PDP11·70/74 11/70 INST EXR
CEQKCEO

AH·7996E·MC
FICHE 1 OF 2

PDP11·70/74 11/70 INST EXR
CEQKCEO

AH-7996E-MC
FICHE 2 OF 2

digital

.REM @

SEQ 0001

IDENTIFICATION

PRODUCT CODE: AC-7994E-MC

PRODUCT NAME: CEQKCE0 11/70 INSTRUCTION EXERCISER

DATE CREATED: MAY, 1980

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHOR(S): DONALD W. MONROE-REV B
JOHN ADAMS-REV A

CONTENTS

```
                REVISION HISTORY
*******************************************************************************

REV E0: 1)ADDED SUPPORT CODE TO PROVIDE FULL APT SCRIPT-MODE COMPATIBILITY.
        2)REVISED TEST 76 AS DESCRIBED IN TEST DESCRIPTION TO PREVENT
          INTERMITTENT FAILURES.
        3)ADDED 'CLR $ERFLG' TO RELOCATION MONITOR TO PREVENTT ENDLESS
          LOOPING THERE ON OCCURENCE OF DEVICE ERROR.
        4)RAISED PRIORITY OF MBT SERVICE ROUTINE FROM 5 TO 7 TO PREVENT
          INTERMITTENT FAILURES.
```

1.0 ABSTRACT

This program is designed to be a comprehensive check of the PDP-11/70 cpu cluster. The program executes each instruction in all address modes and includes tests for traps, interrupts, the mapping box, memory management, memory, the Unibus, and the Mass Bus. If NOT DESELECTED, the program relocates the test code throughout memory (0-2m). Also, if SELECTED, the program will relocate using available disks (RP03,RK05,RP04,RS03/4). See section 9.5 for a description of relocation.

The main differences between revision A and revision B are routines to use the UBE and MBT (manufacturing only), worst case testing occurs with all switches down, standard SYSMAC macros, and floating point processor tests.

Also, the disk driver was rewritten to make each device have a modular driver and to cause I/O to occur concurrently on the available disks. (see section 9.5.4 for a description of disk drivers)

PRECAUTIONS must be taken to ensure the protection of user disks. Refer to section 7.0 for a description of warnings and exceptions.

2.0 REQUIREMENTS

2.1 Equipment

PDP-11/70 (KB11-B/C) Central Processor with 16K of memory, a line clock, and an LA30 (or equivalent) console.
NOTE: THIS DIAGNOSTIC SUPPORTS THE PDP-11/74, AN IN-HOUSE, EXPERIMENTAL PROCESSOR.

2.1.1 Optional Equipment Used

1. Unibus Exerciser
2. Mass Bus Tester
3. RP11/RP03, RK11/RK05, RH70/RP04, RH70/RS03/RS04
4. FP11-B, FP11-C

2.2 Storage

The program loads into the first 12K of memory and runs in all memory (exclusive of the XXDP monitor if running in chain mode).

2.3 Preliminary Programs

SEQ 0005

Although this program is a test of the CPU cluster, it is

advisable that the CPU cluster (and floating point) diagnostics run first. These consist of:

DEKBA DEKBF
DEKBB DEKBG
DEKBC DEMJA
DEKBD DEFPA
DEKBE DEFPB

3.0 LOADING PROCEDURE

3.1 Method

The program is supplied on the diagnostic media. Refer to the XXDP operating manual for further information.

0 STARTING PROCEDURE

4.1 Console Switch Settings

See Section 5.1

4.2 Starting Addresses

The starting address for the exerciser is 200.

By starting at address 210, the switch register and display lights can be checked. This routine just moves the switches to the display register allowing the operator to toggle the switches and see the corresponding lights in the display register.

By starting at address 214, the micro-break register can be checked. This test requires a maintenance card. See Section 9.0 for further details.

START AT ADDRESS 230 AS AN AID TO CUTTING THE JUMPERS FOR THE PROCESSOR ID REGISTER.

4.3 Program and Operator Action

1. Load program into memory (See Section 3)
2. Check for any system disk packs or configuration exceptions as described in section 7.0.
3. Load address 200
4. Set switches (See Section 5.1)
5. Press Start

6. The program will loop and messages will be typed at the end of each sub-pass and each pass. (see section 8.3 for a description of the messages)

5.0 OPERATING PROCEDURE

5.1 Operational Switch Settings

SW15	HALT ON ERROR	This switch when set will halt the processor when an error is detected. Pressing continue will cause an error message to be typed and the processor will again halt. Pressing continue again will resume testing.
SW14	LOOP ON TEST	This switch when set will cause the program to loop on the current subtest.
SW13	INHIBIT ERROR TYPEOUT	This switch when set inhibits the error typeout.
SW12	INHIBIT UBE	This switch when set inhibits the initialization of the Unibus Exerciser. See section 9.2 for a description of the UBE function.
SW11	INHIBIT SUB-TEST ITERATION	This switch when set inhibits subtest iteration after the first pass. Each subtest is executed 10 times before the next subtest is run. Setting SW11 causes each test to be executed once before starting the next subtest.
SW10	RING BELL ON ERROR	This switch when set will ring the bell when an error is detected.
SW9	LOOP ON ERROR	This switch when set will cause the program to loop on the first failure even if the failure is intermittant. See section 6.1 for a description of looping on relocation errors.
SW8	RELOCATE WITH DISK	This switch when set will CAUSE RELOCATION TO BE DONE BY A DISK INSTEAD OF THE CPU. See section 9.5 for a description of relocation.

SW7	INHIBIT SYSTEM SIZE TYPEOUT	This switch when set will inhibit the typeout of the switch definitions and the disks that will be used for relocation. (Typeout only occurs when the program is dumped)
SW6	INHIBIT RELOCATION	This switch when set will inhibit all relocation. Do not change this switch while the program is running.
SW5	INHIBIT ROUND ROBIN	This switch when set will only relocate using the device selected by switches <2:0> rather than all available devices.
SW4	INHIBIT RANDOM DISK ADDRESS	This switch when set will cause relocation to always start at address 0 on the disk(s).
SW3	INHIBIT MBT	This switch when set inhibits the initialization of the Mass Bus Tester. See section 9.3 for a description of the MBT function.
SW2-SW0	DEVICE CODES	These switches (along with SW5) cause the program to relocate the test code using the device specified below:

VALUE	DEVICE
0	RP11/rp03
1	RK05
2	Not used
3	Not used
4	RH70/RP04
5	RH70/RS03/RS04
6	Not used
7	Not used

NOTE

When relocating via a specific device, set in the value(SW<2:0>) to select the device then set switch 5.

Unit 0 of the load device is marked not present if program was loaded in chair

mode, and therefore will not be used to relocate.

5.2 Display Register

While the program is running, the low byte of the display register contains the subtest number and the high byte contains bits <14:7> of KERNEL PAR0. These bits, of kernel par0, correspond to bits <20:13> of the physical address of the relocated code. When an error is detected and loop on error is selected, the high byte contains the error count.

5.3 Operator Action

When the program is loaded* and started with switch 7 on a zero the program will typeout the disks and unit numbers that will be used for relocation and then wait for the operator to type a character. This is to allow the operator to write protect any drive that is not to be used. If there are no devices available for relocation, operator action is not required.

If the program is loaded via ACT11 in QV or AA or with XXDP in chain mode no operator action is required and all disks not write protected (except for the XXDP media) will be used for relocation.

*Except chain mode, QV(manufacturing only), or Auto Accept (manufacturing only)

6.0 ERRORS

6.1 Error Halts and Description

If an error is detected, the program will trap to the error handling routine ($ERROR). If halt on error is enabled, the processor will halt. Pressing continue will cause an error message to be typed and the processor will halt again.

There are many different types of errors. No matter which type occurs a minimum set of information is typed as follows:

```
HHH:MM:SS
ERRORPC PHYSC PC    PSW    MAINT  TEST NO SUB-PASS CNT
UUUUUU  VVVVVVVV  WWWWWW  XXXXXX  YYYYYY  SSSSSS  PPPPPP
```

where:

```
UUUUUU      = Virtual PC of the error call.
VVVVVVVV    = Physical PC of the error call.
WWWWWW      = PSW at the time of the error call.
```

XXXXXX = Contents of the maintenance register(17777750).
YYYYYY = Test number.
SSSSSS = Sub-pass count (0 thru 5)
PPPPPP = Pass count

HHH:MM:SS Represents the elapsed run time of the program,since the most previous start, where: HHH = hours, MM = minutes, and SS = seconds.

The Virtual PC is the 16 bit word that was pushed on the stack when the error call was made. The physical PC is calculated in one of two ways:

1. If memory management is off the contents of location 'FACTOR' is subtracted from the Virtual PC. This generates the corresponding PC for the non-relocated code.

2. If memory management is on the contents of the appropriate PAR is shifted and added to the Virtual PC to generate a physical 22 bit address. In this case the virtual PC corresponds to the non-relocated code.

The contents of the maintenance register will indicate what memory margin was being performed when the error occurred.

Depending on the type of error additional information is typed as described below.

6.1.1 Unexpected Trap to 4

```
PCOFTP   PHYSPC      PSW       CPUERR
VVVVVV  PPPPPPPP    YYYYYY     ZZZZZZ
```

VVVVVV = Virtual PC that was pushed on the stack when the trap occurred.
PPPPPPPP = Physical PC calculated as described above.
YYYYYY = PSW that was pushed on the stack.
ZZZZZZ = Contents of the CPU error register(17777766).

6.1.2 Unexpected Trap to 114

```
PCOFTP    PHYSPC       PSW      ERRREG       ERR ADR REG
VVVVVV   PPPPPPPP     YYYYYY   ZZZZZZ         EEEEEEEE
```

V, P, and Y = are the same as described in 6.1.1.
ZZZZZZ = Contents of the memory error register (777744).
EEEEEEEE = Contents of the error address registers combined into a 22 bit address (777740 & 777742).

6.1.3 Parity Error During Data Check

This error can only occur during the data check that is made on the relocated test code before it is executed. This check is made by comparing the unrelocated code with the relocated code. The source data refers to the unrelocated code and the

destination data to the relocated code.

```
SRCADR   DSTADR   EADRREG   MEM ERR REG
SSSSSS   DDDDDDDD EEEEEEEE     ZZZZZZ
```

```
SSSSSS     = Virtual address of the source data.
DDDDDDDD   = Physical address of the destination data.
EEEEEEEE   = Contents of the error address registers.
ZZZZZZ     = Contents of memory error register (777744).
```

6.1.4 Error During Data Check-Reloc was by CP

This error is similar to 6.1.3 except instead of a parity error, it is a data comparison error. Refer to section 9.5.3 for a description of CP relocation.

Loop on error (SW<9>) has the following effect:

1. Memory Management Off- If switch<9> is set, looping will be performed on the section relocation (see section 9.5.1). If SW<9> is not set, execution will continue at the beginning of the next section.

2. Memory Management On- If SW<9> is set, looping will be performed on the program relocation (see section 9.5.2) to the same memory space that failed. If SW<9> is not set, program relocation will be retried in the same memory space.

6.1.5 Error During Data Check-Reloc was by I/O

This error is the same as 6.1.4 except relocation was performed via a disk rather than the CP. The error printout will identify which device and drive number transferred the particular word that failed. Refer to section 9.5.4 for a description of I/O relocation.

Loop on error (SW<9>) has the following effect:

1. If SW<9> is set, the device that relocated the word (that caused the data check error) is initiated to do the same transfer with the same disk address and memory addresses. This transfer will continually be initiated and checked until SW<9> is not set.

6.1.6 Device Error

This error occurs if a device error occurs while the device is doing a transfer. The device and drive number are identified and the contents of the device registers are typed.

When SW<9> (loop on error) is set, the device that failed is continually restarted with the same disk address, memory address, and function that caused the error.

If SW<9> is not set, relocation is restarted.

6.1.7 Unibus Exerciser Failed

```
  CC      BUSADR      CR2         CR1       PHYS BUS ADR
XXXXXX    VVVVVV    WWWWWW      YYYYYY       ZZZZZZZZ
```

```
XXXXXX      = Cycle count.
VVVVVV      = Virtual bus address that the UBE failed at
WWWWWW      = Control register number 2
YYYYYY      = Control register number 1
ZZZZZZZZ    = Physical memory address that the UBE failed at
```

The physical memory address is calculated by adding the appropriate map register to the virtual bus address, forming a real 22 bit memory address.

6.1.8 UBE Non-Existant Memory Error

This error only occurs when the 'NO SLAVE SYNC'' error occurs in the unibus exerciser. Only the physical address that timed out is typed. This error might indicate that there is a hole in memory or that the size register (777760) is set wrong.

6.1.9 Mass Bus Tester Failed

```
  CS1      WRDCNT  BUSADR  BADREX    MR2       CS2       ST
AAAAAA     BBBBBB  CCCCCC  DDDDDD  EEEEEE    FFFFFF    GGGGGG

  ER         CS3
HHHHHH     JJJJJJ
```

```
AAAAAA      = Control and status register #1 (760100).
BBBBBB      = Word count register (760102).
CCCCCC      = Bus address register (760104).
DDDDDD      = Bus address extended register (760174).
EEEEEE      = Maintenance register #2 (760106).
FFFFFF      = Control and status register #2 (760110).
GGGGGG      = Status register (760112).
HHHHHH      = Error register (760114).
JJJJJJ      = Control and status register #3 (760176).
```

6.1.10 MBT Non-Existant Memory Error

This is the same as 6.1.7 except that it is detected by the NEXM bit in CS2 of the MBT.

6.1.11 Floating Point Error

This error will only occur if the left and right hand sides of the floating point identities do not agree within the expected tolerance. The value of the calculations are typed out.

This error should only be a function of the Floating Point Processor and the FPP diagnostics (DEFPA DEFPB) should be used to isolate the problem.

6.1.12 Device Hung

This error will occur if a device does not finish its relocation function within 2 seconds after its initiation. If a line clock is not installed, a hung device will hang the program. Refer to section 9.5.4.4 to determine which device and drive is hung.

6.2 Error Recovery

Different types of errors recover in different ways as described below.

6.2.1 Errors Within Subtests

Execution starts with the instruction following the error call.

6.2.2 Relocation with Memory Mgmt. Off

Execution starts at the beginning of the next section.

6.2.3 Device Error or CP Relocation with Memory Mgmt. On

Relocation is restarted.

6.2.4 Unexpected Traps Except Parity (4,10,250)

Execution starts at the address pointed to by location "$LPERR". This location contains the address+2 of the most recently executed "SCOPE" instruction.

6.2.5 Unexpected Parity Error

If the parity error is fatal (Bit 2 or 3 set in error reg) the program types a restart message at restarts. Otherwise, execution starts as in 6.2.4.

7.0 WARNINGS AND EXCEPTIONS

7.1 Warnings

Any drive that is not "write protected" will be written on (except unit 0 of the XXDP load device in chain mode).

When the program is dumped (see section 5.3) and SW<7> is set, the devices and drives that are not write protected will be identified on the terminal. Before typing a character to continue, a drive can be write protected without causing an error because, the system is sized again.

7.2 Exceptions

If any of the devices is located at a non-standard address (see below), the device register address tables (in "common tags") should be changed to the correct addresses. Following is the default address of the control and status register of each device:

RP03----176714
RK05----177404
RP04----176700
RS03/4--172040

If the system has both an RP03 and an RP04, the branch instruction at 100$, in the "size routine" must be replaced by a nop (240) for both devices to be used. This branch is approximately at address 4552.

8.0 MISCELLANEOUS

8.1 Execution Time

The execution time is dependent on the amount of memory on the system. Following are two typical run times:

1. Manufacturing Basic Line-32K memory,UBE, MBT, and no disks---3 minutes.
2. System-128K memory, 2 RK05's, RP04, and 2 RS04's ---9 minutes.

8.2 Stack Pointer

The stack pointer is set to 700.

NOTE

When the program is running in either user or supervisor mode, the user/supervisor stack pointer is set to 700 and the Kernel stack pointer is set to 1200. The Kernel stack pointer is used only for the Error and Interrupt Service routines. routines.

8.3 Pass Count

There are two words used for effective pass count. Location "SUBPASS" and "$PASS". Subpass contains the ASCII representation of the subpass count. This is used to index the PSW table and margin table (see section 8.7).

Six subpasses are executed for each pass. This allows all margins and PSW combinations to be tested before reporting end of pass.

At the end of each subpass the subpass number (that is being started) is typed followed by 'THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789''. If running on ACT11 QV or AA, only the sub-pass number is typed. At the end of each pass the elapsed run time and the message 'END PASS X TOTAL ERRORS SINCE LAST REPORT Y'' is typed.

8.4 Iterations

Sub-test iterations are not performed until the pass count ($PASS) is non-zero. This makes a QV pass as short as possible.

After the first pass, full 10 octal iterations are performed on each subtest.

8.5 T-Bit Trapping

T bit trapping is controlled by the PSW table. The default condition is to run with the T-Bit on during subpasses 2, 4, and 6.

8.6 ACT-11 Compatability

The program is fully ACT-11 compatable.

8.7 PSW and Margin Tables

At the end of the program, just before the messages, are the PSW and margin tables. These tables control what mode and register set and which memory margin will be executed on a subpass. Refer to section 9.5.2 for a description of how these tables are used by the program. These tables may be modified if desired.

8.8 I/O Device Address Modification

To modify the program address of the I/O devices patch the appropriate device table (in the common tags area) to the desired addresses.

If you are patching the RP03 or RP04 see section 7.2.

8.9 Power Fail

If a power fail occurs (followed by a power up), the word 'POWER'' is typed on the terminal and the program restarts.

9.0 PROGRAM DESCRIPTION

The program is divided into 9 sections of position independent relocatable test code. Each section is approximately 1K words long.

When the program is initially loaded and started it will identify itself and type the function of the switch register and the devices and drives that will be used for relocation, if SW7=0. It will also type the CP options available indicator word (OPT.CP). The contents of OPT.CP contain the following indicators:

Bit15	=	Not used
Bit14	=	Not used
Bit13=1/0	=	FPP available/not available
Bit12	=	Not used
Bit11	=	Not used
Bit10=1/0	=	MBT available/not available
Bit09=1/0	=	KW11-L available/not available
Bit08=1/0	=	Console tty available/not available
Bit07=1/0	=	UBE available/not available
Bits06-00	=	Not used

Following is a brief description of each section:

Section 0 This section causes a 256 word 3 Xor 9 test pattern to be relocated throughout memory 0 - 28K.

NOTE: This should not be construed to be a complete memory test.

Section 1 This section tests the unary instruction set executing each unary instruction in each address mode (excluding unary instructions using address mode 7).

Section 2 This section tests the unary instructions using address mode 7 and binaries in all address modes (excluding binary byte ops using address mode 7).

Section 3 This section tests binary byte ops using address mode 7, JMP, JSR and program trap (IOT, TRAP, and EMT) instruction.

Section 4 This section checks that each bit in the processor status word (PSW) can be set cleared, reserved instructions, and odd address traps.

Section 5 This section checks the SXT, XOR, SOB, MARK, RTT and RTT instructions.

Section 6 This section checks the ASH, ASHC, MUL, DIV, SPL instructions and the program interrupt request (PIRQ) logic.

Section 7 This section checks the stack limit register memory management abort logic, the memory management registers, and the maping box registers.

Section 8 This section checks the floating point option, (FP11-B or FP11-C) if available.

Following section 8 are two routines to check the teletype printer logic and a routine to start the KW11-L clock. If the KW11-L is available the priority arbitration logic is tested.

9.1 Micro-Break Test

The micro-break test is used to test the micro break comparators and the stop on micro match function of the maintenance card. To run this test the operator must have a maintenance card installed and start the program at address 214.

The program asks the operator to turn on the stop on micro match switch. It then checks certain bit patterns in the micro break register to ensure the processor does not stop when it is not supposed to.

The processor will then stop with zero in the micro address lights. The operator then hits continue, and the processor will stop with one (1) in the lights. This sequence continues with 2, 4, 10, 20, 40, and 200 appearing in the lights. The program types done when it is finished.

9.2 Unibus Exerciser(UBE)

Any one of 4 UBE's will be used. The program looks for a UBE at addresses 17770004, 17770024, 17770034, and 17770044.

Test 77 will initiate the unibus exerciser if it is present. This is only done on pass 1 - subpass 1, since from that point on, the service routine takes care of restarting it.

The UBE is Set up with a bus address of 0. The function that is loaded is 'DATA IN PAUSE-DATA OUT BYTE". The word count is set for ABOUT 1.3K WORDS. It is also set to interrupt on level 5.

When an interrupt occurs a check is made to see if it was caused by an error. If there was no error, 0 is loaded as the bus address and the UBE is started again.

When an error occurs a check is made to see if it was caused by a memory timeout. If it was, the address in the UBE bus address register is compared with the address in the system size registers. If they are the same (no holes in memory) the UBE is restarted at address 0 and the above sequence is repeated. If the addresses are not the same a memory-hole error is reported.

If the error was not due to a timeout a UBE error is reported.

9.3 Mass Bus Tester(MBT)

Any one of 4 MBT's will be used. The program looks for an MBT at addresses 17770100, 17770200, 17770300, and 17770400. If an MBT is found, the drive type register (17770X26) is checked to make sure that it really is an MBT.

Test 77 also initiates the mass bus tester. Again, this is only done on Pass 1 - subpass 1 since the service routine keeps it running.

The bus address register is initially set to 0, the word count to 2K words, and a read function is initiated.

When an interrupt occurs an error check is made. This error check is the same as that described for the UBE. If there was no error, the word count is reloaded and the function is issued. The bus address register is not changed so it will continue from where it left off.

9.4 Line Clock Initialization

Test 76 turns on the line clock. Two locations in "common tags" keep track of the elapsed run time of the program. When the clock interrupts, the low byte of location "lticks"is incremented. When this byte gets to 60(decimal) it is cleared and the high byte is incremented(seconds). When the second count gets to 60(decimal) location 'mticks" is incremented and lticks is cleared. This gives the timer a 64K decimal minute range.

NOTE

For the UBE , MBT, and Line Clock, when an interrupt occurs, program execution returns to Kernel mode and the Kernel PAR's are mapped down to the 0-12K bank of memory. Upon returning from the interrupt the PAR's are mapped back to where they were and the previous processor mode is restored.

9.5 Relocation Algorithm

9.5.1 Section Relocation

As each section is entered the virtual start address is saved in location 'FRSTAD" and the relocation factor (byte offset from non-relocated code) is calculated and saved in location 'FACTOR". The test code is then executed.

At the end of each section, control is transfered to the "relocation routine". If SW<8> is CLEAR, this routine will relocate the section via the CP (see 9.5.3). If SW<8> is set, the length of the section is calculated, saved as a word count, and control is transfered to the "I/O monitor" (see section 9.5.4) which relocates the section by using a disk.

Each section is initially relocated to the end address of the program. Subsequent relocations start at the end of the previous relocation. For example: if section 0 is 1000 bytes long and the end address of the program is 60000, the first relocation starts at address 60000, the second at 61000, the third at 62000, etc. This continues until 28K has been reached at which time execution goes to the start of the next section and the process repeats with the new section.

Each section is written in position independent code so that it can be relocated and executed without the use of memory management.

9.5.2 Program Relocation

When all nine sections have been relocated and executed thru 28K (see section 9.5.1), memory management is setup according to the value in location 'NEXPAR'. This value is initialized to 600 (or 1600 if running under the XXDP monitor), making relocation start at address 60000 (or 160000). The "I/O monitor" is then entered (see section 9.5.4) to relocate the program. When the I/O monitor completes the relocation, execution is transfered to the start of the program at the relocated position.

Each section is executed only once with memory management on. At the end of section 8, 77 is added to 'NEXPAR' and relocation is performed again. This causes the next relocation to move up by 7700 bytes. For example: If nexpar=1600 the first relocation starts at address 160000, the second at address 167700, the third at 177600, etc.

This continues until the end of memory is reached and constitutes a sub-pass. The PSW and maintenance register (for memory margins) are then setup for the next sub-pass and the program restarts.

The value for the PSW and maintenance registers is taken from

the tables (see section 8.7). The particular entry that is used is obtained by indexing the table by the sub-pass number (see section 8.3). For example, sub-pass 3 uses word 3 (the first word is counted as zero) of each table. Therefore, to change the value in the PSW or maintenance register only requires changing the value in the appropriate table.

The completion of 6 sub-passes constitutes a pass and an end of pass message is typed. The program then restarts in pass 2, sub-pass 0.

9.5.3 Relocation VIA CP

If SW<8> is CLEAR, both section and program relocation (see sections 9.5.1 and 9.5.2), are performed by an instruction move loop rather than a disk. For example:

```
1$:  MOV (R0)+,(R2)+
     CMP R0,R3
     BNE 1$
```

where R0 is the address of the code being moved, R2 is the address that it is being moved to, and R3 is the last address that is to be moved.

When this is finished, the relocated data is checked by an instruction compare loop to ensure that the relocation was performed correctly.

9.5.4 Relocation VIA I/O

If SW<8> is set, both section and program relocation (see section 9.5.1 and 9.5.2), are performed by writing the data to a disk and reading it back to the relocated position. This relocation is controlled by the "I/O Monitor".

9.5.4.1 Section Relocation

When the I/O monitor is entered from the "relocation routine" (see section 9.5.1) a device is selected (see 9.5.4.3), the memory addresses (from and to) and word count are passed to the device handler (see section 9.5.4.4), and the handler is called. When the handler finishes, the I/O monitor checks the relocated data with an instruction compare loop to ensure the relocated data is correct, and returns to the "relocation routine" (see 9.5.1).

9.5.4.2 Program Relocation

When the I/O monitor is entered for program relocation (see section 9.5.2) the base address for the relocation is calculated from the contents of kernel par3 which was set up with memory management (see 9.5.2). If SW<8> is CLEAR, relocation is performed VIA the CP (see section 9.5.3).

If SW<8> is set, a device is selected (see 9.5.4.3), the word count is set to 2K, and the memory addresses (from and to) and word count are passed to the device handler (see 9.5.4.4), and the handler is called. The I/O monitor then adds 2K to the memory addresses, selects another device, passes the addresses to the device handler, and calls the handler. This continues until all 12K has been relocated. The relocated data is then checked with an instruction compare loop. The relocated program is then executed as described in 9.5.2.

9.5.4.3 Device Selection

If SW<5> is not set, an index is picked up from location "DEVINDX". This index is used to index the system size table. The system size table consists of 8 words (one for each device type). Bits <7:0> of each word are used to indicate the drive numbers that are available on the device, and are initialized in the size routine. Bits <15:8> of each word are used to indicate whether the drive has been used for a data transfer (unit used bit).

The system size table is then searched, using the index described above, for a drive that has not been used. When a drive is found, the "unit used bit" is set, the current index is put back in location DEVINDX, and execution continues as described in 9.5.4.1 or 9.5.4.2.

If an unused unit is not found, all the "unit used" bits are cleared and the search is restarted. If the search finds the system size table empty (no devices on the system), the message "NO I/O DEVICES" is typed and relocation is performed via the CP as described in 9.5.3.

If SW<5> is set, SW's<2:0> are used to index the system size table. In this case only one word of the table is used corresponding to the device being selected by SW's<2:0> (see section 5.1). In this mode, a round robin selection is performed on the drives of the selected device.

9.5.4.4 Device Handlers

Each device that is used for relocation has a handler. These handlers are functionally the same.

The handler is called by the I/O Monitor (see section 9.5.4). It first clears the done bit (bit 7) in the handler status word. This prevents the monitor from calling this handler again before it is finished.

If a "device hung" error (see section 6.1.12) is detected, the handler status words can be examined to determine which device did not finish (set bit 7). The drive can then be determined by looking in the "device handler unit number" table. The handler status words and device handler unit number tables,

are located in the "common tags" area of the listing.

Then the handler calculates a disk address. This address is either generated from a random number (SW4=0) or is set to zero (SW4=1). The device ID, unit number, and cylinder address are combined and placed in the "RUN TABLE" (RUNTBL). The position in the run table corresponds to which 2K block of the program is being transfered (i.e. the first 2K block is identified by word 1, the second 2K by word 2, etc.). The bit configuration of each word in the run table is as follows:

<15:13> = Device ID
<12:10> = Unit Number
<9> = not used
<8:0> = Cylinder Address

The track-sector address of the transfer is saved in the 'RUN TRACK TABLE'' (RUNTRAK). The position in this table is as described above. The bit configuration of each word is the same as that for the disk address register for the particular device. Bit 15 is used to indicate a device error. It is set by the device service routine.(see section 9.5.4.5)

The handler then initializes the device registers with all the appropriate information and starts a write function. Execution then returns to the I/O Monitor at the point where the handler was called.

9.5.4.5 Device Service Routines

Each device that is used for relocation has a service routine. These routines are all functionally the same.

The routine is entered by a device interrupt. The device is checked for any errors. If no error occurred the device registers are loaded and the next function to perform is initiated. Three functions are executed: Write, Write Check, and Read. All the necessary bus address information is calculated by the I/O Monitor, so the service routine just takes care of the device.

When the read function has been completed successfully, the done bit (bit 7) in the handler status word is set.

Upon initiation of a function, or completion of all three functions, the service routine returns execution to where it was when it was interrupted.

