.SEOP        1#     927
.SERRO       1#
.SERRT       1#
.SPOWE       1#    1633
.SROOC       1#    1307
.SREAD       1#    1204
.SSCOP       1#     994
.STRAP       1#    1492
.STYPE       1#    1066
.STYPO       1#

. ABS.  037234     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 DZKCC,DZKCC/SOL/CRF=DZKCC.MAC,DZKCC.P11
 RUN-TIME: 34 32 2 SECONDS
 RUN-TIME RATIO: 116/69=1.6
 CORE USED:  49K  (98 PAGES)
```