If an error is detected, the function that failed is retried two more times. If the error is still present the done bit and the error bit (bit 15) is set in the handler status word along with bit 15, in the appropriate entry, in the RUN TRACK TABLE, and the routine exits as described above.

```
 4381
 4846
 5191
 5199
 5243
 5244
 5299
 5333
 5334
 5335
 5336
 5337
 5338
 5339
 5340
 5341
 5342
 5343
 5344
 5345
 5346
 5347
 5348                                .TITLE  CEQKC-E PDP 11/70 CPU EXERCISER
  (1)                                ;*COPYRIGHT (C) 1975, 1980
  (1)                                ;*DIGITAL EQUIPMENT CORP.
  (1)                                ;*MAYNARD, MASS. 01754
  (1)                                ;*
  (1)                                ;*PROGRAM BY DONALD W. MONROE
  (1)                                ;*
  (1)                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
  (1)                                ;*PACKAGE (MAINDEC-11-DZQAC-A5-1).
  (1)                                ;*
```

```
5350
5351
5352
5353
5354
5355
5356
5357
5358
5359
5360
  (1)                   .SBTTL  OPERATIONAL SWITCH SETTINGS
  (1)                   ;*
  (1)                   ;*      SWITCH              USE
  (1)                   ;*      ------              ---
  (1)                   ;*        15          HALT ON ERROR
  (1)                   ;*        14          LOOP ON TEST
  (1)                   ;*        13          INHIBIT ERROR TYPEOUTS
  (1)                   ;*        12          INHIBIT UBE
  (1)                   ;*        11          INHIBIT ITERATIONS
  (1)                   ;*        10          BELL ON ERROR
  (1)                   ;*         9          LOOP ON ERROR
  (1)                   ;*         8          ALLOW RELOCATION VIA I/O DEVICE
5361                    ;*         7          INHIBIT SYSTEM SIZE TYPEOUT
  (1)                   ;*         6          INHIBIT RELOCATION
  (1)                   ;*         5          INHIBIT ROUND ROBIN
5362                    ;*         4          INHIBIT RANDOM DISK ADDRESS
  (1)                   ;*         3          INHIBIT MBT
  (1)                   ;*         2          THESE THREE SWITCHES
5363                    ;*         1          ARE ENCODED TO SELECT RELOCATION
  (1)                   ;*         0          ON THE FOLLOWING DEVICES:
5364                    ;*                 0...RP11/RP03
5365                    ;*                 1...RK11/RK05
5366                    ;*                 2...NOT USED
5367                    ;*                 3...NOT USED
5368                    ;*                 4...RH70/RP04
5369                    ;*                 5...RH70/RS04
5370                    ;*                 6...NOT USED
5371                    ;*                 7...NOT USED
```

```
5373
 (1)                            .SBTTL  BASIC DEFINITIONS
 (1)
 (1)                            ;*INITIAL ADDRESS OF THE STACK POINTER *** 1200 ***
 (1)          001200            STACK=  1200            ;;FIRST ADDRESS OF THE STACK
 (1)          001200            KERSTK= STACK           ;;KERNEL STACK
 (1)          000700            SUPSTK= STACK-300       ;;SUPERVISOR STACK
 (1)          000600            USESTK= STACK-400       ;;USER STACK
 (1)                            .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
 (1)                            .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL
 (1)          177776            PS=     177776          ;;PROCESSOR STATUS WORD
 (1)                            .EQUIV  PS,PSW
 (1)          177774            STKLMT= 177774          ;;STACK LIMIT REGISTER
 (1)          177772            PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
 (1)          177570            SWR=    177570          ;;SWITCH REGISTER
 (1)          177570            DISPLAY=SWR
 (1)
 (1)                            ;*MISCELLANEOUS DEFINITIONS
 (1)          000011            HT=     11              ;;CODE FOR HORIZONTAL TAB
 (1)          000012            LF=     12              ;;CODE LINE FEED
 (1)          000015            CR=     15              ;;CODE CARRIAGE RETURN
 (1)          000200            CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
 (1)
 (1)                            ;*GENERAL PURPOSE REGISTER DEFINITIONS
 (1)          000000            R0=     %0              ;;GENERAL REGISTER
 (1)          000001            R1=     %1              ;;GENERAL REGISTER
 (1)          000002            R2=     %2              ;;GENERAL REGISTER
 (1)          000003            R3=     %3              ;;GENERAL REGISTER
 (1)          000004            R4=     %4              ;;GENERAL REGISTER
 (1)          000005            R5=     %5              ;;GENERAL REGISTER
 (1)          000006            R6=     %6              ;;GENERAL REGISTER
 (1)          000007            R7=     %7              ;;GENERAL REGISTER
 (1)                            .EQUIV  R0,R10          ;;GENERAL REGISTER
 (1)                            .EQUIV  R1,R11          ;;GENERAL REGISTER
 (1)                            .EQUIV  R2,R12          ;;GENERAL REGISTER
 (1)                            .EQUIV  R3,R13          ;;GENERAL REGISTER
 (1)                            .EQUIV  R4,R14          ;;GENERAL REGISTER
 (1)                            .EQUIV  R5,R15          ;;GENERAL REGISTER
 (1)          000006            SP=%6
 (1)                            .EQUIV  SP,KSP          ;;KERNEL STACK POINTER
 (1)                            .EQUIV  SP,SSP          ;;SUPERVISOR STACK POINTER
 (1)                            .EQUIV  SP,USP          ;;USER STACK POINTER
 (1)          000007            PC=%7
 (1)
 (1)                            ;*PRIORITY LEVEL DEFINITIONS
 (1)          000000            PR0=    0               ;;PRIORITY LEVEL 0
 (1)          000040            PR1=    40              ;;PRIORITY LEVEL 1
 (1)          000100            PR2=    100             ;;PRIORITY LEVEL 2
 (1)          000140            PR3=    140             ;;PRIORITY LEVEL 3
 (1)          000200            PR4=    200             ;;PRIORITY LEVEL 4
 (1)          000240            PR5=    240             ;;PRIORITY LEVEL 5
 (1)          000300            PR6=    300             ;;PRIORITY LEVEL 6
 (1)          000340            PR7=    340             ;;PRIORITY LEVEL 7
 (1)
 (1)                            ;*''SWITCH REGISTER'' SWITCH DEFINITIONS
 (1)          100000            SW15=   100000
```

```
(1)             040000                  SW14=   40000
(1)             020000                  SW13=   20000
(1)             010000                  SW12=   10000
(1)             004000                  SW11=   4000
(1)             002000                  SW10=   2000
(1)             001000                  SW09=   1000
(1)             000400                  SW08=   400
(1)             000200                  SW07=   200
(1)             000100                  SW06=   100
(1)             000040                  SW05=   40
(1)             000020                  SW04=   20
(1)             000010                  SW03=   10
(1)             000004                  SW02=   4
(1)             000002                  SW01=   2
(1)             000001                  SW00=   1
(1)                                     .EQUIV  SW09,SW9
(1)                                     .EQUIV  SW08,SW8
(1)                                     .EQUIV  SW07,SW7
(1)                                     .EQUIV  SW06,SW6
(1)                                     .EQUIV  SW05,SW5
(1)                                     .EQUIV  SW04,SW4
(1)                                     .EQUIV  SW03,SW3
(1)                                     .EQUIV  SW02,SW2
(1)                                     .EQUIV  SW01,SW1
(1)                                     .EQUIV  SW00,SW0
(1)
(1)                                     ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
(1)             100000                  BIT15=  100000
(1)             040000                  BIT14=  40000
(1)             020000                  BIT13=  20000
(1)             010000                  BIT12=  10000
(1)             004000                  BIT11=  4000
(1)             002000                  BIT10=  2000
(1)             001000                  BIT09=  1000
(1)             000400                  BIT08=  400
(1)             000200                  BIT07=  200
(1)             000100                  BIT06=  100
(1)             000040                  BIT05=  40
(1)             000020                  BIT04=  20
(1)             000010                  BIT03=  10
(1)             000004                  BIT02=  4
(1)             000002                  BIT01=  2
(1)             000001                  BIT00=  1
(1)                                     .EQUIV  BIT09,BIT9
(1)                                     .EQUIV  BIT08,BIT8
(1)                                     .EQUIV  BIT07,BIT7
(1)                                     .EQUIV  BIT06,BIT6
(1)                                     .EQUIV  BIT05,BIT5
(1)                                     .EQUIV  BIT04,BIT4
(1)                                     .EQUIV  BIT03,BIT3
(1)                                     .EQUIV  BIT02,BIT2
(1)                                     .EQUIV  BIT01,BIT1
(1)                                     .EQUIV  BIT00,BIT0
(1)
(1)                                     ;*BASIC ''CPU'' TRAP VECTOR ADDRESSES
(1)             000004                  ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
```

```
(1)  000010                RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
(1)  000014                TBITVEC=14              ;;"T" BIT
(1)  000014                TRTVEC= 14              ;;TRACE TRAP
(1)  000014                BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
(1)  000020                IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
(1)  000024                PWRVEC= 24              ;;POWER FAIL
(1)  000030                EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
(1)  000034                TRAPVEC=34              ;;"TRAP" TRAP
(1)  000060                TKVEC=  60              ;;TTY KEYBOARD VECTOR
(1)  000064                TPVEC=  64              ;;TTY PRINTER VECTOR
(1)  000114                CACHVEC=114             ;;CACHE ERROR INTERRUPT VECTOR
(1)  000240                PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
(1)  000250                MMVEC=  250             ;;MEMORY MANAGEMENT VECTOR
(1)
(1)                        .SBTTL  CACHE   REGISTER DEFINITIONS
(1)
(1)
(1)  177740                LOADRS = 177740         ;;LOWER 16 BITS OF ADDRESS THAT CAUSED ERROR
(1)  177742                HIADRS = 177742         ;;UPPER SIX BITS OF ADDRESS THAT CAUSED ERROR
(1)  177744                MEMERR = 177744         ;;CACHE ERROR REGISTER
(1)  177746                CONTRL = 177746         ;;MEMORY CONTROL REGISTER
(1)  177750                MAINT  = 177750         ;;MEMORY MAINTENENCE REGISTER
(1)  177752                HITMIS = 177752         ;;HIT MISS REGISTER "1" IMPLIES HIT IN CACHE
(1)
(1)
(1)                        .SBTTL  CPU REGISTER DEFINITIONS
(1)
(1)
(1)  177760                SIZELO = 177760         ;;MEMORY SIZE REGISTER NUMBER TO PUT INTO A PAR
(1)                                                ;;TO GET TO THE LAST 32 WORDS OF MEMORY
(1)  177762                SIZEHI = 177762         ;;HIGH SIZE REGISTER, RESERVED FOR FUTURE USE
(1)                                                ;;CURRENTLY ALL ZERO
(1)  177764                SYSTID = 177764         ;;SYSTEM ID REGISTER
(1)  177766                CPUERR = 177766         ;;CPU ERROR REGISTER HOLDS CONDITION THAT CAUSED
(1)                                                ;;THE TRAP TO ERRVEC (000004)
(1)
(1)
(1)
(1)
(1)                        .SBTTL  MEMORY MANAGEMENT DEFINITIONS
(1)
(1)
(1)                        ;*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES
(1)
(1)  177572                MMR0=   177572
(1)  177574                MMR1=   177574
(1)  177576                MMR2=   177576
(1)  172516                MMR3=   172516
(1)                        .EQUIV  MMR0,SR0
(1)                        .EQUIV  MMR1,SR1
(1)                        .EQUIV  MMR2,SR2
(1)                        .EQUIV  MMR3,SR3
(1)
(1)                        ;*USER "I" PAGE DESCRIPTOR REGISTERS
(1)
(1)  177600                UIPDR0= 177600
```

```
(1)          177602                  UIPDR1= 177602
(1)          177604                  UIPDR2= 177604
(1)          177606                  UIPDR3= 177606
(1)          177610                  UIPDR4= 177610
(1)          177612                  UIPDR5= 177612
(1)          177614                  UIPDR6= 177614
(1)          177616                  UIPDR7= 177616
(1)
(1)                                  ;*USER 'D'' PAGE DESCRIPTOR REGISTORS
(1)
(1)          177620                  UDPDR0= 177620
(1)          177622                  UDPDR1= 177622
(1)          177624                  UDPDR2= 177624
(1)          177626                  UDPDR3= 177626
(1)          177630                  UDPDR4= 177630
(1)          177632                  UDPDR5= 177632
(1)          177634                  UDPDR6= 177634
(1)          177636                  UDPDR7= 177636
(1)
(1)                                  ;*USER ''I'' PAGE ADDRESS REGISTERS
(1)
(1)          177640                  UIPAR0= 177640
(1)          177642                  UIPAR1= 177642
(1)          177644                  UIPAR2= 177644
(1)          177646                  UIPAR3= 177646
(1)          177650                  UIPAR4= 177650
(1)          177652                  UIPAR5= 177652
(1)          177654                  UIPAR6= 177654
(1)          177656                  UIPAR7= 177656
(1)
(1)                                  ;*USER 'D'' PAGE ADDRESS REGISTERS
(1)
(1)          177660                  UDPAR0= 177660
(1)          177662                  UDPAR1= 177662
(1)          177664                  UDPAR2= 177664
(1)          177666                  UDPAR3= 177666
(1)          177670                  UDPAR4= 177670
(1)          177672                  UDPAR5= 177672
(1)          177674                  UDPAR6= 177674
(1)          177676                  UDPAR7= 177676
(1)
(1)                                  ;*SUPERVISOR ''I'' PAGE DESCRIPTOR REGISTERS
(1)
(1)          172200                  SIPDR0= 172200
(1)          172202                  SIPDR1= 172202
(1)          172204                  SIPDR2= 172204
(1)          172206                  SIPDR3= 172206
(1)          172210                  SIPDR4= 172210
(1)          172212                  SIPDR5= 172212
(1)          172214                  SIPDR6= 172214
(1)          172216                  SIPDR7= 172216
(1)
(1)                                  ;*SUPERVISOR 'D'' PAGE DESCRIPTOR REGISTERS
(1)
(1)          172220                  SDPDR0= 172220
(1)          172222                  SDPDR1= 172222
```

```
(1)             172224                  SDPDR2= 172224
(1)             172226                  SDPDR3= 172226
(1)             172230                  SDPDR4= 172230
(1)             172232                  SDPDR5= 172232
(1)             172234                  SDPDR6= 172234
(1)             172236                  SDPDR7= 172236
(1)
(1)                                     ;*SUPERVISOR ''I'' PAGE ADDRESS REGISTERS
(1)
(1)             172240                  SIPAR0= 172240
(1)             172242                  SIPAR1= 172242
(1)             172244                  SIPAR2= 172244
(1)             172246                  SIPAR3= 172246
(1)             172250                  SIPAR4= 172250
(1)             172252                  SIPAR5= 172252
(1)             172254                  SIPAR6= 172254
(1)             172256                  SIPAR7= 172256
(1)
(1)                                     ;*SUPERVISOR 'D'' PAGE ADDRESS REGISTERS
(1)
(1)             172260                  SDPAR0= 172260
(1)             172262                  SDPAR1= 172262
(1)             172264                  SDPAR2= 172264
(1)             172266                  SDPAR3= 172266
(1)             172270                  SDPAR4= 172270
(1)             172272                  SDPAR5= 172272
(1)             172274                  SDPAR6= 172274
(1)             172276                  SDPAR7= 172276
(1)
(1)                                     ;*KERNEL ''I'' PAGE DESCRIPTOR REGISTERS
(1)
(1)             172300                  KIPDR0= 172300
(1)             172302                  KIPDR1= 172302
(1)             172304                  KIPDR2= 172304
(1)             172306                  KIPDR3= 172306
(1)             172310                  KIPDR4= 172310
(1)             172312                  KIPDR5= 172312
(1)             172314                  KIPDR6= 172314
(1)             172316                  KIPDR7= 172316
(1)
(1)                                     ;*KERNEL 'D'' PAGE DESCRIPTOR REGISTERS
(1)
(1)             172320                  KDPDR0= 172320
(1)             172322                  KDPDR1= 172322
(1)             172324                  KDPDR2= 172324
(1)             172326                  KDPDR3= 172326
(1)             172330                  KDPDR4= 172330
(1)             172332                  KDPDR5= 172332
(1)             172334                  KDPDR6= 172334
(1)             172336                  KDPDR7= 172336
(1)
(1)                                     ;*KERNEL ''I'' PAGE ADDRESS REGISTERS
(1)
(1)             172340                  KIPAR0= 172340
(1)             172342                  KIPAR1= 172342
(1)             172344                  KIPAR2= 172344
```

```
(1)          172346                  KIPAR3= 172346
(1)          172350                  KIPAR4= 172350
(1)          172352                  KIPAR5= 172352
(1)          172354                  KIPAR6= 172354
(1)          172356                  KIPAR7= 172356
(1)
(1)                                  ;*KERNEL 'D'' PAGE ADDRESS REGISTERS
(1)
(1)          172360                  KDPAR0= 172360
(1)          172362                  KDPAR1= 172362
(1)          172364                  KDPAR2= 172364
(1)          172366                  KDPAR3= 172366
(1)          172370                  KDPAR4= 172370
(1)          172372                  KDPAR5= 172372
(1)          172374                  KDPAR6= 172374
(1)          172376                  KDPAR7= 172376
(1)
(1)
(1)
(1)                                  .SBTTL  UNIBUS MAP REGISTER DEFINITIONS
(1)
(1)
(1)                                  ;*THE LOWER 16 BITS OF THE MAP REGISTERS ARE LABELED 'MAPLXX'
(1)                                  ;*THE UPPER 6 BITS OF THE MAP REGISTERS ARE LABELED 'MAPHXX'
(1)
(1)
(1)          170200                  MAPL00 = 170200
(1)          170202                  MAPH00 = 170202
(1)          170204                  MAPL01 = 170204
(1)          170206                  MAPH01 = 170206
(1)          170210                  MAPL02 = 170210
(1)          170212                  MAPH02 = 170212
(1)          170214                  MAPL03 = 170214
(1)          170216                  MAPH03 = 170216
(1)          170220                  MAPL04 = 170220
(1)          170222                  MAPH04 = 170222
(1)          170224                  MAPL05 = 170224
(1)          170226                  MAPH05 = 170226
(1)          170230                  MAPL06 = 170230
(1)          170232                  MAPH06 = 170232
(1)          170234                  MAPL07 = 170234
(1)          170236                  MAPH07 = 170236
(1)          170240                  MAPL10 = 170240
(1)          170242                  MAPH10 = 170242
(1)          170244                  MAPL11 = 170244
(1)          170246                  MAPH11 = 170246
(1)          170250                  MAPL12 = 170250
(1)          170252                  MAPH12 = 170252
(1)          170254                  MAPL13 = 170254
(1)          170256                  MAPH13 = 170256
(1)          170260                  MAPL14 = 170260
(1)          170262                  MAPH14 = 170262
(1)          170264                  MAPL15 = 170264
(1)          170266                  MAPH15 = 170266
(1)          170270                  MAPL16 = 170270
```

```
(1)          170272                  MAPH16 = 170272
(1)          170274                  MAPL17 = 170274
(1)          170276                  MAPH17 = 170276
(1)          170300                  MAPL20 = 170300
(1)          170302                  MAPH20 = 170302
(1)          170304                  MAPL21 = 170304
(1)          170306                  MAPH21 = 170306
(1)          170310                  MAPL22 = 170310
(1)          170312                  MAPH22 = 170312
(1)          170314                  MAPL23 = 170314
(1)          170316                  MAPH23 = 170316
(1)          170320                  MAPL24 = 170320
(1)          170320                  MAPH24 = 170320
(1)          170324                  MAPL25 = 170324
(1)          170326                  MAPH25 = 170326
(1)          170330                  MAPL26 = 170330
(1)          170332                  MAPH26 = 170332
(1)          170334                  MAPL27 = 170334
(1)          170336                  MAPH27 = 170336
(1)          170340                  MAPL30 = 170340
(1)          170342                  MAPH30 = 170342
(1)          170344                  MAPL31 = 170344
(1)          170346                  MAPH31 = 170346
(1)          170350                  MAPL32 = 170350
(1)          170352                  MAPH32 = 170352
(1)          170354                  MAPL33 = 170354
(1)          170356                  MAPH33 = 170356
(1)          170360                  MAPL34 = 170360
(1)          170362                  MAPH34 = 170362
(1)          170364                  MAPL35 = 170364
(1)          170366                  MAPH35 = 170366
(1)          170370                  MAPL36 = 170370
(1)          170372                  MAPH36 = 170372
(1)          170374                  MAPL37 = 170374
(1)          170376                  MAPH37 = 170376
(1)                                  .EQUIV  MAPL00,MAPL0
(1)                                  .EQUIV  MAPH00,MAPH0
(1)                                  .EQUIV  MAPL01,MAPL1
(1)                                  .EQUIV  MAPH01,MAPH1
(1)                                  .EQUIV  MAPL02,MAPL2
(1)                                  .EQUIV  MAPH02,MAPH2
(1)                                  .EQUIV  MAPL03,MAPL3
(1)                                  .EQUIV  MAPH03,MAPH3
(1)                                  .EQUIV  MAPL04,MAPL4
(1)                                  .EQUIV  MAPH04,MAPH4
(1)                                  .EQUIV  MAPL05,MAPL5
(1)                                  .EQUIV  MAPH05,MAPH5
(1)                                  .EQUIV  MAPL06,MAPL6
(1)                                  .EQUIV  MAPH06,MAPH6
(1)                                  .EQUIV  MAPL07,MAPL7
(1)                                  .EQUIV  MAPH07,MAPH7
(1)
(1)
(1)
(1)
5374         000000                  AC0=    %0
```

```
                        AC1=    %1
                        AC2=    %2
                        AC3=    %3
                        AC4=    %4
                        AC5=    %5

                ;LINE CLOCK AND PROGRAMMABLE LINE CLOCK REGISTERS
                        PLKCSR=172540
                        PLKCSB=172542
                        PLKVEC=104

                        LKS=177546
                        LKVEC=100

                ;UNIBUS EXERCISER REGISTER
                        UBEDB=  170000          ;DATA BUFFER
                        UBECC=  170002          ;CYCLE COUNT
                        UBEBA=  170004          ;BUS ADDRESS
                        UBECR1= 170006          ;CRONTROL REGISTER 1
                        UBECLR= 170010          ;ERROR CLEAR
                        UBEGO=  170014          ;MULTI-EXERCISER GO
                        UBECR2= 170016          ;CONTROL REGISTER 2
                        UBEVEC= 510             ;INTERRUPT VECTOR

                ;MASS BUS TESTER REGISTERS
                        MBTCS1= 160100
                        MBTWC=  160102
                        MBTBA=  160104
                        MBTMR2= 160106
                        MBTCS2= 160110
                        MBTST=  160112
                        MBTER=  160114
                        MBTAS=  160116
                        MBTDB=  160120
                        MBTMR1= 160124
                        MBTDT=  160126
                        MBTBAE= 160174
                        MBTCS3= 160176
                        MBTVEC= 774
                        MBTPSW= 776

                ;MISCELLANEOUS BIT ASSIGNMENTS (USED IN OPT.CP)
                        KTOPT=  100000                  ;BELOW BIT ASSIGNMENTS ARE USED
                        EISOPT= 040000                  ;IN THE CPCHK ROUTINE
                        FPOPT=  020000                  ;A BIT FOR EACH OPTION PRESENT
                        CISOPT= 010000                  ;1174 CIS OPTION PRESENT BIT
                        MBTOPT= 002000
                        LKOPT=  001000
                        TTOPT=  000400
                        UBEOPT= 000200
                        .EQUIV  ERROR,HLT
                        .EQUIV  BIT14,SM
                        .EQUIV  BIT12,PSM
                        .EQUIV  BIT11,REG
                        CALLHANDLER=10
                        KM=0
```

```
5431            140000                  UM=140000
5432            000000                  PKM=0
5433            030000                  PUM=30000
5434            177770                  UBREAK=177770
5435
5436                                ;OPCODES USED IN 1174 CISP TESTS
5437
5438            076020                  L2D0=   076020          ;LOAD 2 DESCRIPTORS @R0 OPCODE
5439            076061                  L3D1=   076061          ;LOAD 3 DESCRIPTORS @R1 OPCODE
5440            076601                  MED74C= 076601          ;CISP DIAGNOSTIC ENTRY OPCODE
5441            006600                  CISTST= 6600            ;ADDRESS OF A U-DIAGNOSTIC INSTRUCTION
5442            000007                  MFPT=7                  ;OPCODE FOR MFPT INSTRUCTION USED FOR 1174 ONLY
```

```
                                .SBTTL CIS OPCODE DEFINITIONS

                        076021          L2D1    =076021
                        076022          L2D2    =076022
                        076023          L2D3    =076023
                        076024          L2D4    =076024
                        076025          L2D5    =076025
                        076026          L2D6    =076026
                        076027          L2D7    =076027
                        076030          MOVC    =076030
                        076031          MOVRC   =076031
                        076032          MOVTC   =076032
                        076040          LOCC    =076040
                        076041          SKPC    =076041
                        076042          SCANC   =076042
                        076043          SPANC   =076043
                        076044          CMPC    =076044
                        076045          MATC    =076045
                        076050          ADDN    =076050
                        076051          SUBN    =076051
                        076052          CMPN    =076052
                        076053          CVTNL   =076053
                        076054          CVTPN   =076054
                        076055          CVTNP   =076055
                        076056          ASHN    =076056
                        076057          CVTLN   =076057
                        076060          L3D0    =076060
                        076062          L3D2    =076062
                        076063          L3D3    =076063
                        076064          L3D4    =076064
                        076065          L3D5    =076065
                        076066          L3D6    =076066
                        076067          L3D7    =076067
                        076070          ADDP    =076070
                        076071          SUBP    =076071
                        076072          CMPP    =076072
                        076073          CVTPL   =076073
                        076074          MULP    =076074
                        076075          DIVP    =076075
                        076076          ASHP    =076076
                        076077          CVTLP   =076077
                        076130          MOVCI   =076130
                        076131          MOVRCI  =076131
                        076132          MOVTCI  =076132
                        076140          LOCCI   =076140
                        076141          SKPCI   =076141
                        076142          SCANCI  =076142
                        076143          SPANCI  =076143
                        076144          CMPCI   =076144
                        076145          MATCI   =076145
                        076150          ADDNI   =076150
                        076151          SUBNI   =076151
                        076152          CMPNI   =076152
                        076153          CVTNLI  =076153
                        076154          CVTPNI  =076154
                        076155          CVTNPI  =076155
```

```
 5500           076156                          ASHNI   =076156
 5501           076157                          CVTLNI  =076157
 5502           076170                          ADDPI   =076170
 5503           076171                          SUBPI   =076171
 5504           076172                          CMPPI   =076172
 5505           076173                          CVTPLI  =076173
 5506           076174                          MULPI   =076174
 5507           076175                          DIVPI   =076175
 5508           076176                          ASHPI   =076176
 5509           076177                          CVTLPI  =076177
 5510           076600                      .   MED6X   =076600
 5511
 5512
  (1)                                .SBTTL  TRAP CATCHER
  (1)
  (1)           000000                          .=0
  (1)                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ''.+2,HALT''
  (1)                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
  (1)                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
  (1)
  (1)                                .SBTTL  STARTING ADDRESS(ES)
  (1)           000200                          .=200
  (1)
  (1)  000200   000137  003612                  JMP     @#START         ;;JUMP TO STARTING ADDRESS OF PROGRAM
 5513           000210                 .=210
 5514  000210   000137  002614                  JMP     @#START1
 5515  000214   000137  002624                  JMP     @#START2
 5516                                ;;**26-APR-78,G.W.**
 5517  000220   000137  003400                  JMP     @#START3                ;ENTRY FOR PID REG. CUTTING AID
 5518                                ;;******************
 5519           000224                 SAV=.
 5520
 5521           000044                          .=44
 5522  000044   001200                 $APTHE
 5523           000224                 .=SAV
 5524                                ;;*************************************************************************
  (1)
  (1)                                .SBTTL          ACT11 HOOKS
  (1)
  (1)                                ;*THE FOLLOWING LOCATIONS ARE SETUP TO BE USED WITH ACT11
  (1)                                ;*
  (1)                                ;*LOCATION 46 WILL CONTAIN THE ADDRESS OF THE LOCICAL
  (1)                                ;*END OF THE PROGRAM.
  (1)                                ;*LOCATION 52 IS USED TO SPECIFY PROGRAM OPERATING REQUIREMENTS
  (1)                                ;*AND/OR RESTRICTIONS. THIS IS ACCOMPLISHED BY SETTING VARIOUS BITS
  (1)                                ;*TO A ONE OR A ZERO. THE BITS USED AND THERE MEANING ARE:
  (1)                                ;*
  (1)                                ;*      BIT 15=1 PROGRAM SHOULD BE POWER FAILED WHILE RUNNING
  (1)                                ;*            =0 NO POWER FAIL DESIRED
  (1)                                ;*
  (1)                                ;*      BIT 14=1 PROGRAM RUN TIME IS MEMORY SIZE DEPENDENT
  (1)                                ;*            =0 RUN TIME IS NOT MEMORY SIZE DEPENDENT
  (1)                                ;*
  (1)                                ;*      BITS 13-0  MUST BE ZERO'S
  (1)
  (1)           000224                          $SVPC=.                 ;;SAVE LOCATION COUNTER
```

```
  (1)           000046                          .=46                    ;;SET LOCATION COUNTER
  (1)   000046  046652                          .WORD   $ENDAD          ;;SET LOC.46 TO ADDRESS $ENDAD
  (1)           000052                          .=52                    ;;SET LOCATION COUNTER
  (1)   000052  040000                          .WORD   40000           ;;SET LOC.52 TO 40000
  (1)           000224                          .=$SVPC                 ;; RESTORE LOCATION COUNTER
 5536
 5920           001200                  .=1200
 5921                                   ;*APT MAILBOX
 5922                                   ;*******************************************************
 5923   001200  000000                  $APTHE: .WORD   0
 5924   001202  001214                  $MBADR: $MAIL
 5925   001204  002734                  $TSTM:  .WORD   1500.
 5926   001206  002734                  $PASTM: .WORD   1500.
 5927   001210  000000                  $UNITM: .WORD   0
 5928   001212  000014                  $MBLTH: .WORD   14
 5929
 5930
 5931   001214                          $MAIL:
 5932   001214  000000                  $MSGTY: .WORD   0
 5933   001216  000000                  $FATAL: .WORD   0       ;CONTAINS ERROR PC
 5934   001220  000000                  $TESTN: .WORD   0
 5935   001222  000000                  $PAS:   0
 5936   001224  000000                  $DEVCT: 0
 5937   001226  000000                  $UNIT:  0
 5938   001230  000000                  $MSGAD: 0
 5939   001232  000000                  $MSGLG: 0
 5940   001234     000                  $ENV:   .BYTE   0
 5941   001235     000                  $ENVM:  .BYTE   0
 5942   001236  000000                  $APTSW: .WORD   0
 5943   001240  000000                  $USWR:  0
 5944   001242  000006                  $CPUOP: 6
 5945   001244  177570                  $SWRP:  177570
```

```
 5946                                   ;;*****************************************************************************
  (1)
  (1)                                   .SBTTL  COMMON TAGS
  (1)
  (1)                                   ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS .
  (1)                                   ;*USED IN THE PROGRAM.
  (1)
  (1)           001250                          .=1250
  (1)
  (1)   001250                         $CMTAG:                         ;;START OF COMMON TAGS
  (1)   001250  000000                 $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
  (1)   001252  000000                 $TSTNM: .WORD   0               ;;CONTAINS THE TEST NUMBER
  (1)   001254     000                  $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
  (1)           001256                          .EVEN
  (1)   001256  000000                 $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
  (1)   001260  000000                 $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP 1250
  (1)   001262  000000                 $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
  (1)   001264  000000                 $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
  (1)   001266     000                  $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
  (1)   001267     001                  $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
  (1)   001270  000000                 $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
  (1)   001272  000000                 $GDADR: .WORD   0               ;;CONTAINS 1250 OF 'GOOD' DATA
  (1)   001274  000000                 $BDADR: .WORD   0               ;;CONTAINS 1250 OF 'BAD' DATA
  (1)   001276  000000                 $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
  (1)   001300  000000                 $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
  (1)   001302  000000  000000  000000         .WORD   0,0,0           ;;RESERVED--NOT TO BE USED
  (1)   001310  177560                 $TKS:   177560                  ;;TTY KBD STATUS
  (1)   001312  177562                 $TKB:   177562                  ;;TTY KBD BUFFER
  (1)   001314  177564                 $TPS:   177564                  ;;TTY PRINTER STATUS REG. 1250
  (1)   001316  177566                 $TPB:   177566                  ;;TTY PRINTER BUFFER REG. 1250
  (1)   001320     000                  $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
  (1)   001321     002                  $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
  (1)   001322     012                  $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A 'LINE FEED''
  (1)   001323     000                  $TPFLG: .BYTE   0               ;;'TERMINAL AVAILABLE'' FLAG (BIT<07>=0=YES)
  (1)   001324  000000                 $REGAD: .WORD   0               ;;CONTAINS THE 1250 FROM
  (1)                                                                   ;;WHICH  ($REG0) WAS OBTAINED
  (3)   001326  000000                 $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
  (3)   001330  000000                 $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
  (3)   001332  000000                 $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
  (3)   001334  000000                 $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
  (3)   001336  000000                 $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
  (3)   001340  000000                 $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
  (3)   001342  000000                 $REG6:  .WORD   0               ;;CONTAINS (($REGAD)+14)
  (3)   001344  000000                 $REG7:  .WORD   0               ;;CONTAINS (($REGAD)+16)
  (3)   001346  000000                 $REG10: .WORD   0               ;;CONTAINS (($REGAD)+20)
  (3)   001350  000000                 $REG11: .WORD   0               ;;CONTAINS (($REGAD)+22)
  (3)   001352  000000                 $TMP0:  .WORD   0               ;;USER DEFINED
  (3)   001354  000000                 $TMP1:  .WORD   0               ;;USER DEFINED
  (3)   001356  000000                 $TMP2:  .WORD   0               ;;USER DEFINED
  (3)   001360  000000                 $TMP3:  .WORD   0               ;;USER DEFINED
  (3)   001362  000000                 $TMP4:  .WORD   0               ;;USER DEFINED
  (3)   001364  000000                 $TMP5:  .WORD   0               ;;USER DEFINED
  (3)   001366  000000                 $TMP6:  .WORD   0               ;;USER DEFINED
  (3)   001370  000000                 $TMP7:  .WORD   0               ;;USER DEFINED
  (3)   001372  000000                 $TMP10: .WORD   0               ;;USER DEFINED
  (3)   001374  000000                 $TMP11: .WORD   0               ;;USER DEFINED
```

```
(1)  001376  000000                $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
(1)  001400  000000                $ESCAPE:0                       ;;ESCAPE ON ERROR 1250
(1)  001402  177607  000377        $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
(1)  001406     077                $QUES:  .ASCII  /?/             ;;QUESTION MARK
(1)  001407     015                $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
(1)  001410  000012                $LF:    .ASCIZ  <12>            ;;LINE FEED
(3)  001412  000000                ERRRTN: .WORD
(3)  001414  000044                $FLBUFF:.BLKB   44      ;BUFFER FOR FLOATING POINT CONVERSION
(3)  001460  000000                $BUFF:  .WORD           ;
(3)  001462  000000                $AC0:   .WORD           ;EXTENDED EXPONENT VALUES
(3)  001464  000000                $AC1:   .WORD           ;FOR THE SIX FLOATING POINT
(3)  001466  000000                $AC2:   .WORD           ;ACCUMULATORS
(3)  001470  000000                $AC3:   .WORD
(3)  001472  000000                $AC4:   .WORD
(3)  001474  000000                $AC5:   .WORD
(3)  001476  000000                $$TMP4: .WORD
(3)  001500  000000                $$TMP6: .WORD
(3)  001502  000004                FLTMP0: .BLKW   4       ;FLOATING POINT DBL PREC BUFFER
(3)  001512  000004                FLTMP1: .BLKW   4
(3)  001522  001524                TKBFRP: .WORD   TKBFR   ;POINTER FOR KEYBOARD BUFFER
(3)  001524  000011                TKBFR:  .BLKW   11      ;KEYBOARD BUFFER
(3)  001546  000000                NOTYPE: .WORD           ;NO TYPEOUT FLAG (INHIBIT WHEN SET)
(3)  001550  000000                OPT.CP: .WORD           ;CPU OPTION FLAGS
(3)  001552     000                KB11E:  .BYTE   0       ; WITHOUT MP CACHE
(3)  001553     000                KB11EM: .BYTE   0       ; WITH MP CACHE
(3)  001554     000                KB11CM: .BYTE   0       ;KB11CM FLAG (1170 WITH MP MODS)
(3)  001555     000                CISP:   .BYTE   0       ;CISP OPTION PRESENT FLAG
(3)  001556  000004                $SAVPAR:        .BLKW   4       ;USED BY INTERUPT SERVICE ROUTINE
(3)  001566  000000                $SAVPSW:.WORD           ;DITTO
(3)  001570  000006                $RTRN:  RTT             ;RETURN FOR T-BIT TRAP
(3)  001572  000000                VADR:   .WORD           ;BUFFER FOR VIRTUAL ADDRESS
(3)  001574  000000                PA1500: .WORD           ;BUFFER FOR PHYSICAL ADDRESS BITS<15:00>
(3)  001576  000000                PA2116: .WORD           ;PHYSICAL ADDRESS BITS<21:16>
(3)  001600     000                NEXEC:  .BYTE           ;NO EXECUTE FLAG(NO TEST EXECUTION WHEN SET)
(3)  001601     000                MMON:   .BYTE           ;MEMORY MGMT FLAG(MGMT IS ON WHEN NON-ZERO)
(3)  001602     000                QV:     .BYTE           ;QV FLAG(QV PASS WHEN SET)
(3)  001603     000                AA:     .BYTE           ;AUTO ACCEPT FLAG (AA PASS WHEN SET)
(3)  001604  000000                FACTOR: .WORD           ;RELOCATION FACTOR(NUMBER OF
(3)  001606  000000                $FACTOR:.WORD           ;BYTES ABOVE BASE CODE)
(3)  001610  000000                FRSTAD: .WORD           ;FIRST ADDRESS OF SECTION BEING EXECUTED
(3)  001612  000000                FRSTMEM:.WORD           ;ADDRESS OF FIRST FREE MEMORY
(3)  001614  000000                LSTMEM:.WORD            ;ADDRESS OF LAST FREEE MEMORY(IN 28K)
(3)  001616  000000                NEXPAR: .WORD           ;NEXT VALUE TO PUT IN PAR0
(3)  001620  123456                $LONUM: .WORD   123456  ;LOW 16 BITS OF RANDOM NUMBER
(3)  001622  065432                $HINUM: .WORD   65432   ;HIGH 16 BITS OF RANDOM NUMBER
(3)  001624     377  377     377   NULLS:  .BYTE   377,377,377,0   ;BUFFER FOR PRINTER TEST
(3)  001627     000
(3)  001630  000060                SUBPASS:.WORD   60      ;SUB-PASS COUNT IN ASCII
(3)  001632  000000                $ERPSW: .WORD           ;ERROR PSW FOR TYPEOUT
(3)  001634  000000                EXITFL: .WORD
(3)  001636  000000                OLDBASE:.WORD           ;SOURCE BASE ADDRESS FOR DEVICE RELOCATION
(3)  001640  000000                NWBASL: .WORD           ;DEST ADDRESS FOR DEVICE RELOC BITS<15:00>
(3)  001642  000000                NWBASH: .WORD           ;DEST ADDRESS FOR DEVICE RELOC BITS<21:16>
(3)  001644  000000                IOWC:   .WORD           ;TWO'S COMPLIMENT WORD COUNT FOR DEVICE RELOC
(3)  001646  000000                DEVICE: .WORD
(3)  001650  000000                DEVINDX:.WORD           ;DEVICE INDEX (0 TO 7)
```

```
(3)  001652  000000                UNITNO: .WORD           ;DEVICE UNIT NUMBER
(3)  001654  000000                RNTBINX:.WORD           ;INDEX TO RUN TABLE
(3)  001656  000000                MXMMHI: .WORD           ;BITS<21:16> OF LAST MEM ADDRESS ON SYSTEM
(3)  001660  000000                MXMMLO: .WORD           ;BITS<15:00> OF LAST MEM ADDRESS ON SYSTEM
(3)  001662  000000                RP310:  .WORD           ;DATA TO LOAD INTO RP03 CS REGISTER
(3)  001664  000000                RP311:  .WORD           ;RP03 FLAG FOR FIRST 2K OF PROGRAM
(3)  001666  000000                RK10:   .WORD           ;DATA TO LOAD INTO RK05 CS REGISTER
(3)  001670  000000                RK11:   .WORD           ;RK05 FLAG FOR FIRST 2K OF PROGRAM
(3)  001672  000000                RP411:  .WORD           ;RP04 FLAG FOR FIRST 2K OF PROGRAM
(3)  001674  000000                RS11:   .WORD           ;RS04 FLAG FOR FIRST 2K OF PROGRAM
(3)  001676  000000                MTICKS: .WORD           ;ELAPSED RUN TIME IN MINUTES
(3)  001700  000000                LTICKS: .WORD           ;LOW BYTE=NUMBER OF CLOCK INTERRUPTS (0 TO 59)
(3)                                                        ;HIGH BYTE=ELAPSED RUN TIME IN SECONDS(0 TO 59)
(3)  001702  000000                LD2PNT: .WORD   0       ;NEXT 3 WORDS USED FOR CISP DETECTION
(3)  001704  000000                LD2PT1: .WORD   0
(3)  001706  000000                LD3PNT: .WORD   0
(3)  001710  000000                $MAINT: .WORD           ;CURRENT VALUE IN MAINTENANCE REGISTER
(3)  001712  000010                SYSSIZE:.BLKW   10      ;SYSTEM SIZE TABLE(ONE ENTRY FOR EACH DEVICE)
(3)  001732  000007                RUNTBL: .BLKW   7       ;RUN TIME TABLE(ONE ENTRY FOR EACH 2K BLOCK)
(3)  001750  000007                RUNTRAK:.BLKW   7       ;RUN TRACK TABLE(ONE ENTRY FOR EACH 2K BLOCK)
(3)  001766  177777                MAPTBL: .WORD   -1      ;MAP TABLE(ONE BYTE FOR EACH UNIBUS DEVICE)
(3)  001770  177777                        .WORD   -1      ;UNUSED=377, USED=LOW 5 BITS OF MAP ADDRESS
(3)  001772  000002                UBESAV: .BLKW   2       ;BASE ADDRESS OF UBE TRANSFER IN PROGRESS
(3)  001776  000002                UBEADR: .BLKW   2       ;ADDRESS THAT GETS LOADED INTO UBE BA REG
(3)  002002  000002                ERRBA:  .BLKW   2       ;18 BIT UNIBUS ADDRESS WHEN DEVICE DETECTED AN ERROR
(3)                                .SBTTL  DEVICE HANDLER STATUS WORDS
(3)                                ;*      EACH WORD HAS THE FOLLOWING BIT ASSIGNMENTS:
(3)                                ;*              7       HANDLER READY
(3)                                ;*              8       REPEAT LAST FUNCTION
(3)                                ;*              15      ERROR
(3)  002006  000200                RP3HSTAT:.WORD  200     ;RP03
(3)  002010  000200                RKHSTAT:.WORD   200     ;RK05
(3)  002012  000200                SPARE0: .WORD   200
(3)  002014  000200                SPARE1: .WORD   200
(3)  002016  000200                RP4HSTAT:.WORD  200     ;RP04
(3)  002020  000200                RSHSTAT:.WORD   200     ;RS04
(3)  002022  000200                        .WORD   200     ;SPARE
(3)  002024  000200                        .WORD   200     ;SPARE
(3)
(3)                                .SBTTL  DEVICE HANDLER WORD COUNTS
(3)                                ;*      THIS TABLE GETS LOADED BY THE I/O
(3)                                ;*      RELOCATION ROUTINE WITH THE TWO'S COMPLIMENT WORD
(3)                                ;*      COUNT FOR THE TRANSFER FOR THE PARTICULAR DEVICE.
(3)  002026  000000                RP3HWC: .WORD           ;RP03
(3)  002030  000000                RKHWC:  .WORD           ;RK05
(3)  002032  000000                        .WORD           ;SPARE
(3)  002034  000000                        .WORD           ;SPARE
(3)  002036  000000                RP4HWC: .WORD           ;RP04
(3)  002040  000000                RSHWC:  .WORD           ;RS04
(3)
(3)                                .SBTTL  DEVICE HANDLER OLD BASE ADDRESS
(3)                                ;*      THIS TABLE GETS LOADED BY THE I/O RELOCATION ROUTINE
(3)                                ;*      WITH THE BASE ADDRESS OF THE SOURCE DATA FOR THE
(3)                                ;*      DEVICE THAT IS GOING TO TRANSFER THE DATA.
(3)  002042  000000                RP3OLD: .WORD           ;RP03
(3)  002044  000000                        .WORD
```

```
(3)  002046  000000                RKOLD:  .WORD           ;RK05
(3)  002050  000000                        .WORD
(3)  002052  000000                        .WORD           ;SPARE
(3)  002054  000000                        .WORD           ;SPARE
(3)  002056  000000                        .WORD
(3)  002060  000000                        .WORD           ;SPARE
(3)  002062  000000                RP4OLD: .WORD           ;RP04
(3)  002064  000000                        .WORD
(3)  002066  000000                RSOLD:  .WORD           ;RS04
(3)  002070  000000                        .WORD
(3)
(3)                                .SBTTL  DEVICE HANDLER NEW BASE ADDRESSES
(3)                                ;*      THIS TABLE GETS LOADED BY THE I/O RELOCATION ROUTINE
(3)                                ;*      WITH THE BASE ADDRESS OF THE DESTINATION FOR THE
(3)                                ;*      PARTICULAR DEVICE THAT IS GOING TO DO THE TRANSFER.
(3)  002072  000000                RP3NWL: .WORD           ;RP03
(3)  002074  000000                RP3NWH: .WORD
(3)  002076  000000                RKNEWL: .WORD           ;RK05
(3)  002100  000000                RKNEWH: .WORD
(3)  002102  000000                        .WORD           ;SPARE
(3)  002104  000000                        .WORD
(3)  002106  000000                        .WORD           ;SPARE
(3)  002110  000000                        .WORD
(3)  002112  000000                RP4NWL: .WORD           ;RP04
(3)  002114  000000                RP4NWH: .WORD
(3)  002116  000000                RSNEWL: .WORD           ;RS04
(3)  002120  000000                RSNEWH: .WORD
(3)
(3)                                .SBTTL  DEVICE HANDLER UNIT NUMBER
(3)                                ;*      THIS TABLE GETS LOADED BY THE I/O RELOCATION ROUTINE.
(3)                                ;*      IT TELLS THE DEVICE HANDLER WHICH UNIT NUMBER IS
(3)                                ;*      TO DO THE TRANSFER.
(3)  002122  000000                RP3UNIT:.WORD           ;RP03
(3)  002124  000000                RKUNIT: .WORD           ;RK05
(3)  002126  000000                        .WORD           ;SPARE
(3)  002130  000000                        .WORD           ;SPARE
(3)  002132  000000                RP4UNIT:.WORD           ;RP04
(3)  002134  000000                RSUNIT: .WORD           ;RS04
(3)
(3)                                .SBTTL  ADDRESS OF THE DEVICE HANDLERS
(3)                                ;*      THIS TABLE CONTAINS THE ADDRESS OF THE DEVICE HANDLER
(3)                                ;*      ROUTINES. IT IS USED BY THE I/O RELOCATION ROUTINE
(3)                                ;*      TO TRANSFER CONTROL TO THE DEVICE HANDLER.
(3)  002136  046672                RP3HANA:.WORD   RP3DRV  ;RP03
(3)  002140  047310                RKHANA: .WORD   RKDRV   ;RK05
(3)  002142  000000                        .WORD           ;SPARE
(3)  002144  000000                        .WORD           ;SPARE
(3)  002146  047704                RP4HANA:.WORD   RP4DRV  ;RP04
(3)  002150  050254                RSHANA: .WORD   RSDRV   ;RS04
(3)
(3)                                .SBTTL  DEVICE HANDLER DISK ADDRESS TABLE
(3)                                ;*      THIS TABLE GETS LOADED BY THE DEVICE HANDLER WITH THE
(3)                                ;*      DISK ADDRESS(SECTOR AND CYLINDER) OF THE CURRENT
(3)                                ;*      TRANSFER.
(3)  002152  000000                RP3HDA: .WORD           ;RP03 DISK ADDRESS
(3)  002154  000000                RP3HDC: .WORD           ;RP03 DESIRED CYLINDER
```

```
(3)  002156  000000                RKHDA:  .WORD           ;RK05 DISK ADDRESS
(3)  002160  000000                        .WORD           ;SPARE
(3)  002162  000000                RP4HDA: .WORD
(3)  002164  000000                RP4HDC: .WORD           ;RP04 DESIRED CYLINDER
(3)  002166  000000                RSHDA:  .WORD           ;RS04 DISK ADDRESS
(3)
(3)                                .SBTTL  DEVICE HANDLER FUNCTION TABLE
(3)                                ;*      THIS TABLE GETS LOADED BY THE DEVICE HANDLERS
(3)                                ;*      AND THE DEVICE SERVICE ROUTINES. IT TELLS THE ROUTINES
(3)                                ;*      WHICH FUNCTION TO DO NEXT.
(3)  002170  000000                RP3FUN: .WORD           ;RP03
(3)  002172  000000                RKFUN:  .WORD           ;RK05
(3)  002174  000000                        .WORD           ;SPARE
(3)  002176  000000                RP4FUN: .WORD           ;RP04
(3)  002200  000000                RSFUN:  .WORD           ;RS04
(3)
(3)                                .SBTTL  DEVICE HANDLER RETRY COUNT
(3)                                ;*      THIS TABLE GETS LOADED BY THE DEVICE HANDLERS AND IS USED
(3)                                ;*      BY THE DEVICE SERVICE ROUTINES. IF AN ERROR OCCURS
(3)                                ;*      THE DEVICE SERVICE ROUTINE WILL RETRY THE FUNCTION UNTIL
(3)                                ;*      THE BYTE IN THIS TABLE GOES TO ZERO. IT IS INITIALIZED
(3)                                ;*      TO A -3.
(3)  002202     000                RP3TRY: .BYTE           ;RP03
(3)  002203     000                RKTRY:  .BYTE           ;RK05
(3)  002204     000                        .BYTE           ;SPARE
(3)  002205     000                RP4TRY: .BYTE           ;RP04
(3)  002206     000                RSTRY:  .BYTE           ;RS04
(3)          002210                        .EVEN
(3)
(3)                                .SBTTL  DEVICE REGISTER TABLES
(3)                                ;*      THE FOLLOWING TABLES CONTAIN THE STANDARD ADDRESS FOR
(3)                                ;*      THE DEVICES USED BY THIS PROGRAM. IF A DEVICE IS PLACED
(3)                                ;*      AT A NON-STANDARD ADDRESS THE APPROPRIATE TABLE CAN BE
(3)                                ;*      CHANGED AND THE PROGRAM WILL OPERATE THAT DEVICE.
(3)                                ;*
(3)                                ;*      EXCEPTION--SEE DOCUMENTATION FOR RP03 AND RP04 PROBLEMS.
(3)                                .SBTTL  RP11/RP03 REGISTERS
(3)  002210  176710                RP3DS:  .WORD   176710  ;DRIVE STATUS
(3)  002212  176712                RP3ER:  .WORD   176712  ;ERROR REGISTER
(3)  002214  176714                RP3CS:  .WORD   176714  ;CONTROL AND STATUS
(3)  002216  176716                RP3WC:  .WORD   176716  ;WORD COUNT
(3)  002220  176720                RP3BA:  .WORD   176720  ;BUS ADDRESS
(3)  002222  176724                RP3DA:  .WORD   176724  ;DISK ADDRESS
(3)  002224  176722                RP3DC:  .WORD   176722  ;DESIRED CYLINDER
(3)  002226  000254                RP3VEC: .WORD   254     ;INTERRUPT VECTOR
(3)  002230  000256                RP3PSW: .WORD   256     ;INTERRUPT VECTOR+2
(3)
(3)                                .SBTTL RK11/RK05 REGISTERS
(3)  002232  177400                RKDS:   .WORD   177400  ;DRIVE STATUS
(3)  002234  177402                RKER:   .WORD   177402  ;ERROR REGISTER
(3)  002236  177404                RKCS:   .WORD   177404  ;CONTROL AND STATUS
(3)  002240  177406                RKWC:   .WORD   177406  ;WORD COUNT
(3)  002242  177410                RKBA:   .WORD   177410  ;BUS ADDRESS
(3)  002244  177412                RKDA:   .WORD   177412  ;DISK ADDRESS
(3)  002246  000220                RKVEC:  .WORD   220     ;INTERRUPT VECTOR
(3)  002250  000222                RKPSW:  .WORD   222     ;INTERRUPT VECTOR+2
```

```
(3)
(3)                                     .SBTTL  RH70/RP04 REGISTE ;
(3)  002252  176700                     RP4CS1: .WORD   176700  ;CONTROL AND STATUS #1
(3)  002254  176702                     RP4WC:  .WORD   176702  ;WORD COUNT
(3)  002256  176704                     RP4BA:  .WORD   176704  ;BUS ADDRESS
(3)  002260  176750                     RP4BAE: .WORD   176750  ;BUS ADDRESS EXTENDED
(3)  002262  176706                     RP4DA:  .WORD   176706  ;DISK ADDRESS
(3)  002264  176710                     RP4CS2: .WORD   176710  ;CONTROL AND STATUS #2
(3)  002266  176752                     RP4CS3: .WORD   176752  ;CONTROL AND STATUS #3
(3)  002270  176712                     RP4DS:  .WORD   176712  ;DRIVE STATUS
(3)  002272  176714                     RP4ER1: .WORD   176714  ;ERROR REG #1
(3)  002274  176734                     RP4DC:  .WORD   176734  ;DESIRED CYLINDER
(3)  002276  176740                     RP4ER2: .WORD   176740  ;ERROR REG #2
(3)  002300  176742                     RP4ER3: .WORD   176742  ;ERROR REG #3
(3)  002302  176736                     RPCC:   .WORD   176736  ;CURRENT CYLINDER
(3)  002304  176732                     RP4OF:  .WORD   176732  ;OFFSET REGISTER
(3)  002306  000254                     RP4VEC: .WORD   254     ;INTERRUPT VECTOR
(3)  002310  000256                     RP4PSW: .WORD   256     ;INTERRUPT VECTOR+2
(3)
(3)                                     .SBTTL  RH70/RS04 REGISTERS
(3)  002312  172040                     RSCS1:  .WORD   172040  ;CONTROL AND STATUS #1
(3)  002314  172042                     RSWC:   .WORD   172042  ;WORD COUNT
(3)  002316  172044                     RSBA:   .WORD   172044  ;BUS ADDRESS
(3)  002320  172070                     RSBAE:  .WORD   172070  ;BUS ADDRESS EXTENDED
(3)  002322  172046                     RSDA:   .WORD   172046  ;DISK ADDRESS
(3)  002324  172050                     RSCS2:  .WORD   172050  ;CONTROL AND STATUS #2
(3)  002326  172072                     RSCS3:  .WORD   172072  ;CONTROL AND STATUS #3
(3)  002330  172052                     RSDS:   .WORD   172052  ;DRIVE STATUS
(3)  002332  172054                     RSER:   .WORD   172054  ;ERROR REG
(3)  002334  000204                     RSVEC:  .WORD   204     ;INTERRUPT VECTOR
(3)  002336  000206                     RSPSW:  .WORD   206     ;INTERRUPT VECTOR+2
(3)
(3)                                     .SBTTL  UNIBUS EXERCISER REGISTER ADDRESS TABLE
(3)                                     ;*      THIS TABLE IS ASSEMBLED FOR UBE #0. IF THE UBE
(3)                                     ;*      ADDRESSES ARE CUT FOR OTHER THAN UNIT #0, THE PROGRAM
(3)                                     ;*      WILL CHANGE THIS TABLE. THE PROGRAM LOOKS FOR A
(3)                                     ;*      UBE AT ADDRESSES 770002, 770022, 770032, AND 770042.
(3)  002340  170002                     UBETBL: .WORD   UBECC   ;CYCLE COUNT
(3)  002342  170004                             .WORD   UBEBA   ;BUS ADDRESS REG
(3)  002344  170016                             .WORD   UBECR2  ;CONTROL REGISTER #2
(3)  002346  170006                             .WORD   UBECR1  ;CONTROL REGISTER #1
(3)  002350  170010                             .WORD   UBECLR  ;UBE CLEAR ADDRESS
(3)  002352  000510                             .WORD   UBEVEC  ;INTERRUPT VECTOR
(3)  002354  000512                             .WORD   UBEVEC+2        ;INTERRUPT VECTOR +2
(3)
(3)                                     .SBTTL  MASS BUS TESTER REGISTER ADDRESSES
(3)                                     ;*      THE PROGRAM IS ASSEMBLED WITH ADDRESSES FOR A MBT
(3)                                     ;*      AT 770100. IF THE MBT IS AT ANOTHER ADDRESS THE PROGRAM
(3)                                     ;*      WILL CHANGE THIS TABLE. THE PROGRAM LOOKS FOR A UBE
(3)                                     ;*      AT ADDRESSES 770100, 770200, 770300, AND 770400.
(3)  002356  160100                     MBTTBL: .WORD   MBTCS1  ;CONTROL AND STATUS #1
(3)  002360  160102                             .WORD   MBTWC   ;WORD COUNT
(3)  002362  160104                             .WORD   MBTBA   ;BUS ADDRESS
(3)  002364  160174                             .WORD   MBTBAE  ;BUS ADDRESS EXTENDED
(3)  002366  160106                             .WORD   MBTMR2  ;MAINTENANCE REGISTER #2
(3)  002370  160110                             .WORD   MBTCS2  ;CONTROL REGISTER #2
```

```
(3)  002372  160112                        .WORD   MBTST   ;STATUS REGISTER
(3)  002374  160114                        .WORD   MBTER   ;ERROR REGISTER
(3)  002376  160176                        .WORD   MBTCS3  ;CONTROL REGISTER #3
(3)  002400  000774                        .WORD   MBTVEC  ;INTERRUPT VECTOR
(3)  002402  000776                        .WORD   MBTPSW  ;INTERRUPT VECTOR+2
(3)  002404  160126                        .WORD   MBTDT   ;DRIVE TYPE REGISTER
(3)  002406  160200              MBTN2:    .WORD   160200  ;MASS BUS TESTER #2
(3)  002410  160300              MBTN3:    .WORD   160300  ;MASS BUS TESTER #3
(3)  002412  160400              MBTN4:    .WORD   160400  ;MASS BUS TESTER #4
(3)
```

```
(2)
(1)
(1)                             .SBTTL  ERROR POINTER TABLE
(1)
(1)                             ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
(1)                             ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
(1)                             ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
(1)                             ;*NOTE1:        IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
(1)                             ;*NOTE2:        EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
(1)
(1)                             ;*      EM              ;;POINTS TO THE ERROR MESSAGE
(1)                             ;*      DH              ;;POINTS TO THE DATA HEADER
(1)                             ;*      DT              ;;POINTS TO THE DATA
(1)                             ;*      DF              ;;POINTS TO THE DATA FORMAT
(1)
(1)
(1)  002414                     $ERRTB:
5947                                    ;ITEM 1
5948  002414  065637                    EM1             ;UNEXPECTED TRAP TO 4
5949  002416  065664                    DH1             ;PCOFTP  PHYSC      PSW  CPUERR
5950  002420  065730                    DT1             ;VADR,VADR,$TMP0,$TMP2
5951  002422  065723                    DF1             ;0,1,0,0,0
5952                                    ;ITEM 2
5953  002424  065742                    EM2             ;UNEXPECTED TRAP TO 10
5954  002426  065770                    DH2             ;PCOFTP  PHYSPC  PSW
5955  002430  066016                    DT2             ;VADR,VADR,$TMP0
5956  002432  065723                    DF1
5957                                    ;ITEM 3
5958  002434  066026                    EM3             ;UNEXPECTED TRAP TO 250(MGMT)
5959  002436  066063                    DH3             ;PCOFTP  PHYSPC    PSW    MMR0    MMR2
5960  002440  066132                    DT3             ;VADR,VADR,$TMP0,,$TMP2,,$TMP3
5961  002442  065723                    DF1
5962                                    ;ITEM 4
5963  002444  066146                    EM4             ;UNEXPECTED TRAP TO 114
5964  002446  066175                    DH4             ;PCOFTP  PHYSPC   PSW    ERADREG MEMERRREG
5965  002450  066132                    DT3             ;VADR,VADR,$TMP0,$TMP3,$TMP2
5966  002452  066253                    DF4             ;0,1,0,2,0
5967                                    ;ITEM 5
5968  002454  066260                    EM5             ;PARITY ERROR DURING DATA CHECK
5969  002456  066317                    DH5             ;SRCADR DSTADR ERRADREG  MEM ERR REG
5970  002460  066370                    DT5             ;$TMP0,PA1500,$TMP3,$TMP2
5971  002462  066364                    DF5
5972                                    ;ITEM 6
5973  002464  066402                    EM6             ;ERROR DURING CHECK OF RELOCATED DATA
5974  002466  066452                    DH6             ;SRCADR  DSTADR
5975  002470  066472                    DT6             ;$TMP0,PA1500
5976  002472  066253                    DF4
5977                                    ;ITEM 7
5978  002474  000000                    0
5979  002476  000000                    0
5980  002500  000000                    0
5981  002502  000000                    0
5982                                    ;ITEM 10
5983  002504  066500                    EM10            ;ERROR DURING DATA CHECK-RELOC WAS BY I/O
5984  002506  066551                    DH10            ;SRCADR   DSTADR   DEVICE THAT DID XFER
5985  002510  066624                    DT10            ;$TMP0,VADR,$TMP2,$TMP3
```

```
 5986  002512  066620                    DF10            ;0,1,3,0
 5987                                    ;ITEM 11
 5988  002514  066636                    EM11            ;BIT(S) STUCK IN MICRO-BREAK REG
 5989  002516  066703                    DH11            ;GOOD DAT  BAD DAT
 5990  002520  066726                    DT11            ;$TMP0,$TMP1
 5991  002522  066724                    DF11            ;0,0
 5992                                    ;ITEM 12
 5993  002524  066734                    EM12            ;UNIBUS EXERCISER NON-EXISTANT MEMOREY
 5994  002526  066772                    DH12            ;PHYSICAL ADDRESS
 5995  002530  067010                    DT12            ;PA1500
 5996  002532  067006                    DF12            ;2
 5997                                    ;ITEM 13
 5998  002534  067014                    EM13            ;MASS BUS TESTER NON-EXISTANT MEMORY
 5999  002536  067052                    DH13            ;PHYSICAL ADDRESS
 6000  002540  067010                    DT12
 6001  002542  067006                    DF12
 6002                                    ;ITEM 14
 6003  002544  067067                    EM14            ;FLOATING POINT ERROR
 6004  002546  067114                    DH14            ;       DATA1           DATA2
 6005  002550  067134                    DT14            ;$TMP4,$REG2,$TMP6,$REG3
 6006  002552  067146                    DF14            ;4,0,4,0
 6007                                    ;ITEM 15
 6008  002554  067152                    EM15            ;DEVICE HUNG
 6009  002556  000000                    0
 6010  002560  000000                    0
 6011  002562  000000                    0
 6012                                    ;ITEM 16
 6013  002564  067067                    EM14            ;FLOATING POINT ERROR
 6014  002566  067166                    DH16
 6015  002570  067220                    DT16            ;FLTMP0,$REG2,FLTMP1,$REG3
 6016  002572  067213                    DF16            ;5,0,5,0
 6017                                    ;ITEM 17
 6018  002574  067232                    EM17            ;R0 FAILED TO LOAD CORRECTLY ON MFPT
 6019  002576  066703                    DH11            ;GOOD DAT  BAD DAT
 6020  002600  066726                    DT11            ;$TMP0,$TMP1
 6021  002602  066724                    DF11            ;0,0
 6022
 6023                                    ;ITEM 20
 6024  002604  067276                    EM20            ;CIS INSTRUCTION FAILURE
 6025  002606  066452                    DH6
 6026  002610  066726                    DT11
 6027  002612  066724                    DF11
```

```
 6055  002614  013737  177570  177570  START1: MOV     @#SWR,@#DISPLAY
 6056  002622  000774                          BR      START1
 6057                                          .SBTTL  PROGRAM INITIALIZATION
 6058
 6059                                  ;;*****************************************************************
 6060                                  .SBTTL  MICRO-BREAK REGISTER TEST
 6061                                  ;*THIS TEST IS EXECUTED BY STARTING THE PROGRAM AT ADDRESS 214.
 6062                                  ;*THIS TEST REQUIRES A MAINTENANCE CARD AND OPERATOR INTERVENTION.
 6063                                  ;*THE PROCESSOR SHOULD STOP 8 TIMES. FOLLOWING IS THE DATA
 6064                                  ;*THAT SHOULD BE IN THE MICRO-ADRESS DATA LIGHTS EACH TIME:
 6065                                  ;*      1       000
 6066                                  ;*      2       001
 6067                                  ;*      3       002
 6068                                  ;*      4       004
 6069                                  ;*      5       010
 6070                                  ;*      6       020
 6071                                  ;*      7       040
 6072                                  ;*      8       200
 6073                                  ;;*****************************************************************
 6074  002624  012706  001100          START2: MOV     #1100,SP        ;SETUP THE SP
 6075  002630  012737  062270  000034          MOV     #$TRAP,@#TRAPVEC        ;SETUP TRAP VECTOR
 6076  002636  012737  054710  000030          MOV     #$ERROR,@#EMTVEC;SETUP EMT VECTOR
 6077  002644  012700  000377                  MOV     #377,R0         ;PUT MICRO-BREAK DATA IN R0
 6078  002650  005737  001552                  TST     @#KB11E         ;IS THIS A KB11-E OR KB11-EM PROCESSOR?
 6079  002654  001402                          BEQ     1$              ;BR IF NOT. 8 BIT U-BREAK REGISTER
 6080  002656  012700  177777                  MOV     #177777,R0      ;KB11-E AND KB11-EM HAVE 16 BIT U-BREAK REGISTER
 6081  002662  010037  177770          1$:     MOV     R0,@#UBREAK     ;LOAD U BREAK REG
 6082  002666  020037  177770                  CMP     R0,@#UBREAK     ;LOAD OK?
 6083  002672  001036                          BNE     UBRERR          ;BRANCH IF NO
 6084  002674  005000                          CLR     R0
 6085  002676  010037  177770                  MOV     R0,@#UBREAK
 6086  002702  020037  177770                  CMP     R0,@#UBREAK
 6087  002706  001030                          BNE     UBRERR
 6088  002710  012700  000125                  MOV     #125,R0
 6089  002714  005737  001552                  TST     @#KB11E         ;IS THIS A KB11-E OR KB11EM PROCESSOR?
 6090  002720  001402                          BEQ     2$              ;BR IF NOT. 8 BIT U-BREAK REGISTER
 6091  002722  012700  052525                  MOV     #52525,R0       ;KB11-E AND KB11-EM HAVE 16 BIT U-BREAK REGISTER
 6092  002726  010037  177770          2$:     MOV     R0,@#UBREAK
 6093  002732  020037  177770                  CMP     R0,@#UBREAK
 6094  002736  001014                          BNE     UBRERR
 6095  002740  012700  000252                  MOV     #252,R0
 6096  002744  005737  001552                  TST     @#KB11E         ;IS THIS A KB11-E OR KB11-EM PROCESSOR?
 6097  002750  001402                          BEQ     3$              ;BR IF NOT. 8 BIT U-BREAK REGISTER
 6098  002752  012700  125252                  MOV     #125252,R0      ;KB11-E AND KB11-EM HAVE 16 BIT U-BREAK REGISTER
 6099  002756  010037  177770          3$:     MOV     R0,@#UBREAK
 6100  002762  020037  177770                  CMP     R0,@#UBREAK
 6101  002766  001411                          BEQ     UBRK2
 6102  002770  010067  176356          UBRERR: MOV     R0,$TMP0
 6103  002774  013737  177770  001354          MOV     @#UBREAK,@#$TMP1
 6104  003002  012737  002624  001262          MOV     #START2,@#$LPERR
 6105  003010  104011                          ERROR   11
 6106                                  ;TEST TO ENSURE U BREAK COMPARATORS DO NOT COME ON.
 6107  003012  012737  000100  177770  UBRK2:  MOV     #100,@#UBREAK           ;PUT SAFE VALUE IN REG
 6108  003020  104400  003026                  TYPE    ,65$            ;;TYPE ASCIZ STRING
  (1)  003024  000421                          BR      64$             ;;GET OVER THE ASCIZ
  (1)                             ;;65$:  .ASCIZ  /SET MAINT TO STOP ON MICRO-BREAK/<CRLF>
```

```
  (1) 003070                          64$:
6109  003070  104400  003076                TYPE    ,67$                ;;TYPE ASCIZ STRING
  (1) 003074  000407                        BR      66$                 ;;GET OVER THE ASCIZ
  (1)                               ;;67$:  .ASCIZ  /HIT CONTINUE/<CRLF>
  (1) 003114                        66$:
6110  003114  000000                        HALT
6111  003116  012737  000012  000010        MOV     #12,@#RESVEC
6112  003124  012737  000002  000012        MOV     #2,@#RESVEC+2
6113  003132  012705  003172                MOV     #2$,R5                      ;SET UP R5 FOR MARK INSTR
6114  003136  012701  000010                MOV     #10,R1              ;SET SOB COUNT
6115  003142  012702  003347                MOV     #UBRTBL+1,R2        ;GET ADRS OF UBREAK DATA TABLE
6116  003146  112237  177770          1$:   MOVB    (R2)+,@#UBREAK      ;LOAD MICRO-BREAK FROM TABLE
6117  003152  000010                        10                          ;EXEC RES INSTR (ROM ADRS 000)
6118  003154  005037  177770                CLR     @#UBREAK
6119  003160  077106                        SOB     R1,1$               ;CONTINUE
6120  003162  012737  000125  177770        MOV     #125,@#UBREAK       ;SET MICRO-BREAK DATA PATTERN
6121  003170  006400                        MARK    0                   ;EXEC MARK (ROM ADRS 252)
6122  003172  005037  177770          2$:   CLR     @#UBREAK
6123  003176  012706  001100                MOV     #1100,SP            ;RESTORE SP
6124  003202  012737  000006  000004        MOV     #6,@#ERRVEC
6125  003210  012737  000002  000006        MOV     #2,@#ERRVEC+2
6126  003216  052737  040000  177776        BIS     #BIT14,@#PSW        ;GO TO SUPER MODE
6127  003224  012706  000700                MOV     #700,SP             ;SET SUPER SP
6128  003230  012746  003252                MOV     #3$,-(SP)           ;SETUP STACKK FOR JSR INSTR
6129  003234  005000                        CLR     R0                  ;SETUP R0
6130  003236  012701  000007                MOV     #7,R1               ;SET SOB COUNT
6131  003242  012702  003362                MOV     #INSTBL+2,R2        ;GET ADRS OF TABLE OF INSTRUCTIONS
6132  003246  012217                  4$:   MOV     (R2)+,(PC)          ;GET INSTRUCTION
6133  003250  000000                        .WORD                       ;EXECUTE INSTRUCTION
6134  003252  077103                  3$:   SOB     R1,4$               ;CONTINUE
6135  003254  012737  000100  177770        MOV     #100,@#UBREAK       ;PUT SAFE VALUE IN UBREAK REG
6136  003262  005000                        CLR     R0
6137  003264  012702  003346                MOV     #UBRTBL,R2
6138  003270  012703  003360                MOV     #INSTBL,R3
6139  003274  012701  000010                MOV     #10,R1
6140  003300  012746  003314                MOV     #5$,-(SP)
6141  003304  112237  177770          6$:   MOVB    (R2)+,@#UBREAK      ;LOAD UBREAK REG FROM TABLE
6142  003310  012317                        MOV     (R3)+,(PC)          ;GET INSTR FROM TABLE
6143  003312  000000                        .WORD                       ;EXECUTE INSTR. PROCESSOR SHOULD STOP
6144                                                                    ;WITH THE CORRECT ROM ADR IN THE LIGHTS
6145  003314  077105                  5$:   SOB     R1,6$               ;CONTINUE
6146  003316  111237  177770                MOVB    (R2),@#UBREAK       ;PUT SAFE VALUE IN UBREAK REG
6147  003322  005037  177776                CLR     @#PSW               ;GO BACK TO KERNEL MODE
6148  003326  104400  003334                TYPE    ,69$                ;;TYPE ASCIZ STRING
  (1) 003332  000403                        BR      68$                 ;;GET OVER THE ASCIZ
  (1)                               ;;69$:  .ASCIZ  /DONE/<CRLF>
  (1) 003342                        68$:
6149  003342  000000                        HALT
6150  003344  000522                        BR      START
6151  003346     000       001       002  UBRTBL: .BYTE   0,1,2,4,10,20,40,200,100
      003351     004       010       020
      003354     040       200       100
6152        003360                        .EVEN
6153  003360  000010  005010  005020  INSTBL: .WORD   10,5010,5020,5040,0,5200,207,5010
      003366  005040  000000  005200
      003374  000207  005010
```

```
 6155  003400  012706  001100          START3: MOV     #1100,SP                ;SET UP STACK
 6156  003404  012737  062270  000034          MOV     #$TRAP,@#TRAPVEC        ;SET UP TRAP VECTOR
 6157  003412  104400  003420                  TYPE    ,65$                ;;TYPE ASCIZ STRING
   (1) 003416  000415                          BR      64$                 ;;GET OVER THE ASCIZ
   (1)                                 ;;65$:  .ASCIZ  <15><12>/PID REGISTER SETUP AID/
   (1) 003452                          64$:
 6158  003452  104400  003460                  TYPE    ,67$                ;;TYPE ASCIZ STRING
   (1) 003456  000430                          BR      66$                 ;;GET OVER THE ASCIZ
   (1)                                 ;;67$:  .ASCIZ  <15><12>/TYPE IN THE DESIRED PROCESSOR SERIAL NUMBER: /
   (1) 003540                          66$:
 6159  003540  104416                          RDDEC   ;GET THE NUMBER.
 6160  003542  104400  003550                  TYPE    ,69$                ;;TYPE ASCIZ STRING
   (1) 003546  000417                          BR      68$                 ;;GET OVER THE ASCIZ
   (1)                                 ;;69$:  .ASCIZ  <15><12>/THE OCTAL EQUIVALENT IS : /
   (1) 003606                          68$:
 6161  003606  104402                          TYPOC   ;TYPE THE NUMBER IN OCTAL
 6162  003610  000673                          BR      START3
 6163
 6164                                 ;;*****************************************************************************
 6165
 6166  003612  012706  001200          START:  MOV     #KERSTK,SP      ;SET KERNEL STACK PTR
 6167  003616  012737  076543  001622          MOV     #76543,@#$HINUM         ;INITIALIZE RANDOM NUM GEN
 6168  003624  012737  123456  001620          MOV     #123456,@#$LONUM
 6169
 6170                                 ;DETERMINE HOW PROGRAM WAS LOADED AND WHAT MODE (IF ACT11)
 6171                                 ;AND SET MEMORY PROTECTION.
 6172  003632  005037  001602                  CLR     @#QV                    ;SET NOT QV NOR AA MODE
 6173  003636  005027                          CLR     (PC)+                   ;SET NOT XXDP
 6174  003640     000          XXDP:   .BYTE   0                       ;XXDP INDICATOR
 6175  003641     000          XXDPC:  .BYTE   0                       ;XXDP CHAIN MODE INDICATOR
 6176  003642  005027                          CLR     (PC)+                   ;CLEAR MEMORY PROTECTION LIMIT
 6177  003644  000000          PROT:   .WORD   0                       ;WILL CONTAIN MEM PROT LIMIT
 6178  003646  005737  001234                  TST     @#$ENV                  ;APT MODE
 6179  003652  001411                          BEQ     15$                     ;NO
 6180  003654  110637  001603                  MOVB    SP,@#AA         ;SET UP FOR APT
 6181  003660  105737  001235                  TSTB    @#$ENVM                 ;APT CONTROL
 6182  003664  100023                          BPL     3$                      ;NO
 6183  003666  012737  001236  001244          MOV     #$APTSW,@#$SWRP         ;USE APT SWR
 6184  003674  000417                          BR      3$
 6185  003676  005737  046656          15$:    TST     @#$ENDAD+4              ;BRANCH IF NOT QV
 6186  003702  100003                          BPL     1$
 6187  003704  110637  001602                  MOVB    SP,@#QV                 ;SET ACT11 QV MODE
 6188  003710  000411                          BR      3$
 6189
 6190  003712  001003          1$:     BNE     2$
 6191  003714  110637  001603                  MOVB    SP,@#AA                 ;SET ACT11 AA MODE
 6192  003720  000405                          BR      3$
 6193
 6194  003722  005737  000042          2$:     TST     @#42                    ;BRANCH IF NOT IN CHAIN MODE
 6195  003726  001402                          BEQ     3$
 6196  003730  110637  003641                  MOVB    SP,@#XXDPC              ;SET CHAIN MODE INDICATOR
 6197
 6198                                 ;SET MEMORY PROTECTION LIMITS
 6199  003734  005737  001602          3$:     TST     @#QV                    ;BRANCH IF QV OR AA
 6200  003740  001006                          BNE     MEMSIZ
 6201  003742  005737  003640                  TST     @#XXDP                  ;BRANCH IF NOT VIA XXDP
```

```
                                                BEQ     MEMSIZ
                                                MOV     #5700,@#PROT            ;PROTECT XXDP MONITOR
                                        MEMSIZ: MOV     #157776,@#LSTMEM        ;SET VALUE INTO LSTMEM
                                                SUB     @#PROT,@#LSTMEM         ;SET PROTECTION
                                                MOV     #ENDTAG+2,@#FRSTMEM     ;SET FIRST RELOCATION ADDRESS

                                        ;GET ADDRESS OF THE LAST MEMORY LOCATION ON THE SYSTEM
                                                ;SIZE MEMORY AND COMPARE IT WITH THE SYSTEM SIZE REGISTER
                                                ;PRINT A WARNING IF THEY DISAGREE.

                                                BIS     #BIT07,$KT11
                                                JSR     PC,$SIZE
                                                ADD     #37,    $LSTBK

                                                MOV     $LSTBK,R2               ;COPY LAST BLOCK COUNT
                                                CMP     @#SIZELO,R2             ;EQUAL?
                                                BEQ     OKSIZ
                                                MOV     #$TRAP,@#TRAPVEC        ;SET UP TRAP VECTOR
                                                TYPE    ,65$                ;;TYPE ASCIZ STRING
                                                BR      64$                 ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <15><12>/WARNING- THE SIZE OF MEMORY IS DIFFERENT FROM THAT/
                                        64$:
                                                TYPE    ,67$                ;;TYPE ASCIZ STRING
                                                BR      66$                 ;;GET OVER THE ASCIZ
                                        ;;67$:  .ASCIZ  <15><12>/INDICATED BY THE SYSTEM SIZE REGISTER./
                                        66$:
                                                TYPE    ,69$                ;;TYPE ASCIZ STRING
                                                BR      68$                 ;;GET OVER THE ASCIZ
                                        ;;69$:  .ASCIZ  <15><12>/     SIZEHI  SIZELO      ACTUAL/
                                        68$:
                                                TYPE    ,$CRLF
                                                MOV     @#SIZEHI,-(SP)      ;;SAVE @#SIZEHI FOR TYPEOUT
                                                TYPOS                       ;;GO TYPE--OCTAL ASCII
                                                .BYTE   6                   ;;TYPE 6 DIGIT(S)
                                                .BYTE   0                   ;;SUPPRESS LEADING ZEROS
                                                TYPE    ,71$                ;;TYPE ASCIZ STRING
                                                BR      70$                 ;;GET OVER THE ASCIZ
                                        ;;71$:  .ASCIZ  /       /
                                        70$:
                                                MOV     @#SIZELO,-(SP)      ;;SAVE @#SIZELO FOR TYPEOUT
                                                TYPOS                       ;;GO TYPE--OCTAL ASCII
                                                .BYTE   6                   ;;TYPE 6 DIGIT(S)
                                                .BYTE   0                   ;;SUPPRESS LEADING ZEROS
                                                TYPE    ,73$                ;;TYPE ASCIZ STRING
                                                BR      72$                 ;;GET OVER THE ASCIZ
                                        ;;73$:  .ASCIZ  /       /
                                        72$:
                                                MOV     $LSTBK,-(SP)        ;;SAVE $LSTBK FOR TYPEOUT
                                                TYPOS                       ;;GO TYPE--OCTAL ASCII
                                                .BYTE   6                   ;;TYPE 6 DIGIT(S)
                                                .BYTE   0                   ;;SUPPRESS LEADING ZEROS
                                        ;
                                        ;FORM MXMMHI, MXMMLO, AND THE HIGHEST MEMORY ADDRESS BASED ON THE SIZE OF
                                        ;THE MEMORY
                                        ;
                                        OKSIZ:
```

```
6234  004354  005002                          CLR     R2
6235  004356  013703  062266                  MOV     @#$LSTBK,R3
6236  004362  073227  000006                  ASHC    #6,R2           ;SHIFT TO FORM CORRECT ADDRESS
6237  004366  052703  000077                  BIS     #77,R3          ;ENSURE LOWER SIX BITS SET
6238  004372  062703  000001                  ADD     #1,R3
6239  004376  005502                          ADC     R2
6240                                  ;;***************
6241  004400  010237  001656                  MOV     R2,@#MXMMHI     ;SAVE UPPER SIX BITS
6242  004404  010337  001660                  MOV     R3,@#MXMMLO     ;SAVE LOWER 16 BITS
6243
6244  004410  012706  001200                  MOV     #KERSTK,SP              ;SET STACK PTR
6245  004414  005037  001250                  CLR     @#$PASS                 ;CLEAR PASS COUNT
6246  004420  105037  001601                  CLRB    @#MMON                  ;SET MEM MGMT ON IND=NOT ON
6247  004424  012737  000700  001616          MOV     #700,@#NEXPAR           ;SET FIRST 'PAR' VALUE
6248  004432  005737  003644                  TST     @#PROT
6249  004436  001403                          BEQ     1$
6250  004440  012737  001600  001616          MOV     #1600,@#NEXPAR
6251  004446                          1$:
6252  004446  012700  000027                  MOV     #27,R0                  ;SET SOB COUNT
6253  004452  005001                          CLR     R1                      ;SETUP INDEX
6254  004454  005061  001676          2$:     CLR     MTICKS(R1)              ;CLEAR TABLES
6255  004460  062701  000002                  ADD     #2,R1
6256  004464  077005                          SOB     R0,2$                   ;CONTINUE
6257  004466  012737  177777  001766          MOV     #-1,@#MAPTBL            ;INITIALIZE MAP TABLE
6258  004474  012737  177777  001770          MOV     #-1,@#MAPTBL+2
6259  004502  012700  000010                  MOV     #10,R0                  ;SET SOB COUNT
6260  004506  012701  002006                  MOV     #RP3HSTAT,R1            ;GET ADDRESS OF HANDLER STAT
6261  004512  012721  000200          3$:     MOV     #200,(R1)+              ;INITIALIZE STATUS TABLE
6262  004516  077003                          SOB     R0,3$                   ;CONTINUE
6263  004520  012737  000060  001630          MOV     #60,@#SUBPASS           ;INIT SUBPASS TO ASCII 0
6264  004526  012700  057060                  MOV     #TIMEBUF,R0             ;GET ADR OF TIME BUFFER
6265  004532  012701  000012                  MOV     #12,R1                  ;SET SOB COUNT
6266  004536  112720  000060          4$:     MOVB    #60,(R0)+               ;INIT TIME BUFFER
6267  004542  077103                          SOB     R1,4$
6268  004544  105040                          CLRB    -(R0)                   ;INSERT TERMINATOR
6269  004546  112737  000072  057063          MOVB    #72,@#TIMEBUF+3         ;INSERT COLON
6270  004554  112737  000072  057066          MOVB    #72,@#TIMEBUF+6
6271  004562  012737  000340  177776          MOV     #340,@#PS       ;;LOCK OUT ALL INTERRUPTS
 (1)  004570  012706  001250                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
 (1)  004574  005026                          CLR     (R6)+           ;;CLEAR MEMORY LOCATION
 (1)  004576  022706  001310                  CMP     #$TKS,R6        ;;DONE?
 (1)  004602  001374                          BNE     .-6             ;;LOOP BACK IF NO
 (1)  004604  012706  001200                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
 (1)  004610  012737  054456  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
 (1)  004616  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
 (1)  004624  012737  054710  000030          MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
 (1)  004632  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
 (1)  004640  012737  062270  000034          MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
 (1)  004646  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
 (1)  004654  012737  061206  000024          MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
 (1)  004662  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
 (1)  004670  016767  041622  041612          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
 (1)  004676  005067  174474                  CLR     $TIMES          ;;INITIALIZE NUMBER OF ITERATIONS
 (1)  004702  005067  174472                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
 (1)  004706  112767  000001  174353          MOVB    #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
 (1)  004714  012767  004714  174336          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
```

```
  (1) 004722 012767 004722 174332          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
6272
6273                                       ;CLEAR PROGRAM INDICATORS
6274 004730 052777 000100 174352          BIS     #100,@$TKS              ;SET IE BIT IN KEYBOARD STATUS REG
6275 004736 012737 063354 000060          MOV     #TKISR,@#TKVEC          ;SETUP KEYBOARD VECTOR
6276 004744 012737 000200 000062          MOV     #PR4,@#TKVEC+2
6277 004752 012737 063566 000064          MOV     #TPISR,@#TPVEC
6278 004760 012737 000200 000066          MOV     #PR4,@#TPVEC+2
6279 004766 005037 001546                 CLR     @#NOTYPE                ;CLEAR 'NO TYPING' INDICATOR
6280
6281                                       ;THE BELOW ROUTINE ASCERTAINS WHICH CP & CP OPTIONS THE PROGRAM IS RUN-
6282                                       ;NING ON AND SETS AN INDICATOR IN OPT.CP ACCORDINGLY.
6283 004772 012737 000006 000004 CPCHK:    MOV     #ERRVEC+2,@#ERRVEC      ;SET UP ERROR TRAP TO RETURN
6284 005000 012737 000002 000006          MOV     #2,@#ERRVEC+2
6285 005006 012737 000012 000010          MOV     #RESVEC+2,@#RESVEC      ;AND ALSO RESERVED INST TRAP
6286 005014 012737 000002 000012          MOV     #2,@#RESVEC+2
6287 005022 012702 144006                 MOV     #144006,R2              ;SET 11/70 NON-OPTION BITS
6288 005026 000261                        SEC
6289 005030 170500                        TSTF    R0                      ;WILL CLEAR CARRY IF 11/70 FLOATING POINT
6290 005032 170000                        CFCC                            ;IS AVAIL. COPY FLOATING CC'S INTO PSW
6291 005034 103402                        BCS     6$                      ;BRANCH IF NO FLOATING POINT
6292 005036 052702 020000                 BIS     #FPOPT,R2               ;SET FP OPTION AVAIL INDICATOR
6293 005042 000261                 6$:    SEC
6294 005044 005737 177546                 TST     @#LKS                   ;BRANCH IF NO KW11-L
6295 005050 103402                        BCS     7$
6296 005052 052702 001000                 BIS     #LKOPT,R2               ;SET OPTION INDICATOR
6297 005056 000261                 7$:    SEC
6298 005060 005777 174230                 TST     @$TPS                   ;BRANCH IF NO CONSOLE TTY
6299 005064 103402                        BCS     9$
6300 005066 052702 000400                 BIS     #TTOPT,R2
6301 005072 005003                 9$:    CLR     R3
6302 005074 000261                        SEC
6303 005076 005737 170000                 TST     @#UBEDB         ;IS UBE1 THERE?
6304 005102 103410                        BCS     12$             ;BRANCH IF NO
6305 005104 105037 170006                 CLRB    @#UBECR1        ;IS THIS A TESTER OR EXERCISER?
6306 005110 105737 170006                 TSTB    @#UBECR1
6307 005114 100045                        BPL     15$             ;BRANCH IF TESTER
6308 005116 052702 000200          16$:   BIS     #UBEOPT,R2      ;SET INDICATOR
6309 005122 000425                        BR      17$
6310 005124 000261                 12$:   SEC
6311 005126 005737 170020                 TST     @#UBEDB+20              ;IS UBE2 THERE?
6312 005132 103403                        BCS     13$             ;BRANCH IF NO
6313 005134 012703 000020                 MOV     #20,R3          ;SET OFFSET IN R3
6314 005140 000766                        BR      16$
6315 005142 000261                 13$:   SEC
6316 005144 005737 170040                 TST     @#UBEDB+40      ;IS UBE3 THERE?
6317 005150 103403                        BCS     14$             ;BRANCH IF NO
6318 005152 012703 000040                 MOV     #40,R3          ;PUT OFFSET IN R3
6319 005156 000757                        BR      16$
6320 005160 000261                 14$:   SEC
6321 005162 005737 170060                 TST     @#UBEDB+60      ;IS UBE4 THERE?
6322 005166 103420                        BCS     15$             ;BRANCH IF NO
6323 005170 012703 000060                 MOV     #60,R3          ;PUT OFFSET IN R3
6324 005174 000750                        BR      16$
6325 005176 005227 177777          17$:   INC     #-1
6326 005202 001012                        BNE     15$
```

```
6327  005204  012704  002340              MOV     #UBETBL,R4      ;GET ADDRESS OF UBE TABLE
6328  005210  012705  000005              MOV     #5,R5           ;SET SOB COUNT
6329  005214  060324              18$:    ADD     R3,(R4)+        ;ADJUST UBE TABLE ENTRIES
6330  005216  077502                      SOB     R5,18$          ;CONTINUE
6331  005220  006003                      ROR     R3
6332  005222  006003                      ROR     R3              ;ADJUST OFFSET FOR UBE VECTOR
6333  005224  060324                      ADD     R3,(R4)+        ;ADJUST UBEVEC ENTRY
6334  005226  060314                      ADD     R3,(R4)         ;ADJUST UBEVEC PSW ENTRY
6335  005230  005003              15$:    CLR     R3                      ;INIT R3
6336  005232  000261                      SEC
6337  005234  005777  175116            TST     @MBTTBL         ;IS MASS BUS TESTER THERE?
6338  005240  103403                      BCS     20$             ;BRANCH IF NO
6339  005242  052702  002000      21$:    BIS     #MBTOPT,R2      ;SET OPTION AVAILABLE
6340  005246  000422                      BR      24$
6341  005250  005777  175132      20$:    TST     @MBTN2          ;IS MBT2 THERE?
6342  005254  103403                      BCS     22$             ;BRANCH IF NO
6343  005256  012703  000100            MOV     #100,R3         ;SETUP R3
6344  005262  000767                      BR      21$
6345  005264  005777  175120      22$:    TST     @MBTN3          ;IS MBT3 THERE?
6346  005270  103403                      BCS     23$             ;BRANCH IF NO
6347  005272  012703  000200            MOV     #200,R3
6348  005276  000761                      BR      21$
6349  005300  005777  175106      23$:    TST     @MBTN4          ;IS MBT4 THERE?
6350  005304  103427                      BCS     30$             ;BRANCH IF NO
6351  005306  012703  000300            MOV     #300,R3
6352  005312  000753                      BR      21$
6353  005314              24$:
6354  005314  000240                      NOP
6355  005316  000240                      NOP
6356  005320  000240                      NOP
6357  005322  012704  002356            MOV     #MBTTBL,R4      ;GET ADDRESS OF MBT TABLE
6358  005326  012705  000011            MOV     #11,R5          ;SET SOB COUNT
6359  005332  060324              25$:    ADD     R3,(R4)+        ;ADJUST MBT TABLE
6360  005334  077502                      SOB     R5,25$          ;CONTINUE
6361  005336  060337  002404            ADD     R3,@#MBTTBL+26  ;ADJUST DRIVE TYPE ADDRESS
6362  005342  112777  000007  175020    MOVB    #7,@MBTTBL+12   ;SET UNIT NUMBER
6363  005350  122777  000040  175026    CMPB    #40,@MBTTBL+26  ;IS THIS REALLY A MBT?
6364  005356  001402                      BEQ     30$             ;BRANCH IF YES
6365  005360  042702  002000            BIC     #MBTOPT,R2      ;CLEAR OPTION AVAILABLE BIT
6366  005364  012737  064422  000004  30$:    MOV     #ERPRT,@#ERRVEC ;RESTORE ERROR TRAP
6367                                      ;
6368                                      ;*** TEST FOR VARIOUS KB11 PROCESSORS ***
6369                                      ;
6370                                      ;*THIS ROUTINE POLES THE RESULTS OF ATTEMPTS TO SET TO ONE
6371                                      ;*CERTAIN CRITICAL BITS THAT ARE KNOWN TO BE OPERATIVE ON A KB11CM,
6372                                      ;*OR KB11EM PROCESSOR. IF TWO OUT OF FOUR OF THE TESTS ARE
6373                                      ;*POSITIVE THEN THE KB11CM OR KB11EM FLAG IS SET,IF LESS THAN TWO OF THE
6374                                      ;*TESTS ARE POSITIVE THEN THE KB11E FLAG OR NO FLAG IS SET. THE DETERMINATION
6375                                      ;*OF WHICH PAIR IS VALID IS BASED ON THE RESULTS OF EXECUTING AN MFPT OPCODE
6376                                      ;*(OPCODE 7).  IF THIS INSTRUCTION TRAPS THIS IS AN KB11CM OR
6377                                      ;*A PLAIN 1170 (KB11-B OR KB11-C).  IF THE INSTRUCTION DOES NOT TRAP THEN
6378                                      ;*THIS IS A KB11-E OR KB11-EM.
6379                                      ;
6380
6381  005372  104420                      SAVREG                  ;SAVE GPRS R5-R0
6382  005374  105037  001554            CLRB    @#KB11CM        ;RESET THE MP FLAG
```

```
 6383  005400  005037  001552                CLR     @#KB11E          ;CLEAR KB11E AND KB11EM FLAGS
 6384  005404  012737  005652  000010        MOV     #MFPTTR,@#RESVEC ;SET UP TRAP ADDRESS FOR MFPT AT RESERV VECTOR
 6385  005412  000007                        MFPT                     ;EXECUTE MFPT. WILL TRAP ON 1170 (KB11B/C) OR
 6386                                                                  ;KB11CM (11/74          )
 6387
 6388  005414  012737  000001  001552        MOV     #1,@#KB11E       ;HERE IF KB11E OR KB11EM. SET FLAG
 6389  005422  005037  177750          T1:   CLR     @#MAINT          ;CLEAR THE MAINTENANCE REGISTER
 6390  005426  005005                        CLR     R5               ;RESET THE TEST COUNTER
 6391  005430  012700  177746                MOV     #CONTRL,R0       ;GET THE ADDRESS OF...
 6392  005434  012701  177750                MOV     #MAINT,R1        ;CCR,MAINT,AND MAPH00...
 6393  005440  012702  170202                MOV     #MAPH00,R2       ;AND PLACE IN R0-R2
 6394  005444  052710  040000                BIS     #BIT14,(R0)      ;TRY TO SET IVSS BIT
 6395  005450  032710  040000                BIT     #BIT14,(R0)      ;DID IT SET?
 6396  005454  001403                        BEQ     T2               ;NO,GO TO NEXT TEST
 6397  005456  042710  040000                BIC     #BIT14,(R0)      ;CLEAR IT.
 6398  005462  005205                        INC     R5               ;TEST IS POSITIVE
 6399  005464  052711  000001          T2:   BIS     #BIT0,(R1)       ;SET EDMA IN MAINT REGISTER
 6400  005470  032711  000001                BIT     #BIT0,(R1)
 6401  005474  001410                        BEQ     T3
 6402  005476  052710  004000                BIS     #BIT11,(R0)      ;TRY TO SET DMMA IN CCR
 6403  005502  032710  004000                BIT     #BIT11,(R0)
 6404  005506  001403                        BEQ     T3
 6405  005510  042710  004000                BIC     #BIT11,(R0)
 6406  005514  005205                        INC     R5
 6407  005516  042711  000001          T3:   BIC     #BIT0,(R1)       ;MAKE SURE EDMA IS CLEAR
 6408  005522  052767  100000  164550        BIS     #BIT15,KIPDR0    ;TRY TO SET BYP ON A PDR
 6409  005530  032767  100000  164542        BIT     #BIT15,KIPDR0
 6410  005536  001404                        BEQ     T4
 6411  005540  042767  100000  164532        BIC     #BIT15,KIPDR0
 6412  005546  005205                        INC     R5
 6413  005550  052712  100000          T4:   BIS     #BIT15,(R2)      ;TRY TO SET BYP ON UNIBUS MAP
 6414  005554  032712  100000                BIT     #BIT15,(R2)
 6415  005560  001403                        BEQ     T.END
 6416  005562  042712  100000                BIC     #BIT15,(R2)
 6417  005566  005205                        INC     R5
 6418  005570  022705  000002        T.END:  CMP     #2,R5            ;IS THE RESULT OF THE TEST >=2
 6419  005574  101021                        BHI     3$               ;BR IF NO,THIS IS A KB11E OR KB11-B/C (11/70)
 6420  005576  005000                        CLR     R0
 6421  005600  005037  177746                CLR     @#CONTRL
 6422  005604  013701  177746          4$:   MOV     @#CONTRL,R1
 6423  005610  001402                        BEQ     5$
 6424  005612  005200                        INC     R0
 6425  005614  001373                        BNE     4$
 6426  005616                          5$:
 6427  005616  005737  001552                TST     @#KB11E          ;IS IS A KB11-E OR KB11-EM?
 6428
 6429  005622  001404                        BEQ     1$               ;BR IF NEITHER. MUST BE KB11CM
 6430  005624  012737  000400  001552        MOV     #BIT8,@#KB11E    ;SET UPPER BYTE (KB11-EM)
 6431
 6432  005632  000405                        BR      2$               ;DONE
 6433  005634  105237  001554          1$:   INCB    @#KB11CM         ;YES, FLAG THIS AS A MODIFIED PROCESSOR
 6434  005640  005737  001552          3$:   TST     @#KB11E          ;IS THIS A KB11E?
 6435  005644  001472                        BEQ     RESTORE          ;BR IF NOT. THIS IS AN 1170
 6436  005646  104422                  2$:   RESREG                   ;RESTORE R5-R0
 6437  005650  000403                        BR      ENDKB            ;DONE DETERMINEING WHICH CPU
 6438
```

```
                                MFPTTR:                         ;HERE IF MFPT TRAPPED. SEE IF 1170 OR KB11CM
005652  012716  005422                  MOV     #T1,(SP)        ;SET UP RTI RETURN ADDRESS
005656  000002                          RTI                     ;RETURN
005660                          ENDKB:

                                ;SEE IF CISP IS PRESENT.  TRY TO EXECUTE 3 CISP INSTRUCTIONS.  IF TWO OUT
                                ;OF THE THREE DON'T TRAP, IT IS ASSUMED THAT THE CISP OPTION IS PRESENT AND
                                ;A FLAG IS SET TO INDICATE THIS.  ALSO A BIT IS SET IN OPT.CP AND A MESSAGE
                                ;IS PRINTED.


005660  052702  010000                  BIS     #CISOPT,R2      ;SET CISP OPTION BIT FOR OPT.CP
005664  104420                          SAVREG                  ;SAVE R5-R0
005666  012737  006016  000010          MOV     #TRPRTN,@#RESVEC ;SET UP TRAP ADDRESS AT RESERVED VECTOR
005674  105037  001555                  CLRB    @#CISP          ;COUNT HOW MANY CIS OPCODES DON'T TRAP
005700  012737  076020  006012          MOV     #L2D0,@#CISOP   ;CIS OPCODE TO TEST (LOAD 2 DESCRIPTORS @R0)
005706  012700  001702                  MOV     #LD2PNT,R0      ;R0 MUST BE EVEN AND POINT TO A WORD WHICH IS ALSO EVEN
005712  004737  006012                  JSR     PC,@#CISOP      ;TEST OPCODE FOR A TRAP
005716  012737  076061  006012          MOV     #L3D1,@#CISOP   ;SET UP OPCODE FOR LOAD 3 DESCRIPTORS @R1
005724  012701  001702                  MOV     #LD2PNT,R1      ;LOAD R1 WITH EVEN WORD AND POINT TO EVEN CONTENTS
005730  004737  006012                  JSR     PC,@#CISOP      ;TEST OPCODE FOR TRAP
005734  052737  100000  177770          BIS     #BIT15,@#UBREAK ;SET MAINT MODE IN U-BREAK REGISTER
005742  012737  076601  006012          MOV     #MED74C,@#CISOP ;OPCODE FOR DIAGNOSTIC ENTRY
005750  012705  006600                  MOV     #CISTST,R5      ;ADDRESS OF DIAGNOSTIC U-CODE
005754  004737  006012                  JSR     PC,@#CISOP      ;TEST OPCODE FOR TRAP
005760  104422                          RESREG                  ;RESTORE R5-R0
005762  122737  000002  001555          CMPB    #2,@#CISP       ;IS RESULT >=2?
005770  101404                          BLOS    1$              ;BR IF CISP IS PRESENT
005772  105037  001555                  CLRB    @#CISP          ;CLEAR CISP PRESENT FLAG
005776  042702  010000                  BIC     #CISOPT,R2      ;AND ALSO IN OPT.CP
006002  042737  100000  177770  1$:     BIC     #BIT15,@#UBREAK ;CLEAR MAINT BIT IN U-BREAK REGISTER
006010  000411                          BR      SETOP           ;GO TO RESTORE VECTOR AND SET OPT.CP

006012  000000                  CISOP:  .WORD   0               ;CISP OPCODE WILL GO HERE FOR EXECUTION
006014  000403                          BR      NOTRAP          ;WILL COME HERE IF NO TRAP
006016  012716  006030          TRPRTN: MOV     #CISTRP,(SP)    ;SET UP RTI RETURN ADDRESS
006022  000002                          RTI                     ;RETURN TO LOCATION FROM TRAP
006024  105237  001555          NOTRAP: INCB    @#CISP          ;INCREMENT CISP INDICATOR
006030  000207                  CISTRP: RTS     PC              ;RETURN

006032  104422                  RESTOR: RESREG                  ;RESTORE R5-R0
006034                          SETOP:
006034  012737  064350  000010          MOV     #RESERR,@#RESVEC;AND ALSO RESERVED INST TRAP
006042  010237  001550                  MOV     R2,@#OPT.CP     ;LOAD INDICATOR
006046  005227  177777                  INC     #-1             ;;FIRST TIME?
(1) 006052  001031                      BNE     64$             ;;BRANCH IF NO
(1) 006054  022737  046652  000042      CMP     #$ENDAD,@#42    ;;ACT-11?
(1) 006062  001425                      BEQ     64$             ;;BRANCH IF YES
(1) 006064  104400  006072              TYPE    ,65$            ;;TYPE ASCIZ STRING
(1) 006070  000422                      BR      64$             ;;GET OVER THE ASCIZ
(1)                             ;;65$:  .ASCIZ  <CRLF>''CEQKC-E...PDP 11/70 CPU EXERCISER''<CRLF>
(1) 006136                      64$:
006136  005227  177777                  INC     #-1             ;FIRST TIME?
006142  001036                          BNE     100$            ;BR IF NO
006144  104400  071227                  TYPE    ,MSG34          ;<15><12>CPU UNDER TEST FOUND TO BE A
```

```
 6488  006150  005737  001552                 TST     @#KB11E          ;IS THIS A KB11-E OR KB11-EM?
 6489  006154  001011                         BNE     101$             ;BR IF EITHER ONE
 6490  006156  105737  001554                 TSTB    @#KB11CM         ;IS IT A 11/74          (KB11CM)
 6491  006162  001003                         BNE     1$               ;BR IF IT IS
 6492  006164  104400  071267                 TYPE    ,MSG35           ;KB11-B/C<15><12>
 6493  006170  000423                         BR      100$             ;SKIP OTHER MESSAGE
 6494  006172  104400  071175         1$:     TYPE    ,MSG32           ;11/74          (KB11CM)<15><12>
 6495  006176  000420                         BR      100$             ;SKIP CISP MESSAGE
 6496  006200  105737  001552         101$:   TSTB    @#KB11E          ;IS IT A KB11-E?
 6497  006204  001403                         BEQ     102$             ;BR IF NOT. MUST BE KB11-EM
 6498  006206  104400  071302                 TYPE    ,MSG36           ;KB11-E<15><12>
 6499  006212  000402                         BR      104$             ;SKIP KB11-EM MESSAGE
 6500  006214  104400  071163         102$:   TYPE    ,MSG31           ;KB11-EM<15><12>
 6501  006220  105767  173331         104$:   TSTB    CISP             ;IS CISP PRESENT?
 6502  006224  001003                         BNE     103$             ;BR IF CISP PRESENT
 6503  006226  104400  071313                 TYPE    ,MSG37           ;CISP OPTION NOT FOUND<15><12>
 6504  006232  000402                         BR      100$             ;SKIP OTHER MESSAGE
 6505  006234  104400  071343         103$:   TYPE    ,MSG38           ;CISP OPTION FOUND<15><12>
 6506  006240                         100$:
   (1) 006240  104400  006246                 TYPE    ,67$             ;;TYPE ASCIZ STRING
   (1) 006244  000415                         BR      66$              ;;GET OVER THE ASCIZ
   (1)                            ;;67$:   .ASCIZ  <15><12>/PROCESSOR ID REGISTER =/
   (1) 006300                     66$:
 6507  006300  013746  177764                 MOV     @#177764,-(SP)   ;;SAVE @#177764 FOR TYPEOUT
   (1) 006304  104402                         TYPOC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 6508  006306  104400  006314                 TYPE    ,69$             ;;TYPE ASCIZ STRING
   (1) 006312  000406                         BR      68$              ;;GET OVER THE ASCIZ
   (1)                            ;;69$:   .ASCIZ  / (OCTAL)  /
   (1) 006330                     68$:
 6509  006330  013746  177764                 MOV     @#177764,-(SP)   ;;SAVE @#177764 FOR TYPEOUT
   (1) 006334  104410                         TYPDS                    ;;GO TYPE--DECIMAL ASCII WITH SIGN
 6510  006336  104400  006344                 TYPE    ,71$             ;;TYPE ASCIZ STRING
   (1) 006342  000406                         BR      70$              ;;GET OVER THE ASCIZ
   (1)                            ;;71$:   .ASCIZ  / (DECIMAL) /
   (1) 006360                     70$:
 6511  006360  104400  001407                 TYPE    ,$CRLF
 6512                             ;;*****************************************************************
 6513                             .SBTTL  SYSTEM SIZER
 6514                             ;       THIS ROUTINE DETERMINES WHAT DRIVES ARE AVAILABLE ON
 6515                             ;       THE FOLLOWING DEVICES: RK05, RP03, RP04, AND RS04. THE
 6516                             ;       INFORMATION IS STORED IN THE TABLE "SYSSIZE" IN THE FOLLOWING FORMAT:
 6517                             ;               A. EACH DEVICE IS ASSIGNED A WORD
 6518                             ;               B. THE LOW BYTE OF THIS WORD INDICATES WHICH DRIVES ARE AVAILABLE
 6519                             ;               C. THE HIGH BYTE INDICATES WHICH DRIVES HAVE BEEN USED
 6520                             ;                  BY THE RELOCATION ROUTINE.
 6521                             ;;*****************************************************************
 6522  006364  012737  006476  000004 SIZE:   MOV     #21$,@#ERRVEC            ;SETUP TIMEOUT VECTOR
 6523  006372  005037  001352                 CLR     @#$TMP0                  ;ENSURE $TMP0 CLEAR
 6524  006376  005000                         CLR     R0                       ;USED TO SET THE UNIT AVAIL BITS
 6525  006400  012701  000010                 MOV     #10,R1                   ;SOB COUNT
 6526  006404  013777  001352  173632 9$:     MOV     @#$TMP0,@RKDA            ;SET UNIT NUMBER
 6527  006412  012777  000015  173616         MOV     #15,@RKCS                ;SEND DRIVE RESET
 6528  006420  032777  000200  173606         BIT     #BIT7,@RKER              ;NON EXISTANT DISK?
 6529  006426  001011                         BNE     7$                       ;BRANCH IF YES
 6530  006430  017702  173576                 MOV     @RKDS,R2                 ;GET DRIVE STATUS
 6531  006434  042702  177537                 BIC     #177537,R2               ;GET BITS 5 & 7 ONLY
```

```
6532  006440  022702  000200          CMP     #200,R2                 ;IS DRIVE READY?
6533  006444  001002                  BNE     7$                      ;BRANCH IF NO
6534  006446  052700  000400          BIS     #BIT8,R0                ;SET UNIT AVAILABLE
6535  006452  006000          7$:     ROR     R0
6536  006454  012777  000001  173554  MOV     #1,@RKCS                ;CLEAR THE ERRORS
6537  006462  062737  020000  001352  ADD     #20000,@#$TMP0          ;SELECT NEXT UNIT
6538  006470  077133                  SOB     R1,9$                   ;CONTINUE
6539  006472  110037  001714          MOVB    R0,@#SYSSIZE+2          ;STORE IN TABLE
6540
6541                          ;;*************************************************************************
6542                          ;THIS CODE DETERMINES IF THERE IS AN RP03 OR AN RP04 OR BOTH.
6543                          ;IF BOTH ARE ON THE SYSTEM, THE OPERATOR MUST CHANGE THE RP04
6544                          ;ADDRESSES IN THE TABLE IN ''COMMON TAGS'' AND 'NOP'' THE BRANCH
6545                          ;AT ''100$''.
6546
6547  006476  012737  007104  000004  21$:    MOV     #11$,@#ERRVEC           ;SET THE ERROR VECTOR
6548  006504  005737  176710          TST     @#176710                ;IS THERE AN RP ON THE SYSTEM?
6549                                                                  ;STAY HERE IF YES
6550  006510  012737  006524  000004          MOV     #1$,@#ERRVEC
6551  006516  005777  173530          TST     @RP4CS1         ;IS THERE AN RP04 ON SYSTEM?
6552  006522  000441          100$:   BR      10$                     ;BRANCH IF YES
6553                          ;;*************************************************************************
6554
6555                          ;;*************************************************************************
6556  006524  012737  006626  000004  1$:     MOV     #10$,@#ERRVEC           ;SETUP TIMEOUT VEC FOR RP03 TEST
6557  006532  012737  000001  001352          MOV     #1,@#$TMP0              ;SETUP TEMP0
6558  006540  005000                  CLR     R0                      ;USED TO SET UNIT AVAILABLE BITS
6559  006542  012701  000010          MOV     #10,R1                  ;SOB COUNT
6560  006546  013777  001352  173440  3$:     MOV     @#$TMP0,@RP3CS          ;SET FUNCTION IDLE WITH UNIT NO
6561  006554  005777  173434          TST     @RP3CS                  ;WAS THERE AN ERROR?
6562  006560  100006                  BPL     6$                      ;BRANCH IF NO
6563  006562  006000          4$:     ROR     R0                      ;UNIT NOT AVAILABLE
6564  006564  062737  000400  001352  ADD     #400,@#$TMP0            ;SELECT NEXT UNIT
6565  006572  077113                  SOB     R1,3$                   ;CONTINUE
6566  006574  000412                  BR      5$
6567  006576  017702  173406  6$:     MOV     @RP3DS,R2               ;GET STATUS REGISTER
6568  006602  042702  037777          BIC     #37777,R2               ;GET BITS 14, 15 ONLY
6569  006606  022702  140000          CMP     #140000,R2              ;IS DRIVE READY?
6570  006612  001363                  BNE     4$                      ;BRANCH IF NO
6571  006614  052700  000400          BIS     #BIT8,R0                ;SET DRIVE AVAILABLE BIT
6572  006620  000760                  BR      4$                      ;CONTINUE
6573  006622  110037  001712  5$:     MOVB    R0,@#SYSSIZE            ;STORE IN TABLE
6574
6575                          ;;*************************************************************************
6576  006626  012737  007104  000004  10$:    MOV     #11$,@#ERRVEC           ;SETUP ERROR VEC FOR RP04 TEST
6577  006634  005037  001352          CLR     @#$TMP0
6578  006640  005000                  CLR     R0                      ;UNIT AVAILABLE WORD
6579  006642  012701  000010          MOV     #10,R1                  ;SOB COUNT
6580  006646  113777  001352  173410  14$:    MOVB    @#$TMP0,@RP4CS2         ;SET UNIT NUMBER
6581  006654  012777  000021  173370          MOV     #21,@RP4CS1             ;TRY READ-IN-PRESET
6582  006662  032777  010000  173374          BIT     #BIT12,@RP4CS2          ;NON EXISTANT DRIVE?
6583  006670  001071                  BNE     12$                     ;BRANCH IF YES
6584  006672  017702  173372          MOV     @RP4DS,R2               ;GET DRIVE STATUS
6585  006676  032702  001000          BIT     #BIT9,R2                ;IS DRIVE IN PROGRAMMABLE MODE?
6586  006702  001455                  BEQ     8$                      ;NO
6587  006704  104400  006712          TYPE    ,65$            ;;TYPE ASCIZ STRING
```

```
(1)  006710  000410                        BR      64$             ;;GET OVER THE ASCIZ
(1)                                ;;65$:   .ASCIZ  <15><12>/RP04 DRIVE #/
(1)  006732                        64$:
6588 006732  013746  001352                MOV     @#$TMP0,-(SP)   ;;SAVE @#$TMP0 FOR TYPEOUT
(1)  006736  104402                        TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
6589 006740  104400  006746                TYPE    ,67$            ;;TYPE ASCIZ STRING
(1)  006744  000433                        BR      66$             ;;GET OVER THE ASCIZ
(1)                                ;;67$:   .ASCIZ  / FOUND IN PROGRAMMABLE MODE-DRIVE WILL NOT BE USED/<15><12>
(1)  007034                        66$:
6590 007034  000407                        BR      12$
6591 007036  042702  163277        8$:     BIC     #163277,R2              ;GET BITS 12, 11, 8, & 6 ONLY
6592 007042  022702  010500                CMP     #10500,R2               ;IS DRIVE READY?
6593 007046  001002                        BNE     12$                     ;BRANCH IF NO
6594 007050  052700  000400                BIS     #BIT8,R0                ;SET UNIT AVAILABLE
6595 007054  006000        12$:    ROR     R0
6596 007056  052777  000040  173200        BIS     #BIT5,@RP4CS2           ;CLEAR ERROR BITS
6597 007064  005237  001352                INC     @#$TMP0                 ;SELECT NEXT DRIVE
6598 007070  005301                        DEC     R1
6599 007072  001402                        BEQ     .+6
6600 007074  000167  177546                JMP     14$
6601 007100  110037  001722                MOVB    R0,@#SYSSIZE+10         ;STORE IN TABLE
6602
6603                               ;;*****************************************************************
6604 007104  012737  007214  000004 11$:    MOV     #15$,@#ERRVEC           ;SETUP ERROR VEC FOR RS04 TEST
6605 007112  005037  001352                CLR     @#$TMP0
6606 007116  005000                        CLR     R0
6607 007120  012701  000010                MOV     #10,R1                  ;SOB COUNT
6608 007124  113777  001352  173172 18$:    MOVB    @#$TMP0,@RSCS2          ;SET UNIT NUMBER
6609 007132  012777  000001  173152        MOV     #1,@RSCS1               ;TRY NOP OPERATION
6610 007140  032777  010000  173156        BIT     #BIT12,@RSCS2           ;NON EXISTANT DRIVE?
6611 007146  001011                        BNE     16$                     ;BRANCH IF YES
6612 007150  017702  173154                MOV     @RSDS,R2                ;GET DRIVE STATUS
6613 007154  042702  163577                BIC     #163577,R2              ;GET BITS 12, 11, & 7 ONLY
6614 007160  022702  010200                CMP     #10200,R2               ;IS DRIVE READY?
6615 007164  001002                        BNE     16$                     ;BRANCH IF NO
6616 007166  052700  000400                BIS     #BIT8,R0                ;SET DRIVE AVAILABLE BIT
6617 007172  006000        16$:    ROR     R0
6618 007174  052777  000040  173122        BIS     #BIT5,@RSCS2            ;CLEAR ANY ERROR BITS
6619 007202  005237  001352                INC     @#$TMP0                 ;SELECT NEXT UNIT
6620 007206  077132                        SOB     R1,18$                  ;CONTINUE
6621 007210  110037  001724                MOVB    R0,@#SYSSIZE+12         ;STORE IN TABLE
6622                               ;
6623                               ;NEXT, DELETE XXDP UNIT 0 FROM TABLE
6624 007214  122737  000002  000041 15$:    CMPB    #2,@#41                 ;RK?
6625 007222  001004                        BNE     19$                     ;BRANCH IF NO
6626 007224  042737  000001  001714        BIC     #BIT0,@#SYSSIZE+2       ;MAKE UNIT ZERO NOT AVAILABLE
6627 007232  000420                        BR      20$
6628 007234  113700  000041        19$:    MOVB    @#41,R0                 ;GET LOCATION 41
6629 007240  042700  177770                BIC     #177770,R0              ;GET LEAST SIG 3 BITS
6630 007244  000241                        CLC                             ;ENSURE C CLEAR
6631 007246  006100                        ROL     R0                      ;ADJUST
6632 007250  122700  000002                CMPB    #2,R0                   ;
6633 007254  002404                        BLT     40$                     ;BRANCH IF NO
6634 007256  042737  000001  001712        BIC     #BIT0,@#SYSSIZE
6635 007264  000403                        BR      20$
6636 007266  042760  000001  001716 40$:    BIC     #BIT0,SYSSIZE+4(R0)
```

```
                                20$:    INC     #-1
                                        BNE     LOOP1                   ;;BRANCH IF NOT FIRST TIME
                                        TYPE    ,MSG25
                                        MOV     @#OPT.CP,-(SP)
                                        TYPOC
                                        TYPE    ,$CRLF
                                        TST     @#0V                            ;ACT11?
                                        BNE     LOOP1                           ;BRANCH IF YES
                                50$:    TSTB    @#XXDPC                         ;XXDP CHAIN MODE?
                                        BNE     LOOP1                   ;;BRANCH IF YES
                                        TSTB    @#$PASS                         ;FIRST PASS?
                                        BNE     LOOP1                   ;;BRANCH IF NO
                                        BIT     #SW7,@$SWRP                     ;INHIBIT SIZE TYPEOUT?
                                        BNE     SW8MSG                  ;;BRANCH IF YES
                                        JSR     PC,TYPSIZ                       ;GO TYPE SYSTEM SIZE
                                        TYPE    ,69$                    ;;TYPE ASCIZ STRING
                                        BR      68$                     ;;GET OVER THE ASCIZ
                                ;;69$:  .ASCIZ  /TYPE A CHARACTER TO CONTINUE/<CRLF>
                                68$:
                                        CLR     @#PSW
                                        WAIT
                                        JMP     @#SIZE                  ;GO CHECK SYSTEM AGAIN
                                SW8MSG: TYPE    ,MSG30                  ;TYPE SWITCH 8 REVERSAL MESSAGE
                                LOOP1:  JMP     LOOP
                                .=10000
                                ;PROGRAM RESTARTS HERE AFTER RELOCATION ABOVE 28K IS COMPLETE.
                                ;INITIALIZE TRAP VECTORS
                                LOOP:   MOV     #SUPSTK,SP                      ;SET THE STACK...WILL BE DIFFERENT
                                                                                ;THAN KERN STACK WHEN IN OUTER MODE
                                        MOV     #ERRVEC,R0
                                        MOV     @#PSW,R1                        ;GET CURRENT PSW
                                        MOV     #ERPRT,(R0)+                    ;SET ERROR VEC
                                        BIS     #PR7,R1                 ;SET PRIORITY 7 IN CURRENT PSW
                                        BIC     #BIT4,R1                ;CLEAR T BIT
                                        MOV     R1,(R0)+
                                        MOV     #RESERR,(R0)+                   ;SET RESERVED INST TRAP VECTOR
                                        MOV     R1,(R0)+
                                        MOV     #$RTRN,(R0)+                    ;SET T BIT VEC
                                        BIC     #PR7,R1
                                        CLR     (R0)+                           ;SET TBIT VEC+2
                                        TST     (R0)+                           ;BUMP R0 TO SCOPE VEC+2
                                        CLR     (R0)+                           ;SET SCOPE VEC+2
                                        ADD     #6,R0                           ;SET R0 TO ERROR TRAP VEC
                                        MOV     #PR7,(R0)+                      ;SET ERROR VEC
                                        TST     (R0)+
                                        MOV     #PR7,(R0)+                      ;SET TRAP VEC+2
                                        MOV     #.PARSRV,@#CACHVEC              ;SET PARITY ERROR VECTOR
                                        BIS     #PR7,R1
                                        MOV     R1,@#CACHVEC+2
                                        MOV     #KTABRT,@#MMVEC                 ;SET KT11 ABORT VECTOR
                                        MOV     R1,@#MMVEC+2
                                        BIC     #PR7,@#PSW
                                ;;**************************************************************
                                ;*TEST 1         MEMORY VERIFICATION TEST
                                ;;**************************************************************
```

```
   (2)  010132  012767  000001  171236          MOV     #1,$TIMES       ;;DO 1 ITERATION
   (2)  010140  000004                  TST1:   SCOPE
   (2)  010142  112737  000001  001252          MOVB    #1,@#$TSTNM             ;LOAD TEST NUMBER
   (2)  010150  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
  6687
   (1)                                          .SBTTL  START OF SECTION 0
   (1)                                  ;0000000000000 FIRST ADDRESS TO BE RELOCATED 000000000
   (1)  010156  010700                  RELO:   MOV     PC,R0           ;GET PC
   (1)  010160  005740                          TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF RELO
   (1)  010162  010037  001610                  MOV     R0,@#FRSTAD     ;SAVE
   (1)  010166  010700                          MOV     PC,R0           ;GET CURRENT PC
   (1)  010170  162700  010170                  SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
   (1)  010174  010037  001604                  MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
   (1)  010200  010737  001262                  MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
   (1)  010204  062737  000030  001262          ADD     #30,@#$LPERR    ;ADJUST
   (1)  010212  013737  001262  001260          MOV     @#$LPERR,@#$LPADR
   (1)  010220  105737  001600                  TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
   (1)  010224  001402                          BEQ     .+6
   (1)  010226  000167  000720                  JMP     RELO
  6688                                  ;MEMORY AND DISK (IF SELECTED) VERIFICATION TEST.
  6689  010232  000167  000714                  JMP     1$
  6692  010236  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
   (1)  010244  177777  000000  000000
   (1)  010252  000000  000000
   (1)  010256  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
   (1)  010264  177777  000000  000000
   (1)  010272  000000  000000
   (1)  010276  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
   (1)  010304  177777  000000  000000
   (1)  010312  000000  000000
   (1)  010316  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
   (1)  010324  177777  000000  000000
   (1)  010332  000000  000000
   (1)  010336  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
   (1)  010344  177777  000000  000000
   (1)  010352  000000  000000
   (1)  010356  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
   (1)  010364  177777  000000  000000
   (1)  010372  000000  000000
   (1)  010376  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
   (1)  010404  177777  000000  000000
   (1)  010412  000000  000000
   (1)  010416  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
   (1)  010424  177777  000000  000000
   (1)  010432  000000  000000
   (1)  010436  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
   (1)  010444  177777  000000  000000
   (1)  010452  000000  000000
   (1)  010456  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
   (1)  010464  177777  000000  000000
   (1)  010472  000000  000000
   (1)  010476  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
   (1)  010504  177777  000000  000000
   (1)  010512  000000  000000
   (1)  010516  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
   (1)  010524  177777  000000  000000
```

```
(1)  010532  000000  000000
(1)  010536  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  010544  177777  000000  000000
(1)  010552  000000  000000
(1)  010556  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  010564  177777  000000  000000
(1)  010572  000000  000000
(1)  010576  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  010604  177777  000000  000000
(1)  010612  000000  000000
(1)  010616  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  010624  177777  000000  000000
(1)  010632  000000  000000
(1)  010636  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  010644  177777  000000  000000
(1)  010652  000000  000000
(1)  010656  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  010664  177777  000000  000000
(1)  010672  000000  000000
(1)  010676  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  010704  177777  000000  000000
(1)  010712  000000  000000
(1)  010716  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  010724  177777  000000  000000
(1)  010732  000000  000000
(1)  010736  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  010744  177777  000000  000000
(1)  010752  000000  000000
(1)  010756  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  010764  177777  000000  000000
(1)  010772  000000  000000
(1)  010776  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  011004  177777  000000  000000
(1)  011012  000000  000000
(1)  011016  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  011024  177777  000000  000000
(1)  011032  000000  000000
(1)  011036  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  011044  177777  000000  000000
(1)  011052  000000  000000
(1)  011056  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  011064  177777  000000  000000
(1)  011072  000000  000000
(1)  011076  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  011104  177777  000000  000000
(1)  011112  000000  000000
(1)  011116  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
(1)  011124  177777  000000  000000
(1)  011132  000000  000000
6693 011136  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0
     011144  177777  000000  000000
6694 011152                          1$:
6695 011152  000004                 RELE0:  SCOPE
(1)  011154  010702                         MOV     PC,R2
(1)  011156  062702  000012                 ADD     #12,R2
(1)  011162  012707  044042                 MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
```

```
  (1)  011166  000000                    REL00:  .WORD   0
  (1)                                    ;0000000000000 LAST ADDRESS OF CODE TO BE RELOCATED 00000000000
  (1)
 6696                                    ;;*****************************************************************
  (3)                                    ;*TEST 2         CHECK BRANCH INSTRUCTIONS
  (3)                                    ;;*****************************************************************
  (2)
  (2)  011170  012767  000001  170200            MOV     #1,$TIMES       ;;DO 1 ITERATION
  (2)  011176  000004            TST2:    SCOPE
  (2)  011200  112737  000002  001252            MOVB    #2,@#$TSTNM             ;LOAD TEST NUMBER
  (2)  011206  013737  001252  177570            MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 6697
  (1)                                            .SBTTL  START OF SECTION 1
  (1)                                    ;1111111111111 FIRST ADDRESS TO BE RELOCATED 111111111
  (1)  011214  010700            REL1:    MOV     PC,R0           ;GET PC
  (1)  011216  005740                     TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL1
  (1)  011220  010037  001610             MOV     R0,@#FRSTAD     ;SAVE
  (1)  011224  010700                     MOV     PC,R0           ;GET CURRENT PC
  (1)  011226  162700  011226             SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
  (1)  011232  010037  001604             MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
  (1)  011236  010737  001262             MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
  (1)  011242  062737  000030  001262     ADD     #30,@#$LPERR    ;ADJUST
  (1)  011250  013737  001262  001260     MOV     @#$LPERR,@#$LPADR
  (1)  011256  105737  001600             TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
  (1)  011262  001402                     BEQ     .+6
  (1)  011264  000167  004146             JMP     RELE1
 6698                                    ;
 6699  011270  000257                     CCC                     ;CC'S=0000
 6700  011272  103407                     BCS     CC0             ;SAME AS BLO
 6701  011274  102406                     BVS     CC0
 6702  011276  001405                     BEQ     CC0
 6703  011300  100404                     BMI     CC0
 6704  011302  002403                     BLT     CC0
 6705  011304  003402                     BLE     CC0
 6706  011306  101401                     BLOS    CC0
 6707  011310  101001                     BHI     .+4
 6708  011312  104000            CC0:     HLT                     ;ONE OF THE ABOVE BRANCHES FAILED
 6709
 6710                                    ;CONTINUE
 6711  011314  000270                     SEN                     ;CC'S=1000
 6712  011316  100003                     BPL     CC1
 6713  011320  002002                     BGE     CC1
 6714  011322  003001                     BGT     CC1
 6715  011324  002401                     BLT     .+4
 6716  011326  104000            CC1:     HLT                     ;ONE OF THE ABOVE BRANCHES FAILED
 6717
 6718                                    ;CONTINUE
 6719  011330  000262                     SEV                     ;CC'S=1010
 6720  011332  102003                     BVC     CC2
 6721  011334  002402                     BLT     CC2
 6722  011336  003401                     BLE     CC2
 6723  011340  002001                     BGE     .+4
 6724  011342  104000            CC2:     HLT                     ;ERROR! ONE OF THE ABOVE BRANCHES FAILED
 6725
 6726                                    ;CONTINUE
 6727  011344  000261                     SEC                     ;CC'S=1011
```

```
 6728  011346  103002                    BCC     CC3
 6729  011350  101001                    BHI     CC3
 6730  011352  003001                    BGT     .+4
 6731  011354  104000            CC3:    HLT                     ;ERROR! ONE OF THE ABOVE BRANCHES FAILED
 6732
 6733                            ;CONTINUE
 6734  011356  000264                    SEZ                     ;CC'S=1111
 6735  011360  001003                    BNE     CC4
 6736  011362  003002                    BGT     CC4
 6737  011364  101001                    BHI     CC4
 6738  011366  003401                    BLE     .+4
 6739  011370  104000            CC4:    HLT                     ;ERROR! ONE OF THE ABOVE BRANCHES FAILED
 6740                            ;;*********************************************************************
  (3)                            ;*TEST 3        TEST UNIARY CONDITION CODES
  (3)                            ;;*********************************************************************
  (2)
  (2)  011372  000004            TST3:   SCOPE
  (2)  011374  112737  000003  001252    MOVB    #3,@#$TSTNM             ;LOAD TEST NUMBER
  (2)  011402  013737  001252  177570    MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 6741                            ;CLR    R0
 6742  011410  000277                    SCC
 6743  011412  000244                    CLZ
 6744  011414  005000                    CLR     R0                      ;R0=0,CC'S=0100
 6745  011416  103404                    BCS     CLR0
 6746  011420  102403                    BVS     CLR0
 6747  011422  001002                    BNE     CLR0
 6748  011424  100401                    BMI     CLR0
 6749  011426  003401                    BLE     .+4
 6750  011430  104000            CLR0:   HLT                     ;ERROR! INCORRECT CC'S AFTER CLR
 6751
 6752  011432  000277                    SCC
 6753  011434  000244                    CLZ
 6754  011436  005700                    TST     R0                      ;R0=0,CC'S=0100
 6755  011440  103404                    BCS     TST0
 6756  011442  102403                    BVS     TST0
 6757  011444  001002                    BNE     TST0
 6758  011446  100401                    BMI     TST0
 6759  011450  101401                    BLOS    .+4
 6760  011452  104000            TST0:   HLT                     ;ERROR! INCORRECT CC'S AFTER TST
 6761
 6762  011454  000257                    CCC
 6763  011456  000266                    +SEZ!SEV
 6764  011460  005100                    COM     R0                      ;R0=-1,CC'S=1001
 6765  011462  103004                    BCC     COM0
 6766  011464  102403                    BVS     COM0
 6767  011466  001402                    BEQ     COM0
 6768  011470  100001                    BPL     COM0
 6769  011472  002401                    BLT     .+4
 6770  011474  104000            COM0:   HLT                     ;ERROR! INCORRECT CC'S AFTER COM
 6771
 6772  011476  000261                    SEC
 6773  011500  005500                    ADC     R0                      ;R0=000000,CC'S=0101
 6774  011502  103003                    BCC     ADC0
 6775  011504  102402                    BVS     ADC0
 6776  011506  001001                    BNE     ADC0
 6777  011510  002001                    BGE     .+4
```

```
 6778  011512  104000                ADC0:   HLT                     ;ERROR! INCORRECT CC'S AFTER ADC
 6779
 6780  011514  000261                        SEC
 6781  011516  006000                        ROR     R0              ;R0=100000,CC'S=1010
 6782  011520  103404                        BCS     ROR0
 6783  011522  102003                        BVC     ROR0
 6784  011524  001402                        BEQ     ROR0
 6785  011526  100001                        BPL     ROR0
 6786  011530  003001                        BGT     .+4
 6787  011532  104000                ROR0:   HLT                     ;ERROR! INCORRECT CC'S AFTER ROR
 6788  011534  000277                        SCC
 6789  011536  000242                        CLV
 6790  011540  005300                        DEC     R0              ;R0=077777,CC'S=0011
 6791  011542  103004                        BCC     DEC0
 6792  011544  102003                        BVC     DEC0
 6793  011546  001402                        BEQ     DEC0
 6794  011550  100401                        BMI     DEC0
 6795  011552  003401                        BLE     .+4
 6796  011554  104000                DEC0:   HLT                     ;ERROR! INCORRECT CC'S AFTER DEC
 6797
 6798  011556  000257                        CCC
 6799  011560  005200                        INC     R0              ;R0=100000,CC'S=1010
 6800  011562  103404                        BCS     INC0
 6801  011564  102003                        BVC     INC0
 6802  011566  001402                        BEQ     INC0
 6803  011570  100001                        BPL     INC0
 6804  011572  003001                        BGT     .+4
 6805  011574  104000                INC0:   HLT                     ;ERROR! INCORRECT CC'S AFTER INC
 6806
 6807  011576  000277                        SCC
 6808  011600  000242                        CLV
 6809  011602  005400                        NEG     R0              ;R0=100000,CC'S=1011
 6810  011604  103003                        BCC     NEG0
 6811  011606  102002                        BVC     NEG0
 6812  011610  001401                        BEQ     NEG0
 6813  011612  002001                        BGE     .+4
 6814  011614  104000                NEG0:   HLT                     ;ERROR! INCORRECT CC'S AFTER NEG
 6815
 6816  011616  000261                        SEC
 6817  011620  006300                        ASL     R0              ;R0=000000,CC'S=0111
 6818  011622  103004                        BCC     ASL0
 6819  011624  102003                        BVC     ASL0
 6820  011626  001002                        BNE     ASL0
 6821  011630  100401                        BMI     ASL0
 6822  011632  101401                        BLOS    .+4
 6823  011634  104000                ASL0:   HLT                     ;ERROR! INCORRECT CC'S AFTER ASL
 6824
 6825  011636  006100                        ROL     R0              ;R0=000001,CC'S=0000
 6826  011640  103402                        BCS     ROL0
 6827  011642  003401                        BLE     ROL0
 6828  011644  002001                        BGE     .+4
 6829  011646  104000                ROL0:   HLT                     ;ERROR! INCORRECT CC'S AFTER ROL
 6830
 6831  011650  006200                        ASR     R0              ;R0=000000,CC'S=0111
 6832  011652  103003                        BCC     ASR0
 6833  011654  102002                        BVC     ASR0
```

```
6834  011656  001001                        BNE     ASR0
6835  011660  002401                        BLT     .+4
6836  011662  104000                ASR0:   HLT                     ;ERROR! INCORRECT CC'S AFTER ASR
6837
6838  011664  000277                        SCC
6839  011666  005600                        SBC     R0              ;R0=-1,CC'S=1001
6840  011670  103002                        BCC     SBC0
6841  011672  102401                        BVS     SBC0
6842  011674  003401                        BLE     .+4
6843  011676  104000                SBC0:   HLT                     ;ERROR! INCORRECT CC'S AFTER SBC
6844
6845  011700  005400                        NEG     R0              ;R0=000001,CC'S=00001
6846  011702  000300                        SWAB    R0              ;R0=000400 ,CC'S=0100
6847  011704  103403                        BCS     SWAB0
6848  011706  102402                        BVS     SWAB0
6849  011710  001001                        BNE     SWAB0
6850  011712  002001                        BGE     .+4
6851  011714  104000                SWAB0:  HLT                     ;ERROR! INCORRECT CC'S AFTER SWAB
6852                                        ;;*******************************************************************
  (3)                                       ;*TEST 4         CHECK REGISTER SELECTION
  (3)                                       ;;*******************************************************************
  (2)
  (2)   011716  000004                TST4:   SCOPE
  (2)   011720  112737  000004  001252        MOVB    #4,@#$TSTNM             ;LOAD TEST NUMBER
  (2)   011726  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
6853    011734  012737  000005  001376        MOV     #5,@#$TIMES     ;SET ITERATION COUNT TO 5
6854    011742  005000                        CLR     R0
6855    011744  000277                        SCC
6856    011746  006100                        ROL     R0              ;R0=1
6857    011750  010002                        MOV     R0,R2
6858    011752  006302                        ASL     R2              ;R2=2
6859    011754  010203                        MOV     R2,R3
6860    011756  006303                        ASL     R3              ;R3=4
6861    011760  010304                        MOV     R3,R4
6862    011762  006304                        ASL     R4              ;R4=10
6863    011764  010405                        MOV     R4,R5
6864    011766  006305                        ASL     R5              ;R5=20
6865    011770  010546                        MOV     R5,-(SP)        ;SET BITS SET IN REGISTERS
6866    011772  050416                        BIS     R4,(SP)         ;INTO STACK ADDRESS
6867    011774  050316                        BIS     R3,(SP)
6868    011776  050216                        BIS     R2,(SP)
6869    012000  050016                        BIS     R0,(SP)
6870    012002  022726  000037                CMP     #37,(SP)+
6871    012006  001401                        BEQ     .+4             ;WERE SET
6872    012010  104000                        HLT                     ;MISSING BIT(S) REPRESENT
6873                                                                    ;INCORRECT REGISTER SELECTION
6874
6875                                ;CHECK THAT ALL BITS CAN BE SET & CLEARED IN ALL REGISTERS
6876    012012  000257                        CCC
6877    012014  112700  000377                MOVB    #377,R0         ;SET ALL BITS (MOVB EXTENDS SIGN)
6878    012020  006100                1$:     ROL     R0              ;ROTATE A 0 THROUGH ALL BIT
6879    012022  103776                        BCS     1$              ;POSITIONS
6880    012024  005200                        INC     R0              ;FINAL RESULT IS -1
6881    012026  001401                        BEQ     .+4
6882    012030  104000                        HLT                     ;ERROR!
6883
```

```
 6884  012032  012700  000020                MOV     #16.,R0         ;SET SHIFT COUNT
 6885  012036  005002                        CLR     R2
 6886  012040  000261              2$:       SEC
 6887  012042  006002                        ROR     R2              ;ROTATE 1 THROUGH ALL BIT POSITS
 6888  012044  005300                        DEC     R0              ;DECREMENT SHIFT COUNT
 6889  012046  001374                        BNE     2$
 6890  012050  005102                        COM     R2              ;R2 SHOULD CONTAIN -1
 6891  012052  001401                        BEQ     .+4
 6892  012054  104000                        HLT                     ;ERROR! CHECK R2 SHOULD = 0
 6893
 6894  012056  012703  100000                MOV     #100000,R3
 6895  012062  006203              3$:       ASR     R3              ;EXTEND 1 BIT THROUGH ALL POSITIONS
 6896  012064  103376                        BCC     3$
 6897  012066  005203                        INC     R3
 6898  012070  001401                        BEQ     .+4
 6899  012072  104000                        HLT                     ;ERROR!
 6900
 6901  012074  112704  177401                MOVB    #177401,R4      ;R4=1
 6902  012100  060404              4$:       ADD     R4,R4           ;HAS THE AFFECT OF SHIFTING A BIT
 6903  012102  103376                        BCC     4$              ;THROUGH ALL POSITIONS
 6904  012104  005704                        TST     R4              ;RESULT SHOULD BE 0
 6905  012106  001401                        BEQ     .+4
 6906  012110  104000                        HLT
 6907
 6908  012112  012705  000001                MOV     #1,R5
 6909  012116  006305              5$:       ASL     R5
 6910  012120  102376                        BVC     5$
 6911  012122  006305                        ASL     R5
 6912  012124  103002                        BCC     6$
 6913  012126  005705                        TST     R5
 6914  012130  001401                        BEQ     .+4
 6915  012132  104000              6$:       HLT
 6916
 6917                                  ;CHECK REGISTER VOLITILITY
 6918  012134  005002                        CLR     R2
 6919  012136  005102                        COM     R2              ;R2=-1
 6920  012140  010203                        MOV     R2,R3
 6921  012142  000257                        CCC
 6922  012144  006002                        ROR     R2              ;R2=LOOP COUNT
 6923  012146  006202                        ASR     R2
 6924  012150  010304              7$:       MOV     R3,R4
 6925  012152  005302                        DEC     R2              ;DECREMENT LOOP COUNT
 6926  012154  001375                        BNE     7$
 6927  012156  005203                        INC     R3              ;CHECK R3
 6928  012160  001002                        BNE     8$
 6929  012162  005204                        INC     R4              ;CHECK R4
 6930  012164  001401                        BEQ     .+4
 6931  012166  104000              8$:       HLT
 6932
 6933                                  ;CHECK TRANSFER OF REGISTER DATA BETWEEN THE GS AND GD REGISTERS
 6934  012170  032737  000020  177776  GSTST:  BIT     #20,@#PSW       ;CHECK IF 'T' BIT IS SET
 6935  012176  001050                        BNE     7$              ;SKIP TEST IF 'T' BIT SET
 6936  012200  010627                        MOV     SP,(PC)+        ;SAVE STACK PTR
 6937  012202  000000              1$:       .WORD   0               ;CONTAINS SAVED STACK PTR
 6938  012204  010727                        MOV     PC,(PC)+        ;LOAD DATA. THE CURRENT PC IS USED AS
 6939  012206  000000              2$:       .WORD   0               ;DATA. IF THIS TEST FAILS 2$ CON-
```

```
                                                ;TAINS THE DATA BEING USED.
                        INC     2$              ;MAKE ODD TO CHECK BIT 0
                3$:     MOV     2$,R0           ;LOAD GD REGISTER 0
                        MOV     R0,R1           ;TRANSFER GS REG 0 TO GD REG 1
                        MOV     R1,R2           ;AND GS REG 1 TO GD REG 2
                        MOV     R2,R3           ;ETC...
                        MOV     R3,R4
                        MOV     R4,R5
                        BISB    #340,@#PSW      ;SET PRIORITY LEVEL 7
                        MOV     R5,SP           ;TRANSFER GS REG 5 TO GD STK PTR
                        MOV     SP,(PC)+        ;TRANSFER GS STK PTR TO MEMORY
                4$:     .WORD   0               ;CONTAINS GS STACK PTR
                        MOV     1$,SP           ;RESTORE STK PTR NEEDED FOR HLT/SCOPE
                        BICB    #340,@#PSW      ;SET PRIORITY LEVEL 0
                        CMP     4$,R0           ;COMPARE GS/GD STACK WITH GS REG 0
                        BNE     5$              ;BRANCH IF THEY WERE NOT =
                        ASL     2$              ;SHIFT TEST DATA UNTIL = 000000
                        BNE     3$
                        BR      6$
                5$:     MOV     R0,-(SP)        ;GET GS REG 0
                        MOV     R1,-(SP)        ;ETC...
                        MOV     R2,-(SP)
                        MOV     R3,-(SP)
                        MOV     R4,-(SP)
                        MOV     R5,-(SP)
                        HLT                     ;ERROR! DATA IN GS STK PTR NOT = GS REG 0
                                                ;GS REG 0-GS REG 5 ARE ON THE STACK
                        MOV     1$,SP           ;RESTORE STACK PTR
                6$:
                7$:
                ;;*******************************************************************
                ;*TEST 5         TEST UNIARY WORD INSTRUCTIONS USING ADDRESS MODE 1
                ;;*******************************************************************
                TST5:   SCOPE
                        MOVB    #5,@#$TSTNM             ;LOAD TEST NUMBER
                        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
                        MOV     #5,@#$TIMES
                        BR      .+4
                        .WORD   0               ;RESERVE ADDRESS FOR TESTS
                        MOV     PC,R2
                        SUB     #4,R2           ;R2 POINTS TO RESERVED WORD
                        CLR     (R2)            ;PRESET (R2)

                        SEC
                        ROR     (R2)            ;(R2)=100000,CC=1010
                        BLOS    ROR1
                        BPL     ROR1
                        BGE     .+4
                ROR1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

                        CCC
                        SEC
                        DEC     (R2)            ;(R2)=077777,CC=0011
                        BCC     DEC1
                        BLE     .+4
```

```
6990  012406  104000                DEC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
6991
6992  012410  000257                        CCC
6993  012412  000261                        SEC
6994  012414  005512                        ADC     (R2)            ;(R2)=100000,CC=1010
6995  012416  103403                        BCS     ADC1
6996  012420  102002                        BVC     ADC1
6997  012422  100001                        BPL     ADC1
6998  012424  001001                        BNE     .+4
6999  012426  104000                ADC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7000
7001  012430  006112                        ROL     (R2)            ;(R2)=000000,CC=0111
7002  012432  103003                        BCC     ROL1
7003  012434  102002                        BVC     ROL1
7004  012436  001001                        BNE     ROL1
7005  012440  100001                        BPL     .+4
7006  012442  104000                ROL1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7007
7008  012444  006112                        ROL     (R2)            ;(R2)=000001,CC=0000
7009  012446  101402                        BLOS    ROL1A           ;BRANCH IF C OR Z IS SET
7010  012450  102401                        BVS     ROL1A
7011  012452  100001                        BPL     .+4
7012  012454  104000                ROL1A:  HLT
7013
7014  012456  006212                        ASR     (R2)            ;(R2)=000000,CC=0111
7015  012460  103003                        BCC     ASR1
7016  012462  102002                        BVC     ASR1
7017  012464  001001                        BNE     ASR1
7018  012466  100001                        BPL     .+4
7019  012470  104000                ASR1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7020
7021  012472  006012                        ROR     (R2)            ;(R2)=100000,CC=1010
7022  012474  103403                        BCS     ROR1A
7023  012476  102002                        BVC     ROR1A
7024  012500  001401                        BEQ     ROR1A
7025  012502  100401                        BMI     .+4
7026  012504  104000                ROR1A:  HLT
7027
7028  012506  000261                        SEC
7029  012510  005212                        INC     (R2)            ;(R2)=100001,CC=1001
7030  012512  103003                        BCC     INC1
7031  012514  102402                        BVS     INC1
7032  012516  001401                        BEQ     INC1
7033  012520  100401                        BMI     .+4
7034  012522  104000                INC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7035
7036  012524  005612                        SBC     (R2)            ;(R2)=100000,CC=1000
7037  012526  103403                        BCS     SBC1
7038  012530  102402                        BVS     SBC1
7039  012532  001401                        BEQ     SBC1
7040  012534  100401                        BMI     .+4
7041  012536  104000                SBC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7042
7043  012540  000261                        SEC
7044  012542  005612                        SBC     (R2)            ;(R2)=077777,CC=0010
7045  012544  103403                        BCS     SBC1A
```

```
7046  012546  102002                        BVC     SBC1A
7047  012550  001401                        BEQ     SBC1A
7048  012552  100001                        BPL     .+4
7049  012554  104000              SBC1A:    HLT                     ;ERROR! INCORRECT CC'S AS SHOEN ABOVE
7050
7051  012556  000261                        SEC
7052  012560  005512                        ADC     (R2)            ;(R2)=100000,CC=1010
7053  012562  100401                        BMI     .+4
7054  012564  104000                        HLT
7055
7056  012566  000261                        SEC
7057  012570  006312                        ASL     (R2)            ;(R2)=000000,CC=0111
7058  012572  103003                        BCC     ASL1
7059  012574  102002                        BVC     ASL1
7060  012576  001001                        BNE     ASL1
7061  012600  100001                        BPL     .+4
7062  012602  104000              ASL1:     HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7063
7064  012604  005112                        COM     (R2)            ;(R2)=177777,CC=1001
7065  012606  103002                        BCC     COM1
7066  012610  102401                        BVS     COM1
7067  012612  100401                        BMI     .+4
7068  012614  104000              COM1:     HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7069
7070  012616  000250                        CLN
7071  012620  005712                        TST     (R2)            ;(R2)=177777,CC=1000
7072  012622  103403                        BCS     TEST1
7073  012624  102402                        BVS     TEST1
7074  012626  100001                        BPL     TEST1
7075  012630  001001                        BNE     .+4
7076  012632  104000              TEST1:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7077
7078  012634  000262                        SEV
7079  012636  005412                        NEG     (R2)            ;(R2)=000001,CC=0000
7080  012640  103002                        BCC     NEG1
7081  012642  102401                        BVS     NEG1
7082  012644  001001                        BNE     .+4
7083  012646  104000              NEG1:     HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7084
7085  012650  005312                        DEC     (R2)            ;(R2)=000000,CC=0101
7086  012652  103001                        BCC     DEC1A
7087  012654  001401                        BEQ     .+4
7088  012656  104000              DEC1A:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7089                                      ;;*********************************************************************
 (3)                                    ;*TEST 6        CHECK UNIARY BYTE INSTRUCTIONS USING ADDRESS MODE 1
 (3)                                    ;;*********************************************************************
 (2)
 (2)  012660  000004              TST6:     SCOPE
 (2)  012662  112737  000006 001252        MOVB    #6,@#$TSTNM             ;LOAD TEST NUMBER
 (2)  012670  013737  001252 177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
7090  012676  000401                        BR      .+4             ;RESERVE A WORD
7091  012700  000000                        .WORD   0               ;ADDRESS RESERVED FOR TESTS
7092  012702  010703                        MOV     PC,R3
7093  012704  162703  000004              SUB     #4,R3           ;R3 POINTS TO EVEN BYTE OF WORD
7094  012710  010304                        MOV     R3,R4           ;R4 POINTS TO ODD BYTE OF WORD
7095  012712  005204                        INC     R4
```

```
7096  012714  005013                    CLR     (R3)            ;PRESET DATA
7097
7098  012716  000261              1$:   SEC
7099  012720  105513                    ADCB    (R3)            ;ADD CARRY TO EVEN BYTE
7100  012722  100402                    BMI     2$              ;UNTIL EVEN BYTE BECOMES NEGATIVE
7101  012724  105214                    INCB    (R4)            ;INCREMENT ODD BYTE
7102  012726  000773                    BR      1$
7103  012730  102401              2$:   BVS     .+4             ;(R3)=077600=[0774][200],CC=1010
7104  012732  104000                    HLT
7105  012734  000242                    CLV
7106  012736  105214                    INCB    (R4)            ;(R3)=100200=[1000][200],CC=1010
7107  012740  103402                    BCS     INCB1
7108  012742  102001                    BVC     INCB1
7109  012744  100401                    BMI     .+4
7110  012746  104000              INCB1:HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7111
7112  012750  106114                    ROLB    (R4)            ;(R3)=000200=[0000][200],CC=0111
7113  012752  103002                    BCC     ROLB1
7114  012754  102001                    BVC     ROLB1
7115  012756  001401                    BEQ     .+4
7116  012760  104000              ROLB1:HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7117
7118  012762  105614                    SBCB    (R4)            ;(R3)=177600=[1774][200], CC=1001
7119  012764  103002                    BCC     SBCB1
7120  012766  102401                    BVS     SBCB1
7121  012770  100401                    BMI     .+4
7122  012772  104000              SBCB1:HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7123
7124  012774  106313                    ASLB    (R3)            ;(R3)=177400,CC=0111
7125  012776  103002                    BCC     ASLB1
7126  013000  102001                    BVC     ASLB1
7127  013002  001401                    BEQ     .+4
7128  013004  104000              ASLB1:HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7129
7130  013006  105413                    NEGB    (R3)            ;(R3)=177400,CC=0100
7131  013010  103402                    BCS     NEGB1
7132  013012  102401                    BVS     NEGB1
7133  013014  001401                    BEQ     .+4
7134  013016  104000              NEGB1:HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7135
7136  013020  000277                    SCC
7137  013022  105313                    DECB    (R3)            ;(R3)=177777,CC=1001
7138  013024  103002                    BCC     DECB1
7139  013026  102401                    BVS     DECB1
7140  013030  001001                    BNE     .+4
7141  013032  104000              DECB1:HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7142
7143  013034  000241                    CLC
7144  013036  106013                    RORB    (R3)            ;(R3)=177577,CC=0011
7145  013040  103002                    BCC     RORB1
7146  013042  102001                    BVC     RORB1
7147  013044  100001                    BPL     .+4
7148  013046  104000              RORB1:HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7149
7150  013050  000241                    CLC
7151  013052  105114                    COMB    (R4)            ;(R3)=000177,CC=0101
```

```
7152  013054  103002                    BCC     COMB1
7153  013056  102401                    BVS     COMB1
7154  013060  001401                    BEQ     .+4
7155  013062  104000            COMB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7156
7157  013064  106213            1$:     ASRB    (R3)            ;SHIFT EVEN BYTE UNTIL V CLEARS
7158  013066  102002                    BVC     2$
7159  013070  105514                    ADCB    (R4)            ;AND ADD CARRY TO ODD BYTE
7160  013072  000774                    BR      1$
7161  013074  103401            2$:     BCS     ASRB1
7162  013076  001401                    BEQ     .+4
7163  013100  104000            ASRB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7164
7165  013102  106214                    ASRB    (R4)
7166  013104  106214                    ASRB    (R4)            ;(R3)=000400,CC=0011
7167  013106  103002                    BCC     ASRB1A
7168  013110  102001                    BVC     ASRB1A
7169  013112  001001                    BNE     .+4
7170  013114  104000            ASRB1A: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7171
7172  013116  105314                    DECB    (R4)            ;(R3)=000000,CC=0100
7173  013120  001401                    BEQ     .+4
7174  013122  104000                    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7175
7176  013124  000261                    SEC
7177  013126  106014                    RORB    (R4)            ;(R3)=100000,CC=1010
7178  013130  103402                    BCS     RORB1A
7179  013132  102001                    BVC     RORB1A
7180  013134  100401                    BMI     .+4
7181  013136  104000            RORB1A: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7182
7183  013140  000242                    CLV
7184  013142  105314                    DECB    (R4)            ;(R3)=077400,CC=0100
7185  013144  102401                    BVS     .+4
7186  013146  104000                    HLT
7187
7188  013150  000261                    SEC
7189  013152  105313                    DECB    (R3)            ;(R3)=077777,CC=1001
7190  013154  103002                    BCC     DECB1A
7191  013156  102401                    BVS     DECB1A
7192  013160  100401                    BMI     .+4
7193  013162  104000            DECB1A: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7194
7195  013164  000277                    SCC
7196  013166  000313                    SWAB    (R3)            ;(R3)=177577=[1774][177],CC=0000
7197  013170  103402                    BCS     SWAB1
7198  013172  102401                    BVS     SWAB1
7199  013174  100001                    BPL     .+4
7200  013176  104000            SWAB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7201
7202  013200  105714                    TSTB    (R4)            ;(R3)=177577=[1774][177],CC=1000
7203  013202  103402                    BCS     TSTB1
7204  013204  102401                    BVS     TSTB1
7205  013206  100401                    BMI     .+4
7206  013210  104000            TSTB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7207
```

```
 7208  013212  105014                          CLRB    (R4)            ;(R3)=000177=[0000][177],CC=0100
 7209  013214  001401                          BEQ     .+4
 7210  013216  104000                          HLT
 7211  013220  106313                          ASLB    (R3)            ;(R3)=000376 ,CC=1010
 7212  013222  103402                          BCS     ASLB1A
 7213  013224  102001                          BVC     ASLB1A
 7214  013226  100401                          BMI     .+4
 7215  013230  104000              ASLB1A:     HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
 7216
 7217
 7218  013232  105113                          COMB    (R3)            ;(R3)=000001,CC=0001
 7219  013234  103002                          BCC     COMB1A
 7220  013236  102401                          BVS     COMB1A
 7221  013240  100001                          BPL     .+4
 7222  013242  104000              COMB1A:     HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
 7223
 7224  013244  000313                          SWAB    (R3)            ;(R3)=000400, CC=0100
 7225  013246  001401                          BEQ     .+4
 7226  013250  104000                          HLT
 7227
 7228  013252  105213                          INCB    (R3)
 7229  013254  000261                          SEC
 7230  013256  105613                          SBCB    (R3)            ;(R3)=000400,CC=0100
 7231  013260  001401                          BEQ     .+4
 7232  013262  104000                          HLT
 7233  013264  022713  000400                  CMP     #400,(R3)       ;CHECK REMAINING RESULT
 7234  013270  001401                          BEQ     .+4
 7235  013272  104000                          HLT
 7236                          ;;*****************************************************************
  (3)                          ;*TEST 7        CHECK UNIARY WORD OPS USING ADDRESS MODES 2 & 4
  (3)                          ;;*****************************************************************
  (2)
  (2)  013274  000004          TST7:   SCOPE
  (2)  013276  112737  000007  001252          MOVB    #7,@#$TSTNM             ;LOAD TEST NUMBER
  (2)  013304  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 7237  013312  000401                          BR      .+4
 7238  013314  000000                          .WORD   0               ;ADDRESS RESERVED FOR TESTS
 7239  013316  010704                          MOV     PC,R4
 7240  013320  162704  000004                  SUB     #4,R4           ;R4 AND R5 POINT TO
 7241  013324  010405                          MOV     R4,R5           ;RESERVED WORD
 7242  013326  005015                          CLR     (R5)            ;PRESET DATA=0
 7243
 7244  013330  000277                          SCC
 7245  013332  000244                          CLZ
 7246  013334  005725                          TST     (R5)+           ;(R5)=000000,CC=0100
 7247  013336  103402                          BCS     TEST2
 7248  013340  102401                          BVS     TEST2
 7249  013342  001401                          BEQ     .+4
 7250  013344  104000              TEST2:      HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
 7251
 7252  013346  005145                          COM     -(R5)           ;(R5)=177777,CC=1001
 7253  013350  103001                          BCC     COM4
 7254  013352  100401                          BMI     .+4
 7255  013354  104000              COM4:       HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
 7256
 7257  013356  000241                          CLC
```

```
7258  013360  006024                    ROR     (R4)+           ;(R4)=077777,CC=0011
7259  013362  103002                    BCC     ROR2
7260  013364  102001                    BVC     ROR2
7261  013366  100001                    BPL     .+4
7262  013370  104000            ROR2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7263
7264  013372  000257                    CCC
7265  013374  005244                    INC     -(R4)           ;(R4)=100000,CC=1010
7266  013376  102002                    BVC     INC4
7267  013400  001401                    BEQ     INC4
7268  013402  100401                    BMI     .+4
7269  013404  104000            INC4:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7270
7271  013406  000261                    SEC
7272  013410  000324                    SWAB    (R4)+           ;(R4)=000200,CC=1000
7273  013412  103401                    BCS     SWAB2
7274  013414  100401                    BMI     .+4
7275  013416  104000            SWAB2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7276
7277  013420  005425                    NEG     (R5)+           ;(R5)=177600,CC=1001
7278  013422  103001                    BCC     NEG2
7279  013424  100401                    BMI     .+4
7280  013426  104000            NEG2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7281
7282  013430  005044                    CLR     -(R4)           ;(R4)=000000,CC=0100
7283  013432  001401                    BEQ     .+4
7284  013434  104000                    HLT
7285
7286  013436  000261                    SEC
7287  013440  006045                    ROR     -(R5)           ;(R5)=100000,CC=1010
7288  013442  000261                    SEC
7289  013444  005525                    ADC     (R5)+           ;(R5)=100001,CC=1000
7290  013446  102401                    BVS     ADC2
7291  013450  100401                    BMI     .+4
7292  013452  104000            ADC2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7293
7294  013454  000262                    SEV
7295  013456  006224                    ASR     (R4)+           ;(R4)=140000,CC=1001
7296  013460  103002                    BCC     ASR2
7297  013462  102401                    BVS     ASR2
7298  013464  100401                    BMI     .+4
7299  013466  104000            ASR2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7300
7301  013470  000262                    SEV
7302  013472  006144                    ROL     -(R4)           ;(R4)=100001, CC=1001
7303  013474  103002                    BCC     ROL4
7304  013476  102401                    BVS     ROL4
7305  013500  100401                    BMI     .+4
7306  013502  104000            ROL4:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7307
7308  013504  005645                    SBC     -(R5)           ;(R5)=100000,CC=1000
7309  013506  103001                    BCC     .+4
7310  013510  104000                    HLT                     ;ERROR! 'C' BIT FAILED TO CLEAR
7311
7312  013512  005325                    DEC     (R5)+           ;(R5)=077777,CC=0010
7313  013514  103402                    BCS     DEC2
```

```
 7314  013516  102001                        BVC     DEC2
 7315  013520  100001                        BPL     .+4
 7316  013522  104000                DEC2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
 7317
 7318  013524  006324                        ASL     (R4)+           ;(R4)=177776,CC=1010
 7319  013526  102401                        BVS     .+4
 7320  013530  104000                        HLT
 7321  013532  006344                        ASL     -(R4)           ;(R4)=177774,CC=1001
 7322  013534  103003                        BCC     ASL4
 7323  013536  102402                        BVS     ASL4
 7324  013540  001401                        BEQ     ASL4
 7325  013542  100401                        BMI     .+4
 7326  013544  104000                ASL4:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
 7327
 7328  013546  022724  177774              CMP     #177774,(R4)+
 7329  013552  001401                        BEQ     .+4
 7330  013554  104000                        HLT
 7331  013556  020405                        CMP     R4,R5
 7332  013560  001401                        BEQ     .+4
 7333  013562  104000                        HLT
 7334                                ;;*****************************************************************
  (3)                                ;*TEST 10        CHECK UNIARY BYTE OPS USING ADDRESS MODES 2 & 4
  (3)                                ;;*****************************************************************
  (2)
  (2)   013564  000004                TST10:  SCOPE
  (2)   013566  112737  000010  001252        MOVB    #10,@#$TSTNM            ;LOAD TEST NUMBER
  (2)   013574  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 7335  013602  000401                        BR      .+4             ;RESERVE A WORD
 7336  013604  000000                        .WORD   0               ;RESERVED WORD
 7337  013606  010705                        MOV     PC,R5
 7338  013610  162705  000004              SUB     #4,R5           ;R5 POINTS TO EVEN BYTE OF RESERVED WORD
 7339  013614  010500                        MOV     R5,R0
 7340  013616  010002                        MOV     R0,R2
 7341  013620  005202                        INC     R2              ;R2 POINTS TO ODD BYTE OF RESERVED WORD
 7342  013622  005010                        CLR     (R0)            ;PRESET
 7343
 7344  013624  000277                        SCC
 7345  013626  000241                        CLC
 7346  013630  105125                        COMB    (R5)+           ;(R0)=000377,CC=1001
 7347  013632  103002                        BCC     COMB2
 7348  013634  102401                        BVS     COMB2
 7349  013636  100401                        BMI     .+4
 7350  013640  104000                COMB2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
 7351
 7352  013642  105542                        ADCB    -(R2)           ;(R0)=000000,CC=0101
 7353  013644  001401                        BEQ     .+4
 7354  013646  104000                        HLT                     ;ERROR! INCORRECT RESULT AS SHOWN ABOVE
 7355  013650  105525                        ADCB    (R5)+           ;(R0)=000400,CC=0000
 7356  013652  103401                        BCS     ADCB2
 7357  013654  001001                        BNE     .+4
 7358  013656  104000                ADCB2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
 7359
 7360  013660  000263                        +SEC!SEV
 7361  013662  106045                        RORB    -(R5)           ;(R0)=100000,CC=1001
 7362  013664  103003                        BCC     RORB4
 7363  013666  102402                        BVS     RORB4
```

```
7364  013670  001401                    BEQ     RORB4
7365  013672  100401                    BMI     .+4
7366  013674  104000          RORB4:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7367
7368  013676  000277                    SCC
7369  013700  106122                    ROLB    (R2)+           ;(R0)=100001,CC=0000
7370  013702  103403                    BCS     ROLB2
7371  013704  102402                    BVS     ROLB2
7372  013706  001401                    BEQ     ROLB2
7373  013710  100001                    BPL     .+4
7374  013712  104000          ROLB2:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7375
7376  013714  000257                    CCC
7377  013716  106225                    ASRB    (R5)+           ;(R0)=140001, CC=1010
7378  013720  103402                    BCS     ASRB2
7379  013722  102001                    BVC     ASRB2
7380
7381  013724  100401                    BMI     .+4
7382  013726  104000          ASRB2:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7383
7384  013730  105242                    INCB    -(R2)           ;(R0)=140002,CC=0000
7385  013732  000277                    SCC
7386  013734  106222                    ASRB    (R2)+           ;(R0)=140001,CC=0000
7387  013736  103402                    BCS     ASRB2A
7388  013740  102401                    BVS     ASRB2A
7389  013742  100001                    BPL     .+4
7390  013744  104000          ASRB2A:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7391
7392  013746  000266                    +SEZ!SEV                ;SET Z,V
7393  013750  106345                    ASLB    -(R5)           ;(R0)=100001,CC=1001
7394  013752  103003                    BCC     ASLB4
7395  013754  102402                    BVS     ASLB4
7396  013756  001401                    BEQ     ASLB4
7397  013760  100401                    BMI     .+4
7398  013762  104000          ASLB4:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7399
7400  013764  105322                    DECB    (R2)+           ;(R0)=077401=[0774][001] ,CC=0010
7401  013766  103002                    BCC     DECB2
7402  013770  102001                    BVC     DECB2
7403  013772  100001                    BPL     .+4
7404  013774  104000          DECB2:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWNABOVE
7405
7406  013776  105645                    SBCB    -(R5)           ;(R0)=077400, CC=0100
7407  014000  103402                    BCS     SBCB4
7408  014002  102401                    BVS     SBCB4
7409  014004  001401                    BEQ     .+4
7410  014006  104000          SBCB4:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7411
7412  014010  105442                    NEGB    -(R2)           ;(R0)=10400,CC=1001
7413  014012  103002                    BCC     NEGB4
7414  014014  102401                    BVS     NEGB4
7415  014016  100401                    BMI     .+4
7416  014020  104000          NEGB4:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
7417
7418  014022  105725                    TSTB    (R5)+           ;(R0)=100400,CC=0100
7419  014024  103401                    BCS     TSTB2
```

```
 7420  014026  001401                        BEQ     .+4
 7421  014030  104000                TSTB2:  HLT
 7422
 7423  014032  105722                        TSTB    (R2)+           ;(R0)=100400,CC=1000
 7424  014034  001401                        BEQ     TSTB2A
 7425  014036  100401                        BMI     .+4
 7426  014040  104000                TSTB2A: HLT
 7427
 7428  014042  000261                        SEC
 7429  014044  000342                        SWAB    -(R2)           ;(R0)=000201,CC=1000
 7430  014046  103401                        BCS     SWAB4
 7431  014050  100401                        BMI     .+4
 7432  014052  104000                SWAB4:  HLT
 7433
 7434  014054  000277                        SCC
 7435  014056  105225                        INCB    (R5)+           ;(R0)=000601=[0004][201],CC=0000
 7436  014060  103003                        BCC     INCB2
 7437  014062  102402                        BVS     INCB2
 7438  014064  001401                        BEQ     INCB2
 7439  014066  100001                        BPL     .+4
 7440
 7441  014070  104000                INCB2:  HLT
 7442
 7443
 7444
 7445  014072  022227  000601              CMP     (R2)+,#000601   ;CHECK END RESULT
 7446  014076  001401                        BEQ     .+4
 7447  014100  104000                        HLT
 7448  014102  020205                        CMP     R2,R5           ;CHECK REGISTERS
 7449  014104  001401                        BEQ     .+4
 7450  014106  104000                        HLT
 7451                                ;;*****************************************************************
   (3)                               ;*TEST 11       CHECK UNIARY WORD OPS USING ADDRESS MODES 3 & 5
   (3)                               ;;*****************************************************************
   (2)
   (2)  014110  000004                TST11:  SCOPE
   (2)  014112  112737  000011  001252        MOVB    #11,@#$TSTNM            ;LOAD TEST NUMBER
   (2)  014120  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 7452  014126  000402                        BR      .+6             ;RESERVE 2 WORDS
 7453  014130  000000                        .WORD   0               ;1 FOR THE ADDRESS
 7454  014132  000000                        .WORD   0               ;AND 1 FOR DATA
 7455  014134  010703                        MOV     PC,R3
 7456  014136  162703  000004              SUB     #4,R3
 7457  014142  005013                        CLR     (R3)            ;PRESET DATA
 7458  014144  010300                        MOV     R3,R0           ;R0 POINTS TO DATA WORD
 7459  014146  005743                        TST     -(R3)
 7460  014150  010013                        MOV     R0,(R3)
 7461  014152  010304                        MOV     R3,R4
 7462
 7463  014154  000257                        CCC
 7464  014156  005733                        TST     @(R3)+          ;(R0)=000000,CC=0100
 7465  014160  001401                        BEQ     .+4
 7466  014162  104000                        HLT
 7467
 7468  014164  000261                        SEC
 7469  014166  006053                        ROR     @-(R3)          ;(R0)=100000,CC=1010
```

```
7470  014170  103402                    BCS     ROR5
7471  014172  102001                    BVC     ROR5
7472  014174  100401                    BMI     .+4
7473  014176  104000            ROR5:   HLT
7474
7475  014200  000257                    CCC
7476  014202  006234                    ASR     @(R4)+          ;(R0)=140000,CC=1010
7477  014204  102001                    BVC     ASR3
7478  014206  100401                    BMI     .+4
7479  014210  104000            ASR3:   HLT
7480
7481  014212  000250                    CLN
7482  014214  006333                    ASL     @(R3)+          ;(R0)=100000,CC=1001
7483  014216  103002                    BCC     ASL3
7484  014220  102401                    BVS     ASL3
7485  014222  100401                    BMI     .+4
7486  014224  104000            ASL3:   HLT
7487
7488  014226  000277                    SCC
7489  014230  005354                    DEC     @-(R4)          ;(R0)=077777, CC=0010
7490  014232  103003                    BCC     DEC5
7491  014234  102002                    BVC     DEC5
7492  014236  001401                    BEQ     DEC5
7493  014240  100001                    BPL     .+4
7494  014242  104000            DEC5:   HLT
7495
7496  014244  005453                    NEG     @-(R3)          ;(R0)=100001, CC=1001
7497  014246  103002                    BCC     NEG5
7498  014250  102401                    BVS     NEG5
7499  014252  100401                    BMI     .+4
7500  014254  104000            NEG5:   HLT
7501
7502  014256  000262                    SEV
7503  014260  005134                    COM     @(R4)+          ;(R0)=077776, CC=0001
7504  014262  103001                    BCC     COM3
7505  014264  102001                    BVC     .+4
7506  014266  104000            COM3:   HLT
7507
7508  014270  005233                    INC     @(R3)+          ;(R0)=077777, CC=0001
7509  014272  103001                    BCC     INC3
7510  014274  100001                    BPL     .+4
7511  014276  104000            INC3:   HLT
7512
7513  014300  005554                    ADC     @-(R4)          ;(R0)=100000, CC=1010
7514  014302  103402                    BCS     ADC5
7515  014304  102001                    BVC     ADC5
7516  014306  100401                    BMI     .+4
7517  014310  104000            ADC5:   HLT
7518
7519  014312  000257                    CCC
7520  014314  006134                    ROL     @(R4)+          ;(R0)=000000,CC=0111
7521  014316  103002                    BCC     ROL3
7522  014320  102001                    BVC     ROL3
7523  014322  001401                    BEQ     .+4
7524  014324  104000            ROL3:   HLT
7525
```

```
 7526  014326  005253                        INC     @-(R3)          ;(R0)=000001, CC=0001
 7527  014330  005654                        SBC     @-(R4)          ;(R0)=000000, CC=0100
 7528  014332  103401                        BCS     SBC5
 7529  014334  001401                        BEQ     .+4
 7530  014336  104000                SBC5:   HLT
 7531                                ;;*****************************************************************
  (3)                                ;*TEST 12        CHECK UNIARY BYTE OPS USING ADDRESS MODES 3 & 5
  (3)                                ;;*****************************************************************
  (2)
  (2)  014340  000004                TST12:  SCOPE
  (2)  014342  112737  000012  001252        MOVB    #12,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  014350  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 7532  014356  000403                        BR      .+10                    ;RESERVE 3 WORDS
 7533  014360  000000                        .WORD   0                       ;1 FOR EVEN BYTE ADDRESS
 7534  014362  000000                        .WORD   0                       ;1 FOR ODD BYTE ADDRESS
 7535  014364  000000                        .WORD   0                       ;AND 1 FOR DATA
 7536  014366  010702                        MOV     PC,R2
 7537  014370  005742                        TST     -(R2)                   ;BACK R2 UP TO
 7538  014372  005742                        TST     -(R2)                   ;DATA WORD
 7539  014374  010200                        MOV     R2,R0                   ;R0 POINTS TO THE DATA WORD
 7540  014376  005010                        CLR     (R0)                    ;PRESET DATA
 7541  014400  005742                        TST     -(R2)                   ;BACK R2 UP TO
 7542  014402  005742                        TST     -(R2)                   ;EVEN BYTE ADDRESS WORD
 7543  014404  010022                        MOV     R0,(R2)+                ;LOAD ADDRESS
 7544  014406  005200                        INC     R0                      ;ODD BYTE ADDRESS
 7545  014410  010022                        MOV     R0,(R2)+                ;LOAD ODD BYTE ADDRESS
 7546  014412  010200                        MOV     R2,R0                   ;RESET R0
 7547  014414  010205                        MOV     R2,R5
 7548  014416  105152                        COMB    @-(R2)                  ;(R0)=177400,CC=1001
 7549  014420  103001                        BCC     COMB5
 7550  014422  100401                        BMI     .+4
 7551  014424  104000                COMB5:  HLT
 7552  014426  105752                        TSTB    @-(R2)          ;(R0)=177400, CC=0100
 7553  014430  001401                        BEQ     .+4
 7554  014432  104000                        HLT                                     '
 7555  014434  000262                        SEV
 7556  014436  106255                        ASRB    @-(R5)          ;(R0)=177400, CC=1001
 7557  014440  103002                        BCC     ASRB5
 7558  014442  102401                        BVS     ASRB5
 7559  014444  100401                        BMI     .+4
 7560  014446  104000                ASRB5:  HLT
 7561
 7562  014450  105232                        INCB    @(R2)+          ;(R0)=177401, CC=000
 7563  014452  103001                        BCC     INCB3
 7564  014454  100001                        BPL     .+4
 7565  014456  104000                INCB3:  HLT
 7566
 7567  014460  000241                        CLC
 7568  014462  106055                        RORB    @-(R5)          ;(R0)=177400, CC=0111
 7569  014464  103003                        BCC     RORB5
 7570  014466  102002                        BVC     RORB5
 7571  014470  001001                        BNE     RORB5
 7572  014472  100001                        BPL     .+4
 7573  014474  104000                RORB5:  HLT
 7574
 7575  014476  106332                        ASLB    @(R2)+          ;(R0)=177000, CC=1001
```

```
 7576  014500  103002                        BCC     ASLB3
 7577  014502  102401                        BVS     ASLB3
 7578  014504  100401                        BMI     .+4
 7579  014506  104000              ASLB3:    HLT
 7580
 7581  014510  105552                        ADCB    @-(R2)          ;(R0)=177400, CC=1000
 7582  014512  103401                        BCS     ADCB5
 7583  014514  100401                        BMI     .+4
 7584  014516  104000              ADCB5:    HLT
 7585
 7586  014520  000277                        SCC
 7587  014522  106135                        ROLB    @(R5)+          ;(R0)=177401, CC=0000
 7588  014524  101402                        BLOS    ROLB3           ;BRANCH IF C OR Z IS SET
 7589  014526  102401                        BVS     ROLB3
 7590  014530  100001                        BPL     .+4
 7591  014532  104000              ROLB3:    HLT
 7592
 7593  014534  000352                        SWAB    @-(R2)          ;(R0)=000777, CC=1000
 7594  014536  100401                        BMI     .+4
 7595  014540  104000                        HLT
 7596
 7597  014542  000261                        SEC
 7598  014544  105635                        SBCB    @(R5)+          ;(R0)=000377, CC=0100
 7599  014546  103401                        BCS     SBCB3
 7600  014550  001401                        BEQ     .+4
 7601  014552  104000              SBCB3:    HLT
 7602
 7603  014554  105432                        NEGB    @(R2)+          ;(R0)=000001
 7604  014556  105352                        DECB    @-(R2)          ;(R0)=000000, CC=0101
 7605  014560  103001                        BCC     DECB5
 7606  014562  001401                        BEQ     .+4
 7607  014564  104000              DECB5:    HLT
 7608                              ;;*****************************************************************
  (3)                             ;*TEST 13       CHECK UNIARY WORD OPS USING ADDRESS MODE 6 (PC)
  (3)                             ;;*****************************************************************
  (2)
  (2)  014566  000004             TST13:    SCOPE
  (2)  014570  112737  000013  001252        MOVB    #13,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  014576  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 7609  014604  005027                        CLR     (PC)+           ;PRESET DATA = 0
 7610  014606  000000              UWM6:     .WORD   0               ;RESERVED FOR DATA
 7611  014610  010700                        MOV     PC,R0
 7612  014612  024040                        CMP     -(R0),-(R0)     ;R0 POINTS TO DATA WORD
 7613  014614  000277                        SCC
 7614  014616  006167  177764               ROL     UWM6            ;(R0)=000001,CC=0000
 7615  014622  103403                        BCS     ROL6
 7616  014624  102402                        BVS     ROL6
 7617  014626  001401                        BEQ     ROL6
 7618  014630  100001                        BPL     .+4
 7619  014632  104000              ROL6:     HLT
 7620
 7621  014634  005167  177746               COM     UWM6            ;(R0)=177776, CC=1001
 7622  014640  103002                        BCC     COM6
 7623  014642  102401                        BVS     COM6
 7624  014644  100401                        BMI     .+4
 7625  014646  104000              COM6:     HLT
```

```
 7626  014650  006267  177732            ASR     UWM6            ;(R0)=177777, CC=1010
 7627  014654  103402                    BCS     ASR6
 7628  014656  102001                    BVC     ASR6
 7629  014660  100401                    BMI     .+4
 7630  014662  104000            ASR6:   HLT
 7631
 7632  014664  000277                    SCC
 7633  014666  005467  177714            NEG     UWM6            ;(R0)=000001, CC=0001
 7634  014672  103003                    BCC     NEG6
 7635  014674  102402                    BVS     NEG6
 7636  014676  001401                    BEQ     NEG6
 7637  014700  100001                    BPL     .+4
 7638  014702  104000            NEG6:   HLT
 7639
 7640  014704  000277                    SCC
 7641  014706  006067  177674            ROR     UWM6            ;(R0)=100000, CC=1001
 7642  014712  103003                    BCC     ROR6
 7643  014714  102402                    BVS     ROR6
 7644  014716  001401                    BEQ     ROR6
 7645  014720  100401                    BMI     .+4
 7646  014722  104000            ROR6:   HLT
 7647
 7648  014724  005667  177656            SBC     UWM6            ;(R0)=077777, CC=0010
 7649  014730  103402                    BCS     SBC6
 7650  014732  102001                    BVC     SBC6
 7651  014734  100001                    BPL     .+4
 7652  014736  104000            SBC6:   HLT
 7653
 7654  014740  000242                    CLV
 7655  014742  005267  177640            INC     UWM6            ;(R0)=100000, CC=1011
 7656  014746  103403                    BCS     INC6
 7657  014750  102002                    BVC     INC6
 7658  014752  001401                    BEQ     INC6
 7659  014754  100401                    BMI     .+4
 7660  014756  104000            INC6:   HLT
 7661
 7662  014760  006267  177622            ASR     UWM6            ;(R0)=140000, CC=1010
 7663  014764  000261                    SEC
 7664  014766  006367  177614            ASL     UWM6            ;(R0)=100000, CC=1001
 7665  014772  103002                    BCC     ASL6
 7666  014774  102401                    BVS     ASL6
 7667  014776  100401                    BMI     .+4
 7668  015000  104000            ASL6:   HLT
 7669
 7670  015002  005367  177600            DEC     UWM6            ;(R0)=077777, CC=0011
 7671  015006  103002                    BCC     DEC6
 7672  015010  102001                    BVC     DEC6
 7673  015012  100001                    BPL     .+4
 7674  015014  104000            DEC6:   HLT
 7675
 7676  015016  005567  177564            ADC     UWM6            ;(R0)=100000, CC=1010
 7677  015022  103402                    BCS     ADC6
 7678  015024  102001                    BVC     ADC6
 7679  015026  100401                    BMI     .+4
 7680  015030  104000            ADC6:   HLT
 7681  015032  000242                    CLV
```

```
 7682  015034  000367  177546                SWAB    UWM6
 7683  015040  100401                        BMI     .+4
 7684  015042  104000                        HLT
 7685  015044  022710  000200                CMP     #200,(R0)
 7686  015050  001401                        BEQ     .+4
 7687  015052  104000                        HLT
 7688                          ;;*****************************************************************
  (3)                          ;*TEST 14       CHECK UNIARY BYTE OPS (EVEN/ODD) USING ADDRESS MODE 6 (PC)
  (3)                          ;;*****************************************************************
  (2)
  (2)  015054  000004          TST14:  SCOPE
  (2)  015056  112737  000014  001252          MOVB    #14,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  015064  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 7689  015072  012700  015434                MOV     #UBM6,R0
 7690  015076  063700  001604                ADD     @#FACTOR,R0     ;R0 POINTS TO ADDRESS OF DATA
 7691  015102  005067  000326                CLR     UBM6            ;CLEAR DATA
 7692  015106  000277                        SCC
 7693  015110  000244                        CLZ
 7694  015112  105767  000316                TSTB    UBM6
 7695  015116  103403                        BCS     TSTB6
 7696  015120  102402                        BVS     TSTB6
 7697  015122  001001                        BNE     TSTB6
 7698  015124  100001                        BPL     .+4
 7699  015126  104000          TSTB6:  HLT
 7700
 7701  015130  000257                        CCC
 7702  015132  105767  000277                TSTB    UBM6+1          ;TEST ODD BYTE
 7703  015136  001401                        BEQ     .+4
 7704  015140  104000                        HLT
 7705
 7706  015142  105667  000266                SBCB    UBM6            ;(R0)=000000, CC=0100
 7707  015146  103402                        BCS     SBCB6
 7708  015150  102401                        BVS     SBCB6
 7709  015152  001401                        BEQ     .+4
 7710  015154  104000          SBCB6:  HLT
 7711
 7712  015156  000261          1$:     SEC
 7713  015160  105267  000250                INCB    UBM6            ;LOOP UNTIL (R0)=077600, CC=1011
 7714  015164  100403                        BMI     2$
 7715  015166  105567  000243                ADCB    UBM6+1          ;INCB INST INCREMENTS EVEN BYTE
 7716  015172  000771                        BR      1$              ;ADCB INCREMENTS ODD BYTE
 7717  015174  103001          2$:     BCC     INCB6
 7718  015176  102401                        BVS     .+4
 7719  015200  104000          INCB6:  HLT
 7720
 7721  015202  106367  000226                ASLB    UBM6            ;(R0)=077400, CC=0111
 7722  015206  103003                        BCC     ASLB6
 7723  015210  102002                        BVC     ASLB6
 7724  015212  001001                        BNE     ASLB6
 7725  015214  100001                        BPL     .+4
 7726  015216  104000          ASLB6:  HLT
 7727
 7728  015220  000242                        CLV
 7729  015222  105567  000207                ADCB    UBM6+1          ;(R0)=100000, CC=1010
 7730  015226  103402                        BCS     ADCB6
 7731  015230  102001                        BVC     ADCB6
```

```
7732  015232  100401                    BMI     .+4
7733  015234  104000            ADCB6:  HLT
7734
7735  015236  000261                    SEC
7736  015240  106067  000171            RORB    UBM6+1          ;(R0)=140000, CC=1010
7737  015244  103402                    BCS     RORB6
7738  015246  102001                    BVC     RORB6
7739  015250  100401                    BMI     .+4
7740  015252  104000            RORB6:  HLT
7741
7742  015254  105167  000154            COMB    UBM6            ;(R0)=140377 CC=1001
7743  015260  103002                    BCC     COMB6
7744  015262  102401                    BVS     COMB6
7745  015264  100401                    BMI     .+4
7746  015266  104000            COMB6:  HLT
7747
7748  015270  000262                    SEV
7749  015272  105467  000137            NEGB    UBM6+1          ;(R0)=040377, CC=0001
7750  015276  103002                    BCC     NEGB6
7751  015300  102401                    BVS     NEGB6
7752  015302  100001                    BPL     .+4
7753  015304  104000            NEGB6:  HLT
7754
7755  015306  106167  000123            ROLB    UBM6+1          ;(R0)=100777, CC=1010
7756  015312  103402                    BCS     ROLB6
7757  015314  102001                    BVC     ROLB6
7758  015316  100401                    BMI     .+4
7759  015320  104000            ROLB6:  HLT
7760
7761  015322  106267  000106            ASRB    UBM6            ;(R0)=100777, CC=1001
7762  015326  103002                    BCC     ASRB6
7763  015330  102401                    BVS     ASRB6
7764  015332  100401                    BMI     .+4
7765  015334  104000            ASRB6:  HLT
7766
7767  015336  105267  000072            INCB    UBM6            ;(R0)=100400, CC=0101
7768  015342  103002                    BCC     INCB6A
7769  015344  102401                    BVS     INCB6A
7770  015346  001401                    BEQ     .+4
7771  015350  104000            INCB6A: HLT
7772
7773  015352  105367  000057            DECB    UBM6+1          ;(R0)=100000, CC=1001
7774  015356  103003                    BCC     DECB6A
7775  015360  102402                    BVS     DECB6A
7776  015362  001401                    BEQ     DECB6A
7777  015364  100401                    BMI     .+4
7778  015366  104000            DECB6A: HLT
7779
7780  015370  000367  000040            SWAB    UBM6            ;(R0)=000200, CC=1000
7781  015374  103401                    BCS     SWAB6
7782  015376  100401                    BMI     .+4
7783  015400  104000            SWAB6:  HLT
7784
7785  015402  106167  000026            ROLB    UBM6            ;(R0)=000000, CC=0111
7786  015406  103002                    BCC     ROLB6A
7787  015410  102001                    BVC     ROLB6A
```

```
 7788  015412  001401                        BEQ     .+4
 7789  015414  104000              ROLB6A:   HLT
 7790
 7791  015416  005767  000012                TST     UBM6            ;(R0)=000000, CC=0100
 7792  015422  103402                        BCS     TEST6
 7793  015424  102401                        BVS     TEST6
 7794  015426  001401                        BEQ     .+4
 7795  015430  104000              TEST6:    HLT
 7796
 7797  015432  000401                        BR      .+4             ;RESERVE A WORD
 7798  015434  000000              UBM6:     .WORD   0               ;WORD RESERVED FOR DATA
 7799  015436  000004              RELE1:    SCOPE
   (1) 015440  010702                        MOV     PC,R2
   (1) 015442  062702  000012                ADD     #12,R2
   (1) 015446  012707  044042                MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
   (1) 015452  000000              REL11:    .WORD   0
   (1)                             ;1111111111111 LAST ADDRESS OF CODE TO BE RELOCATED 11111111111
   (1)
 7800                              ;;*****************************************************************
   (3)                             ;*TEST 15       CHECK UNIARY WORD OPS USING ADDRESS MODE 7
   (3)                             ;;*****************************************************************
   (2)
   (2)  015454  012767  000001  163714       MOV     #1,$TIMES       ;;DO 1 ITERATION
   (2)  015462  000004              TST15:    SCOPE
   (2)  015464  112737  000015  001252       MOVB    #15,@#$TSTNM            ;LOAD TEST NUMBER
   (2)  015472  013737  001252  177570       MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 7801
   (1)                                       .SBTTL  START OF SECTION 2
   (1)                             ;2222222222222 FIRST ADDRESS TO BE RELOCATED 222222222
   (1)  015500  010700              REL2:     MOV     PC,R0           ;GET PC
   (1)  015502  005740                        TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL2
   (1)  015504  010037  001610                MOV     R0,@#FRSTAD     ;SAVE
   (1)  015510  010700                        MOV     PC,R0           ;GET CURRENT PC
   (1)  015512  162700  015512                SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
   (1)  015516  010037  001604                MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
   (1)  015522  010737  001262                MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
   (1)  015526  062737  000030  001262        ADD     #30,@#$LPERR    ;ADJUST
   (1)  015534  013737  001262  001260        MOV     @#$LPERR,@#$LPADR
   (1)  015542  105737  001600                TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
   (1)  015546  001402                        BEQ     .+6
   (1)  015550  000167  004170                JMP     RELE2
 7802  015554  000403                        BR      UW7             ;RESERVE 3 WORDS FOR ADDRESSES & DATA
 7803  015556  000000                        .WORD   0               ;CONTAINS ADDRESS OF UWM7
 7804  015560  000000              UWM7:     .WORD   0               ;CONTAINS DATA
 7805  015562  000000                        .WORD   0               ;CONTAINS ADDRESS OF UWM7
 7806
 7807  015564  010700              UW7:      MOV     PC,R0
 7808  015566  005740                        TST     -(R0)
 7809  015570  005740                        TST     -(R0)
 7810  015572  005040                        CLR     -(R0)           ;CLEAR TEST DATA
 7811  015574  010002                        MOV     R0,R2
 7812  015576  010240                        MOV     R2,-(R0)        ;SET UP ADDRESS
 7813  015600  005720                        TST     (R0)+           ;MOVE R0 TO NEXT ADDRESS
 7814  015602  005720                        TST     (R0)+
 7815  015604  010210                        MOV     R2,(R0)         ;SET NEXT ADDRESS
 7816  015606  010200                        MOV     R2,R0           ;SET R0 POINTING TO DATA
```

```
7817  015610  000277                        SCC
7818  015612  000244                        CLZ
7819  015614  005772  000002                TST     @2(2)           ;(R0)=000000, CC=0100
7820  015620  001401                        BEQ     .+4
7821  015622  104000                        HLT
7822
7823  015624  000277                        SCC
7824  015626  005672  177776                SBC     @-2(2)          ;(R0)=177777, CC=1001
7825  015632  103002                        BCC     SBC7
7826  015634  102401                        BVS     SBC7
7827  015636  100401                        BMI     .+4
7828  015640  104000              SBC7:     HLT
7829
7830  015642  000277                        SCC
7831  015644  000241                        CLC
7832  015646  006372  000002                ASL     @2(2)           ;(R0)=177776, CC=1001
7833  015652  103002                        BCC     ASL7
7834  015654  102401                        BVS     ASL7
7835  015656  100401                        BMI     .+4
7836  015660  104000              ASL7:     HLT
7837
7838  015662  000257                        CCC
7839  015664  005372  000002                DEC     @2(2)           ;(R0)=177775, CC=1000
7840  015670  103402                        BCS     DEC7
7841  015672  102401                        BVS     DEC7
7842  015674  100401                        BMI     .+4
7843  015676  104000              DEC7:     HLT
7844
7845  015700  000262                        SEV
7846  015702  006272  177776                ASR     @-2(2)          ;(R0)=177776, CC=1001
7847  015706  103002                        BCC     ASR7
7848  015710  102401                        BVS     ASR7
7849  015712  100401                        BMI     .+4
7850  015714  104000              ASR7:     HLT
7851
7852  015716  000241                        CLC
7853  015720  000262                        SEV
7854  015722  006072  177776                ROR     @-2(2)          ;(R0)=077777, CC=0000
7855  015726  101402                        BLOS    ROR7            ;BRANCH IF C OR Z IS SET
7856  015730  102401                        BVS     ROR7
7857  015732  100001                        BPL     .+4
7858  015734  104000              ROR7:     HLT
7859
7860  015736  000262                        SEV
7861  015740  005472  000002                NEG     @2(2)           ;(R0)=100001, CC=1001
7862  015744  103002                        BCC     NEG7
7863  015746  102401                        BVS     NEG7
7864  015750  100401                        BMI     .+4
7865  015752  104000              NEG7:     HLT
7866
7867  015754  000250                        CLN
7868  015756  000372  177776                SWAB    @-2(2)          ;(R0)=000600, CC=1000
7869  015762  103401                        BCS     SWAB7
7870  015764  100401                        BMI     .+4
7871  015766  104000              SWAB7:    HLT
7872
```

```
 7873  015770  000262                          SEV
 7874  015772  005172  000002                  COM     @2(2)           ;(R0)=177177, CC=1001
 7875  015776  103002                          BCC     COM7
 7876  016000  102401                          BVS     COM7
 7877  016002  100401                          BMI     .+4
 7878  016004  104000                  COM7:   HLT
 7879
 7880  016006  000372  000002                  SWAB    @2(2)           ;(R0)=077776, CC=1000
 7881  016012  100401                          BMI     .+4
 7882  016014  104000                          HLT
 7883
 7884  016016  000277                          SCC
 7885  016020  005572  177776                  ADC     @-2(2)          ;(R0)=077777, CC=0000
 7886  016024  103402                          BCS     ADC7
 7887  016026  102401                          BVS     ADC7
 7888  016030  100001                          BPL     .+4
 7889  016032  104000                  ADC7:   HLT
 7890
 7891  016034  005272  000002                  INC     @2(2)           ;(R0)=100000, CC=1010
 7892  016040  102001                          BVC     INC7
 7893  016042  100401                          BMI     .+4
 7894  016044  104000                  INC7:   HLT
 7895
 7896  016046  000257                          CCC
 7897  016050  006172  177776                  ROL     @-2(2)          ;(R0)=000000, CC=0111
 7898  016054  103002                          BCC     ROL7
 7899  016056  102001                          BVC     ROL7
 7900  016060  001401                          BEQ     .+4
 7901  016062  104000                  ROL7:   HLT
 7902                                  ;;*******************************************************************
   (3)                                 ;*TEST 16       CHECK UNIARY BYTE OPS USING ADDRESS MODE 7
   (3)                                 ;;*******************************************************************
   (2)
   (2)  016064  000004                  TST16:  SCOPE
   (2)  016066  112737  000016  001252          MOVB    #16,@#$TSTNM            ;LOAD TEST NUMBER
   (2)  016074  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 7903  016102  012700  015560                  MOV     #UWM7,R0
 7904  016106  063700  001604                  ADD     @#FACTOR,R0
 7905  016112  010002                          MOV     R0,R2
 7906  016114  010067  177442                  MOV     R0,UWM7+2
 7907  016120  005720                          TST     (R0)+
 7908  016122  005210                          INC     (R0)            ;WORD FOLLOWING UWM7 CONTAINS ADDRESS
 7909  016124  005740                          TST     -(R0)           ;OF ODD BYTE, R0 POINTS TO DATA WORD
 7910  016126  005010                          CLR     (R0)            ;PRESET DATA
 7911  016130  010067  177422                  MOV     R0,UWM7-2
 7912                                  ;NOTE:  @2(2) REFERENCES THE ODD BYTE, AND @-2(2) REFERENCES THE EVEN BYTE.
 7913
 7914  016134  000263                          +SEC!SEV                        ;SET C AND V
 7915  016136  105672  000002                  SBCB    @2(2)           ;(R0)=177400, CC=1001
 7916  016142  103003                          BCC     SBCB7
 7917  016144  102402                          BVS     SBCB7
 7918  016146  001401                          BEQ     SBCB7
 7919  016150  100401                          BMI     .+4
 7920  016152  104000                  SBCB7:  HLT
 7921
 7922  016154  000277                          SCC                             ;SET CONDITION CODES
```

```
 7923  016156  105572  177776            ADCB    @-2(2)          ;(R0)=177401, CC=0000
 7924  016162  103403                    BCS     ADCB7
 7925  016164  102402                    BVS     ADCB7
 7926  016166  001401                    BEQ     ADCB7
 7927  016170  100001                    BPL     .+4
 7928  016172  104000            ADCB7:  HLT
 7929
 7930  016174  105172  177776            COMB    @-2(2)          ;(R0)=177776, CC=1001
 7931  016200  103002                    BCC     COMB7
 7932  016202  102401                    BVS     COMB7
 7933  016204  100401                    BMI     .+4
 7934  016206  104000            COMB7:  HLT
 7935
 7936  016210  000241                    CLC                     ;CLEAR CARRY
 7937  016212  106072  000002            RORB    @2(2)           ;(R0)=077776, CC=0011
 7938  016216  103002                    BCC     RORB7
 7939  016220  102001                    BVC     RORB7
 7940  016222  100001                    BPL     .+4
 7941  016224  104000            RORB7:  HLT
 7942
 7943  016226  105272  000002            INCB    @2(2)           ;(R0)=100376, CC=1011
 7944  016232  103002                    BCC     INCB7
 7945  016234  102001                    BVC     INCB7
 7946  016236  100401                    BMI     .+4
 7947  016240  104000            INCB7:  HLT
 7948
 7949  016242  105372  177776            DECB    @-2(2)          ;(R0)=100375, CC=1001
 7950  016246  103002                    BCC     DECB7
 7951  016250  102401                    BVS     DECB7
 7952  016252  100401                    BMI     .+4
 7953  016254  104000            DECB7:  HLT
 7954
 7955  016256  106372  000002            ASLB    @2(2)           ;(R0)=000375, CC=0111
 7956  016262  103002                    BCC     ASLB7
 7957  016264  102001                    BVC     ASLB7
 7958  016266  001401                    BEQ     .+4
 7959  016270  104000            ASLB7:  HLT
 7960
 7961  016272  000241                    CLC                     ;CLEAR CARRY
 7962  016274  106272  177776            ASRB    @-2(2)          ;(R0)=000376, CC=1001
 7963  016300  103002                    BCC     ASRB7
 7964  016302  102401                    BVS     ASRB7
 7965  016304  100401                    BMI     .+4
 7966  016306  104000            ASRB7:  HLT
 7967
 7968  016310  105472  000002            NEGB    @2(2)           ;(R0)=000376, CC=0100
 7969  016314  103402                    BCS     NEGB7
 7970  016316  102401                    BVS     NEGB7
 7971  016320  001401                    BEQ     .+4
 7972  016322  104000            NEGB7:  HLT
 7973
 7974  016324  000262                    SEV
 7975  016326  106172  177776            ROLB    @-2(2)          ;(R0)=00374, CC=1001
 7976  016332  103002                    BCC     ROLB7
 7977  016334  102401                    BVS     ROLB7
 7978  016336  100401                    BMI     .+4
```

```
 7979  016340  104000                  ROLB7:  HLT
 7980
 7981  016342  105272  177776                  INCB    @-2(2)          ;(R0)=000375, CC=1001
 7982  016346  105272  177776                  INCB    @-2(2)          ;(R0)=000376, CC=1001
 7983  016352  105572  177776                  ADCB    @-2(2)          ;(R0)=000377, CC=1000
 7984  016356  105172  177776                  COMB    @-2(2)          ;(R0)=000000, CC=0100
 7985  016362  001401                          BEQ     .+4
 7986  016364  104000                          HLT
 7987                                  ;;*****************************************************************
  (3)                                  ;*TEST 17        CHECK BINARY OPS USING ADDRESS MODE 0
  (3)                                  ;;*****************************************************************
  (2)
  (2)  016366  000004                  TST17:  SCOPE
  (2)  016370  112737  000017  001252          MOVB    #17,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  016376  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 7988  016404  000277                          SCC                             ;SET CONDITION CODES
 7989  016406  010700                          MOV     PC,R0                   ;R0=PC, CC=X001
 7990  016410  103002                          BCC     MOV0
 7991  016412  102401                          BVS     MOV0
 7992  016414  001001                          BNE     .+4
 7993  016416  104000                  MOV0:   HLT
 7994
 7995  016420  010002                          MOV     R0,R2                   ;R2=R0
 7996  016422  000262                          SEV                             ;SET V
 7997  016424  160002                          SUB     R0,R2                   ;R2=000000, CC=0100
 7998  016426  103402                          BCS     SUB0
 7999  016430  102401                          BVS     SUB0
 8000  016432  001401                          BEQ     .+4
 8001  016434  104000                  SUB0:   HLT
 8002
 8003  016436  000244                          CLZ
 8004  016440  010203                          MOV     R2,R3                   ;R2=R3=000000, CC=0100
 8005  016442  103401                          BCS     MOV0A
 8006  016444  001401                          BEQ     .+4
 8007  016446  104000                  MOV0A:  HLT
 8008
 8009  016450  000257                          CCC
 8010  016452  000272                          +SEV!SEN                        ;SET V & N
 8011  016454  020203                          CMP     R2,R3                   ;R2=R3=000000, CC=0100
 8012  016456  103403                          BCS     CMP0
 8013  016460  102402                          BVS     CMP0
 8014  016462  001001                          BNE     CMP0
 8015  016464  100001                          BPL     .+4
 8016  016466  104000                  CMP0:   HLT
 8017
 8018  016470  010002                          MOV     R0,R2                   ;R0=R2
 8019  016472  010203                          MOV     R2,R3                   ;R0=R2=R3
 8020  016474  060203                          ADD     R2,R3                   ;R3=2*R0
 8021  016476  006302                          ASL     R2                      ;R2=2*R0
 8022  016500  020203                          CMP     R2,R3                   ;R2=R3=2*R0
 8023  016502  001401                          BEQ     .+4
 8024  016504  104000                          HLT                             ;ERROR! CHECK ADD INSTRUCTION
 8025
 8026                                  ;THE FOLLOWING SUBTEST SHIFTS A BIT THROUGH R2 AND R5 AND DOES A
 8027                                  ;BIT TEST (BIT) USING R2 AND R5.
 8028  016506  005002                          CLR     R2
```

```
 8029  016510  005202                    INC     R2              ;R2=1
 8030  016512  000402                    BR      2$
 8031  016514  006302            1$:     ASL     R2
 8032  016516  100407                    BMI     4$
 8033  016520  010205            2$:     MOV     R2,R5
 8034  016522  000277                    SCC                     ;SET CC'S
 8035  016524  030205                    BIT     R2,R5           ;R2=R5, CC=X001
 8036  016526  103002                    BCC     3$
 8037  016530  102401                    BVS     3$
 8038  016532  001370                    BNE     1$
 8039  016534  104000            3$:     HLT
 8040  016536  010205            4$:     MOV     R2,R5           ;R2 AND R5=100000(OCTAL)
 8041  016540  000257                    CCC                     ;CLEAR CC'S
 8042  016542  030205                    BIT     R2,R5           ;R2=R5, CC=1000
 8043  016544  100401                    BMI     .+4
 8044  016546  104000                    HLT
 8045
 8046  016550  005002                    CLR     R2
 8047  016552  000277                    SCC                     ;SET CC'S
 8048  016554  050002                    BIS     R0,R2           ;R0=PC (NON-ZERIO DATA), CC=X001
 8049  016556  103002                    BCC     BIS0
 8050  016560  102401                    BVS     BIS0
 8051  016562  001001                    BNE     .+4
 8052  016564  104000            BIS0:   HLT
 8053
 8054  016566  010003                    MOV     R0,R3
 8055  016570  000277                    SCC                     ;CC=1111
 8056  016572  000244                    CLZ
 8057  016574  040003                    BIC     R0,R3           ;R0=R3, CC=0101
 8058  016576  103003                    BCC     BIC0
 8059  016600  102402                    BVS     BIC0
 8060  016602  001001                    BNE     BIC0
 8061  016604  100001                    BPL     .+4
 8062  016606  104000            BIC0:   HLT
 8063
 8064  016610  010004                    MOV     R0,R4
 8065  016612  005104                    COM     R4
 8066  016614  040004                    BIC     R0,R4           ;R0=COMPLEMENT OF R4, R4 REMAINS UNCHANGED
 8067  016616  005104                    COM     R4
 8068  016620  020004                    CMP     R0,R4           ;R0=R4
 8069  016622  001401                    BEQ     .+4
 8070  016624  104000                    HLT
 8071
 8072  016626  010004                    MOV     R0,R4
 8073  016630  005104                    COM     R4
 8074  016632  010403                    MOV     R4,R3           ;R3=R4
 8075  016634  050003                    BIS     R0,R3           ;R3=COMPLEMENT OF R0, CC=1001
 8076  016636  103001                    BCC     BIS0A
 8077  016640  100401                    BMI     .+4
 8078  016642  104000            BIS0A:  HLT
 8079  016644  005203                    INC     R3              ;R3=0 AFTER INC
 8080  016646  001401                    BEQ     .+4
 8081  016650  104000                    HLT
 8082  016652  010304                    MOV     R3,R4           ;R3=R4=0
 8083  016654  005103                    COM     R3              ;R3=177777
 8084  016656  000261                    SEC                     ;SET C
```

```
 8085  016660  006004                        ROR     R4              ;R4=100000
 8086  016662  060304                        ADD     R3,R4           ;R3=177777,R4=077777, CC=0011
 8087  016664  103003                        BCC     ADD0
 8088  016666  102002                        BVC     ADD0
 8089  016670  001401                        BEQ     ADD0
 8090  016672  100001                        BPL     .+4
 8091  016674  104000                ADD0:   HLT
 8092  016676  010700                        MOV     PC,R0           ;R0=PC
 8093  016700  022020                        CMP     (R0)+,(R0)+     ;R0=R0+4
 8094  016702  020007                        CMP     R0,PC           ;PC=PC+4=R0
 8095  016704  001401                        BEQ     .+4
 8096  016706  104000                        HLT
 8097
 8098  016710  010700                        MOV     PC,R0           ;R0=PC
 8099  016712  062700  000010                ADD     #10,R0          ;R0=PC+10(8)
 8100  016716  010002                        MOV     R0,R2           ;R2=R0
 8101  016720  020700                        CMP     PC,R0           ;R0=PC
 8102  016722  001002                        BNE     CMP0A
 8103  016724  020200                        CMP     R2,R0           ;R2=R0
 8104  016726  001401                        BEQ     .+4
 8105  016730  104000                CMP0A:  HLT
 8106                                  ;;**********************************************************************
  (3)                                  ;*TEST 20        CHECK BINARY OPS USING ADDRESS MODE 1
  (3)                                  ;;**********************************************************************
  (2)
  (2)  016732  000004                TST20:  SCOPE
  (2)  016734  112737  000020  001252        MOVB    #20,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  016742  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 8107  016750  000402                        BR      .+6             ;RESERVE TWO WORDS
 8108  016752  000000                        .WORD   0               ;RESERVED FOR SOURCE DATA
 8109  016754  000000                        .WORD   0               ;RESERVED FOR DESTINATION DATA
 8110  016756  010704                        MOV     PC,R4
 8111  016760  005744                        TST     -(R4)
 8112  016762  005044                        CLR     -(R4)           ;R4 POINTS TO DESTINATION DATA
 8113  016764  010403                        MOV     R4,R3
 8114  016766  005043                        CLR     -(R3)           ;R3 POINTS TO SOUCE DATA
 8115
 8116  016770  005113                        COM     (R3)            ;(R3)=177777
 8117  016772  005214                        INC     (R4)            ;(R4)=000001
 8118  016774  000262                        SEV                     ;SET V
 8119  016776  061314                        ADD     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=0101
 8120  017000  103002                        BCC     ADD1
 8121  017002  102401                        BVS     ADD1
 8122  017004  001401                        BEQ     .+4
 8123  017006  104000                ADD1:   HLT
 8124
 8125  017010  000277                        SCC
 8126  017012  000250                        CLN
 8127  017014  021314                        CMP     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=1000
 8128  017016  103403                        BCS     CMP1
 8129  017020  102402                        BVS     CMP1
 8130  017022  001401                        BEQ     CMP1
 8131  017024  100401                        BMI     .+4
 8132  017026  104000                CMP1:   HLT
 8133
 8134  017030  000277                        SCC
```

```
 8135  017032  000244                        CLZ
 8136  017034  031314                        BIT     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=0101
 8137  017036  103002                        BCC     BITT1
 8138  017040  102401                        BVS     BITT1
 8139  017042  001401                        BEQ     .+4
 8140  017044  104000              BITT1:    HLT
 8141
 8142  017046  000277                        SCC
 8143  017050  000245                        +CLC!CLZ
 8144  017052  005114                        COM     (R4)            ;(R4)=177777
 8145  017054  161314                        SUB     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=0100
 8146  017056  103402                        BCS     SUB1
 8147  017060  102401                        BVS     SUB1
 8148  017062  001401                        BEQ     .+4
 8149  017064  104000              SUB1:     HLT
 8150
 8151  017066  105013                        CLRB    (R3)            ;(R3)=177400
 8152  017070  000313                        SWAB    (R3)            ;(R3)=000377
 8153  017072  000270                        SEN
 8154  017074  011314                        MOV     (R3),(R4)       ;(R3)=(R4)=000377
 8155  017076  100001                        BPL     .+4
 8156  017100  104000                        HLT
 8157  017102  000314                        SWAB    (R4)            ;(R3)=000377,(R4)=177400
 8158  017104  000263                        +SEC!SEV                ;SET C & V
 8159  017106  051314                        BIS     (R3),(R4)       ;(R3)=000377,(R4)=177777, CC=1001
 8160  017110  103002                        BCC     BIS1
 8161  017112  102401                        BVS     BIS1
 8162  017114  100401                        BMI     .+4
 8163  017116  104000              BIS1:     HLT
 8164
 8165  017120  041314                        BIC     (R3),(R4)       ;(R3)=000377,(R4)=177400, CC=1001
 8166  017122  103002                        BCC     BIC1
 8167  017124  102401                        BVS     BIC1
 8168  017126  100401                        BMI     .+4
 8169  017130  104000              BIC1:     HLT
 8170
 8171  017132  000262                        SEV                     ;SET V
 8172  017134  021314                        CMP     (R3),(R4)       ;(R3)=000377,(R4)=177400, CC=0001
 8173  017136  103003                        BCC     CMP1A
 8174  017140  102402                        BVS     CMP1A
 8175  017142  001401                        BEQ     CMP1A
 8176  017144  100001                        BPL     .+4
 8177  017146  104000              CMP1A:    HLT
 8178
 8179  017150  005013                        CLR     (R3)            ;(R3)=000000
 8180  017152  000261                        SEC
 8181  017154  006013                        ROR     (R3)            ;(R3)=100000
 8182  017156  011314                        MOV     (R3),(R4)       ;(R3)=(R4)=100000
 8183  017160  005114                        COM     (R4)            ;(R4)=077777
 8184  017162  161314                        SUB     (R3),(R4)       ;(R3)=100000,(R4)=177777, CC=1011
 8185  017164  103002                        BCC     SUB1A
 8186  017166  102001                        BVC     SUB1A
 8187  017170  100401                        BMI     .+4
 8188  017172  104000              SUB1A:    HLT
 8189
 8190  017174  000277                        SCC
```

```
 8191  017176  161314                        SUB     (R3),(R4)       ;(R3)=100000,(R4)=077777, CC=0000
 8192  017200  101402                        BLOS    SUB1B           ;BRANCH IF C OR Z IS SET
 8193  017202  102401                        BVS     SUB1B
 8194  017204  100001                        BPL     .+4
 8195  017206  104000                SUB1B:  HLT
 8196
 8197  017210  011314                        MOV     (R3),(R4)       ;(R3)=100000,(R4)=100000, CC=1000
 8198  017212  001401                        BEQ     MOV1
 8199  017214  100401                        BMI     .+4
 8200  017216  104000                MOV1:   HLT
 8201
 8202  017220  061314                        ADD     (R3),(R4)       ;(R3)=100000,(R4)=000000, CC=0111
 8203  017222  103003                        BCC     ADD1A
 8204  017224  102002                        BVC     ADD1A
 8205  017226  001001                        BNE     ADD1A
 8206  017230  100001                        BPL     .+4
 8207  017232  104000                ADD1A:  HLT
 8208
 8209  017234  005113                        COM     (R3)            ;(R3)=077777
 8210  017236  011314                        MOV     (R3),(R4)       ;(R4)=077777
 8211  017240  061314                        ADD     (R3),(R4)       ;(R3)=077777,(R4)=177776, CC=1010
 8212  017242  103402                        BCS     ADD1B
 8213  017244  102001                        BVC     ADD1B
 8214  017246  100401                        BMI     .+4
 8215  017250  104000                ADD1B:  HLT
 8216
 8217
 8218  017252  062714  000002                ADD     #2,(R4)
 8219  017256  005714                        TST     (R4)            ;CHECK FINAL RESULT
 8220  017260  001401                        BEQ     .+4
 8221  017262  104000                        HLT
 8222                                ;;*****************************************************************
   (3)                               ;*TEST 21       CHECK BINARY BYTE OPS USING ADDRESS MODE 1
   (3)                               ;;*****************************************************************
   (2)
   (2)  017264  000004          TST21:  SCOPE
   (2)  017266  112737  000021  001252  MOVB    #21,@#STSTNM            ;LOAD TEST NUMBER
   (2)  017274  013737  001252  177570  MOV     @#STSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 8223  017302  000402                    BR      .+6
 8224  017304  000000                    .WORD   0
 8225  017306  000000                    .WORD   0
 8226  017310  010705                    MOV     PC,R5
 8227  017312  005745                    TST     -(R5)
 8228  017314  005045                    CLR     -(R5)           ;(R5)=000000
 8229  017316  010502                    MOV     R5,R2
 8230  017320  005042                    CLR     -(R2)           ;(R2)=000000
 8231  017322  005202                    INC     R2              ;R2 POINTS TO ODD BYTE
 8232  017324  105112                    COMB    (R2)            ;(R2)=177400
 8233
 8234  017326  000277                    SCC
 8235  017330  111215                    MOVB    (R2),(R5)       ;(R2)=177400,(R5)=000377,CC=1001
 8236  017332  103005                    BCC     MOVB1
 8237  017334  102404                    BVS     MOVB1
 8238  017336  001403                    BEQ     MOVB1
 8239  017340  100002                    BPL     MOVB1
 8240  017342  105215                    INCB    (R5)            ;CHECK RESULT
```

```
 8241  017344  001401                    BEQ     .+4
 8242  017346  104000            MOVB1:  HLT
 8243
 8244  017350  106312                    ASLB    (R2)            ;SHIFT (R2) UNTIL
 8245  017352  102376                    BVC     .-2             ;(R2)=000000
 8246  017354  106012                    RORB    (R2)            ;(R2)=100000
 8247  017356  105315                    DECB    (R5)            ;(R5)=00377
 8248  017360  106015                    RORB    (R5)            ;(R5)=000177
 8249  017362  000257                    CCC
 8250  017364  121512                    CMPB    (R5),(R2)       ;(R5)=000177,(R2)=100000, CC=1010
 8251  017366  102001                    BVC     CMPB1
 8252  017370  100401                    BMI     .+4
 8253  017372  104000            CMPB1:  HLT
 8254
 8255  017374  005003                    CLR     R3
 8256  017376  000261                    SEC
 8257  017400  006003                    ROR     R3              ;R3=100000
 8258  017402  050315                    BIS     R3,(R5)         ;(R5)=100177
 8259  017404  000273                    +SEC!SEV!SEN            ;SET C,V, & N
 8260  017406  131215                    BITB    (R2),(R5)       ;(R2)=100000,(R5)=100177, CC=0101
 8261  017410  103002                    BCC     BITB1
 8262  017412  102401                    BVS     BITB1
 8263  017414  001401                    BEQ     .+4
 8264  017416  104000            BITB1:  HLT
 8265
 8266  017420  151215                    BISB    (R2),(R5)       ;(R2)=100000,(R5)=100377, CC=1001
 8267  017422  103001                    BCC     BISB1
 8268  017424  100401                    BMI     .+4
 8269  017426  104000            BISB1:  HLT
 8270
 8271  017430  141215                    BICB    (R2),(R5)       ;(R2)=100000,(R5)=100177, CC=0001
 8272  017432  103002                    BCC     BICB1
 8273  017434  001401                    BEQ     BICB1
 8274  017436  100001                    BPL     .+4
 8275  017440  104000            BICB1:  HLT
 8276
 8277  017442  105112                    COMB    (R2)            ;(R2)=077400,(R5)=100177
 8278  017444  121215                    CMPB    (R2),(R5)
 8279  017446  001401                    BEQ     .+4
 8280  017450  104000                    HLT
 8281
 8282  017452  141512                    BICB    (R5),(R2)       ;(R5)=100177,(R2)=000000, CC=0100
 8283  017454  001002                    BNE     BICB1A
 8284  017456  105712                    TSTB    (R2)
 8285  017460  001401                    BEQ     .+4
 8286  017462  104000            BICB1A: HLT
 8287
 8288  017464  000402                    BR      .+6             ;RESERVE TWO WORDS FOR DATA
 8289  017466  000000                    .WORD   0               ;SOURCE DATA
 8290  017470  000000                    .WORD   0               ;DEST DATA
 8291  017472  010705                    MOV     PC,R5
 8292  017474  005745                    TST     -(R5)
 8293  017476  105045                    CLRB    -(R5)           ;R5 POINTS TO DEST ODD BYTE
 8294  017500  010504                    MOV     R5,R4
 8295  017502  105044                    CLRB    -(R4)           ;R4 POINTS TO DEST EVEN BYTE
 8296  017504  010403                    MOV     R4,R3
```

```
 8297  017506  105043                        CLRB    -(R3)           ;R3 POINTS TO SOURCE ODD BYTE
 8298  017510  010302                        MOV     R3,R2
 8299  017512  105042                        CLRB    -(R2)           ;R2 POINTS TO SOURCE EVEN BYTE
 8300
 8301                                  ;COMMENTS ARE LEAST SIGNIFICANT 4 BITS OF BYTES POINTED TO BY R2,R3
 8302                                  ;R4, AND R5 RESPECTIVELY AND THE REMAINING BITS ARE 0'S.
 8303  017514  000261                        SEC                     ;SET CARRY
 8304                                                                ;(R2),(R3),(R4),(R5)
 8305  017516  106112                        ROLB    (R2)            ;0001,0000,0000,0000
 8306  017520  111214                        MOVB    (R2),(R4)       ;0001,0000,0001,0000
 8307  017522  106112                        ROLB    (R2)            ;0010,0000,0001,0000
 8308  017524  111213                        MOVB    (R2),(R3)       ;0010,0010,0001,0000
 8309  017526  106112                        ROLB    (R2)            ;0100,0010,0001,0000
 8310  017530  111315                        MOVB    (R3),(R5)       ;0100,0010,0001,0010
 8311  017532  106112                        ROLB    (R2)            ;1000,0010,0001,0010
 8312  017534  106113                        ROLB    (R3)            ;1000,0100,0001,0010
 8313  017536  151215                        BISB    (R2),(R5)       ;1000,0100,0001,1010
 8314  017540  131512                        BITB    (R5),(R2)       ;1000,0100,0001,1010
 8315  017542  001426                        BEQ     BIN1
 8316  017544  151314                        BISB    (R3),(R4)       ;1000,0100,0101,1010
 8317  017546  131413                        BITB    (R4),(R3)       ;1000,0100,0101,1010
 8318  017550  001423                        BEQ     BIN1
 8319  017552  105213                        INCB    (R3)            ;1000,0101,0101,1010
 8320  017554  121314                        CMPB    (R3),(R4)       ;1000,0101,0101,1010
 8321  017556  001020                        BNE     BIN1
 8322  017560  106113                        ROLB    (R3)            ;1000,1010,0101,1010
 8323  017562  121315                        CMPB    (R3),(R5)       ;1000,1010,0101,1010
 8324  017564  001015                        BNE     BIN1
 8325  017566  106212                        ASRB    (R2)            ;0100,1010,0101,1010
 8326  017570  131214                        BITB    (R2),(R4)       ;0100,1010,0101,1010
 8327  017572  001412                        BEQ     BIN1
 8328  017574  106015                        RORB    (R5)            ;0100,1010,0101,0101
 8329  017576  121415                        CMPB    (R4),(R5)       ;0100,1010,0101,0101
 8330  017600  001007                        BNE     BIN1
 8331  017602  105314                        DECB    (R4)            ;0100,1010,0100,0101
 8332  017604  141214                        BICB    (R2),(R4)       ;0100,1010,0000,0101
 8333  017606  001004                        BNE     BIN1
 8334  017610  111314                        MOVB    (R3),(R4)       ;0100,1010,1010,0101
 8335  017612  106213                        ASRB    (R3)            ;0100,0101,1010,0101
 8336  017614  141315                        BICB    (R3),(R5)       ;0100,0101,1010,0101
 8337  017616  001401                        BEQ     .+4
 8338  017620  104000                  BIN1: HLT
 8339                                  ;;=====================================================================
  (3)                                  ;*TEST 22       CHECK BINARY WORD OPS USING ADDRESS MODE 2 & 4
  (3)                                  ;;=====================================================================
  (2)
  (2)   017622  000004                  TST22:  SCOPE
  (2)   017624  112737  000022  001252        MOVB    #22,@#$TSTNM            ;LOAD TEST NUMBER
  (2)   017632  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 8340   017640  012704  017470                MOV     #BICB1A+6,R4
 8341   017644  012702  017466                MOV     #BICB1A+4,R2
 8342   017650  063702  001604                ADD     @#FACTOR,R2
 8343   017654  063704  001604                ADD     @#FACTOR,R4
 8344   017660  010405                        MOV     R4,R5           ;SET DESTINATION REGISTER
 8345   017662  012715  000001                MOV     #1,(R5)
 8346   017666  012712  177777                MOV     #-1,(R2)
```

```
 8347  017672  000257                        CCC
 8348  017674  000262                        SEV
 8349  017676  062225                        ADD     (R2)+,(R5)+     ;(R2)=177777,(R5)=000000, CC=0101
 8350  017700  103002                        BCC     ADD2
 8351  017702  102401                        BVS     ADD2
 8352  017704  001401                        BEQ     .+4
 8353  017706  104000                ADD2:   HLT
 8354
 8355  017710  000262                        SEV                     ;SET V
 8356  017712  024527  000001                CMP     -(R5),#1        ;(R5)=000000, CC=1001
 8357  017716  103002                        BCC     CMP2
 8358  017720  102401                        BVS     CMP2
 8359  017722  100401                        BMI     .+4
 8360  017724  104000                CMP2:   HLT
 8361
 8362  017726  054225                        BIS     -(R2),(R5)+     ;(R2)=177777,(R5)=177777,CC=1001
 8363  017730  103001                        BCC     BIS2
 8364  017732  100401                        BMI     .+4
 8365  017734  104000                BIS2:   HLT
 8366  017736  000277                        SCC
 8367  017740  000244                        CLZ
 8368  017742  162245                        SUB     (R2)+,-(R5)     ;(R2)=177777,(R5)=000000, CC=0100
 8369  017744  103402                        BCS     SUB2
 8370  017746  102401                        BVS     SUB2
 8371  017750  001401                        BEQ     .+4
 8372  017752  104000                SUB2:   HLT
 8373
 8374  017754  005442                        NEG     -(R2)           ;(R2)=000001
 8375  017756  005115                        COM     (R5)            ;(R5)=177777
 8376  017760  000277                        SCC
 8377  017762  000250                        CLN
 8378  017764  042225                        BIC     (R2)+,(R5)+     ;(R2)=000001,(R5)=177776, CC=1001
 8379  017766  103003                        BCC     BIC2
 8380
 8381  017770  102402                        BVS     BIC2
 8382  017772  001401                        BEQ     BIC2
 8383  017774  100401                        BMI     .+4
 8384  017776  104000                BIC2:   HLT
 8385
 8386  020000  012742  125252                MOV     #125252,-(R2)
 8387  020004  012245                        MOV     (R2)+,-(R5)
 8388  020006  005125                        COM     (R5)+           ;(R5)=052525
 8389  020010  000262                        SEV
 8390  020012  034245                        BIT     -(R2),-(R5)     ;(R2)=125252,(R5)=052525, CC=0101
 8391  020014  103002                        BCC     BITT2
 8392  020016  102401                        BVS     BITT2
 8393  020020  001401                        BEQ     .+4
 8394  020022  104000                BITT2:  HLT
 8395
 8396  020024  000262                        SEV
 8397  020026  052225                        BIS     (R2)+,(R5)+     ;(R2)=125252,(R5)=177777, CC=1001
 8398  020030  103002                        BCC     BIS2A
 8399  020032  102401                        BVS     BIS2A
 8400  020034  100401                        BMI     .+4
 8401  020036  104000                BIS2A:  HLT
 8402
```

```
 8403  020040  042745  125252                BIC     #125252,-(R5)   ;(R5)=052525
 8404  020044  005125                        COM     (R5)+           ;(R5)=125252
 8405  020046  024245                        CMP     -(R2),-(R5)
 8406  020050  001401                        BEQ     .+4
 8407  020052  104000                        HLT
 8408
 8409  020054  005012                        CLR     (R2)
 8410  020056  005122                        COM     (R2)+           ;(R2)=177777
 8411  020060  162742  000001                SUB     #1,-(R2)        ;(R2)=177776, CC=1000
 8412  020064  103402                        BCS     SUB2A
 8413  020066  102401                        BVS     SUB2A
 8414  020070  100401                        BMI     .+4
 8415  020072  104000              SUB2A:    HLT
 8416  020074  010702                        MOV     PC,R2           ;GET CURRENT PC
 8417  020076  010205                        MOV     R2,R5           ;MOVE TO R5
 8418  020100  124245              1$:       CMPB    -(R2),-(R5)     ;COMPARE ALL PREVIOUS MEMORY ADDRESSES
 8419  020102  001401                        BEQ     .+4
 8420  020104  104000                        HLT                     ;ERROR!
 8421  020106  020237  001610                CMP     R2,@#FRSTAD     ;CHECK FOR LOW LIMIT
 8422  020112  001372                        BNE     1$
 8423                              ;;*****************************************************************
  (3)                              ;*TEST 23       CHECK BINARY BYTE OPS USING ADDRESS MODE 2 & 4
  (3)                              ;;*****************************************************************
  (2)
  (2)  020114  000004              TST23:    SCOPE
  (2)  020116  112737  000023  001252        MOVB    #23,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  020124  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 8424  020132  000402                        BR      .+6             ;RESERVE TWO WORDS
 8425  020134  000000                        .WORD   0               ;SOURCE DATA
 8426  020136  000000                        .WORD   0               ;DESTINATION DATA
 8427  020140  010703                        MOV     PC,R3
 8428  020142  005743                        TST     -(R3)
 8429
 8430                              ;FIRST CHECK AUTO INCREMENT/DECREMENT
 8431  020144  010300                        MOV     R3,R0           ;R0=ADDRESS OF MOV ABOVE
 8432  020146  010002                        MOV     R0,R2           ;R2=R0
 8433  020150  005302                        DEC     R2              ;R2=R0-1
 8434  020152  010604                        MOV     SP,R4
 8435  020154  010605                        MOV     SP,R5
 8436  020156  005745                        TST     -(R5)           ;R5=SP-2
 8437
 8438  020160  114046                        MOVB    -(R0),-(SP)     ;R0=R0-1, SP=SP-2
 8439  020162  020506                        CMP     R5,SP           ;R5=SP
 8440  020164  001021                        BNE     BINB
 8441  020166  020200                        CMP     R2,R0           ;R2=R0
 8442  020170  001017                        BNE     BINB
 8443  020172  122026                        CMPB    (R0)+,(SP)+     ;R0=R0+1, SP=SP+2
 8444  020174  020406                        CMP     R4,SP           ;R4=SP (SP BACK TO ORIGINAL)
 8445  020176  001014                        BNE     BINB
 8446
 8447  020200  020003                        CMP     R0,R3           ;R0=R3 (R0 BACK TO ORIGINAL)
 8448  020202  001012                        BNE     BINB
 8449  020204  154640                        BISB    -(SP),-(R0)     ;SP=SP-2, R0=R0-1
 8450  020206  020506                        CMP     R5,SP           ;R5=SP
 8451  020210  001007                        BNE     BINB
 8452  020212  020200                        CMP     R2,R0           ;R2=R0
```

```
 8453  020214  001005                        BNE     BINB
 8454  020216  142620                        BICB    (SP)+,(R0)+     ;SP=SP+2,R0=R0+1 (SP BACK TO ORIGINAL)
 8455  020220  020406                        CMP     R4,SP           ;R4=SP
 8456  020222  001002                        BNE     BINB
 8457  020224  020003                        CMP     R0,R3           ;R0=R3 .
 8458  020226  001401                        BEQ     .+4
 8459  020230  104000              BINB:     HLT
 8460  020232  010003                        MOV     R0,R3           ;R0=R3
 8461  020234  112743  000200                MOVB    #200,-(R3)      ;R3=ODD BYTE (UPPER BYTE)
 8462  020240  112743  000377                MOVB    #377,-(R3)      ;(R3)=100377, R3=EVEN BYTE (LOWER BYTE)
 8463  020244  010304                        MOV     R3,R4
 8464  020246  112744  000177                MOVB    #177,-(R4)      ;R4= ODD BYTE (UPPER BYTE)
 8465  020252  112744  000000                MOVB    #0,-(R4)        ;(R4)=077400, R4=EVEN BYTE (LOWER BYTE)
 8466  020256  001401                        BEQ     .+4
 8467  020260  104000                        HLT
 8468
 8469  020262  152324                        BISB    (R3)+,(R4)+     ;(R3)=100377,(R4)=077777
 8470  020264  100401                        BMI     .+4
 8471  020266  104000                        HLT
 8472
 8473  020270  122324                        CMPB    (R3)+,(R4)+     ;CC=0X10
 8474  020272  103402                        BCS     CMPB2
 8475  020274  102001                        BVC     CMPB2
 8476  020276  100001                        BPL     .+4
 8477  020300  104000              CMPB2:    HLT
 8478
 8479  020302  000261                        SEC                     ;SET C BIT, CC=0X11
 8480  020304  134344                        BITB    -(R3),-(R4)     ;CC=X101
 8481  020306  103002                        BCC     BITB2
 8482  020310  102401                        BVS     BITB2
 8483  020312  001401                        BEQ     .+4
 8484  020314  104000              BITB2:    HLT
 8485
 8486  020316  000244                        CLZ
 8487  020320  144344                        BICB    -(R3),-(R4)     ;(R3)=100377,(R4)=077400
 8488  020322  001401                        BEQ     .+4
 8489  020324  104000                        HLT
 8490                              ;;*****************************************************************
  (3)                              ;*TEST 24       CHECK BINARY WORD OPS USING ADDRESS MODES 3 & 5
  (3)                              ;;*****************************************************************
  (2)
  (2)  020326  000004              TST24:    SCOPE
  (2)  020330  112737  000024 001252         MOVB    #24,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  020336  013737  001252 177570         MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 8491  020344  000404                        BR      2$              ;RESERVE SPACE FOR DATA AND ADDRESSES
 8492  020346  000000                        .WORD   0               ;CONTAINS ADDRESS OF SOURCE DATA
 8493  020350  000000                        .WORD   0               ;CONTAINS ADDRESS OF DEST DATA
 8494  020352  000000                        .WORD   0               ;CONTAINS SOURCE DATA
 8495  020354  000000                        .WORD   0               ;CONTAINS DEST DATA
 8496  020356  010701              2$:       MOV     PC,R1
 8497  020360  010100                        MOV     R1,R0           ;SET SCOPE PTR
 8498  020362  024040                        CMP     -(R0),-(R0)     ;ADJUST R0
 8499  020364  010005                        MOV     R0,R5           ;R5 POINTS TO DEST DATA
 8500  020366  024545                        CMP     -(R5),-(R5)     ;SUB 4 FROM R5
 8501  020370  010015                        MOV     R0,(R5)         ;R5 POINTS TO ADDRESS OF DEST DATA
 8502  020372  010502                        MOV     R5,R2
```

```
 8503  020374  010004                        MOV     R0,R4           ;R4 POINTS TO DEST DATA
 8504  020376  005740                        TST     -(R0)
 8505  020400  010003                        MOV     R0,R3           ;R3 POINTS TO SOURCE DATA
 8506  020402  010042                        MOV     R0,-(R2)        ;R2 POINTS TO ADDRESS OF SOURCE DATA
 8507  020404  005013                        CLR     (R3)            ;PRESET SOURCE DATA
 8508  020406  005014                        CLR     (R4)            ;PRESET DEST DATA
 8509
 8510  020410  000277                        SCC
 8511  020412  000244                        CLZ
 8512  020414  163235                        SUB     @(R2)+,@(R5)+   ;(R3)=000000,(R4)=000000, CC=0100
 8513  020416  103402                        BCS     SUB3
 8514  020420  102401                        BVS     SUB3
 8515  020422  001401                        BEQ     .+4
 8516  020424  104000              SUB3:     HLT
 8517
 8518  020426  052752  100000              BIS     #100000,@-(R2)  ;(R3)=100000
 8519  020432  062755  000001              ADD     #1,@-(R5)       ;(R4)=000001
 8520  020436  163235                        SUB     @(R2)+,@(R5)+   ;(R3)=100000,(R4)=100001, CC=1011
 8521  020440  103002                        BCC     SUB3A
 8522  020442  102001                        BVC     SUB3A
 8523  020444  100401                        BMI     .+4
 8524  020446  104000              SUB3A:    HLT
 8525
 8526  020450  005414                        NEG     (R4)            ;(R4)=077777
 8527  020452  035255                        BIT     @-(R2),@-(R5)   ;(R3)=100000,(R4)=077777
 8528  020454  001401                        BEQ     .+4
 8529  020456  104000                        HLT
 8530  020460  023235                        CMP     @(R2)+,@(R5)+
 8531  020462  102401                        BVS     .+4
 8532  020464  104000                        HLT
 8533  020466  005152                        COM     @-(R2)
 8534  020470  000257                        CCC
 8535  020472  063255                        ADD     @(R2)+,@-(R5)
 8536  020474  102001                        BVC     ADD3
 8537  020476  100401                        BMI     .+4
 8538  020500  104000              ADD3:     HLT
 8539  020502  000261                        SEC
 8540  020504  045235                        BIC     @-(R2),@(R5)+   ;(R3)=077777,(R4)=100000
 8541  020506  103001                        BCC     BIC3
 8542  020510  100401                        BMI     .+4
 8543  020512  104000              BIC3:     HLT
 8544
 8545  020514  005155                        COM     @-(R5)          ;(R4)=077777
 8546  020516  023235                        CMP     @(R2)+,@(R5)+   ;(R3)=077777,(R4)=077777
 8547  020520  001401                        BEQ     .+4
 8548  020522  104000                        HLT
 8549                              ;;*****************************************************************
  (3)                              ;*TEST 25       CHECK BINARY BYTE OPS USING ADDRESS MODES 3 & 5
  (3)                              ;;*****************************************************************
  (2)
  (2)  020524  000004              TST25:    SCOPE
  (2)  020526  112737  000025  001252        MOVB    #25,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  020534  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 8550  020542  000406                        BR      1$              ;RESERVE SPACE FOR ADDRESS AND DATA
 8551  020544  000000                        .WORD   0               ;CONTAINS ADDRESS OF SOURCE DATA (EVEN BYTE)
 8552  020546  000000                        .WORD   0               ;CONTAINS ADDRESS OF SOURCE DATA (ODD BYTE)
```

```
8553  020550  000000                        .WORD   0               ;CONTAINS ADDRESS OF DEST DATA (EVEN BYTE)
8554  020552  000000                        .WORD   0               ;CONTAINS ADDRESS OF DEST DATA (ODD BYTE)
8555  020554  000000                        .WORD   0               ;CONTAINS SOURCE DATA
8556  020556  000000                        .WORD   0               ;CONTAINS DEST DATA
8557
8558  020560  010700                1$:     MOV     PC,R0
8559  020562  024040                        CMP     -(R0),-(R0)     ;R0=ADDRESS OF DEST DATA
8560  020564  010003                        MOV     R0,R3           ;R3     "       "
8561  020566  010305                        MOV     R3,R5           ;R5     "       "
8562  020570  005743                        TST     -(R3)           ;SUB 2 FROM R3
8563  020572  010043                        MOV     R0,-(R3)        ;R3 POINTS TO ADDRESS OF DEST DATA
8564  020574  005213                        INC     (R3)            ;ODD BYTE
8565  020576  010043                        MOV     R0,-(R3)        ;EVEN BYTE
8566  020600  010304                        MOV     R3,R4
8567  020602  005740                        TST     -(R0)           ;R0=ADDRESS OF SOURCE DATA
8568  020604  010044                        MOV     R0,-(R4)        ;R4 POINTS TO ADDRESS OF SOURCE DATA
8569  020606  005214                        INC     (R4)            ;ODD BYTE
8570  020610  010044                        MOV     R0,-(R4)        ;EVEN BYTE
8571
8572  020612  000261                        SEC                     ;SET CARRY
8573  020614  012734  177001                MOV     #177001,@(R4)+
8574  020620  112734  000200                MOVB    #200,@(R4)+     ;SOURCE DATA=100001
8575  020624  115433                        MOVB    @-(R4),@(R3)+
8576  020626  115433                        MOVB    @-(R4),@(R3)+   ;DEST DATA=000600
8577  020630  103401                        BCS     .+4
8578  020632  104000                        HLT                     ;ERROR! MOV DOES AFFECT C BIT IN PSW
8579  020634  022715  000600                CMP     #600,(R5)       ;CHECK DEST DATA
8580  020640  001401                        BEQ     .+4
8581  020642  104000                        HLT                     ;ERROR! INCORRECT RESULT
8582  020644  024343                        CMP     -(R3),-(R3)     ;POINT R4 BACK TO EVEN BYTE
8583  020646  153433                        BISB    @(R4)+,@(R3)+
8584  020650  153433                        BISB    @(R4)+,@(R3)+   ;DEST DATA=100601
8585  020652  022715  100601                CMP     #100601,(R5)    ;CHECK RESULT
8586  020656  001401                        BEQ     .+4
8587  020660  104000                        HLT                     ;ERROR! INCORRECT DEST DATA AFTER BISB
8588  020662  145453                        BICB    @-(R4),@-(R3)
8589  020664  145453                        BICB    @-(R4),@-(R3)
8590  020666  133433                        BITB    @(R4)+,@(R3)+
8591  020670  001002                        BNE     BITB3
8592  020672  135433                        BITB    @-(R4),@(R3)+
8593  020674  001001                        BNE     .+4
8594  020676  104000            BITB3:      HLT
8595
8596  020700  123453                        CMPB    @(R4)+,@-(R3)
8597  020702  001002                        BNE     CMPB3
8598  020704  123453                        CMPB    @(R4)+,@-(R3)
8599  020706  001401                        BEQ     .+4
8600  020710  104000            CMPB3:      HLT
8601                                ;;*****************************************************************
 (3)                                ;*TEST 26        CHECK BINARY OPS USING ADDRESS MODE 6
 (3)                                ;;*****************************************************************
 (2)
 (2)  020712  000004                TST26:  SCOPE
 (2)  020714  112737  000026  001252        MOVB    #26,@#$TSTNM            ;LOAD TEST NUMBER
 (2)  020722  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
8602  020730  000402                        BR      .+6             ;RESERVE TWO LOCATIONS
```

```
8603  020732  000000                  SDATA:  .WORD   0               ;RESERVED FOR SOURCE DATA
8604  020734  000000                  DDATA:  .WORD   0               ;RESERVED FOR DESTINATION DATA
8605
8606  020736  013702  001604                  MOV     @#FACTOR,R2     ;GET RELOCATION FACTOR AND USE AS AN
8607  020742  010205                          MOV     R2,R5           ;INDEX VALUE TO POINT TO DATA
8608  020744  005065  020734                  CLR     DDATA(5)        ;PRESET DESTINATION DATA
8609  020750  012762  000001  020732          MOV     #1,SDATA(2)     ;THIS ROUTSINE PUT A 1 BIT INTO EVERY
8610  020756  056265  020732  020734  1$:     BIS     SDATA(2),DDATA(5)       ;OTHER BIT POSITION IN THE DEST-
8611  020764  006362  020732                  ASL     SDATA(2)                ;INATION ADDRESS (52525)
8612  020770  006362  020732                  ASL     SDATA(2)
8613  020774  103370                          BCC     1$
8614  020776  022765  052525  020734          CMP     #52525,DDATA(5) ;CHECK RESULT
8615  021004  001401                          BEQ     .+4
8616  021006  104000                          HLT                     ;ERROR! INCORRECT RESULT
8617  021010  012762  177777  020732          MOV     #-1,SDATA(2)
8618  021016  046562  020734  020732          BIC     DDATA(5),SDATA(2)       ;SOURCE DATA=125252
8619  021024  036265  020732  020734          BIT     SDATA(2),DDATA(5)
8620  021032  001401                          BEQ     .+4
8621  021034  104000                          HLT                     ;ERROR! BIT INST FAILED
8622  021036  006365  020734                  ASL     DDATA(5)        ;DDATA=125252
8623  021042  026265  020732  020734          CMP     SDATA(2),DDATA(5)
8624  021050  001401                          BEQ     .+4
8625
8626  021052  104000                          HLT                     ;ERROR! CMP INST FAILED
8627  021054  000257                          CCC
8628  021056  066265  020732  020734          ADD     SDATA(2),DDATA(5)
8629  021064  103002                          BCC     ADD6
8630  021066  102001                          BVC     ADD6
8631  021070  100001                          BPL     .+4
8632  021072  104000                  ADD6:   HLT
8633
8634  021074  006362  020732                  ASL     SDATA(2)        ;SDATA=52524
8635  021100  166265  020732  020734          SUB     SDATA(2),DDATA(5)
8636  021106  103401                          BCS     SUB6
8637  021110  001401                          BEQ     .+4
8638  021112  104000                  SUB6:   HLT
8639
8640  021114  112700  000377                  MOVB    #377,R0         ;R0=177777 (MOVB %R EXTENDS SIGN)
8641  021120  010062  020732                  MOV     R0,SDATA(2)
8642  021124  012765  177777  020734          MOV     #-1,DDATA(5)
8643  021132  166500  020734                  SUB     DDATA(5),R0
8644  021136  001401                          BEQ     .+4
8645  021140  104000                          HLT
8646  021142  066265  020732  020734  1$:     ADD     SDATA(2),DDATA(5)
8647  021150  006362  020732                  ASL     SDATA(2)
8648  021154  005162  020732                  COM     SDATA(2)
8649  021160  036265  020732  020734          BIT     SDATA(2),DDATA(5)
8650  021166  001401                          BEQ     .+4
8651  021170  104000                          HLT
8652  021172  005162  020732                  COM     SDATA(2)
8653  021176  026265  020732  020734          CMP     SDATA(2),DDATA(5)
8654  021204  001401                          BEQ     .+4
8655  021206  104000                          HLT
8656  021210  026200  020732                  CMP     SDATA(2),R0
8657  021214  001352                          BNE     1$
8658                                ;;*****************************************************************
```

```
(3)                                     ;*TEST 27       CHECK BINARY BYTE OPS USING ADDRESS MODE 6
(3)                                     ;;*****************************************************************
(2)
(2)  021216  000004                     TST27:  SCOPE
(2)  021220  112737  000027  001252             MOVB    #27,@#$TSTNM            ;LOAD TEST NUMBER
(2)  021226  013737  001252  177570             MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
8659                                    ;NOTE: SDATAB(2), AND DDATAB(4) REFERENCE EVEN BYTE OF SOURCE & DEST DATA
8660                                    ;AND SDATAB(3), AND DDATAB(5) FEFERENCE ODD BYTE OF SOURCE & DEST DATA
8661
8662  021234  013702  001604                    MOV     @#FACTOR,R2     ;GET INDEX VALUE
8663  021240  010204                            MOV     R2,R4           ;R2 FOR SOURCE EVEN BYTE INDEX, R4 FOR
8664  021242  010403                            MOV     R4,R3           ;DEST ODD BYTE, R3 FOR SOURCE EVEN
8665  021244  005203                            INC     R3              ;AND R5 FOR DEST ODD BYTE
8666  021246  010305                            MOV     R3,R5
8667  021250  000261                            SEC                             ;SET CARRY
8668  021252  012762  125252  021374            MOV     #125252,SDATAB(2)
8669  021260  112763  177125  021374            MOVB    #177125,SDATAB(3)       ;SOURCE DATA = 052652
8670  021266  016264  021374  021376            MOV     SDATAB(2),DDATAB(4)
8671  021274  052764  125125  021376            BIS     #125125,DDATAB(4)       ;DEST DATA = 177777
8672  021302  136263  021374  021374            BITB    SDATAB(2),SDATAB(3)
8673  021310  001401                            BEQ     .+4
8674  021312  104000                    BITB6:  HLT
8675
8676  021314  146264  021374  021376            BICB    SDATAB(2),DDATAB(4)
8677  021322  103401                            BCS     .+4
8678  021324  104000                            HLT                     ;ERROR MOV,BIS,BIT;BIC DO NOT AFFECT 'C'
8679  021326  126364  021374  021376            CMPB    SDATAB(3),DDATAB(4)
8680  021334  001401                            BEQ     .+4
8681  021336  104000                            HLT
8682
8683  021340  146365  021374  021376            BICB    SDATAB(3),DDATAB(5)
8684  021346  126265  021374  021376            CMPB    SDATAB(2),DDATAB(5)
8685  021354  001401                            BEQ     .+4
8686  021356  104000                            HLT
8687
8688  021360  136564  021376  021376            BITB    DDATAB(5),DDATAB(4)
8689  021366  001401                            BEQ     .+4
8690  021370  104000                            HLT
8691  021372  000415                            BR      UB7             ;RESERVE TWO WORDS
8692  021374  000000                    SDATAB: .WORD   0               ;RESERVED FOR SOURCE DATA
8693  021376  000000                    DDATAB: .WORD   0               ;RESERVED FOR DEST DATA
8694
8698                                    ;;*****************************************************************
(3)                                     ;*TEST 30       CHECK BINARY WORD OPS USING ADDRESS MODE 7
(4)                                     ;*      R2=ADDRESS OF SOURCE DATA, AND R3= ADDRESS OF DEST DATA
(3)                                     ;;*****************************************************************
(2)
(2)  021400  000004                     TST30:  SCOPE
(2)  021402  112737  000030  001252             MOVB    #30,@#$TSTNM            ;LOAD TEST NUMBER
(2)  021410  013737  001252  177570             MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
8699  021416  000000                   SBIN7:  .WORD   0               ;CONTAINS ADDRESS OF SOURCE DATA
8700  021420  000000                   DBIN7:  .WORD   0               ;CONTAINS ADDRESS OF DEST DATA
8701  021422  000000                           .WORD   0               ;CONTAINS SOURCE DATA
8702  021424  000000                           .WORD   0               ;CONTAINS DEST DATA
8703
8704  021426  010700                   UB7:    MOV     PC,R0
```

```
 8705  021430  024040                        CMP     -(R0),-(R0)
 8706  021432  010002                        MOV     R0,R2
 8707  021434  024242                        CMP     -(R2),-(R2)
 8708  021436  010012                        MOV     R0,(R2)
 8709  021440  010203                        MOV     R2,R3
 8710  021442  024043                        CMP     -(R0),-(R3)
 8711  021444  010013                        MOV     R0,(R3)
 8712
 8713  021446  000261                        SEC
 8714  021450  012777  100000  177740        MOV     #100000,@SBIN7  ;SOURCE DATA = 100000
 8715  021456  017777  177734  177734        MOV     @SBIN7,@DBIN7   ;DEST DATA = 100000
 8716  021464  103001                        BCC     MOV7
 8717  021466  100401                        BMI     .+4
 8718  021470  104000              MOV7:     HLT
 8719  021472  006377  177722                ASL     @DBIN7          ;DEST DATA = 000000
 8720  021476  102001                        BVC     .+4
 8721  021500  001401                        BEQ     .+4
 8722  021502  104000                        HLT
 8723
 8724  021504  027777  177706  177706        CMP     @SBIN7,@DBIN7   ;(R2)=100000,(R3)=000000
 8725  021512  103402                        BCS     CMP7
 8726  021514  102401                        BVS     CMP7
 8727  021516  100401                        BMI     .+4
 8728  021520  104000              CMP7:     HLT
 8729
 8730  021522  167777  177670  177670        SUB     @SBIN7,@DBIN7   ;(R2)=100000,(R3)=100000
 8731  021530  103003                        BCC     SUB7
 8732  021532  102002                        BVC     SUB7
 8733  021534  001401                        BEQ     SUB7
 8734  021536  100401                        BMI     .+4
 8735  021540  104000              SUB7:     HLT
 8736
 8737  021542  006277  177650                ASR     @SBIN7          ;(R2)=140000
 8738  021546  067777  177644  177644        ADD     @SBIN7,@DBIN7   ;(R2)=140000,(R3)=040000
 8739  021554  103003                        BCC     ADD7
 8740  021556  102002                        BVC     ADD7
 8741  021560  001401                        BEQ     ADD7
 8742  021562  100001                        BPL     .+4
 8743  021564  104000              ADD7:     HLT
 8744
 8745  021566  047777  177624  177624        BIC     @SBIN7,@DBIN7   ;(R2)=140000,(R3)=000000
 8746  021574  001401                        BEQ     .+4
 8747  021576  104000                        HLT
 8748
 8749  021600  057777  177612  177612        BIS     @SBIN7,@DBIN7   ;(R2)=140000,(R3)=140000
 8750  021606  100401                        BMI     .+4
 8751  021610  104000                        HLT
 8752
 8753  021612  027777  177600  177600        CMP     @SBIN7,@DBIN7
 8754  021620  001401                        BEQ     .+4
 8755  021622  104000                        HLT
 8759                            ;;*****************************************************************
  (3)                            ;*TEST 31        SOME MISCELLANEOUS OPERATIONS INVOLVING THE PC
  (4)                            ;*       NOTE: NONE OF THESE OPERATIONS SHOULD AFFECT THE PC
  (3)                            ;;*****************************************************************
  (2)
```

```
  (2) 021624 000004                    TST31:  SCOPE
  (2) 021626 112737 000031 001252              MOVB    #31,@#$TSTNM            ;LOAD TEST NUMBER
  (2) 021634 013737 001252 177570              MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 8760 021642 005000                            CLR     R0
 8761 021644 005067 000072                     CLR     1$
 8762 021650 010707                            MOV     PC,PC
 8763 021652 120707                            CMPB    PC,PC
 8764 021654 030707                            BIT     PC,PC
 8765 021656 060007                            ADD     R0,PC
 8766 021660 105707                            TSTB    PC
 8767 021662 005507                            ADC     PC
 8768 021664 021007                            CMP     (R0),PC
 8769 021666 131007                            BITB    (R0),PC
 8770 021670 062707 000000                     ADD     #0,PC
 8771 021674 023707 001604                     CMP     @#FACTOR,PC
 8772 021700 133707 001604                     BITB    @#FACTOR,PC
 8773 021704 000240                            NOP
 8774                                 ;THE NEXT TWO INSTRUCTION CAUSE THE PROGRAM TO JUMP TO THE UNRELOCATED
 8775                                 ;CODE AND TO RETURN ON THE FOLLOWING INST (IF THE CODE IS RELOCATED)
 8776 021706 163707 001604                     SUB     @#FACTOR,PC     ;JUMPS TO UNRELOCATED CODE
 8777 021712 063707 001604                     ADD     @#FACTOR,PC     ;RETURNS
 8778 021716 000240                            NOP
 8779 021720 024607                            CMP     -(SP),PC
 8780 021722 132607                            BITB    (SP)+,PC
 8781 021724 026707 000012                     CMP     1$,PC
 8782 021730 166707 000006                     SUB     1$,PC
 8783 021734 046707 000002                     BIC     1$,PC
 8784 021740 000401                            BR      .+4             ;BRANCH OVER 1$
 8785 021742 000000                   1$:      0
 8786 021744 000004                   RELE2:   SCOPE
  (1) 021746 010702                            MOV     PC,R2
  (1) 021750 062702 000012                     ADD     #12,R2
  (1) 021754 012707 044042                     MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
  (1) 021760 000000                   REL22:   .WORD   0
  (1)                                  ;2222222222222 LAST ADDRESS OF CODE TO BE RELOCATED 22222222222
  (1)
 8787                                  ;;*****************************************************************
  (3)                                  ;*TEST 32        CHECK BINARY BYTE OPS USING ADDRESS MODE 0
  (3)                                  ;;*****************************************************************
  (2)
  (2) 021762 012767 000001 157406              MOV     #1,$TIMES       ;;DO 1 ITERATION
  (2) 021770 000004                    TST32:  SCOPE
  (2) 021772 112737 000032 001252              MOVB    #32,@#$TSTNM            ;LOAD TEST NUMBER
  (2) 022000 013737 001252 177570              MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 8788
  (1)                                          .SBTTL  START OF SECTION 3
  (1)                                  ;3333333333333 FIRST ADDRESS TO BE RELOCATED 333333333
  (1) 022006 010700                    REL3:   MOV     PC,R0           ;GET PC
  (1) 022010 005740                            TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL3
  (1) 022012 010037 001610                     MOV     R0,@#FRSTAD     ;SAVE
  (1) 022016 010700                            MOV     PC,R0           ;GET CURRENT PC
  (1) 022020 162700 022020                     SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
  (1) 022024 010037 001604                     MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
  (1) 022030 010737 001262                     MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
  (1) 022034 062737 000030 001262              ADD     #30,@#$LPERR    ;ADJUST
  (1) 022042 013737 001262 001260              MOV     @#$LPERR,@#$LPADR
```

```
  (1)  022050  105737  001600          TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
  (1)  022054  001402                  BEQ     .+6
  (1)  022056  000167  002314          JMP     RELE3
 8789  022062  012703  125252          MOV     #125252,R3
 8790  022066  010304                  MOV     R3,R4           ;R3=R4=125252
 8791  022070  140304                  BICB    R3,R4           ;R3=125252,R4=125000
 8792  022072  022704  125000          CMP     #125000,R4      ;CHECK RESULT
 8793  022076  001401                  BEQ     .+4
 8794  022100  104000                  HLT
 8795
 8796  022102  005004                  CLR     R4              ;R3=125252,R4=0
 8797  022104  150304                  BISB    R3,R4           ;R3=125252,R4=000252
 8798  022106  022704  000252          CMP     #252,R4         ;CHECK RESULT
 8799  022112  001401                  BEQ     .+4
 8800  022114  104000                  HLT
 8801
 8802  022116  110404                  MOVB    R4,R4           ;R4=177652
 8803  022120  022704  177652          CMP     #177652,R4      ;CHECK RESULT
 8804  022124  001401                  BEQ     .+4
 8805  022126  104000                  HLT
 8806
 8807  022130  132704  177525          BITB    #177525,R4
 8808  022134  001401                  BEQ     .+4
 8809  022136  104000                  HLT
 8810
 8811  022140  105104                  COMB    R4              ;R4=177525
 8812  022142  110404                  MOVB    R4,R4           ;R4=000125
 8813  022144  022704  000125          CMP     #125,R4         ;CHECK RESULT
 8814  022150  001401                  BEQ     .+4
 8815  022152  104000                  HLT
 8816
 8817  022154  150304                  BISB    R3,R4           ;R3=125252,R4=000377
 8818  022156  105204                  INCB    R4
 8819  022160  001401                  BEQ     .+4
 8820  022162  104000                  HLT
 8821                          ;;*****************************************************************
  (3)                          ;*TEST 33       CHECK BINARY BYTE OPS USING ADDRESS MODE 7
  (3)                          ;;*****************************************************************
  (2)
  (2)  022164  000004          TST33:  SCOPE
  (2)  022166  112737  000033  001252          MOVB    #33,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  022174  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 8822  022202  000406                  BR      BINB7           ;RESERVE SPACE FOR ADDRESSES & DATA
 8823  022204  000000          SBINB7: .WORD   0               ;CONTAINS ADDRESS OF SOURCE EVEN BYTE
 8824  022206  000000                  .WORD   0               ;CONTAINS ADDRESS OF SOURCE ODD BYTE
 8825  022210  000000                  .WORD   0               ;CONTAINS ADDRESS OF DEST EVEN BYTE
 8826  022212  000000                  .WORD   0               ;CONTAINS ADDRESS OF DEST ODD BYTE
 8827  022214  000000          DBINB7: .WORD   0               ;CONTAINS SOURCE DATA
 8828  022216  000000                  .WORD   0               ;CONTAINS DEST DATA
 8829
 8830  022220  010700          BINB7:  MOV     PC,R0
 8831  022222  024040                  CMP     -(R0),-(R0)     ;R0 = ADDRESS OF DEST DATA
 8832  022224  010060  177772          MOV     R0,-6(R0)       ;LOAD ADDRES OF DEST EVEN BYTE DATA
 8833  022230  010060  177774          MOV     R0,-4(R0)
 8834  022234  005260  177774          INC     -4(R0)          ;LOAD ADDRESS OF DEST ODD BYTE DATA
 8835  022240  005740                  TST     -(R0)           ;R0=ADDRESS OF SOURCE DATA
```

```
8836  022242  010060  177770                MOV     R0,-10(R0)      ;LOAD ADDRESS OF SOURCE EVEN BYTE DATA
8837  022246  010060  177772                MOV     R0,-6(R0)
8838  022252  005260  177772                INC     -6(R0)          ;LOAD ADDRESS OF SOURCE ODD BYTE DATA
8839
8840  022256  005002                         CLR     R2              ;SET INDEX REGISTERS
8841  022260  012703  000002                MOV     #2,R3           ;@SBINB7(2);@SBINB7(3) REFERENCE EVEN &
8842  022264  012704  177774                MOV     #-4,R4          ;ODD BYTE SOURCE DATA; @DBINB7(4);@DBINB7(5)
8843  022270  012705  177776                MOV     #-2,R5          ;REFERENCE DEST EVEN& ODD BYTE DATA
8844
8845
8846  022274  005020                         CLR     (R0)+           ;PRESET SOURCE DATA
8847  022276  005010                         CLR     (R0)            ;PRESET DEST DATA
8848  022300  013746  001604                MOV     @#FACTOR,-(SP)  ;GET RELOCATION FACTOR
8849  022304  061602                         ADD     (SP),R2         ;AND ADD TO INDEX VALUES
8850  022306  061603                         ADD     (SP),R3
8851  022310  061604                         ADD     (SP),R4
8852  022312  062605                         ADD     (SP)+,R5
8853
8854  022314  112773  177777  022204        MOVB    #-1,@SBINB7(3)  ;SRC DATA = 177400
8855  022322  132772  000377  022204        BITB    #377,@SBINB7(2) ;CHECK THAT EVEN BYTE WAS NOT AFFECTED
8856  022330  001401                         BEQ     .+4             ;BY MOVB INSTRUCTION
8857  022332  104000                         HLT
8858
8859  022334  157374  022204  022214        BISB    @SBINB7(3),@DBINB7(4)
8860  022342  105274  022214                INCB    @DBINB7(4)      ;CHECK THAT BIS SET ALL BITS
8861  022346  001401                         BEQ     .+4
8862  022350  104000                         HLT
8863
8864  022352  105375  022214                DECB    @DBINB7(5)      ;DEST DATA = 177400
8865  022356  005274  022214                INC     @DBINB7(4)      ;DEST DATA = 177401
8866  022362  127375  022204  022214        CMPB    @SBINB7(3),@DBINB7(5)
8867  022370  001401                         BEQ     .+4
8868  022372  104000                         HLT
8869
8870  022374  147375  022204  022214        BICB    @SBINB7(3),@DBINB7(5)
8871  022402  001401                         BEQ     .+4
8872  022404  104000                         HLT
8873
8874  022406  105073  022204                CLRB    @SBINB7(3)      ;SRC DATA = 000000
8875                                ;THIS ROUTINE SETS ALL BITS IN THE SOURCE ODD BYTE BY BISING A BIT FROM
8876                                ;THE DEST EVEN BYTE INTO THE SOURCE ODD BYTE
8877  022412  157473  022214  022204  BIS7:   BISB    @DBINB7(4),@SBINB7(3)
8878  022420  106174  022214                ROLB    @DBINB7(4)
8879  022424  103372                         BCC     BIS7
8880  022426  022772  177400  022204        CMP     #177400,@SBINB7(2)      ;CHECK RESULT
8881  022434  001401                         BEQ     .+4
8882  022436  104000                         HLT
8883
8884  022440  000372  022204                SWAB    @SBINB7(2)      ;SRC DATA = 000377
8885  022444  112775  000200  022214        MOVB    #200,@DBINB7(5) ;DEST DATA = 100000
8886
8887  022452  147572  022214  022204  BIC7:   BICB    @DBINB7(5),@SBINB7(2)
8888  022460  106075  022214                RORB    @DBINB7(5)
8889  022464  103372                         BCC     BIC7
8890  022466  005772  022204                TST     @SBINB7(2)
8891  022472  001401                         BEQ     .+4
```

```
 8892  022474  104000                          HLT
 8893
 8894  022476  012702  000001          OAERR:  MOV     #1,R2           ;LOAD R2 WITH ODD #
 8895  022502  010703                          MOV     PC,R3
 8896  022504  000401                          BR      .+4             ;RESERVE SPACE FOR A WORD
 8897  022506  000000                          .WORD   0               ;WILL CONTAIN AN ODD ADDRESS
 8898  022510  005723                          TST     (R3)+           ;STEP R3 TO POINT TO WORD ABOVE
 8899  022512  010313                          MOV     R3,(R3)
 8900  022514  005213                          INC     (R3)            ;AND MAKE ODD
 8901  022516  012737  022644  000004          MOV     #1$,@#ERRVEC    ;SET ODD ADDRESS & RESERVED INSTRUCTION
 8902  022524  063737  001604  000004          ADD     @#FACTOR,@#ERRVEC
 8903  022532  013737  000004  000010          MOV     @#ERRVEC,@#RESVEC       ;TO TRAP TO 1$ BELOW
 8904
 8905  022540  000277                          SCC                     ;SET ALL CC'S
 8906  022542  160212                          SUB     R2,(R2)
 8907  022544  104000                          HLT
 8908  022546  060222                          ADD     R2,(R2)+
 8909  022550  104000                          HLT
 8910  022552  006342                          ASL     -(R2)
 8911  022554  104000                          HLT
 8912  022556  106512                          MFPD    (R2)
 8913  022560  104000                          HLT
 8914  022562  170412                          CLRF    (R2)
 8915  022564  104000                          HLT
 8916  022566  042202                          BIC     (R2)+,R2
 8917  022570  104000                          HLT
 8918  022572  164202                          SUB     -(R2),R2
 8919  022574  104000                          HLT
 8920  022576  155202                          BISB    @-(R2),R2
 8921  022600  104000                          HLT
 8922  022602  105532                          ADCB    @(R2)+
 8923  022604  104000                          HLT
 8924  022606  163302                          SUB     @(R3)+,R2
 8925  022610  104000                          HLT
 8926  022612  005733                          TST     @(R3)+
 8927  022614  104000                          HLT
 8928  022616  106533                          MFPD    @(R3)+
 8929  022620  104000                          HLT
 8930  022622  170453                          CLRD    @-(R3)
 8931  022624  104000                          HLT
 8932  022626  137702  177775                  BITB    @.+1,R2
 8933  022632  104000                          HLT
 8934  022634  105477  177773                  NEGB    @.-1
 8935  022640  104000                          HLT
 8936  022642  000406                          BR      2$
 8937
 8938  022644  062716  000002          1$:     ADD     #2,(SP)         ;ADJUST RETURN PC
 8939  022650  052766  000017  000002          BIS     #17,2(SP)       ;SET CONDITION CODES ON RETURN
 8940  022656  000002                          RTI
 8941
 8942  022660  012706  000700          2$:     MOV     #SUPSTK,SP      ;RESET STACK PTR
 8943  022664  012737  064422  000004          MOV     #ERPRT,@#ERRVEC ;RESET TIME OUT VECTOR
 8944  022672  012737  064350  000010          MOV     #RESERR,@#RESVEC
 8945
   (3)                                  ;;*****************************************************************
   (3)                                  ;*TEST 34        CHECK JUMP INSTRUCTIONS
                                        ;;*****************************************************************
```

```
  (2)
  (2)  022700  000004                    TST34:  SCOPE
  (2)  022702  112737  000034  001252             MOVB    #34,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  022710  013737  001252  177570             MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 8946  022716  010700                             MOV     PC,R0
 8947  022720  062700  000012                     ADD     #12,R0          ;SET ADDRESS FOR JMP INST
 8948  022724  000277                             SCC                     ;SET CC'S
 8949  022726  000110                             JMP     (R0)
 8950  022730  000402                             BR      .+6
 8951  022732  000250                             CLN                     ;JMP INST JUMPS HERE
 8952  022734  000775                             BR      .-4
 8953
 8954  022736  103003                             BCC     JMP1
 8955  022740  102002                             BVC     JMP1
 8956  022742  001001                             BNE     JMP1
 8957  022744  100001                             BPL     .+4
 8958  022746  104000                    JMP1:    HLT                     ;ERROR! INCORRECT CC'S AFTER JMP
 8959
 8960  022750  005002                             CLR     R2              ;SET INDICATOR
 8961  022752  010703                             MOV     PC,R3
 8962  022754  000401                             BR      .+4             ;RESERVE WORD FOR JMP ADDRESS
 8963  022756  000000                             .WORD   0               ;CONTAINS ADDRESS FOR JMP INST
 8964  022760  005723                             TST     (R3)+
 8965  022762  010313                             MOV     R3,(R3)
 8966  022764  010300                             MOV     R3,R0
 8967  022766  062713  000022                     ADD     #22,(R3)        ;(R3) IS JMP ADDRESS
 8968  022772  010300                             MOV     R3,R0
 8969  022774  000133                             JMP     @(R3)+          ;JUMP TO ADDRESS CONTAINED IN R3
 8970  022776  000402                             BR      .+6
 8971  023000  005102                             COM     R2              ;COMPLEMENT INDICATOR
 8972  023002  000775                             BR      .-4
 8973  023004  005202                             INC     R2              ;CHECK INDICATOR
 8974  023006  001003                             BNE     JMP3
 8975  023010  005720                             TST     (R0)+
 8976  023012  020003                             CMP     R0,R3           ;CHECK AUTO-INC R3
 8977  023014  001401                             BEQ     .+4
 8978  023016  104000                    JMP3:    HLT
 8979
 8980  023020  005002                             CLR     R2              ;SET INDICATOR
 8981  023022  010704                             MOV     PC,R4           ;SET UP JMP REGISTER
 8982  023024  010400                             MOV     R4,R0           ;SET UP CHECK REGISTER
 8983  023026  000402                             BR      1$
 8984  023030  005102                             COM     R2              ;COMPLEMENT INDICATOR
 8985  023032  000403                             BR      2$
 8986  023034  022424                    1$:      CMP     (R4)+,(R4)+
 8987  023036  005724                             TST     (R4)+           ;R4=JMP ADDRESS
 8988  023040  000144                             JMP     -(R4)           ;USE R4 AS ADDRESS
 8989  023042  005202                    2$:      INC     R2              ;CHECK INDICATOR
 8990  023044  001003                             BNE     JMP4
 8991  023046  022020                             CMP     (R0)+,(R0)+
 8992  023050  020004                             CMP     R0,R4           ;CHECK AUTO-DEC R4
 8993  023052  001401                             BEQ     .+4
 8994  023054  104000                    JMP4:    HLT
 8995
 8996  023056  010703                             MOV     PC,R3
 8997  023060  000401                             BR      .+4             ;RESERVE WORD FOR JMP ADDRESS
```

```
 8998  023062  000000                  1$:     .WORD   0               ;CONTAINS JUMP ADDRESS
 8999  023064  005723                          TST     (R3)+
 9000  023066  010313                          MOV     R3,(R3)
 9001  023070  062723  000016                  ADD     #16,(R3)+
 9002  023074  010300                          MOV     R3,R0           ;LOAD CHECK REGISTER
 9003  023076  000402                          BR      3$
 9004  023100  005102                  2$:     COM     R2
 9005  023102  000401                          BR      4$
 9006  023104  000153                  3$:     JMP     @-(R3)          ;JUMP TO 2$ VIA 1$ ABOVE
 9007  023106  005202                  4$:     INC     R2              ;CHECK INDICATOR
 9008  023110  001003                          BNE     JMP5
 9009  023112  005740                          TST     -(R0)
 9010  023114  020003                          CMP     R0,R3           ;CHECK AUTO-DEC R3
 9011  023116  001401                          BEQ     .+4
 9012  023120  104000                  JMP5:   HLT
 9013
 9014  023122  000402                          BR      2$
 9015  023124  005102                  1$:     COM     R2              ;COMPLEMENT INDICATOR
 9016  023126  000402                          BR      3$
 9017  023130  000167  177770          2$:     JMP     1$
 9018  023134  005202                  3$:     INC     R2
 9019  023136  001401                          BEQ     .+4
 9020  023140  104000                  JMP6:   HLT
 9021
 9022  023142  012767  023160  000020          MOV     #1$,7$          ;SET UP JMP ADDRESS
 9023  023150  063767  001604  000012          ADD     @#FACTOR,7$     ;ADD RELOCATION FACTOR
 9024  023156  000402                          BR      2$              ;GO TO JMP  @7$ INST
 9025  023160  005102                  1$:     COM     R2              ;COMPLEMENT INDICATOR
 9026  023162  000403                          BR      3$              ;GO TO CHECK ROUTINE
 9027  023164  000177  000000          2$:     JMP     @7$             ;JMP TO 1$ ABOVE VIA 7$
 9028  023170  000000                  7$:     .WORD   0               ;CONTAINS JMP ADDRESS
 9029  023172  005202                  3$:     INC     R2              ;CHECK INDICATOR
 9030  023174  001401                          BEQ     .+4
 9031  023176  104000                  JMP7:   HLT
 9032                                  ;;*******************************************************************************
  (3)                                  ;*TEST 35       CHECK JSR INSTRUCTIONS
  (3)                                  ;;*******************************************************************************
  (2)
  (2)   023200  000004                  TST35:  SCOPE
  (2)   023202  112737  000035  001252          MOVB    #35,@#$TSTNM            ;LOAD TEST NUMBER
  (2)   023210  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 9033  023216  013705  001604          JSR1:   MOV     @#FACTOR,R5     ;GET RELOCATION FACTOR
 9034  023222  012702  023254                  MOV     #3$,R2          ;FORM DEST ADRS
 9035  023226  060502                          ADD     R5,R2           ;ADD RELOCATION FACTOR
 9036  023230  000277                          SCC                     ;PRESET CC'S
 9037  023232  000242                          CLV
 9038  023234  004512                          JSR     R5,(R2)         ;GO TO 3$ VIA R2
 9039  023236  005702                  1$:     TST     R2              ;CHECK INDICATOR
 9040  023240  001017                          BNE     4$              ;R2 SHOULD=0
 9041  023242  023705  001604                  CMP     @#FACTOR,R5     ;CHECK THAT RTS  R5 RESTORED R5
 9042  023246  001014                          BNE     4$
 9043  023250  000414                          BR      JSR3            ;GO TO NEXT TEST
 9044  023252  000205                  2$:     RTS     R5              ;RETURN FROM SUBROUTINE
 9045  023254  103011                  3$:     BCC     4$              ;CHECK THAT JSR DID NOT
 9046  023256  102410                          BVS     4$
 9047  023260  001007                          BNE     4$              ;AFFECT CC'S
```

```
 9048  023262  100006                        BPL     4$
 9049  023264  005002                        CLR     R2                      ;CLEAR INDICATOR
 9050  023266  012704  023236                MOV     #1$,R4                  ;GET UNRELOCATED RETURN ADDRESS
 9051  023272  061604                        ADD     (SP),R4                 ;ADD RELOCATION FACTOR (OLD R5)
 9052  023274  020405                        CMP     R4,R5                   ;CHECK THAT OLD R5 WAS PLACED ON THE
 9053  023276  001765                        BEQ     2$                      ;STACK,& THAT NEW R5 CONTAINS RETURN PC
 9054  023300  104000              4$:       HLT                             ;ERROR! ABOVE
 9055
 9056                              ;CHECK JSR INSTRUCTION ADDRESS MODE 3
 9057  023302  013704  001604      JSR3:     MOV     @#FACTOR,R4             ;GET RELOCATION FACTOR
 9058  023306  005000                        CLR     R0                      ;SET INDICATOR
 9059  023310  012705  023330                MOV     #1$,R5
 9060  023314  060405                        ADD     R4,R5                   ;SET UP JSR DEFERRED ADRS
 9061  023316  010502                        MOV     R5,R2
 9062  023320  012715  023346                MOV     #5$,(R5)
 9063  023324  060415                        ADD     R4,(R5)                 ;(R5)=DEST ADRS
 9064  023326  000401                        BR      2$                      ;RESERVE WORD FOR ADDRESS
 9065  023330  000000              1$:       .WORD   0                       ;CONTAINS DEST ADRS FOR JSR
 9066  023332  004435              2$:       JSR     R4,@(R5)+               ;JSR TO 5$ VIA 1$ ABOVE
 9067  023334  005200              3$:       INC     R0                      ;CHECK INDICATOR
 9068  023336  001013                        BNE     6$
 9069  023340  000413                        BR      JSR4
 9070  023342  005100              4$:       COM     R0                      ;COMPLEMENT INDICATOR
 9071  023344  000204                        RTS     4                       ;RETURN FROM SUBROUTINE
 9072  023346  012703  023334      5$:       MOV     #3$,R3                  ;GET UNRELOCATED RETURN ADDRESS
 9073  023352  061603                        ADD     (SP),R3                 ;ADD RELOCATION FACTOR (OLD R4)
 9074  023354  020403                        CMP     R4,R3
 9075  023356  001003                        BNE     6$
 9076  023360  005722                        TST     (R2)+
 9077  023362  020205                        CMP     R2,R5                   ;CHECK AUTO-INC R5
 9078  023364  001766                        BEQ     4$                      ;GO TO RTS
 9079  023366  104000              6$:       HLT                             ;ERROR ABOVE
 9080
 9081                              ;CHECK JSR INST ADDRESS MODE 4
 9082  023370  013704  001604      JSR4:     MOV     @#FACTOR,R4
 9083  023374  010405                        MOV     R4,R5
 9084  023376  010703                        MOV     PC,R3
 9085  023400  000401                        BR      2$
 9086  023402  000405              1$:       BR      4$
 9087  023404  022323              2$:       CMP     (R3)+,(R3)+
 9088  023406  000277                        SCC
 9089  023410  004443                        JSR     R4,-(R3)                ;GO TO 2$
 9090  023412  104000              3$:       HLT
 9091  023414  000414                        BR      JSR6                    ;GO TO NEXT TEST
 9092  023416  103012              4$:       BCC     5$
 9093  023420  102011                        BVC     5$
 9094  023422  001010                        BNE     5$
 9095  023424  100007                        BPL     5$
 9096  023426  012702  023412                MOV     #3$,R2                  ;GET UNRELOCATED RETURN ADDRESS
 9097  023432  061602                        ADD     (SP),R2                 ;ADD RELOCATION FACTOR (OLD R4)
 9098  023434  020204                        CMP     R2,R4                   ;CHECK THAT CALCULATED RETURN
 9099  023436  001002                        BNE     5$                      ;PC = NEW R4
 9100  023440  005724                        TST     (R4)+
 9101  023442  000204                        RTS     R4
 9102  023444  104000              5$:       HLT
 9103
```

```
 9104
 9105  023446  000401                  ;TEST JSR INST ADDRESS MODE 6
 9106  023450  000405                  JSR6:   BR      2$
 9107  023452  010700                  1$:     BR      3$
 9108  023454  004767  177770          2$:     MOV     PC,R0
 9109  023460  100407                          JSR     PC,1$
 9110  023462  104000                          BMI     JSR7            ;GO TO NEXT TEST
 9111  023464  022020                          HLT                     ;ERROR ON CC'S
 9112  023466  020016                  3$:     CMP     (R0)+,(R0)+
 9113  023470  001401                          CMP     R0,(SP)         ;CHECK THAT RETURN ADDRESS IS ON THE
 9114  023472  104000                          BEQ     .+4             ;STACK
 9115  023474  000270                          HLT
 9116  023476  000207                          SEN                     ;SET N
 9117                                          RTS     PC
 9118
 9119  023500  013746  001604          ;TEST JSR INST ADDRESS MODE 7
 9120  023504  062716  023524          JSR7:   MOV     @#FACTOR,-(SP)  ;GET RELOCATION FACTOR
 9121  023510  000277                          ADD     #1$,(SP)        ;FORM ADDRESS OF 1$ BELOW
 9122  023512  004076  000000                  SCC                     ;SET ALL CC'S
 9123  023516  003003                          JSR     R0,@(SP)        ;JSR TO 1$
 9124  023520  102002                          BGT     3$
 9125  023522  000402                          BVC     3$
 9126                                          BR      4$
 9127  023524  000200                  1$:     RTS     R0              ;RETURN
 9128  023526  104000                  3$:     HLT                     ;ERROR!! INCORRECT CC'S
 9133  023530                          4$:
  (4)                             ;;**********************************************************************
  (3)                             ;*TEST 36       CHECK IOT TRAP (AND ROLB/ASLB)
  (4)                             ;*      THIS TEST CHECKS THAT THE PSW IS CORRECT AFTER THE IOT AND THAT THE
  (4)                             ;*      'NEW'PSW (FROM IOTVEC+2) IS CORRECT.
  (3)                             ;;**********************************************************************
  (2)
  (2)  023530  000004              TST36:  SCOPE
  (2)  023532  112737  000036  001252      MOVB    #36,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  023540  013737  001252  177570      MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 9134  023546  012705  000022  IOTTST: MOV     #IOTVEC+2,R5
 9135  023552  005000                  CLR     R0
 9136  023554  052740  000200          BIS     #PR4,-(R0)              ;SET PRIORITY LEVEL 4 IN PSW
 9137  023560  011015                  MOV     (R0),(R5)               ;SET IOTVEC+2 = PSW
 9138  023562  011504                  MOV     (R5),R4                 ;SAVE IN R4
 9139  023564  010746                  MOV     PC,-(SP)
 9140  023566  062716  000036          ADD     #1$-.,(SP)
 9141  023572  012645                  MOV     (SP)+,-(R5)             ;LOAD IOT TRAP VECTOR
 9142  023574  042710  000357          BIC     #PR7+17,(R0)
 9143  023600  052710  000244          BIS     #PR5+4,(R0)             ;PSW=X XXX X00 101 1X1 000
 9144  023604  012003                  MOV     (R0)+,R3                ;R3 = PSW ABOVE
 9145  023606  010340                  MOV     R3,-(R0)                ;RESTORE PSW (MOV CHANGED IT)
 9146  023610  000004                  IOT
 9147  023612  012737  054456  000020  10$:    MOV     #$SCOPE,@#IOTVEC        ;RESTORE IOT VECTOR
 9148  023620  104000                  HLT                             ;ERROR! IOT FAILED TO TRAP
 9149  023622  000457                  BR      TST37           ;;GO TO NEXT TEST
 9150
 9151  023624  012002                  1$:     MOV     (R0)+,R2                ;GET PSW AFTER IOT TRAP
 9152                                                                          ;NOTE: R0=0
 9153  023626  012725  054456                  MOV     #$SCOPE ,(R5)+          ;RESTORE IOTVEC
 9154  023632  012715  000200                  MOV     #PR4,(R5)               ;AND IOTVEC+2
```

```
 9155  023636  010746                        MOV     PC,-(SP)                ;FORM PC OF 10$ ABOVE
 9156  023640  062716  177752                ADD     #10$-.,(SP)
 9157  023644  022626                        CMP     (SP)+,(SP)+             ;CHECK RETURN PC ON STACK
 9158  023646  001036                        BNE     99$
 9159  023650  022603                        CMP     (SP)+,R3                ;CHECK SAVED PSW
 9160  023652  001034                        BNE     99$
 9161  023654  032703  140000                BIT     #UM,R3                  ;BRANCH TO 3$ IF IN USER MODE
 9162  023660  100413                        BMI     3$
 9163  023662  001003                        BNE     2$                      ;BRANCH TO 2$ IF IN SUPER MODE
 9164  023664  020204                        CMP     R2,R4                   ;CHECK PSW AFTER IOT
 9165  023666  001026                        BNE     99$
 9166  023670  000413                        BR      4$
 9167
 9168  023672  042704  030000        2$:     BIC     #PUM,R4                 ;CLEAR PREV MODE BITS
 9169  023676  052704  010000                BIS     #PSM,R4                 ;SET PREV SUPER MODE
 9170  023702  020204                        CMP     R2,R4                   ;CHECK PSW AFTER IOT
 9171  023704  001017                        BNE     99$
 9172  023706  000404                        BR      4$
 9173
 9174  023710  052704  030000        3$:     BIS     #PUM,R4                 ;SET PREV USER MODE
 9175  023714  020204                        CMP     R2,R4                   ;CHECK PSW AFTER IOT
 9176  023716  001012                        BNE     99$
 9177
 9178  023720  005002                4$:     CLR     R2
 9179  023722  000261                        SEC
 9180  023724  106100                        ROLB    R0                      ;ROTATE R0
 9181  023726  102376                        BVC     .-2                     ;UNTIL V SETS (R0=200)
 9182
 9183  023730  106300                        ASLB    R0                      ;SHIFT SHOULD SET CARRY
 9184  023732  103004                        BCC     99$
 9185  023734  102003                        BVC     99$
 9186  023736  001002                        BNE     99$
 9187  023740  005700                        TST     R0
 9188  023742  001401                        BEQ     .+4
 9189  023744  104000                99$:    HLT                             ;ERROR! ROL/ASL FAILED TO SET
 9190                                                                        ;CC'S PROPERLY (IF R2=0) OR IN-
 9191                                                                        ;CORRECT PSW AFTER IOT (IF R2 NOT 0)
 9192  023746  042704  000340                BIC     #PR7,R4
 9193  023752  010437  177776                MOV     R4,@#PSW                ;RESTORE PSW
 9194  023756  012706  000700                MOV     #SUPSTK,SP              ;RESTORE STACK PTR
 9195                                ;;=====================================================================
  (3)                                ;*TEST 37       CHECK EMT TRAP SEQUENCE
  (3)                                ;;=====================================================================
  (2)
  (2)   023762  000004          TST37:  SCOPE
  (2)   023764  112737  000037  001252        MOVB    #37,@#$TSTNM            ;LOAD TEST NUMBER
  (2)   023772  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 9196                                .EQUIV  IOT,HLT                 ;REDEFINE HLT CALL
 9197  024000  012737  054710  000020        MOV     #$ERROR,@#IOTVEC        ;SETUP VECTOR
 9198  024006  012737  000340  000022        MOV     #PR7,@#IOTVEC+2
 9199  024014  005000                        CLR     R0
 9200  024016  010746                        MOV     PC,-(SP)
 9201  024020  062716  000030                ADD     #EMT1-.,(SP)
 9202  024024  012637  000030                MOV     (SP)+,@#EMTVEC
 9203  024030  000262                        SEV                             ;SET V
 9204  024032  013737  177776  000032        MOV     @#PSW,@#EMTVEC+2        ;RETAIN CURRENT PSW ON TRAP
```

```
 9205  024040  000265                        +SEZ!SEC
 9206  024042  104000                        EMT                     ;TRAP TO EMT1
 9207  024044  001433                        BEQ     EMT1C           ;GO TO EMT1C
 9208  024046  000004                        HLT                     ;ERROR! INCORRECT CC'S WERE SET ON RETURN
 9209  024050  102027              EMT1:     BVC     EMT1B           ;'V' SHOULD'VE SET ON EMT TRAP
 9210  024052  105100                        COMB    R0              ;R0=000377,CC'S=1001
 9211  024054  105500                        ADCB    R0              ;R0=000000,CC'S=0101
 9212  024056  106000                        RORB    R0              ;R0=000200,CC'S=1010
 9213  024060  102023                        BVC     EMT1B
 9214  024062  100022                        BPL     EMT1B
 9215  024064  000257                        CCC
 9216  024066  105400                        NEGB    R0              ;R0=000200,CC'S=1010
 9217  024070  102017                        BVC     EMT1B
 9218
 9219  024072  100016                        BPL     EMT1B
 9220  024074  000242                        CLV                     ;CLEAR 'V'
 9221  024076  000261                        SEC                     ;AND SET 'C'
 9222  024100  105300                        DECB    R0              ;R0=000177,CC'S=0011
 9223  024102  102012                        BVC     EMT1B
 9224  024104  100411                        BMI     EMT1B
 9225  024106  000242                        CLV                     ;CLEAR 'V'
 9226  024110  105200                        INCB    R0              ;R0=000200,CC'S=1011
 9227  024112  103006                        BCC     EMT1B
 9228  024114  102005                        BVC     EMT1B
 9229  024116  100004                        BPL     EMT1B
 9230  024120  000242                        CLV                     ;CLEAR 'V'
 9231  024122  106200                        ASRB    R0              ;SHIFT R0 UNTIL 'V' CLEARS
 9232  024124  102776                        BVS     .-2
 9233  024126  000401                        BR      .+4
 9234  024130  000004              EMT1B:    HLT                     ;ERROR!
 9235  024132  000002                        RTI                     ;EXIT WITH R0=000377
 9236  024134  105500              EMT1C:    ADCB    R0              ;R0=000000
 9237  024136  103003                        BCC     EMT1D
 9238  024140  001002                        BNE     EMT1D
 9239  024142  005700                        TST     R0
 9240  024144  001401                        BEQ     .+4
 9241  024146  000004              EMT1D:    HLT
 9242  024150  012737  054710  000030        MOV     #$ERROR,@#EMTVEC        ;RESTORE EMT TO ERROR
 9243  024156  012737  000340  000032        MOV     #PR7,@#EMTVEC+2         ;SET PRIORITY 7 ON ERROR
 9244  024164  012737  054456  000020        MOV     #$SCOPE,@#IOTVEC        ;RESTORE IOT VECTOR
 9245  024172  005037  000022                CLR     @#IOTVEC+2
 9246                                        .EQUIV  ERROR,HLT               ;REDEFINE HLT CALL
 9247                              ;;*****************************************************************
   (3)                             ;*TEST 40       CHECK TRAP INSTRUCTION TRAP SEQUENCE
   (3)                             ;;*****************************************************************
   (2)
   (2)  024176  000004              TST40:  SCOPE
   (2)  024200  112737  000040  001252        MOVB    #40,@#$TSTNM            ;LOAD TEST NUMBER
   (2)  024206  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 9248  024214  052737  000340  177776        BIS     #PR7,@#PSW              ;LOCK OUT LINE CLOCK
 9249  024222  052737  000340  000016        BIS     #PR7,@#TBITVEC+2
 9250  024230  010746                        MOV     PC,-(SP)
 9251  024232  062716  000056                ADD     #TRAP1-.,(SP)
 9252  024236  012637  000034                MOV     (SP)+,@#TRAPVEC
 9253  024242  000270                        SEN                             ;SET N
 9254  024244  013737  177776  000036        MOV     @#PSW,@#TRAPVEC+2       ;RETAIN CURRENT PSW ON TRAP
```

```
 9255  024252  000261                          SEC                                     ;SET CARRY
 9256  024254  010700                          MOV     PC,R0
 9257  024256  000264                          SEZ                             ;SET Z BIT
 9258  024260  104400                          TRAP                            ;TRAP TO TRAP1
 9259  024262  103404                          BCS     .+12
 9260  024264  012737  062270  000034          MOV     #$TRAP,@#TRAPVEC        ;RESTORE TRAP VECTOR
 9261  024272  104000                          HLT
 9262  024274  001404                          BEQ     .+12
 9263  024276  012737  062270  000034          MOV     #$TRAP,@#TRAPVEC        ;RESTORE TRAP VECTOR
 9264  024304  104000                          HLT
 9265  024306  000420                          BR      TRAP1C
 9266  024310  100404                  TRAP1:  BMI     .+12                    ;N BIT GOT SET ON TRAP
 9267  024312  012737  062270  000034          MOV     #$TRAP,@#TRAPVEC        ;RESTORE TRAP VECTOR
 9268  024320  104000                          HLT
 9269  024322  062700  000004                  ADD     #4,R0
 9270  024326  020016                          CMP     R0,(SP)                 ;CHECK LOW BYTE OF RETURN PC ON
 9271  024330  001404                          BEQ     .+12                    ;STACK
 9272  024332  012737  062270  000034          MOV     #$TRAP,@#TRAPVEC        ;RESTORE TRAP VECTOR
 9273  024340  104000                          HLT
 9274  024342  124646                          CMPB    -(SP),-(SP)
 9275  024344  032626                          BIT     (SP)+,(SP)+
 9276  024346  000002                          RTI                             ;RETURN TO INST FOLLOWING TRAP (1$)
 9277
 9278  024350  012702  000036          TRAP1C: MOV     #TRAPVEC+2,R2           ;RESTORE VECTORS
 9279  024354  012712  000340                  MOV     #PR7,(R2)
 9280  024360  012742  062270                  MOV     #$TRAP,-(R2)
 9281  024364  042737  000340  000016          BIC     #PR7,@#TBITVEC+2
 9282  024372  105037  177776                  CLRB    @#PSW                   ;GO BACK TO PRIORITY 0
 9283
 9284  024376  000004                  RELE3:  SCOPE
  (1)  024400  010702                          MOV     PC,R2
  (1)  024402  062702  000012                  ADD     #12,R2
  (1)  024406  012707  044042                  MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
  (1)  024412  000000                  REL33:  .WORD   0
  (1)                                ;3333333333333 LAST ADDRESS OF CODE TO BE RELOCATED 33333333333
  (1)
 9285                                ;;=====================================================================
  (3)                                ;*TEST 41       CHECK STACK OVERFLOW
  (3)                                ;;=====================================================================
  (2)
  (2)  024414  012767  000001  154754          MOV     #1,$TIMES       ;;DO 1 ITERATION
  (2)  024422  000004                  TST41:  SCOPE
  (2)  024424  112737  000041  001252          MOVB    #41,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  024432  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 9286
  (1)                                          .SBTTL  START OF SECTION 4
  (1)                                ;444444444444 FIRST ADDRESS TO BE RELOCATED 444444444
  (1)  024440  010700                  REL4:   MOV     PC,R0                   ;GET PC
  (1)  024442  005740                          TST     -(R0)                   ;R0 CONTAINS THE ADDRESS OF REL4
  (1)  024444  010037  001610                  MOV     R0,@#FRSTAD             ;SAVE
  (1)  024450  010700                          MOV     PC,R0                   ;GET CURRENT PC
  (1)  024452  162700  024452                  SUB     #.,R0                   ;SUBTRACT RELOCATION FACTOR
  (1)  024456  010037  001604                  MOV     R0,@#FACTOR             ;SAVE RELOCATION FACTOR
  (1)  024462  010737  001262                  MOV     PC,@#$LPERR             ;SET LOOP ADDRESS
  (1)  024466  062737  000030  001262          ADD     #30,@#$LPERR            ;ADJUST
  (1)  024474  013737  001262  001260          MOV     @#$LPERR,@#$LPADR
```

```
(1)  024502  105737  001600                TSTB    @#NEXEC          ;BR IF TEST CODE TO BE EXECUTED
(1)  024506  001402                        BEQ     .+6
(1)  024510  000167  001512                JMP     RELE4
9287
9288  024514  013767  177776  000334  OVFLW:  MOV     @#PSW,7$         ;SAVE STATUS IN 7$ BELOW
9289  024522  005037  177776                CLR     @#PSW            ;SET KERNEL MODE
9290  024526  004737  063314                JSR     PC,@#CLRTBIT     ;GO CLEAR 'T' BIT IF SET
9291  024532  052737  000340  177776        BIS     #PR7,@#PSW       ;SET PRIORITY LEVEL 7 TO BLOCK CLOCK
9292  024540  010746                        MOV     PC,-(SP)         ;PUSH CURRENT PC ONTO STACK
9293  024542  062716  000152                ADD     #2$-.,(SP)       ;FORM ADDRESS OF 2$ BELOW
9294  024546  011637  000004                MOV     (SP),@#ERRVEC    ;SET ERROR VECTOR
9295  024552  012737  000340  000006        MOV     #340,@#ERRVEC+2  ;SET PRIORITY LEVEL 7 ON TRAP
9296  024560  013727  000014                MOV     @#BPTVEC,(PC)+   ;SAVE BPT VECTOR ADRS
9297  024564  000000                43$:    .WORD   0
9298  024566  062716  000100                ADD     #41$-2$,(SP)     ;FORM ADDRESS OF 41$ BELOW
9299  024572  012637  000014                MOV     (SP)+,@#BPTVEC   ;SET BPT TRAP VECTOR TO 41$
9300  024576  012737  000340  000016        MOV     #340,@#BPTVEC+2
9301
9302  024604  012703  000376                MOV     #376,R3
9303  024610  010313                        MOV     R3,(R3)          ;LOAD 376 INTO ADDRESS 376
9304  024612  010306                        MOV     R3,SP            ;SET STACK PTR AT BOUNDARY
9305  024614  032767  140000  000234        BIT     #UM,7$           ;CHECK IF ENTERED TEST IN KERNEL
9306  024622  001015                        BNE     1$               ;MODE. BRANCH IF NOT IN KERNEL
9307
9308                            ;THE BELOW INSTRUCTIONS SHOULD NOT CAUSE AN OVERFLOW TRAP
9309  024624  005716                        TST     (SP)             ;BECAUSE TST IS A NON MODIFYING INST
9310  024626  021666  177776                CMP     (SP),-2(SP)      ;SO IS COMPARE
9311  024632  012656                        MOV     (SP)+,@-(SP)     ;BECAUSE OF ADDRESS MODE 5
9312  024634  057636  000000                BIS     @(SP),@(SP)+     ;BECAUSE OF ADDRESS MODE 3
9313  024640  054676  000000                BIS     -(SP),@(SP)      ;BECAUSE OF ADDRESS MODE 7
9314  024644  005006                        CLR     SP
9315  024646  013766  020000  020000        MOV     @#20000,20000(SP)
9316  024654  000425                        BR      3$               ;BRANCH OVER NON KERNEL MODE TESTS
9317
9318                            ;NOTE:  NO OVEFLOW TRAP WILL OCCUR IF NOT IN KERNEL MODE!!!
9319  024656  156737  000175  177777  1$:     BISB    7$+1,@#PSW+1     ;RESTORE MODE BITS IN PSW
9320  024664  012706  000376                MOV     #376,SP          ;SET STACK PTR
9321  024670  016646  177776                MOV     -2(SP),-(SP)     ;SHOULD NOT TRAP
9322  024674  051616                        BIS     (SP),(SP)
9323  024676  061666  177776                ADD     (SP),-2(SP)
9324  024702  105037  177777                CLRB    @#PSW+1          ;SET KERNEL MODE
9325  024706  012706  000700                MOV     #SUPSTK,SP       ;RESTORE THE STACK
9326  024712  000451                        BR      6$               ;EXIT TEST
9327
9328                            ;ERROR SERVICE ROUTINE
9329  024714  012600                2$:     MOV     (SP)+,R0         ;SAVE PC OF INSTRUCTION THAT TRAPPED
9330  024716  012602                        MOV     (SP)+,R2         ;SAVE PSW
9331  024720  012706  000700                MOV     #SUPSTK,SP       ;SET STACK PTR
9332  024724  104000                        HLT                      ;ERROR! AN INSTRUCTION THAT WAS NOT
9333                                                                 ;SUPPOESED TO TRAP TRAPPED
9334                                                                 ;R0 CONTAINS PC, R2 CONTAINS PSW
9335  024726  000443                        BR      6$               ;EXIT TEST
9336                            ;THE BELOW INSTRUCTIONS WILL CAUSE A STACK OVERFLOW
9337                            ;STACK PTR IS AT 376
9338  024730  062737  000066  000004  3$:     ADD     #4$-2$,@#ERRVEC  ;SET ERROR VECTOR TO 4$
9339  024736  010306                        MOV     R3,SP            ;SET STACK PTR AT 376
```

```
 9340  024740  112702  000001                MOVB    #1,R2
 9341  024744  005000                        CLR     R0
 9342  024746  005016                        CLR     (SP)            ;SETS BIT 0 IN R0
 9343  024750  006302                        ASL     R2              ;SHIFT INDICATOR BIT
 9344  024752  105226                        INCB    (SP)+           ;SETS BIT 1 IN R0
 9345  024754  006302                        ASL     R2
 9346  024756  060746                        ADD     PC,-(SP)        ;SETS BIT 2 IN R0
 9347  024760  006302                        ASL     R2
 9348  024762  000003                        BPT                     ;SETS BIT 3 IN R0
 9349  024764  006302                        ASL     R2
 9350  024766  004767  000014                JSR     PC,40$          ;SETS BIT 4 IN R0
 9351  024772  006302                        ASL     R2
 9352  024774  050666  177776                BIS     SP,-2(SP)       ;SETS BIT 5 IN R0
 9353  025000  000410                        BR      5$
 9354
 9355                                  ;PROGRAM WILL TRAP HERE ON OVERFLOW TRAP
 9356  025002  050200                4$:     BIS     R2,R0           ;SET APPROPRIATE BIT IN R0
 9357  025004  000002                        RTI                     ;RETURN FROM TRAP
 9358
 9359  025006  052700  001000        40$:    BIS     #1000,R0                ;SET IND THAT JSR WAS EXECUTED
 9360  025012  000207                        RTS     PC
 9361
 9362  025014  052700  000400        41$:    BIS     #400,R0                 ;SET IND THAT BPT WAS EXECUTED
 9363  025020  000002                        RTI
 9364
 9365                                  ;CHECK THAT ABOVE INSRUCTIONS DID TRAP
 9366  025022  012706  000700        5$:     MOV     #SUPSTK,SP      ;SET STACK PTR
 9367  025026  022700  001477                CMP     #1477,R0        ;EACH INSTRUCTION SET A BIT IN R0
 9368  025032  001401                        BEQ     .+4             ;R0= 1477
 9369  025034  104000                        HLT
 9370
 9371                                  ;EXIT ROUTINE
 9372  025036  012706  001200        6$:     MOV     #KERSTK,SP      ;SET KERNEL STACK PTR
 9373  025042  016737  177516  000014        MOV     43$,@#BPTVEC    ;RESTORE BPT VECTOR
 9374  025050  005037  000016                CLR     @#BPTVEC+2
 9375  025054  012746                        MOV     (PC)+,-(SP)     ;PUSH OLD PSW ONTO STACK
 9376  025056  000000                7$:     .WORD   0               ;CONTAINS SAVED PSW
 9377  025060  010746                        MOV     PC,-(SP)        ;PUSH CURRENT PC ONTO STACK
 9378  025062  062716  000006                ADD     #6,(SP)         ;ADD OFFSET
 9379  025066  000002                        RTI
 9380
 9381  025070  012706  000700                MOV     #SUPSTK,SP      ;SET STACK PTR
 9382  025074  012737  064422  000004        MOV     #ERPRT,@#ERRVEC ;RESET TIME OUT VECTOR
 9383  025102  013737  177776  000006        MOV     @#PSW,@#ERRVEC+2
 9384  025110  052737  000340  000006        BIS     #PR7,@#ERRVEC+2
 9385  025116  042737  000020  000006        BIC     #BIT4,@#ERRVEC+2
 9386  025124  005037  177766                CLR     @#CPUERR
 9387                                  ;;======================================================================
   (3)                                 ;*TEST 42       CHECK THAT ALL RESERVED INSTRUCTIONS TRAP
   (3)                                 ;;======================================================================
   (2)
   (2)  025130  000004                TST42:  SCOPE
   (2)  025132  112737  000042  001252        MOVB    #42,@#$TSTNM            ;LOAD TEST NUMBER
   (2)  025140  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 9388  025146  005737  001552        RESTRP: TST     @#KB11E         ;IS THIS A KB11-E OR KB11-EM?
 9389  025152  001403                        BEQ     10$             ;BR IF NOT
```

```
 9390  025154  012767  000010  000122          MOV     #10,5$          ;KB11-E AND KB11-EM USES OPCODE 7, START WITH OPCODE 10
 9391  025162  012702  025304          10$:    MOV     #5$,R2          ;GET ADDRESS OF RESERVED INSTRUCTION TABLE
 9392  025166  105737  001555                  TSTB    @#CISP          ;IS CISP OPTION PRESENT?
 9393  025172  001402                          BEQ     8$              ;BR IF NOT
 9394  025174  012702  025342                  MOV     #6$,R2          ;ADDRESS OF RESERVED INSTRUCTION TABLE WITH CIS
 9395  025200  063702  001604          8$:     ADD     @#FACTOR,R2
 9396  025204  132737  000040  001551          BITB    #40,@#OPT.CP+1  ;CHECK IF 11/45 FLOATING POINT IS AVAIL.
 9397  025212  001404                          BEQ     9$              ;BRANCH IF NOT AVAILABLE
 9398  025214  005067  000212                  CLR     51$             ;SET CIS TABLE TERMINATOR AT GROUP 7
 9399  025220  005067  000110                  CLR     50$             ;SET TABLE TERMINATOR AT GROUP 7
 9400  025224  012737  025262  000010  9$:     MOV     #4$,@#RESVEC    ;SET RESERVED INSTRUCTION TRAP
 9401  025232  063737  001604  000010          ADD     @#FACTOR,@#RESVEC
 9402  025240  012203                  1$:     MOV     (R2)+,R3        ;GET FIRST RESERVED INSTRUCTION
 9403  025242  001476                          BEQ     7$              ;0 TERMINATES THE TABLE
 9404  025244  012204                          MOV     (R2)+,R4        ;GET LAST RESERVED INSTRUCTION IN GROUP
 9405  025246  010317                  2$:     MOV     R3,(PC)         ;EXECUTE RESERVED INSTRUCTION
 9406  025250  000000                  3$:     .WORD   0               ;CONTAINS RESERVED INSTRUCTION
 9407  025252  000240                          NOP                     ;ERROR! INSTRUCTION IN R3
 9408  025254  000240                          NOP                     ;(2$) ABOVE FAILED TO CAUSE A
 9409  025256  104000                          HLT                     ;RESERVED INSTRUCTION TRAP
 9410  025260  000405                          BR      41$
 9411  025262  012716  025274          4$:     MOV     #41$,(SP)       ;ADJUST RETURN PC
 9412  025266  063716  001604                  ADD     @#FACTOR,(SP)   ;TO RETURN TO 41$
 9413  025272  000002                          RTI                     ;RETURN TO 41$
 9414  025274  020304                  41$:    CMP     R3,R4           ;HAS GROUP OF RESERVED INSTRUCTIONS
 9415  025276  001760                          BEQ     1$              ;BEEN EXECUTED
 9416  025300  005203                          INC     R3              ;INCREMENT THIS RESERVED INSTRUCTION
 9417  025302  000761                          BR      2$              ;TO NEXT ONE AND EXECUTE
 9418                          ;TABLE OF 1170 RESERVED INSTRUCTIONS (0 TERMINATES THE TABLE)
 9419  025304  000007          5$:     7               ;GROUP 1 (GETS A 10 IF KB11-E OR KB11-EM)
 9420  025306  000077                  77              ;  "
 9421  025310  000210                  210             ;GROUP 2
 9422  025312  000227                  227             ;  "
 9423  025314  007000                  7000            ;GROUP 3
 9424  025316  007777                  7777            ;  "
 9425  025320  075040                  75040           ;GROUP 4
 9426  025322  076777                  76777           ;  "
 9427  025324  106400                  106400          ;GROUP 5
 9428  025326  106477                  106477          ;  "
 9429  025330  106700                  106700          ;GROUP 6
 9430  025332  107777                  107777          ;  "
 9431  025334  170000          50$:    170000          ;GROUP 7        FLOATING POINT
 9432  025336  177777                  177777          ;  "              INSTRUCTIONS
 9433  025340  000000                  0               ;0 TERMINATES THE TABLE
 9434
 9435
 9436                          ;TABLE OF KB11-E/EM WITH CIS RESERVED INSTRUCTIONS (0 TERMINATES THE TABLE)
 9437  025342  000010          6$:     10              ;GROUP 1
 9438  025344  000077                  77              ;  "
 9439  025346  000210                  210             ;GROUP 2
 9440  025350  000227                  227             ;  "
 9441  025352  007000                  7000            ;GROUP 3
 9442  025354  007777                  7777            ;  "
 9443  025356  075040                  75040           ;GROUP 4A
 9444  025360  076017                  76017           ;  "
 9445  025362  076033                  76033           ;GROUP 4B
```

```
 9446  025364  076037                          76037                   ;   ''
 9447  025366  076046                          76046                   ;GROUP 4C
 9448
 9449  025370  076047                          76047                   ;   ''
 9450  025372  076100                          76100                   ;GROUP 4D
 9451  025374  076127                          76127                   ;   ''
 9452  025376  076133                          76133                   ;GROUP 4E
 9453  025400  076137                          76137                   ;   ''
 9454  025402  076146                          76146                   ;GROUP 4F
 9455  025404  076147                          76147                   ;   ''
 9456  025406  076160                          76160                   ;GROUP 4G
 9457  025410  076167                          76167                   ;   ''
 9458  025412  076200                          76200                   ;GROUP 4H
 9459  025414  076600                          76600                   ;   ''
 9460  025416  076602                          76602                   ;GROUP 4I
 9461  025420  076777                          76777                   ;   ''
 9462  025422  106400                          106400                  ;GROUP 5
 9463  025424  106477                          106477                  ;   ''
 9464  025426  106700                          106700                  ;GROUP 6
 9465  025430  107777                          107777                  ;   ''
 9466  025432  170000                  51$:    170000                  ;GROUP 7        FLOATING POINT
 9467  025434  177777                          177777                  ;   ''          INSTRUCTIONS
 9468  025436  000000                          0                       ;0 TERMINATES THE TABLE
 9469
 9470  025440  012737  064350  000010  7$:     MOV     #RESERR,@#RESVEC        ;RESTORE RESERVED TRAP
 9471                                  ;;*************************************************************************
  (3)                                  ;*TEST 43       CHECK THAT ALL BITS IN THE PSW CAN BE SET AND CLEARED
  (3)                                  ;;*************************************************************************
  (2)
  (2)  025446  000004                  TST43:  SCOPE
  (2)  025450  112737  000043  001252          MOVB    #43,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  025456  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 9472  025464  105737  001601          PSWCHK: TSTB    @#MMON          ;IF MEM MGMT IS ON SKIP THIS TEST
 9473  025470  001070                          BNE     4$
 9474  025472  013767  177776  000140          MOV     @#PSW,3$        ;SAVE STATUS
 9475  025500  005037  177776                  CLR     @#PSW           ;CLEAR MODE BITS IN PSW
 9476  025504  004737  063314                  JSR     PC,@#CLRTBIT    ;GO CLEAR 'T' BIT IF SET
 9477  025510  013746  000016                  MOV     @#TBITVEC+2,-(SP)
 9478  025514  012704  177776                  MOV     #PSW,R4         ;LOAD ADDRESS OF PSW INTO R4
 9479  025520  000250                          CLN
 9480  025522  005714                          TST     (R4)            ;CHECK THAT PSW WAS CLEARED
 9481  025524  001401                          BEQ     .+4
 9482  025526  104000                          HLT                     ;ERROR! PSW FAILED TO CLEAR
 9483  025530  012700  170357                  MOV     #170357,R0
 9484  025534  005737  001552                  TST     @#KB11E         ;IS THIS A KB11-E OR KB11-EM PROCESSOR?
 9485  025540  001402                          BEQ     10$             ;BR IF NOT
 9486  025542  052700  000400                  BIS     #400,R0         ;ALSO TEST PS08 IF KB11-E
 9487  025546  012702  000001          10$:    MOV     #1,R2           ;R2 = TEST BIT
 9488  025552  030200                  1$:     BIT     R2,R0           ;CHECK IF BIT CAN BE SET/CLEARED
 9489  025554  001423                          BEQ     2$
 9490  025556  005037  000016                  CLR     @#TBITVEC+2
 9491  025562  030227  000020                  BIT     R2,#20          ;CHECK IF TEST WILL SET 'T' BIT
 9492  025566  001403                          BEQ     20$
 9493  025570  012737  000002  000016          MOV     #RTI,@#TBITVEC+2;SET RTI INTO RETURN
 9494  025576  005014                  20$:    CLR     (R4)            ;CLEAR PSW
 9495  025600  050214                          BIS     R2,(R4)         ;SET R2 INTO PSW
```

```
 9496  025602  011403                        MOV     (R4),R3         ;GET BIT
 9497  025604  020203                        CMP     R2,R3           ;CHECK THAT BIT WAS SET IN PSW
 9498  025606  001401                        BEQ     .+4
 9499  025610  104000                        HLT                     ;ERROR! BIT IN R2 FAILED TO SET IN PSW
 9500  025612  000244                        CLZ                     ;CLEAR Z BIT
 9501  025614  040214                        BIC     R2,(R4)         ;CLEAR BIT IN PSW
 9502  025616  011403                        MOV     (R4),R3         ;GET PSW RESULT
 9503  025620  001401                        BEQ     2$              ;BRANCH IF BIC ABOVE CLEARED BIT IN PSW
 9504  025622  104000                        HLT                     ;ERROR! BIT IN R2 FAILED TO CLEAR IN PSW
 9505  025624  006302                2$:     ASL     R2              ;SHIFT TEST BIT
 9506  025626  103351                        BCC     1$              ;BRANCH IF ALL BITS NOT TESTED
 9507  025630  005014                        CLR     (R4)            ;CLEAR STATUS
 9508  025632  012637  000016                MOV     (SP)+,@#TBITVEC+2       ;RESTORE T BIT RETURN
 9509  025636  012746                        MOV     (PC)+,-(SP)     ;PUSH ORIGINAL STATUS ON STACK
 9510  025640  000000                3$:     .WORD   0               ;CONTAINS ORIGINAL PSW
 9511  025642  010746                        MOV     PC,-(SP)        ;SET RETURN PC
 9512  025644  062716  000006                ADD     #6,(SP)
 9513  025650  000002                        RTI                     ;RETURN
 9514  025652  013704  177776        4$:     MOV     @#PSW,R4        ;SAVE PSW IN R4
 9515  025656  112737  000340  177776        MOVB    #340,@#PSW      ;SET PRIORITY LEVEL 7
 9516  025664  004737  063314                JSR     PC,@#CLRTBIT    ;GO CLEAR 'T' BIT IF SET
 9517                                ;;*********************************************************************
  (3)                                ;*TEST 44       CHECK THAT ALL BITS IN THE CURRENT STACK PTR CAN BE SET CLEARED
  (3)                                ;;*********************************************************************
  (2)
  (2)   025670  000004                TST44:  SCOPE
  (2)   025672  112737  000044  001252        MOVB    #44,@#$TSTNM            ;LOAD TEST NUMBER
  (2)   025700  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 9518  025706  010603                CHKSP:  MOV     SP,R3           ;SAVE STACK PTR
 9519  025710  000257                        CCC
 9520  025712  112706  000377                MOVB    #377,SP         ;SET STACK PTR = -1
 9521  025716  006006                1$:     ROR     SP              ;ROTATE 0 BIT THROUGH ALL BIT
 9522  025720  103776                        BCS     1$              ;BIT POSITIONS
 9523  025722  005206                        INC     SP              ;SHOULD INCREMENT SP TO 0
 9524  025724  001403                        BEQ     2$
 9525  025726  010602                        MOV     SP,R2           ;SAVE ERROR STACK PTR
 9526  025730  010306                        MOV     R3,SP           ;SET STACK PTR FOR TRAP
 9527  025732  104000                        HLT                     ;ERROR!
 9528
 9529  025734  010306                2$:     MOV     R3,SP           ;RESTORE ORIGINAL STACK PTR
 9530
 9531                                ;CHECK BYTE OPERATIONS USING THE STACK
 9532  025736  010600                SPCHK:  MOV     SP,R0           ;SAVE STACK PTR
 9533  025740  010003                        MOV     R0,R3
 9534
 9535  025742  005043                        CLR     -(R3)
 9536  025744  112746  177777                MOVB    #-1,-(SP)       ;(SP) = 377
 9537  025750  022713  000377                CMP     #377,(R3)       ;CHECK THAT ONLY EVEN BYTE WAS AFFECTED
 9538  025754  001002                        BNE     1$
 9539  025756  020306                        CMP     R3,SP           ;CHECK AUTO-DEC
 9540  025760  001401                        BEQ     .+4
 9541  025762  104000                1$:     HLT
 9542
 9543  025764  105226                        INCB    (SP)+
 9544  025766  005723                        TST     (R3)+           ;CHECK RESULT
 9545  025770  001002                        BNE     2$
```

```
 9546  025772  020006                          CMP     R0,SP           ;CHECK AUTO-INC
 9547  025774  001401                          BEQ     .+4
 9548  025776  104000                  2$:     HLT
 9549
 9550  026000  005143                          COM     -(R3)           ;(R3)=177777
 9551  026002  144613                          BICB    -(SP),(R3)
 9552  026004  022713  177400                  CMP     #177400,(R3)    ;CHECK RESULT
 9553  026010  001002                          BNE     3$
 9554  026012  020603                          CMP     SP,R3
 9555  026014  001401                          BEQ     .+4
 9556  026016  104000                  3$:     HLT
 9557
 9558  026020  132627  000377                  BITB    (SP)+,#377
 9559  026024  001002                          BNE     4$
 9560  026026  020600                          CMP     SP,R0
 9561  026030  001401                          BEQ     .+4
 9562  026032  104000                  4$:     HLT
 9563
 9564  026034  012746  000001                  MOV     #1,-(SP)
 9565  026040  062706  000002                  ADD     #2,SP
 9566  026044  012702  177401                  MOV     #177401,R2
 9567  026050  120246                          CMPB    R2,-(SP)
 9568  026052  001004                          BNE     5$
 9569  026054  122602                          CMPB    (SP)+,R2
 9570  026056  001002                          BNE     5$
 9571  026060  020006                          CMP     R0,SP
 9572  026062  001401                          BEQ     .+4
 9573  026064  104000                  5$:     HLT
 9574  026066  105037  177776                  CLRB    @#PSW
 9575  026072  010446                          MOV     R4,-(SP)        ;RESTORE ORIGINAL PSW TO STACK
 9576  026074  010746                          MOV     PC,-(SP)
 9577  026076  062716  000006                  ADD     #6,(SP)
 9578  026102  000002                          RTI
 9579                          ;;*****************************************************************
  (3)                          ;*TEST 45       CHECK THAT 'C' BIT SETS/CLEARS PROPERLY
  (3)                          ;;*****************************************************************
  (2)
  (2)  026104  000004          TST45:  SCOPE
  (2)  026106  112737  000045  001252          MOVB    #45,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  026114  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 9580  026122  012727  177776  CBIT:   MOV     #177776,(PC)+   ;LOAD CONSTANT
 9581  026126  000000  1$:     .WORD   0
 9582  026130  010700                  MOV     PC,R0           ;GET CURRENT PC
 9583  026132  162700  000004          SUB     #4,R0           ;POINT R0 TO 1$ ABOVE
 9584  026136  005520          2$:     ADC     (R0)+           ;ADD 'C' BIT TO 1$ ABOVE
 9585  026140  006340                  ASL     -(R0)           ;SHIFT 1$
 9586  026142  102375                  BVC     2$              ;UNTIL 'V' BIT SETS
 9587  026144  022767  077776  177754  CMP     #077776,1$      ;CHECK RESULT
 9588  026152  001401                  BEQ     .+4
 9589  026154  104000                  HLT                     ;ERROR! INCORRECT RESULT IN 1$ ABOVE
 9590                                                          ;R0=ADDRESS OF DATA
 9591
 9592                          ;CHECK THAT CONDITION CODES ARE SET PROPERLY WHEN A NUMBER (CURRENT PC)
 9593                          ;AND THAT NUMBER +1 ARE COMPARED, AND VICE VERSA.
 9594  026156  010700          CMPNUM: MOV     PC,R0           ;GET CURRENT PC
 9595  026160  010002                  MOV     R0,R2           ;SAVE IN R2
```

```
 9596  026162  005202                        INC     R2              ;MAKE R2 = R0+1
 9597  026164  000277                        SCC
 9598  026166  000251                        +CLC!CLN                ;CLEAR C & N BITS
 9599  026170  020002                        CMP     R0,R2           ;COMPARE # WITH #+1
 9600  026172  103003                        BCC     1$              ;CARRY BIT SHOULD SET
 9601  026174  102402                        BVS     1$              ;V BIT SHOULD CLEAR
 9602  026176  001401                        BEQ     1$              ;Z BIT SHOULD CLEAR
 9603  026200  100401                        BMI     .+4             ;N BIT SHOULD SET
 9604  026202  104000              1$:       HLT                     ;ERROR! COMPARE # WITH #+1 FAILED TO
 9605                                                                ;SET CONDITION CODES IN PSW CORRECTLY
 9606
 9607  026204  000277                        SCC                     ;SET CONDITION CODES IN PSW
 9608  026206  120200                        CMPB    R2,R0           ;COMPARE #+1 WITH #
 9609  026210  103403                        BCS     2$              ;C BIT SHOULD CLEAR
 9610  026212  102402                        BVS     2$              ;V BIT SHOULD CLEAR
 9611  026214  001401                        BEQ     2$              ;Z BIT SHOULD CLEAR
 9612  026216  100001                        BPL     .+4             ;N BIT SHOLD CLEAR
 9613  026220  104000              2$:       HLT                     ;ERROR! COMPARE #+1 WITH # FAILED TO SET
 9614                                                                ;CONDITION CODES IN PSW CORRECTLY
 9615  026222  105037  177776              CLRB    @#PSW           ;ENSURE PRIORITY 0
 9616  026226  000004              RELE4:    SCOPE
   (1) 026230  010702                        MOV     PC,R2
   (1) 026232  062702  000012                ADD     #12,R2
   (1) 026236  012707  044042                MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
   (1) 026242  000000              REL44:    .WORD   0
   (1)                             ;444444444444 LAST ADDRESS OF CODE TO BE RELOCATED 44444444444
   (1)
 9617                              ;;*****************************************************************
   (3)                             ;*TEST 46       CHECK EXTENDED INSTRUCTION SET
   (3)                             ;;*****************************************************************
   (2)
   (2) 026244  012767  000001  153124        MOV     #1,$TIMES       ;;DO 1 ITERATION
   (2) 026252  000004              TST46:    SCOPE
   (2) 026254  112737  000046  001252        MOVB    #46,@#$TSTNM            ;LOAD TEST NUMBER
   (2) 026262  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 9618
   (1)                                     .SBTTL  START OF SECTION 5
   (1)                             ;5555555555555 FIRST ADDRESS TO BE RELOCATED 555555555
   (1) 026270  010700              REL5:     MOV     PC,R0           ;GET PC
   (1) 026272  005740                        TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL5
   (1) 026274  010037  001610                MOV     R0,@#FRSTAD     ;SAVE
   (1) 026300  010700                        MOV     PC,R0           ;GET CURRENT PC
   (1) 026302  162700  026302                SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
   (1) 026306  010037  001604                MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
   (1) 026312  010737  001262                MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
   (1) 026316  062737  000030  001262        ADD     #30,@#$LPERR    ;ADJUST
   (1) 026324  013737  001262  001260        MOV     @#$LPERR,@#$LPADR
   (1) 026332  105737  001600                TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
   (1) 026336  001402                        BEQ     .+6
   (1) 026340  000167  001510                JMP     RELE5
 9619  026344  005000              EXTINST:  CLR     R0
 9620  026346  000277                        SCC                     ;PRESET CC'S
 9621  026350  006700                        SXT     R0              ;EXTEND SIGN (1) INTO R0
 9622  026352  103005                        BCC     SXT0            ;CHECK RESULT CC'S
 9623  026354  102404                        BVS     SXT0
 9624  026356  001403                        BEQ     SXT0
```

```
 9625  026360  100002                        BPL     SXT0
 9626  026362  005200                        INC     R0              ;CHECK RESULT
 9627  026364  001401                        BEQ     .+4
 9628  026366  104000                SXT0:   HLT
 9629
 9630  026370  010700                        MOV     PC,R0
 9631  026372  010002                        MOV     R0,R2
 9632  026374  012703  177777                MOV     #-1,R3
 9633  026400  005102                        COM     R2
 9634  026402  000243                        +CLV!CLC                ;CLEAR C AND V BITS
 9635  026404  074003                        XOR     R0,R3           ;R3 SHOULD CONTAIN COMPLEMENT OF R0
 9636  026406  103404                        BCS     XOR0            ;CHECK THAT C WAS NOT AFFECTED
 9637  026410  102403                        BVS     XOR0            ;AND THAT V WAS CLEARED
 9638  026412  001402                        BEQ     XOR0
 9639  026414  020203                        CMP     R2,R3           ;CHECK RESULT
 9640  026416  001401                        BEQ     .+4
 9641  026420  104000                XOR0:   HLT                     ;ERROR!   XOR FAILED
 9642
 9643  026422  010700                        MOV     PC,R0
 9644  026424  022020                        CMP     (R0)+,(R0)+     ;SET ADDRESS REGISTER
 9645  026426  000401                        BR      1$              ;RESERVE WORD FOR TEST DATA
 9646  026430  000000                        .WORD   0               ;CONTAINS TEST DATA
 9647  026432  005700                1$:     TST     R0              ;EXTEND SIGN OF ADDRESS INTO
 9648  026434  006710                        SXT     (R0)            ;ADDRESS (R0)=-1 IF MSB R0=1
 9649  026436  005002                        CLR     R2              ;OTHERWISE, (R0)=0
 9650  026440  005700                        TST     R0              ;CHECK SIGN OF ADDRESS
 9651  026442  100001                        BPL     .+4
 9652  026444  005102                        COM     R2              ;COMPLEMENT CHECK REG IF NEG
 9653  026446  021002                        CMP     (R0),R2         ;CHECK RESULT OF SXT
 9654  026450  001401                        BEQ     .+4
 9655  026452  104000                SXT1:   HLT                     ;ERROR!   SXT FAILED TO EXTEND SIGN PROPERLY
 9656
 9657  026454  012710  100000                MOV     #100000,(R0)    ;PRESET DATA
 9658  026460  011002                        MOV     (R0),R2
 9659  026462  000277                        SCC                     ;PRESET CC'S
 9660  026464  074210                        XOR     R2,(R0)         ;XOR 100000 WITH 100000 RESULT = 0
 9661  026466  103007                        BCC     XOR1            ;CHECK CC'S AFTER XOR
 9662  026470  102406                        BVS     XOR1
 9663  026472  001005                        BNE     XOR1
 9664  026474  100404                        BMI     XOR1
 9665  026476  005710                        TST     (R0)            ;CHECK RESULT (0)
 9666  026500  001002                        BNE     XOR1
 9667  026502  005402                        NEG     R2              ;CHECK THAT REG WAS NOT AFFECTED
 9668  026504  102401                        BVS     .+4
 9669  026506  104000                XOR1:   HLT
 9670
 9671  026510  010702                        MOV     PC,R2
 9672  026512  022222                        CMP     (R2)+,(R2)+
 9673  026514  000401                        BR      SXT4            ;PRESERVE WORD FOR DATA
 9674  026516  000000                        .WORD   0               ;RESERVED FOR DATA
 9675  026520  012722  125252        SXT4:   MOV     #125252,(R2)+   ;PRESET DATA
 9676  026524  006742                        SXT     -(R2)           ;EXTEND SIGN
 9677  026526  074722                        XOR     PC,(R2)+
 9678  026530  010700                        MOV     PC,R0           ;GET PC
 9679  026532  005740                        TST     -(R0)           ;SUBTRACT 2 FROM PC
 9680  026534  005100                        COM     R0              ;R0=RESULT OF XOR PC-1 ABOVE
```

```
 9681  026536  074042                        XOR     R0,-(R2)        ;CHECK RESULT OF SXT AND XOR ABOVE
 9682  026540  001401                        BEQ     .+4
 9683  026542  104000              XOR24:    HLT                     ;ERROR!   SXT & XOR ABOVE INCORRECT
 9684
 9685  026544  012704  000001              MOV     #1,R4           ;SET R4
 9686  026550  006767  000060              SXT     XOR6A           ;PRESET DATA=0
 9687  026554  074467  000054      2$:     XOR     R4,XOR6A
 9688  026560  100423                      BMI     XOR6
 9689  026562  006304                      ASL     R4              ;SHIFT R4
 9690  026564  102373                      BVC     2$              ;UNTIL V SETS (R4=100000)
 9691  026566  100020                      BPL     XOR6            ;BRANCH IF 'N' IS CLEAR
 9692  026570  074467  000040              XOR     R4,XOR6A        ;XOR6A=177777
 9693  026574  100015                      BPL     XOR6
 9694  026576  074767  000032              XOR     PC,XOR6A        ;XOR PC WITH XOR6A (177777)
 9695  026602  010767  000030              MOV     PC,XOR6B        ;FORM PC AS USED IN XOR ABOVE
 9696  026606  162767  000004  000022      SUB     #4,XOR6B
 9697  026614  005167  000016              COM     XOR6B
 9698  026620  026767  000012  000006      CMP     XOR6B,XOR6A     ;XOR6A SHOULD = COMPLEMENT OF PC
 9699  026626  001401                      BEQ     .+4
 9700  026630  104000              XOR6:     HLT                     ;ERROR!  XOR TESTS ABOVE FAILED
 9701
 9702  026632  000402                      BR      .+6
 9703
 9704  026634  000000              XOR6A:    .WORD   0               ;CONTAINS DATA USED BY TEST ABOVE
 9705  026636  000000              XOR6B:    .WORD   0
 9706
 9707
 9708  026640  012700  077777              MOV     #077777,R0      ;SET SOURCE OPERAND FOR ADD
 9709  026644  006767  177764              SXT     XOR6A           ;CLEAR XOR6A
 9710  026650  001004                      BNE     SXT6            ;CHECK CC'S AFTER EXTENDING ZERO'S
 9711  026652  100403                      BMI     SXT6
 9712  026654  103402                      BCS     SXT6
 9713  026656  102401                      BVS     SXT6
 9714  026660  000401                      BR      .+4
 9715  026662  104000              SXT6:     HLT                     ;ERROR!  SXT FAILED
 9716
 9717  026664  012702  000001              MOV     #1,R2           ;SET DEST OPERAND FOR ADD
 9718  026670  013703  001604              MOV     @#FACTOR,R3     ;LOAD INDEX REGISTER
 9719  026674  060002                      ADD     R0,R2           ;RESULT OF ADD=100000
 9720  026676  006763  026634              SXT     XOR6A(3)        ;EXTEND SIGN OF ADD ABOVE
 9721  026702  001403                      BEQ     SXT6A
 9722  026704  005267  177724              INC     XOR6A           ;CHECK RESULT OF SXT
 9723  026710  001401                      BEQ     .+4
 9724  026712  104000              SXT6A:    HLT                     ;ERROR!  SXT ABOVE FAILED TO EXTEND
 9725                                                                ;SIGN
 9726  026714  010703                      MOV     PC,R3
 9727  026716  000402                      BR      .+6             ;PRESERVE 2 WORDS FOR DATA
 9728  026720  000000              SXRA:     .WORD   0               ;RESERVED WORD FOR DATA
 9729  026722  000000              SXRB:     .WORD   0               ;RESERVED WORD FOR DATA
 9730  026724  005723                      TST     (R3)+
 9731  026726  010304                      MOV     R3,R4           ;R3 = ADDRESS OF SXRA
 9732  026730  000250                      CLN                     ;CLEAR N BIT
 9733  026732  006724                      SXT     (R4)+           ;EXTEND ZEROS INTO SXRA
 9734  026734  001401                      BEQ     .+4
 9735  026736  104000              SXT2:     HLT                     ;ERROR!  SXT FAILED
 9736
```

```
 9737  026740  010467  177754                MOV     R4,SXRA         ;SXRA = ADDRESS OF SXRB
 9738  026744  000257                       CCC                     ;CLEAR CONDITION CODES
 9739  026746  006733                       SXT     @(R3)+          ;EXTEND ZEROS INTO SXRB
 9740  026750  001401                       BEQ     .+4
 9741  026752  104000              SXT3:    HLT                     ;ERROR!
 9742
 9743  026754  000270                       SEN                     ;SET N BIT
 9744  026756  006753                       SXT     @-(R3)          ;EXTEND ONES INTO SXRB
 9745  026760  100401                       BMI     .+4
 9746  026762  104000              SXT5:    HLT                     ;ERROR!
 9747
 9748  026764  012704  025252               MOV     #025252,R4      ;R4 = 025252
 9749  026770  074433                       XOR     R4,@(R3)+       ;SXRB = 152525 (COMPLEMENT OF R4)
 9750  026772  005002                       CLR     R2
 9751  026774  074253                       XOR     R2,@-(R3)       ;SXRB REMAINS UNCHANGED
 9752  026776  001405                       BEQ     XOR35           ;CHECK CONDITION CODES
 9753  027000  100004                       BPL     XOR35
 9754  027002  005104                       COM     R4              ;R4 = 152525
 9755  027004  020467  177712               CMP     R4,SXRB         ;CHECK XOR
 9756  027010  001401                       BEQ     .+4
 9757  027012  104000              XOR35:   HLT                     ;ERROR!  XOR FAILED
 9758
 9759  027014  005743                       TST     -(R3)           ;R3 = ADDRESS OF SXRA-2
 9760  027016  000250                       CLN                     ;CLEAR N BIT
 9761  027020  006773  000002               SXT     @2(R3)          ;SXRB = 0
 9762  027024  001401                       BEQ     .+4
 9763  027026  104000              SXT7:    HLT                     ;ERROR!  SXT FAILED
 9764
 9765  027030  074473  000002               XOR     R4,@2(R3)       ;SXRB = R4
 9766  027034  020473  000002               CMP     R4,@2(R3)       ;CHECK XOR
 9767  027040  001401                       BEQ     .+4
 9768  027042  104000              XOR7:    HLT                     ;ERROR!  XOR FAILED
 9773                                      ;;*****************************************************************
  (3)                                      ;*TEST 47       SOB TEST
  (4)                                      ;*      NOTE:   DO NOT INSERT ANY CODE IN FOLLOWING SOB TESTS
  (4)                                      ;               SINCE IT TESTS THE MAXIMUM BRANCH WIDTH OF THE INSTRUCTION.
  (3)                                      ;;*****************************************************************
  (2)
  (2)  027044  000004              TST47:   SCOPE
  (2)  027046  112737  000047  001252       MOVB    #47,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  027054  013737  001252  177570       MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
 9774
 9775  027062  005005                       CLR     R5              ;CLEAR ERROR INDICATOR
 9776  027064  000407                       BR      SOB0            ;BRANCH TO SOB TEST
 9777
 9778  027066  005004              SOB10:   CLR     R4              ;R4 = 0
 9779  027070  005705                       TST     R5              ;CHECK ERROR INDICATOR
 9780  027072  001401                       BEQ     .+4             ;SOB BRANCHED CORRECTLY
 9781  027074  104000                       HLT                     ;ERROR!
 9782
 9783  027076  005005              SOB9:    CLR     R5              ;CLEAR INDICATOR (R5)
 9784  027100  006004                       ROR     R4              ;ROTATE RIGHT R4
 9785  027102  000467                       BR      SOB8
 9786
 9787  027104  012700  000010      SOB0:    MOV     #10,R0          ;R0=10
 9788  027110  000277                       SCC                     ;SET CONDITION CODES
```

```
 9789  027112  001012                SOB1:   BNE     SOB2            ;CHECK CONDITION CODES AFTER SOB
 9790  027114  100011                        BPL     SOB2            ;SOB SHOULD NOT EFFECT THE
 9791  027116  102010                        BVC     SOB2            ;CONDITION CODES.
 9792  027120  103007                        BCC     SOB2
 9793  027122  077005                        SOB     R0,SOB1
 9794  027124  001005                        BNE     SOB2            ;CHECK CONDITION CODES AFTER
 9795  027126  100004                        BPL     SOB2            ;SOB FALLS THROUGH,
 9796  027130  102003                        BVC     SOB2            ;SOB SHOULD NOT EFFECT
 9797  027132  103002                        BCC     SOB2            ;CONDITION CODES.
 9798  027134  005700                        TST     R0              ;CHECK IF R0=0
 9799  027136  001401                        BEQ     .+4
 9800  027140  104000                SOB2:   HLT                     ;ERROR!
 9801
 9802  027142  012702  000100                MOV     #100,R2         ;R2=100
 9803  027146  012700  000101                MOV     #101,R0         ;SET CHECK REGISTER, R0=101
 9804  027152  001414                SOB3:   BEQ     SOB4            ;CHECK CONDITION CODES AFTER
 9805  027154  100413                        BMI     SOB4            ;SOB BRANCH,
 9806  027156  102412                        BVS     SOB4            ;SOB SHOULD NOT EFFECT
 9807  027160  103411                        BCS     SOB4            ;CONDITION CODES.
 9808  027162  005300                        DEC     R0              ;DECREMENT CHECK REGISTER
 9809  027164  020002                        CMP     R0,R2           ;CHECK THAT SOB DECREMENTS
 9810  027166  001006                        BNE     SOB4
 9811  027170  000257                        CCC                     ;SET CONDITION CODES BEFORE SOB
 9812  027172  077211                        SOB     R2,SOB3         ;BRANCH TO SOB3 UNTIL R2=0
 9813  027174  001403                        BEQ     SOB4            ;CHECK CONDITION CODES AFTER
 9814  027176  100402                        BMI     SOB4            ;SOB FALLS THROUGH
 9815  027200  005702                        TST     R2              ;CHECK IF R2=0
 9816  027202  001401                        BEQ     .+4
 9817  027204  104000                SOB4:   HLT                     ;ERROR!
 9818
 9819  027206  012700  000001        SOB5:   MOV     #1,R0           ;R0=1
 9820  027212  000401                        BR      .+4
 9821  027214  104000                        HLT                     ;ERROR!
 9822  027216  077002                        SOB     R0,.-2          ;SOB SHOULD NOT BRANCH
 9823
 9824  027220  005700                        TST     R0              ;CHECK IF R0=0 AFTER SOB
 9825  027222  001401                        BEQ     .+4
 9826  027224  104000                        HLT                     ;ERROR!
 9827
 9828  027226  012704  100000        SOB5A:  MOV     #100000,R4      ;R4=100000
 9829  027232  000403                        BR      1$
 9830  027234  005204                3$:     INC     R4              ;R4=100000
 9831  027236  100403                        BMI     2$              ;N BIT SHOULD BE SET
 9832  027240  104000                        HLT                     ;ERROR!   SOB DID NOT
 9833                                                                ;INCREMENT PROPERLY
 9834
 9835  027242  077404                1$:     SOB     R4,3$           ;SOB SHOULD BRANCH
 9836  027244  104000                        HLT                     ;ERROR!  SOB DID NOT BRANCH
 9837
 9838  027246  012703  000100        2$:     MOV     #100,R3         ;R3=100
 9839  027252  077301                SOB6:   SOB     R3,SOB6         ;USE SOB TO BRANCH TO ITSELF
 9840  027254  005703                        TST     R3              ;CHECK IF R3=0
 9841  027256  001703                        BEQ     SOB10
 9842  027260  104000                SOB7:   HLT                     ;ERROR!
 9843
 9844  027262  005705                SOB8:   TST     R5              ;CHECK INDICATOR (R5)
```

```
 9845                                                   ;IF SOB BRANCHES INCORRECTLY
 9846                                                   ;WHEN CHECKING MAX. BRANCH,
 9847                                                   ;R5 WILL NOT BE CLEARED AT
 9848                                                   ;THIS POINT INDICATING AN ERROR.
 9849
 9850  027264  001401                BEQ     .+4       ;BRANCH IF SOB BRANCHES CORRECTLY
 9851  027266  104000                HLT               ;ERROR!
 9852
 9853  027270  005205                INC     R5        ;SET INDICATOR (R5)
 9854  027272  077477                SOB     R4,SOB9   ;TEST MAX. BRANCH OF SOB
 9855  027274  005704                TST     R4        ;CHECK IF R4=0
 9856  027276  001401                BEQ     .+4
 9857  027300  104000                HLT               ;ERROR!
 9858                          ;;*****************************************************************
  (3)                          ;*TEST 50       CHECK THE MARK INSTRUCTION
  (3)                          ;;*****************************************************************
  (2)
  (2)  027302  000004           TST50:  SCOPE
  (2)  027304  112737  000050  001252   MOVB    #50,@#$TSTNM          ;LOAD TEST NUMBER
  (2)  027312  013737  001252  177570   MOV     @#$TSTNM,@#DISPLAY    ;;DISPLAY TEST NUMBER
 9859  027320  010602           MRKTST: MOV     SP,R2
 9860  027322  010705                   MOV     PC,R5     ;THE STACK LOOKS LIKE THIS AFTER
 9861  027324  010500                   MOV     R5,R0     ;THE JSR INSTRUCTION
 9862  027326  010546                   MOV     R5,-(SP)  ; -2(SP)= R0    THIS IS A
 9863  027330  010746                   MOV     PC,-(SP)  ; -4(SP)= PC    STRING
 9864  027332  010746                   MOV     PC,-(SP)  ; -6(SP)= PC+2  OF
 9865  027334  010746                   MOV     PC,-(SP)  ;-10(SP)= PC+4  FIVE
 9866  027336  010746                   MOV     PC,-(SP)  ;-12(SP)= PC+6  DUMMY
 9867  027340  010746                   MOV     PC,-(SP)  ;-14(SP)= PC+10 ARGUMENTS
 9868  027342  012746  006405           MOV     #MARK+5,-(SP) ;-16(SP)= MARK 5
 9869  027346  010605                   MOV     SP,R5     ;-20(SP)= PC PUSHED BY JSR
 9870  027350  004767  000002           JSR     PC,MARK1
 9871  027354  000403                   BR      .+10
 9872  027356  000205           MARK1:  RTS     R5
 9873  027360  104000                   HLT               ;ERROR! SHOULD BE DOING MARK 5 INST.
 9874  027362  000407                   BR      MARKEX
 9875  027364  020602                   CMP     SP,R2
 9876  027366  001402                   BEQ     .+6
 9877  027370  104000                   HLT               ;ERROR! SP NOT RETURNED TO PROPER
 9878  027372  000403                   BR      MARKEX    ;VALUE BY MARK INSTRUCTION
 9879  027374  020005                   CMP     R0,R5
 9880  027376  001401                   BEQ     .+4
 9881  027400  104000                   HLT               ;ERROR! DID NOT RESTORE R5 FROM STACK
 9882  027402  010206           MARKEX: MOV     R2,SP     ;RESTORE SP
 9906                          ;;*****************************************************************
  (3)                          ;*TEST 51       RTT/RTI TEST
  (4)                          ;*      RTT/RTI TEST INSURES THAT CP DOES THE INSTRUCTION FOLLOWING
  (4)                          ;*      AN RTT IF THE 'T'BIT IS SET IN THE PSW,BUT DOES HONOR
  (4)                          ;*      THE TRAP IMMEDIATELY IF IT EXECUTES AN RTI
  (4)                          ;*      INSTRUCTION SEQUENCE-RTT
  (4)                          ;*      2$:     RTT                   ;NO 'T' TRAP AFTER RTT
  (4)                          ;*              INC     R0            ;R0=000001
  (4)                          ;*                                    ;'T' TRAP TO 5$ AFTER INC
  (4)                          ;*      5$:     COM     R0            ;R0=177776
  (4)                          ;*              MOV     SAVPSW,2(SP)  ;CLEAR 'T' BIT IN RETURN PSW
  (4)                          ;*              RTI                   ;RETURN TO INSTRUCTION FOLLOWING INC
```

```
(4)                                     ;*              CMP     #RTT,2$         ;CHECK
(4)                                     ;*                      ETC
(4)
(4)                                     ;*      INSTRUCTION SQUENCE-RTI
(4)                                     ;*      2$:     RTI                     ;'T' TRAP AFTER RTI
(4)                                     ;*      5$:     COM     R0              ;R0=177777
(4)                                     ;*              MOV     SAVPSW,2(SP)    ;CLEAR 'T' BIT IN RETURN PSW
(4)                                     ;*              RTI                     ;RETURN TO INC INSTRUCTION
(4)                                     ;*              INC     R0              ;R0=000000
(4)                                     ;*              CMP     #RTT,2$         ;CHECK
(4)                                     ;*              ETC
(3)                                     ;;*************************************************************************
(2)
(2)  027404  000004                     TST51:  SCOPE
(2)  027406  112737  000051  001252             MOVB    #51,@#$TSTNM            ;LOAD TEST NUMBER
(2)  027414  013737  001252  177570             MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
9907 027422  013767  177776  000214  RTT1:     MOV     @#PSW,SAVPSW            ;SAVE PSW
9908 027430  032767  000020  000206             BIT     #20,SAVPSW              ;CHECK IF 'T'BIT SET
9909 027436  001402                             BEQ     1$                      ;CONTINUE IF NOT
9910 027440  000167  000402                     JMP     RTT2EX                  ;BRANCH TO EXIT
9911 027444  010746                     1$:     MOV     PC,-(SP)                ;GET CURRENT PC
9912 027446  062716  000116                     ADD     #5$-.,(SP)              ;FORM RELOCATED PC
9913 027452  012637  000014                     MOV     (SP)+,@#TBITVEC         ;LOAD INTO TRAP VECTOR
9914 027456  016746  000162                     MOV     SAVPSW,-(SP)            ;GET CURRENT PSW
9915 027462  011637  000016                     MOV     (SP),@#TBITVEC+2
9916 027466  052737  000340  177776             BIS     #PR7,@#PSW              ;SET PRIORITY LEVEL 7
9917 027474  005000                             CLR     R0
9918 027476  052716  000360                     BIS     #PR7+20,(SP)            ;SET 'T'BIT IN PSW ON STACK
9919 027502  010746                             MOV     PC,-(SP)                ;PUT THE PC ON THE STACK
9920 027504  062716  000006                     ADD     #6,(SP)                 ;ADJUST PC FOR NEXT INSTRUCTION
9921 027510  000006                     2$:     RTT
9922 027512  005200                             INC     R0                      ;DONE TO SEE IF INSTR. FOLLOWING
9923                                                                             ;RTT IS EXECUTED IF T-BIT SET
9924 027514  042737  000340  177776             BIC     #PR7,@#PSW              ;SET PRIORITY LEVEL 0
9925 027522  022767  000006  177760             CMP     #RTT,2$
9926 027530  001005                             BNE     3$                      ;CHECK IF INC WAS EXECUTED
9927 027532  022700  177776                     CMP     #177776,R0              ;CHECK IF COM-R0 EXECUTED
9928 027536  001406                             BEQ     4$
9929 027540  104000                             HLT                             ;ERROR!R0 NOT COMPLIMENTED
9930 027542  000415                             BR      6$                      ;EXIT TEST
9931 027544  005700                     3$:     TST     R0                      ;TEST IF TRAPED BEFORE INC INST.
9932                                                                             ;WAS EXECUTED
9933 027546  001413                             BEQ     6$
9934 027550  104000                             HLT                             ;ERROR!
9935 027552  000411                             BR      6$                      ;EXIT TEST
9936 027554  012767  000002  177726     4$:     MOV     #RTI,2$
9937 027562  000730                             BR      1$
9938 027564  005100                     5$:     COM     R0                      ;RTT CHECK
9939 027566  016766  000052  000002             MOV     SAVPSW,2(SP)
9940 027574  000002                             RTI
9941 027576  012767  000006  177704     6$:     MOV     #RTT,2$
9942 027604  012737  001570  000014             MOV     #$RTRN,@#TBITVEC                ;RESTORE 'T' TRAP VECTOR
9943 027612  005037  000016                     CLR     @#TBITVEC+2
9944 027616  042737  000360  000016             BIC     #PR7+BIT4,@#TBITVEC+2
9945 027624                     RTT1EX:
 (4)                                    ;;*************************************************************************
```

```
(3)                                     ;*TEST 52       SECOND RTT TEST
(3)                                     ;;**************************************************************************
(2)
(2)  027624  000004                     TST52:  SCOPE
(2)  027626  112737  000052  001252             MOVB    #52,@#$TSTNM                    ;LOAD TEST NUMBER
(2)  027634  013737  001252  177570             MOV     @#$TSTNM,@#DISPLAY              ;;DISPLAY TEST NUMBER
9946 027642  000401                             BR      RTT2A
9947 027644  000000                     SAVPSW: .WORD   0
9948 027646  016700  177772             RTT2A:  MOV     SAVPSW,R0                       ;GET SAVED PSW
9949 027652  105000                             CLRB    R0                              ;CLEAR PRIOITY LEVEL,T, AND COND CODES
9950 027654  012702  144000                     MOV     #UM+REG,R2
9951 027660  074002                             XOR     R0,R2
9952 027662  001435                             BEQ     2$                              ;USER MODE REG. SET #1 ON
9953 027664  012702  044000                     MOV     #SM+REG,R2
9954 027670  074002                             XOR     R0,R2
9955 027672  001447                             BEQ     3$                              ;SUPER MODE REG. SET #1 ON
9956 027674  032700  140000                     BIT     #UM,R0
9957 027700  001062                             BNE     RTT2EX
9958
9959                                    ;TEST THAT RTT CLEARS BITS 11,12,13 & PRIORITY LEVEL BITS IN KERNEL MODE
9960 027702  012702  177777                     MOV     #-1,R2                          ;KERNEL MODE REG. SET 0 ON
9961 027706  012737  034240  177776             MOV     #PUM+REG+PR5,@#PSW              ;SELECT REG. SET #1
9962 027714  005002                             CLR     R12                             ;SHOULD CLEAR REG #12
9963 027716  012746  000100                     MOV     #PR2,-(SP)
9964 027722  010746                             MOV     PC,-(SP)
9965 027724  062716  000006                     ADD     #1$-.,(SP)                      ;FORM NEW PC
9966 027730  000006                             RTT
9967 027732  013700  177776             1$:     MOV     @#PSW,R0                        ;NOW USING REG SET 0
9968 027736  005702                             TST     R2                              ;SHOULD TEST R2 NOT R12
9969 027740  001001                             BNE     4$
9970 027742  104000                             HLT                                     ;ERROR!DID NOT CLEAR BIT #11 OF PSW
9971 027744  022700  000100             4$:     CMP     #PR2,R0         ;TESTS THE PSW AFTER THE RTT
9972 027750  001436                             BEQ     RTT2EX
9973 027752  104000                             HLT                                     ;ERROR! INCORRECT PSW AFTER THE RTT
9974 027754  000434                             BR      RTT2EX
9975
9976                                    ;TEST TO INSURE THAT RTI DOES NOT CLEAR BITS 11-15 IN USER MODE
9977 027756  052737  030340  177776     2$:     BIS     #PUM+PR7,@#PSW  ;PSW<15-5>=144X
9978 027764  005046                             CLR     -(SP)
9979 027766  010746                             MOV     PC,-(SP)
9980 027770  062716  000006                     ADD     #5$-.,(SP)
9981 027774  000002                             RTI                                     ;ATTEMPS TO INSERT A PSW OF 0
9982 027776  022737  174340  177776     5$:     CMP     #UM+PUM+REG+PR7,@#PSW   ;SHOULD CHECK AGAINST REG #0
9983 030004  001420                             BEQ     RTT2EX
9984 030006  104000                             HLT                                     ;ERROR! RTI CLEARED BITS IN PSW
9985 030010  000416                             BR      RTT2EX
9986
9987                                    ;TEST THAT BITS 11-15 AND PRIORTY BITS ARE NOT ALTERED IN SUPER MODE
9988 030012  052737  030200  177776     3$:     BIS     #PUM+PR4,@#PSW  ;PSW<15-5>=044X
9989 030020  012746  000340                     MOV     #PR7,-(SP)
9990 030024  010746                             MOV     PC,-(SP)
9991 030026  062716  000006                     ADD     #6$-.,(SP)
9992 030032  000006                             RTT                                     ;ATTEMPTS TO CLEAR 11-15 AND ALTER PR
9993
9994 030034  022737  074200  177776     6$:     CMP     #SM+PUM+REG+PR4,@#PSW
9995 030042  001401                             BEQ     RTT2EX
```

```
 9996  030044  104000                          RTT2EX: HLT                             ;ERROR! RTT ALTERED PR IN
 9997                                                                                  ;SUPER MODE OR BITS 11-15.
 9998  030046  016737  177572  177776                  MOV     SAVPSW,@#PSW
 9999  030054  000004                          RELE5:  SCOPE
   (1) 030056  010702                                  MOV     PC,R2
   (1) 030060  062702  000012                          ADD     #12,R2
   (1) 030064  012707  044042                          MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
   (1) 030070  000000                          REL55:  .WORD   0
   (1)                                         ;5555555555555 LAST ADDRESS OF CODE TO BE RELOCATED 55555555555
   (1)
10000                                          ;;*******************************************************************
   (3)                                         ;*TEST 53        CHECK ASH, ASHC, MUL, AND DIV INSTRUCTIONS
   (3)                                         ;;*******************************************************************
   (2)
   (2) 030072  012767  000001  151276                  MOV     #1,$TIMES       ;;DO 1 ITERATION
   (2) 030100  000004                          TST53:  SCOPE
   (2) 030102  112737  000053  001252                  MOVB    #53,@#$TSTNM            ;LOAD TEST NUMBER
   (2) 030110  013737  001252  177570                  MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10001
   (1)                                                 .SBTTL  START OF SECTION 6
   (1)                                         ;6666666666666 FIRST ADDRESS TO BE RELOCATED 666666666
   (1) 030116  010700                          REL6:   MOV     PC,R0           ;GET PC
   (1) 030120  005740                                  TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL6
   (1) 030122  010037  001610                          MOV     R0,@#FRSTAD     ;SAVE
   (1) 030126  010700                                  MOV     PC,R0           ;GET CURRENT PC
   (1) 030130  162700  030130                          SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
   (1) 030134  010037  001604                          MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
   (1) 030140  010737  001262                          MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
   (1) 030144  062737  000030  001262                  ADD     #30,@#$LPERR    ;ADJUST
   (1) 030152  013737  001262  001260                  MOV     @#$LPERR,@#$LPADR
   (1) 030160  105737  001600                          TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
   (1) 030164  001402                                  BEQ     .+6
   (1) 030166  000167  002120                          JMP     RELE6
10002  030172  012700  000001          ASHL0:  MOV     #1,R0           ;R0 WILL BE THE SHIFT COUNT
10003  030176  012703  000021                  MOV     #17.,R3         ;MAX SHIFT COUNT
10004  030202  005067  000014          1$:     CLR     2$              ;PRESET SAVED CC'S LOCATION=0
10005  030206  010002                          MOV     R0,R2           ;GET SHIFT COUNT FOR PASS
10006  030210  010705                          MOV     PC,R5           ;R5 & R4 WILL BE DATA SHIFTED BY
10007  030212  010504                          MOV     R5,R4           ;ASH & ASL INSTRUCTIONS
10008  030214  072502                          ASH     R2,R5           ;SHIFT R5
10009  030216  113727  177776                  MOVB    @#PSW,(PC)+     ;SAVE CC'S
10010  030222  000000                  2$:     .WORD   0               ;CONTAINS ASH CC'S IN EVEN BYTE
10011                                                                   ;ASL CC'S IN ODD BYTE
10012  030224  006304                  3$:     ASL     R4              ;SHIFT R4
10013  030226  113746  177776                  MOVB    @#PSW,-(SP)     ;SAVE PSW ON STACK
10014  030232  132716  000002                  BITB    #2,(SP)         ;CHECK IF ASL SET V BIT
10015  030236  001403                          BEQ     30$
10016  030240  152767  000002  177755          BISB    #2,2$+1         ;IF ASL SET V THEN SET V IN 2$+1
10017  030246  112637  177776          30$:    MOVB    (SP)+,@#PSW     ;RESTORE ORIGINAL PSW
10018  030252  077214                          SOB     R2,3$           ;SHIFT R4 R2 TIMES
10019  030254  153767  177776  177741          BISB    @#PSW,2$+1      ;SAVE CC'S AFTER ASL
10020  030262  020504                          CMP     R5,R4           ;CHECK ASH & ASL RESULTS
10021  030264  001004                          BNE     4$
10022  030266  126767  177730  177727          CMPB    2$,2$+1         ;CHECK ASH & ASL CC'S
10023  030274  001401                          BEQ     .+4
10024  030276  104000                  4$:     HLT                     ;ERROR! INCORRECT RESULT OR CC'S
```

```
10025  030300  005200                        INC     R0              ;INCREMENT PASS SHIFT COUNT
10026  030302  020003                        CMP     R0,R3
10027  030304  001336                        BNE     1$
10028
10029  030306  012700  177777        ASHR0:  MOV     #-1,R0          ;R0 = RIGHT SHIFT COUNT FOR PASS
10030  030312  012703  177757                MOV     #-17.,R3        ;MAX SHIFT COUNT
10031  030316  010002                1$:     MOV     R0,R2           ;GET SHIFT COUNT FOR PASS
10032  030320  010705                        MOV     PC,R5           ;R5 & R4 = DATA TO BE SHIFTED
10033  030322  010504                        MOV     R5,R4           ;BY ASH & ASR INSTRUCTIONS
10034  030324  072502                        ASH     R2,R5           ;SHIFT R5 R2 TIMES
10035  030326  113727  177776              MOVB    @#PSW,(PC)+     ;SAVE CC'S IN EVEN BYTE
10036  030332  000000                2$:     .WORD   0               ;CONTAINS ASH CC'S IN EVEN BYTE
10037                                                                 ;ASR CC'S IN ODD BYTE
10038  030334  005402                        NEG     R2
10039  030336  006204                3$:     ASR     R4              ;SHIFT R4
10040  030340  077202                        SOB     R2,3$           ;SHIFT R4 R2 TIMES
10041  030342  113767  177776  177763      MOVB    @#PSW,2$+1      ;SAVE CC'S AFTER ASR
10042  030350  142767  000002  177755      BICB    #2,2$+1         ;ASH RIGHT WILL NOT SET V ASR MAY SET V
10043  030356  020504                        CMP     R5,R4           ;CHECK ASH & ASR RESULTS
10044  030360  001004                        BNE     4$
10045  030362  126767  177744  177743      CMPB    2$,2$+1         ;CHECK ASH & ASR CC'S
10046  030370  001401                        BEQ     .+4
10047  030372  104000                4$:     HLT
10048  030374  005300                        DEC     R0              ;DECREMENT PASS SHIFT COUNT
10049  030376  020003                        CMP     R0,R3
10050  030400  001346                        BNE     1$
10051
10052  030402  012746  000037        ASHCL0: MOV     #31.,-(SP)      ;PUT MAX SHIFT COUNT ON STACK
10053  030406  012746  000001                MOV     #1,-(SP)        ;PUT LEFT SHIFT COUNT ON STACK
10054  030412  011600                1$:     MOV     (SP),R0         ;GET PASS SHIFT COUNT
10055  030414  010705                        MOV     PC,R5           ;CURRENT PC IS DATA TO BE SHIFTED
10056  030416  010503                        MOV     R5,R3           ;ASHC SHIFTS R4,R5;ASL,ROL SHIFTS R2,R3
10057  030420  005004                        CLR     R4
10058  030422  005002                        CLR     R2
10059  030424  073400                        ASHC    R0,R4           ;SHIFT R4 LEFT AS SPECIFIED BY R0
10060  030426  006303                2$:     ASL     R3              ;SHIFT R2,R3 LEFT
10061  030430  006102                        ROL     R2              ;AS SPECIFIED BY R0
10062  030432  077003                        SOB     R0,2$
10063  030434  020402                        CMP     R4,R2           ;CHECK RESULTS
10064  030436  001002                        BNE     3$
10065  030440  020503                        CMP     R5,R3
10066  030442  001401                        BEQ     .+4
10067  030444  104000                3$:     HLT
10068  030446  005216                        INC     (SP)            ;INCREMENT NEXT PASS SHIFT COUNT
10069  030450  021666  000002              CMP     (SP),2(SP)      ;REACHED MAX COUNT (31.)
10070  030454  001356                        BNE     1$
10071  030456  022626                        CMP     (SP)+,(SP)+     ;RESTORE STACK PTR
10072
10073  030460  012746  177740        ASHCR0: MOV     #-32.,-(SP)     ;PUT MAX RIGHT SHIFT COUNT ON STACK
10074  030464  012746  177777                MOV     #-1,-(SP)       ;PUT PASS SHIFT COUNT ON STACK
10075  030470  011600                1$:     MOV     (SP),R0         ;GET PASS SHIFT COUNT
10076  030472  010702                        MOV     PC,R2           ;R2,R3 & R4,R5 ARE THE DATA REGISTERS
10077  030474  010204                        MOV     R2,R4           ;TO BE SHIFTED BY TEST
10078  030476  005003                        CLR     R3
10079  030500  005005                        CLR     R5
10080  030502  000262                        SEV                     ;SET V BIT IN PSW
```

```
10081  030504  073200                        ASHC    R0,R2           ;SHIFT R2,R3 RIGHT R0 TIMES
10082  030506  102410                        BVS     3$              ;SHIFT RIGHT CLEARS V
10083  030510  005400                        NEG     R0              ;NEGATE SHIFT COUNT FOR SOB
10084  030512  006204                2$:     ASR     R4              ;SHIFT R4,R5 RIGHT R0 TIMES
10085  030514  006005                        ROR     R5
10086  030516  077003                        SOB     R0,2$
10087  030520  020204                        CMP     R2,R4           ;CHECK RESULT
10088  030522  001002                        BNE     3$
10089  030524  020305                        CMP     R3,R5
10090  030526  001401                        BEQ     .+4
10091  030530  104000                3$:     HLT
10092  030532  005316                        DEC     (SP)            ;SET SHIFT COUNT FOR NEXT PASS
10093  030534  021666  000002                CMP     (SP),2(SP)      ;CHECK IF MAX SHIFT COUNT
10094  030540  001353                        BNE     1$
10095  030542  022626                        CMP     (SP)+,(SP)+     ;RESTORE STACK PTR
10101                                  ;;*****************************************************************
   (3)                                 ;*TEST 54       CHECK MUL
   (4)                                 ;*      THE BELOW TEST OF THE MUL INSTRUCTION MULTIPLIES THE CURRENT PC
   (4)                                 ;*      BY 1,2,4,8 ETC AND SHIFTS THE SAME PC VALUE USING AN ASHC LEFT BY
   (4)                                 ;*      0,1,2,3,ETC AND COMPARES THE RESULTS. CONDITION CODE RESULTS ARE NOT CHECKED.
   (3)                                 ;;*****************************************************************
   (2)
   (2)  030544  000004                TST54:  SCOPE
   (2)  030546  112737  000054 001252         MOVB    #54,@#$TSTNM            ;LOAD TEST NUMBER
   (2)  030554  013737  001252 177570         MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10102  030562  012700  000001         MUL0:   MOV     #1,R0           ;R0 CONTAINS MULTIPLIER FOR MUL
10103  030566  012706  000700                 MOV     #SUPSTK,SP      ;SETUP THE STACK
10104  030572  005016                         CLR     (SP)            ;(SP) CONTAINS SHIFT VALUE FOR ASHC
10105  030574  010702                1$:      MOV     PC,R2           ;R3,R2 & R5,R4 ARE DATA REGISTERS
10106  030576  010227                         MOV     R2,(PC)+        ;SAVE MULTIPICAND
10107  030600  000000                         .WORD   0               ;CONTAINS ORIGINAL MULTIPICAND
10108  030602  005003                         CLR     R3
10109  030604  005004                         CLR     R4
10110  030606  010205                         MOV     R2,R5           ;FOR MUL AND ASHC
10111  030610  100001                         BPL     .+4             ;IF MULTPICAND IS NEG THEN SET R4 = -1
10112  030612  005104                         COM     R4              ;FOR ASHC
10113  030614  000277                         SCC                     ;PRESET CC'S
10114  030616  070200                         MUL     R0,R2           ;MULTIPLY R2 BY R0 LEAVE PRODUCT
10115                                                                  ;IN R2,R3 MSH IN R2,LSH IN R3
10116  030620  102406                         BVS     2$
10117  030622  001405                         BEQ     2$              ;PRODUCT WILL NEVER BE = 0
10118  030624  073416                         ASHC    (SP),R4         ;'MULTIPLY' R4,R5 BY (SP) LEAVE PRODUCT
10119                                                                  ;IN R4,R5 MSH IN R4,LSH IN R5
10120  030626  020204                         CMP     R2,R4           ;CHECK MSH RESULT
10121  030630  001002                         BNE     2$
10122  030632  020305                         CMP     R3,R5           ;CHECK LSH RESULT
10123  030634  001401                         BEQ     .+4
10124  030636  104000                2$:      HLT
10125  030640  005216                         INC     (SP)            ;INCREMENT ASHC SHIFT COUNT
10126  030642  006300                         ASL     R0              ;SHIFT MUL MULTIPLIER
10127  030644  102353                         BVC     1$
10128                                  ;CHECK MUL INST WITH MULTIPIER (R0) = 100000
10129  030646  010702                         MOV     PC,R2           ;R2 = MULTIPICAND
10130  030650  005202                         INC     R2
10131  030652  010227                         MOV     R2,(PC)+        ;SAVE MULTIPICAND
10132  030654  000000                         .WORD   0               ;CONTAINS ORIGINAL MULTIPICAND
```

```
030656  005103                          COM     R3
030660  010204                          MOV     R2,R4           ;R4 WILL BE MSH 'PRODUCT'
030662  006204                          ASR     R4              ;FORM 'PRODUCT'
030664  005104                          COM     R4              ;COMPLEMENT MSH 'PRODUCT'
030666  070200                          MUL     R0,R2           ;MULTIPLY R2 BY 100000 LEAVING
                                                                ;R2 = MSH, R3 = LSH PRODUCT
030670  020204                          CMP     R2,R4           ;COMPARE MSH PRODUCTS
030672  001002                          BNE     3$
030674  020003                          CMP     R0,R3           ;CHECK LSH PRODUCT
030676  001401                          BEQ     .+4
030700  104000                  3$:     HLT
                                ;;*************************************************************
                                ;*TEST 55        CHECK THE DIV INSTRUCTION
                                ;*       THE BELOW TEST OF THE DIV INSTRUCTION DIVIDES THE CURRENT PC BY
                                ;*       1,2,4,8,ETC LEAVING THE QUOTIENT/REMAINDER IN R2/R3. NEXT THE QUOTIENT
                                ;*       IS MULTPLIED BY 1,2,4,8,ETC AND THE REMAINDER ADDED. THE RESULT IS
                                ;*       THEN COMPARED WITH THE ORIGINAL CURRENT PC.
                                ;;*************************************************************

030702  000004                  TST55:  SCOPE
030704  112737  000055  001252          MOVB    #55,@#$TSTNM            ;LOAD TEST NUMBER
030712  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
030720  012700  000001          DIV0:   MOV     #1,R0           ;R0=DIVISOR
030724  010716                          MOV     PC,(SP)         ;SAVE DATA ON STACK
030726  011603                  1$:     MOV     (SP),R3         ;GET DATA
030730  005002                          CLR     R2              ;CLEAR MSH DIVIDEND
030732  000277                          SCC
030734  071200                          DIV     R0,R2           ;DIVIDE R2 BY R0 LEAVING QUOTIENT IN R2
                                                                ;AND REMAINDER IN R3
030736  103417                          BCS     2$
030740  100416                          BMI     2$
030742  102007                          BVC     20$             ;BRANCH IF DIVIDE WORKED
030744  022700  000001                  CMP     #1,R0           ;V BIT SHOULD ONLY SET IF DIVIDING BY 1
030750  001012                          BNE     2$              ;AND THE LSH OF DIVIDEND
030752  032716  100000                  BIT     #100000,(SP)    ;IS NEGATIVE
030756  001407                          BEQ     2$
030760  000407                          BR      3$
030762  010204                  20$:    MOV     R2,R4           ;GET QUOTIENT
030764  070400                          MUL     R0,R4           ;MULTIPLY QUOTIENT BY DIVISOR
030766  060305                          ADD     R3,R5           ;ADD REMAINDER TO LSH PRODUCT
030770  103402                          BCS     2$              ;SHOULD BE NO CARRY
030772  021605                          CMP     (SP),R5         ;CHECK RESULT
030774  001401                          BEQ     .+4
030776  104000                  2$:     HLT                     ;ERROR! DIVIDE FAILED
                                                                ;QUOTIENT IS IN R2,REMAINDER IN R3
                                                                ;ORIGINAL PC IS ON STACK AND FINAL
                                                                ;PRODUCT IN R4,R5 [MSH][LSH]
031000  006300                  3$:     ASL     R0              ;GET NEXT DIVISOR
031002  102351                          BVC     1$

                                ;CHECK ASH,ASHC,MUL, AND DIV INSTRUCTIONS USING ADDRESS MODE 1
031004  005016                  ASHL1:  CLR     (SP)            ;(SP) = SHIFT COUNT
031006  005000                          CLR     R0              ;R0 = SHIFT COUNT FOR CHECK ASH
031010  012702  000020                  MOV     #16.,R2         ;R2 = MAX LEFT SHIFT COUNT
031014  005067  000012          1$:     CLR     2$              ;CLEAR CC'S HOLDING ADDRESS
031020  010703                          MOV     PC,R3           ;R3,R4 = DATA TO BE SHIFTED
```

```
10185  031022  010304                          MOV     R3,R4
10186  031024  072316                          ASH     (SP),R3         ;SHIFT R3 LEFT (SP) TIMES
10187  031026  013727  177776                  MOV     @#PSW,(PC)+     ;SAVE CC'S
10188  031032  000000                  2$:     .WORD   0               ;CONTAINS ASH (SP),R3 CC'S IN EVEN BYTE
10189                                                                  ;AND ASH R0,R4 CC'S IN ODD BYTE
10190  031034  072400                          ASH     R0,R4           ;SHIFT R4 LEFT R0 TIMES
10191  031036  113767  177776  177767          MOVB    @#PSW,2$+1      ;SAVE CC'S IN ODD BYTE OF 2$
10192  031044  020304                          CMP     R3,R4           ;COMPARE RESULTS
10193  031046  001004                          BNE     3$              ;BRANCH IF THEY DO NOT COMPARE
10194  031050  126767  177756  177755          CMPB    2$,2$+1         ;CHECK CC'S AFTER ASH INSTRUCTIONS
10195  031056  001401                          BEQ     .+4
10196  031060  104000                  3$:     HLT                     ;ERROR! EITHER RESULTS OF SHIFT OR
10197                                                                  ;RESULT CC'S ARE INCORRECT
10198  031062  005200                          INC     R0              ;INCREMENT SHIFT COUNT FOR ASH R0,R4
10199  031064  005216                          INC     (SP)            ;INCRENENT SHIFT COUNT FOR ASH (SP),R3
10200  031066  020200                          CMP     R2,R0           ;CHECK FOR MAX SHIFT COUNT
10201  031070  001351                          BNE     1$
10202
10203  031072  005016                  ASHR1:  CLR     (SP)            ;(SP) = SHIFT COUNT FOR ASH (SP),R4
10204  031074  005000                          CLR     R0              ;R0 = SHIFT COUNT FOR ASH R0,R5
10205  031076  005402                          NEG     R2              ;R2 = MAX RIGHT SHIFT COUNT (SET BY
10206                                                                  ;ABOVE TEST TO 16. NOW = -16.
10207  031100  005067  000012          1$:     CLR     2$              ;CLEAR CC'S HOLDING ADDRESS
10208  031104  010704                          MOV     PC,R4           ;R4,R5 = DATA TO BE SHIFTED RIGHT
10209  031106  010405                          MOV     R4,R5
10210  031110  072416                          ASH     (SP),R4         ;SHIFT R4 RIGHT (SP) TIMES
10211  031112  013727  177776                  MOV     @#PSW,(PC)+     ;SAVE CC'S
10212  031116  000000                  2$:     .WORD   0               ;CONTAINS ASH (SP),R4 CC'S IN EVEN BYTE
10213                                                                  ;AND ASH R0,R5 CC'S IN ODD BYTE
10214  031120  072500                          ASH     R0,R5           ;SHIFT R5 RIGHT R0 TIMES
10215  031122  113767  177776  177767          MOVB    @#PSW,2$+1      ;SAVE CC'S IN ODD BYTE 2$
10216  031130  020405                          CMP     R4,R5           ;CHECK RESULTS
10217  031132  001004                          BNE     3$
10218  031134  126767  177756  177755          CMPB    2$,2$+1         ;CHECK RESULT CC'S
10219
10220  031142  001401                          BEQ     .+4
10221  031144  104000                  3$:     HLT                     ;ERROR! EITHER RESULTS OR RESULT CC'S
10222                                                                  ;DID NOT COMPARE
10223  031146  005300                          DEC     R0              ;DECREMENT SHIFT COUNT
10224  031150  005316                          DEC     (SP)            ;DECREMENT SHIFT COUNT FOR ASH (SP),R4
10225  031152  020002                          CMP     R0,R2           ;CHECK FOR MAX RIGHT SHIFT
10226  031154  001351                          BNE     1$
10232                                  ;;*****************************************************************
  (3)                                   ;*TEST 56       DIVIDE AGAIN
  (4)                                   ;*      THE BELOW TEST CHECKS THE DIVIDE INSTRUCTION BY DIVIDING
  (4)                                   ;*      THE CURRENT PC BY ITSELF+1. THE QUOTIENT (IN R2) ALWAYS = 0,
  (4)                                   ;*      AND THE REMAINDER (IN R3) ALWAYS = THE CURRENT PC.
  (3)                                   ;;*****************************************************************
  (2)
  (2)   031156  000004          TST56:  SCOPE
  (2)   031160  112737  000056  001252          MOVB    #56,@#$TSTNM            ;LOAD TEST NUMBER
  (2)   031166  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10233  031174  010703                  DIV1:   MOV     PC,R3           ;CURRENT PC IS LSH DIVIDEND
10234  031176  006702                          SXT     R2              ;EXTEND SIGN TO R2 (MSH DIVIDEND)
10235  031200  010304                          MOV     R3,R4           ;SAVE ORIGINAL DIVIDEND
10236  031202  010316                          MOV     R3,(SP)         ;PUT ON STACK
```

```
10237  031204  005216                          INC     (SP)            ;ADD 1 (WILL BE DIVISOR)
10238  031206  100002                          BPL     1$              ;BRANCH IF POSITIVE
10239  031210  162716  000002                  SUB     #2,(SP)         ;MAKE DIVISOR 1 LESS THAN DIVIDEND
10240  031214  071216                  1$:     DIV     (SP),R2         ;DIVIDE R2 BY (SP)
10241  031216  103410                          BCS     2$              ;CHECK CONDITION CODES
10242  031220  102407                          BVS     2$
10243  031222  001006                          BNE     2$
10244  031224  100405                          BMI     2$
10245  031226  005702                          TST     R2              ;CHECK QUOTIENT (R2 = 0)
10246  031230  001361                          BNE     DIV1
10247  031232  010416                          MOV     R4,(SP)         ;GET ORIGINAL DIVISOR
10248  031234  020316                          CMP     R3,(SP)         ;CHECK REMAINDER
10249  031236  001401                          BEQ     .+4
10250  031240  104000                  2$:     HLT                     ;REPORT ERROR
10251                                  ;;*******************************************************************
  (3)                                   ;*TEST 57       CHECK SPL INSTRUCTION
  (3)                                   ;;*******************************************************************
  (2)
  (2)   031242  000004                  TST57:  SCOPE
  (2)   031244  112737  000057  001252          MOVB    #57,@#$TSTNM            ;LOAD TEST NUMBER
  (2)   031252  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10252  031260  012702                  SPLTST: MOV     (PC)+,R2        ;R2 CONTAINS OP CODE FOR SPL  7
10253  031262  000237                          SPL     7
10254  031264  005004                          CLR     R4
10255  031266  042744  000340                  BIC     #PR7,-(R4)      ;CLEAR PRIORITY LEVEL BITS IN PSW
10256  031272  011403                          MOV     (R4),R3         ;GET CURRENT PSW
10257  031274  042703  177757                  BIC     #177757,R3      ;R3 CONTAINS CORRECT PSW AFTER SPL
10258
10259  031300  012767  000230  000010          MOV     #SPL+0,2$       ;INITIALIZE SPL INSTRUCTIONS
10260  031306  012767  000237  000050          MOV     #SPL+7,5$
10261  031314  000257                  1$:     CCC                     ;CLEAR CONDITION CODES
10262  031316  000230                  2$:     SPL     0               ;SET PRIORITY LEVEL (NOTE: SPL=NOP IF USER/SUPER MODE)
10263  031320  121403                          CMPB    (R4),R3         ;CHECK RESULT OF SPL ABOVE
10264  031322  001401                          BEQ     .+4
10265  031324  104000                          HLT                     ;ERROR! SPL ABOVE FAILED
10266  031326  032714  140000                  BIT     #UM,(R4)        ;IF NOT IN KERNEL MODE THEN SPL
10267  031332  001002                          BNE     3$              ;ACTS AS A NOP
10268  031334  062703  000040                  ADD     #40,R3          ;SET NEXT CORRECT PSW RESULT
10269  031340  005267  177752          3$:     INC     2$              ;SET NEXT SPL INSTRUCTION
10270  031344  026702  177746                  CMP     2$,R2           ;CHECK IF DONE
10271  031350  002761                          BLT     1$              ;LOOP UNTIL DONE CHANGING SPL EACH PASS
10272  031352  012702                          MOV     (PC)+,R2        ;R2 CONTAINS SPL INSTRUCTION BELOW
10273  031354  000230                          SPL     0
10274  031356  052703  000017                  BIS     #17,R3          ;SET CONDITION CODE RESULT INTO R3
10275  031362  000277                  4$:     SCC                     ;SET CONDITION CODES
10276  031364  000237                  5$:     SPL     7               ;SET PRIORITY LEVEL
10277  031366  121403                          CMPB    (R4),R3         ;CHECK RESULT OF SPL ABOVE
10278  031370  001401                          BEQ     .+4
10279  031372  104000                          HLT                     ;ERROR! SPL ABOVE FAILED
10280  031374  032714  140000                  BIT     #UM,(R4)        ;CHECK IF IN KERNEL MODE
10281  031400  001002                          BNE     6$
10282  031402  162703  000040                  SUB     #40,R3          ;SET NEXT CORRECT PSW RESULT
10283  031406  005367  177752          6$:     DEC     5$              ;SET NEXT SPL
10284  031412  026702  177746                  CMP     5$,R2           ;CHECK IF DONE ALL SPL'S
10285  031416  002361                          BGE     4$
10292                                  ;;*******************************************************************
```

```
(3)                                     ;*TEST 60       CHECK PIRQ LOGIC
(4)                                     ;*      THIS TEST CHECKS THAT WHEN A REQUEST IS MADE AT A LEVEL = TO THE
(4)                                     ;*      CURRENT PROCESSER PRIORITY LEVEL THAT NO INTERRUPT TAKES PLACE, AND
(4)                                     ;*      THAT WHEN A REQUEST IS MADE AT A LEVEL 1 GREATER THAN THE CURRENT PRO-
(4)                                     ;*      CESSER LEVEL THAT AN INTERRUPT OCCURS
(3)                                     ;;*****************************************************************************
(2)
(2)  031420  000004                     TST60:  SCOPE
(2)  031422  112737  000060  001252             MOVB    #60,@#$TSTNM            ;LOAD TEST NUMBER
(2)  031430  013737  001252  177570             MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10293  031436  012700  031602           PIRQ0:  MOV     #4$,R0          ;R0 POINTS TO A TABLE OF CORRECT PIRQ
10294                                                                   ;CONTENTS AFTER AN INTERRUPT
10295  031442  012702  000400                   MOV     #400,R2         ;R2 CONTAINS INTERRUPT REQUEST LEVEL
10296  031446  005003                           CLR     R3              ;R3 CONTAINS PROCESSER PRIORITY LEVEL
10297  031450  012704  177772                   MOV     #PIRQ,R4        ;R4 CONTAINS ADDRESS OF PIRQ REGISTER
10298  031454  005014                           CLR     (R4)            ;INITIALZE REQUEST LEVEL TO 0
10299  031456  013737  177776  000242           MOV     @#PSW,@#PIRQVEC+2       ;RETAIN MODE & REG SET ON TRAP
10300  031464  112737  000340  000242           MOVB    #PR7,@#PIRQVEC+2        ;ASSUME LEVEL 7 ON INTERRUPT
10301  031472  112737  000340  000016           MOVB    #PR7,@#TBITVEC+2        ;PRIORITY LEVEL 7 ON TRAP
10302  031500  012737  031540  000240   1$:     MOV     #2$,@#PIRQVEC           ;SET PIRQ ERROR INTERRUPT VECTOR
10303  031506  063737  001604  000240           ADD     @#FACTOR,@#PIRQVEC      ;ADD RELOCATION FACTOR
10304  031514  110337  177776                   MOVB    R3,@#PSW                ;SET CP PRIORITY LEVEL
10305  031520  050214                           BIS     R2,(R4)                 ;MAKE REQUEST AT LEVEL = TO CP LEVEL
10306  031522  100436                           BMI     5$                      ;BRANCH WHEN DONE
10307  031524  062737  000002  000240           ADD     #3$-2$,@#PIRQVEC                ;SET PIRQ INTERRUPT VECTOR TO 3$
10308  031532  006302                           ASL     R2
10309  031534  050214                           BIS     R2,(R4)                 ;MAKE REQUEST AT LEVEL 1 HIGHER
10310  031536  000240                           NOP
10311  031540  104000                   2$:     HLT                             ;ERROR! EITHER AN INTERRUPT OCCURED
10312                                                                   ;WHEN RQST LEVEL = CP LEVEL (PIRQVEC)=2$
10313                                                                   ;OR INTERRUPT FAILED (PIRQVEC)=3$
10314  031542  022014                   3$:     CMP     (R0)+,(R4)              ;CHECK CONTENTS OF PIRQ REGISTER
10315  031544  001406                           BEQ     6$
10316  031546  013737  177772  001352           MOV     @#PIRQ,@#$TMP0          ;SAVE PIRQ
10317  031554  005037  177772                   CLR     @#PIRQ
10318  031560  104000                           HLT                             ;ERROR! INCORRECT PIRQ CONTENTS
10319  031562  062703  000040           6$:     ADD     #40,R3                  ;SET NEXT CP PRIORITY LEVEL
10320  031566  040214                           BIC     R2,(R4)                 ;LOWER LEVEL BY 1
10321  031570  012716  031500                   MOV     #1$,(SP)                ;ADJUST RETURN ADDRESS
10322  031574  063716  001604                   ADD     @#FACTOR,(SP)           ;TO RETURN TO 1$
10323  031600  000006                   30$:    RTT
10324
10325                                   ;TABLE OF CORRECT PIRQ REGISTER CONTENTS ON INTERRUPT
10326  031602  001042                   4$:     1042                    ;PIR1+PIA1
10327  031604  003104                           3104                    ;PIR2+PIR1+PIA2
10328  031606  007146                           7146                    ;PIR3+PIR2+PIR1+PIA3
10329  031610  017210                           17210                   ;PIR4+PIR3+PIR2+PIR1+PIA4
10330  031612  037252                           37252                   ;PIR5+PIR4+PIR3+PIR2+PIR1+PIA5
10331  031614  077314                           77314                   ;PIR6+PIR5+PIR4+PIR3+PIR2+PIR1+PIA6
10332  031616  177356                           177356                  ;PIR7+PIR6+PIR5+PIR4+PIR3+PIR2+PIR1+PIA7
10333
10334  031620  005014                   5$:     CLR     (R4)                    ;CLEAR PIRQ REGISTER
10335  031622  012737  000242  000240           MOV     #PIRQVEC+2,@#PIRQVEC    ;RESET PIRQVEC TO HALT AT PIRQVEC+2
10336  031630  005037  000242                   CLR     @#PIRQVEC+2
10337  031634  105037  177776                   CLRB    @#PSW
10338  031640  042737  000340  000016           BIC     #PR7,@#TBITVEC+2
```

```
10342                                   ;;*****************************************************************
  (3)                                   ;*TEST 61        CHECK MICRO-BREAK REGISTER
  (4)                                   ;*      THIS TEST SHIFTS A '0' BIT THRU ALL BIT POSITIONS.
  (3)                                   ;;*****************************************************************
  (2)
  (2)  031646  000004                   TST61:  SCOPE
  (2)  031650  112737  000061  001252           MOVB    #61,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  031656  013737  001252  177570           MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10343  031664  012702  177770           MBRK:   MOV     #UBREAK,R2      ;SET ADDRESS OF MICRO BREAK REGISTER
10344  031670  011246                           MOV     (R2),-(SP)      ;SAVE ORIG CONTENTS
10345  031672  012700  177776                   MOV     #177776,R0      ;SET DATA PATTERN
10346  031676  010003                   1$:     MOV     R0,R3           ;GOING TO COMPARE DATA WITH R3
10347  031700  005737  001552                   TST     @#KB11E         ;IS THIS A KB11-E OR KB11-EM PROCESSOR?
10348  031704  001002                           BNE     5$              ;BR IF IT IS
10349  031706  042703  177400                   BIC     #177400,R3      ;ONLY 8 BITS IN U-BREAK OF KB11-B/C
10350  031712  010012                   5$:     MOV     R0,(R2)         ;LOAD REGISTER WITH PATTERN
10351  031714  021203                           CMP     (R2),R3         ;AND CHECK
10352  031716  001004                           BNE     3$              ;BRANCH IF INCORRECT
10353  031720  000261                   2$:     SEC                     ;SET 'C'
10354  031722  006100                           ROL     R0              ;SHIFT DATA
10355  031724  103764                           BCS     1$
10356  031726  000402                           BR      4$
10357  031730  104000                   3$:     HLT                     ;ERROR DATA IN R0 NOT IN UBREAK REG
10358  031732  000772                           BR      2$              ;CONTINUE TEST
10359  031734  012612                   4$:     MOV     (SP)+,(R2)      ;RESTORE ORIG UBREAK CONTENTS
10360                                   ;;*****************************************************************
  (3)                                   ;*TEST 62        CHECK MFPI/MTPI INSTRUCTIONS
  (3)                                   ;;*****************************************************************
  (2)
  (2)  031736  000004                   TST62:  SCOPE
  (2)  031740  112737  000062  001252           MOVB    #62,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  031746  013737  001252  177570           MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10361  031754  032737  140000  177776   MPI:    BIT     #UM,@#PSW       ;KERNEL MODE?
10362  031762  001553                           BEQ     ENDCP           ;YES EXIT TEST
10363  031764  010746                           MOV     PC,-(SP)
10364  031766  062716  000134                   ADD     #5$-.,(SP)
10365  031772  012637  000250                   MOV     (SP)+,@#MMVEC   ;SET MEM MGMT ABORT VECTOR
10366  031776  005046                           CLR     -(SP)           ;CLEAR CHECK WORD
10367  032000  010603                           MOV     SP,R3
10368  032002  010346                           MOV     R3,-(SP)        ;PUT ADDRESS OF CHECK WORD ON THE STACK
10369  032004  105737  001601                   TSTB    @#MMON          ;CHECK IF MEM MGMT IS ENABLED
10370  032010  001417                           BEQ     1$              ;BRANCH IF OFF
10371  032012  013737  177640  177654           MOV     @#UIPAR0,@#UIPAR6       ;SET UP USER PAGE ADDR. REG.
10372  032020  012737  006006  177614           MOV     #6006,@#UIPDR6          ;SET USER PAGE DESC REG R/W UP 6 PAGES
10373  032026  013737  172240  172254           MOV     @#SIPAR0,@#SIPAR6
10374  032034  012737  006006  172214           MOV     #6006,@#SIPDR6  ;SET SUPER PAGE DESC. REG.
10375  032042  062706  140000           10$:    ADD     #140000,SP      ;SET CURRENT MODE'S STACK POINTER
10376  032046  000240                           NOP
10377  032050  010746                   1$:     MOV     PC,-(SP)
10378  032052  062716  000024                   ADD     #3$-.,(SP)
10379  032056  012637  000020                   MOV     (SP)+,@#IOTVEC  ;SET IOT TRAP VECTOR
10380
10381  032062  000004                           IOT                     ;TRAP TO 3$ BELOW
10382  032064  005266  000002                   INC     2(SP)           ;INCREMENT CHECK WORD
10383  032070  001417                           BEQ     6$
10384  032072  104000                   4$:     HLT                     ;ERROR! MFPI,MTPI FAILURE-FOR BETTER
```

```
10385  032074  000415                          BR      6$              ;ISOLATION SUGGEST RUNNING MFPI DIAG. DCKTD/E
10386  032076  000240                  3$:     NOP                     ;PSW=KERNEL MODE,PREV USER OR SUPER MODE
10387  032100  006506                          MFPI    SP              ;GET PREV. MODES STACK POINTER
10388  032102  006536                          MFPI    @(SP)+          ;GET DATA (AN ADDRESS) ON PREV MODE'S STACK
10389  032104  006576  000000                  MFPI    @(SP)           ;GET DATA (=0) FROM PREV MODES ADDRESS
10390  032110  000240                          NOP                     ;SPACE AND PUSH ONTO KERNEL STACK
10391  032112  001367                          BNE     4$              ;ERROR IF BRANCH TAKEN! SHOULD HAVE A ZERO ON THE STACK
10392  032114  005116                          COM     (SP)            ;COMPLEMENT OPERAND
10393  032116  006636                          MTPI    @(SP)+          ;POP OPERAND OFF KERNEL STACK AND MOVE
10394                                                                  ;IT TO PREV MODE'S SPACE
10395  032120  000002                          RTI                     ;RETURN TO INST FOLLOWING IOT ABOVE
10396  032122  104000                  5$:     HLT                     ;ERROR! MEMORY MANG. ABORT
10397  032124  105037  177776                  CLRB    @#PSW           ;SET PRIORITY LEVEL BACK TO 0
10398  032130  012737  064254  000250  6$:     MOV     #KTABRT,@#MMVEC ;RESTORE VECTOR
10399  032136  012737  054456  000020          MOV     #$SCOPE,@#IOTVEC
10400  032144  012706  000700                  MOV     #SUPSTK,SP      ;RESTORE STACK POINTER
10401                                  ;;**********************************************************************
  (3)                               ;*TEST 63       CHECK ILLEGAL HALT
  (3)                               ;;**********************************************************************
  (2)
  (2)   032150  000004          TST63:  SCOPE
  (2)   032152  112737  000063  001252          MOVB    #63,@#$TSTNM            ;LOAD TEST NUMBER
  (2)   032160  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10402  032166  010746                  HALT1:  MOV     PC,-(SP)        ;GET CURRENT PC
10403  032170  062716  000022                  ADD     #2$-.,(SP)
10404  032174  011637  000004                  MOV     (SP),@#ERRVEC   ;SET ERROR TRAP VECTOR TO 2$ BELOW
10405  032200  012637  000010                  MOV     (SP)+,@#RESVEC  ;LOAD RESERVED INST TRAP VECTOR (11/40)
10406  032204  000000                          HALT                    ;SHOULD TRAP TO 4 IN USER/SUPER MODE
10407  032206  104000                  1$:     HLT                     ;ERROR! HALT ABOVE FAILED IN USER/SUPER MODE
10408  032210  000404                          BR      3$
10409  032212  010716                  2$:     MOV     PC,(SP)         ;REPLACE RETURN PC WITH
10410  032214  062716  000006                  ADD     #3$-.,(SP)      ;ADDRESS OF 3$ BELOW
10411  032220  000002                          RTI                     ;RETURN (TO 3$)
10412
10413  032222  012737  064422  000004  3$:     MOV     #ERPRT,@#ERRVEC ;RESTORE ERROR TRAP VECTOR
10414  032230  012737  064350  000010          MOV     #RESERR,@#RESVEC
10415  032236  105037  177776                  CLRB    @#PSW
10416  032242  005037  177766                  CLR     @#CPUERR
10417                                  ;;**********************************************************************
  (3)                               ;*TEST 64       CHECK RESET IN SUPER/USER MODE
  (3)                               ;;**********************************************************************
  (2)
  (2)   032246  000004          TST64:  SCOPE
  (2)   032250  112737  000064  001252          MOVB    #64,@#$TSTNM            ;LOAD TEST NUMBER
  (2)   032256  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10418  032264  000277                  RESET1: SCC
10419  032266  013700  177776                  MOV     @#PSW,R0        ;GET CURRENT PSW
10420  032272  000277                          SCC
10421  032274  000005                          RESET
10422  032276  023700  177776                  CMP     @#PSW,R0        ;CHECK THAT PSW UNCHANGED BY RESET ABOVE
10423  032302  001401                          BEQ     .+4
10424  032304  104000                          HLT                     ;ERROR! RESET CLEARED MODE BITS IN PSW
10425  032306  010037  177776                  MOV     R0,@#PSW        ;RESTORE PSW (FOR ERROR)
10426  032312                          ENDCP:
  (1)   032312  000004          RELE6:  SCOPE
  (1)   032314  010702                  MOV     PC,R2
```

```
(1)  032316  062702  000012              ADD     #12,R2
(1)  032322  012707  044042              MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
(1)  032326  000000              REL66:   .WORD   0
(1)                              ;6666666666666 LAST ADDRESS OF CODE TO BE RELOCATED 66666666666
(1)
10427                            ;;*****************************************************************
(3)                              ;*TEST 65        TEST STACK LIMIT REGISTER
(3)                              ;;*****************************************************************
(2)
(2)  032330  012767  000001  147040        MOV     #1,$TIMES       ;;DO 1 ITERATION
(2)  032336  000004              TST65:   SCOPE
(2)  032340  112737  000065  001252        MOVB    #65,@#$TSTNM            ;LOAD TEST NUMBER
(2)  032346  013737  001252  177570        MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10428
(1)                                 .SBTTL  START OF SECTION 7
(1)                              ;7777777777777 FIRST ADDRESS TO BE RELOCATED 777777777
(1)  032354  010700              REL7:    MOV     PC,R0           ;GET PC
(1)  032356  005740                       TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL7
(1)  032360  010037  001610               MOV     R0,@#FRSTAD     ;SAVE
(1)  032364  010700                       MOV     PC,R0           ;GET CURRENT PC
(1)  032366  162700  032366               SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
(1)  032372  010037  001604               MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
(1)  032376  010737  001262               MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
(1)  032402  062737  000030  001262        ADD     #30,@#$LPERR    ;ADJUST
(1)  032410  013737  001262  001260        MOV     @#$LPERR,@#$LPADR
(1)  032416  105737  001600               TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
(1)  032422  001402                       BEQ     .+6
(1)  032424  000167  001302               JMP     RELE7
10429                            ;THIS TEST SHIFTS A '1' BIT THROUGH ALL BIT POSITIONS
10430  032430  012702  177774              MOV     #STKLMT,R2              ;GET ADDRESS OF STACK LIM REG
10431  032434  005022                      CLR     (R2)+           ;CLEAR STACK LIMIT REG
10432  032436  032712  000020              BIT     #20,(R2)                ;EXIT TEST IF 'T' BIT IS SET
10433  032442  001116                      BNE     101$
10434  032444  052712  000340              BIS     #340,(R2)       ;SET PRIORITY LEVEL 7 TO PREVENT
10435                                                                ;ANY INTERRUPTS FROM OCCURRING
10436  032450  012700  000400              MOV     #400,R0         ;SET CHECK DATA
10437  032454  010042           1$:        MOV     R0,-(R2)        ;MOVE TO STACK LIMIT REG
10438  032456  022200                      CMP     (R2)+,R0        ;AND CHECK RESULT
10439  032460  001401                      BEQ     2$
10440  032462  104000                      HLT                     ;ERROR! STACK LIMIT DID NOT
10441                                                                ;LOAD CORRECTLY. CORRECT RESULT
10442                                                                ;IS IN R0
10443  032464  006300           2$:        ASL     R0              ;SHIFT '1' BIT LEFT
10444  032466  103372                      BCC     1$              ;LOOP UNTIL 1 BIT SHIFTS OUT
10445  032470  005042                      CLR     -(R2)           ;CLEAR STACK LIMIT REG
10446
10447                            ;THIS TEST CHECKS THAT A PROPER 'RED' ZONE VIOLATION OCCURS, NOTE THAT
10448                            ;NO 'RED ZONE' VIOLATION WILL OCCUR IF IN USER/SUPER MODES.
10449                            ;A RED ZONE VIOLATION PUSHES THE CURRENT PSW,PC ON A STACK AT 2 AND 0
10450
10451                            ;AND TAKES THE NEXT INSTRUCTION FROM THE PC IN LOCATION4. THE INST-
10452                            ;RUCTION CAUSING THE RED ZONE VIOLATION IS 'ABORTED'.
10453  032472  010746                      MOV     PC,-(SP)        ;GET CURRENT PC
10454  032474  062716  000060              ADD     #4$-.,(SP)      ;FORM ADDRESS OF 4$ BELOW
10455  032500  012637  000004              MOV     (SP)+,@#ERRVEC  ;SET ERROR TRAP VECTOR TO 4$ BELOW
10456  032504  013737  177776  000006      MOV     @#PSW,@#ERRVEC+2        ;RETAIN CURRENT STATUS ON TRAP
```

```
10457  032512  010712                          MOV     PC,(R2)                 ;SET STACK LIMIT TO CURRENT PC
10458                                                                           ;+400
10459  032514  011206                          MOV     (R2),SP                 ;AND STACK PTR = STACK LIMIT REG
10460  032516  010603                          MOV     SP,R3                   ;SAVE STACK PTR
10461  032520  016304  000336                  MOV     336(R3),R4              ;SAVE MEMORY LOC CONTENTS
10462                                                                           ;AT 'RED ZONE' BOUNDARY
10463  032524  032737  140000  177776          BIT     #UM,@#PSW               ;BRANCH IF IN KERNEL MODE
10464  032532  001403                          BEQ     20$
10465  032534  010466  000336                  MOV     R4,336(SP)              ;SHOULD NOT CAUSE TRAP
10466  032540  000432                          BR      100$
10467
10468  032542  005066  000336          20$:    CLR     336(SP)                 ;SHOULD CAUSE 'RED ZONE' TRAP
10469  032546  012706  000700          3$:     MOV     #SUPSTK,SP              ;RESTORE THE STACK
10470  032552  104000                          HLT                             ;ERROR! FAILED TO TRAP
10471
10472  032554  032737  140000  000002  4$:     BIT     #UM,@#2                 ;CHECK IF TRAPPED WHEN IN USER
10473                                                                           ;/SUPER MODES (2 CONTAINS OLD PSW)
10474  032562  001013                          BNE     99$                     ;GO TO ERROR CALL
10475  032564  010600                          MOV     SP,R0                   ;STACK PTR SHOULD = 0
10476  032566  001011                          BNE     99$                     ;GO TO ERROR CALL IF NOT 0
10477  032570  026304  000336                  CMP     336(R3),R4              ;CHECK THAT INST WAS ABORTED
10478  032574  001006                          BNE     99$                     ;GO REPORT ERRPR
10479  032576  005012                  5$:     CLR     (R2)                    ;CLEAR STACK LIMIT REG
10480  032600  010705                          MOV     PC,R5                   ;GET CURRENT PC
10481  032602  062705  177744                  ADD     #3$-.,R5                ;FORM ADDRESS OF  3$ ABOVE
10482  032606  020516                          CMP     R5,(SP)                 ;CHECK THAT RETURN PC IS ON
10483                                                                           ;THE STACK (AT 0)
10484  032610  001406                          BEQ     100$                    ;EXIT TEST
10485
10486                                  ;ERROR
10487  032612  005012                  99$:    CLR     (R2)                    ;CLEAR STACK LIMIT REG
10488  032614  010463  000336                  MOV     R4,336(R3)              ;RESTORE MEM LOCATION
10489  032620  012706  000700                  MOV     #SUPSTK,SP              ;SET STACK PTR
10490  032624  104000                          HLT                             ;ERROR!
10491  032626  010463  000336          100$:   MOV     R4,336(R3)              ;RESTORE MEM LOCATION
10492  032632  005022                          CLR     (R2)+                   ;CLEAR STACK LIM REG
10493  032634  012706  000700                  MOV     #SUPSTK,SP              ;SET STACK PTR
10494  032640  042712  000340                  BIC     #340,(R2)               ;SET PRIORITY LEVEL BACK TO 0
10495  032644  012737  064422  000004          MOV     #ERPRT,@#ERRVEC         ;RESTORE ERROR TRAP VECTOR
10496  032652  013737  177776  000006          MOV     @#PSW,@#ERRVEC+2
10497  032660  112737  000340  000006          MOVB    #PR7,@#ERRVEC+2
10498  032666  042737  000020  000006          BIC     #BIT4,@#ERRVEC+2
10499  032674  005037  177766                  CLR     @#CPUERR                        ;CLEAR ERROR REG
10504  032700                          101$:
  (4)                               ;;*************************************************************************
  (3)                               ;*TEST 66       MEMORY MANAGEMENT REGISTER TESTS
  (4)                               ;*      PDR TEST - THIS TEST WRITES 64. RANDOM #'S INTO EACH PDR REGISTER
  (4)                               ;*      NOTE:  IF MEM MGMT IS ENABLED ONLY PDR/PAR PAIRS 4-6 ARE TESTED.
  (3)                               ;;*************************************************************************
  (2)
  (2)  032700  000004          TST66:  SCOPE
  (2)  032702  112737  000066  001252          MOVB    #66,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  032710  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10505
10506  032716  012702  033150          KTPDR:  MOV     #PDRTBL,R2      ;SET TABLE ADDRESS OF PDR'S
10507  032722  012705  100360                  MOV     #100360,R5      ;SET BIT MASK
```

```
10508  032726  012200                  1$:     MOV     (R2)+,R0        ;GET PDR ADDRESS
10509  032730  001435                          BEQ     100$            ;EXIT ON '0' TERMINATOR
10510  032732  012716  000010          2$:     MOV     #8.,(SP)        ;SET LOOP COUNT (FOR 8 REGS)
10511  032736  105737  001601                  TSTB    @#MMON          ;BRANCH IF MEM MGMT DISABLED
10512  032742  001404                          BEQ     3$
10513  032744  062700  000010                  ADD     #10,R0          ;SET R0 TO PDR4
10514  032750  012716  000003                  MOV     #3,(SP)         ;AND LIMIT TO TEST 3 PDRS
10515  032754  012703  000040          3$:     MOV     #32.,R3         ;SET DATA COUNT
10516  032760  005004                          CLR     R4              ;INITIALIZE DATA TO BE WRITTEN
10517  032762  040504                  4$:     BIC     R5,R4           ;CLEAR NON-SETTABLE BITS
10518  032764  010410                          MOV     R4,(R0)         ;WRITE INTO PDR
10519  032766  021004                          CMP     (R0),R4         ;AND CHECK DATA READ BACK
10520  032770  001013                          BNE     99$             ;GO TO ERROR CALL
10521  032772  005104                          COM     R4              ;COMPLEMENT DATA
10522  032774  040504                          BIC     R5,R4           ;CLEAR NON-SETTABLE BITS
10523  032776  010410                          MOV     R4,(R0)         ;WRITE COMPLEMENT DATA INTO PDR
10524  033000  021004                          CMP     (R0),R4         ;AND CHECK
10525  033002  001006                          BNE     99$             ;GO TO ERROR CALL
10526  033004  060104                          ADD     R1,R4           ;STEP DATA
10527  033006  077313                          SOB     R3,4$
10528  033010  005020                  5$:     CLR     (R0)+           ;STEP TO NEXT REGISTER
10529  033012  005316                          DEC     (SP)            ;DECREMENT REGISTER COUNT
10530  033014  001357                          BNE     3$
10531  033016  000743                          BR      1$              ;GET NEXT SET OF 8 REGISTERS
10532
10533  033020  104000                  99$:    HLT                     ;ERROR! INCORRECT DATA READ
10534                                                                  ;BACK FROM PDR. ADDRESS OF
10535                                                                  ;PDR IS IN R0, DATA IS IN R4
10536  033022  000772                          BR      5$              ;STEP TO NEXT REGISTER
10540  033024                          100$:
  (4)                             ;;**********************************************************************
  (3)                             ;*TEST 67       PAR TEST
  (4)                             ;*      PAR TEST - THIS TEST WRITES 64. COMPLEMENTING RANDOM #'S INTO EACH PAR.
  (3)                             ;;**********************************************************************
  (2)
  (2)  033024  000004           TST67:  SCOPE
  (2)  033026  112737  000067  001252           MOVB    #67,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  033034  013737  001252  177570           MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10541  033042  012702  033166          KTPAR:  MOV     #PARTBL,R2      ;GET TABLE ADDRESS OF PAR'S
10542  033046  005005                          CLR     R5
10543  033050  012200                  1$:     MOV     (R2)+,R0        ;GET PAR ADDRESS
10544  033052  001435                          BEQ     100$            ;EXIT ON '0' TERMINATOR
10545  033054  012716  000010          2$:     MOV     #8.,(SP)        ;SET LOOP COUNT (FOR 8 REGS.)
10546  033060  105737  001601                  TSTB    @#MMON          ;BRANCH IF MEM MGMT DISABLED
10547  033064  001404                          BEQ     3$
10548  033066  062700  000010                  ADD     #10,R0          ;SET R0 TO PAR4
10549  033072  012716  000003                  MOV     #3,(SP)         ;AND LIMIT TEST TO 3 PARS
10550  033076  012703  000040          3$:     MOV     #32.,R3         ;SET DATA COUNT
10551  033102  005004                          CLR     R4              ;INITIALIZE DATA
10552  033104  040504                  4$:     BIC     R5,R4           ;CLEAR NON-SETTABLE BITS
10553  033106  010410                          MOV     R4,(R0)         ;WRITE INTO PAR
10554  033110  021004                          CMP     (R0),R4         ;AND CHECK
10555  033112  001013                          BNE     99$             ;TAKE ERROR EXIT
10556  033114  005104                          COM     R4              ;COMPLEMENT DATA
10557  033116  040504                          BIC     R5,R4           ;CLEAR NON-SETTABLE BITS
10558  033120  010410                          MOV     R4,(R0)         ;WRITE COMPLEMENT DATA
```

```
10559  033122  021004                  CMP     (R0),R4         ;AND CHECK
10560  033124  001006                  BNE     99$             ;TAKE ERROR EXIT
10561  033126  060104                  ADD     R1,R4           ;STEP DATA
10562  033130  077313                  SOB     R3,4$           ;LOOP UNTIL FINISHED
10563
10564  033132  005020          5$:     CLR     (R0)+
10565  033134  005316                  DEC     (SP)            ;DECREMENT REGISTER COUNT
10566  033136  001357                  BNE     3$              ;BRANCH IF 8 REGS NOT DONE
10567  033140  000743                  BR      1$
10568
10569  033142  104000          99$:    HLT                     ;ERROR! INCORRECT DATA READ BACK
10570                                                          ;FROM PAR. ADDRESS OF PAR IS IN
10571                                                          ;R0, DATA IS IN R4
10572  033144  000772                  BR      5$              ;DO NEXT REGISTER
10573  033146                  100$:
  (2)  033146  000416                  BR      TST70           ;;GO TO NEXT TEST
10574                          ;TABLES FOR PDR & PAR TESTS ABOVE
10575  033150  172300          PDRTBL: .WORD   KIPDR0
10576  033152  177600                  .WORD   UIPDR0
10577  033154  172200                  .WORD   SIPDR0          ;CHANGED TO '0' IF 11/40
10578  033156  172320                  .WORD   KDPDR0
10579  033160  177620                  .WORD   UDPDR0
10580  033162  172220                  .WORD   SDPDR0
10581  033164  000000                  .WORD   0               ;TERMINATOR
10582
10583  033166  172340          PARTBL: .WORD   KIPAR0
10584  033170  177640                  .WORD   UIPAR0
10585  033172  172240                  .WORD   SIPAR0          ;CHANGED TO '0' IF 11/40
10586  033174  172360                  .WORD   KDPAR0
10587  033176  177660                  .WORD   UDPAR0
10588  033200  172260                  .WORD   SDPAR0
10589  033202  000000                  .WORD   0               ;TERMINATOR
10590
10596                          ;;*********************************************************************
  (3)                          ;*TEST 70       CHECK KT ABORT LOGIC
  (4)                          ;*      THIS TEST CHECKS KT ABORT LOGIC. TEST CREATES AN ABORT CONDITION
  (4)                          ;*      AND INSURES THAT ABORT IS TAKEN PROPERLY. NOTE: TEST IS EXECUTED ONLY
  (4)                          ;*      IF TEST IS ENTERED WITH MEM MGMT ENABLED.
  (3)                          ;;*********************************************************************
  (2)
  (2)  033204  000004          TST70:  SCOPE
  (2)  033206  112737  000070  001252  MOVB    #70,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  033214  013737  001252  177570  MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10597  033222  105737  001601          KTABT:  TSTB    @#MMON          ;BRANCH IF MEM MGMT DISABLED
10598  033226  001515                  BEQ     KTEX
10599  033230  005037  172350                  CLR     @#KIPAR4        ;SET UP MEM MGMT REGISTERS
10600  033234  005037  172310                  CLR     @#KIPDR4        ;TO ABORT IF A MEMORY
10601  033240  005037  177650                  CLR     @#UIPAR4        ;REFERENCE IS MADE TO
10602  033244  005037  177610                  CLR     @#UIPDR4        ;ADDRESSES (VIRTUAL) BETWEEN
10603  033250  005037  172250                  CLR     @#SIPAR4
10604  033254  005037  172210                  CLR     @#SIPDR4
10605  033260  013746  000250          1$:     MOV     @#MMVEC,-(SP)   ;SAVE MEM MGMT VECTOR
10606  033264  013746  000252                  MOV     @#MMVEC+2,-(SP) ;AND PRIORITY
10607  033270  010746                          MOV     PC,-(SP)        ;SET MEM MGMT
10608  033272  062716  000040                  ADD     #4$-.,(SP)      ;VECTOR TO 4$ BELOW
10609  033276  012637  000250                  MOV     (SP)+,@#MMVEC
```

```
10610  033302  013737  177776  000252          MOV     @#PSW,@#MMVEC+2
10611  033310  005000                          CLR     R0              ;CLEAR ABORT INDICATOR
10612  033312  010702                          MOV     PC,R2           ;SET R2 AND R3 NOTE:
10613  033314  012703  100000                  MOV     #100000,R3      ;THE REF VIA R3 CAUSES THE
10614  033320  014223                  2$:     MOV     -(R2),(R3)+     ;ABORT
10615  033322  005700                  3$:     TST     R0              ;BRANCH IF THE ABORT OCCURRED
10616  033324  001001                          BNE     .+4
10617  033326  104000                          HLT                     ;REPORT ERROR
10618  033330  000445                          BR      100$
10619                                  ;ABORT HERE
10620  033332  013700  177776          4$:     MOV     @#PSW,R0        ;SR0 SHOULD CONTAIN
10621  033336  000300                          SWAB    R0              ;CAUSE FOR ABORT AND
10622  033340  006200                          ASR     R0              ;ALSO WHICH SEGMENT
10623  033342  042700  177637                  BIC     #177637,R0      ;WAS IN USE WHEN ABORT
10624  033346  062700  100011                  ADD     #100011,R0      ;OCCURRED.
10625  033352  020037  177572                  CMP     R0,@#SR0
10626  033356  001025                          BNE     99$
10627  033360  012700  033320                  MOV     #2$,R0          ;GET ADDRESS OF INST
10628  033364  020037  177576                  CMP     R0,@#SR2        ;THAT ABORTED
10629  033370  001020                          BNE     99$
10630  033372  012700  000362                  MOV     #362,R0
10631  033376  120037  177574                  CMPB    R0,@#SR1        ;SR1 CONTAINS REGISTER
10632  033402  001013                          BNE     99$             ;MODIFICATIONS MADE
10633  033404  012700  000023                  MOV     #23,R0
10634  033410  120037  177575                  CMPB    R0,@#SR1+1
10635  033414  001006                          BNE     99$
10636  033416  012700  033320                  MOV     #2$,R0
10637  033422  005720                  5$:     TST     (R0)+           ;R0=ADDRESS OF INST FOLLOWING ABORT
10638  033424  020016                          CMP     R0,(SP)         ;(3$)
10639  033426  001001                          BNE     99$
10640  033430  000002                          RTI                     ;RETURN
10641                                  ;ENTER HERE ON ERROR
10642  033432  104000                  99$:    HLT                     ;REPORT ERROR
10643  033434  010716                          MOV     PC,(SP)
10644  033436  062716  177664                  ADD     #3$-.,(SP)
10645  033442  000002                          RTI                     ;RETURN
10646  033444  012637  000252          100$:   MOV     (SP)+,@#MMVEC+2 ;RESTORE ABORT VECTOR
10647  033450  012637  000250                  MOV     (SP)+,@#MMVEC   ;& PRIORITY.
10648  033454  012737  000001  177572          MOV     #1,@#SR0        ;CLEAR ERROR CONDITIONS
10652  033462                          KTEX:
  (4)                               ;;*****************************************************************
  (3)                               ;*TEST 71       MAPPING REGISTER TESTS
  (4)                               ;*      THIS TEST LOADS RANDOM #'S INTO EACH MAPPING REGISTER
  (3)                               ;;*****************************************************************
  (2)
  (2)   033462  000004          TST71:  SCOPE
  (2)   033464  112737  000071  001252  MOVB    #71,@#$TSTNM            ;LOAD TEST NUMBER
  (2)   033472  013737  001252  177570  MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10653  033500  032737  000040  172516          BIT     #BIT5,@#MMR3    ;IS MAP ON?
10654  033506  001070                          BNE     MAPTWO          ;BRANCH IF YES
10655  033510  012700  170200          MAPTST: MOV     #MAPL0,R0       ;SET ADRS OF FIRST MAP REGISTER
10656  033514  023737  000042  000046          CMP     @#42,@#46       ;ACT11 ???
10657  033522  001002                          BNE     3$
10658  033524  062700  000020                  ADD     #20,R0          ;USE ONLY MP4 AND UP...
10659  033530                          3$:
10660  033530  012706  000700                  MOV     #SUPSTK,SP      ;SETUP THE SP
```

```
10661  033534  012716  000001                  MOV     #1,(SP)          ;SET BIT MASK FOR MAPL0 <15-01>
10662  033540  012702  177700                  MOV     #177700,R2       ;AND ALSO FOR MAPH0 <21-16>
10663  033544  023737  000042  000046          CMP     @#42,@#46        ;ACT11 ???
10664  033552  001003                          BNE     1$
10665  033554  012703  000034                  MOV     #28.,R3 ;
10666  033560  000402                          BR      4$
10667  033562  012703  000040          1$:     MOV     #32.,R3          ;SET DATA COUNT
10668  033566  005005                  4$:     CLR     R5               ;SET INITIAL DATA
10669  033570  010504                  2$:     MOV     R5,R4            ;GET DATA
10670  033572  041604                          BIC     (SP),R4          ;CLEAR UNUSED BITS
10671  033574  010410                          MOV     R4,(R0)          ;LOAD DATA INTO MAPL0 <15-01>
10672  033576  021004                          CMP     (R0),R4          ;CHECK DATA
10673  033600  001032                          BNE     99$              ;BRANCH IF INCORRECT
10674  033602  005105                          COM     R5               ;COMPLEMENT TEST DATA
10675  033604  010504                          MOV     R5,R4            ;GET TEST DATA
10676  033606  041604                          BIC     (SP),R4          ;CLEAR UNUSED BITS
10677  033610  010410                          MOV     R4,(R0)          ;LOAD COMPLEMENT DATA
10678  033612  021004                          CMP     (R0),R4          ;AND CHECK
10679  033614  001024                          BNE     99$
10680  033616  005720                          TST     (R0)+            ;STEP TO NEXT REGISTER
10681  033620  010504                          MOV     R5,R4            ;GET COMPLEMENT TEST DATA
10682  033622  040204                          BIC     R2,R4            ;CLEAR UNUSED BITS
10683  033624  010410                          MOV     R4,(R0)          ;LOAD TEST DATA INTO MAPH0 <21-16>
10684  033626  021004                          CMP     (R0),R4          ;AND CHECK
10685  033630  001016                          BNE     99$
10686  033632  005105                          COM     R5               ;COMPLEMENT TEST DATA
10687  033634  010504                          MOV     R5,R4            ;GET TEST DATA
10688  033636  040204                          BIC     R2,R4            ;CLEAR UNUSED BITS
10689  033640  010410                          MOV     R4,(R0)          ;LOAD TEST DATA
10690  033642  021004                          CMP     (R0),R4          ;AND CHECK
10691  033644  001010                          BNE     99$
10692  033646  060705                          ADD     PC,R5            ;FORM NEXT TEST DATA
10693  033650  005740                          TST     -(R0)            ;RESET PTR TO REGISTER <15-01>
10694  033652  077332                          SOB     R3,2$            ;AND TEST UNTIL ALL #'S USED
10695  033654  022020                          CMP     (R0)+,(R0)+      ;STEP TO NEXT REGISTER PAIR
10696  033656  022700  170400                  CMP     #MAPL0+128.,R0   ;BRANCH IF NOT LAST PAIR
10697  033662  001337                          BNE     1$
10698  033664  000401                          BR      MAPTWO
10699
10700  033666  104000                  99$:    HLT                      ;ERROR! INCORRECT DATA READ BACK
10701                                                                   ;FROM MAP REG. ADRS OF REGISTER 15
10702                                                                   ;IN R0, GOOD DATA IS IN R4
10703  033670  005737  001602          MAPTWO: TST     @#QV             ;QV OR AUTO-ACCEPT?
10704  033674  001416                          BEQ     RELE7            ;BRANCH IF NO
10705  033676  012737  040000  170210          MOV     #40000,@#MAPL2
10706  033704  005037  170212                  CLR     @#MAPH2
10707  033710  012737  020000  170204          MOV     #20000,@#MAPL1   ;SET MAP 1 AND 2 INCASE ACT11
10708  033716  005037  170206                  CLR     @#MAPH1
10709  033722  005037  170200                  CLR     @#MAPL0
10710  033726  005037  170202                  CLR     @#MAPH0
10711  033732  000004                  RELE7:  SCOPE
  (1)  033734  010702                          MOV     PC,R2
  (1)  033736  062702  000012                  ADD     #12,R2
  (1)  033742  012707  044042                  MOV     #RELOC,PC        ;GO RELOCATE PROGRAM CODE
  (1)  033746  000000                  REL77:  .WORD   0
  (1)                                  ;7777777777777 LAST ADDRESS OF CODE TO BE RELOCATED 77777777777
```

```
  (1)
10721                                   ;;*****************************************************************
  (3)                                   ;*TEST 72       FLOATING POINT TEST 1
  (4)                                   ;*
  (4)                                   ;*       THIS TEST TAKES TWO RANDOM NUMBERS (A AND B) AND
  (4)                                   ;*       COMPARES THE RESULTS OF TWO EQUAL CALCULATIONS.
  (4)                                   ;*       EACH SECTION EVALUATES A DIFFERENT EQUATION AS DESCRIBED BELOW:
  (4)                                   ;*       SECT1   (A+B)**2=A**2+2*A*B+B**2
  (4)                                   ;*       SECT2   (A+B)*(A-B)=A**2-B**2
  (4)                                   ;*       SECT3   A/B*B=A
  (3)                                   ;;*****************************************************************
  (2)
  (2)  033750  012767  000001  145420           MOV     #1,$TIMES       ;;DO 1 ITERATION
  (2)  033756  000004                   TST72:  SCOPE
  (2)  033760  112737  000072  001252           MOVB    #72,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  033766  013737  001252  177570           MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10722  033774  012737  000001  001376           MOV     #1,@#$TIMES     ;SET ITTERATIONS TO 1
10723
  (1)                                           .SBTTL  START OF SECTION 8
  (1)                                   ;8888888888888 FIRST ADDRESS TO BE RELOCATED 888888888
  (1)  034002  010700                   REL8:   MOV     PC,R0           ;GET PC
  (1)  034004  005740                           TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL8
  (1)  034006  010037  001610                   MOV     R0,@#FRSTAD     ;SAVE
  (1)  034012  010700                           MOV     PC,R0           ;GET CURRENT PC
  (1)  034014  162700  034014                   SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
  (1)  034020  010037  001604                   MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
  (1)  034024  010737  001262                   MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
  (1)  034030  062737  000030  001262           ADD     #30,@#$LPERR    ;ADJUST
  (1)  034036  013737  001262  001260           MOV     @#$LPERR,@#$LPADR
  (1)  034044  105737  001600                   TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
  (1)  034050  001402                           BEQ     .+6
  (1)  034052  000167  002422                   JMP     RELE8
10724  034056  032737  020000  001550           BIT     #FPOPT,@#OPT.CP ;FLOATING POINT AVAILABLE?
10725  034064  001002                           BNE     100$            ;BRANCH IF YES
10726  034066  000167  002424                   JMP     REL88+2
10727  034072  004737  060666           100$:   JSR     PC,@#FLTSGL     ;GET RANDOM OPERANDS
10728  034076  170127  000000                   LDFPS   #0              ;INIT FPS
10729  034102  172537  001352                   LDF     @#$TMP0,AC1     ;LOAD A OPERAND
10730  034106  172437  001356                   LDF     @#$TMP2,AC0     ;LOAD B OPERAND
10731  034112  013737  001326  001464           MOV     @#$REG0,@#$AC1  ;SETUP EXTENDED
10732  034120  013737  001330  001462           MOV     @#$REG1,@#$AC0  ;EXPONENTS
10733  034126  004767  002210                   JSR     PC,FLTADD       ;PERFORM THE ADD
10734  034132  174100                           STF     AC1,AC0         ;SETUP AC0 TO
10735  034134  013737  001464  001462           MOV     @#$AC1,@#$AC0   ;PERFORM THE SQUARE
10736  034142  004767  002124                   JSR     PC,FLTMPY       ;DO THE MULTIPLY
10737  034146  174137  001362                   STF     AC1,@#$TMP4     ;SAVE RESULT
10738  034152  013737  001464  001332           MOV     @#$AC1,@#$REG2  ;AND SOFTWARE EXP
10739                                   ;
10740                                   ;NOW DO THE RIGHT HAND SIDE OF THE EQUATION
10741                                   ;DO THE A*A FIRST
10742  034160  013737  001326  001462           MOV     @#$REG0,@#$AC0  ;GET EXT EXPONENT
10743  034166  172437  001352                   LDF     @#$TMP0,AC0     ;LOAD OPERAND A
10744  034172  013737  001462  001464           MOV     @#$AC0,@#$AC1   ;SET OPERAND B EXT EXPONENT
10745  034200  172500                           LDF     AC0,AC1         ;LOAD B OPERAND
10746  034202  004767  002064                   JSR     PC,FLTMPY       ;EXECUTE THE MULTIPLY
10747  034206  174102                           STF     AC1,AC2         ;SAVE RESULT
```

```
10748  034210  013737  001464  001466          MOV     @#$AC1,@#$AC2
10749                                  ;
10750                                  ;NOW DO THE B*B
10751  034216  172437  001356                  LDF     @#$TMP2,AC0     ;LOAD B OPERAND
10752  034222  172500                          LDF     AC0,AC1
10753  034224  013737  001330  001462          MOV     @#$REG1,@#$AC0  ;AND EXT EXPONENT
10754  034232  013737  001462  001464          MOV     @#$AC0,@#$AC1
10755  034240  004767  002026                  JSR     PC,FLTMPY       ;DO THE MULTIPLY
10756  034244  174103                          STF     AC1,AC3         ;SAVE THE RESULT
10757  034246  013737  001464  001470          MOV     @#$AC1,@#$AC3
10758                                  ;
10759                                  ;NOW DO THE 2*B*A
10760  034254  012701  001356                  MOV     #$TMP2,R1
10761  034260  172411                          LDF     (R1),AC0        ;LOAD THE B OPERAND
10762  034262  172541                          LDF     -(R1),AC1       ;LOAD THE A OPERAND
10763  034264  013737  001330  001462          MOV     @#$REG1,@#$AC0  ;AND THE EXT EXPONENTS
10764  034272  013737  001326  001464          MOV     @#$REG0,@#$AC1
10765  034300  004767  001766                  JSR     PC,FLTMPY       ;DO THE MULTIPLY
10766  034304  172427  040000                  LDF     #^O40000,AC0    ;SETUP TO MULTIPLY BY TWO
10767  034310  012737  000002  001462          MOV     #2,@#$AC0
10768  034316  004767  001750                  JSR     PC,FLTMPY       ;DO THE MULTIPLY
10769                                  ;
10770                                  ;NOW SUM THE RESULTS
10771  034322  013737  001470  001462          MOV     @#$AC3,@#$AC0
10772  034330  172403                          LDF     AC3,AC0         ;GET RESULT OF B*B
10773  034332  004767  002004                  JSR     PC,FLTADD       ;ADD THE RESULT
10774  034336  172402                          LDF     AC2,AC0         ;GET RESULT OF A*A
10775  034340  013737  001466  001462          MOV     @#$AC2,@#$AC0
10776  034346  004767  001770                  JSR     PC,FLTADD       ;ADD THIS RESULT
10777  034352  174137  001366                  STF     AC1,@#$TMP6     ;SAVE FINAL RESULT
10778  034356  013737  001464  001334          MOV     @#$AC1,@#$REG3
10779                                  ;
10780                                  ;NOW CHECK BOTH SIDES OF THE EQUATION
10781                                  ;CALCULATE THE NUMBER OF CORRECT BITS
10782                                  ;PUT LARGEST EXPONENT OF A**2 OR B**2 IN $AC2
10783  034364  023737  001466  001470          CMP     @#$AC2,@#$AC3
10784  034372  002003                          BGE     1$              ;BRANCH IF $AC2 ALREADY HAS LARGEST
10785  034374  013737  001470  001466          MOV     @#$AC3,@#$AC2   ;$AC3 WAS LARGER
10786  034402  163737  001464  001466  1$:     SUB     @#$AC1,@#$AC2   ;NOW CALCULATE NUMBER
10787  034410  162737  000023  001466          SUB     #19.,@#$AC2     ;OF CORRECT BITS WITHIN 2
10788  034416  005437  001466                  NEG     @#$AC2          ;MAKE RESULT POSITIVE
10789  034422  172437  001362                  LDF     @#$TMP4,AC0     ;LOAD RESULT OF LEFT HAND SIDE
10790  034426  013737  001332  001462          MOV     @#$REG2,@#$AC0  ;AND EXTENDED EXPONENT
10791  034434  004767  001676                  JSR     PC,FLTSUB       ;SUBTRACT TO SEE HOW CLOSE THEY ARE
10792  034440  163737  001334  001464          SUB     @#$REG3,@#$AC1  ;GET DIFFERENCE IN EXT EXPONENTS
10793                                                                  ;ACTUAL EXP'S ARE EQUAL TO 200
10794  034446  100002                          BPL     3$              ;ENSURE RESULT IS POSITIVE
10795  034450  005437  001464                  NEG     @#$AC1
10796  034454  023737  001466  001464  3$:     CMP     @#$AC2,@#$AC1   ;ANSWERS WITHIN ALLOWABLE NUMBER?
10797  034462  003401                          BLE     SECT2           ;BRANCH IF YES
10798  034464  104014                  4$:     ERROR   14              ;RESULTS ARE WRONG
10799                                  ;;*************************************************************************
10800  034466  170127  000000          SECT2:  LDFPS   #0
10801                                  ;DO A+B
10802  034472  172537  001352                  LDF     @#$TMP0,AC1     ;LOAD A OPERAND
10803  034476  172437  001356                  LDF     @#$TMP2,AC0     ;LOAD B OPERAND
```

```
10804  034502  013737  001326  001464          MOV     @#$REG0,@#$AC1
10805  034510  013737  001330  001462          MOV     @#$REG1,@#$AC0
10806  034516  004767  001620                  JSR     PC,FLTADD       ;ADD THEM
10807  034522  174102                          STF     AC1,AC2         ;SAVE IN AC2
10808  034524  013737  001464  001466          MOV     @#$AC1,@#$AC2   ;AND EXT EXPONENT
10809                          ;NOW DO THE A-B
10810  034532  172537  001352                  LDF     @#$TMP0,AC1     ;LOAD OPERAND A
10811  034536  013737  001326  001464          MOV     @#$REG0,@#$AC1  ;AND EXT EXPONENT
10812  034544  172437  001356                  LDF     @#$TMP2,AC0     ;LOAD OPERAND B
10813  034550  013737  001330  001462          MOV     @#$REG1,@#$AC0
10814  034556  004767  001554                  JSR     PC,FLTSUB       ;SUBTRACT THEM
10815                          ;NOW DO (A+B)*(A-B)
10816  034562  172402                          LDF     AC2,AC0         ;GET RESULT OF (A+B)
10817  034564  013737  001466  001462          MOV     @#$AC2,@#$AC0
10818  034572  004767  001474                  JSR     PC,FLTMPY       ;FORM THE PRODUCT
10819  034576  174137  001362                  STF     AC1,@#$TMP4     ;SAVE RESULT
10820  034602  013737  001464  001332          MOV     @#$AC1,@#$REG2  ;AND EXT EXPONENT
10821                          ;NOW DO THE B*B
10822  034610  172437  001356                  LDF     @#$TMP2,AC0     ;LOAD OPERAND B
10823  034614  013737  001330  001462          MOV     @#$REG1,@#$AC0
10824  034622  172500                          LDF     AC0,AC1         ;B OPERAND IS IN AC0
10825  034624  013737  001462  001464          MOV     @#$AC0,@#$AC1   ;AND EXT EXPONENT
10826  034632  004767  001434                  JSR     PC,FLTMPY       ;
10827  034636  174102                          STF     AC1,AC2         ;SAVE RESULT IN AC2
10828  034640  013737  001464  001466          MOV     @#$AC1,@#$AC2
10829                          ;NOW DO THE A*A
10830  034646  172437  001352                  LDF     @#$TMP0,AC0     ;LOAD OPERAND A
10831  034652  013737  001326  001462          MOV     @#$REG0,@#$AC0
10832  034660  172500                          LDF     AC0,AC1
10833  034662  013737  001462  001464          MOV     @#$AC0,@#$AC1
10834  034670  004767  001376                  JSR     PC,FLTMPY       ;EXECUTE THE MULTIPLY
10835  034674  013737  001464  001470          MOV     @#$AC1,@#$AC3   ;SAVE EXT EXPO OF A*A
10836                          ;NOW DO A**2-B**2
10837  034702  172402                          LDF     AC2,AC0         ;GET B*B
10838  034704  013737  001466  001462          MOV     @#$AC2,@#$AC0   ;A*A IN AC1
10839  034712  004767  001420                  JSR     PC,FLTSUB
10840  034716  174137  001366                  STF     AC1,@#$TMP6     ;SAVE IN MEMORY
10841  034722  013737  001464  001334          MOV     @#$AC1,@#$REG3
10842                          ;NOW COMPUTE THE RESULTS
10843                          ;CALCULATE THE NUMBER OF CORRECT BITS
10844  034730  023737  001466  001470          CMP     @#$AC2,@#$AC3   ;DETERMINE WHICH EXP IS LARGER
10845  034736  002003                          BGE     2$              ;BRANCH IF AC2 LARGER
10846  034740  013737  001470  001466          MOV     @#$AC3,@#$AC2   ;PUT LARGEST IN AC2
10847  034746  163737  001464  001466  2$:     SUB     @#$AC1,@#$AC2
10848  034754  162737  000025  001466          SUB     #21.,@#$AC2
10849  034762  005437  001466                  NEG     @#$AC2
10850  034766  172437  001362                  LDF     @#$TMP4,AC0     ;GET LEFT HAND SIDE
10851  034772  013737  001332  001462          MOV     @#$REG2,@#$AC0
10852  035000  004767  001332                  JSR     PC,FLTSUB       ;SUBTRACT TO SEE HOW CLOSE THEY ARE
10853  035004  163737  001334  001464          SUB     @#$REG3,@#$AC1  ;SUB EXT EXPONENTS
10854                                          ;ACTUAL EXPONENTS ARE EQUAL
10855  035012  100002                          BPL     1$              ;MAKE SURE RESULT IS POSITIVE
10856  035014  005437  001464                  NEG     @#$AC1
10857  035020  023737  001466  001464  1$:     CMP     @#$AC2,@#$AC1   ;RESULTS WITHIN RANGE ALLOWED?
10858  035026  003401                          BLE     SECT3           ;BRANCH IF YES
10859  035030  104014                          ERROR   14              ;RESULTS WRONG
```

```
10860
10861                                   ;;*****************************************************************
10862  035032  172537  001352           SECT3:  LDF     @#$TMP0,AC1     ;LOAD OPERAND A
10863  035036  172437  001356                   LDF     @#$TMP2,AC0     ;AND OPERAND B
10864  035042  013737  001326  001464           MOV     @#$REG0,@#$AC1
10865  035050  013737  001330  001462           MOV     @#$REG1,@#$AC0
10866  035056  004767  001232                   JSR     PC,FLTDIV       ;GO DIVIDE THEM
10867  035062  004767  001204                   JSR     PC,FLTMPY       ;MULTIPLY RESULT BY B
10868  035066  174137  001362                   STF     AC1,@#$TMP4     ;SAVE RESULT
10869  035072  013737  001464  001332           MOV     @#$AC1,@#$REG2
10870  035100  172437  001352                   LDF     @#$TMP0,AC0     ;LOAD OPERAND A
10871  035104  174037  001366                   STF     AC0,@#$TMP6     ;SAVE INCASE TYPE OUT
10872  035110  013737  001326  001462           MOV     @#$REG0,@#$AC0
10873  035116  013737  001326  001334           MOV     @#$REG0,@#$REG3
10874  035124  004767  001206                   JSR     PC,FLTSUB       ;SUBTRACT RIGHT AND LEFT HAND SIDES
10875  035130  163737  001326  001464           SUB     @#$REG0,@#$AC1  ;SEE IF RESULT OK
10876  035136  100002                   1$:     BPL     1$              ;ENSURE DIFFERANCE IS POSITIVE
10877  035140  005437  001464                   NEG     @#$AC1
10878  035144  022737  000026  001464           CMP     #22.,@#$AC1     ;RESULTS WITHIN 2 BITS?
10879  035152  003001                           BGT     2$              ;BRANCH IF NO
10880  035154  000401                           BR      TST73           ;;GO TO NEXT TEST
10881  035156  104014                   2$:     ERROR   14              ;RESULTS WRONG
10882
10883                                   ;;*****************************************************************
  (3)                                   ;*TEST 73       FLOATING POINT TEST 2
  (4)                                   ;*
  (4)                                   ;*      THIS TEST TAKES TWO RANDOM NUMBERS (A AND B) AND
  (4)                                   ;*      COMPARES THE RESULTS OF TWO EQUAL CALCULATIONS.
  (4)                                   ;*      EACH SECTION EVALUATES A DIFFERENT EQUATION AS DESCRIBED BELOW:
  (4)                                   ;*      SECT1   (A+B)**2=A**2+2*A*B+B**2
  (4)                                   ;*      SECT2   (A+B)*(A-B)=A**2-B**2
  (4)                                   ;*      SECT3   A/B*B=A
  (3)                                   ;;*****************************************************************
  (2)
  (2)  035160  000004           TST73:  SCOPE
  (2)  035162  112737  000073  001252           MOVB    #73,@#$TSTNM            ;LOAD TEST NUMBER
  (2)  035170  013737  001252  177570           MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
10884  035176  012737  000001  001376           MOV     #1,@#$TIMES
10885  035204  004737  060660           100$:   JSR     PC,@#FLTDBL     ;GET RANDOM OPERANDS
10886  035210  170127  000200                   LDFPS   #200            ;INIT FPS
10887  035214  172537  001352                   LDF     @#$TMP0,AC1     ;LOAD A OPERAND
10888  035220  172437  001362                   LDF     @#$TMP4,AC0     ;LOAD B OPERAND
10889  035224  013737  001326  001464           MOV     @#$REG0,@#$AC1  ;SETUP EXTENDED
10890  035232  013737  001330  001462           MOV     @#$REG1,@#$AC0  ;EXPONENTS
10891  035240  004767  001076                   JSR     PC,FLTADD       ;PERFORM THE ADD
10892  035244  174100                           STF     AC1,AC0         ;SETUP AC0 TO
10893  035246  013737  001464  001462           MOV     @#$AC1,@#$AC0   ;PERFORM THE SQUARE
10894  035254  004767  001012                   JSR     PC,FLTMPY       ;DO THE MULTIPLY
10895  035260  174137  001502                   STF     AC1,@#FLTMP0    ;SAVE RESULT
10896  035264  013737  001464  001332           MOV     @#$AC1,@#$REG2  ;AND SOFTWARE EXP
10897                                  ;
10898                                  ;NOW DO THE RIGHT HAND SIDE OF THE EQUATION
10899                                  ;DO THE A*A FIRST
10900  035272  013737  001326  001462           MOV     @#$REG0,@#$AC0  ;GET EXT EXPONENT
10901  035300  172437  001352                   LDF     @#$TMP0,AC0     ;LOAD OPERAND A
10902  035304  013737  001462  001464           MOV     @#$AC0,@#$AC1   ;SET OPERAND B EXT EXPONENT
```

```
10903  035312  172500                          LDF     AC0,AC1         ;LOAD B OPERAND
10904  035314  004767  000752                  JSR     PC,FLTMPY       ;EXECUTE THE MULTIPLY
10905  035320  174102                          STF     AC1,AC2         ;SAVE RESULT
10906  035322  013737  001464  001466          MOV     @#$AC1,@#$AC2
10907                                  ;
10908                                  ;NOW DO THE B*B
10909  035330  172437  001362                  LDF     @#$TMP4,AC0     ;LOAD B OPERAND
10910  035334  172500                          LDF     AC0,AC1
10911  035336  013737  001330  001462          MOV     @#$REG1,@#$AC0  ;AND EXT EXPONENT
10912  035344  013737  001462  001464          MOV     @#$AC0,@#$AC1
10913  035352  004767  000714                  JSR     PC,FLTMPY       ;DO THE MULTIPLY
10914  035356  174103                          STF     AC1,AC3         ;SAVE THE RESULT
10915  035360  013737  001464  001470          MOV     @#$AC1,@#$AC3
10916                                  ;
10917                                  ;NOW DO THE 2*B*A
10918  035366  012701  001362                  MOV     #$TMP4,R1
10919  035372  172411                          LDF     (R1),AC0        ;LOAD THE B OPERAND
10920  035374  172541                          LDF     -(R1),AC1       ;LOAD THE A OPERAND
10921  035376  013737  001330  001462          MOV     @#$REG1,@#$AC0  ;AND THE EXT EXPONENTS
10922  035404  013737  001326  001464          MOV     @#$REG0,@#$AC1
10923  035412  004767  000654                  JSR     PC,FLTMPY       ;DO THE MULTIPLY
10924  035416  172427  040000                  LDF     #^O40000,AC0    ;SETUP TO MULTIPLY BY TWO
10925  035422  012737  000002  001462          MOV     #2,@#$AC0
10926  035430  004767  000636                  JSR     PC,FLTMPY       ;DO THE MULTIPLY
10927                                  ;
10928                                  ;NOW SUM THE RESULTS
10929  035434  013737  001470  001462          MOV     @#$AC3,@#$AC0
10930  035442  172403                          LDF     AC3,AC0         ;GET RESULT OF B*B
10931  035444  004767  000672                  JSR     PC,FLTADD       ;ADD THE RESULT
10932  035450  172402                          LDF     AC2,AC0         ;GET RESULT OF A*A
10933  035452  013737  001466  001462          MOV     @#$AC2,@#$AC0
10934  035460  004767  000656                  JSR     PC,FLTADD       ;ADD THIS RESULT
10935  035464  174137  001512                  STF     AC1,@#FLTMP1    ;SAVE FINAL RESULT
10936  035470  013737  001464  001334          MOV     @#$AC1,@#$REG3
10937                                  ;
10938                                  ;NOW CHECK BOTH SIDES OF THE EQUATION
10939                                  ;CALCULATE THE NUMBER OF CORRECT BITS
10940                                  ;PUT LARGEST EXPONENT OF A**2 OR B**2 IN $AC2
10941  035476  023737  001466  001470          CMP     @#$AC2,@#$AC3
10942  035504  002003                          BGE     1$              ;BRANCH IF $AC2 ALREADY HAS LARGEST
10943  035506  013737  001470  001466          MOV     @#$AC3,@#$AC2   ;$AC3 WAS LARGER
10944  035514  163737  001464  001466  1$:     SUB     @#$AC1,@#$AC2   ;NOW CALCULATE NUMBER
10945  035522  162737  000064  001466          SUB     #52.,@#$AC2     ;OF CORRECT BITS WITHIN 2
10946  035530  005437  001466                  NEG     @#$AC2          ;MAKE RESULT POSITIVE
10947  035534  172437  001502                  LDF     @#FLTMP0,AC0    ;LOAD RESULT OF LEFT HAND SIDE
10948  035540  013737  001332  001462          MOV     @#$REG2,@#$AC0  ;AND EXTENDED EXPONENT
10949  035546  004767  000564                  JSR     PC,FLTSUB       ;SUBTRACT TO SEE HOW CLOSE THEY ARE
10950  035552  163737  001334  001464          SUB     @#$REG3,@#$AC1  ;GET DIFFERENCE IN EXT EXPONENTS
10951                                                                  ;ACTUAL EXP'S ARE EQUAL TO 200
10952  035560  100002                          BPL     3$              ;ENSURE RESULT IS POSITIVE
10953  035562  005437  001464                  NEG     @#$AC1
10954  035566  023737  001466  001464  3$:     CMP     @#$AC2,@#$AC1   ;ANSWERS WITHIN ALLOWABLE NUMBER?
10955  035574  003401                          BLE     SECT2D          ;BRANCH IF YES
10956  035576  104016                  4$:     ERROR   16              ;RESULTS ARE WRONG
10957                                  ;;*****************************************************************
10958  035600  170127  000200          SECT2D: LDFPS   #200
```

```
10959                                   ;DO A+B
10960  035604  172537  001352                   LDF     @#$TMP0,AC1     ;LOAD A OPERAND
10961  035610  172437  001362                   LDF     @#$TMP4,AC0     ;LOAD B OPERAND
10962  035614  013737  001326  001464           MOV     @#$REG0,@#$AC1
10963  035622  013737  001330  001462           MOV     @#$REG1,@#$AC0
10964  035630  004767  000506                   JSR     PC,FLTADD       ;ADD THEM
10965  035634  174102                           STF     AC1,AC2         ;SAVE IN AC2
10966  035636  013737  001464  001466           MOV     @#$AC1,@#$AC2   ;AND EXT EXPONENT
10967                                   ;NOW DO THE A-B
10968  035644  172537  001352                   LDF     @#$TMP0,AC1     ;LOAD OPERAND A
10969  035650  013737  001326  001464           MOV     @#$REG0,@#$AC1  ;AND EXT EXPONENT
10970  035656  172437  001362                   LDF     @#$TMP4,AC0     ;LOAD OPERAND B
10971  035662  013737  001330  001462           MOV     @#$REG1,@#$AC0
10972  035670  004767  000442                   JSR     PC,FLTSUB       ;SUBTRACT THEM
10973                                   ;NOW DO (A+B)*(A-B)
10974  035674  172402                           LDF     AC2,AC0         ;GET RESULT OF (A+B)
10975  035676  013737  001466  001462           MOV     @#$AC2,@#$AC0
10976  035704  004767  000362                   JSR     PC,FLTMPY       ;FORM THE PRODUCT
10977  035710  174137  001502                   STF     AC1,@#FLTMP0    ;SAVE RESULT
10978  035714  013737  001464  001332           MOV     @#$AC1,@#$REG2  ;AND EXT EXPONENT
10979                                   ;NOW DO THE B*B
10980  035722  172437  001362                   LDF     @#$TMP4,AC0     ;LOAD OPERAND B
10981  035726  013737  001330  001462           MOV     @#$REG1,@#$AC0
10982  035734  172500                           LDF     AC0,AC1
10983  035736  013737  001462  001464           MOV     @#$AC0,@#$AC1   ;AND EXT EXPONENT
10984  035744  004767  000322                   JSR     PC,FLTMPY       ;
10985  035750  174102                           STF     AC1,AC2         ;SAVE RESULT IN AC2
10986  035752  013737  001464  001466           MOV     @#$AC1,@#$AC2
10987                                   ;NOW DO THE A*A
10988  035760  172437  001352                   LDF     @#$TMP0,AC0     ;LOAD OPERAND A
10989  035764  013737  001326  001462           MOV     @#$REG0,@#$AC0
10990  035772  172500                           LDF     AC0,AC1
10991  035774  013737  001462  001464           MOV     @#$AC0,@#$AC1
10992  036002  004767  000264                   JSR     PC,FLTMPY       ;EXECUTE THE MULTIPLY
10993  036006  013737  001464  001470           MOV     @#$AC1,@#$AC3   ;SAVE EXT EXPO OF A*A
10994                                   ;NOW DO A**2-B**2
10995  036014  172402                           LDF     AC2,AC0         ;GET B*B
10996  036016  013737  001466  001462           MOV     @#$AC2,@#$AC0   ;A*A IN AC1
10997  036024  004767  000306                   JSR     PC,FLTSUB
10998  036030  174137  001512                   STF     AC1,@#FLTMP1    ;SAVE IN MEMORY
10999  036034  013737  001464  001334           MOV     @#$AC1,@#$REG3
11000                                   ;NOW COMPUTE THE RESULTS
11001                                   ;CALCULATE THE NUMBER OF CORRECT BITS
11002  036042  023737  001466  001470           CMP     @#$AC2,@#$AC3   ;DETERMINE WHICH EXP IS LARGER
11003  036050  002003                           BGE     2$              ;BRANCH IF AC2 LARGER
11004  036052  013737  001470  001466           MOV     @#$AC3,@#$AC2   ;PUT LARGEST IN AC2
11005  036060  163737  001464  001466  2$:      SUB     @#$AC1,@#$AC2
11006  036066  162737  000065  001466           SUB     #53.,@#$AC2
11007  036074  005437  001466                   NEG     @#$AC2
11008  036100  172437  001502                   LDF     @#FLTMP0,AC0    ;GET LEFT HAND SIDE
11009  036104  013737  001332  001462           MOV     @#$REG2,@#$AC0
11010  036112  004767  000220                   JSR     PC,FLTSUB       ;SUBTRACT TO SEE HOW CLOSE THEY ARE
11011  036116  163737  001334  001464           SUB     @#$REG3,@#$AC1  ;SUB EXT EXPONENTS
11012                                           ;ACTUAL EXPONENTS ARE EQUAL
11013  036124  100002                           BPL     1$              ;MAKE SURE RESULT IS POSITIVE
11014  036126  005437  001464                   NEG     @#$AC1
```

```
11015  036132  023737  001466  001464  1$:     CMP     @#$AC2,@#$AC1   ;RESULTS WITHIN RANGE ALLOWED?
11016  036140  003401                          BLE     SECT3D          ;BRANCH IF YES
11017  036142  104016                          ERROR   16              ;RESULTS WRONG
11018
11019                                  ;;*****************************************************************
11020  036144  172537  001352          SECT3D: LDF     @#$TMP0,AC1     ;LOAD OPERAND A
11021  036150  172437  001362                  LDF     @#$TMP4,AC0     ;AND OPERAND B
11022  036154  013737  001326  001464          MOV     @#$REG0,@#$AC1
11023  036162  013737  001330  001462          MOV     @#$REG1,@#$AC0
11024  036170  004767  000120                  JSR     PC,FLTDIV       ;GO DIVIDE THEM
11025  036174  004767  000072                  JSR     PC,FLTMPY       ;MULTIPLY RESULT BY B
11026  036200  174137  001502                  STF     AC1,@#FLTMP0    ;SAVE RESULT
11027  036204  013737  001464  001332          MOV     @#$AC1,@#$REG2
11028  036212  172437  001352                  LDF     @#$TMP0,AC0     ;LOAD OPERAND A
11029  036216  174037  001512                  STF     AC0,@#FLTMP1    ;SAVE INCASE TYPE OUT
11030  036222  013737  001326  001462          MOV     @#$REG0,@#$AC0
11031  036230  013737  001326  001334          MOV     @#$REG0,@#$REG3
11032  036236  004767  000074                  JSR     PC,FLTSUB       ;SUBTRACT RIGHT AND LEFT HAND SIDES
11033  036242  163737  001326  001464          SUB     @#$REG0,@#$AC1  ;SEE IF RESULT OK
11034  036250  100002                          BPL     1$              ;ENSURE DIFFERANCE IS POSITIVE
11035  036252  005437  001464                  NEG     @#$AC1
11036  036256  022737  000066  001464  1$:     CMP     #54.,@#$AC1     ;RESULTS WITHIN 2 BITS?
11037  036264  003505                          BLE     RELE8           ;BRANCH IF YES
11038  036266  104016                          ERROR   16              ;RESULTS WRONG
11039  036270  000503                          BR      RELE8
11040
11041                                  ;;*****************************************************************
11042                                  .SBTTL  FLOATING POINT MULTIPLY ROUTINE
11043                                  ;*      THIS ROUTINE MULTIPLIES THE CONTENTS OF AC0 AND AC1
11044                                  ;*      AND LEAVES THE RESULT IN AC1. IT ALSO TAKES CARE OF
11045                                  ;*      THE SOFTWARE EXPONENTS THAT ARE KEPT IN $AC0 AND $AC1.
11046                                  ;;*****************************************************************
11047  036272  063737  001462  001464  FLTMPY: ADD     @#$AC0,@#$AC1   ;ADD SOFTWARE EXPONENTS
11048  036300  171100                          MULF    AC0,AC1         ;DO THE MULTIPLY
11049  036302  012746  100400                  MOV     #100400,-(SP)   ;PUT CONTROL WORD ON STACK
11050  036306  004737  061020                  JSR     PC,@#EXPEXT     ;CALCULATE EXT EXPONENT
11051  036312  000207                  1$:     RTS     PC              ;RETURN
11052
11053                                  ;;*****************************************************************
11054                                  .SBTTL  FLOATING POINT DIVIDE ROUTINE
11055                                  ;*      THIS ROUTINE DIVIDES THE CONTENTS OF AC1 BY AC0
11056                                  ;*      AND LEAVES THE RESULT IN AC1.
11057                                  ;;*****************************************************************
11058  036314  163737  001462  001464  FLTDIV: SUB     @#$AC0,@#$AC1   ;ADJUST SOFTWARE EXPONENTS
11059  036322  174500                          DIVF    AC0,AC1         ;EXECUTE THE DIVIDE
11060  036324  012746  100400                  MOV     #100400,-(SP)   ;PUT CONTROL WORD ON STACK
11061  036330  004737  061020                  JSR     PC,@#EXPEXT     ;CALCULATE EXT EXPONENT
11062  036334  000207                  1$:     RTS     PC              ;RETURN
11063
11064                                  ;;*****************************************************************
11065                                  .SBTTL  FLOATING POINT ADD ROUTINE
11066                                  ;*      THIS ROUTINE ADDS THE CONTENTS OF AC0 TO AC1.
11067                                  ;*      THIS CAN ONLY BE DONE IF THE SOFTWARE EXPONENTS
11068                                  ;*      ARE CLOSE ENOUGH TOGETHER SUCH THAT AN ADJUSTMENT
11069                                  ;*      OF THE REAL EXPONENT LEAVES A NON-ZERO NUMBER.
11070                                  ;;*****************************************************************
```

```
11071  036336  010667  000134          FLTSUB: MOV     SP,SUBFLG       ;SET SUBTRACT FLAG
11072  036342  023737  001462  001464  FLTADD: CMP     @#$AC0,@#$AC1   ;CHECK SOFTWARE EXPONENTS
11073  036350  003016                          BGT     1$
11074  036352  001434                          BEQ     2$
11075                                  ;ACCUMULATOR 1 IS LARGER THAN ACCUMULATOR 0
11076  036354  013702  001464                  MOV     @#$AC1,R2       ;GET OPERAND B SOFTWARE EXP
11077  036360  163702  001462                  SUB     @#$AC0,R2       ;GET DIFFERENCE IN SOFTWARE EXP'S
11078  036364  020227  000071                  CMP     R2,#57.         ;EXP WITHIN DBL PREC RANGE?
11079  036370  002003                          BGE     7$              ;BRANCH IF ADD NOT REQUIRED
11080                                                                  ;RESULT IS OPERAND B
11081  036372  005402                          NEG     R2
11082  036374  176402                          LDEXP   R2,AC0          ;RELOAD THE EXPONENT
11083  036376  000422                          BR      2$
11084  036400  176427  177703          7$:     LDEXP   #-75,AC0        ;FAKE EXPONENT SO HARDWARE
11085  036404  000417                          BR      2$              ;WILL DETECT OUT OF RANGE
11086                                  ;
11087                                  ;ACCUMULATOR 0 IS LARGER THAN ACCUMULATOR 1
11088  036406  013702  001462          1$:     MOV     @#$AC0,R2       ;GET SOFTWARE EXP OF OPERAND A
11089  036412  163702  001464                  SUB     @#$AC1,R2       ;GET DIFFERENCE IN EXP'S
11090  036416  013737  001462  001464          MOV     @#$AC0,@#$AC1   ;MAKE SOFTWARE EXP'S EQUAL
11091  036424  020227  000071                  CMP     R2,#57. ;EXP WITHIN DBL PREC RANGE?
11092  036430  002003                          BGE     4$              ;BRANCH IF NO
11093  036432  005402                          NEG     R2
11094  036434  176502                          LDEXP   R2,AC1          ;RELOAD THE EXPONENT
11095  036436  000402                          BR      2$
11096                                  ;
11097                                  ;ACCUMULATOR 0 IS MUCH LARGER THAN ACCUMULATOR 1 SO RESULT IS 0
11098  036440  176527  177703          4$:     LDEXP   #-75,AC1        ;FAKE EXPONENT SO HARDWARE
11099                                                                  ;WILL DETECT OUT OF RANGE
11100  036444  005767  000026          2$:     TST     SUBFLG          ;ADD OR SUBTRACT?
11101  036450  001402                          BEQ     5$              ;BRANCH IF ADD
11102  036452  173100                          SUBF    AC0,AC1
11103  036454  000401                          BR      6$
11104  036456  172100                  5$:     ADDF    AC0,AC1         ;EXECUTE THE ADD
11105  036460  012746  100400          6$:     MOV     #100400,-(SP)   ;PUT CONTROL WORD ON STACK
11106  036464  004737  061020                  JSR     PC,@#EXPEXT     ;CALCULATE EXT EXPONENT
11107  036470  005067  000002          3$:     CLR     SUBFLG          ;INIT SUBTRACT FLAG
11108  036474  000207                          RTS     PC              ;RETURN
11109  036476  000000                  SUBFLG: .WORD
11110  036500  000004                  RELE8:  SCOPE
  (1)  036502  010702                          MOV     PC,R2
  (1)  036504  062702  000012                  ADD     #12,R2
  (1)  036510  012707  044042                  MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
  (1)  036514  000000                  REL88:  .WORD   0
  (1)                                  ;8888888888888 LAST ADDRESS OF CODE TO BE RELOCATED 88888888888
  (1)
11111
11112
11113  036516  005737  001552                  TST     @#KB11E         ;IS THIS A KB11-E OR KB11-EM?
11114  036522  001002                          BNE     DOMFPT          ;BR IF IT IS
11115  036524  000167  004060                  JMP     ENDCIS          ;SKIP THIS TEST IF NOT. MFPT AND CIS NOT THERE
11116  036530                          DOMFPT:
11126                                  ;;*****************************************************************
  (3)                                  ;*TEST 74        CHECK MFPT INSTRUCTION (KB11-E/EM ONLY)
  (4)                                  ;*      THE MFPT INSTRUCTION IS NOT AVAILABLE ON THE KB11-B/C/CM BUT
  (4)                                  ;*      IF THIS IS A KB11-E/EM THIS TEST IS RUN.  MFPT RETURNS
```

```
   (4)                                  ;*      DATA TO R0 IN THE FOLLOWING FORMAT:
   (4)                                  ;*      BIT 0 - 1  INDICATES 11/44 CPU
   (4)                                  ;*      BIT 1 - 1 INDICATES KB11-E/EM CPU (SHOULD ALWAYS COME UP IN THIS TEST)
   (4)                                  ;*      BIT 8 - 1 INDICATES CISP PRESENT
   (4)                                  ;*      BIT 9 - 1 INDICATES FP PRESENT
   (3)                                  ;;**********************************************************************
   (2)
   (2)  036530  012767  000001  142640          MOV     #1,$TIMES       ;;DO 1 ITERATION
   (2)  036536  000004          TST74:  SCOPE
   (2)  036540  112737  000074  001252          MOVB    #74,@#$TSTNM            ;LOAD TEST NUMBER
   (2)  036546  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
11127
   (1)                                          .SBTTL  START OF SECTION 9
   (1)                                  ;9999999999999 FIRST ADDRESS TO BE RELOCATED 999999999
   (1)  036554  010700          REL9:   MOV     PC,R0                   ;GET PC
   (1)  036556  005740                  TST     -(R0)                   ;R0 CONTAINS THE ADDRESS OF REL9
   (1)  036560  010037  001610          MOV     R0,@#FRSTAD             ;SAVE
   (1)  036564  010700                  MOV     PC,R0                   ;GET CURRENT PC
   (1)  036566  162700  036566          SUB     #.,R0                   ;SUBTRACT RELOCATION FACTOR
   (1)  036572  010037  001604          MOV     R0,@#FACTOR             ;SAVE RELOCATION FACTOR
   (1)  036576  010737  001262          MOV     PC,@#$LPERR             ;SET LOOP ADDRESS
   (1)  036602  062737  000030  001262  ADD     #30,@#$LPERR            ;ADJUST
   (1)  036610  013737  001262  001260  MOV     @#$LPERR,@#$LPADR
   (1)  036616  105737  001600          TSTB    @#NEXEC                 ;BR IF TEST CODE TO BE EXECUTED
   (1)  036622  001402                  BEQ     .+6
   (1)  036624  000167  003742          JMP     RELE9
11128   036630  012703  000002          MOV     #2,R3                   ;R3 IS DATA PATTERN. BIT 1 WILL ALWAYS BE SET
11129   036634  105737  001555          TSTB    @#CISP                  ;CISP FOUND?
11130   036640  001402                  BEQ     1$                      ;BR IF NOT
11131   036642  052703  000400          BIS     #400,R3                 ;BIT 8 SHOULD BE SET FOR CISP
11132   036646  032737  020000  001550  1$:     BIT     #20000,@#OPT.CP         ;FP FOUND?
11133   036654  001402                  BEQ     2$                      ;BR IF NOT
11134   036656  052703  001000          BIS     #1000,R3                ;BIT 9 WILL BE SET FOR FP
11135   036662  000007                  2$:     MFPT                    ;EXECUTE INSTRUCTION
11136   036664  020003                  CMP     R0,R3                   ;MATCH?
11137   036666  001405                  BEQ     DONE7                   ;DONE IF SO
11138   036670  010337  001352          MOV     R3,@#$TMP0              ;SET UP EXPECTED (GOOD) DATA
11139   036674  010037  001354          MOV     R0,@#$TMP1              ;SET UP RECIEVED (BAD) DATA
11140   036700  104017                  ERROR   17                      ;ERROR PRINTOUT
11141   036702  105737  001555  DONE7:  TSTB    @#CISP                          ;IS CISP PRESENT?
11142   036706  001002                  BNE     DOCIS                           ;BR IF IT IS
11143   036710  000167  003656          JMP     RELE9                           ;SKIP CIS TEST BUT RELOCATE SECTION FOR MFPT
11144   036714                  DOCIS:
11145                                   ;;**********************************************************************
   (3)                                  ;*TEST 75        COMMERCIAL INSTRUCTION SET TEST
   (3)                                  ;;**********************************************************************
   (2)
   (2)  036714  000004          TST75:  SCOPE
   (2)  036716  112737  000075  001252          MOVB    #75,@#$TSTNM            ;LOAD TEST NUMBER
   (2)  036724  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
11146   036732  013767  001620  002272          MOV     @#$LONUM,SAVRNL         ;SAVE SEED LOW
11147   036740  013767  001622  002266          MOV     @#$HINUM,SAVRNH         ;SAVE SEED HIGH
11148   036746  062737  000036  001262          ADD     #30.,@#$LPERR           ;ADJUST LOOP ON ERROR POINTER
11149                                           ;START OF TEST TO MODIFY ABSOLUTE ADDRESSES WHEN RELOCATING
11150   036754  016737  002252  001620          MOV     SAVRNL,@#$LONUM         ;RESTORE SEED LO
11151   036762  016737  002246  001622          MOV     SAVRNH,@#$HINUM         ;RESTORE SEED HI
```

```
11152  036770  012700  041332                    MOV     #OFFTAB,R0              ;GET OFFSET TO OFFSET TABLE
11153  036774  063700  001604                    ADD     @#FACTOR,R0             ;ADD FACTOR TO GET ADDRESS OF TABLE
11154  037000  012001                    1$:     MOV     (R0)+,R1                ;GET OFFSET OF STRING
11155  037002  001407                            BEQ     3$                      ;DONE IF OFFSET = 0
11156  037004  010102                            MOV     R1,R2                   ;SAVE R1
11157  037006  063702  001604                    ADD     @#FACTOR,R2             ;ADD FACTOR TO OFFSET
11158  037012  011112                            MOV     (R1),(R2)               ;MOVE ORIGINAL DATA TO ADDRESS
11159  037014  063712  001604                    ADD     @#FACTOR,(R2)           ;ADD FACTOR FOR ADDRESS
11160  037020  000767                            BR      1$                      ;GO MODIFY NEXT ADDRESS
11161                                            ;SET PASS COUNT AND GET RANDOM DATA
11162  037022  016700  002302            3$:     MOV     BUFFAD,R0               ;SET SRC ADDRES FOR DATA
11163  037026  016701  002276                    MOV     BUFFAD,R1               ;SET DST ADDRESS FOR DATA
11164  037032  062701  000310                    ADD     #200.,R1                ;ADJUST
11165  037036  004737  060562            4$:     JSR     PC,@#$RAND              ;GET RANDOM NUMBER
11166  037042  013710  001620                    MOV     @#$LONUM,(R0)           ;STORE NUMBER IN SOURCE ONE
11167  037046  042710  100200                    BIC     #100200,(R0)            ;MAKE NUMBER BETWEEN 0 AND 177
11168  037052  012021                            MOV     (R0)+,(R1)+             ;STORE NUMBER IN DST FOR TEST CMPC
11169  037054  020067  002164                    CMP     R0,DST.1A               ;DONE FILLING SOURCE ONE YET
11170  037060  002766                            BLT     4$                      ;NO GET NEXT RANDOM NUMBER
11171                                                                            ;YES GO TO FIRST TEST
11172                                            ;TEST CMPC INSTRUCTION  COMPARE SRC 1 TO SRC 1
11173  037062  004767  001560            COMP:   JSR     PC,SETUP                ;SET UP DESCRIPTORS
11174  037066  076144                            CMPCI                           ; COMPARE STRINGS
11175  037070  041236            SRC1:   .WORD   SRC.1D                          ;SOURCE ONE DESCRIPTOR
11176  037072  041242            DST1:   .WORD   DST.1D                          ;DST DESCRIPTOR
11177  037074  000040                    .WORD   ' '                             ;FILL WITH SPACES
11178  037076  001403                            BEQ     MOVE                    ;NO ERROR GO TO NEXT TEST
11179  037100  004767  001602                    JSR     PC,CISER                ;GET ERROR DATA
11180  037104  104020                            ERROR   20                      ;REPORT ERROR
11181                                            ;MOVE CHAR STRING
11182  037106  004767  001534            MOVE:   JSR     PC,SETUP                ;SET STRING DESCRIPTORS
11183  037112  076067                            L3D7                            ;LOAD DESC INTO REG
11184  037114  041236            SRC2:   .WORD   SRC.1D                          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11185  037116  041242            DST2:   .WORD   DST.1D                          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11186  037120  041320            CHAR1:  .WORD   CHAR
11187  037122  076030                            MOVC                            ;MOVE STRING
11188  037124  076144                            CMPCI                           ;COMPARE SRC AND DST
11189  037126  041236            SRC3:   .WORD   SRC.1D                          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11190  037130  041242            DST3:   .WORD   DST.1D                          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11191  037132  000040                    .WORD   ' '                             ;FILL WITH SPACES
11192  037134  001403                            BEQ     SCAN                    ;IF EQUAL NEXT TEST
11193  037136  004767  001544                    JSR     PC,CISER                ;GET ERROR DATA
11194  037142  104020                            ERROR   20                      ;REPORT ERROR
11195                                            ;SCAN,MOVC
11196  037144  004767  001476            SCAN:   JSR     PC,SETUP                ;SET UP DESCRIPTORS
11197  037150  112767  000001  002124            MOVB    #1,SET.1D               ;SET CHAR MASK FOR SPAN AND SCAN
11198  037156  076142            NXSCAN: SCANCI                                  ;SCAN
11199  037160  041236            SRC4:   .WORD   SRC.1D                          ;SOURCE DESC
11200  037162  041302            SET1:   .WORD   SET.1D                          ;PTR TO CHAR SET DESC
11201  037164  001003                            BNE     FNDSCN                  ;CHAR FOUND MOVE STRING
11202  037166  106367  002110                    ASLB    SET.1D                  ;NOT FOUND SHIFT MASK
11203  037172  000771                            BR      NXSCAN                  ;LOOK AGAIN
11204  037174  010067  002036            FNDSCN: MOV     R0,SRC.1D               ;MOV NEW ADDRESS TO DESC
11205  037200  010167  002034                    MOV     R1,SRC.1A               ;MOV NEW LENGTH TO DESC
11206  037204  076130                            MOVCI                           ;MOV TEXT STARTING WITH CHAR FOUND
11207  037206  041236            SRC5:   .WORD   SRC.1D                          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
```

```
11208  037210  041242                    DST4:   .WORD   DST.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11209  037212  000040                            .WORD   '               ;FILL WITH SPACES
11210  037214  076144                            CMPCI                   ;COMPARE SRC AND DST
11211  037216  041236                    SRC6:   .WORD   SRC.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11212  037220  041242                    DST5:   .WORD   DST.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11213  037222  000040                            .WORD   '               ;FILL WITH SPACES
11214  037224  001403                            BEQ     SPAN            ;STRINGS EQUAL NEXT TEST
11215  037226  004767  001454                    JSR     PC,CISER        ;NOT EQUAL GET ERROR DATA
11216  037232  104020                            ERROR   20              ;REPORT ERROR
11217                                            ;SPAN AND MOVC
11218  037234  004767  001406            SPAN:   JSR     PC,SETUP        ;SETUP DESC
11219  037240  012767  000001  002034            MOV     #1,SET.1D       ;SET MASK
11220
11221  037246  076067                    NXSPAN: L3D7                    ;LOAD DESC
11222  037250  041236                    SRC7:   .WORD   SRC.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11223  037252  041242                    DST6:   .WORD   DST.1D          ;DUMMY DESC TO GET SET.1D TO R4
11224  037254  041302                    SET2:   .WORD   SET.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11225  037256  076043                            SPANC                   ;FIND CHAR
11226  037260  001003                            BNE     FNDSPN          ;FOUND MOVE STRING
11227  037262  106367  002014                    ASLB    SET.1D          ;NOT SHIFT MASK
11228  037266  000767                            BR      NXSPAN          ;AND LOOK AGAIN
11229  037270  010067  001742            FNDSPN: MOV     R0,SRC.1D       ;GET ADDRESS
11230  037274  010167  001740                    MOV     R1,SRC.1A       ;SET LENGTH
11231  037300  076067                            L3D7                    ;LOAD DESC
11232  037302  041236                    SRC8:   .WORD   SRC.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11233  037304  041242                    DST7:   .WORD   DST.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11234  037306  041320                    CHAR2:  .WORD   CHAR
11235  037310  076030                            MOVC                    ;MOVE STING BEGINNING WITH CHAR
11236  037312  076067                            L3D7                    ;LOAD DESC
11237  037314  041236                    SRC9:   .WORD   SRC.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11238  037316  041242                    DST8:   .WORD   DST.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11239  037320  041320                    CHAR3:  .WORD   CHAR            ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11240  037322  076044                            CMPC                    ;COMPARE SRC AND DST
11241  037324  001403                            BEQ     MATCH           ;STRINGS EQUAL NEXT TEST
11242  037326  004767  001354                    JSR     PC,CISER        ;NOT EQUAL GET ERROR DATA
11243  037332  104020                            ERROR   20              ;REPORT ERROR
11244                                            ;MATCH AND MOVE REVERSE
11245  037334  004767  001306            MATCH:  JSR     PC,SETUP        ;SET UP DESC
11246  037340  076027                            L2D7                    ;LOAD DESC INTO REGISTERS
11247  037342  041236                    SRC10:  .WORD   SRC.1D          ;SOURCE POINTER
11248  037344  041246                    OBJ1:   .WORD   OBJ.1D          ;OBJECT POINTER
11249  037346  076045                            MATC                    ;MATCH STRINGS
11250  037350  010167  001664                    MOV     R1,SRC.1A       ;GET NEW SRC ADDRESS
11251  037354  012767  000031  001654            MOV     #^D25,SRC.1D    ;GET NEW SRC LENGTH
11252  037362  076067                            L3D7                    ;LOAD DESCRIPTORS
11253  037364  041236                    SRC11:  .WORD   SRC.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11254  037366  041246                    DST9:   .WORD   OBJ.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11255  037370  041320                    CHAR4:  .WORD   CHAR            ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11256  037372  076044                            CMPC                    ;COMPARE RESULTS
11257  037374  001403                            BEQ     MOVER           ;FOUND NEXT TEST
11258  037376  004767  001304                    JSR     PC,CISER        ;ERROR DATA
11259  037402  104020                            ERROR   20              ;REPORT ERROR
11260                                            ;MOVE REVERSE
11261  037404  004767  001236            MOVER:  JSR     PC,SETUP        ;SET UP DESC
11262  037410  076131                            MOVRCI                  ;MOVE REVERSE
11263  037412  041236                    SRC12:  .WORD   SRC.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
```

```
11264  037414  041242                DST10:  .WORD   DST.1D         ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11265  037416  000040                        .WORD   '               ;FILL WITH SPACES
11266  037420  076067                        L3D7                    ;LOAD DESCRIPTORS
11267  037422  041242                DST11:  .WORD   DST.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11268  037424  041242                DST12:  .WORD   DST.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11269  037426  000040                        .WORD   '
11270  037430  076031                        MOVRC                   ;MOVE REVERSE AGAIN
11271  037432  076067                        L3D7                    ;LOAD DESC FORCOMPARE
11272  037434  041236                SRC13:  .WORD   SRC.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11273  037436  041242                DST13:  .WORD   DST.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11274  037440  000040                        .WORD   '               ;FILL WITH SPACES
11275  037442  076044                        CMPC                    ;COMPARE STRINGS
11276  037444  001403                        BEQ     MOVT            ;EQUAL NEXT TEST
11277  037446  004767  001234              JSR     PC,CISER        ;GET ERROR DATA
11278  037452  104020                        ERROR   20              ;REPORT ERROR
11279                                        ;MOVE TRANSLATE
11280  037454  004767  001166        MOVT:   JSR     PC,SETUP        ;SET UP DESC
11281  037460  076132                        MOVTCI                  ;MOVE TRANSLATE
11282  037462  041236                SRC14:  .WORD   SRC.1D          ;SRC DESC PTR
11283  037464  041242                DST14:  .WORD   DST.1D          ;DEST DESC PTR
11284  037466  000040                        .WORD   '               ;FILL WITH SPACES
11285  037470  040776                TRANS1: .WORD   TRANS           ;TRANSLATE TABLE ADDRESS
11286  037472  076132                        MOVTCI                  ;MOVE TRANS AGAIN
11287  037474  041242                DST15:  .WORD   DST.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11288  037476  041242                DST16:  .WORD   DST.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11289  037500  000040                        .WORD   '               ;FILL WITH SPACES
11290  037502  040776                TRANS2: .WORD   TRANS           ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11291  037504  076144                        CMPCI                   ;COMPARE SRC AND DST
11292  037506  041236                SRC15:  .WORD   SRC.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11293  037510  041242                DST17:  .WORD   DST.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11294  037512  000040                        .WORD   '               ;FILL WITH SPACES
11295  037514  001403                        BEQ     LOCATE          ;STRINGS EQUAL NEXT TST
11296  037516  004767  001164        ERR4:   JSR     PC,CISER        ;GET ERROR DATA
11297  037522  104020                        ERROR   20              ;REPORT ERROR
11298                                        ;LOCATE AND MOVE CHARACTER
11299  037524  004767  001116        LOCATE: JSR     PC,SETUP        ;SETUP DESCIPTORS
11300  037530  076140                NXLOC:  LOCCI                   ;LOCATE CHARACTER
11301  037532  041236                SRC16:  .WORD   SRC.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11302  037534  000040                LOCCHR: .WORD   '
11303  037536  001003                        BNE     FNDLOC          ;FOUND MOVE STRING
11304  037540  105267  177770              INCB    LOCCHR          ;NOT FOUND INC CHAR FOR SEARCH
11305  037544  000771                        BR      NXLOC           ;LOOK AGAIN FOR NEW CHAR
11306  037546  010067  001464        FNDLOC: MOV     R0,SRC.1D       ;ADDRESS OF CHAR FOUND TO SCR.1
11307  037552  010167  001462                MOV     R1,SRC.1A       ;LENGTH OF STRING
11308  037556  016703  001462                MOV     DST.1A,R3       ;MOVE DST ADDRESS TO R2
11309  037562  016702  001454                MOV     DST.1D,R2       ;MOVE STRING LENGTH TO R3
11310  037566  012704  000040                MOV     #' ,R4          ;FILL CHAR
11311  037572  076030                        MOVC                    ;MOVE STRING BEGINING WITH CHAR FOUND
11312  037574  076144                        CMPCI                   ;COMPARE SOURCE AND DEST
11313  037576  041236                SRC17:  .WORD   SRC.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11314  037600  041242                DST18:  .WORD   DST.1D          ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11315  037602  000040                        .WORD   '               ;FILL WITH SPACES
11316  037604  001403                        BEQ     SKIP            ;STRINGS EQUAL NEXT TEST
11317  037606  004767  001074              JSR     PC,CISER        ;NOT EQUAL ERROR
11318  037612  104020                        ERROR   20              ;REPORT ERROR
11319                                        ;SKIP AND MOVE CHAR STRING
```

```
11320  037614  004767  001026          SKIP:   JSR     PC,SETUP                ;SETUP DESCRIPTORS
11321  037620  076141                  NXSKIP: SKPCI                           ;SKIP CHAR
11322  037622  041236                  SRC18:  .WORD   SRC.1D                  ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11323  037624  000040                  SKPCHR: .WORD   '
11324  037626  001003                          BNE     FNDSKP                  ;CHAR FOUND GO MOVE STRING
11325  037630  005267  177770                  INC     SKPCHR                  ;NOT FOUND INC CHAR
11326  037634  000771                          BR      NXSKIP                  ;LOOK AGAIN
11327  037636  010067  001374          FNDSKP: MOV     R0,SRC.1D               ;GET NEW SRC ADDRESS
11328  037642  010167  001372                  MOV     R1,SRC.1A               ;NEW SOURCE LENGTH
11329  037646  076130                          MOVCI                           ;MOVE STRING
11330  037650  041236                  SRC19:  .WORD   SRC.1D                  ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11331  037652  041242                  DST19:  .WORD   DST.1D                  ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11332  037654  000040                          .WORD   '                       ;FILL WITH SPACES
11333  037656  076067                          L3D7                            ;LOAD DESCIPTORS FOR COMPARE
11334  037660  041236                  SRC20:  .WORD   SRC.1D                  ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11335  037662  041242                  DST20:  .WORD   DST.1D                  ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11336  037664  041320                  CHAR5:  .WORD   CHAR
11337  037666  076044                          CMPC                            ;COMPARE STRINGS
11338  037670  001403                          BEQ     DECDAT                  ;EQUAL NEXT TEST
11339  037672  004767  001010                  JSR     PC,CISER                ;NOT EQUAL GET ERROR DATA
11340  037676  104020                          ERROR   20                      ;REPORT ERROR
11341                                          ;DECIMAL ARITHMETIC TESTS
11342                                          ;SETUP DECIMAL DATA
11343  037700  016700  001424          DECDAT: MOV     BUFFAD,R0               ;GET BUFFAD TO R0 FOR INDEX
11344  037704  016705  001420                  MOV     BUFFAD,R5
11345  037710  062705  000070                  ADD     #56.,R5
11346  037714  005002                  1$:     CLR     R2                      ;CLR  R2 TO ACCUMULATE NIBBLES
11347  037716  012703  041176                  MOV     #MSKTAB,R3              ;GET OFFSET OF MSKTAB
11348  037722  063703  001604                  ADD     @#FACTOR,R3             ;ADJUST ADDRESS
11349  037726  011001                          MOV     (R0),R1                 ;LOAD R1
11350  037730  005004                          CLR     R4                      ;CLR FOR COUNTER
11351  037732  042301                  2$:     BIC     (R3)+,R1                ;CLEAR OFF UNDESIRED NIBBLES
11352  037734  022301                          CMP     (R3)+,R1                ;IS NIBBLE LESS THAN 9
11353  037736  002001                          BGE     3$                      ;YES DONT SUBTRACT
11354  037740  161301                          SUB     (R3),R1                 ;GREATER THAN 9 SUB 6
11355  037742  005723                  3$:     TST     (R3)+                   ;INC R3 TWICE IF NO SUB
11356  037744  050102                          BIS     R1,R2                   ;STORE NIBBLE IN R2
11357  037746  005204                          INC     R4                      ;INC NIBBLE COUNT
11358  037750  022704  000004                  CMP     #4,R4                   ;4 NIBBLES DONE YET
11359  037754  002366                          BGE     2$                      ;NO DO AGAIN
11360  037756  010220                          MOV     R2,(R0)+                ;STORE VALID DATA IN SOURCE
11361  037760  020500                          CMP     R5,R0
11362  037762  103354                          BHIS    1$                      ;NO DO AGAIN
11363  037764  012701  041252                  MOV     #A.DSC,R1                       ;SET DATA TYPE
11364  037770  063701  001604                  ADD     @#FACTOR,R1             ;ADD FACTOR FOR ADDRESS
11365  037774  012702  041276                  MOV     #D.DSC,R2                       ;GET OFFSET OF D.DSC
11366  040000  063702  001604                  ADD     @#FACTOR,R2             ;ADJUST TO GET ADDRESS
11367  040004  042711  070000          4$:     BIC     #070000,(R1)            ;CLEAR TYPE BITS
11368  040010  052711  010000                  BIS     #10000,(R1)             ;MAKE UNSIGNED ZONED DATA
11369  040014  062701  000004                  ADD     #4,R1                   ;GET NEXT DATA TYPE SPECIFIER
11370  040020  020102                          CMP     R1,R2                   ;TEST FOR DONE
11371  040022  103770                          BLO     4$                      ;NOT DONE DO AGAIN
11372                                          ;TEST COMPARE NUMERIC
11373  040024  076152                          CMPNI                           ;COMPARE EQUAL STRINGS
11374  040026  041252                  A1:     .WORD   A.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11375  040030  041252                  A2:     .WORD   A.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
```

```
11376  040032  001403                  BEQ     NUMRIC                  ;EQUAL NEXT TEST
11377  040034  004767  000720          JSR     PC,CISER3               ;GET ERROR DATA
11378  040040  104020                  ERROR   20                      ;REPORT ERROR
11379                                  ;CALCULATE [(10A+10B)-10C]
11380
11381  040042  076156          NUMRIC: ASHNI                           ;SHIFT A
11382  040044  041252          A3:     .WORD   A.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11383  040046  041276          D1:     .WORD   D.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11384  040050  000001                  .WORD   1                       ;SHIFT COUNT
11385  040052  076156                  ASHNI                           ;SHIFT B
11386  040054  041256          B1:     .WORD   B.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11387  040056  041266          E1:     .WORD   E.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11388  040060  000001                  .WORD   1                       ;SHIFT COUNT
11389  040062  076150                  ADDNI                           ;10A+10B
11390  040064  041276          D2:     .WORD   D.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11391  040066  041266          E2:     .WORD   E.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11392  040070  041272          F1:     .WORD   F.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11393  040072  076156                  ASHNI                           ;SHIFT C
11394  040074  041262          C1:     .WORD   C.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11395  040076  041276          D3:     .WORD   D.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11396  040100  000001                  .WORD   1                       ;SHIFT COUNT
11397  040102  076151                  SUBNI                           ;10A+10B-10C
11398  040104  041276          F2:     .WORD   D.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11399  040106  041272          D4:     .WORD   F.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11400  040110  041266          E3:     .WORD   E.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11401                                  ;CALCULATE 10*[(A-C)+B]
11402  040112  076067                  L3D7                            ;LOAD DESCRIPTORS
11403  040114  041252          A4:     .WORD   A.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11404  040116  041262          C2:     .WORD   C.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11405  040120  041272          F3:     .WORD   F.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11406  040122  076051                  SUBN                            ;SUB A-C
11407  040124  076067                  L3D7                            ;LOAD DESC
11408  040126  041256          B2:     .WORD   B.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11409  040130  041272          F4:     .WORD   F.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11410  040132  041276          D5:     .WORD   D.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11411  040134  076050                  ADDN                            ;ADD A-C+B
11412  040136  076067                  L3D7                            ;LOAD DESC
11413  040140  041276          D6:     .WORD   D.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11414  040142  041272          F5:     .WORD   F.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11415  040144  041316          ONE1:   .WORD   ONE
11416  040146  076056                  ASHN                            ;MULT BY 10
11417                                  ;COMPARE RESULTS
11418  040150  076027                  L2D7                            ;LOAD DESC
11419  040152  041266          E4:     .WORD   E.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11420  040154  041272          F6:     .WORD   F.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11421  040156  076052                  CMPN                            ;COMPARE STRINGS
11422  040160  001403                  BEQ     CONNUM                  ;NEXT TEST IF EQUAL
11423  040162  004767  000554          JSR     PC,CISER2               ;GET ERROR DATA
11424  040166  104020                  ERROR   20                      ;REPORT ERROR
11425                                  ;CONVERT DATA TYPES
11426                                  ;LONG -> NUMERIC -> LONG
11427                                  ;NUMERIC -> PACKED -> NUMERIC
11428  040170  076157          CONNUM: CVTLNI                          ;CONVERT LONG TO NUMERIC
11429  040172  041266          E5:     .WORD   E.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11430  040174  041306          LONG1:  .WORD   LONG.1                  ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11431  040176  076153                  CVTNLI                          ;CONVERT NUMERIC TO LONG
```

```
11432  040200  041266                    E18:    .WORD   E.DSC                    ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11433  040202  041312                    LONG4:  .WORD   LONG.2                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11434  040204  026767  001076  001100            CMP     LONG.1,LONG.2            ;CHECK FIRST HALF #LONG WORD
11435  040212  001004                            BNE     1$                       ;NOT EQUAL ERROR
11436  040214  026767  001070  001072            CMP     LONG.1+2,LONG.2+2        ;EQUAL CHECK SECOND HALF
11437  040222  001403                            BEQ     NUMPAC                   ;EQUAL NEXT TEST
11438  040224  004767  000474            1$:     JSR     PC,CISER1                ;GET ERROR DATA
11439  040230  104020                            ERROR   20                       ;REPORT ERROR
11440                                            ;CONVERT NUM TO PACK TO NUM
11441  040232  076155                    NUMPAC: CVTNPI                           ;CONVERT NUM TO PACKED
11442  040234  041252                    A5:     .WORD   A.DSC                    ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11443  040236  041266                    E6:     .WORD   E.DSC                    ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11444  040240  076154                            CVTPNI                           ;CONVERT BACK TO NUM
11445  040242  041266                    E7:     .WORD   E.DSC                    ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11446  040244  041272                    F7:     .WORD   F.DSC                    ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11447  040246  076156                            ASHNI                            ;TRANSFER A RO E 32 BYTES
11448  040250  041252                    A6:     .WORD   A.DSC                    ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11449  040252  041266                    E8:     .WORD   E.DSC                    ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11450  040254  000000                            .WORD   0                        ;SHIFT COUNT
11451  040256  076152                            CMPNI                            ;COMPARE RESULTS
11452
11453  040260  041266                    E9:     .WORD   E.DSC                    ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11454  040262  041272                    F8:     .WORD   F.DSC                    ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11455  040264  001403                            BEQ     PACDAT                   ;EQUAL NEXT TEST
11456  040266  004767  000450                    JSR     PC,CISER2                ;GO GET ERROR DATA
11457  040272  104020                            ERROR   20                       ;REPORT ERROR
11458                                            ;PACKED DECIMAL ARITHMETIC
11459  040274  012701  041252            PACDAT: MOV     #A.DSC,R1                ;SET DATA TYPE
11460  040300  063701  001604                    ADD     @#FACTOR,R1              ;ADJUST FOR ADDRESS
11461  040304  012702  041276                    MOV     #D.DSC,R2                ;GET OFFSET TO D.DSC
11462  040310  063702  001604                    ADD     @#FACTOR,R2              ;ADJUST FOR ADDRESS
11463
11464  040314  042711  070000            1$:     BIC     #070000,(R1)             ;MAKE UNSIGNED PACKED DATA
11465  040320  052711  060000                    BIS     #060000,(R1)             ;SET TYPE BITS
11466  040324  062701  000004                    ADD     #4,R1                    ;NEXT DATA TYPE SPEC
11467  040330  020102                            CMP     R1,R2                    ;DONE YET
11468  040332  101770                            BLOS    1$                       ;NO DO AGAIN
11469  040334  146777  000666  000712            BICB    HIMASK,@A                ;CLR HI NIBBLE TO MAKE VALID PACKED STRING
11470  040342  146777  000660  000710            BICB    HIMASK,@B                ;CLR HI NIB OF B
11471  040350  146777  000652  000706            BICB    HIMASK,@C                ;CLR HI NIB OF C
11472  040356  016700  000746                    MOV     BUFFAD,R0                ;GET ADDRESS OF BUFF
11473  040362  146760  000642  000016            BICB    LOMASK,14.(R0)           ;CLEAR SIGN NIBBLE
11474  040370  156760  000633  000016            BISB    SIGN,14.(R0)             ;SET SIGN NIBBLE OF A
11475  040376  146760  000626  000043            BICB    LOMASK,35.(R0)           ;CLEAR SIGN NIBBLE
11476  040404  156760  000617  000043            BISB    SIGN,35.(R0)             ;SET SIGN
11477  040412  146760  000612  000057            BICB    LOMASK,47.(R0)           ;CLEAR SIGN NIBBLE
11478  040420  156760  000603  000057            BISB    SIGN,47.(R0)             ;SET SIGN
11479                                            ;TEST COMPARE PACKED
11480  040426  076172                    CMPPAK: CMPPI                            ;COMPARE EQUAL STRINGS
11481  040430  041252                    A7:     .WORD   A.DSC                    ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11482  040432  041252                    A8:     .WORD   A.DSC                    ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11483  040434  001401                            BEQ     PACKED                   ;EQUAL GO TEST
11484  040436  104020                            ERROR   20                       ;REPORT ERROR
11485                                            ;CALCULATE 10*[(B+C)]=10*[(B**2)-(C**2)/(B-C)]
11486  040440  076174                    PACKED: MULPI                            ;MULT A*A
11487  040442  041256                    B3:     .WORD   B.DSC                    ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
```

```
11488  040444  041256                  B4:     .WORD   B.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11489  040446  041276                  E10:    .WORD   D.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11490  040450  076174                          MULPI                           ;MULT   B*B
11491  040452  041262                  C3:     .WORD   C.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11492  040454  041262                  C4:     .WORD   C.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11493  040456  041266                  F9:     .WORD   E.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11494  040460  076171                          SUBPI                           ;SUB    E-F
11495  040462  041262                  F10:    .WORD   C.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11496  040464  041256                  E11:    .WORD   B.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11497  040466  041272                  E12:    .WORD   F.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11498  040470  001005                          BNE     NOZERO                  ;BRANCH IF RESULT NOT ZERO
11499  040472  076176                          ASHPI
11500  040474  041262                  C7:     .WORD   C.DSC
11501  040476  041256                  B7:     .WORD   B.DSC
11502  040500  000001                          .WORD   1
11503  040502  000756                          BR      PACKED                  ;DO ANOTHER CALCULATION TO GET RID OF ZERO
11504  040504  076171                  NOZERO: SUBPI                           ;SUB A-B
11505  040506  041266                  C5:     .WORD   E.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11506  040510  041276                  B5:     .WORD   D.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11507  040512  041266                  F11:    .WORD   E.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11508  040514  076067                          L3D7
11509  040516  041276                  D8:     .WORD   D.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11510  040520  041266                  E14:    .WORD   E.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11511  040522  041316                  ONE3:   .WORD   ONE                     ;SHIFT COUNT
11512  040524  076175                          DIVPI                           ;DIVIDE E/F
11513  040526  041272                  F12:    .WORD   F.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11514  040530  041266                  E13:    .WORD   E.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11515  040532  041276                  D7:     .WORD   D.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11516  040534  076076                          ASHP                            ;SHIFT  10*SCRTCH
11517                                           ;CALCULATE 10*(A+B) REGISTER MODE
11518  040536  076067                          L3D7                            ;LOAD DESCRIPTORS
11519  040540  041256                  B6:     .WORD   B.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11520  040542  041262                  C6:     .WORD   C.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11521  040544  041276                  D9:     .WORD   D.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11522  040546  076070                          ADDP                            ;ADD A+B
11523  040550  076067                          L3D7                            ;LOAD DESCRIPTORS
11524  040552  041276                  D10:    .WORD   D.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11525  040554  041272                  F13:    .WORD   F.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11526  040556  041316                  ONE2:   .WORD   ONE
11527  040560  076076                          ASHP                            ;SHIFT  10*SCRTCH
11528                                           ;COMPARE RESULTS
11529  040562  076172                          CMPPI                           ;E=F ????
11530  040564  041266                  E15:    .WORD   E.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11531  040566  041272                  F14:    .WORD   F.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11532  040570  001403                          BEQ     CONPAK
11533  040572  004767  000110                  JSR     PC,CISER                ;GET ERROR DATA
11534  040576  104020                          ERROR   20                      ;REPORT ERROR
11535                                           ;CONVERT DATA TYPES
11536                                           ;LONG -> PACKED -> LONG
11537                                           ;LONG -> NUMERIC -> NUMERIC
11538  040600  076177                  CONPAK: CVTLPI                          ;CONVERT LONG TO PACKED
11539  040602  041266                  E16:    .WORD   E.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11540  040604  041306                  LONG2:  .WORD   LONG.1                  ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11541  040606  076173                          CVTPLI                          ;CONVERT PACKED TO LONG
11542  040610  041266                  E17:    .WORD   E.DSC                   ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
11543  040612  041312                  LONG3:  .WORD   LONG.2                  ;THIS LOCATION MODIFIED WHEN TEST RELOCATES
```

```
11544  040614  026767  000466  000470          CMP     LONG.1,LONG.2           ;COMPARE RESULTS
11545  040622  001004                          BNE     1$
11546  040624  026767  000460  000462          CMP     LONG.1+2,LONG.2+2       ;COMPARE RESULTS OF SECOND WORD
11547  040632  001403                          BEQ     DONE
11548  040634  004767  000064          1$:     JSR     PC,CISER1               ;GET PC,ERROR DATA
11549  040640  104020                          ERROR   20                      ;REPORT ERROR
11550  040642  000167  001724          DONE:   JMP     RELE9                   ;GO ON TO NEXT TEST
11551
11552                                          ;SUBROUTINE-SETUP
11553                                          ;SETS UP CHAR STRING DESCRIPTORS
11554                                          ;USAGE: JSR     PC,SETUP
11555                                          ;NO ARGUMENTS
11556  040646  012767  000310  000362  SETUP:  MOV     #200.,SRC.1D            ;SET SOURCE LENGTH
11557  040654  016767  000450  000356          MOV     BUFFAD,SRC.1A           ;SET SOURCE ADDRESS
11558  040662  012767  000310  000352          MOV     #200.,DST.1D            ;DEST LENGTH
11559  040670  016767  000434  000346          MOV     BUFFAD,DST.1A           ;DEST ADDRESS
11560  040676  062767  000310  000340          ADD     #200.,DST.1A            ;ADJUST FOR DST
11561  040704  000207                          RTS     PC                      ;RETURN
11562                                          ;SUBROUTINES-CISER, CISER1, CISER2 AND CISER3
11563                                          ;GETS SHOULD AND WAS DATA AND ADDRESSES
11564                                          ;USAGE: JSR     PC,CISER(X)
11565                                          ;NO ARGUMENTS
11566  040706  016737  000326  001352  CISER:  MOV     SRC.1A,@#$TMP0          ;SHOULD BE ADDRESS
11567  040714  016737  000324  001354          MOV     DST.1A,@#$TMP1          ;WAS ADDRESS
11568  040722  000207                          RTS     PC                      ;RETURN
11569  040724  016737  000356  001352  CISER1: MOV     LONG.1P,@#$TMP0         ;SHOULD BE ADDRESS
11570  040732  016737  000354  001354          MOV     LONG.2P,@#$TMP1         ;WAS ADDRESS
11571  040740  000207                          RTS     PC                      ;RETURN
11572  040742  016737  000322  001352  CISER2: MOV     E,@#$TMP0               ;SHOULD BE ADDRESS
11573  040750  016737  000320  001354          MOV     F,@#$TMP1               ;WAS ADRRESS
11574  040756  000207                          RTS     PC                      ;RETURN
11575  040760  016737  000270  001352  CISER3: MOV     A,@#$TMP0               ;SHOULD BE ADDRESS
11576  040766  016737  000262  001354          MOV     A,@#$TMP1               ;WAS ADDRESS
11577  040774  000207                          RTS     PC                      ;RETURN
11578
11579                                          ;TRANSLATE TABLE
11580                                          ;USED BY MOVE TRANSLATE
11581                                          ;USED BY SPAN AND SCAN INSTRUCTIONS
11582                                          ;128 CHAR ASCII
11583
11584  040776      177     176     175  TRANS:  .BYTE   177,176,175,174,173,172,171,170
       041001      174     173     172
       041004      171     170
11585  041006      167     166     165          .BYTE   167,166,165,164,163,162,161,160
       041011      164     163     162
       041014      161     160
11586  041016      157     156     155          .BYTE   157,156,155,154,153,152,151,150
       041021      154     153     152
       041024      151     150
11587  041026      147     146     145          .BYTE   147,146,145,144,143,142,141,140
       041031      144     143     142
       041034      141     140
11588  041036      137     136     135          .BYTE   137,136,135,134,133,132,131,130
       041041      134     133     132
       041044      131     130
11589  041046      127     126     125          .BYTE   127,126,125,124,123,122,121,120
```

```
                041051  124     123     122
                041054  121     120
11590           041056  117     116     115             .BYTE   117,116,115,114,113,112,111,110
                041061  114     113     112
                041064  111     110
11591           041066  107     106     105             .BYTE   107,106,105,104,103,102,101,100
                041071  104     103     102
                041074  101     100
11592           041076  077     076     075             .BYTE   077,076,075,074,073,072,071,070
                041101  074     073     072
                041104  071     070
11593           041106  067     066     065             .BYTE   067,066,065,064,063,062,061,060
                041111  064     063     062
                041114  061     060
11594           041116  057     056     055             .BYTE   057,056,055,054,053,052,051,050
                041121  054     053     052
                041124  051     050
11595           041126  047     046     045             .BYTE   047,046,045,044,043,042,041,040
                041131  044     043     042             .
                041134  041     040
11596           041136  037     036     035             .BYTE   037,036,035,034,033,032,031,030
                041141  034     033     032
                041144  031     030
11597           041146  027     026     025             .BYTE   027,026,025,024,023,022,021,020
                041151  024     023     022
                041154  021     020
11598           041156  017     016     015             .BYTE   017,016,015,014,013,012,011,010
                041161  014     013     012
                041164  011     010
11599           041166  007     006     005             .BYTE   007,006,005,004,003,002,001,000
                041171  004     003     002
                041174  001     000
11600
11601                                                   ;MASK TABLES
11602                                                   ;FOR MAKING VALID DECIMAL DATA
11603
11604   041176  177760                                  MSKTAB: .WORD   177760
11605   041200  000011                                          .WORD   11
11606   041202  000006                                          .WORD   6
11607   041204  177417                                          .WORD   177417
11608   041206  000220                                  NINTAB: .WORD   220
11609   041210  000140                                          .WORD   140
11610   041212  170377                                          .WORD   170377
11611   041214  004400                                          .WORD   4400
11612   041216  003000                                  SIXTAB: .WORD   3000
11613   041220  007777                                          .WORD   7777
11614   041222  070000                                          .WORD   70000
11615   041224  100000                                          .WORD   100000
11616   041226     360                                  HIMASK: .BYTE   360
11617   041227     014                                  SIGN:   .BYTE   014
11618   041230     017                                  LOMASK: .BYTE   017
11619           041232                                  .EVEN
11620   041232  000000                                  SAVRNL: .WORD   0
11621   041234  000000                                  SAVRNH: .WORD   0
11622
11623                                                   ;CHARACTER STRING DESCRIPTOR TABLE
```

```
11624
11625  041236  000310                  SRC.1D: .WORD   200.
11626  041240  041752                  SRC.1A: .WORD   BUFF
11627  041242  000310                  DST.1D: .WORD   200.
11628  041244  042262                  DST.1A: .WORD   BUFF+200.
11629  041246  000031                  OBJ.1D: .WORD   25.
11630  041250  042034                  OBJ.1A: .WORD   BUFF+50.
11631  041252  000034                  A.DSC:  .WORD   28.
11632  041254  041752                  A:      .WORD   BUFF
11633  041256  000016                  B.DSC:  .WORD   14.
11634  041260  042006                  B:      .WORD   BUFF+28.
11635  041262  000012                  C.DSC:  .WORD   10.
11636  041264  042024                  C:      .WORD   BUFF+42.
11637  041266  000037                  E.DSC:  .WORD   31.
11638  041270  042042                  E:      .WORD   BUFF+56.
11639  041272  000037                  F.DSC:  .WORD   31.
11640  041274  042102                  F:      .WORD   BUFF+88.
11641  041276  000037                  D.DSC:  .WORD   31.
11642  041300  042142                  D:      .WORD   BUFF+120.
11643  041302  000001                  SET.1D: .WORD   1
11644  041304  040776                  TRANS4: .WORD   TRANS
11645  041306  001020  000000          LONG.1: .WORD   528.,0
11646  041312  000002                  LONG.2: .BLKW   2
11647  041316  000001                  ONE:    .WORD   1
11648  041320  000040                  CHAR:   .WORD   '
11649  041322  040776                  TRANS3: .WORD   TRANS
11650  041324  041306                  LONG1P: .WORD   LONG.1
11651  041326  041312                  LONG2P: .WORD   LONG.2
11652  041330  041752                  BUFFAD: .WORD   BUFF
11653
11654                                  ;OFFTAB CONTAINS ALL ABSOLUTE ADDRESSES TO BE MODIFIED WHEN RELOCATING
11655
11656  041332  037070  037114  037126  OFFTAB: .WORD   SRC1,SRC2,SRC3,SRC4,SRC5,SRC6,SRC7,SRC8,SRC9,SRC10
       041340  037160  037206  037216
       041346  037250  037302  037314
       041354  037342
11657  041356  037364  037412  037434          .WORD   SRC11,SRC12,SRC13,SRC14,SRC15,SRC16,SRC17,SRC18,SRC19,SRC20
       041364  037462  037506  037532
       041372  037576  037622  037650
       041400  037660
11658  041402  037072  037116  037130          .WORD   DST1,DST2,DST3,DST4,DST5,DST6,DST7,DST8,DST9,DST10
       041410  037210  037220  037252
       041416  037304  037316  037366
       041424  037414
11659  041426  037422  037424  037436          .WORD   DST11,DST12,DST13,DST14,DST15,DST16,DST17,DST18,DST19,DST20
       041434  037464  037474  037476
       041442  037510  037600  037652
       041450  037662
11660  041452  037470  037502  041322          .WORD   TRANS1,TRANS2,TRANS3,OBJ1,SET1,SET2
       041460  037344  037162  037254
11661  041466  037120  037306  037320          .WORD   CHAR1,CHAR2,CHAR3,CHAR4,CHAR5
       041474  037370  037664
11662  041500  040026  040030  040044          .WORD   A1,A2,A3,A4,A5,A6,A7,A8
       041506  040114  040234  040250
       041514  040430  040432
11663  041520  040054  040126  040442          .WORD   B1,B2,B3,B4,B5,B6,B7
```

```
                041526  040444  040510  040540
                041534  040476
11664           041536  040074  040116  040452          .WORD   C1,C2,C3,C4,C5,C6,C7
                041544  040454  040506  040542
                041552  040474
11665           041554  040046  040064  040076          .WORD   D1,D2,D3,D4,D5,D6,D7,D8,D9,D10
                041562  040106  040132  040140
                041570  040532  040516  040544
                041576  040552
11666           041600  040056  040066  040110          .WORD   E1,E2,E3,E4,E5,E6,E7,E8,E9,E10
                041606  040152  040172  040236
                041614  040242  040252  040260
                041622  040446
11667           041624  040464  040466  040530          .WORD   E11,E12,E13,E14,E15,E16,E17,E18
                041632  040520  040564  040602
                041640  040610  040200
11668           041644  040070  040104  040120          .WORD   F1,F2,F3,F4,F5,F6,F7,F8,F9,F10
                041652  040130  040142  040154
                041660  040244  040262  040456
                041666  040462
11669           041670  040512  040526  040554          .WORD   F11,F12,F13,F14
                041676  040566
11670           041700  041240  041244  041250          .WORD   SRC.1A,DST.1A,OBJ.1A
11671           041706  041254  041260  041264          .WORD   A,B,C,D,E,F,BUFFAD,TRANS4
                041714  041300  041270  041274
                041722  041330  041304
11672           041726  041306  041312  040174          .WORD   LONG.1P,LONG.2P,LONG1,LONG2,LONG3,LONG4
                041734  040604  040612  040202
11673           041742  040144  040556  040522          .WORD   ONE1,ONE2,ONE3
11674           041750  000000                          .WORD   0                       ;TABLE TERMINATER
11675
11676
11677                                                   ;BUFFER SPACE
11678                                                   ;200 WORDS LONG
11679                                                   ;USED FOR SOURCE AND DESTINATIONS
11680
11681           041752  000310                  BUFF:   .BLKW   200.
11682
11683           042572  000004                  RELE9:  SCOPE
  (1)           042574  010702                          MOV     PC,R2
  (1)           042576  062702  000012                ADD     #12,R2
  (1)           042602  012707  044042                MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
  (1)           042606  000000                  REL99:  .WORD   0
  (1)                                           ;9999999999999 LAST ADDRESS OF CODE TO BE RELOCATED 99999999999
  (1)
11684           042610                          ENDCIS:
11685                                           ;;*****************************************************************
  (3)                                           ;*TEST 76        TELETYPE AND CLOCK TESTS
  (3)                                           ;;*****************************************************************
  (2)
  (2)           042610  000240                  TST76:  NOP
  (2)           042612  112737  000076  001252          MOVB    #76,@#$TSTNM            ;LOAD TEST NUMBER
  (2)           042620  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
11686           042626  005037  001604          TTYCHK: CLR     @#FACTOR
11687           042632  012704  000100                  MOV     #100,R4         ;SET R4 = CONSTANT 100
11688           042636  032737  000400  001550          BIT     #TTOPT,@#OPT.CP ;BRANCH IF TTY
```

```
11689  042644  001002                          BNE     1$                  ;ON SYSTEM
11690  042646  000167  000220                  JMP     ARBFIN              ;JUMP IF NOT
11691  042652  132777  000200  136434  1$:     BITB    #200,@$TPS          ;CHECK IF TTY IS READY
11692  042660  001774                          BEQ     1$
11693  042662  012737  001623  001340          MOV     #NULLS-1,@#$REG5;SET ADDRESS OF ASCII STRING TO TYPE
11694  042670  106277  136420                  ASRB    @$TPS               ;SET IE BIT. SEE TPISR FOR INT SERVICE.
11695  042674  000001                          WAIT                        ;WAIT FOR INTERRUPT
11696
11697
11698  042676                          DUMMY:
11699                                  ;ROUTINE TO CHECK PRIORITY ARBITRATION LOGIC
11700                                  ;THE BELOW TEST WILL INHIBIT INTERRUPTS ON LEVEL 6 AND ABOVE (LOCKING
11701                                  ;OUT THE LINE CLOCK) AND THEN SET UP THE TTY TO INTERRUPT. NEXT THE
11702                                  ;PRIORITY LEVEL WILL BE SET TO 0 ALLOWING INTERRUPTS IN WHICH CASE
11703                                  ;THE LINE CLOCK (AT LEVEL 6) SHOULD INTERRUPT BEFORE THE TTY (AT LEVEL 4).
11704  042676  132737  000020  177776  1$:     BITB    #20,@#PSW
11705  042704  001104                          BNE     ARBEX                   ;EXIT TEST IF 'T' BIT SET
11706  042706  030477  136402          2$:     BIT     R4,@$TPS                ;WAIT FOR TTY TO BE NOT
11707  042712  001375                          BNE     2$                      ;BUSY
11708  042714  112737  000300  177776          MOVB    #300,@#PSW              ;SET PRIORITY LEVEL 6
11709  042722  150477  136366          3$:     BISB    R4,@$TPS                ;SET IE BIT
11710  042726  100375                          BPL     3$                      ;AND WAIT FOR READY
11711  042730  032737  001000  001550          BIT     #LKOPT,@#OPT.CP         ;LINE CLOCK AVAILABLE?
11712  042736  001455                          BEQ     ARBFIN                  ;BRANCH IF NO
11713  042740  012737  043044  000064          MOV     #7$,@#TPVEC             ;SET TTY VECTOR
11714  042746  012737  000340  000066          MOV     #PR7,@#TPVEC+2
11715  042754  012737  043056  000100          MOV     #8$,@#LKVEC             ;SET CLOCK VECTORS
11716  042762  012737  000340  000102          MOV     #PR7,@#LKVEC+2
11717  042770  005027                          CLR     (PC)+                   ;CLEAR CHECK WORD
11718  042772  000000                  4$:     .WORD   0
11719
11720
11721                                  ;REV E....BECAUSE OF ATIMING DELAY IN THE LINE CLOCK PORTION OF A DL11-W
11722                                  ;IT IS OCCASIONALLY POSSIBLE FOR 'READY' TO SET WITHOUT GENERATING AN IN-
11723                                  ;TERRUPT REQUEST ON THE FIRST LINE CLOCK TICK FOLLOWING PROGRAM SETTING
11724                                  ;OF 'I.E.'  REV E WAITS FOR A SECOND CLOCK TICK PRIOR TO EXECUTING TEST.
11725                                  ;PRIORITY MANIPULATION WAS ALSO MODIFIED TO LOCK OUT 'LATE' INTERRUPTS.
11726
11727
11728  042774  012700  000002                  MOV     #2,R0               ;SET SOB COUNT
11729  043000  010437  177546          5$:     MOV     R4,@#LKS            ;SET I.E. AND CLEAR READY
11730  043004  105737  177546          6$:     TSTB    @#LKS               ;SEE IF LINE CLOCK READY
11731  043010  100375                          BPL     6$                  ;LOOP IF NOT
11732  043012  077006                          SOB     R0,5$               ;DO IT ONE MORE TIME
11733  043014  000240                          NOP                                 ;AT THIS TIME BOTH THE CLOCK
11734                                                                       ;ARE READY TO INTERRUPT
11735  043016  105037  177776                  CLRB    @#PSW                       ;SET PRIORITY LEVEL 0
11736                                  ;A CLOCK INTERRUPT WILL OCCUR (8$) AND LOC 4$ WILL BE INCREMENTED
11737                                  ;AFTER THE CLOCK SERVICE A TTY INTERRUPT WILL OCCUR. THE TTY INT SERV
11738                                  ;ICE WILL SHIFT LEFT 4$.
11739
11740  043022  012737  000340  177776          MOV     #PR7,@#PSW              ;DON'T ALLOW 'LATE' INTERRUPTS
11741  043030  022767  000002  177734          CMP     #2,4$                   ;CHECK THAT THE CLOCK
11742  043036  001415                          BEQ     ARBFIN                  ;& TTY INTERRUPTED IN
11743  043040  104000                          HLT                             ;THE PROPER SEQUENCE
11744  043042  000413                          BR      ARBFIN
```

```
11745
11746  043044  005077  136244          7$:     CLR     @$TPS                   ;CLEAR IE BIT
11747  043050  006367  177716                  ASL     4$                      ;SHIFT INDICATOR
11748  043054  000002                          RTI                             ;RETURN
11749
11750  043056  005267  177710          8$:     INC     4$
11751  043062  012737  054362  000100          MOV     #LKSRV,@#LKVEC          ;SET CLOCK VECTORS
11752  043070  000002                          RTI
11753
11754
11755  043072  012737  063566  000064  ARBFIN: MOV     #TPISR,@#TPVEC          ;RESTORE TTY VECTOR
11756  043100  012737  000200  000066          MOV     #200,@#TPVEC+2
11757  043106  105037  177776                  CLRB    @#PSW                   ;RESTORE PROCESSOR PRIORITY
11758  043112  005077  136176                  CLR     @$TPS                   ;CLEAR IE BIT
11762  043116                          ARBEX:
  (4)                                   ;;**********************************************************************
  (3)                                   ;*TEST 77        TURN ON UBE AND MBT
  (4)                                   ;*      TURN ON THE MASS BUS TESTER AND UNIBUS EXERCISER IF PRESENT
  (3)                                   ;;**********************************************************************
  (2)
  (2)   043116  000240          TST77:  NOP
  (2)   043120  112737  000077  001252          MOVB    #77,@#$TSTNM            ;LOAD TEST NUMBER
  (2)   043126  013737  001252  177570          MOV     @#$TSTNM,@#DISPLAY      ;;DISPLAY TEST NUMBER
11763  043134  032737  001000  001550          BIT     #LKOPT,@#OPT.CP         ;BRANCH IF NOT AVAIL
11764  043142  001411                          BEQ     UBESET
11765  043144  012737  054362  000100          MOV     #LKSRV,@#LKVEC
11766  043152  012737  000340  000102          MOV     #PR7,@#LKVEC+2
11767  043160  052737  000100  177546          BIS     #100,@#LKS              ;SET IE BIT
11768
11769                                   ;;**********************************************************************
11770                                   ;TURN ON THE UNIBUS EXERCISER IF PRESENT
11771  043166  105737  001550          UBESET: TSTB    @#OPT.CP                ;IS UBE OPTION AVAILABLE?
11772  043172  100015                          BPL     MBTSET                  ;BRANCH IF NO
11773  043174  032777  010000  136042          BIT     #SW12,@$SWRP            ;INHIBIT UBE?
11774  043202  001011                          BNE     MBTSET                  ;BRANCH IF YES
11775  043204  032737  000040  172516          BIT     #BIT5,@#MMR3            ;IS MAP ON?
11776  043212  001045                          BNE     STMM                    ;BRANCH IF YES
11777  043214  004737  062340                  JSR     PC,@#UBEINIT            ;INITIALIZE UBE
11778  043220  012772  064545  000000          MOV     #64545,@(R2)            ;START UBE
11779
11780                                   ;;**********************************************************************
11781                                   ;TURN ON THE MASS BUS TESTER IF PRESENT
11782  043226  032737  002000  001550  MBTSET: BIT     #MBTOPT,@#OPT.CP        ;IS MBT AVAILABLE?
11783  043234  001434                          BEQ     STMM                    ;BRANCH IF NO
11784  043236  032777  000010  136000          BIT     #SW3,@$SWRP             ;INHIBIT MBT?
11785  043244  001030                          BNE     STMM                    ;BRANCH IF YES
11786  043246  122737  000060  001630          CMPB    #60,@#SUBPASS           ;FIRST SUB-PASS?
11787  043254  001024                          BNE     STMM                    ;BRANCH IF NO
11788  043256  105737  001601                  TSTB    @#MMON                  ;MEM MGMT ON?
11789  043262  001021                          BNE     STMM                    ;BRANCH IF YES
11790  043264  052777  000047  137076  MBT1:   BIS     #47,@MBTTBL+12          ;CLEAR THE MBT
11791  043272  012777  000007  137070          MOV     #7,@MBTTBL+12           ;SELECT UNIT 7
11792  043300  005077  137054                  CLR     @MBTTBL+2               ;CLEAR THE WORD COUNT
11793  043304  012777  054050  137066          MOV     #MBTSRV,@MBTTBL+22      ;SETUP INTERRUPT VECTOR
11794  043312  012777  000340  137062          MOV     #PR7,@MBTTBL+24         ;SET VECTOR PSW
11795  043320  112777  000161  137030          MOVB    #161,@MBTTBL            ;START MBT
```

```
11796                                   ;;*****************************************************************
11797                                   .SBTTL  STMM ROUTINE
11798                                   ;ROUTINE TO SET UP MEMORY MANAGEMENT TO RELOCATE PROGRAM CODE ABOVE 16K
11799                                   ;CHECK IF PROGRAM IS TO BE RELOCATED.
11800                                   ;SW6=1=NO RELOCATION
11801                                   ;;*****************************************************************
11802  043326  112737  000100  001252  STMM:   MOVB    #100,@#$TSTNM   ;LOAD TEST NUMBER
11803  043334  112737  000100  177570          MOVB    #100,@#DISPLAY  ;IN SWICH REG TOO
11804                                                                  ;ANY TIME TEST 100 IS IN ERROR REPORT
11805                                                                  ;THEN ERROR OCCURRED DURING RELOCATION
11806  043342  032777  000100  135674          BIT     #SW6,@$SWRP     ;RELOCATION DISABLED?
11807  043350  001402                          BEQ     3$              ;BRANCH IF NO
11808  043352  000167  002626                  JMP     ENDM
11809
11810                                   ;THE PROGRAM IS GOING TO RELOCATE.
11811                                   ;RELOCATION WILL BE PERFORMED IN KERNEL MODE WITH PSW SET AT PRIORITY
11812                                   ;LEVEL 4 (TO PREVENT TTY INTERRUPT-WHICH CHANGES DATA IN PROGRAM)
11813                                   ;THE 'T' BIT IS CLEARED (IF SET). AFTER THE DATA HAS BEEN WRITTEN IT IS
11814                                   ;VERIFIED BEFORE EXECUTION.
11815  043356  013727  177776          3$:     MOV     @#PSW,(PC)+     ;SAVE CURRENT PSW
11816  043362  000000                  OLDPSW: .WORD   0.
11817  043364  012737  000200  177776          MOV     #PR4,@#PSW
11818  043372  004767  017716                  JSR     PC,CLRTBIT      ;CO CLEAR 'T' BIT IF SET
11819
11820                                   ;NOW SETUP MEMORY MANAGEMENT REGISTERS.
11821  043376  012700  077406                  MOV     #77406,R0       ;SET CONSTANT=R/W UP 4K WORDS
11822  043402  010037  172300                  MOV     R0,@#KIPDR0     ;SET KIPDR0 THROUGH 7 R/W UP 4K WORDS
11823  043406  010037  172302                  MOV     R0,@#KIPDR1
11824  043412  010037  172304                  MOV     R0,@#KIPDR2
11825  043416  010037  172306                  MOV     R0,@#KIPDR3
11826  043422  010037  172310                  MOV     R0,@#KIPDR4
11827  043426  010037  172312                  MOV     R0,@#KIPDR5
11828  043432  010037  172314                  MOV     R0,@#KIPDR6
11829  043436  010037  172316                  MOV     R0,@#KIPDR7
11830
11831  043442  005037  172340                  CLR     @#KIPAR0        ;SET UP KIPDR0 THROUGH 3 FOR NO RELOCATION
11832  043446  012737  000200  172342          MOV     #200,@#KIPAR1
11833  043454  012737  000400  172344          MOV     #400,@#KIPAR2
11834  043462  012737  000600  172346          MOV     #600,@#KIPAR3
11835  043470  013737  001616  172350          MOV     @#NEXPAR,@#KIPAR4       ;PAR4 MAPS TO BEGINNING OF RELOCATION SPOT
11836  043476  013737  172350  172352          MOV     @#KIPAR4,@#KIPAR5
11837  043504  062737  000600  172352          ADD     #600,@#KIPAR5   ;PAR5 MAPS TO TOP 4K PAGE (NEXPAR START ADDRESS + 12K)
11838  043512  012737  177600  172356          MOV     #177600,@#KIPAR7        ;AND OF COUSE THE I/O PAGE
11839                                  ;NOW SETUP USER MEM MGMT REGISTERS
11840  043520  010037  177600          1$:     MOV     R0,@#UIPDR0     ;SET UP USER MEM MGMT REGS
11841  043524  010037  177602                  MOV     R0,@#UIPDR1
11842  043530  010037  177604                  MOV     R0,@#UIPDR2
11843  043534  010037  177606                  MOV     R0,@#UIPDR3
11844  043540  010037  177616                  MOV     R0,@#UIPDR7
11845  043544  016737  136046  177640          MOV     NEXPAR,@#UIPAR0
11846  043552  013737  177640  177642          MOV     @#UIPAR0,@#UIPAR1
11847  043560  062737  000200  177642          ADD     #200,@#UIPAR1
11848  043566  013737  177640  177644          MOV     @#UIPAR0,@#UIPAR2
11849  043574  062737  000400  177644          ADD     #400,@#UIPAR2
11850  043602  013737  177640  177646          MOV     @#UIPAR0,@#UIPAR3
11851  043610  062737  000600  177646          ADD     #600,@#UIPAR3
```

```
11852  043616  013737  172356  177656          MOV     @#KIPAR7,@#UIPAR7
11853
11854  043624  010037  172200                  MOV     R0,@#SIPDR0     ;SET UP SUPERVISOR MEM MGMT REGS
11855  043630  010037  172202                  MOV     R0,@#SIPDR1
11856  043634  010037  172204                  MOV     R0,@#SIPDR2
11857  043640  010037  172206                  MOV     R0,@#SIPDR3
11858  043644  010037  172216                  MOV     R0,@#SIPDR7
11859  043650  016737  135742  172240          MOV     NEXPAR,@#SIPAR0
11860  043656  013737  172240  172242          MOV     @#SIPAR0,@#SIPAR1
11861  043664  062737  000200  172242          ADD     #200,@#SIPAR1
11862  043672  013737  172240  172244          MOV     @#SIPAR0,@#SIPAR2
11863  043700  062737  000400  172244          ADD     #400,@#SIPAR2
11864  043706  013737  172240  172246          MOV     @#SIPAR0,@#SIPAR3
11865  043714  062737  000600  172246          ADD     #600,@#SIPAR3
11866  043722  013737  172356  172256          MOV     @#KIPAR7,@#SIPAR7
11867  043730  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEM MGMT
11868  043736  012737  000060  172516          MOV     #60,@#SR3       ;SETUP SR3
11869  043744  110637  001601                  MOVB    SP,@#MMON       ;SET MEM MGMT ON IND = ON
11870  043750  005037  000006          RETRY:  CLR     @#ERRVEC+2
11871  043754  012737  046202  000004          MOV     #ENDMEM,@#ERRVEC;SET TIME OUT TRAP VECTOR
11872  043762  013701  000116                  MOV     @#CACHVEC+2,R1  ;SAVE CACHVEC PSW
11873  043766  005037  000116                  CLR     @#CACHVEC+2
11874  043772  012737  046202  000114          MOV     #ENDMEM,@#CACHVEC       ;SET UP CACHE VECTOR FOR HOLE
11875  044000  012702  100000                  MOV     #100000,R2      ;SETUP GENERAL REGISTERS
11876  044004  012700  000000                  MOV     #0,R0           ;DATA WILL BE RELOCATED FROM
11877                                                                  ;ADDRESS IN R0 TO ADDRESS IN R2 AFTER MAPPING
11878  044010  012703  127776                  MOV     #127776,R3      ;SELECT PAR5 + 2K (14K TOTAL REQUIRED)
11879  044014  010013                          MOV     R0,(R3)         ;TRAP TO ENDMEM IF INSUFFICIENT MEMORY
11880  044016  012737  064422  000004          MOV     #ERPRT,@#ERRVEC ;RESTORE ERROR TRAP VECTOR
11881  044024  010137  000116                  MOV     R1,@#CACHVEC+2
11882  044030  012737  063616  000114          MOV     #.PARSRV,@#CACHVEC      ;RESTORE CAC E VECTOR
11883  044036  000137  044236                  JMP     @#IOMON
11884                                  ;;*****************************************************************
11885                                  .SBTTL  RELOCATION ROUTINE
11886                                  ;*      THIS ROUTINE IS USED TO RELOCATE THE 9 SUBTESTS UP TO 28K.
11887                                  ;*      IF RELOCATION BY AN I/O DEVICE IS SELECTED, CONTROL IS PASSED
11888                                  ;*      TO THE I/O MONITOR.
11889                                  ;*      ENTER WITH:
11890                                  ;*              FRSTAD=PHYSICAL ADDRESS OF FIRST CODE
11891                                  ;*              FACTOR=NUMBER OF BYTES ABOVE BASE CODE
11892                                  ;*              R2     =LAST PHYSICAL ADDRESS OF THE SECTION
11893                                  ;*      EXIT TO I/O MONITOR WITH:
11894                                  ;*              OLDBASE=FIRST PHYSICAL ADDRESS TO BE RELOCATED
11895                                  ;*              NWBASL =FIRST PHYSICAL ADDRESS TO RELOCATE TO
11896                                  ;*              IOWC   =TWO'S COMPLIMENT WORD COUNT
11897                                  ;;*****************************************************************
11898  044042  032777  000100  135174  RELOC:  BIT     #SW6,@$SWRP             ;IS RELOCATION DISABLED?
11899  044050  001067                          BNE     EXITRE                  ;BRANCH IF YES
11900  044052  105737  001601                  TSTB    @#MMON                  ;IS MEMORY MGMT ON?
11901  044056  001064                          BNE     EXITRE                  ;BRANCH IF YES
11902  044060  013700  001610                  MOV     @#FRSTAD,R0             ;GET FIRST ADDRESS TO BE RELOCATED
11903  044064  010005                          MOV     R0,R5
11904                                  ;LAST ADDRESS IS IN R2
11905  044066  010203                          MOV     R2,R3                   ;SAVE LAST ADDRESS
11906  044070  010204                          MOV     R2,R4
11907  044072  160004                          SUB     R0,R4                   ;R4 NOW HAS BYTE COUNT
```

```
11908  044074  010437  001606                  MOV     R4,@#$FACTOR            ;SAVE BYTE COUNT
11909  044100  005737  001604                  TST     @#FACTOR                ;FIRST RELOC IS TO ENDTAG+2
11910  044104  001004                          BNE     1$                      ;BRANCH IF NOT EXECUTING BASE CODE
11911  044106  010237  044234                  MOV     R2,@#RETPC              ;SAVE RETURN PC TO NEXT SECTION
11912  044112  013702  001612                  MOV     @#FRSTMEM,R2            ;GET FIRST ADDRESS TO RELOCATE TO
11913  044116  060204                  1$:     ADD     R2,R4                   ;R4 NOW CONTAINS LAST MEM ADDRESS
11914  044120  020437  001614                  CMP     R4,@#LSTMEM             ;ENOUGH MEMORY?
11915  044124  101042                          BHI     NOMEM                   ;BRANCH IF NO
11916  044126  160204                          SUB     R2,R4                   ;R4 NOW HAS BYTE COUNT
11917  044130  005037  001604                  CLR     @#FACTOR
11918  044134  032777  000400  135102          BIT     #SW8,@$SWRP             ;INHIBIT RELOC BY I/O DEVICE?
11919  044142  001414                          BEQ     RELNIO                  ;BRANCH IF YES
11920  044144  010037  001636                  MOV     R0,@#OLDBASE            ;SAVE START ADDRESS
11921  044150  010237  001640                  MOV     R2,@#NWBASL             ;SAVE NEW BASE ADDRESS
11922  044154  005037  001642                  CLR     @#NWBASH                ;
11923  044160  006204                          ASR     R4                      ;MAKE IT A WORD COUNT
11924  044162  005404                          NEG     R4                      ;GET TWO'S COMPLIMENT
11925  044164  010437  001644                  MOV     R4,@#IOWC               ;SAVE R4 AS WORDCOUNT
11926  044170  000167  000122                  JMP     ENTER2                  ;GO TO I/O MONITOR
11927                                  ;RELOCATE BY CPU-MEMORY MANAGEMENT OFF
11928  044174  012022                  RELNIO: MOV     (R0)+,(R2)+             ;RELOCATE CODE
11929  044176  020003                          CMP     R0,R3                   ;DONE YET?
11930  044200  001375                          BNE     RELNIO                  ;BRANCH IF NO
11931  044202  004737  062660                  JSR     PC,@#CHKDAT             ;GO CHECK DATA
11932  044206  102010                          BVC     EXITRE
11933  044210  010037  001352                  MOV     R0,@#$TMP0              ;SAVE R0 FOR TYPEOUT
11934  044214  010237  001572                  MOV     R2,@#VADR               ;SAVE R2
11935  044220  004737  062556                  JSR     PC,@#CNVADR             ;CONVERT R2 TO A PHYSICAL ADR
11936  044224  104006                          ERROR   6
11937  044226  000401                          BR      NOMEM
11938  044230  010207                  EXITRE: MOV     R2,PC                   ;GO EXECUTE RELOCATED CODE
11939  044232  011707                  NOMEM:  MOV     (PC),PC                 ;GO TO NEXT SECTION
11940  044234  000000                  RETPC:  .WORD   0                       ;CONTAINS PC OF NEXT SECTION
11941                                  ;;*****************************************************************
11942                                  .SBTTL  I/O RELOCATION MONITOR
11943                                  ;*      THIS ROUTINE IS USED TO SCHEDULE I/O DEVICES FOR SUBTEST
11944                                  ;*      RELOCATION AND PROGRAM RELOCATION. THE I/O DEVICE UNIT
11945                                  ;*      NUMBER IS DETERMINED, THE BUS ADDRESS CALCULATED, THE WORD
11946                                  ;*      COUNT CALCULATED AND PASSED TO THE DEVICE HANDLER.
11947                                  ;;*****************************************************************
11948  044236  012737  044244  001262  IOMON:  MOV     #1$,@#$LPERR    ;SETUP ERROR LOOP
11949  044244  012737  000000  001636  1$:     MOV     #0,@#OLDBASE
11950  044252  013705  001616                  MOV     @#NEXPAR,R5             ;SETUP R4 AND R5
11951  044256  005004                          CLR     R4                      ;TO FORM 22 BIT ADDRESS
11952  044260  073427  000006                  ASHC    #6,R4                   ;FORM 22 BIT ADDRESS
11953  044264  010537  001640                  MOV     R5,@#NWBASL             ;SAVE LOWER 16 BITS
11954  044270  010437  001642                  MOV     R4,@#NWBASH             ;SAVE UPPER 6 BITS
11955  044274  032777  000400  134742          BIT     #SW8,@$SWRP             ;RELOCATE VIA I/O?
11956  044302  001002                          BNE     2$                      ;BRANCH UF YES
11957  044304  000167  001444                  JMP     RELOCP                  ;GO RELOCATE VIA CP
11958  044310  012737  174000  001644  2$:     MOV     #174000,@#IOWC          ;SET WORD COUNT TO 2K
11959  044316  005037  001356          ENTER2: CLR     @#$TMP2
11960  044322  012737  177776  001654          MOV     #-2,@#RNTBINX           ;SETUP RUN TABLE INDEX
11961  044330  005037  001352                  CLR     @#$TMP0
11962  044334  005002                          CLR     R2                      ;CLEAR LEGAL DEV FLAG
11963  044336  032777  000040  134700  41$:    BIT     #SW5,@$SWRP             ;INHIBIT ROUND ROBIN?
```

```
11964  044344  001416                        BEQ     50$                     ;BRANCH IF NO
11965  044346  005737  001352                TST     @#$TMP0                 ;FLAG SET?
11966  044352  001027                        BNE     43$                     ;BRANCH IF YES
11967  044354  117737  134664  001650        MOVB    @$SWRP,@#DEVINDX                ;GET DEVICE FROM SWITCHES
11968  044362  042737  177770  001650        BIC     #177770,@#DEVINDX       ;MASK LOWER 3 BITS
11969  044370  006337  001650                ASL     @#DEVINDX               ;ADJUST FOR WORD INDEX
11970  044374  005237  001352                INC     @#$TMP0                 ;SET FLAG
11971  044400  000414                        BR      43$                     ;CONTINUE
11972  044402  012705  000010        50$:    MOV     #10,R5                  ;SET SOB COUNT
11973  044406  022737  000016  001650  40$:   CMP     #16,@#DEVINDX           ;LAST DEVICE YET?
11974  044414  001003                        BNE     42$                     ;BRANCH IF NO
11975  044416  012737  177776  001650  48$:   MOV     #-2,@#DEVINDX           ;INIT DEVICE INDEX
11976  044424  062737  000002  001650  42$:   ADD     #2,@#DEVINDX            ;INCREMENT INDEX
11977  044432  013703  001650          43$:   MOV     @#DEVINDX,R3            ;GET INDEX
11978  044436  012737  000401  001354        MOV     #401,@#$TMP1            ;INIT UNIT MASK
11979  044444  012704  000010                MOV     #10,R4                  ;SET SOB COUNT
11980  044450  133763  001354  001712  44$:   BITB    @#$TMP1,SYSSIZE(R3)     ;IS THIS UNIT EXISTENT?
11981  044456  001405                        BEQ     52$                     ;BRANCH IF NO
11982  044460  005202                        INC     R2                      ;SET LEGAL DEVICE FLAG
11983  044462  133763  001355  001713        BITB    @#$TMP1+1,SYSSIZE+1(R3) ;HAS IT BEEN USED?
11984  044470  001516                        BEQ     11$                     ;BRANCH IF NO
11985  044472  006337  001354          52$:   ASL     @#$TMP1                 ;SELECT NEXT UNIT
11986  044476  077414                        SOB     R4,44$                  ;CONTINUE
11987  044500  005737  001352                TST     @#$TMP0                 ;INHIBIT ROUND ROBIN?
11988  044504  001013                        BNE     45$                     ;BRANCH IF YES
11989  044506  077541                        SOB     R5,40$                  ;CONTINUE
11990  044510  005702                        TST     R2                      ;ANY DEVICES AT ALL?
11991  044512  001442                        BEQ     46$                     ;BRANCH IF NO
11992  044514  012704  000010                MOV     #10,R4                  ;SET SOB COUNT
11993  044520  012701  001713                MOV     #SYSSIZE+1,R1           ;GET ADR OF SIZE TABLE
11994  044524  105021                47$:    CLRB    (R1)+                   ;CLEAR ALL USED BITS
11995  044526  005201                        INC     R1                      ;IN ALL DEVICES
11996  044530  077403                        SOB     R4,47$                  ;CONTINUE
11997  044532  000701                        BR      41$
11998  044534  005702                45$:    TST     R2                      ;WAS IT A LEGAL DEVICE?
11999  044536  001403                        BEQ     49$                     ;BRANCH IF NO
12000  044540  105063  001713                CLRB    SYSSIZE+1(R3)           ;CLEAR ALL USED BITS THIS DEV
12001  044544  000732                        BR      43$
12002  044546  010367  000016          49$:   MOV     R3,60$                  ;
12003  044552  062767  064572  000010        ADD     #MSGINX,60$             ;GEN MESSAGE ADR
12004  044560  017767  000004  000002        MOV     @60$,60$
12005  044566  104400                        TYPE
12006  044570  000000                60$:    .WORD
12007  044572  104400  044600                TYPE    ,65$            ;;TYPE ASCIZ STRING
  (1)  044576  000407                        BR      64$             ;;GET OVER THE ASCIZ
  (1)                               ;;65$:  .ASCIZ  /UNAVAILABLE/<CRLF>
  (1)  044616                       64$:
12008  044616  000637                        BR      ENTER2
12009  044620  105737  001602          46$:   TSTB    @#QV                    ;ACT11?
12010  044624  001016                        BNE     51$                     ;BRANCH IF YES
12011  044626  005227  177777                INC     #-1
12012  044632  001013                        BNE     51$
12013  044634  104400  044642                TYPE    ,67$            ;;TYPE ASCIZ STRING
  (1)  044640  000410                        BR      66$             ;;GET OVER THE ASCIZ
  (1)                               ;;67$:  .ASCIZ  ?NO I/O DEVICES?<CRLF>
  (1)  044662                       66$:
```

```
12014  044662  105737  001601          51$:    TSTB    @#MMON                  ;MGMT ON?
12015  044666  001012                          BNE     61$                     ;BRANCH IF YES
12016  044670  013700  001636                  MOV     @#OLDBASE,R0            ;RESTORE R0
12017  044674  013702  001640                  MOV     @#NWBASL,R2             ;RESTORE R2
12018  044700  013703  001606                  MOV     @#$FACTOR,R3            ;GET RELOCATION FACTOR
12019  044704  060003                          ADD     R0,R3                   ;FORM LAST ADDRESS
12020  044706  010005                          MOV     R0,R5                   ;SETUP R5
12021  044710  000167  177260                  JMP     RELNIO                  ;GO RELOCATE WITH CP
12022  044714  012702  100000          61$:    MOV     #100000,R2              ;SETUP REGISTERS
12023  044720  005000                          CLR     R0                      ;WITH FROM AND TO ADDRESS
12024  044722  000137  045754                  JMP     @#RELOCP                ;RELOCATE VIA CP
12025  044726  105763  002006          11$:    TSTB    RP3HSTAT(R3)            ;IS HANDLER BUSY?
12026  044732  100405                          BMI     8$                      ;BRANCH IF NO
12027  044734  005737  001352                  TST     @#$TMP0                 ;ROUND ROBIN?
12028  044740  001372                          BNE     11$                     ;BRANCH IF NO
12029  044742  000167  177370                  JMP     41$
12030  044746  005763  002006          8$:     TST     RP3HSTAT(R3)            ;DID HANDLER FAIL?
12031  044752  100005                          BPL     62$                     ;BRANCH IF NO
12032  044754  005737  001352                  TST     @#$TMP0                 ;ROUND ROBIN
12033  044760  001402                          BEQ     62$                     ;BRANCH IF YES
12034  044762  000137  045734                  JMP     @#15$
12035  044766  153763  001355  001713  62$:    BISB    @#$TMP1+1,SYSSIZE+1(R3) ;SET UNIT USED BIT
12036  044774  005002                          CLR     R2
12037  044776  006037  001354          30$:    ROR     @#$TMP1                 ;ENCODE THE BIT POSITION
12038  045002  005202                          INC     R2                      ;INTO A UNIT NUMBER
12039  045004  103374                          BCC     30$
12040  045006  005302                          DEC     R2
12041  045010  010237  001652                  MOV     R2,@#UNITNO             ;SAVE UNIT NUMBER
12042  045014  013763  001644  002026  10$:    MOV     @#IOWC,RP3HWC(R3)       ;GIVE WORD COUNT TO HANDLER
12043  045022  010304                          MOV     R3,R4                   ;
12044  045024  072427  000003                  ASH     #3,R4                   ;ENCODE DEVICE FOR RUNTABLE
12045  045030  053704  001652                  BIS     @#UNITNO,R4             ;ENCODE UNIT NUMBER
12046  045034  006304                          ASL     R4
12047  045036  062737  000002  001654          ADD     #2,@#RNTBINX            ;INCREMENT RUN TABLE INDEX
12048  045044  013702  001654                  MOV     @#RNTBINX,R2            ;GET RUN TABLE INDEX
12049  045050  110462  001733                  MOVB    R4,RUNTBL+1(R2)         ;ENTER DEV & UNIT IN TABLE
12050  045054  013763  001652  002122          MOV     @#UNITNO,RP3UNIT(R3)    ;GIVE HANDLER UNIT NUMBER
12051  045062  012737  000240  000012          MOV     #PR5,@#RESVEC+2         ;SETUP RESERVED VECTOR PSW
12052  045070  016337  002136  000010          MOV     RP3HANA(R3),@#RESVEC    ;SETUP RESERVED VECTOR
12053  045076  006303                          ASL     R3                      ;ADJUST INDEX
12054  045100  013763  001636  002042          MOV     @#OLDBASE,RP3OLD(R3)    ;GIVE HANDLER OLD BASE ADDRESS
12055  045106  013763  001640  002072          MOV     @#NWBASL,RP3NWL(R3)     ;GIVE HANDLER
12056  045114  013763  001642  002074          MOV     @#NWBASH,RP3NWH(R3)     ;NEW BASE ADDRESS
12057  045122  005063  002044                  CLR     RP3OLD+2(R3)            ;ENSURE OLD BASE HIGH IS CLR
12058  045126  000010                  CALLHANDLER
12059  045130  105737  001601                  TSTB    @#MMON                  ;IS MEMORY MANAGEMENT ON?
12060  045134  001416                          BEQ     13$                     ;BRANCH IF NO
12061  045136  022737  000014  001654          CMP     #14,@#RNTBINX           ;TRANSFERED 14K YET?
12062  045144  001412                          BEQ     13$                     ;BRANCH IF YES
12063  045146  062737  010000  001636          ADD     #10000,@#OLDBASE        ;ADD 2K
12064  045154  062737  010000  001640          ADD     #10000,@#NWBASL         ;TO BASE
12065  045162  005537  001642                  ADC     @#NWBASH                ;ADDRESSES
12066  045166  000137  044336                  JMP     @#41$
12067  045172  113705  001701          13$:    MOVB    @#LTICKS+1,R5           ;GET SECOND COUNT
12068  045176  062705  000002                  ADD     #2,R5                   ;INCREMENT BY TWO
12069  045202  162705  000074                  SUB     #60.,R5                 ;ENSURE RESULT IS 59 OR LESS
```

```
12070  045206  100002                    BPL     31$                     ;
12071  045210  062705  000074            ADD     #60.,R5                 ;COUNT WAS LESS THAN 58-RESTORE
12072  045214  012700  000010    31$:    MOV     #10,R0                  ;SET SOB COUNT
12073  045220  005002                    CLR     R2
12074  045222  005003                    CLR     R3
12075  045224  005004                    CLR     R4
12076  045226  066203  002006    14$:    ADD     RP3HSTAT(R2),R3         ;ADD ALL THE HANDLER
12077  045232  005504                    ADC     R4                      ;STATUS WORDS. WHEN ALL
12078  045234  062702  000002            ADD     #2,R2                   ;TRANSFERS ARE FINISHED
12079  045240  077006                    SOB     R0,14$                  ;RESULT WILL BE 2000
12080  045242  006103                    ROL     R3                      ;(WITHOUT ROTATE)
12081  045244  005504                    ADC     R4
12082  045246  022703  004000            CMP     #4000,R3                ;ALL DONE?
12083  045252  001406                    BEQ     32$                     ;BRANCH IF YES
12084  045254  123705  001701            CMPB    @#LTICKS+1,R5           ;TWO SECONDS ELAPSED YET?
12085  045260  001355                    BNE     31$                     ;BRANCH IF NO
12086  045262  104015                    ERROR   15                      ;DEVICE HUNG
12087  045264  000177  133772            JMP     @$LPERR                 ;RESTART RELOCATION
12088  045270  005704    32$:    TST     R4                      ;ANY DEVICE ERRORS?
12089  045272  001402                    BEQ     82$                     ;BRANCH IF NO
12090  045274  000167  000434            JMP     15$                     ;ERROR
12091  045300  105737  001601    82$:    TSTB    @#MMON                  ;MEM MGMT ON?
12092  045304  001012                    BNE     25$                     ;BRANCH IF YES
12093  045306  013705  001636            MOV     @#OLDBASE,R5            ;SETUP R5 FOR DATA CHECK
12094  045312  010500                    MOV     R5,R0
12095  045314  063700  001606            ADD     @#$FACTOR,R0            ;GET LAST ADDRESS
12096                                                                    ;OF GOOD DATA
12097  045320  013702  001640            MOV     @#NWBASL,R2
12098  045324  063702  001606            ADD     @#$FACTOR,R2            ;GET LAST ADDRESS
12099                                                                    ;OF DATA TO BE CHECKED
12100  045330  000411                    BR      22$                     ;CONTINUE
12101  045332  012700  070000    25$:    MOV     #70000,R0               ;GET LAST ADR + 2 OF GOOD DATA
12102  045336  012702  110000            MOV     #110000,R2              ;GET LAST ADR + 2 OF DATA TO BE CHECKED
12103  045342  013737  172352  172350    MOV     @#KIPAR5,@#KIPAR4       ;SET UP PAR4 FOR TOP 4K BANK
12104  045350  012705  002414            MOV     #$ERRTB,R5              ;DON'T CHECK BELOW $ERRTB
12105  045354  004737  062660    22$:    JSR     PC,@#CHKDAT             ;GO CHECK DATA
12106  045360  102413                    BVS     81$                     ;BRANCH IF ERROR
12107  045362  105737  001254            TSTB    @#$ERFLG                ;ANY ERRORS?
12108  045366  001002                    BNE     83$                     ;BRANCH IF YES
12109  045370  000167  000466            JMP     EXIT                    ;RETURN
12110  045374  032777  001000  133642  83$:    BIT     #SW9,@$SWRP             ;LOOP ON ERROR?
12111  045402  001473                    BEQ     100$+2                  ;BRANCH IF NO
12112  045404  000167  000250            JMP     20$                     ;GO DO FUNCTION AGAIN
12113  045410  005001    81$:    CLR     R1
12114  045412  010037  001352            MOV     R0,@#$TMP0
12115  045416  010237  001572            MOV     R2,@#VADR
12116  045422  010003                    MOV     R0,R3                   ;SAVE ERROR ADDRESS
12117  045424  005004                    CLR     R4
12118  045426  105737  001601            TSTB    @#MMON                  ;IS MEM MGMT ON?
12119  045432  001406                    BEQ     16$                     ;BRANCH IF NO
12120  045434  162703  010000    17$:    SUB     #10000,R3               ;SUBTRACT 2K FROM ERROR ADDRESS
12121  045440  100403                    BMI     16$                     ;BRANCH IF BLOCK IS FOUND
12122  045442  062704  000002            ADD     #2,R4                   ;COUNT ONE MORE BLOCK
12123  045446  000772                    BR      17$                     ;CONTINUE
12124                            ;R4 NOW CONTAINS INDEX OF ERROR FOR RUN TIME TABLE
12125  045450  116404  001733    16$:    MOVB    RUNTBL+1(R4),R4         ;GET DEVICE THAT FAILED
```

```
12126  045454  042704  177400              BIC     #177400,R4              ;ENSURE HIGH BYTE CLEAR
12127  045460  006204                      ASR     R4                      ;THROW AWAY LSB
12128  045462  005005                      CLR     R5                      ;ENSURE R5 CLEAR
12129  045464  073427  177775              ASHC    #-3,R4                  ;GET UNIT NUMBER IN R5
12130  045470  010500                      MOV     R5,R0
12131  045472  072027  177763              ASH     #-15,R0
12132  045476  042700  177770              BIC     #177770,R0
12133  045502  010037  001360              MOV     R0,@#$TMP3
12134  045506  010403                      MOV     R4,R3                   ;AND DEVICE INDEX IN R4 & R3
12135  045510  010337  001356              MOV     R3,@#$TMP2
12136  045514  012737  000001  001354      MOV     #1,@#$TMP1                      ;ENCODE 3 BIT UNIT NO INTO
12137  045522  162705  020000      19$:    SUB     #20000,R5               ;ONE BIT IN THE LOW BYTE OF $TMP1
12138  045526  103403                      BCS     18$                     ;BRANCH IF DONE
12139  045530  006137  001354              ROL     @#$TMP1                 ;SELECT NEXT UNIT
12140  045534  000772                      BR      19$                     ;CONTINUE
12141  045536  012737  045660  001262  18$:  MOV     #20$,@#$LPERR           ;SETUP LOOP RETURN
12142  045544  005701                      TST     R1                      ;DEVICE ERROR?
12143  045546  001010                      BNE     100$                    ;BRANCH IF YES
12144  045550  104010                      ERROR   10                      ;DATA CHECK ERROR
12145  045552  105737  001601              TSTB    @#MMON                  ;MGMT ON?
12146  045556  001002                      BNE     70$                     ;BRANCH IF YES
12147  045560  000137  044316      71$:    JMP     @#ENTER2
12148  045564  000137  044236      70$:    JMP     @#IOMON
12149  045570  104007      100$:   ERROR   7
12150  045572  042763  100000  002006      BIC     #BIT15,RP3HSTAT(R3)     ;CLEAR THE ERROR
12151  045600  105037  001254              CLRB    @#$ERFLG
12152  045604  022703  000002              CMP     #2,R3                   ;RK05 ERROR?
12153  045610  002405                      BLT     90$                     ;BRANCH IF RH70
12154  045612  003016                      BGT     92$                     ;BRANCH IF RP03
12155  045614  112777  000001  134414      MOVB    #1,@RKCS                ;RK CONTROLLER CLEAR
12156  045622  000412                      BR      92$
12157  045624  022703  000012      90$:    CMP     #12,R3                  ;RS04?
12158  045630  001004                      BNE     91$                     ;BRANCH IF NO
12159  045632  052777  000040  134464      BIS     #BIT5,@RSCS2            ;CLEAR RS CONTROLLER
12160  045640  000403                      BR      92$
12161  045642  052777  000040  134414  91$:  BIS     #BIT5,@RP4CS2           ;CLEAR RP04 CONTROLLER
12162  045650  105737  001601      92$:    TSTB    @#MMON                  ;MGMT ON?
12163  045654  001343                      BNE     70$                     ;BRANCH IF YES
12164  045656  000740                      BR      71$
12165  045660  052763  000400  002006  20$:  BIS     #BIT8,RP3HSTAT(R3)      ;SET REPEAT FLAG IN HANDLER
12166  045666  016337  002136  000010      MOV     RP3HANA(R3),@#RESVEC    ;SETUP RESERVED INSTRUCTION VECTOR
12167  045674  000010                      CALLHANDLER
12168  045676  105763  002006      21$:    TSTB    RP3HSTAT(R3)            ;HANDLER FINISHED?
12169  045702  100375                      BPL     21$                     ;BRANCH IF NO
12170  045704  005763  002006              TST     RP3HSTAT(R3)            ;ANY ERROR?
12171  045710  100712                      BMI     18$                     ;BRANCH IF YES
12172  045712  005701                      TST     R1                      ;DEVICE ERROR?
12173  045714  001002                      BNE     80$                     ;BRANCH IF YES
12174  045716  000167  177352              JMP     32$+4                   ;GO CHECK DATA
12175  045722  032777  001000  133314  80$:  BIT     #BIT9,@$SWRP            ;STILL LOOPING?
12176  045730  001353                      BNE     20$                     ;BRANCH IF YES
12177  045732  000717                      BR      100$+2                  ;CONTINUE TEST
12178                          ;THE FOLLOWING CODE HANDLES DEVICE ERROR ON RELOCATION
12179  045734  005004      15$:    CLR     R4                      ;SET INDEX
12180  045736  010601                      MOV     SP,R1
12181  045740  005764  001750      24$:    TST     RUNTRAK(R4)             ;SEARCH FOR DEVICE ERROR
```

```
12182  045744  100641                          BMI     16$                     ;BRANCH IF ERROR
12183  045746  062704  000002                  ADD     #2,R4                   ;INCREMENT INDEX
12184  045752  000772                          BR      24$                     ;CONTINUE SEARCH
12185                                  ;RELOCATE P   C U-MEMORY MANAGEMENT ON
12186  045754  012703  010000          RELOCP: MO\     #4096.,R3               ;4K COUNTER
12187  045760  012022                  1$:     MO:     (R0)+,(R2)+             ;RELOCATE CODE
12188  045762  077302                          SOB     R3,1$                   ;BR IF NOT DONE 4K WORDS
12189  045764  023737  172350  172352          CMP     @#KIPAR4,@#KIPAR5       ;DONE 16K YET?
12190  045772  001414                          BEQ     2$                      ;BR IF DONE
12191  045774  062737  000200  172350          ADD     #200,@#KIPAR4           ;MAP TO NEXT 4K SPACE
12192  046002  012702  100000                  MOV     #100000,R2              ;MAP WITH R2 (PAR4)
12193  046006  023737  172350  172352          CMP     @#KIPAR4,@#KIPAR5       ;DOING LAST 4K BANK?
12194  046014  001357                          BNE     RELOCP                  ;BR IF NOT
12195  046016  012703  004000                  MOV     #2048.,R3               ;2K COUNTER
12196  046022  000756                          BR      1$                      ;RELOCATE LAST 2K ONLY (14K TOTAL)
12197  046024  012705  002006          2$:     MOV     #RP3HSTAT,R5    ;DON'T CHECK BELOW RP3HSTAT
12198  046030  004737  062660                  JSR     PC,@#CHKDAT             ;CHECK DATA
12199  046034  102012                          BVC     EXIT
12200  046036  010037  001352                  MOV     R0,@#$TMP0
12201  046042  010237  001572                  MOV     R2,@#VADR
12202  046046  104006                          ERROR   6
12203  046050  013737  001616  172350          MOV     @#NEXPAR,@#KIPAR4       ;RESTORE PAR4
12204  046056  000167  175666                  JMP     RETRY
12205  046062  105737  001601          EXIT:   TSTB    @#MMON                  ;MEM MGMT ON?
12206  046066  001002                          BNE     .+6                     ;BRANCH IF YES
12207  046070  000137  044230                  JMP     @#EXITRE
12208  046074  062737  000077  001616          ADD     #77,@#NEXPAR            ;SET VALUE FOR NEXT RELOCATION
12209  046102  013737  172350  172340          MOV     @#KIPAR4,@#KIPAR0
12210  046110  063737  172350  172342          ADD     @#KIPAR4,@#KIPAR1
12211  046116  063737  172350  172344          ADD     @#KIPAR4,@#KIPAR2
12212  046124  063737  172350  172346          ADD     @#KIPAR4,@#KIPAR3
12213                                  ;*****************************************************************************
12214                                  ;PROGRAM IS NOW EXECUTING IN KERNEL MODE RELOCATED TO ADDRESS AS SPEC-
12215                                  ;IFIED IN KIPAR0. FOR EX. IF KIPAR0=1600 THEN PROGRAM EXECUTING AT
12216                                  ;ADDRESS 160000+(PC)
12217  046132  013700  172340                  MOV     @#KIPAR0,R0     ;GET PAR0
12218  046136  072027  177771                  ASH     #-7,R0          ;GET BITS <14:7> IN LOW BYTE
12219  046142  110037  001253                  MOVB    R0,@#$TSTNM+1   ;PUT IN DSPLAY REG HIGH BYTE
12220  046146  012706  001200                  MOV     #KERSTK,SP      ;SET KERNEL STACK PTR
12221
12222  046152  005037  177776                  CLR     @#PSW
12223  046156  016746  175200                  MOV     OLDPSW,-(SP)    ;RESTORE OLD PSW
12224  046162  012746  010000                  MOV     #LOOP,-(SP)
12225  046166  105737  001600                  TSTB    @#NEXEC         ;BRANCH IF TEST CODE TO
12226  046172  001402                          BEQ     1$              ;BE EXECUTED
12227  046174  012716  043326                  MOV     #STMM,(SP)
12228  046200  000002                  1$:     RTI                     ;RESTART PROGRAM AT LOOP
12229
12230                                  ;WHEN RELOCATION ABOVE 28K IS COMPLETE PROGRAM TRAPS TO ENDMEM.
12231  046202  022626                  ENDMEM: CMP     (SP)+,(SP)+     ;POP STACK TWICE
12232  046204  005037  177572          ENDM:   CLR     @#SR0           ;DISABLE MEM MGMT
12233  046210  042737  000020  172516          BIC     #BIT4,@#MMR3    ;CLEAR 22 BIT MODE
12234
12235                                  ;*****************************************************************************
12236                                  ;AT THIS TIME A 'SUB-PASS' HAS BEEN COMPLETED.
12237                                  ;PROGRAM NOW EXECUTING IN KERNEL MODE AT PC AS SHOWN (NO RELOCATION)
```

```
12238  046216  012737  000700  001616          MOV     #700;@#NEXPAR   ;RESET NEXT VALUE FOR PAR REGISTERS
12239  046224  005737  003644                  TST     @#PROT
12240  046230  001403                          BEQ     2$
12241  046232  012737  001600  001616          MOV     #1600,@#NEXPAR
12242  046240  105037  001601          2$:     CLRB    @#MMON          ;SET MEM MGMT ON IND = OFF
12243                                  ;;**********************************************************************
12244                                  .SBTTL  END OF SUB-PASS ROUTINE
12245                                  ;*      THIS ROUTINE SETSUP THE PSW AND MAINTENANCE REGISTERS
12246                                  ;*      FOR THE NEXT SUB-PASS. IT THEN STARTS THE PRINTER
12247                                  ;*      (IF NOT ON ACT11) FOR TYPING THE END OF SUB-PASS MESSAGE.
12248                                  ;;**********************************************************************
12249  046244                          END:
12250  046244  012737  064422  000004 END1:   MOV     #ERPRT,@#ERRVEC
12251  046252  012737  063616  000114          MOV     #.PARSRV,@#CACHVEC      ;RESTORE CACHE PARITY VECTOR
12252  046260  010137  000116                  MOV     R1,@#CACHVEC+2  ;AND PSW
12253  046264  005037  177776                  CLR     @#PSW           ;CLEAR MODE BITS IN PSW
12254  046270  004767  015020                  JSR     PC,CLRTBIT      ;GO CLEAR 'T' BIT IF SET
12255  046274  012706  001200                  MOV     #KERSTK,SP      ;SET KERNEL STACK PTR
12256  046300  032777  000100  133006          BIT     #100,@$TPS      ;CHECK IF OUTPUT DEVICE IS BUSY
12257  046306  001374                          BNE     .-6             ;IS AVAILABLE
12258  046310  105237  001630          1$:     INCB    @#SUBPASS
12259  046314  113702  001630                  MOVB    @#SUBPASS,R2
12260  046320  162702  000060                  SUB     #60,R2
12261  046324  022702  000006                  CMP     #6,R2                   ;END OF TEST?
12262  046330  001013                          BNE     2$              ;BRANCH IF NOT AT END
12263  046332  012737  000060  001630          MOV     #60,@#SUBPASS           ;INIT SUBPASS COUNT TO ASCII 0
12264  046340  005037  177750                  CLR     @#MAINT                 ;CLEAR MAINTENANCE REG
12265  046344  005037  001710                  CLR     @#$MAINT                ;CLEAR SOFTWARE VALUE
12266  046350  005046                          CLR     -(SP)
12267  046352  012746  046460                  MOV     #$EOP,-(SP)
12268  046356  000002                          RTI
12269  046360  006302          2$:     ASL     R2
12270  046362  012737  001570  000014          MOV     #$RTRN,@#TBITVEC        ;SET 'T' TRAP VECTOR
12271  046370  012737  001627  001340          MOV     #SUBPASS-1,@#$REG5
12272  046376  106277  132712                  ASRB    @$TPS
12273  046402  016246  064516                  MOV     PSWTAB(2),-(SP) ;PUSH NEXT PASS PSW ON STACK
12274  046406  012746  010000                  MOV     #LOOP,-(SP)     ;RESART PROGRAM AT LOOP
12275  046412  016237  064532  001710          MOV     MRGTAB(R2),@#$MAINT
12276  046420  016237  064532  177750          MOV     MRGTAB(R2),@#MAINT
12277  046426  105737  001602          3$:     TSTB    @#QV            ;QV PASS?
12278  046432  001011                          BNE     RTI1            ;BRANCH IF YES
12279  046434  122777  000200  132652          CMPB    #200,@$TPS      ;IS PRINTER READY?
12280  046442  001371                          BNE     3$              ;BRANCH IF NO
12281  046444  012737  065445  001340          MOV     #MSG20-1,@#$REG5
12282  046452  106277  132636                  ASRB    @$TPS           ;TYPE END SUBPASS MESSAGE
12283  046456  000002          RTI1:   RTI                             ;RESTART PROGRAM AT LOOP WITH NEW PSW
12284                                                                  ;(FROM TABLE BELOW)
12285
12286                                  ;;**********************************************************************
  (1)
  (1)                                   .SBTTL  END OF PASS ROUTINE
  (1)
  (1)                                   ;*INCREMENT THE PASS NUMBER ($PASS)
  (1)                                   ;*TYPE 'END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY''
  (1)                                   ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
  (1)                                   ;*IF THERES A MONITOR GO TO IT
```

```
(1)                                     ;*IF THERE ISN'T JUMP TO LOOP
(1)
(1)  046460                             $EOP:
(2)  046460  004737  056632                     JSR     PC,@#TYPTIME
(1)  046464  005067  132562                     CLR     $TSTNM          ;;ZERO THE TEST NUMBER
(1)  046470  005067  132702                     CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
(1)  046474  005267  132550                     INC     $PASS           ;;INCREMENT THE PASS NUMBER
(1)  046500  042767  100000  132542           BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
(1)  046506  005327                            DEC     (PC)+           ;;LOOP?
(1)  046510  000001                     $EOPCT: .WORD   1
(1)  046512  003063                             BGT     $DOAGN          ;;YES
(1)  046514  012737                             MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
(1)  046516  000001                     $ENDCT: .WORD   1
(1)  046520  046510                             $EOPCT
(2)  046522  104400  046530                     TYPE    ,65$            ;;TYPE ASCIZ STRING
(2)  046526  000407                             BR      64$             ;;GET OVER THE ASCIZ
(2)                                     ;;65$:  .ASCIZ  <12><15>/END PASS #/
(2)  046546                             64$:
(2)  046546  016746  132476                     MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
(2)                                                                     ;;TYPE PASS NUMBER
(2)  046552  104410                             TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
(2)  046554  104400  046562                     TYPE    ,67$            ;;TYPE ASCIZ STRING
(2)  046560  000421                             BR      66$             ;;GET OVER THE ASCIZ
(2)                                     ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
(2)  046624                             66$:
(2)  046624  016746  132434                     MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
(2)                                                                     ;;TOTAL NUMBER OF ERRORS
(2)  046630  104410                             TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
(1)  046632  104400  001407                     TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
(1)  046636  005067  132422                     CLR     $ERTTL          ;;CLEAR ERROR TOTAL
(1)  046642  013700  000042             $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
(1)  046646  001405                             BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
(1)  046650  000005                             RESET                   ;;CLEAR THE WORLD
(1)  046652  004710                     $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
(1)  046654  000240                             NOP                     ;;SAVE ROOM
(1)  046656  000240                             NOP                     ;;FOR
(1)  046660  000240                             NOP                     ;;ACT11
(1)  046662                             $DOAGN:
(1)  046662  000137  010000                     JMP     @#LOOP          ;;RETURN
(1)  046666     377     377     000     $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
(1)          046672                             .EVEN
12287                                   ;;*****************************************************************
12288                                   .SBTTL  RP11/RP03 HANDLER
12289                                   ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
12290                                   ;;*****************************************************************
12291  046672  104420                   RP3DRV: SAVREG
12292  046674  105037  002006                   CLRB    @#RP3HSTA               ;CLEAR DONE FLAG
12293  046700  032737  000400  002006           BIT     #BIT8,@#RP3HSTA         ;REPEAT FLAG SET?
12294  046706  001403                           BEQ     8$                      ;BRANCH IF NO
12295  046710  104422                           RESREG
12296  046712  000137  050544                   JMP     @#RP3RPT
12297  046716  013737  001654  001664   8$:     MOV     @#RNTBINX,@#RP311       ;SAVE RUN TABLE INDEX
12298  046724  032777  000020  132312           BIT     #SW4,@$SWRP             ;INHIBIT RND DSK ADR?
12299  046732  001403                           BEQ     1$                      ;BRANCH IF NO
12300  046734  005000                           CLR     R0
12301  046736  005001                           CLR     R1
```

```
12302  046740  000410                        BR      4$
12303  046742  004737  060562        1$:     JSR     PC,@#$RAND              ;GO GET RANDOM NUMBER
12304  046746  013700  001622                MOV     @#$HINUM,R0             ;GET HI NUMBER
12305  046752  013701  001620                MOV     @#$LONUM,R1             ;GET LO NUMBER
12306  046756  073027  177771                ASHC    #-7,R0                  ;ADJUST TO FORM CYL ADR
12307  046762  042700  177000        4$:     BIC     #177000,R0              ;GET RID OF UNUSED BITS
12308  046766  022700  000624                CMP     #624,R0                 ;LEGAL CYL?
12309  046772  100003                        BPL     5$                      ;BRANCH IF YES
12310  046774  062700  000624                ADD     #624,R0                 ;MAKE IT LEGAL
12311  047000  000770                        BR      4$
12312  047002  013702  001664        5$:     MOV     @#RP311,R2              ;GET RUN TABLE INDEX
12313  047006  016203  001732                MOV     RUNTBL(R2),R3           ;GET DEVICE ID
12314  047012  042703  000777                BIC     #777,R3                 ;ID ONLY
12315  047016  050300                        BIS     R3,R0                   ;COMBINE WITH CYL ADR
12316  047020  010062  001732                MOV     R0,RUNTBL(R2)           ;PUT BACK IN TABLE
12317  047024  072127  177775                ASH     #-3,R1                  ;GEN TRK-SECT ADR
12318  047030  010103                        MOV     R1,R3                   ;SAVE
12319  047032  042701  160377        6$:     BIC     #160377,R1              ;GET RID OF ALL BUT TRK
12320  047036  022701  011400                CMP     #11400,R1               ;LEGAL TRAK?
12321  047042  100003                        BPL     2$                      ;BRANCH IF YES
12322  047044  062701  011400                ADD     #11400,R1               ;MAKE IT LEGAL
12323  047050  000770                        BR      6$
12324  047052  042703  177760        2$:     BIC     #177760,R3              ;GET SECTOR ADR
12325  047056  022703  000011                CMP     #11,R3                  ;IS IT LEGAL?
12326  047062  100003                        BPL     3$                      ;BRANCH IF YES
12327  047064  062703  000011                ADD     #11,R3                  ;MAKE IT LEGAL
12328  047070  000770                        BR      2$
12329  047072  050301        3$:             BIS     R3,R1                   ;COMBINE TRK-SECT
12330  047074  010162  001750                MOV     R1,RUNTRAK(R2)          ;PUT IN TABLE
12331  047100  010037  002154                MOV     R0,@#RP3HDC             ;SAVE DESIRED CYL
12332  047104  010137  002222                MOV     R1,@#RP3DA              ;SAVE DSK ADR
12333  047110  112737  177775  002202        MOVB    #-3,@#RP3TRY            ;INIT TRY COUNT
12334  047116  032737  000040  172516        BIT     #BIT5,@#MMR3            ;MAP ON?
12335  047124  001405                        BEQ     7$                      ;BRANCH IF NO
12336  047126  005046                        CLR     -(SP)                   ;PUT DEVICE ID ON STACK
12337  047130  013746  002042                MOV     @#RP3OLD,-(SP)          ;PUT ADR OF BUS ADR ON STK
12338  047134  004737  063022                JSR     PC,@#GETMAP             ;GET MAP REGISTER
12339  047140  012737  000103  001662  7$:   MOV     #103,@#RP310            ;GET FUNCTION
12340  047146  013700  002044                MOV     @#RP3OLD+2,R0           ;GET BAE BITS
12341  047152  072027  000004                ASH     #4,R0                   ;SHIFT TO BITS 4 & 5
12342  047156  050037  001662                BIS     R0,@#RP310              ;COMBINE WITH FUNCTION
12343  047162  010037  002044                MOV     R0,@#RP3OLD+2
12344  047166  013700  002122                MOV     @#RP3UNIT,R0
12345  047172  072027  000010                ASH     #10,R0                  ;SHIFT UNIT NO TO RIGHT BITS
12346  047176  050037  001662                BIS     R0,@#RP310              ;COMBINE WITH FUNC & BAE
12347  047202  010037  002122                MOV     R0,@#RP3UNIT
12348  047206  104422                        RESREG
12349  047210  005777  132774  RP3WTRY:TST   @RP3DS                           ;IS DRIVE READY?
12350  047214  100375                        BPL     RP3WTRY                 ;BRANCH IF NO
12351  047216  053777  002122  132770        BIS     @#RP3UNIT,@RP3CS        ;SET UNIT BITS
12352  047224  004737  047256                JSR     PC,@#LDRP3              ;LOAD RP3 REGISTERS
12353  047230  012777  050574  132770        MOV     #RP3SRV,@RP3VEC         ;SET VECTOR
12354  047236  005077  132766                CLR     @RP3PSW
12355  047242  005037  002170                CLR     @#RP3FUN                ;SET FUNCTION TO WRITE
12356  047246  013777  001662  132740        MOV     @#RP310,@RP3CS          ;LOAD FUNCT AND GO
12357  047254  000002                        RTI                             ;RETURN
```

```
12358  047256  013777  002152  132736  LDRP3:  MOV     @#RP3HDA,@RP3DA         ;LOAD DSK ADR
12359  047264  013777  002154  132732          MOV     @#RP3HDC,@RP3DC         ;LOAD CYL ADR
12360  047272  013777  002026  132716          MOV     @#RP3HWC,@RP3WC         ;LOAD WORD COUNT
12361  047300  013777  002042  132712          MOV     @#RP3OLD,@RP3BA         ;LOAD BUS ADR
12362  047306  000207                          RTS     PC                      ;RETURN
12363
12364                                  ;;*****************************************************************
12365                                  .SBTTL  RK11/RK05 HANDLER
12366                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
12367                                  ;;*****************************************************************
12368  047310  104420                  RKDRV:  SAVREG
12369  047312  105037  002010                  CLRB    @#RKHSTAT               ;CLEAR DONE FLAG IN HANDLER STAT
12370  047316  032737  000400  002010          BIT     #BIT8,@#RKHSTAT         ;REPEAT FLAG SET?
12371  047324  001403                          BEQ     5$                      ;BRANCH IF NO
12372  047326  104422                          RESREG
12373  047330  000137  051362                  JMP     @#RKRPT
12374  047334  013737  001654  001670  5$:     MOV     @#RNTBINX,@#RK11        ;SAVE RUN TABLE INDEX
12375  047342  105037  002010                  CLRB    @#RKHSTAT               ;CLEAR DONE FLAG IN HANDLER STAT
12376  047346  032777  000020  131670          BIT     #SW4,@$SWRP             ;RANDOM DSK ADDRESS?
12377  047354  001403                          BEQ     6$                      ;BRANCH IF YES
12378  047356  005000                          CLR     R0                      ;CLEAR REGISTERS
12379  047360  005001                          CLR     R1
12380
12381  047362  000404                          BR      7$                      ;FOR ADDRESS CHECKING
12382  047364  004737  060562          6$:     JSR     PC,@#$RAND              ;GET RANDOM NUMBER
12383  047370  013700  001622                  MOV     @#$HINUM,R0             ;GET HIGH NUMBER
12384  047374  072027  177775          7$:     ASH     #-3,R0                  ;ADJUST TO FORM
12385                                                                          ;CYLINDER ADDRESS
12386  047400  010001                          MOV     R0,R1                   ;SAVE IN R1
12387  047402  042701  160037          4$:     BIC     #160037,R1              ;GET RID OF SURF-SECT BITS
12388  047406  022701  014300                  CMP     #14300,R1               ;IS IT A LEGAL CYLINDER?
12389  047412  100003                          BPL     3$                      ;BRANCH IF YES
12390  047414  062701  014340                  ADD     #14340,R1               ;ADD MAXIMUM CYLINDER
12391  047420  000770                          BR      4$                      ;TRY AGAIN
12392  047422  072127  177773          3$:     ASH     #-5,R1                  ;ADJUST CYLINDER ADDRESS
12393  047426  013702  001670                  MOV     @#RK11,R2               ;GET RUN TABLE INDEX
12394  047432  016203  001732                  MOV     RUNTBL(R2),R3           ;GET RUN TABLE ENTRY
12395  047436  042703  000777                  BIC     #777,R3                 ;SAVE ID AND UNIT NO.
12396  047442  050103                          BIS     R1,R3                   ;INSERT CYLINDER ADDR
12397  047444  010362  001732                  MOV     R3,RUNTBL(R2)           ;ENTER CYLINDER ADR IN RUN TABLE
12398  047450  072027  177770                  ASH     #-10,R0                 ;GENER SECTOR-SURF ADDRESS
12399  047454  042700  177740                  BIC     #177740,R0              ;GET RID OF EXTRA BITS
12400  047460  010003                          MOV     R0,R3                   ;SAVE
12401  047462  042700  000020                  BIC     #BIT4,R0                ;GET RID OF SURFACE BIT
12402  047466  022700  000012                  CMP     #12,R0                  ;IS SECTOR ADDRESS LEGAL?
12403  047472  100004                          BPL     1$                      ;BRANCH IF YES
12404  047474  062700  000012                  ADD     #12,R0                  ;MAKE IT LEGAL
12405  047500  042700  000020                  BIC     #BIT4,R0                ;GET RID OF CARRY FROM ADD
12406  047504  042703  000017          1$:     BIC     #17,R3                  ;GET SURFACE ADDRESS
12407  047510  050300                          BIS     R3,R0                   ;GENER COMP SECT-SURF ADDRESS
12408  047512  010062  001750                  MOV     R0,RUNTRAK(R2)          ;SAVE IN RUN TRAK TABLE
12409  047516  072127  000005                  ASH     #5,R1                   ;ADJUST CYLINDER ADDRESS
12410  047522  050100                          BIS     R1,R0                   ;CONCATINATE TRK & SECT ADDR
12411  047524  013701  002124                  MOV     @#RKUNIT,R1             ;GET UNIT NUMBER
12412  047530  072127  000015                  ASH     #15,R1          ;ADJUST
12413  047534  050100                          BIS     R1,R0                   ;CONCATINATE UNIT,TRK,SURF,SECT
```

```
12414  047536  010037  002156                  MOV     R0,@#RKHDA              ;SAVE
12415  047542  112737  177775  002203          MOVB    #-3,@#RKTRY             ;SET RETRY COUNT
12416  047550  032737  000040  172516          BIT     #BIT5,@#MMR3            ;MAP ON?
12417  047556  001406                          BEQ     2$                      ;BRANCH IF NO
12418  047560  012746  000001                  MOV     #1,-(SP)                ;PUT DEVICE ID ON STACK
12419  047564  012746  002046                  MOV     #RKOLD,-(SP)            ;PUT ADDRESS OF ADR ON STACK
12420  047570  004737  063022                  JSR     PC,@#GETMAP             ;GET MAP REG
12421  047574  012767  000103  132064  2$:     MOV     #103,RK10               ;SET FUNCTION
12422  047602  013700  002050                  MOV     @#RKOLD+2,R0            ;GET BA EXTENDED
12423  047606  072027  000004                  ASH     #4,R0                   ;ADJUST
12424  047612  050037  001666                  BIS     R0,@#RK10               ;PUT IN WITH FUNCTION
12425  047616  010037  002050                  MOV     R0,@#RKOLD+2            ;SAVE IN MEMORY
12426  047622  104422                          RESREG
12427  047624  013777  002156  132412  RKWTRY: MOV     @#RKHDA,@RKDA           ;LOAD DISK ADDRESS
12428  047632  032777  000100  132372          BIT     #BIT6,@RKDS             ;UNIT READY?
12429  047640  001774                          BEQ     .-6                     ;BRANCH IF NO
12430  047642  013777  002030  132370          MOV     @#RKHWC,@RKWC           ;LOAD WORD COUNT
12431  047650  013777  002046  132364          MOV     @#RKOLD,@RKBA           ;LOAD BUS ADDRESS
12432  047656  012777  051412  132362          MOV     #RKSRV,@RKVEC           ;LOAD INTERRUPT VECTOR
12433  047664  005077  132360                  CLR     @RKPSW
12434  047670  005037  002172                  CLR     @#RKFUN                 ;SET FUNCTION TO WRITE
12435  047674  013777  001666  132334          MOV     @#RK10,@RKCS            ;LOAD FUNCTION AND GO
12436  047702  000006                          RTT                             ;RETURN
12437
12438                                  ;;*******************************************************************
12439                                  .SBTTL  RH70/RP04 HANDLER
12440                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
12441                                  ;;*******************************************************************
12442  047704  104420                  RP4DRV: SAVREG
12443  047706  105037  002016                  CLRB    @#RP4HSTA               ;CLEAR DONE FLAG
12444  047712  032737  000400  002016          BIT     #BIT8,@#RP4HST          ;REPEAT FLAG SET?
12445  047720  001403                          BEQ     6$                      ;BRANCH IF NO
12446  047722  104422                          RESREG
12447  047724  000137  052240                  JMP     @#RP4RPT
12448  047730  013737  001654  001672  6$:     MOV     @#RNTBINX,@#RP411       ;SAVE RUN TABLE INDEX
12449  047736  105037  002016                  CLRB    @#RP4HSTA               ;CLEAR DONE FLAG
12450  047742  032777  000020  131274          BIT     #SW4,@$SWRP             ;RANDOM DSK ADDRESS?
12451  047750  001403                          BEQ     1$                      ;BRANCH IF YES
12452  047752  005000                          CLR     R0
12453  047754  005001                          CLR     R1
12454  047756  000410                          BR      4$
12455  047760  004737  060562          1$:     JSR     PC,@#$RAND              ;GET RANDOM NUMBER
12456  047764  013700  001622                  MOV     @#$HINUM,R0             ;GET HI NUMBER
12457  047770  013701  001620                  MOV     @#$LONUM,R1             ;GET LO NUMBER
12458  047774  073027  177771                  ASHC    #-7,R0                  ;ADJUST TO FORM CYL. ADR.
12459  050000  042700  177000          4$:     BIC     #177000,R0              ;GET RID OF UNUSED BITS
12460  050004  022700  000631                  CMP     #631,R0                 ;LEGAL CYLINDER
12461  050010  100003                          BPL     5$                      ;BRANCH IF YES
12462  050012  062700  000631                  ADD     #631,R0                 ;MAKE IT LEGAL
12463  050016  000770                          BR      4$
12464
12465
12466  050020  013702  001672          5$:     MOV     @#RP411,R2              ;GET RUN TABLE INDEX
12467
12468  050024  016203  001732                  MOV     RUNTBL(R2),R3           ;GET DEVICE ID
12469  050030  042703  000777                  BIC     #777,R3                 ;SAVE ID ONLY
```

```
12470  050034  050003                          BIS     R0,R3                   ;COMBINE WITH CYL ADR
12471  050036  010362  001732                  MOV     R3,RUNTBL(R2)           ;PUT IN RUN TABLE
12472  050042  072127  177775                  ASH     #-3,R1                  ;GEN TRAK-SECT ADR
12473  050046  042701  160340                  BIC     #160340,R1              ;GET RID OF UNUSED BITS
12474  050052  010103                          MOV     R1,R3                   ;SAVE
12475  050054  042701  000037                  BIC     #37,R1                  ;GET RID OF SECT BITS
12476  050060  022701  011000                  CMP     #11000,R1               ;LEGAL TRAK?
12477  050064  100004                          BPL     2$                      ;BRANCH IF YES
12478  050066  062701  011000                  ADD     #11000,R1               ;MAKE IT LEGAL
12479  050072  042701  020000                  BIC     #BIT13,R1               ;GET RID OF ADD CARRY
12480  050076  042703  177740          2$:     BIC     #177740,R3              ;GET SECTOR ADR
12481  050102  022703  000025                  CMP     #25,R3                  ;LEGAL SECTOR
12482  050106  100004                          BPL     3$                      ;BRANCH IF YES
12483  050110  062703  000025                  ADD     #25,R3                  ;MAKE IT LEGAL
12484  050114  042703  000040                  BIC     #BIT5,R3                ;GET RID OF ADD CARRY
12485  050120  050301                  3$:     BIS     R3,R1                   ;COMBINE TRAK-SECTOR
12486  050122  010162  001750                  MOV     R1,RUNTRAK(R2)          ;PUT TRAK-SECT IN TABLE
12487  050126  010037  002164                  MOV     R0,@#RP4HDC             ;SAVE CYLINDER ADR
12488  050132  010137  002162                  MOV     R1,@#RP4HDA             ;SAVE TRAK-SECTOR ADR
12489  050136  112737  177775  002205          MOVB    #-3,@#RP4TRY            ;SET TRY COUNT
12490  050144  104422                          RESREG
12491  050146  004767  000026          RP4WTRY:JSR     PC,LDRP4                ;LOAD RP4 REGISTERS
12492  050152  012777  052264  132126          MOV     #RP4SRV,@RP4VEC ;LOAD INTERRUPT VECTOR
12493  050160  005077  132124                  CLR     @RP4PSW                 ;
12494  050164  005037  002176                  CLR     @#RP4FUN                ;SET FUNCTION TO WRITE
12495  050170  112777  000161  132054          MOVB    #161,@RP4CS1            ;LOAD FUNCTION AND GO
12496  050176  000002                          RTI                             ;RETURN
12497
12498  050200  013777  002132  132056  LDRP4:  MOV     @#RP4UNIT,@RP4CS2       ;LOAD UNIT NUMBER
12499  050206  012777  010000  132070          MOV     #BIT12,@RP4OF           ;SET FORMAT TO 16 BIT
12500  050214  013777  002164  132052          MOV     @#RP4HDC,@RP4DC         ;LOAD CYLINDER ADR
12501  050222  013777  002162  132032          MOV     @#RP4HDA,@RP4DA         ;LOAD TRAK-SECTOR
12502  050230  013777  002036  132016          MOV     @#RP4HWC,@RP4WC         ;LOAD WORD COUNT
12503  050236  013777  002064  132014          MOV     @#RP4OLD+2,@RP4BAE      ;LOAD EXTENDED ADR BITS
12504  050244  013777  002062  132004          MOV     @#RP4OLD,@RP4BA         ;LOAD BUS ADR
12505  050252  000207                          RTS     PC                      ;RETURN
12506
12507                                  ;;*****************************************************************
12508                                  .SBTTL  RH70/RS04 HANDLER
12509                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
12510                                  ;;*****************************************************************
12511  050254  104420                  RSDRV:  SAVREG
12512  050256  105037  002020                  CLRB    @#RSHSTAT               ;CLEAR DONE FLAG
12513  050262  032737  000400  002020          BIT     #BIT8,@#RSHSTAT         ;REPEAT FLAG SET?
12514  050270  001403                          BEQ     3$                      ;BRANCH IF NO
12515  050272  104422                          RESREG
12516  050274  000137  052750                  JMP     @#RSRPT
12517  050300  013737  001654  001674  3$:     MOV     @#RNTBINX,@#RS11        ;SAVE RUN TABLE INDEX
12518  050306  032777  000020  130730          BIT     #SW4,@$SWRP             ;RANDOM DSK ADR?
12519  050314  001403                          BEQ     1$                      ;BRANCH IF YES
12520  050316  005000                          CLR     R0
12521  050320  005001                          CLR     R1
12522  050322  000407                          BR      4$
12523  050324  004737  060562          1$:     JSR     PC,@#$RAND              ;GET RANDOM NUMBER
12524  050330  013700  001622                  MOV     @#$HINUM,R0
12525  050334  072027  177774                  ASH     #-4,R0
```

```
12526  050340  010001                      MOV     R0,R1                   ;SAVE RANDOM NUMBER
12527  050342  042700  170077        4$:   BIC     #170077,R0              ;GET TRACK ADR
12528  050346  022700  007600              CMP     #7600,R0                ;IS IT LEGAL?
12529  050352  100003                      BPL     5$                      ;BRANCH IF YES
12530  050354  062700  007600              ADD     #7600,R0                ;MAKE IT LEGAL
12531  050360  000770                      BR      4$
12532  050362  013702  001674        5$:   MOV     @#RS11,R2               ;GET RUN TABLE INDEX
12533  050366  072027  177772              ASH     #-6,R0                  ;ADJUST TRACK ADR
12534  050372  110062  001732              MOVB    R0,RUNTBL(R2)           ;SAVE TRAK ADR IN RUN TBL
12535  050376  042701  177700        6$:   BIC     #177700,R1              ;GET SECTOR ADR
12536  050402  022701  000077              CMP     #77,R1                  ;IS IT LEGAL?
12537  050406  100003                      BPL     2$                      ;BRANCH IF YES
12538  050410  062701  000077              ADD     #77,R1                  ;MAKE IT LEGAL
12539  050414  000770                      BR      6$
12540  050416  010162  001750        2$:   MOV     R1,RUNTRAK(R2)          ;SAVE IN RUN TRAK TABLE
12541  050422  072027  000006              ASH     #6,R0                   ;ADJUST TRACK ADDR
12542  050426  050100                      BIS     R1,R0                   ;COMBINE SECTOR TRAK
12543  050430  010037  002166              MOV     R0,@#RSHDA              ;SAVE AS DSK ADR
12544  050434  112737  177775  002206      MOVB    #-3,@#RSTRY             ;SET TRY COUNT
12545  050442  104422                      RESREG
12546  050444  004737  050504  RSWTRY:     JSR     PC,@#LDRS               ;GO LOAD REGISTERS
12547  050450  012777  052774  131656      MOV     #RSSRV,@RSVEC           ;SET INTERRUPT VECTOR
12548  050456  005077  131654              CLR     @RSPSW
12549  050462  005037  002200              CLR     @#RSFUN                 ;SET FUNCTION TO WRITE
12550  050466  105777  131636        1$:   TSTB    @RSDS                   ;IS DRIVE READY?
12551  050472  001775                      BEQ     1$                      ;BRANCH IF NO
12552  050474  112777  000161  131610      MOVB    #161,@RSCS1             ;LOAD FUNCTION AND GO
12553  050502  000002                      RTI
12554
12555  050504  013777  002134  131612  LDRS:   MOV     @#RSUNIT,@RSCS2         ;LOAD UNIT NUMBER
12556  050512  013777  002166  131602          MOV     @#RSHDA,@RSDA           ;LOAD DSK ADR
12557  050520  013777  002040  131566          MOV     @#RSHWC,@RSWC           ;LOAD WORD COUNT
12558  050526  013777  002070  131564          MOV     @#RSOLD+2,@RSBAE        ;LOAD EXTENDED ADDRESS
12559  050534  013777  002066  131554          MOV     @#RSOLD,@RSBA           ;LOAD BUS ADDRESS
12560  050542  000207                          RTS     PC                      ;RETURN
12561
12562                          ;;*******************************************************************************
12563                          .SBTTL  RP11/RP03 SERVICE ROUTINE
12564                          ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
12565                          ;;*******************************************************************************
12566  050544  000005          RP3RPT: RESET
12567  050546  005337  002170          DEC     @#RP3FUN                ;RESTORE FUNCTION
12568  050552  022737  000001  002170  CMP     #1,@#RP3FUN             ;WHAT IS IT?
12569  050560  001472                  BEQ     RP31                    ;BRANCH IF WC
12570  050562  002402                  BLT     1$                      ;BRANCH IF WRITE
12571  050564  000137  047210          JMP     @#RP3WTRY               ;BRANCH TO READ
12572  050570  000167  000414  1$:     JMP     RP33
12573  050574  005237  002170  RP3SRV: INC     @#RP3FUN                ;INCREMENT FUNCTION
12574  050600  022737  000002  002170  CMP     #2,@#RP3FUN             ;WHAT IS IT?
12575  050606  001501                  BEQ     RP3WCK                  ;BRANCH TO WRITE CHECK
12576  050610  100002                  BPL     .+6
12577  050612  000137  051250          JMP     @#RP3READ
12578
12579                          ;FUNCTION JUST EXECUTED WAS A WRITE
12580  050616  032737  000400  002006  BIT     #BIT8,@#RP3HSTAT        ;REPEAT FLAG SET?
12581  050624  001036                  BNE     RP3LOOP                 ;BRANCH IF YES
```

```
12582  050626  005777  131362                 TST     @RP3CS                  ;ANY ERRORS?
12583  050632  100045                         BPL     RP31                    ;BRANCH IF NO
12584  050634  105737  002202                 TSTB    @#RP3TRY                ;TRIED 3 TIMES?
12585  050640  001415                         BEQ     RP3ERR                  ;BRANCH IF YES
12586  050642  112777  000001  131344         MOVB    #BIT0,@RP3CS            ;CLEAR THE DRIVE
12587  050650  105777  131340                 TSTB    @RP3CS                  ;CONTROLLER READY?
12588  050654  100375                         BPL     .-4                     ;BRANCH IF NO
12589  050656  105237  002202                 INCB    @#RP3TRY                ;INCREMENT TRY COUNT
12590  050662  013746  177776                 MOV     @#PSW,-(SP)             ;MAINTAIN SAME PSW
12591  050666  012746  047210                 MOV     #RP3WTRY,-(SP)          ;SET RETRY ADDRESS
12592  050672  000002                         RTI                             ;RETURN
12593  050674  012737  100200  002006 RP3ERR: MOV     #100200,@#RP3HSTA       ;SET ERROR BIT IN HAND. STA
12594  050702  010046                         MOV     R0,-(SP)                ;SAVE R0
12595  050704  013700  001664                 MOV     @#RP311,R0              ;GET RUNTABLE INDEX
12596  050710  052760  100000  001750         BIS     #BIT15,RUNTRAK(R0)      ;SET ERROR BIT
12597  050716  012600                         MOV     (SP)+,R0                ;RESTORE R0
12598  050720  000002                         RTI                             ;RETURN
12599
12600  050722  012737  100200  002006 RP3LOOP:MOV     #100200,@#RP3HSTAT      ;SET DONE AND ERROR
12601  050730  005777  131260                 TST     @RP3CS                  ;ANY ERRORS?
12602  050734  100403                         BMI     1$                      ;BRANCH IF YES
12603  050736  042737  100000  002006         BIC     #BIT15,@#RP3HSTAT       ;CLEAR ERROR BIT
12604  050744  000002                 1$:     RTI                             ;RETURN
12605                                 ;WRITE WAS OK- NOW DO A WRITE CHECK
12606  050746  112737  177775  002202 RP31:   MOVB    #-3,@#RP3TRY            ;INIT TRY COUNT
12607  050754  012737  000107  001662         MOV     #107,@#RP310            ;SET FUNCTION
12608  050762  053737  002044  001662         BIS     @#RP3OLD+2,@#RP310      ;SET BAE BITS
12609  050770  053737  002122  001662         BIS     @#RP3UNIT,@#RP310       ;SET UNIT BITS
12610  050776  004737  047256         RP32:   JSR     PC,@#LDRP3              ;LOAD RP3 REGISTERS
12611  051002  013777  001662  131204         MOV     @#RP310,@RP3CS          ;LOAD FUNCTION AND GO
12612  051010  000002                         RTI                             ;RETURN
12613
12614                                 ;FUNCTION JUST EXECUTED WAS A WRITE CHECK
12615  051012  032737  000400  002006 RP3WCK: BIT     #BIT8,@#RP3HSTAT                ;REPEAT FLAG SET?
12616  051020  001340                         BNE     RP3LOOP                 ;BRANCH IF YES
12617  051022  005777  131166                 TST     @RP3CS                  ;ANY ERRORS?
12618  051026  100031                         BPL     1$                      ;BRANCH IF NO
12619  051030  005737  001664                 TST     @#RP311                 ;FIRST 2K?
12620  051034  001422                         BEQ     4$                      ;BRANCH IF YES
12621  051036  105737  002202         5$:     TSTB    @#RP3TRY                ;TRIED 3 TIMES?
12622  051042  001714                         BEQ     RP3ERR                  ;BRANCH IF YES
12623  051044  005337  002170                 DEC     @#RP3FUN                ;RESTORE FUNCTION
12624  051050  112777  000001  131136         MOVB    #BIT0,@RP3CS            ;CLEAR THE DRIVE
12625  051056  105777  131132                 TSTB    @RP3CS                  ;CONTROLLER READY?
12626  051062  100375                         BPL     .-4                     ;BRANCH IF NO
12627  051064  105237  002202                 INCB    @#RP3TRY                ;INCREMENT TRY COUNT
12628  051070  013746  177776                 MOV     @#PSW,-(SP)
12629  051074  012746  050776                 MOV     #RP32,-(SP)
12630  051100  000002                         RTI                             ;GO TRY AGAIN
12631  051102  032777  000010  131102 4$:     BIT     #BIT3,@RP3ER            ;WRITE CHECK ERROR?
12632  051110  001752                         BEQ     5$                      ;BRANCH IF NO
12633
12634                                 ;WRITE CHECK OK- NOW DO A READ
12635  051112  112737  177775  002202 1$:     MOVB    #-3,@#RP3TRY            ;RESTORE TRY COUNT
12636  051120  032737  000040  172516         BIT     #BIT5,@#MMR3            ;MAP ON?
12637  051126  001407                         BEQ     2$                      ;BRANCH IF NO
```

```
12638  051130  005046                          CLR     -(SP)                   ;PUT DEVICE ID ON STACK
12639  051132  004737  063274                  JSR     PC,@#GIVEMAP            ;RETURN MAP REGISTER
12640  051136  012746  002072                  MOV     #RP3NWL,-(SP)           ;PUT ADR OF BUS ADR ON STK
12641  051142  004737  063022                  JSR     PC,@#GETMAP             ;GET MAP REGISTERS
12642  051146  010046                  2$:     MOV     R0,-(SP)                ;SAVE R0
12643  051150  013700  002074                  MOV     @#RP3NWH,R0             ;GET BAE BITS
12644  051154  072027  000004                  ASH     #4,R0                   ;ADJUST
12645  051160  010037  002074                  MOV     R0,@#RP3NWH             ;SAVE
12646  051164  012600                          MOV     (SP)+,R0                ;RESTORE R0
12647  051166  012737  000105  001662          MOV     #105,@#RP310            ;SET FUNCTION
12648  051174  053737  002074  001662          BIS     @#RP3NWH,@#RP310        ;SET BAE BITS
12649  051202  053737  002122  001662          BIS     @#RP3UNIT,@#RP310       ;SET UNIT NUMBER
12650  051210  013777  002152  131004  RP33:   MOV     @#RP3HDA,@RP3DA         ;LOAD DSK ADR
12651  051216  013777  002154  131000          MOV     @#RP3HDC,@RP3DC         ;LOAD CYL
12652  051224  013777  002026  130764          MOV     @#RP3HWC,@RP3WC         ;LOAD WORD COUNT
12653  051232  013777  002072  130760          MOV     @#RP3NWL,@RP3BA         ;LOAD BUS ADR
12654  051240  013777  001662  130746          MOV     @#RP310,@RP3CS          ;LOAD FUNCTION AND GO
12655  051246  000002                          RTI                             ;RETURN
12656
12657                                  ;FUNCTION JUST EXECUTED WAS A READ
12658  051250  032737  000400  002006  RP3READ:BIT     #BIT8,@#RP3HSTAT                ;REPEAT FLAG SET?
12659  051256  001221                          BNE     RP3LOOP                 ;BRANCH IF YES
12660  051260  005777  130730                  TST     @RP3CS                  ;ANY ERRORS?
12661  051264  100022                          BPL     1$                      ;BRANCH IF NO
12662  051266  105737  002202                  TSTB    @#RP3TRY                ;TRIED 3 TIMES?
12663  051272  001600                          BEQ     RP3ERR                  ;BRANCH IF YES
12664  051274  005337  002170                  DEC     @#RP3FUN                ;RESTORE FUNCTION
12665  051300  112777  000001  130706          MOVB    #BIT0,@RP3CS            ;CLEAR THE DRIVE
12666  051306  105777  130702                  TSTB    @RP3CS                  ;CONTROLLER READY?
12667  051312  100375                          BPL     .-4                     ;BRANCH OF NO
12668  051314  105237  002202                  INCB    @#RP3TRY                ;INCREMENT TRY COUNT
12669  051320  013746  177776                  MOV     @#PSW,-(SP)
12670  051324  012746  051210                  MOV     #RP33,-(SP)
12671  051330  000002                          RTI                             ;GO TRY AGAIN
12672  051332  032737  000040  172516  1$:     BIT     #BIT5,@#MMR3            ;MAP ON?
12673  051340  001404                          BEQ     2$                      ;BRANCH IF NO
12674  051342  005046                          CLR     -(SP)                   ;PUT DEVICE ID IN STK
12675  051344  004737  063274                  JSR     PC,@#GIVEMAP            ;RETURN MAP REGISTERS
12676  051350  005726                          TST     (SP)+                   ;RESTORE STACK
12677  051352  112737  000200  002006  2$:     MOVB    #200,@#RP3HSTA          ;SET DONE FLAG
12678  051360  000002                          RTI                             ;RETURN
12679                                  ;;*****************************************************************
12680                                  .SBTTL  RK11/RK05 SERVICE ROUTINE
12681                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
12682                                  ;;*****************************************************************
12683  051362  000005                  RKRPT:  RESET
12684  051364  005337  002172                  DEC     @#RKFUN                 ;RESTORE FUNCTION
12685  051370  022737  000001  002172          CMP     #1,@#RKFUN              ;WHAT IS IT?
12686  051376  001475                          BEQ     RK1                     ;BRANCH IF WC
12687  051400  002402                          BLT     1$                      ;BRANCH IF WRITE
12688  051402  000137  047624                  JMP     @#RKWTRY                ;IT WAS A WRITE
12689  051406  000137  052044          1$:     JMP     @#RK3
12690  051412  062737  000001  002172  RKSRV:  ADD     #1,@#RKFUN              ;FIND OUT WHAT FUNCTION
12691                                                                          ;WAS EXECUTED
12692  051420  022737  000002  002172          CMP     #2,@#RKFUN              ;WAS IT A WRITE CHECK?
12693  051426  001507                          BEQ     RKWRCK                  ;BRANCH IF YES
```

```
12694  051430  100002                          BPL     .+6                     ;BRANCH IF IT WAS A WRITE
12695  051432  000137  052076                  JMP     @#RKREAD
12696
12697                                  ;FUNCTION JUST EXECUTED WAS A WRITE. ANY ERRORS?
12698  051436  032737  000400  002010          BIT     #BIT8,@#RKHSTAT         ;REPEAT FLAG SET?
12699  051444  001040                          BNE     RKLOOP                  ;BRANCH IF YES
12700  051446  005777  130564                  TST     @RKCS                   ;ANY ERRORS?
12701  051452  100047                          BPL     RK1                     ;BRANCH IF NO
12702  051454  105737  002203                  TSTB    @#RKTRY                 ;TRYED 3 TIMES?
12703  051460  001417                          BEQ     RKERR                   ;BRANCH IF YES
12704  051462  012777  000001  130546          MOV     #1,@RKCS                ;CLEAR THE ERROR
12705  051470  004737  052222                  JSR     PC,@#TIMER              ;WAIT A LITTLE
12706  051474  105777  130536                  TSTB    @RKCS                   ;WAIT FOR CONT CLR TO FINISH
12707  051500  100375                          BPL     .-4
12708  051502  105237  002203                  INCB    @#RKTRY                 ;INCREMENT TRY COUNT
12709  051506  013746  177776                  MOV     @#PSW,-(SP)
12710  051512  012746  047624                  MOV     #RKWTRY,-(SP)
12711  051516  000002                          RTI
12712  051520  012737  100200  002010  RKERR:  MOV     #100200,@#RKHSTAT       ;SET ERROR & DONE FLAG
12713  051526  010046                          MOV     R0,-(SP)                ;SAVE R0
12714  051530  013700  001670                  MOV     @#RK11,R0               ;GET SAVED RUN TABLE INDEX
12715  051534  052760  100000  001750          BIS     #BIT15,RUNTRAK(R0)      ;SET ERROR BIT IN RUN TABLE
12716  051542  012600                          MOV     (SP)+,R0                ;RESTORE R0
12717  051544  000002                          RTI                             ;RETURN
12718
12719  051546  012737  100200  002010  RKLOOP:MOV     #100200,@#RKHSTAT       ;SET DONE AND ERROR BITS
12720  051554  005777  130456                  TST     @RKCS                   ;ANY ERRORS?
12721  051560  100403                          BMI     1$                      ;BRANCH IF YES
12722  051562  042737  100000  002010          BIC     #BIT15,@#RKHSTAT        ;CLEAR ERROR BIT
12723  051570  000002                  1$:     RTI                             ;RETURN
12724                                  ;WRITE WAS OK, NOW DO A WRITE CHECK
12725  051572  112737  177775  002203  RK1:    MOVB    #-3,@#RKTRY             ;RESTORE TRY COUNT
12726  051600  012767  000507  130060          MOV     #507,RK10               ;SET FUNCTION TO WRITE
12727  051606  053767  002050  130052          BIS     @#RKOLD+2,RK10          ;SET BA EXT BITS
12728  051614  013777  002156  130422  RK2:    MOV     @#RKHDA,@RKDA           ;LOAD DISK ADDRESS
12729  051622  013777  002030  130410          MOV     @#RKHWC,@RKWC           ;LOAD WORD COUNT
12730  051630  013777  002046  130404          MOV     @#RKOLD,@RKBA           ;LOAD BUS ADDRESS
12731  051636  016777  130024  130372          MOV     RK10,@RKCS              ;START FUNCTION
12732  051644  000002                          RTI                             ;RETURN
12733
12734                                  ;FUNCTION JUST EXECUTED WAS A WRITE CHECK. ANY ERRORS?
12735  051646  032737  000400  002010  RKWRCK:BIT     #BIT8,@#RKHSTAT                 ;REPEAT FLAG SET?
12736  051654  001334                          BNE     RKLOOP                  ;BRANCH IF YES
12737  051656  005777  130354                  TST     @RKCS                   ;ANY ERRORS?
12738  051662  100033                          BPL     1$                      ;BRANCH IF NO
12739  051664  005737  001670                  TST     @#RK11                  ;FIRST 2K?
12740  051670  001424                          BEQ     4$                      ;BRANCH IF YES
12741  051672  105737  002203          5$:     TSTB    @#RKTRY                 ;TRYED 3 TIMES?
12742  051676  001710                          BEQ     RKERR                   ;BRANCH IF YES
12743  051700  005337  002176                  DEC     @#RP4FUN                        ;SET FUNCTION BACK TO WC
12744  051704  012777  000001  130324          MOV     #1,@RKCS                ;CLEAR THE ERROR
12745  051712  004737  052222                  JSR     PC,@#TIMER              ;WAIT A LITTLE
12746  051716  105777  130314                  TSTB    @RKCS                   ;WAIT FOR CLR TO FINISH
12747  051722  100375                          BPL     .-4
12748  051724  105237  002203                  INCB    @#RKTRY                 ;INCREMENT TRY COUNT
12749  051730  013746  177776                  MOV     @#PSW,-(SP)
```

```
12750  051734  012746  051614                  MOV     #RK2,-(SP)
12751  051740  000002                          RTI
12752  051742  032777  040000  130266  4$:     BIT     #BIT14,@RKCS            ;HARD ERROR?
12753  051750  001350                          BNE     5$                      ;BRANCH IF YES
12754
12755                                  ;WRITE CHECK WAS OK, NOW DO A READ.
12756  051752  112737  177775  002203  1$:     MOVB    #-3,@#RKTRY             ;RESTORE TRY COUNT
12757  051760  032737  000040  172516          BIT     #BIT5,@#MMR3            ;MAP ON?
12758  051766  001410                          BEQ     2$                      ;BRANCH IF NO
12759  051770  012746  000001                  MOV     #1,-(SP)                ;PUT DEVICE ID ON STACK
12760  051774  004767  011274                  JSR     PC,GIVEMAP              ;RELINQUISH MAP REG
12761  052000  012746  002076                  MOV     #RKNEWL,-(SP)           ;PUT ADR OF BADR ON STACK
12762  052004  004737  063022                  JSR     PC,@#GETMAP             ;GET MAPREGISTER
12763  052010  010046                  2$:     MOV     R0,-(SP)                ;SAVE R0
12764  052012  013700  002100                  MOV     @#RKNEWH,R0             ;GET BA EXT
12765  052016  072027  000004                  ASH     #4,R0                   ;ADJUST
12766  052022  010037  002100                  MOV     R0,@#RKNEWH             ;SAVE
12767  052026  012600                          MOV     (SP)+,R0                ;RESTORE R0
12768  052030  012767  000105  127630          MOV     #105,RK10               ;SET FUNCTION
12769  052036  053767  002100  127622          BIS     @#RKNEWH,RK10           ;SET BA EXT BITS IN FUNCTION
12770  052044  013777  002156  130172  RK3:    MOV     @#RKHDA,@RKDA           ;LOAD DISK ADDRESS
12771  052052  013777  002030  130160          MOV     @#RKHWC,@RKWC           ;LOAD WORD COUNT
12772  052060  013777  002076  130154          MOV     @#RKNEWL,@RKBA          ;LOAD BUS ADDRESS
12773  052066  016777  127574  130142          MOV     RK10,@RKCS              ;LOAD FUNCTION AND GO
12774  052074  000002                          RTI                             ;RETURN
12775
12776                                  ;FUNCTION JUST EXECUTED WAS A READ. ANY ERRORS?
12777  052076  032737  000400  002010  RKREAD: BIT     #BIT8,@#RKHSTAT                 ;REPEAT FLAG SET?
12778  052104  001220                          BNE     RKLOOP                  ;BRANCH IF YES
12779  052106  005777  130124                  TST     @RKCS                   ;ANY ERRORS?
12780  052112  100026                          BPL     1$                      ;BRANCH IF NO
12781  052114  105737  002203                  TSTB    @#RKTRY                 ;TRYED 3 TIMES?
12782  052120  001002                          BNE     3$                      ;BRANCH IF NO
12783  052122  000167  177372                  JMP     RKERR
12784  052126  005337  002172          3$:     DEC     @#RKFUN                 ;SET FUNCTION BACK TO READ
12785  052132  012777  000001  130076          MOV     #1,@RKCS                ;CLEAR THE ERROR
12786  052140  004737  052222                  JSR     PC,@#TIMER              ;WAIT A LITTLE
12787  052144  105777  130066                  TSTB    @RKCS                   ;WAIT FOR CLR TO FINISH
12788  052150  100375                          BPL     .-4
12789  052152  105237  002203                  INCB    @#RKTRY                 ;INCREMENT TRY COUNT
12790  052156  013746  177776                  MOV     @#PSW,-(SP)
12791  052162  012746  052044                  MOV     #RK3,-(SP)
12792  052166  000002                          RTI
12793  052170  032737  000040  172516  1$:     BIT     #BIT5,@#MMR3            ;MAP ON?
12794  052176  001405                          BEQ     2$                      ;BRANCH IF NO
12795  052200  012746  000001                  MOV     #1,-(SP)                ;PUT RK ID ON STACK
12796  052204  004737  063274                  JSR     PC,@#GIVEMAP            ;RELINQUISH MAP REGSTER
12797  052210  005726                          TST     (SP)+                   ;POP THE STACK
12798  052212  112737  000200  002010  2$:     MOVB    #200,@#RKHSTAT          ;SET DON E FLAG
12799  052220  000002                          RTI                             ;RETURN
12800  052222  005067  000010          TIMER:  CLR     1$
12801  052226  105267  000004          2$:     INCB    1$
12802  052232  001375                          BNE     2$
12803  052234  000207                          RTS     PC
12804  052236  000000                  1$:     .WORD
12805
```

```
12806                                   ;;*****************************************************************
12807                                   .SBTTL  RH70/RP04 SERVICE ROUTINE
12808                                   ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
12809                                   ;;*****************************************************************
12810  052240  000005                   RP4RPT: RESET
12811  052242  005337  002176                  DEC     @#RP4FUN                ;RESTORE FUNCTION
12812  052246  022737  000001  002176          CMP     #1,@#RP4FUN             ;WHAT IS IT?
12813  052254  001501                          BEQ     RP41                    ;BRANCH IF WC
12814  052256  002560                          BLT     RP43                    ;BRANCH IF READ
12815  052260  000137  050146                  JMP     @#RP4WTRY               ;GO TO WRITE
12816  052264  005237  002176          RP4SRV: INC     @#RP4FUN                ;FIND OUT WHAT FUNCTION
12817  052270  022737  000002  002176          CMP     #2,@#RP4FUN             ;WAS JUST EXECUTED
12818  052276  001504                          BEQ     RP4WCK
12819  052300  100566                          BMI     RP4READ
12820
12821                                   ;WRITE FUNCTION WAS JUST EXECUTED.
12822  052302  032737  000400  002016          BIT     #BIT8,@#RP4HSTAT        ;REPEAT FLAG SET?
12823  052310  001050                          BNE     RP4LOOP                 ;BRANCH IF YES
12824  052312  032777  040000  127750          BIT     #BIT14,@RP4DS           ;ANY ERRORS
12825  052320  001457                          BEQ     RP41                    ;BRANCH IF NO
12826  052322  105737  002205                  TSTB    @#RP4TRY                ;TRIED 3 TIMES?
12827  052326  001426                          BEQ     RP4ERR                  ;BRANCH IF YES
12828  052330  052777  000040  127726          BIS     #BIT5,@RP4CS2           ;CLEAR ALL ERRORS
12829  052336  004737  050200                  JSR     PC,@#LDRP4              ;RELOAD THE UNIT NO
12830  052342  105237  002205                  INCB    @#RP4TRY                ;INCREMENT TRY COUNT
12831  052346  013746  177776                  MOV     @#PSW,-(SP)             ;SETUP THE STACK TO
12832  052352  012746  050146                  MOV     #RP4WTRY,-(SP)          ;TRY WRITE AGAIN
12833  052356  032737  000400  002016          BIT     #BIT8,@#RP4HSTAT        ;REPEAT FLAG SET?
12834  052364  001006                          BNE     2$                      ;BRANCH IF YES
12835  052366  012777  000007  127656          MOV     #7,@RP4CS1              ;RECALIBRATE
12836  052374  105777  127670          1$:     TSTB    @RP4DS                  ;DRIVE READY?
12837  052400  100375                          BPL     1$                      ;BRANCH IF NO
12838  052402  000002                  2$:     RTI
12839  052404  012737  100200  002016  RP4ERR: MOV     #100200,@#RP4HSTA       ;SET ERROR & DONE BIT
12840  052412  010046                          MOV     R0,-(SP)                ;SAVE R0
12841  052414  013700  001672                  MOV     @#RP411,R0              ;GET RUN TABLE INDEX
12842  052420  052760  100000  001750          BIS     #BIT15,RUNTRAK(R0)      ;SET ERROR BIT
12843  052426  012600                          MOV     (SP)+,R0                ;RESTORE R0
12844  052430  000002                          RTI                             ;RETURN
12845
12846  052432  012737  100200  002016  RP4LOOP:MOV     #100200,@#RP4HSTAT      ;SET DONE AND ERROR BITS
12847  052440  032777  040000  127622          BIT     #BIT14,@RP4DS           ;ANY ERRORS?
12848  052446  001003                          BNE     1$                              ;BRANCH IF YES
12849  052450  042737  100000  002016          BIC     #BIT15,@#RP4HSTAT       ;CLEAR ERROR BIT
12850  052456  000002                  1$:     RTI                             ;RETURN
12851                                   ;WRITE OK...NOW DO A WRITE CHECK.
12852  052460  112737  177775  002205  RP41:   MOVB    #-3,@#RP4TRY            ;INITIALIZE TRY COUNT
12853  052466  105777  127576          RP42:   TSTB    @RP4DS                  ;IS DRIVE READY?
12854  052472  001775                          BEQ     RP42                    ;BRANCH IF NO
12855  052474  004737  050200                  JSR     PC,@#LDRP4
12856  052500  112777  000151  127544          MOVB    #151,@RP4CS1            ;LOAD FUNCTION AND GO
12857  052506  000002                          RTI
12858
12859                                   ;FUNCTION JUST EXECUTED WAS A WRITE CHECK
12860  052510  032737  000400  002016  RP4WCK: BIT     #BIT8,@#RP4HSTAT                ;REPEAT FLAG SET?
12861  052516  001345                          BNE     RP4LOOP                 ;BRANCH IF YES
```

```
12862  052520  032777  040000  127542          BIT     #BIT14,@RP4DS           ;ANY ERRORS?
12863  052526  001421                          BEQ     1$                      ;BRANCH IF NO
12864  052530  105737  002205          3$:     TSTB    @#RP4TRY                ;TRIED 3 TIMES?
12865  052534  001723                          BEQ     RP4ERR                  ;BRANCH IF YES
12866  052536  005337  002176                  DEC     @#RP4FUN                        ;SET TO WRITE CHECK
12867  052542  052777  000040  127514          BIS     #BIT5,@RP4CS2           ;CLEAR ALL ERRORS
12868  052550  004737  050200                  JSR     PC,@#LDRP4              ;RELOAD THE UNIT NO
12869  052554  105237  002205                  INCB    @#RP4TRY                ;INCREMENT TRY COUNT
12870  052560  013746  177776                  MOV     @#PSW,-(SP)
12871  052564  012746  052466                  MOV     #RP42,-(SP)
12872  052570  000002                          RTI                             ;TRY AGAIN
12873  052572  032777  040000  127464  1$:     BIT     #BIT14,@RP4CS2          ;WRITE CHECK ERROR?
12874  052600  001404                          BEQ     2$                      ;BRANCH IF NO
12875  052602  005737  001672                  TST     @#RP411                 ;FIRST 2K?
12876  052606  001401                          BEQ     2$
12877  052610  000747                          BR      3$
12878
12879                                  ;WRITE CHECK WAS OK...NOW DO A READ.
12880  052612  112737  177775  002205  2$:     MOVB    #-3,@#RP4TRY            ;INITIALIZE TRY COUNT
12881  052620  105777  127444          RP43:   TSTB    @RP4DS                  ;IS DRIVE READY?
12882  052624  001775                          BEQ     RP43                    ;BRANCH IF NO
12883  052626  004737  050200                  JSR     PC,@#LDRP4              ;LOAD REGISTERS
12884  052632  013777  002114  127420          MOV     @#RP4NWH,@RP4BAE        ;LOAD EXTENDED ADR BITS
12885  052640  013777  002112  127410          MOV     @#RP4NWL,@RP4BA         ;LOAD BUS ADR
12886  052646  112777  000171  127376          MOVB    #171,@RP4CS1            ;LOAD FUNCTION AND GO
12887  052654  000002                          RTI                             ;RETURN
12888
12889                                  ;FUNCTION JUST EXECUTED WAS A READ.
12890  052656  032737  000400  002016  RP4READ:BIT     #BIT8,@#RP4HSTAT        ;REPEAT FLAG SET?
12891  052664  001262                          BNE     RP4LOOP                 ;BRANCH IF YES
12892  052666  032777  040000  127374          BIT     #BIT14,@RP4DS           ;ANY ERRORS?
12893  052674  001421                          BEQ     1$                      ;BRANCH IF NO
12894  052676  105737  002205                  TSTB    @#RP4TRY                ;TRIED 3 TIMES?
12895  052702  001640                          BEQ     RP4ERR                  ;BRANCH IF YES
12896  052704  005337  002176                  DEC     @#RP4FUN                ;SET FUNCTION TO A READ
12897  052710  052777  000040  127346          BIS     #BIT5,@RP4CS2           ;CLEAR ALL ERRORS
12898  052716  004737  050200                  JSR     PC,@#LDRP4              ;RELOAD THE UNIT NO
12899  052722  105237  002205                  INCB    @#RP4TRY                ;INCREMENT TRY COUNT
12900  052726  013746  177776                  MOV     @#PSW,-(SP)
12901  052732  012746  052620                  MOV     #RP43,-(SP)
12902  052736  000002                          RTI                             ;TRY AGAIN
12903  052740  112737  000200  002016  1$:     MOVB    #200,@#RP4HSTA          ;SET DONE FLAG
12904  052746  000002                          RTI                             ;RETURN
12905                                  ;;*******************************************************************
12906                                  .SBTTL  RH70/RS04 SERVICE ROUTINE
12907                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
12908                                  ;;*******************************************************************
12909  052750  000005                  RSRPT:  RESET
12910  052752  005337  002200                  DEC     @#RSFUN                 ;RESTORE FUNCTION
12911  052756  022737  000001  002200          CMP     #1,@#RSFUN              ;WHAT IS IT?
12912  052764  001467                          BEQ     RS41                    ;BRANCH IF WC
12913  052766  002546                          BLT     RS43                    ;BRANCH IF WRITE
12914  052770  000137  050444                  JMP     @#RSWTRY
12915  052774  005237  002200          RSSRV:  INC     @#RSFUN                 ;FIND OUT WHAT FUNCTION
12916  053000  022737  000002  002200          CMP     #2,@#RSFUN              ;WAS JUST EXECUTED
12917  053006  001472                          BEQ     RSWCK
```

```
12918  053010  100554                          BMI     RSREAD
12919
12920                                  ;WRITE FUNCTION WAS JUST EXECUTED
12921  053012  032737  000400  002020          BIT     #BIT8,@#RSHSTAT ;REPEAT FLAG SET?
12922  053020  001036                          BNE     RSLOOP                  ;BRANCH IF YES
12923  053022  032777  040000  127300          BIT     #BIT14,@RSDS            ;ANY ERRORS?
12924  053030  001445                          BEQ     RS41                    ;BRANCH IF NO
12925  053032  105737  002206                  TSTB    @#RSTRY                 ;TRIED 3 TIMES?
12926  053036  001414                          BEQ     RSERR                   ;BRANCH IF YES
12927  053040  052777  000040  127256          BIS     #BIT5,@RSCS2            ;CLEAR ALL ERRORS
12928  053046  004737  050504                  JSR     PC,@#LDRS               ;LOAD UNIT #
12929  053052  105237  002206                  INCB    @#RSTRY                 ;INCREMENT TRY COUNT
12930  053056  013746  177776                  MOV     @#PSW,-(SP)             ;SETUP THE STACK TO
12931  053062  012746  050444                  MOV     #RSWTRY,-(SP)           ;TRY THE WRITE AGAIN
12932  053066  000002                          RTI
12933  053070  012737  100200  002020  RSERR:  MOV     #100200,@#RSHSTAT       ;SET ERROR AND DONE BIT
12934  053076  010046                          MOV     R0,-(SP)                ;SAVE R0
12935  053100  013700  001674                  MOV     @#RS11,R0               ;GET RUN TBL INDEX
12936  053104  052760  100000  001750          BIS     #BIT15,RUNTRAK(R0)      ;SET ERROR BIT
12937  053112  012600                          MOV     (SP)+,R0                ;RESTORE R0
12938  053114  000002                          RTI
12939
12940  053116  012737  100200  002020  RSLOOP: MOV     #100200,@#RSHSTAT       ;SET DONE AND ERROR BITS
12941  053124  032777  040000  127176          BIT     #BIT14,@RSDS            ;ANY ERRORS?
12942  053132  001003                          BNE     1$                      ;BRANCH IF YES
12943  053134  042737  100000  002020          BIC     #BIT15,@#RSHSTAT        ;CLEAR ERROR BIT
12944  053142  000002                  1$:     RTI                             ;RETURN
12945                                  ;WRITE OK...NOW DO A WRITE CHECK
12946  053144  112737  177775  002206  RS41:   MOVB    #-3,@#RSTRY             ;INIT TRY COUNT
12947  053152  105777  127152          RS42:   TSTB    @RSDS                   ;IS DRIVE READY?
12948  053156  001775                          BEQ     RS42                    ;BRANCH IF NO
12949  053160  004737  050504                  JSR     PC,@#LDRS               ;LOAD RS REGISTERS
12950  053164  112777  000151  127120          MOVB    #151,@RSCS1             ;LOAD FUNCTION AND GO
12951  053172  000002                          RTI                             ;RETURN
12952
12953                                  ;FUNCTION JUST EXECUTED WAS A WRITE CHECK
12954  053174  032737  000400  002020  RSWCK:  BIT     #BIT8,@#RSHSTAT         ;REPEAT FLAG SET?
12955  053202  001345                          BNE     RSLOOP                  ;BRANCH IF YES
12956  053204  032777  040000  127116          BIT     #BIT14,@RSDS            ;ANY ERRORS?
12957  053212  001421                          BEQ     1$                      ;BRANCH IF NO
12958  053214  105737  002206          3$:     TSTB    @#RSTRY                 ;TRIED 3 TIMES?
12959  053220  001723                          BEQ     RSERR                   ;BRANCH IF YES
12960  053222  005337  002200                  DEC     @#RSFUN                 ;SET FUNCTION BACK TO WC
12961  053226  052777  000040  127070          BIS     #BIT5,@RSCS2            ;CLEAR THE ERROR
12962  053234  004737  050504                  JSR     PC,     @#LDRS
12963  053240  105237  002206                  INCB    @#RSTRY                 ;INCREMENT THE TRY COUNT
12964  053244  013746  177776                  MOV     @#PSW,-(SP)
12965  053250  012746  053152                  MOV     #RS42,-(SP)
12966  053254  000002                          RTI                             ;TRY AGAIN
12967
12968  053256  032777  040000  127040  1$:     BIT     #BIT14,@RSCS2           ;WRITE CHECK ERROR?
12969  053264  001404                          BEQ     2$                      ;BRANCH IF NO
12970  053266  005737  001674                  TST     @#RS11                  ;FIRST 2K?
12971  053272  001401                          BEQ     2$                      ;BRANCH IF YES
12972  053274  000747                          BR      3$
12973
```

```
12974                                   ;WRITE CHECK WAS OK...NOW DO A READ.
12975  053276  112737  177775  002206   2$:     MOVB    #-3,@#RSTRY             ;INIT TRY COUNT
12976  053304  105777  127020           RS43:   TSTB    @RSDS                   ;IS DRIVE READY?
12977  053310  001775                           BEQ     RS43                    ;BRANCH IF NO
12978  053312  004737  050504                   JSR     PC,@#LDRS               ;LOAD RS REGISTERS
12979  053316  013777  002120  126774           MOV     @#RSNEWH,@RSBAE         ;LOAD BAE
12980  053324  013777  002116  126764           MOV     @#RSNEWL,@RSBA          ;LOAD BUS ADR
12981  053332  112777  000171  126752           MOVB    #171,@RSCS1             ;LOAD FUNCTION AND GO
12982  053340  000002                           RTI                             ;RETURN
12983
12984                                   ;FUNCTION JUST EXECUTED WAS A READ.
12985  053342  032737  000400  002020   RSREAD: BIT     #BIT8,@#RSHSTAT         ;REPEAT FLAG SET?
12986  053350  001262                           BNE     RSLOOP                  ;BRANCH IF YES
12987  053352  032777  040000  126750           BIT     #BIT14,@RSDS            ;ANY ERRORS?
12988  053360  001421                           BEQ     1$                      ;BRANCH IF NO
12989  053362  105737  002206                   TSTB    @#RSTRY                 ;TRIED 3 TIMES?
12990  053366  001640                           BEQ     RSERR                   ;BRANCH IF YES
12991  053370  005337  002200                   DEC     @#RSFUN                 ;RESTORE FUN TO READ
12992  053374  052777  000040  126722           BIS     #BIT5,@RSCS2            ;CLEAR ALL ERRORS
12993  053402  004737  050504                   JSR     PC,@#LDRS               ;LOAD UNIT #
12994  053406  105237  002206                   INCB    @#RSTRY                 ;INCREMENT TRY COUNT
12995  053412  013746  177776                   MOV     @#PSW,-(SP)
12996  053416  012746  053304                   MOV     #RS43,-(SP)
12997  053422  000002                           RTI                             ;TRY AGAIN
12998  053424  112737  000200  002020   1$:     MOVB    #200,@#RSHSTAT          ;SET DONE FLAG
12999  053432  000002                           RTI                             ;RETURN
13000                                   ;;*****************************************************************
13001                                   .SBTTL  UNIBUS EXERCISER SERVICE ROUTINE
13002                                   ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
13003                                   ;;*****************************************************************
13004  053434  104420           UBESRV: SAVREG
13005  053436  004737  064104           JSR     PC,@#LDKT               ;GO TO LOW CORE
13006  053442  012704  002346           MOV     #UBETBL+6,R4            ;GET ADDRESS OF UBECR1
13007  053446  005774  000000           TST     @(R4)                   ;WAS THERE AN ERROR?
13008  053452  100437                   BMI     UBE2                    ;BRANCH IF YES
13009  053454  012746  000003           MOV     #3,-(SP)                ;PUT DEVICE ID IN STACK
13010  053460  004737  063274           JSR     PC,@#GIVEMAP            ;GIVE UP MAP REG
13011  053464  012767  002414  126300   MOV     #$ERRTB,UBESAV          ;INITIALIZE UBE
13012  053472  005067  126276           CLR     UBESAV+2
13013  053476  012767  002414  126272   MOV     #$ERRTB,UBEADR
13014  053504  005067  126270           CLR     UBEADR+2
13015  053510  012746  001776           MOV     #UBEADR,-(SP)
13016  053514  004737  063022           JSR     PC,@#GETMAP
13017  053520  013754  002000           MOV     @#UBEADR+2,@-(R4)       ;LOAD UBECR2
13018  053524  013754  001776           MOV     @#UBEADR,@-(R4)         ;LOAD UBEBA
13019  053530  012754  172400           MOV     #172400,@-(R4)          ;LOAD UBECC
13020  053534  004737  064202           JSR     PC,@#RESKT              ;GO BACK TO ORIGINAL CORE
13021  053540  104422                   RESREG
13022  053542  012777  064545  126576   MOV     #64545,@UBETBL+6        ;RESTART UBE
13023  053550  000002                   RTI                             ;RETURN
13024
13025                                   ;UBE ERROR-IS IT LAST MEMORY?
13026  053552  005037  001372   UBE2:   CLR     @#$TMP10
13027  053556  162704  000004           SUB     #4,R4                   ;ADJUST R4
13028  053562  017403  000002           MOV     @2(R4),R3               ;GET BECR2
13029  053566  042703  070037           BIC     #70037,R3               ;CLEAR ALL BUT ERROR FLAGS
```

```
13030  053572  022703  000400              CMP     #400,R3                 ;WAS ERROR A TIMEOUT?
13031  053576  001052                      BNE     UBEERR                  ;BRANCH IF NO
13032  053600  017437  000000  002002      MOV     @(R4),@#ERRBA           ;SAVE BUS ADR OF ERROR
13033  053606  017437  000002  002004      MOV     @2(R4),@#ERRBA+2
13034  053614  042737  177774  002004      BIC     #177774,@#ERRBA+2
13035  053622  004737  062450              JSR     PC,@#PHYMAP             ;GET PHYSICAL ADDRESS THAT TIMED OUT
13036  053626  162737  000004  001574      SUB     #4,@#PA1500             ;ADJUST PHYSICAL ADR THAT FAILED
13037  053634  005637  001576              SBC     @#PA2116                ;UBE STOPS AT ADR+4
13038  053640  023737  001576  001656      CMP     @#PA2116,@#MXMMHI       ;AT MAX MEM HIGH?
13039  053646  101006                      BHI     1$                      ;BRANCH IF HIGHER
13040  053650  103423                      BLO     MHOLE                   ;BRANCH IF LOWER
13041  053652  023737  001574  001660      CMP     @#PA1500,@#MXMMLO       ;AT MAX MEM LO?
13042  053660  101001                      BHI     1$                      ;BRANCH IF HIGHER
13043  053662  103416                      BLO     MHOLE                   ;BRANCH IF LOWER
13044  053664  012746  000003      1$:     MOV     #3,-(SP)                ;PUT DEVICE ID ON STACK
13045  053670  004737  063274              JSR     PC,@#GIVEMAP
13046  053674  005726                      TST     (SP)+
13047  053676  004737  062340              JSR     PC,@#UBEINIT
13048  053702  004737  064202              JSR     PC,@#RESKT
13049  053706  104422                      RESREG
13050  053710  012777  064545  126430      MOV     #64545,@UBETBL+6
13051  053716  000002                      RTI
13052
13053  053720  010637  001372      MHOLE:  MOV     SP,@#$TMP10
13054  053724  013737  001262  001374  UBEERR: MOV  @#$LPERR,@#$TMP11       ;SAVE LOOP ARROR ADR
13055  053732  012737  053774  001262      MOV     #UBE3,@#$LPERR          ;SET LOOP ADR
13056  053740  012703  000022              MOV     #22,R3
13057  053744  005737  001372              TST     @#$TMP10
13058  053750  001002                      BNE     1$
13059  053752  104007                      ERROR   7
13060  053754  000407                      BR      UBE3
13061  053756  013737  001574  001276  1$:     MOV     @#PA1500,@#$GDDAT
13062  053764  013737  001576  001300          MOV     @#PA2116,@#$BDDAT
13063  053772  104012                      ERROR   12
13064
13065                                  ;RESTART UBE IN SAME MEMORY
13066  053774  013737  001374  001262  UBE3:   MOV     @#$TMP11,@#$LPERR       ;RESTORE ERROR LOOP ADR
13067  054002  010446                      MOV     R4,-(SP)                ;SAVE R4
13068  054004  012704  002340              MOV     #UBETBL,R4              ;GET ADDRESS OF UBE TABLE
13069  054010  012734  172400              MOV     #172400,@(R4)+          ;SET UBECC
13070  054014  013734  001776              MOV     @#UBEADR,@(R4)+         ;SET UBEBA <15:00>
13071  054020  005074  000004              CLR     @4(R4)                  ;CLEAR ALL ERRORS
13072  054024  013734  002000              MOV     @#UBEADR+2,@(R4)+       ;SET EXT ADR BITS
13073  054030  012774  064545  000000      MOV     #64545,@(R4)            ;START UBE
13074  054036  012604                      MOV     (SP)+,R4                ;RESTORE R4
13075  054040  004737  064202              JSR     PC,@#RESKT
13076  054044  104422                      RESREG
13077  054046  000002                      RTI                             ;RETURN
13078
13079                                  ;;*******************************************************************************
13080                                  .SBTTL  MASS BUS TESTER SERVICE ROUTINE
13081                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
13082                                  ;;*******************************************************************************
13083  054050  104420                  MBTSRV: SAVREG
13084  054052  004737  064104                  JSR     PC,@#LDKT               ;GO TO LOW CORE
13085  054056  005037  001372                  CLR     @#$TMP10
```

```
13086  054062  012704  002356              MOV     #MBTTBL,R4              ;GET ADDRESS OF ADDRESS OF CS1 REG
13087  054066  032734  040000              BIT     #BIT14,@(R4)+           ;ANY ERRORS?
13088  054072  001007                      BNE     1$                      ;BRANCH IF YES
13089  054074  004737  064202      2$:     JSR     PC,@#RESKT              ;GO BACK TO ORIGINAL CORE
13090  054100  104422                      RESREG
13091  054102  112777  000161  126246      MOVB    #161,@MBTTBL
13092  054110  000002                      RTI                             ;RESTART MBT AND RETURN
13093  054112  062704  000010      1$:     ADD     #10,R4                  ;ADJUST R4
13094  054116  032774  004000  000000      BIT     #BIT11,@(R4)            ;NON-EXISTANT MEMORY ERROR?
13095  054124  001436                      BEQ     MBTERR                  ;BRANCH IF NO
13096  054126  162704  000006              SUB     #6,R4                   ;ADJUST R4
13097  054132  013437  001574              MOV     @(R4)+,@#PA1500         ;GET BUS ADR
13098  054136  013437  001576              MOV     @(R4)+,@#PA2116         ;GET BUS ADR EXT
13099  054142  162737  000004  001574      SUB     #4,@#PA1500             ;ADJUST BUS ADR
13100  054150  005637  001576              SBC     @#PA2116
13101  054154  023737  001574  001660      CMP     @#PA1500,@#MXMMLO       ;IS IT LAST MEMORY?
13102  054162  001015                      BNE     MEMHOLE                 ;BRANCH IF NO
13103  054164  023737  001576  001656      CMP     @#PA2116,@#MXMMHI       ;CHECK EXT ADR BITS
13104  054172  001011                      BNE     MEMHOLE
13105  054174  005724                      TST     (R4)+                   ;INCREMENT R4
13106  054176  052774  000047  000000      BIS     #47,@(R4)               ;CLEAR THE ERROR
13107  054204  012734  000007              MOV     #7,@(R4)+               ;SELECT UNIT 7
13108  054210  005074  177766              CLR     @-12(R4)                ;CLEAR WORD COUNT
13109  054214  000727                      BR      2$                      ;CONTINUE
13110
13111  054216  010637  001372      MEMHOLE:MOV     SP,@#$TMP10
13112  054222  013737  001262  001374  MBTERR: MOV     @#$LPERR,@#$TMP11       ;SAVE LOOP ADDRESS
13113  054230  012737  054272  001262      MOV     #1$,@#$LPERR            ;SET NEW LOOP ADR
13114  054236  012703  000020              MOV     #20,R3                  ;PUT DEVICE ID IN R3
13115  054242  005737  001372              TST     @#$TMP10
13116  054246  001002                      BNE     2$
13117  054250  104007                      ERROR   7
13118  054252  000407                      BR      1$
13119  054254  013737  001574  001276  2$: MOV     @#PA1500,@#$GDDAT
13120  054262  013737  001576  001300      MOV     @#PA2116,@#$BDDAT
13121  054270  104013                      ERROR   13
13122  054272  013737  001374  001262  1$: MOV     @#$TMP11,@#$LPERR       ;RESTORE LOOP ADR
13123  054300  012704  002366              MOV     #MBTTBL+10,R4           ;GET ADR OF MBTTBL+10
13124  054304  015400                      MOV     @-(R4),R0               ;GET BUS ADR EXTENDED
13125  054306  015401                      MOV     @-(R4),R1               ;GET BUS ADR
13126  054310  015402                      MOV     @-(R4),R2               ;GET WORD COUNT
13127  054312  006302                      ASL     R2                      ;ADJUST WORD COUNT
13128  054314  160201                      SUB     R2,R1                   ;FORM START ADR OF THIS XFER
13129  054316  005600                      SBC     R0
13130  054320  052774  000047  000010      BIS     #47,@10(R4)             ;CLEAR THE WORLD
13131  054326  012774  000007  000010      MOV     #7,@10(R4)              ;SELECT UNIT 7
13132  054334  005724                      TST     (R4)+                   ;ADJUST R4
13133  054336  010134                      MOV     R1,@(R4)+               ;RESTORE BUS ADR
13134  054340  010074  000000              MOV     R0,@(R4)
13135  054344  004737  064202              JSR     PC,@#RESKT              ;GO BACK TO ORIGINAL CORE
13136  054350  104422                      RESREG
13137  054352  112777  000161  125776      MOVB    #161,@MBTTBL            ;START MBT AGAIN
13138  054360  000002                      RTI                             ;RETURN
13139                          ;;*********************************************************************
13140                          .SBTTL  LINE CLOCK SERVICE ROUTINE
13141                          ;*      THIS ROUTINE FIRST REMAPS PROGRAM EXECUTION TO LOW
```

```
13142                                   ;*      MEMORY. IT THEN INCREMENTS AND KEEPS TRACK OF THE
13143                                   ;*      SECOND AND MINUTE COUNTS KEPT IN LOCATIONS 'LTICKS'
13144                                   ;*      AND 'MTICKS' RESPECTIVELY.
13145                                   ;;*****************************************************************
13146  054362  104420                  LKSRV:  SAVREG
13147  054364  004737  064104                  JSR     PC,@#LDKT               ;GO TO LOW CORE
13148  054370  105237  001700                  INCB    @#LTICKS                ;INCREMENT TICK COUNT
13149  054374  122737  000074  001700        CMPB    #60.,@#LTICKS           ;ONE SECOND YET?
13150  054402  001016                  BNE     1$                      ;BRANCH IF NO
13151  054404  105237  001701                  INCB    @#LTICKS+1              ;INCREMENT SECOND COUNT
13152  054410  005267  124610                  INC     $DEVCT                  ;KEEP APT GOING.....
13153  054414  105037  001700                  CLRB    @#LTICKS                ;CLEAR SECOND COUNT
13154  054420  122737  000074  001701        CMPB    #60.,@#LTICKS+1         ;ONE MINUTE YET?
13155  054426  001004                  BNE     1$                      ;BRANCH IF NO
13156  054430  105037  001701                  CLRB    @#LTICKS+1
13157  054434  005237  001676                  INC     @#MTICKS                ;INCREMENT MINUTE COUNT
13158  054440  004737  064202          1$:     JSR     PC,@#RESKT              ;RESTORE THE KT
13159  054444  104422                  RESREG
13160  054446  012737  000100  177546        MOV     #BIT6,@#LKS             ;CLEAR READY BIT IN CLOCK
13161  054454  000002                  RTI                             ;RETURN
13162
13163                                   ;;*****************************************************************
  (1)
  (1)                                   .SBTTL  SCOPE HANDLER ROUTINE
  (1)
  (1)                                   ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
  (1)                                   ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
  (1)                                   ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
  (1)                                   ;*SW14=1        LOOP ON TEST
  (1)                                   ;*SW11=1        INHIBIT ITERATIONS
  (1)                                   ;*SW09=1        LOOP ON ERROR
  (1)                                   ;*CALL
  (1)                                   ;*      SCOPE           ;;SCOPE=IOT
  (1)
  (1)  054456                           $SCOPE:
  (1)  054456  032777  040000  124560          BIT     #SW14,@$SWRP    ;;LOOP ON PRESENT TEST?
  (1)  054464  001077                   BNE     $OVER           ;;YES IF SW14=1
  (1)                                   ;#####START OF CODE FOR THE XOR TESTER#####
  (1)  054466  000416                   $XTSTR: BR      6$              ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
  (1)                                                                   ;;THIS INSTRUCTION TO A 'NOP' (NOP=240)
  (1)  054470  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
  (1)  054474  012737  054514  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
  (1)  054502  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
  (1)  054506  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
  (1)  054512  000453                   BR      $SVLAD          ;;GO TO THE NEXT TEST
  (1)  054514  022626                   5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
  (1)  054516  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
  (1)  054522  000413                   BR      7$              ;;LOOP ON THE PRESENT TEST
  (1)  054524                           6$:;#####END OF CODE FOR THE XOR TESTER#####
  (1)  054524  105767  124524          2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
  (1)  054530  001421                   BEQ     3$              ;;BR IF NO
  (1)  054532  126767  124531  124514        CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
  (1)  054540  101015                   BHI     3$              ;;BR IF NO
  (1)  054542  032777  001000  124474          BIT     #BIT09,@$SWRP   ;;LOOP ON ERROR?
  (1)  054550  001404                   BEQ     4$              ;;BR IF NO
  (1)  054552  016767  124504  124500  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
```

```
(1)  054560  000441                        BR      $OVER
(1)  054562  105067  124466          4$:   CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
(1)  054566  005067  124604                CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
(1)  054572  000415                        BR      1$              ;;ESCAPE TO THE NEXT TEST
(1)  054574  032777  004000  124442  3$:   BIT     #BIT11,@$SWRP   ;;INHIBIT ITERATIONS?
(1)  054602  001011                        BNE     1$              ;;BR IF YES
(1)  054604  105767  124440                TSTB    $PASS           ;;IF FIRST PASS OF PROGRAM
(1)  054610  001406                        BEQ     1$              ;;      INHIBIT ITERATIONS
(1)  054612  005267  124440                INC     $ICNT           ;;INCREMENT ITERATION COUNT
(1)  054616  026767  124554  124432        CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
(1)  054624  002017                        BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
(1)  054626  012767  000001  124422  1$:   MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
(1)  054634  016767  000046  124534        MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
(1)  054642  011667  124412          $SVLAD: MOV   (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
(1)  054646  011667  124410                MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
(1)  054652  005067  124522                CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
(1)  054656  112767  000001  124403        MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
(1)  054664  105767  124364          $OVER:  TSTB  $ERFLG          ;;ANY ERRORS?
(1)  054670  001403                        BEQ     1$              ;;BRANCH IF NO
(1)  054672  116737  124356  001253        MOVB    $ERFLG,@#$TSTNM+1
(1)  054700  016716  124354          1$:   MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
(1)  054704  000002                        RTI                     ;;FIXES PS
(1)  054706  000010                $MXCNT: 10                      ;;MAX. NUMBER OF ITERATIONS
13164                              ;;**************************************************************************
(1)
(1)                                .SBTTL  ERROR HANDLER ROUTINE
(1)
(1)                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
(1)                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
(1)                                ;*AND GO TO $ERRTYP ON ERROR
(1)                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
(1)                                ;*SW15=1        HALT ON ERROR
(1)                                ;*              HALT CAN OCCUR BEFORE AND AFTER THE ERROR TYPEOUT
(1)                                ;*SW13=1        INHIBIT ERROR TYPEOUTS
(1)                                ;*SW10=1        BELL ON ERROR
(1)                                ;*SW09=1        LOOP ON ERROR
(1)                                ;*CALL
(1)                                ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
(1)
(1)  054710                        $ERROR:
(2)  054710  116737  124340  001253        MOVB    $ERFLG,@#$TSTNM+1
(1)  054716  105267  124332          7$:   INCB    $ERFLG          ;;SET THE ERROR FLAG
(1)  054722  001775                        BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
(1)  054724  016737  124322  177570        MOV     $TSTNM,@#DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
(1)  054732  005767  124276                TST     $ENV            ;;APT MODE ?
(1)  054736  001414                        BEQ     10$
(1)  054740  011667  124252                MOV     (SP),$FATAL     ;;LOAD APT MAILBOX
(1)  054744  162767  000002  124244        SUB     #2,$FATAL
(1)  054752  016767  124274  124240        MOV     $TSTNM,$TESTN
(1)  054760  012767  000001  124226        MOV     #1,$MSGTY       ;;FATAL ERROR
(1)  054766  000000                        HALT
(1)  054770                        10$:
(1)  054770  005777  124250                TST     @$SWRP          ;;HALT ON ERROR = 1?
(1)  054774  100001                        BPL     8$              ;;BRANCH IF NO
(1)  054776  000000                        HALT                    ;;YES--HALT
(1)  055000  032777  002000  124236  8$:   BIT     #BIT10,@$SWRP   ;;BELL ON ERROR?
```

```
(1)  055006  001402                        BEQ     1$              ;;NO - SKIP
(1)  055010  104400  001402                TYPE    ,$BELL          ;;RING BELL
(1)  055014  005267  124244          1$:   INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
(1)  055020  011667  124244                MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
(1)  055024  162767  000002  124236        SUB     #2,$ERRPC
(1)  055032  117767  124232  124226        MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
(1)  055040  032777  020000  124176        BIT     #BIT13,@$SWRP   ;;SKIP TYPEOUT IF SET
(1)  055046  001004                        BNE     2$              ;;SKIP TYPEOUTS
(1)  055050  004767  000056                JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
(1)  055054  104400  001407                TYPE    ,$CRLF
(1)  055060  005777  124160          2$:   TST     @$SWRP          ;;HALT ON ERROR
(1)  055064  100001                        BPL     9$              ;;SKIP IF CONTINUE
(1)  055066  000000                        HALT                    ;;HALT ON ERROR!
(1)  055070  022767  046652  122744  9$:   CMP     #$ENDAD,42      ;;ACT-11?
(1)  055076  001001                        BNE     3$              ;;BRANCH IF NO
(1)  055100  000000                        HALT                    ;;YES
(1)  055102  032777  001000  124134  3$:   BIT     #BIT09,@$SWRP   ;;LOOP ON ERROR SWITCH SET?
(1)  055110  001402                        BEQ     4$              ;;BR IF NO
(1)  055112  016716  124144                MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
(1)  055116  005767  124256          4$:   TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
(1)  055122  001402                        BEQ     5$              ;;BR IF NONE
(1)  055124  016716  124250                MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
(1)  055130                          5$:
(1)  055130  000002                        RTI                     ;;RETURN
13165                           ;;**********************************************************************
13166                           .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
13167
13168                           ;*THIS ROUTINE FIRST TYPES A STANDARD MESSAGE CONSISTING OF THE
13169                           ;*VIRTUAL PC, THE PHYSICAL PC, THE PSW AT THE TIME OF THE ERROR CALL,
13170                           ;*AND THE  SUB-PASS COUNT. THE SUB-PASS COUNT CONSISTS OF THE SUB PASS COUNT IN THE
13171                           ;*HIGH BYTE AND THE PASS COUNT IN THE LOW BYTE.
13172                           ;*
13173                           ;*IT THEN USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
13174                           ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE"
13175                           ;*THE ERROR MESSAGE POINTER AND TYPES THE ERROR MESSAGE. THE DATA
13176                           ;*HEADER POINTER IS THEN OBTAINED AND A DATA HEADER IS TYPED.
13177                           ;*THE DATA POINTER AND DATA FORMAT ARE THEN OBTAINED. THERE ARE
13178                           ;*FOUR TYPES OF DATA FORMAT, AS FOLLOWS:
13179                           ;*
13180                           ;*      0       TYPE THE CONTENTS OF THE DATA TABLE WORD IN
13181                           ;*              6 DIGIT OCTAL FORMAT
13182                           ;*      1       CONVERT THE CONTENTS OF THE DATA TABLE WORD TO
13183                           ;*              22 BITS AND TYPE AN 8 DIGIT OCTAL NUMBER
13184                           ;*      2       TYPE THE CONTENTS OF THE DATA TABLE WORD AND
13185                           ;*              THE WORD+2 IN 8 DIGIT OCTAL FORMAT
13186                           ;*      3       USE THE CONTENTS OF THE DATA TABLE WORD AS A
13187                           ;*              DEVICE ID AND TYPE THE DEVICES NAME
13188                           ;*      4       CONVERT THE TWO WORDS POINTED TO BY THE DATA
13189                           ;*              TABLE TO FLOATING POINT FORMAT AND TYPE.
13190                           ;*      5       CONVERT THE FOUR WORDS POINTED TO BY THE DATA
13191                           ;*              TABLE TO FLOATING  DOUBLE FORMAT AND TYPE
13192                           ;;**********************************************************************
13193
13194  055132  104420          $ERRTYP:SAVREG
13195  055134  104400  001407          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
13196  055140  004737  056632          JSR     PC,@#TYPTIME    ;GO TYPE THE TIME
```

```
13197  055144  104400  064642                  TYPE    ,MSG3
13198  055150  104400  001407                  TYPE    ,$CRLF
13199  055154  016746  124110                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
  (1)                                                                   ;;TYPE THE VIRTUAL PC
  (1)  055160  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
13200  055162  104400  056404                  TYPE    ,8$
13201  055166  013700  001572                  MOV     @#VADR,R0               ;SAVE VADR
13202  055172  013737  001270  001572          MOV     @#$ERRPC,@#VADR         ;SAVE THE VIR PC FOR CONVERSION
13203  055200  122737  000014  001266          CMPB    #14,@#$ITEMB
13204  055206  003403                          BLE     51$
13205  055210  105737  001266                  TSTB    @#$ITEMB                ;ERROR ZERO?
13206  055214  001005                          BNE     42$                     ;BRANCH IF NO
13207  055216  004737  062556          51$:    JSR     PC,@#CNVADR             ;CONVERT TO 22 BITS
13208  055222  010037  001572                  MOV     R0,@#VADR
13209  055226  000407                          BR      41$
13210  055230  013737  001572  001574  42$:    MOV     @#VADR,@#PA1500
13211  055236  005037  001576                  CLR     @#PA2116
13212  055242  010037  001572                  MOV     R0,@#VADR
13213  055246  012746  001574          41$:    MOV     #PA1500,-(SP)           ;PUT ADDRESS OFPC ON STACK
13214  055252  004737  057670                  JSR     PC,@#$DB2O              ;CONVERT TO ASCII
13215  055256  062716  000003                  ADD     #3,(SP)                 ;GET RID OF 3 MS DIGITS
13216  055262  012667  000002                  MOV     (SP)+,30$               ;SAVE POINTER TO ASCII
13217  055266  104400                          TYPE                            ;TYPE IT
13218  055270  000000                  30$:    .WORD
13219  055272  104400  056404                  TYPE    ,8$
13220  055276  016646  000030                  MOV     30(SP),-(SP)            ;GET PSW AT TIME OF ERROR
13221  055302  104402                          TYPOC                           ;TYPE IT
13222
13223  055304  104400  056404                  TYPE    ,8$
13224  055310  016746  124374                  MOV     $MAINT,-(SP)    ;;SAVE $MAINT FOR TYPEOUT
  (1)                                                                   ;;TYPE THE MAINTENANCE REG
  (1)  055314  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
13225  055316  104400  056404                  TYPE    ,8$
13226  055322  116746  123724                  MOVB    $TSTNM,-(SP)
13227  055326  105066  000001                  CLRB    1(SP)
13228  055332  104402                          TYPOC                   ;TYPE THE TEST NUMBER
13229  055334  104400  056404                  TYPE    ,8$
13230  055340  013746  001630                  MOV     @#SUBPASS,-(SP)
13231  055344  162716  000060                  SUB     #60,(SP)
13232  055350  104402                          TYPOC
13233  055352  104400  056404                  TYPE    ,8$
13234  055356  016746  123666                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
  (1)                                                                   ;;TYPE THE PASS COUNT
  (1)  055362  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
13235  055364  104400  001407                  TYPE    ,$CRLF
13236  055370  005000                          CLR     R0
13237  055372  153700  001266                  BISB    @#$ITEMB,R0             ;PICK UP THE INDEX
13238  055376  001431                          BEQ     6$                      ;EXIT IF ZERO
13239  055400  022700  000007          1$:     CMP     #7,R0           ;IS THIS ERROR 7?
13240  055404  001551                          BEQ     15$             ;BRANCH IF YES
13241  055406  005300                          DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
13242  055410  006300                          ASL     R0              ;;      WORK FOR THE ERROR TABLE
13243  055412  006300                          ASL     R0
13244  055414  006300                          ASL     R0
13245  055416  062700  002414                  ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
13246  055422  012067  000004                  MOV     (R0)+,2$        ;;PICKUP 'ERROR MESSAGE' POINTER
```

```
13247  055426  001404                        BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
13248  055430  104400                        TYPE                    ;;TYPE THE 'ERROR MESSAGE'
13249  055432  000000                2$:     .WORD   0               ;;'ERROR MESSAGE' POINTER GOES HERE
13250  055434  104400  001407                TYPE    ,$CRLF          ;;'CARRIAGE RETURN' & 'LINE FEED'
13251  055440  012067  000004        3$:     MOV     (R0)+,4$        ;;PICKUP 'DATA HEADER' POINTER
13252  055444  001404                        BEQ     5$              ;;SKIP TYPEOUT IF 0
13253  055446  104400                        TYPE                    ;;TYPE THE 'DATA HEADER'
13254  055450  000000                4$:     .WORD   0               ;;'DATA HEADER' POINTER GOES HERE
13255  055452  104400  001407                TYPE    ,$CRLF          ;;'CARRIAGE RETURN' & 'LINE FEED'
13256  055456  012001                5$:     MOV     (R0)+,R1                ;;PICKUP 'DATA TABLE' POINTER
13257  055460  001004                        BNE     7$              ;;GO TYPE THE DATA
13258  055462  104422                6$:     RESREG
13259  055464  104400  001407                TYPE    ,$CRLF          ;;'CARRIAGE RETURN' & 'LINE FEED'
13260  055470  000207                        RTS     PC              ;;RETURN
13261  055472  011002                7$:     MOV     (R0),R2         ;GET 'DATA FORMAT' POINTER
13262  055474  122712  000001        10$:    CMPB    #1,(R2)         ;DATA FORMAT 1?
13263  055500  001424                        BEQ     9$              ;BRANCH IF YES
13264  055502  122712  000002                CMPB    #2,(R2)         ;DATA FORMAT 2?
13265  055506  001441                        BEQ     11$             ;BRANCH IF YES
13266  055510  122712  000003                CMPB    #3,(R2)         ;DATA FORMAT 3?
13267  055514  001445                        BEQ     24$             ;BRANCH IF YES
13268  055516  122712  000004                CMPB    #4,(R2)         ;DATA FORMAT 4?
13269  055522  001456                        BEQ     40$             ;BRANCH IF YES
13270  055524  122712  000005                CMPB    #5,(R2)         ;DATA FORMAT 5?
13271  055530  001465                        BEQ     60$             ;BRANCH IF YES
13272                                  ;;*****************************************************************
13273                                  ;DATA FORMAT 0
13274  055532  005202                        INC     R2              ;INCREMENT FORMAT POINTER
13275  055534  013146                        MOV     @(R1)+,-(SP)    ;PUSH DATA TO BE TYPED
13276  055536  104402                        TYPOC
13277  055540  005711                13$:    TST     (R1)            ;ANY MORE DATA?
13278  055542  001747                        BEQ     6$              ;BRANCH IF NO
13279  055544  104400  056404                TYPE    ,8$             ;TYPE TWO SPACES
13280  055550  000751                        BR      10$
13281                                  ;;*****************************************************************
13282                                  ;DATA FORMAT 1
13283  055552  005202                9$:     INC     R2              ;INCREMENT FORMAT POINTER
13284  055554  004737  062556                JSR     PC,@#CNVADR     ;GET 22 BIT ADR
13285  055560  012746  001574        14$:    MOV     #PA1500,-(SP)   ;PUSH ADR OF 22 BIT ADR
13286  055564  004737  057670                JSR     PC,@#$DB2O      ;CONVERT TO ASCII
13287  055570  062716  000003                ADD     #3,(SP)         ;DELETE LEADING ZEROS
13288  055574  012667  000002                MOV     (SP)+,12$       ;GET ADR OF ASCII STRING
13289  055600  104400                        TYPE
13290  055602  000000                12$:    .WORD
13291  055604  062701  000002                ADD     #2,R1           ;INCREMENT R1
13292  055610  000753                        BR      13$
13293                                  ;;*****************************************************************
13294                                  ;DATA FORMAT 2
13295  055612  005202                11$:    INC     R2              ;INCREMENT FORMAT POINTER
13296  055614  011100                        MOV     (R1),R0
13297  055616  012037  001574                MOV     (R0)+,@#PA1500
13298  055622  011037  001576                MOV     (R0),@#PA2116
13299  055626  000754                        BR      14$
13300                                  ;;*****************************************************************
13301                                  ;DATA FORMAT 3
13302  055630  005202                24$:    INC     R2              ;INCREMENT FORMAT POINTER
```

```
13303  055632  013167  000016                MOV     @(R1)+,25$      ;GET DEVICE ID
13304  055636  062767  064572  000010        ADD     #MSGINX,25$     ;FORM ADR OF ASCIZ ADR
13305  055644  017767  000004  000002        MOV     @25$,25$        ;GET ADR OF ASCIZ
13306  055652  104400                        TYPE
13307  055654  000000                25$:    .WORD
13308  055656  000730                        BR      13$             ;CONTINUE
13309                                ;;*****************************************************************
13310                                ;DATA FORMAT 4
13311  055660  005202                40$:    INC     R2
13312  055662  012167  000002                MOV     (R1)+,44$       ;GET ADDRESS OF DATA
13313  055666  104424                        FL2O                    ;CONVERT TO FLOATING FORMAT
13314  055670  000000                44$:    .WORD
13315  055672  012667  000002                MOV     (SP)+,45$       ;GET ADDRESS OF ASCIZ STRING
13316  055676  104400                        TYPE                    ;TYPE THE DATA
13317  055700  000000                45$:    .WORD
13318  055702  000716                        BR      13$
13319                                ;;*****************************************************************
13320                                ;DATA FORMAT 5
13321  055704  005202                60$:    INC     R2              ;INCREMENT FORMAT POINTER
13322  055706  012167  000002                MOV     (R1)+,61$       ;GET ADDRESS OF DATA
13323  055712  104426                        FLD2O                   ;CONVERT TO FLOATING ASCIZ
13324  055714  000000                61$:    .WORD
13325  055716  012667  000002                MOV     (SP)+,62$       ;GET ADDRESS OF ASCIZ STRING
13326  055722  104400                        TYPE                    ;TYPE THE DATA
13327  055724  000000                62$:    .WORD
13328  055726  000704                        BR      13$
13329                                ;;*****************************************************************
13330                                ;ERROR 7 DECODE
13331  055730  010300                15$:    MOV     R3,R0           ;SAVE R3
13332  055732  062700  064572                ADD     #MSGINX,R0      ;GEN ADRS OF ASCIZ
13333  055736  011067  000002                MOV     (R0),16$
13334  055742  104400                        TYPE
13335  055744  000000                16$:    .WORD
13336  055746  104400  055754                TYPE    ,65$            ;;TYPE ASCIZ STRING
  (1)  055752  000404                        BR      64$             ;;GET OVER THE ASCIZ
  (1)                           ;;65$:    .ASCIZ  /FAILED/<CRLF>
  (1)  055764                   64$:
13337  055764  010300                        MOV     R3,R0           ;SAVE DEVICE ID
13338  055766  022700  000010                CMP     #10,R0          ;MASS BUS DEVICE?
13339  055772  003403                        BLE     17$             ;BRANCH IF YES
13340  055774  104400  064761                TYPE    ,MSG12
13341  056000  000411                        BR      18$
13342                                ;;*****************************************************************
13343                                ;MASS BUS ERR
13344  056002  022703  000020       17$:     CMP     #20,R3          ;MBT ERROR?
13345  056006  001426                        BEQ     26$             ;BRANCH IF MBT ERROR
13346  056010  002435                        BLT     27$             ;BRANCH IF UBE ERROR
13347  056012  104400  065074                TYPE    ,MSG13
13348  056016  022700  000012                CMP     #12,R0          ;WAS IT RS?
13349  056022  001140                        BNE     29$             ;BRANCH IF NO
13350                                ;;*****************************************************************
13351                                ;UNIBUS ERROR OR RS04 ERROR
13352  056024  062700  064546       18$:     ADD     #REGINX,R0      ;FORM ADR OF REG TABLE
13353  056030  011000                        MOV     (R0),R0         ;GET ADR OF REG TABLE
13354  056032  022703  000002                CMP     #2,R3           ;RP3 OR RK?
13355  056036  001404                        BEQ     20$             ;BRANCH IF RK
```

```
13356  056040  100406                          BMI     21$             ;BRANCH IF NOT RP03
13357  056042  012704  000007                  MOV     #7,R4           ;SET RP03 SOB COUNT
13358  056046  000423                          BR      22$
13359  056050  012704  000006          20$:    MOV     #6,R4           ;SET RK05 SOB COUNT
13360  056054  000420                          BR      22$
13361  056056  012704  000011          21$:    MOV     #11,R4          ;SET RS04 SOB COUNT
13362  056062  000415                          BR      22$
13363                                  ;;*****************************************************************
13364                                  ;MBT ERROR
13365  056064  104400  065263          26$:    TYPE    ,MSG16
13366  056070  012704  000011                  MOV     #11,R4          ;SET MBT SOB COUNT
13367  056074  062700  064546          28$:    ADD     #REGINX,R0
13368  056100  011000                          MOV     (R0),R0         ;GET ADR OF MBT TABLE
13369  056102  000405                          BR      22$             ;GO TYPE REGISTERS
13370                                  ;UNIBUS  EXERCISER ERROR
13371  056104  104400  065372          27$:    TYPE    ,MSG17
13372  056110  012704  000004                  MOV     #4,R4           ;SET UBE SOB COUNT
13373  056114  000767                          BR      28$             ;GO TYPE UBE REGISTERS
13374  056116  013046                  22$:    MOV     @(R0)+,-(SP)    ;GET DATA IN REG
13375  056120  104402                          TYPOC                   ;TYPE IT
13376  056122  104400  056404                  TYPE    ,8$             ;TYPE TWO SPACES
13377  056126  077405                          SOB     R4,22$          ;CONTINUE
13378                                  ;;*****************************************************************
13379                                  ;THIS CODE TYPES A PHYSICAL BUS ADDRESS IF THE ERROR WAS AN RP03, RK05, OR UBE
13380
13381  056130  022703  000022                  CMP     #22,R3          ;UBE ERROR?
13382  056134  001454                          BEQ     73$             ;BRANCH IF YES
13383  056136  022703  000002                  CMP     #2,R3           ;RK05?
13384  056142  002445                          BLT     32$             ;BRANCH IF NOT RK OR RP03
13385  056144  001005                          BNE     70$             ;BRANCH IF RP03
13386                                  ;RK05 ERROR
13387  056146  104400  065566                  TYPE    ,MSG22
13388  056152  012700  002236                  MOV     #RKCS,R0        ;GET ADR OF ADR OF RKCS REG
13389  056156  000404                          BR      71$
13390                                  ;RP03 ERROR
13391  056160  012700  002214          70$:    MOV     #RP3CS,R0       ;GET ADR OF ADR OF RP3CS REG
13392  056164  104400  065576                  TYPE    ,MSG23
13393                                  ;GET, CALCULATE, & TYPE PHYSICAL BUS ADDRESS
13394  056170  013001                  71$:    MOV     @(R0)+,R1       ;GET BUS ADR EXTENDED BITS
13395  056172  005720                          TST     (R0)+           ;ADJUST R0
13396  056174  013037  002002                  MOV     @(R0)+,@#ERRBA  ;GET BUS ADRESS THAT FAILED
13397  056200  072127  177774                  ASH     #-4,R1          ;GET BITS 4&5 INTO BITS 0&1
13398  056204  042701  177774                  BIC     #177774,R1      ;GET RID OF UNUSED BITS
13399  056210  010137  002004                  MOV     R1,@#ERRBA+2    ;SAVE EXTENDED BITS
13400  056214  162737  000002  002002  74$:    SUB     #2,@#ERRBA      ;DECREMENT BUS ADR
13401  056222  005637  002004                  SBC     @#ERRBA+2
13402  056226  004737  062450                  JSR     PC,@#PHYMAP     ;GO CONVERT TO 22 BIT PHYSICAL
13403  056232  012746  001574                  MOV     #PA1500,-(SP)
13404  056236  004737  057670                  JSR     PC,@#$DB2O      ;CONVERT TO ASCIZ STRING
13405  056242  062716  000003                  ADD     #3,(SP)         ;GET RID OF LEADING ZEROS
13406  056246  012667  000002                  MOV     (SP)+,72$
13407  056252  104400                          TYPE
13408  056254  000000          72$:    .WORD
13409  056256  104400  001407          32$:    TYPE    ,$CRLF
13410  056262  000167  177174                  JMP     6$              ;EXIT
13411                                  ;GET UBE VIRTUAL ADDRESS
```

```
13412  056266  012700  002342          73$:    MOV     #UBETBL+2,R0    ;GET ADR OF UBE TABLE +2
13413  056272  013037  002002                  MOV     @(R0)+,@#ERRBA  ;GET BUS ADR THAT FAILED
13414  056276  013037  002004                  MOV     @(R0)+,@#ERRBA+2;GET BAE BITS
13415  056302  042737  177774  002004          BIC     #177774,@#ERRBA+2;MASK OFF ADR BITS
13416  056310  162737  000002  002002          SUB     #2,@#ERRBA
13417  056316  005637  002004                  SBC     @#ERRBA+2
13418  056322  000734                          BR      74$             ;GO CONVERT & TYPE PHYSICAL ADR
13419                                  ;;**********************************************************************
13420                                  ;RP04 ERROR
13421  056324  062700  064546          29$:    ADD     #REGINX,R0
13422  056330  011000                          MOV     (R0),R0         ;FORM ADR OF RP04 TABLE
13423  056332  012704  000011                  MOV     #11,R4          ;SET SOB COUNT
13424  056336  013046                  31$:    MOV     @(R0)+,-(SP)    ;GET DATA TO BE TYPED
13425  056340  104402                          TYPOC                   ;TYPE DATA
13426  056342  104400  056404                  TYPE    ,8$
13427  056346  077405                          SOB     R4,31$          ;CONTINUE
13428  056350  104400  001407                  TYPE    ,$CRLF
13429  056354  104400  001407                  TYPE    ,$CRLF
13430  056360  012704  000004                  MOV     #4,R4           ;SET SOB COUNT
13431  056364  104400  065204                  TYPE    ,MSG14
13432  056370  013046                  50$:    MOV     @(R0)+,-(SP)    ;GET DTA TO BE TYPED
13433  056372  104402                          TYPOC                   ;TYPE IT
13434  056374  104400  056404                  TYPE    ,8$
13435  056400  077405                          SOB     R4,50$          ;CONTINUE
13436  056402  000725                          BR      32$
13437  056404  020040      000         8$:     .ASCIZ  /  /            ;;TWO(2) SPACES
13438          056410                          .EVEN
13439                                  ;;**********************************************************************
  (1)
  (1)                                   .SBTTL  TYPE ROUTINE
  (1)
  (1)                                   ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
  (1)                                   ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
  (1)                                   ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
  (1)                                   ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
  (1)                                   ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
  (1)                                   ;*
  (1)                                   ;*CALL:
  (1)                                   ;*1) USING A TRAP INSTRUCTION
  (1)                                   ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
  (1)                                   ;*OR
  (1)                                   ;*      TYPE
  (1)                                   ;*      MESADR
  (1)                                   ;*
  (1)                                   ;*2) USING A JSR INSTRUCTION
  (1)                                   ;*      MOV     PS,-(SP)        ;;PUSH PROCESSOR STATUS WORD ON THE STACK
  (1)                                   ;*      JSR     PC,$TYPE        ;;CALL TYPE ROUTINE
  (1)                                   ;*      MESADDR                 ;;FIRST ADRESS OF MESSAGE
  (1)
  (1)   056410  105767  122707          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
  (1)   056414  100002                          BPL     1$              ;;BR IF YES
  (1)   056416  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
  (1)   056420  000407                          BR      3$              ;;LEAVE
  (1)   056422  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
  (1)   056424  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
  (1)   056430  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
```

```
(1)  056432  001005                        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
(1)  056434  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
(1)  056436  012600                        MOV     (SP)+,R0        ;;RESTORE R0
(1)  056440  062716  000002          3$:   ADD     #2,(SP)         ;;ADJUST RETURN PC
(1)  056444  000002                        RTI                     ;;RETURN
(1)  056446  122716  000011          4$:   CMPB    #HT,(SP)        ;;BRANCH IF <HT>
(1)  056452  001426                        BEQ     8$
(1)  056454  122716  000200                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT
(1)  056460  001004                        BNE     5$
(1)  056462  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
(1)  056464  104400  001407                TYPE    ,$CRLF
(1)  056470  000757                        BR      2$              ;;GET NEXT CHARACTER
(1)  056472  004767  000056          5$:   JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
(1)  056476  126726  122620          6$:   CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
(1)  056502  001352                        BNE     2$              ;;IF NO GO GET NEXT CHAR.
(1)  056504  016746  122610                MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
(1)                                                                 ;;AND THE NULL CHAR.
(1)  056510  105366  000001          7$:   DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
(1)  056514  002770                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
(1)  056516  004767  000032                JSR     PC,$TYPEC       ;;GO TYPE A NULL
(1)  056522  105367  000100                DECB    $CHARCNT        ;;DON'T COUNT THE NULL AS A CHARACTER
(1)  056526  000770                        BR      7$              ;;LOOP
(1)
(1)                                 ;;HORIZONTAL TAB PROCESSOR
(1)
(1)  056530  112716  000040          8$:   MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
(1)  056534  004767  000014          9$:   JSR     PC,$TYPEC       ;;TYPE A SPACE
(1)  056540  132767  000007  000060        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
(1)  056546  001372                        BNE     9$              ;;TAB STOP
(1)  056550  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
(1)  056552  000726                        BR      2$              ;;GET NEXT CHARACTER
(1)  056554  005737  001546          $TYPEC: TST   @#NOTYPE        ;;INHIBIT TYPING?
(1)  056560  100423                        BMI     $TYPEX          ;;BRANCH IF YES
(1)  056562  105777  122526                TSTB    @$TPS           ;WAIT UNTIL PRINTER IS READY
(1)  056566  100372                        BPL     $TYPEC
(1)  056570  116677  000002  122520        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
(1)  056576  122766  000015  000002        CMPB    #CR,2(SP)       ;;BRANCH IF
(1)  056604  001003                        BNE     1$              ;;NOT <CR>
(1)  056606  105067  000014                CLRB    $CHARCNT        ;;
(1)  056612  000406                        BR      $TYPEX          ;;EXIT
(1)  056614  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;BRANCH IF
(1)  056622  001402                        BEQ     $TYPEX          ;;<LF>
(1)  056624  105227                        INCB    (PC)+           ;;INC SPACE
(1)  056626  000000                $CHARCNT:.WORD  0               ;;COUNT
(1)  056630  000207                $TYPEX: RTS     PC
(1)
13440                               ;;**************************************************************************
13441                               .SBTTL  ROUTINE TO TYPE THE ELAPSED RUN TIME OF THE PROGRAM
13442                               ;*      THIS ROUTINE CONVERTS THE CONTENTS OF LOCATIONS 'LTICKS''
13443                               ;*      AND 'MTICKS'' TO SECONDS AND MINUTES/HOURS RESPECTIVELY
13444                               ;*      AND TYPES THEM IN THE FOLLOWING FORMAT:
13445                               ;*                  HHH:MM:SS
13446                               ;;**************************************************************************
13447  056632  104420                TYPTIME:SAVREG
13448  056634  004737  064104                JSR     PC,@#LDKT       ;GO BACK TO LOW CORE
13449  056640  113701  001701                MOVB    @#LTICKS+1,R1   ;GET SECOND COUNT
```

```
13450  056644  005000                        CLR     R0
13451  056646  071027  000012                DIV     #10.,R0
13452  056652  062701  000060                ADD     #60,R1
13453  056656  110137  057070                MOVB    R1,@#TIMEBUF+10
13454  056662  010001                        MOV     R0,R1
13455  056664  005000                        CLR     R0
13456  056666  071027  000006                DIV     #6,R0
13457  056672  062701  000060                ADD     #60,R1
13458  056676  110137  057067                MOVB    R1,@#TIMEBUF+7
13459  056702  013701  001676                MOV     @#MTICKS,R1     ;GET MINUTE COUNT
13460  056706  005000                        CLR     R0
13461  056710  071027  000012                DIV     #10.,R0         ;GET HOURS AND MINUTES
13462  056714  062701  000060                ADD     #60,R1          ;MAKE REMAINDER ASCII
13463  056720  110167  000141                MOVB    R1,TIMEBUF+5    ;PUT IN BUFFER
13464  056724  010001                        MOV     R0,R1
13465
13466  056726  005000                        CLR     R0
13467  056730  071027  000006                DIV     #6.,R0
13468  056734  062701  000060                ADD     #60,R1
13469  056740  110167  000120                MOVB    R1,TIMEBUF+4
13470  056744  005700                        TST     R0
13471  056746  001434                        BEQ     2$
13472  056750  010001                        MOV     R0,R1
13473
13474  056752  005000                        CLR     R0
13475
13476  056754  071027  000012                DIV     #10.,R0
13477  056760  062701  000060                ADD     #60,R1
13478  056764  110167  000072                MOVB    R1,TIMEBUF+2
13479  056770  005700                        TST     R0
13480  056772  001422                        BEQ     2$
13481  056774  010001                        MOV     R0,R1
13482  056776  005000                        CLR     R0
13483  057000  071027  000010                DIV     #10,R0
13484  057004  062701  000060                ADD     #60,R1
13485  057010  110167  000045                MOVB    R1,TIMEBUF+1
13486  057014  005700                        TST     R0
13487  057016  001410                        BEQ     2$
13488  057020  010001                        MOV     R0,R1
13489  057022  005000                        CLR     R0
13490  057024  071027  000012                DIV     #10.,R0
13491  057030  062701  000060                ADD     #60,R1
13492  057034  110167  000020                MOVB    R1,TIMEBUF
13493  057040  104400  057060        2$:     TYPE    ,TIMEBUF
13494  057044  104400  001407                TYPE    ,$CRLF
13495  057050  004737  064202                JSR     PC,@#RESKT      ;GO BACK TO ORIGINAL MEMORY
13496  057054  104422                        RESREG
13497  057056  000207                        RTS     PC
13498  057060    001     001     001 TIMEBUF:.BYTE   1,1,1,72,1,1,72,60,60,0
       057063    072     001     001
       057066    072     060     060
       057071    000
13499                                       .EVEN
13500                                   ;;*****************************************************************************
13501                                   .SBTTL  ROUTINE TO TYPE THE AVAILABLE DEVICES AND UNIT NUMBERS
13502                                   ;*      THIS ROUTINE SEARCHES THE SYSTEM SIZE TABLE FOR NON-
```

```
13503                                   ;*      ZERO ENTRIES.WHEN IT FINDS ONE, IT TYPES THE NAME OF THE
13504                                   ;*      DEVICE AND THE UNIT NUMBERS THAT WERE FOUND TO BE
13505                                   ;*      AVAILABLE FOR THAT DEVICE.
13506                                   ;;*****************************************************************
13507  057072  104400  067330          TYPSIZ: TYPE    ,SWITCH
13508  057076  104400  070651                  TYPE    ,MSG30          ;NOTE SWITCH REG BIT 8 REVERSAL
13509  057102  104400  070516                  TYPE    ,MSG4
13510  057106  012700  000010                  MOV     #10,R0          ;SET SOB COUNT
13511  057112  005001                          CLR     R1
13512  057114  105761  001712          1$:     TSTB    SYSSIZE(R1)     ;DEVICE AVAILABLE?
13513  057120  001004                          BNE     2$              ;BRANCH IF YES
13514  057122  062701  000002          7$:     ADD     #2,R1           ;INCREMENT INDEX
13515  057126  077006                          SOB     R0,1$           ;CONTINUE
13516  057130  000207                          RTS     PC              ;RETURN
13517  057132  010102                  2$:     MOV     R1,R2           ;GET INDEX
13518  057134  062702  064572                  ADD     #MSGINX,R2      ;GET ADR OF MESSAGE ADR
13519  057140  011267  000002                  MOV     (R2),3$         ;GET ADDRESS OF MESSAGE
13520  057144  104400                          TYPE
13521  057146  000000                  3$:     .WORD
13522  057150  112767  000060  000034        MOVB    #60,4$          ;INIT UNIT NO. BUFFER (ASCII)
13523  057156  116102  001712                  MOVB    SYSSIZE(R1),R2  ;GET WORD WITH AVAILABLE UNITS
13524  057162  012703  000010                  MOV     #10,R3          ;SET SOB COUNT
13525  057166  006002                  6$:     ROR     R2              ;GET UNITS
13526  057170  103002                          BCC     5$              ;BRANCH IF NOT A UNIT
13527  057172  104400  057212                  TYPE    ,4$
13528  057176  005267  000010          5$:     INC     4$
13529  057202  077307                          SOB     R3,6$           ;CONTINUE
13530  057204  104400  001407                  TYPE    ,$CRLF
13531  057210  000744                          BR      7$
13532  057212     000     054     040  4$:     .BYTE   0,54,40,0       ;NUMBER,COMMA,SPACE,TERMINATOR
       057215     000
13533                                   ;;*****************************************************************
  (1)
  (1)                                   .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
  (1)
  (1)                                   ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
  (1)                                   ;*OCTAL (ASCII) NUMBER AND TYPE IT.
  (1)                                   ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
  (1)                                   ;*CALL:
  (1)                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
  (1)                                   ;*      TYPOS                   ;;CALL FOR TYPEOUT
  (1)                                   ;*      .BYTE   N               ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
  (1)                                   ;*      .BYTE   M               ;;M=1 OR 0
  (1)                                   ;*                                      ;;1=TYPE LEADING ZEROS
  (1)                                   ;*                                      ;;0=SUPPRESS LEADING ZEROS
  (1)                                   ;*
  (1)                                   ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
  (1)                                   ;*$TYPOS OR $TYPOC
  (1)                                   ;*CALL:
  (1)                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
  (1)                                   ;*      TYPON                   ;;CALL FOR TYPEOUT
  (1)                                   ;*
  (1)                                   ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
  (1)                                   ;*CALL:
  (1)                                   ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
  (1)                                   ;*      TYPOC                   ;;CALL FOR TYPEOUT
```

```
(1)
(1)  057216  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
(1)  057222  116667  000001  000211          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
(1)  057230  112667  000207                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
(1)  057234  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
(1)  057240  000406                          BR      $TYPON
(1)  057242  112767  000001  000171  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
(1)  057250  112767  000006  000165          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
(1)  057256  112767  000005  000154  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
(1)  057264  010346                          MOV     R3,-(SP)        ;;SAVE R3
(1)  057266  010446                          MOV     R4,-(SP)        ;;SAVE R4
(1)  057270  010546                          MOV     R5,-(SP)        ;;SAVE R5
(1)  057272  116704  000145                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
(1)  057276  005404                          NEG     R4
(1)  057300  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
(1)  057304  110467  000132                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
(1)  057310  116704  000125                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
(1)  057314  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
(1)  057320  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
(1)  057322  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
(1)  057324  000404                          BR      3$              ;;GO DO MSB
(1)  057326  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
(1)  057330  006105                          ROL     R5
(1)  057332  006105                          ROL     R5
(1)  057334  010503                          MOV     R5,R3
(1)  057336  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
(1)  057340  105367  000076                  DECB    $OMODE          ;;TYPE THIS DIGIT?
(1)  057344  100016                          BPL     7$              ;;BR IF NO
(1)  057346  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
(1)  057352  001002                          BNE     4$              ;;TEST FOR 0
(1)  057354  005704                          TST     R4              ;;SUPPRESS THIS 0?
(1)  057356  001403                          BEQ     5$              ;;BR IF YES
(1)  057360  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
(1)  057362  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
(1)  057366  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
(1)  057372  110367  000040                  MOVB    R3,8$           ;;SAVE FOR TYPING
(1)  057376  104400  057436                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
(1)  057402  105367  000032          7$:     DECB    $OCNT           ;;COUNT BY 1
(1)  057406  003347                          BGT     2$              ;;BR IF MORE TO DO
(1)  057410  002402                          BLT     6$              ;;BR IF DONE
(1)  057412  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
(1)  057414  000744                          BR      2$              ;;GO DO THE LAST DIGIT
(1)  057416  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
(1)  057420  012604                          MOV     (SP)+,R4        ;;RESTORE R4
(1)  057422  012603                          MOV     (SP)+,R3        ;;RESTORE R3
(1)  057424  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
(1)  057432  012616                          MOV     (SP)+,(SP)
(1)  057434  000002                          RTI                     ;;RETURN
(1)  057436     000                  8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
(1)  057437     000                          .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
(1)  057440     000                  $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
(1)  057441     000                  $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
(1)  057442  000000                  $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
13534                                ;;**************************************************************************
(1)
(1)                                  .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
```

```
(1)
(1)                                     ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
(1)                                     ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
(1)                                     ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
(1)                                     ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
(1)                                     ;*REPLACED WITH SPACES.
(1)                                     ;*CALL:
(1)                                     ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
(1)                                     ;*      TYPDS                   ;;GO TO THE ROUTINE
(1)
(1)  057444                             $TYPDS:
(3)  057444  010046                             MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  057446  010146                             MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  057450  010246                             MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  057452  010346                             MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  057454  010546                             MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(1)  057456  012746  020200                     MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
(1)  057462  016605  000020                     MOV     20(SP),R5       ;;GET THE INPUT NUMBER
(1)  057466  100004                             BPL     1$              ;;BR IF INPUT IS POS.
(1)  057470  005405                             NEG     R5              ;;MAKE THE BINARY NUMBER POS.
(1)  057472  112766  000055  000001             MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
(1)  057500  005000                     1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
(1)  057502  012703  057660                     MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
(1)  057506  112723  000040                     MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
(1)  057512  005002                     2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
(1)  057514  016001  057650                     MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
(1)  057520  160105                     3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
(1)  057522  002402                             BLT     4$              ;;BR IF DONE
(1)  057524  005202                             INC     R2              ;;INCREASE THE BCD DIGIT BY 1
(1)  057526  000774                             BR      3$
(1)  057530  060105                     4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
(1)  057532  005702                             TST     R2              ;;CHECK IF BCD DIGIT=0
(1)  057534  001002                             BNE     5$              ;;FALL THROUGH IF 0
(1)  057536  105716                             TSTB    (SP)            ;;STILL DOING LEADING 0'S?
(1)  057540  100407                             BMI     7$              ;;BR IF YES
(1)  057542  106316                     5$:     ASLB    (SP)            ;;MSD?
(1)  057544  103003                             BCC     6$              ;;BR IF NO
(1)  057546  116663  000001  177777             MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
(1)  057554  052702  000060             6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
(1)  057560  052702  000040             7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
(1)  057564  110223                             MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
(1)  057566  005720                             TST     (R0)+           ;;JUST INCREMENTING
(1)  057570  020027  000010                     CMP     R0,#10          ;;CHECK THE TABLE INDEX
(1)  057574  002746                             BLT     2$              ;;GO DO THE NEXT DIGIT
(1)  057576  003002                             BGT     8$              ;;GO TO EXIT
(1)  057600  010502                             MOV     R5,R2           ;;GET THE LSD
(1)  057602  000764                             BR      6$              ;;GO CHANGE TO ASCII
(1)  057604  105726                     8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
(1)  057606  100003                             BPL     9$              ;;BR IF NO
(1)  057610  116663  177777  177776             MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
(1)  057616  105013                     9$:     CLRB    (R3)            ;;SET THE TERMINATOR
(3)  057620  012605                             MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  057622  012603                             MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  057624  012602                             MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  057626  012601                             MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  057630  012600                             MOV     (SP)+,R0        ;;POP STACK INTO R0
```

```
(1)  057632  104400  057660                 TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
(1)  057636  016666  000002  000004         MOV     2(SP),4(SP)     ;;ADJUST THE STACK
(1)  057644  012616                         MOV     (SP)+,(SP)
(1)  057646  000002                         RTI                     ;;RETURN TO USER
(1)  057650  023420                 $DTBL:  10000.
(1)  057652  001750                         1000.
(1)  057654  000144                         100.
(1)  057656  000012                         10.
(1)  057660  000004                 $DBLK:  .BLKW   4
13535                                       ;;**************************************************************************
(1)                                         
(1)                                         .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
(1)                                         
(1)                                         ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
(1)                                         ;*UNSIGNED OCTAL ASCIZ NUMBER.
(1)                                         ;*CALL
(1)                                         ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
(1)                                         ;*      JSR     PC,@#$DB2O      ;;CALL THE ROUTINE
(1)                                         ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK
(1)                                         
(1)                                         
(1)  057670  104420                 $DB2O:  SAVREG                  ;;SAVE ALL REGISTERS
(1)  057672  016601  000002                 MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
(1)  057676  012705  060025                 MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
(1)  057702  012704  000014                 MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
(1)  057706  012703  177770                 MOV     #^C7,R3         ;;MASK
(1)  057712  012100                         MOV     (R1)+,R0        ;;LOWER WORD
(1)  057714  012101                         MOV     (R1)+,R1        ;;HIGH WORD
(1)  057716  005002                         CLR     R2              ;;TERMINATOR
(1)  057720  110245                 1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
(1)  057722  010002                         MOV     R0,R2           ;;GET THIS DIGIT
(1)  057724  005304                         DEC     R4              ;;COUNT THIS CHARACTER
(1)  057726  003016                         BGT     3$              ;;BR IF NOT THE LAST DIGIT
(1)  057730  001414                         BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
(1)  057732  005205                         INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
(1)  057734  010566  000002                 MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
(1)  057740  122765  000061  000003         CMPB    #61,3(R5)       ;LAST NUBER LEGAL?
(1)  057746  002003                         BGE     4$              ;BRANCH IF YES
(1)  057750  112765  000060  000003         MOVB    #60,3(R5)       ;MAKE IT ZERO
(1)  057756  104422                 4$:     RESREG                  ;;RESTORE ALL REGISTERS
(1)  057760  000207                         RTS     PC              ;;RETURN TO USER
(1)  057762  006203                 2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
(1)  057764  006001                 3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
(1)  057766  006000                         ROR     R0              ;;      THE NEXT OCTAL DIGIT
(1)  057770  006001                         ROR     R1
(1)  057772  006000                         ROR     R0
(1)  057774  006001                         ROR     R1
(1)  057776  006000                         ROR     R0
(1)  060000  040302                         BIC     R3,R2           ;;MASK OUT ALL JUNK
(1)  060002  062702  000060                 ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
(1)  060006  000744                         BR      1$              ;;GO PUT IT IN THE DATA TABLE
(1)  060010  000016                 $OCTVL: .BLKB   14.             ;;RESERVE DATA TABLE
13536                                       ;;**************************************************************************
(1)                                         
(1)                                         .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
(1)                                         
```

```
(1)                                     ;*SAVE R0-R5
(1)                                     ;*CALL:
(1)                                     ;*      SAVREG
(1)                                     ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
(1)                                     ;*
(1)                                     ;*TOP---(+16)
(1)                                     ;* +2---(+18)
(1)                                     ;* +4---R5
(1)                                     ;* +6---R4
(1)                                     ;* +8---R3
(1)                                     ;*+10---R2
(1)                                     ;*+12---R1
(1)                                     ;*+14---R0
(1)
(1)  060026                             $SAVREG:
(3)  060026  010046                             MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  060030  010146                             MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  060032  010246                             MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(3)  060034  010346                             MOV     R3,-(SP)        ;;PUSH R3 ON STACK
(3)  060036  010446                             MOV     R4,-(SP)        ;;PUSH R4 ON STACK
(3)  060040  010546                             MOV     R5,-(SP)        ;;PUSH R5 ON STACK
(1)  060042  016646  000022                     MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
(1)  060046  016646  000022                     MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
(1)  060052  016646  000022                     MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
(1)  060056  016646  000022                     MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
(1)  060062  000002                             RTI
(1)
(1)                                     ;*RESTORE R0-R5
(1)                                     ;*CALL:
(1)                                     ;*      RESREG
(1)  060064                             $RESREG:
(1)  060064  012666  000022                     MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
(1)  060070  012666  000022                     MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
(1)  060074  012666  000022                     MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
(1)  060100  012666  000022                     MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
(3)  060104  012605                             MOV     (SP)+,R5        ;;POP STACK INTO R5
(3)  060106  012604                             MOV     (SP)+,R4        ;;POP STACK INTO R4
(3)  060110  012603                             MOV     (SP)+,R3        ;;POP STACK INTO R3
(3)  060112  012602                             MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  060114  012601                             MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  060116  012600                             MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  060120  000002                             RTI
13537                                   ;;*****************************************************************
13538                                   .SBTTL  CONVERT FLOATING BINARY TO OCTAL ASCIZ
13539                                   ;*
13540                                   ;*THIS ROUTINE CONVERTS A 32 BIT FLOATING NUMBER TO AN OCTAL
13541                                   ;*ASCIZ STRING IN THE FOLLOWING FORMAT:
13542                                   ;*
13543                                   ;*              W  XXX  YYY ZZZZZZ
13544                                   ;*
13545                                   ;*      WHERE   W = SIGN BIT
13546                                   ;*              X = 8-BIT EXPONENT (RIGHT JUSTIFIED)
13547                                   ;*              Y = FRACTION BITS <57:51> (RIGHT JUSTIFIED)
13548                                   ;*              Z = FRACTION BITS <50:35>
13549                                   ;*
13550                                   ;*IT IS ENTERED BY A TRAP CALL WITH THE ADDRESS OF THE FLOATING
```

```
13551                                   ;*NUMBER IN THE WORD FOLLOWING THE CALL.
13552                                   ;*IT RETURNS WITH THE ADDRESS OF THE ASCIZ STRING ON THE STACK.
13553                                   ;;*****************************************************************
13554  060122  104420                  $FL20:  SAVREG
13555  060124  017600  000000                  MOV     @(SP),R0        ;GET ADDRESS OF DATA
13556  060130  062716  000002                  ADD     #2,(SP)         ;ADJUST RETURN PC
13557  060134  016001  000002                  MOV     2(R0),R1        ;PUT SECOND DATA WORD IN R1
13558  060140  011000                          MOV     (R0),R0         ;PUT FIRST DATA WORD IN R0
13559  060142  012704  001437                  MOV     #$FLBUFF+23,R4  ;GET ADDRESS OF BUFFER END IN R4
13560  060146  112744  000000                  MOVB    #0,-(R4)        ;PUT TERMINATOR IN BUFFER
13561  060152  012705  000005                  MOV     #5,R5           ;SET SOB COUNT FOR FRACTION DIGITS
13562  060156  010103                  1$:     MOV     R1,R3           ;GET LSB'S OF FRACTION
13563  060160  042703  177770                  BIC     #^C7,R3         ;SAVE LS 3 BITS
13564  060164  062703  000060                  ADD     #60,R3          ;MAKE THEM ASCII
13565  060170  110344                          MOVB    R3,-(R4)        ;STORE IN BUFFER
13566  060172  073027  177775                  ASHC    #-3,R0          ;SHIFT NUMBER TO NEXT 3 BITS
13567  060176  077511                          SOB     R5,1$           ;CONTINUE FOR 7 DIGITS
13568  060200  010103                          MOV     R1,R3           ;GET NEXT DIGITS
13569  060202  042703  177776                  BIC     #^C1,R3         ;ONLY WANT 1 BIT
13570  060206  062703  000060                  ADD     #60,R3          ;MAKE THEM ASCII
13571  060212  110344                          MOVB    R3,-(R4)        ;STORE IN BUFFER
13572  060214  112744  000040                  MOVB    #40,-(R4)       ;PUT SPACE IN BUFFER
13573  060220  073027  177777                  ASHC    #-1,R0
13574  060224  012705  000002                  MOV     #2,R5           ;SET SOB COUNT
13575  060230  010103                  3$:     MOV     R1,R3           ;GET LOW WORD
13576  060232  042703  177770                  BIC     #^C7,R3         ;MASK 3 BITS
13577  060236  062703  000060                  ADD     #60,R3          ;MAKE THEM ASCII
13578  060242  110344                          MOVB    R3,-(R4)        ;PUT IN BUFFER
13579  060244  073027  177775                  ASHC    #-3,R0          ;GET NEXT 3 BITS
13580  060250  077511                          SOB     R5,3$           ;CONVERT THEM
13581  060252  010103                          MOV     R1,R3           ;
13582  060254  042703  177776                  BIC     #^C1,R3         ;ONLY WANT 1 BIT
13583  060260  062703  000060                  ADD     #60,R3          ;MAKE IT ASCII
13584  060264  110344                          MOVB    R3,-(R4)        ;PUT IN BUFFER
13585  060266  112744  000040                  MOVB    #40,-(R4)       ;PUT SPACE IN BUFFER
13586  060272  112744  000040                  MOVB    #40,-(R4)
13587  060276  072127  177777                  ASH     #-1,R1          ;GET FIRST 3 BITS OF EXPONENT
13588  060302  012705  000002                  MOV     #2,R5           ;SET SOB COUNT FOR 2 DIGITS
13589  060306  010103                  2$:     MOV     R1,R3           ;GET LSB'S OF EXPONENT
13590  060310  042703  177770                  BIC     #^C7,R3         ;SAVE 3 BITS
13591  060314  062703  000060                  ADD     #60,R3          ;MAKE THEM ASCII
13592  060320  110344                          MOVB    R3,-(R4)        ;STORE IN BUFFER
13593  060322  072127  177775                  ASH     #-3,R1          ;GET NEXT 3 BITS
13594  060326  077511                          SOB     R5,2$           ;CONTINUE
13595  060330  010103                          MOV     R1,R3           ;GET LAST 2 BITS OF EXPONENT
13596  060332  042703  177774                  BIC     #^C3,R3         ;MAKE SURE ONLY 2 BITS
13597  060336  062703  000060                  ADD     #60,R3          ;MAKE THEM ASCII
13598  060342  110344                          MOVB    R3,-(R4)        ;STORE IN BUFFER
13599  060344  112744  000040                  MOVB    #40,-(R4)       ;PUT SPACE IN BUFFER
13600  060350  112744  000040                  MOVB    #40,-(R4)
13601  060354  042700  177776                  BIC     #^C1,R0         ;GET SIGN BIT (IT WAS EXTENDED)
13602  060360  062700  000060                  ADD     #60,R0          ;MAKE IT ASCII
13603  060364  110044                          MOVB    R0,-(R4)        ;PUT IT IN THE BUFFER
13604  060366  104422                          RESREG
13605  060370  011646                          MOV     (SP),-(SP)      ;SAVE RETURN PC
13606  060372  016666  000004  000002          MOV     4(SP),2(SP)     ;AND RETURN PSW
```

```
13607  060400  012766  001414  000004          MOV     #$FLBUFF,4(SP)  ;PUT BUFFER ADDRESS ON STACK
13608  060406  000006                          RTT                     ;RETURN
13609                                           .EVEN
13610                                  ;;*****************************************************************
13611                                  .SBTTL  CONVERT FLOATING DOUBLE BINARY TO OCTAL ASCIZ
13612                                  ;*
13613                                  ;*THIS ROUTINE CONVERTS A 64 BIT FLOATING NUMBER TO AN OCTAL
13614                                  ;*ASCIZ STRING IN THE FOLLOWING FORMAT:
13615                                  ;*
13616                                  ;*              U  VVV  WWW XXXXXX YYYYYY ZZZZZZ
13617                                  ;*
13618                                  ;*      WHERE   U = SIGN BIT
13619                                  ;*              V = 8-BIT EXPONENT (RIGHT JUSTIFIED)
13620                                  ;*              W = FRACTION BITS<57:51> (RIGHT JUSTIFIED)
13621                                  ;*              X = FRACTION BITS <50:35>
13622                                  ;*              Y = FRACTION BITS <34:19>
13623                                  ;*              Z = FRACTION BITS <18:03>
13624                                  ;*
13625                                  ;*IT IS ENTERED BY A TRAP CALL WITH THE ADDRESS OF THE FLOATING
13626                                  ;*NUMBER IN THE WORD FOLLOWING THE CALL.
13627                                  ;*IT RETURNS WITH THE ADDRESS OF THE ASCIZ STRING ON THE STACK.
13628                                  ;;*****************************************************************
13629  060410  104420                  $FLD20: SAVREG
13630  060412  017667  000000  000006          MOV     @(SP),1$        ;GET ADDRESS OF DATA TO CONVERT
13631  060420  062716  000002                  ADD     #2,(SP)         ;ADJUST RETURN PC
13632  060424  104424                          FL20                    ;CONVERT MS 32 BITS
13633  060426  000000                  1$:     .WORD
13634  060430  012600                          MOV     (SP)+,R0        ;GET ADDRESS OF CONVERTED DATA
13635  060432  010067  121022                  MOV     R0,$BUFF        ;SAVE IT
13636  060436  062700  000041                  ADD     #41,R0          ;ADJUST TO END OF BUFFER
13637  060442  105040                          CLRB    -(R0)           ;PUT TERMINATOR IN BUFFER
13638  060444  016701  177756                  MOV     1$,R1           ;GET ADDRESS OF DATA TO CONVERT
13639  060450  062701  000004                  ADD     #4,R1           ;ADJUST TO LOWER 32 BITS
13640  060454  012102                          MOV     (R1)+,R2        ;SAVE THE DATA
13641  060456  012103                          MOV     (R1)+,R3
13642  060460  012701  000002                  MOV     #2,R1           ;SET LOOP COUNT
13643  060464  012704  000005          3$:     MOV     #5,R4           ;SET LOOP COUNT
13644  060470  010305          4$:     MOV     R3,R5           ;GET LS 32 BITS OF DATA
13645  060472  042705  177770                  BIC     #^C7,R5         ;MASK 3 BITS
13646  060476  062705  000060                  ADD     #60,R5          ;MAKE THEM ASCII
13647  060502  110540                          MOVB    R5,-(R0)        ;PUT IN BUFFER
13648  060504  073227  177775                  ASHC    #-3,R2          ;GET NEXT 3 BITS
13649  060510  077411                          SOB     R4,4$           ;CONTINUE
13650  060512  010305                          MOV     R3,R5           ;GET LS 32 BITS
13651  060514  042705  177776                  BIC     #^C1,R5         ;ONLY WANT 1 BIT
13652  060520  062705  000060                  ADD     #60,R5          ;MAKE IT ASCII
13653  060524  110540                          MOVB    R5,-(R0)        ;PUT IN TABLE
13654  060526  112740  000040                  MOVB    #40,-(R0)       ;PUT SPACE IN TABLE
13655  060532  073227  177777                  ASHC    #-1,R2
13656  060536  077126                          SOB     R1,3$           ;CONVERT NEXT 16 BITS
13657  060540  104422                          RESREG
13658  060542  011646                          MOV     (SP),-(SP)      ;ADJUST STACK
13659  060544  016666  000004  000002          MOV     4(SP),2(SP)     ;TO RETURN WITH ADDRESS
13660  060552  016766  120702  000004          MOV     $BUFF,4(SP)     ;OF BUFFER ON STACK
13661  060560  000006                          RTT                     ;RETURN
13662
```

```
13663                                   ;;*****************************************************************
  (1)
  (1)                                   .SBTTL  RANDOM NUMBER GENERATOR ROUTINE
  (1)
  (1)                                   ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
  (1)                                   ;*WITH A RANGE OF 0 TO 2(+33)-1.
  (1)                                   ;*CALL:
  (1)                                   ;*      JSR     PC,$RAND        ;;CALL THE ROUTINE
  (1)                                   ;*      RETURN                  ;;RETURN HERE THE RANDOM
  (1)                                   ;*                              ;;NUMBER WILL BE IN
  (1)                                   ;*                              ;;$HINUM,$LONUM
  (1)
  (1)  060562                           $RAND:
  (3)  060562  010046                           MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (3)  060564  010146                           MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (3)  060566  010246                           MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (1)  060570  016700  121024                   MOV     $LONUM,R0       ;;SET R0 WITH LOW
  (1)  060574  016701  121022                   MOV     $HINUM,R1       ;;SET R1 WITH HIGH
  (1)  060600  012702  177771                   MOV     #-7,R2          ;;SET SHIFT COUNT
  (1)  060604  006300                   1$:     ASL     R0              ;;SHIFT R0 LEFT AND
  (1)  060606  006101                           ROL     R1              ;;ROTATE CARRY INTO R1 AND
  (1)  060610  005202                           INC     R2              ;;CHECK FOR DONE
  (1)  060612  001374                           BNE     1$              ;;CONTINUE SHIFT LOOP
  (1)  060614  066700  121000                   ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
  (1)  060620  005501                           ADC     R1              ;;PROPOGATE CARRY
  (1)  060622  066701  120774                   ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
  (1)  060626  062700  001057                   ADD     #1057,R0        ;;ADD LOW CONSTANT
  (1)
  (1)  060632  005501                           ADC     R1              ;;PROPOGATE CARRY
  (1)  060634  062701  047401                   ADD     #47401,R1       ;;ADD HIGH CONSTANT
  (1)  060640  010067  120754                   MOV     R0,$LONUM       ;;SAVE R0
  (1)  060644  010167  120752                   MOV     R1,$HINUM       ;;SAVE R1
  (3)  060650  012602                           MOV     (SP)+,R2        ;;POP STACK INTO R2
  (3)  060652  012601                           MOV     (SP)+,R1        ;;POP STACK INTO R1
  (3)  060654  012600                           MOV     (SP)+,R0        ;;POP STACK INTO R0
  (1)  060656  000207                           RTS     PC              ;;RETURN
13664                                   ;;*****************************************************************
13665                                   .SBTTL  FLOATING POINT NUMBER GENERATOR
13666                                   ;*      THIS ROUTINE GENERATES TWO RANDOM FLOATING POINT NUMBERS
13667                                   ;*      IN EITHER SINGLE OR DOUBLE PRECISION. FOR SINGLE PRECISION
13668                                   ;*      THE NUMBERS ARE STORED IN $TMP0 AND $TMP2. DOUBLE PRECISION
13669                                   ;*      NUMBERS ARE STORED IN $TMP0 AND $TMP4.
13670                                   ;*      IN EITHER SINGLE OR DOUBLE THE EXTENDED EXPONENT IS STORED
13671                                   ;*      IN $REG0 AND $REG1.
13672                                   ;;*****************************************************************
13673  060660  012767  000002  000130  FLTDBL: MOV     #2,SOBDBL       ;SET LOOP FOR 2, FOUR WORD NUMBERS
13674  060666  016700  000124          FLTSGL: MOV     SOBDBL,R0       ;SET WORD LENGTH LOOP
13675  060672  012702  001352                  MOV     #$TMP0,R2       ;GET ADDRESS TO STORE WORDS IN
13676  060676  012701  000002          2$:     MOV     #2,R1           ;SET NUMBER OF WORDS TO 2
13677  060702  004767  177654          1$:     JSR     PC,$RAND        ;GET RANDOM NUMBER
13678  060706  022701  000002                  CMP     #2,R1           ;FIRST TIME?
13679  060712  001404                          BEQ     3$              ;BRANCH IF YES
13680  060714  022767  000002  000074          CMP     #2,SOBDBL       ;DOUBLE PRECISION?
13681  060722  001407                          BEQ     4$              ;BRANCH IF YES
13682  060724  016703  120672          3$:     MOV     $HINUM,R3       ;GET EXPONENT PART
13683  060730  042703  000177                  BIC     #177,R3         ;CHECK FOR MINUS ZERO
```

```
13684  060734  022703  100000                  CMP     #BIT15,R3
13685  060740  001760                          BEQ     1$              ;BRANCH IF MINUS ZERO
13686  060742  016722  120654          4$:     MOV     $HINUM,(R2)+    ;SAVE HINUM
13687  060746  016722  120646                  MOV     $LONUM,(R2)+    ;SAVE LONUM
13688  060752  077125                          SOB     R1,1$           ;CONTINUE
13689  060754  077030                          SOB     R0,2$           ;CONTINUE FOR DOUBLE PREC
13690  060756  012746  001352                  MOV     #$TMP0,-(SP)    ;PUT ADDRESS OF NUMBER ON STACK
13691  060762  012746  001002                  MOV     #1002,-(SP)     ;PUT CONTROL WORD ON STACK
13692  060766  022767  000002  000022          CMP     #2,SOBDBL       ;DOUBLE PREC?
13693  060774  001002                          BNE     5$              ;BRANCH IF NO
13694  060776  012716  001004                  MOV     #1004,(SP)      ;CHANGE CONTROL WORD
13695  061002  004767  000012          5$:     JSR     PC,EXPEXT       ;CALCULATE EXT EXPONENTS
13696  061006  012767  000001  000002          MOV     #1,SOBDBL       ;INIT SOBDBL FOR SINGLE PREC
13697  061014  000207                          RTS     PC              ;RETURN
13698  061016  000001                  SOBDBL: .WORD   1
13699                                  ;;*****************************************************************
13700                                  .SBTTL  FLOATING POINT EXPONENT EXTENSION
13701                                  ;*      THIS ROUTINE CONVERTS THE ACTUAL EXPONENT OF A FLOATING POINT
13702                                  ;*      NUMBER INTO AN ACTUAL EXPONENT OF 200 AND AN EXTENDED
13703                                  ;*      EXPONENT EQUAL TO THE DIFFERANCE BETWEEN THE ORIGINAL
13704                                  ;*      ACTUAL EXPONENT AND 200.
13705                                  ;*
13706                                  ;*      THE ROUTINE IS ENTERED WITH A CONTROL WORD ON THE STACK.
13707                                  ;*      BIT 15 OF THE CONTROL WORD INDICATES WHETHER THE NUMBER
13708                                  ;*      IS IN MEMORY (<15>=0) OR IN AN ACCUMULATOR (<15>=1).
13709                                  ;*      IF THE NUMBER IS IN AN ACCUMULATOR, BITS <9:8> INDICATE
13710                                  ;*      THE ACCUMULATOR NUMBER. IF THE NUMBER(S) IS IN MEMORY,
13711                                  ;*      BITS <9:8> INDICATE THE NUMBER OF NUMBERS TO CONVERT AND
13712                                  ;*      BITS <2:0> INDICATE THE WORD LENGTH OF THE NUMBER(S).
13713                                  ;*      IN THE CASE OF A MEMORY CONVERSION, THE ADDRESS OF THE
13714                                  ;*      FIRST WORD TO CONVERT IS ALSO ON THE STACK (PRECEDING
13715                                  ;*      THE CONTROL WORD).
13716                                  ;;*****************************************************************
13717  061020  012605                  EXPEXT: MOV     (SP)+,R5        ;SAVE RETURN PC
13718  061022  012600                          MOV     (SP)+,R0        ;GET CONTROL WORD
13719  061024  100437                          BMI     1$              ;BRANCH IF ACC CONVERSION
13720  061026  012601                          MOV     (SP)+,R1        ;GET START ADDRESS
13721  061030  162700  000400                  SUB     #400,R0
13722  061034  012702  001352                  MOV     #$TMP0,R2       ;GET OFFSET FROM $TMP0
13723  061040  160102                          SUB     R1,R2
13724  061042  005402                          NEG     R2
13725  061044  006202                          ASR     R2
13726  061046  062702  001326                  ADD     #$REG0,R2       ;GEN ADDRESS OF EXT WORD
13727  061052  011103                  3$:     MOV     (R1),R3         ;GET DATA
13728  061054  042703  100177                  BIC     #100177,R3      ;GET EXPONENT
13729  061060  072327  177771                  ASH     #-7,R3          ;RIGHT JUSTIFY EXPONENT
13730  061064  162703  000200                  SUB     #200,R3         ;CONVERT TO 2'S COMPLIMENT
13731  061070  010312                          MOV     R3,(R2)         ;ADD TO EXTENDED EXPONENT
13732  061072  042711  077600                  BIC     #77600,(R1)     ;MAKE ACTUAL
13733  061076  052711  040000                  BIS     #BIT14,(R1)     ;EXPONENT 200
13734  061102  162700  000400                  SUB     #400,R0         ;ANY MORE WORDS?
13735  061106  100435                          BMI     2$              ;BRANCH IF NO
13736  061110  110003                          MOVB    R0,R3           ;GET WORD LENGTH
13737  061112  006303                          ASL     R3
13738  061114  060301                          ADD     R3,R1           ;SELECT NEXT NUMBER ADDRESS
13739  061116  062702  000002                  ADD     #2,R2           ;SELECT NEXT EXTENDED ADDRESS
```

```
13740  061122  000753                          BR      3$              ;CONTINUE
13741                                  ;ACCUMULATOR CONVERSION
13742  061124  072027  177776          1$:     ASH     #-2,R0          ;GET ACCUMULATOR NUMBER
13743  061130  042700  177477                  BIC     #177477,R0      ;
13744  061134  010002                          MOV     R0,R2           ;GENERATE
13745  061136  072227  177773                  ASH     #-5,R2          ;ADDRESS OF
13746  061142  062702  001462                  ADD     #$AC0,R2        ;EXTENDED EXPONENT
13747  061146  042767  000300  000004          BIC     #300,5$         ;GENERATE INSTRUCTION
13748  061154  050067  000000                  BIS     R0,5$           ;TO GET EXPONENT
13749  061160  175003                  5$:     STEXP   AC0,R3          ;GET EXPONENT
13750  061162  060312                          ADD     R3,(R2)         ;ADD TO EXTENDED EXPONENT
13751  061164  005003                          CLR     R3
13752  061166  042767  000300  000004          BIC     #300,4$         ;GENERATE INSTRUCTION
13753  061174  050067  000000                  BIS     R0,4$           ;TO LOAD EXPONENT BACK TO ACC
13754  061200  176403                  4$:     LDEXP   R3,AC0          ;LOAD EXPONENT OF 200
13755  061202  010546                  2$:     MOV     R5,-(SP)        ;RESTORE RETURN PC
13756  061204  000207                          RTS     PC              ;RETURN
13757                                  ;;**********************************************************************
  (1)
  (1)                                   .SBTTL  POWER DOWN AND UP ROUTINES
  (1)
  (1)                                   ;POWER DOWN ROUTINE
  (1)   061206  012737  061334  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
  (1)   061214  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
  (3)   061222  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
  (3)   061224  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
  (3)   061226  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
  (3)   061230  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
  (3)   061232  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
  (3)   061234  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
  (1)   061236  010667  000076                  MOV     SP,$SAVR6       ;;SAVE SP
  (1)   061242  012737  061254  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
  (1)   061250  000000                          HALT
  (1)   061252  000776                          BR      .-2             ;;HANG UP
  (1)
  (1)                                   ;POWER UP ROUTINE
  (1)   061254  016706  000060  $PWRUP: MOV     $SAVR6,SP       ;;GET SP
  (1)   061260  005067  000054                  CLR     $SAVR6          ;;WAIT LOOP FOR THE TTY
  (1)   061264  005267  000050          1$:     INC     $SAVR6          ;;WAIT FOR THE INC
  (1)   061270  001375                          BNE     1$              ;;OF  WORD
  (3)   061272  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
  (3)   061274  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
  (3)   061276  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
  (3)   061300  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
  (3)   061302  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
  (3)   061304  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
  (1)   061306  012737  061206  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
  (1)   061314  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
  (1)   061322  104400                          TYPE                    ;REPORT THE POWER FAILURE
  (1)   061324  061342  $PWRMG: .WORD   $POWER           ;;POWER FAIL MESSAGE POINTER
  (1)   061326  012716          MOV     (PC)+,(SP)      ;;RESTART AT START
  (1)   061330  003612  $PWRAD: .WORD   START           ;;RESTART ADDRESS
  (1)   061332  000002          RTI
  (1)   061334  000000  $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
  (1)   061336  000776          BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
  (1)   061340  000000  $SAVR6: 0                       ;;PUT THE SP HERE
```

```
  (1)  061342  005015  047520  042527  $POWER: .ASCIZ  <15><12>'POWER'
  (1)  061350  000122
  (1)                                          .EVEN
13758                                  ;;*****************************************************************
  (1)
  (1)                                  .SBTTL  TTY INPUT ROUTINE
  (1)
  (1)                                  ;*THIS ROUTINE WILL INPUT A SINGLE CHARACTER FROM THE TTY
  (1)                                  ;*CALL:
  (1)                                  ;*      RDCHR                   ;;INPUT A SINGLE CHARACTER FROM THE TTY
  (1)                                  ;*      RETURN HERE             ;;CHARACTER IS ON THE STACK
  (1)                                  ;*                              ;;WITH PARITY BIT STRIPPED OFF
  (1)                                  ;
  (1)
  (1)  061352  011646                  $RDCHR: MOV     (SP),-(SP)      ;;PUSH DOWN THE PC
  (1)  061354  016666  000004  000002          MOV     4(SP),2(SP)     ;;SAVE THE PS
  (1)  061362  105777  117722          1$:     TSTB    @$TKS           ;;WAIT FOR
  (1)  061366  100375                          BPL     1$              ;;A CHARACTER
  (1)  061370  117766  117716  000004        MOVB    @$TKB,4(SP)     ;;READ THE TTY
  (1)  061376  042766  177600  000004        BIC     #^C<177>,4(SP)  ;;GET RID OF JUNK IF ANY
  (1)  061404  000002                          RTI                     ;;GO BACK TO USER
  (2)                                  ;;*****************************************************************
  (1)                                  ;*THIS ROUTINE WILL INPUT A STRING FROM THE TTY
  (1)                                  ;*CALL:
  (1)                                  ;*      RDLIN                   ;;INPUT A STRING FROM THE TTY
  (1)                                  ;*      RETURN HERE             ;;ADDRESS OF FIRST CHARACTER WILL BE ON THE STACK
  (1)                                  ;*                              ;;TERMINATOR WILL BE A BYTE OF ALL 0'S
  (1)
  (1)  061406  010346                  $RDLIN: MOV     R3,-(SP)        ;;SAVE R3
  (1)  061410  012703  061514          1$:     MOV     #$TTYIN,R3      ;;GET ADDRESS
  (1)  061414  022703  061524          2$:     CMP     #$TTYIN+8.,R3   ;;BUFFER FULL?
  (1)  061420  101405                          BLOS    4$              ;;BR IF YES
  (1)  061422  104412                          RDCHR                   ;;GO READ ONE CHARACTER FROM THE TTY
  (1)  061424  112613                          MOVB    (SP)+,(R3)      ;;GET CHARACTER
  (1)  061426  122713  000177          10$:    CMPB    #177,(R3)       ;;IS IT A RUBOUT
  (1)  061432  001003                          BNE     3$              ;;SKIP IF NOT
  (1)  061434  104400  001406          4$:     TYPE    ,$QUES          ;;TYPE A '?'
  (1)  061440  000763                          BR      1$              ;;CLEAR THE BUFFER AND LOOP
  (1)  061442  111367  000044          3$:     MOVB    (R3),9$         ;;ECHO THE CHARACTER
  (1)  061446  104400  061512                  TYPE    ,9$
  (1)  061452  122723  000015                  CMPB    #15,(R3)+       ;;CHECK FOR RETURN
  (1)  061456  001356                          BNE     2$              ;;LOOP IF NOT RETURN
  (1)  061460  105063  177777                  CLRB    -1(R3)          ;;CLEAR RETURN (THE 15)
  (1)  061464  104400  001410                  TYPE    ,$LF            ;;TYPE A LINE FEED
  (1)  061470  012603                          MOV     (SP)+,R3        ;;RESTORE R3
  (1)  061472  011646                          MOV     (SP),-(SP)      ;;ADJUST THE STACK AND PUT ADDRESS OF THE
  (1)  061474  016666  000004  000002          MOV     4(SP),2(SP)     ;;      FIRST ASCII CHARACTER ON IT
  (1)  061502  012766  061514  000004          MOV     #$TTYIN,4(SP)
  (1)  061510  000002                          RTI                     ;;RETURN
  (1)  061512     000                  9$:     .BYTE   0               ;;STORAGE FOR ASCII CHAR. TO TYPE
  (1)  061513     000                          .BYTE   0               ;;TERMINATOR
  (1)  061514  000010                  $TTYIN: .BLKB   8.              ;;RESERVE 8 BYTES FOR TTY INPUT
13759                                  ;;*****************************************************************
  (1)
  (1)                                  .SBTTL  READ A DECIMAL NUMBER FROM THE TTY
  (1)
```

```
(1)                                     ;*THIS ROUTINE WILL READ A DECIMAL (ASCII) NUMBER FROM THE TTY AND
(1)                                     ;*CHANGE IT TO BINARY. IF TOO MANY CHARACTERS OR ANY ILLEGAL CHARACTERS
(1)                                     ;*ARE READ A "?" FOLLOWED BY A CARRIAGE RETURN-LINE FEED WILL BE TYPED.
(1)                                     ;*THE COMPLETE NUMBER MUST BE RETYPED. THE INPUT IS TERMINATED BY THE
(1)                                     ;*USER TYPING A CARRIAGE RETURN. THE RANGE OF THE INPUT NUMBER IS
(1)                                     ;*POSITIVE 32767 TO NEGATIVE 32768.
(1)                                     ;*CALL:
(1)                                     ;*      RDDEC                   ;;READ A DECIMAL NUMBER
(1)                                     ;*      RETURN HERE             ;;NUMBER IS ON TOP OF THE STACK
(1)                                     ;
(1)
(1)  061524  011646                     $RDDEC: MOV     (SP),-(SP)      ;;PROVIDE SPACE FOR
(1)  061526  016666  000004  000002             MOV     4(SP),2(SP)     ;;THE INPUT NUMBER
(3)  061534  010046                             MOV     R0,-(SP)        ;;PUSH R0 ON STACK
(3)  061536  010146                             MOV     R1,-(SP)        ;;PUSH R1 ON STACK
(3)  061540  010246                             MOV     R2,-(SP)        ;;PUSH R2 ON STACK
(1)  061542  104414                     1$:     RDLIN                   ;;READ AN ASCIZ LINE
(1)  061544  012600                             MOV     (SP)+,R0        ;;ADDRESS OF 1ST CHAR.
(1)  061546  010067  000120                     MOV     R0,6$           ;;SAVE INCASE OF BAD INPUT
(1)  061552  005046                             CLR     -(SP)           ;;CLEAR DATA WORD
(1)  061554  005002                             CLR     R2              ;;SIGN SET POSITIVE
(1)  061556  122710  000055                     CMPB    #'-,(R0)        ;;SEE IF A MINUS SIGN WAS TYPED
(1)  061562  001001                             BNE     2$              ;;BR IF NO MINUS SIGN
(1)  061564  112002                             MOVB    (R0)+,R2        ;;SAVE FOR LATER USE
(1)  061566  112001                     2$:     MOVB    (R0)+,R1        ;;PICKUP THIS CHARACTER
(1)  061570  001424                             BEQ     3$              ;;GET OUT IF ZERO
(1)  061572  122701  000060                     CMPB    #'0,R1          ;;MAKE SURE THIS CHARACTER
(1)  061576  003032                             BGT     5$              ;;IS A DIGIT BETWEEN 0 & 9
(1)  061600  122701  000071                     CMPB    #'9,R1
(1)  061604  002427                             BLT     5$
(1)  061606  032716  170000                     BIT     #^C7777,(SP)    ;;DON'T LET NUMBER GET TO BIG
(1)  061612  001024                             BNE     5$              ;;BR IF NUMBER WOULD OVERFLOW
(1)  061614  006316                             ASL     (SP)            ;;*2
(1)  061616  011646                             MOV     (SP),-(SP)      ;;SAVE FOR LATER
(1)  061620  006316                             ASL     (SP)            ;;*4
(1)  061622  006316                             ASL     (SP)            ;;*8.
(1)  061624  062616                             ADD     (SP)+,(SP)      ;;*10.
(1)  061626  102416                             BVS     5$              ;;OVERFLOW ISN'T ALLOWED
(1)  061630  162701  000060                     SUB     #'0,R1          ;;STRIP AWAY THE ASCII JUNK
(1)  061634  060116                             ADD     R1,(SP)         ;;ADD IN THIS DIGIT
(1)  061636  102412                             BVS     5$              ;;OVERFLOW ISN'T ALLOWED
(1)  061640  000752                             BR      2$              ;;LOOP
(1)  061642  005702                     3$:     TST     R2              ;;CHECK IF NUMBER IS NEG
(1)  061644  001401                             BEQ     4$              ;;BR IF NO
(1)  061646  005416                             NEG     (SP)            ;;YES--NEGATE THE NUMBER
(1)  061650  012666  000012             4$:     MOV     (SP)+,12(SP)    ;;SAVE THE RESULT
(3)  061654  012602                             MOV     (SP)+,R2        ;;POP STACK INTO R2
(3)  061656  012601                             MOV     (SP)+,R1        ;;POP STACK INTO R1
(3)  061660  012600                             MOV     (SP)+,R0        ;;POP STACK INTO R0
(1)  061662  000002                             RTI                     ;;RETURN
(1)
(1)  061664  005726                     5$:     TST     (SP)+           ;;CLEAN PARTIAL NUMBER FROM STACK
(1)  061666  105010                             CLRB    (R0)            ;;SET A TERMINATOR
(1)  061670  104400                             TYPE                    ;;TYPE THE INPUT UP TO BAD CHAR.
(1)  061672  000000                     6$:     .WORD   0               ;;POINTER GOES HERE
(1)  061674  104400  001406                     TYPE    ,$QUES          ;;"?" "CR" & "LF"
```

```
   (1)  061700  000720                          BR      1$                ;;TRY AGAIN
 13760
   (1)                                ;;*****************************************************************
   (1)
   (1)                                .SBTTL  ROUTINE TO SIZE MEMORY
   (1)
   (1)                                ;*CALL:
   (1)                                ;*      JSR     PC,$SIZE
   (1)                                ;*      RETURN
   (1)                                ;*$LSTAD WILL CONTAIN:
   (1)                                ;*      WITH KT11 OPTION        -- LAST VIRTUAL ADDRESS OF THE LAST BANK
   (1)                                ;*      WITHOUT KT11 OPTION     -- LAST ABSOLUTE ADDRESS OF AVAILABLE MEMORY
   (1)                                ;*$LSTBK WILL CONTAIN THE LAST BANK AS A SAF
   (1)                                ;*$KT11 IS THE MEMORY MANAGEMENT KEY
   (1)                                ;*BIT07 = 0 DON'T USE MEMORY MANAGEMENT
   (1)                                ;*      MUST BE SETUP BEFORE THE CALL
   (1)                                ;*BIT15 = 0 DON'T HAVE MEMORY MANAGEMENT OPTION
   (1)                                ;*      DETERMINED BY ROUTINE
   (1)                                ;*  --NOTE--
   (1)                                ;*THIS ROUTINE SUPPORTS PDP 11/74.
   (1)                                ;*IF ACTUAL MEMORY IS LESS THAN THAT INDICATED BY SIZE REGISTER
   (1)                                ;*AND A REFERENCE IS MADE TO A MEMORY ADDRESS THAT IS GREATER THAN
   (1)                                ;*ACTUAL MEMORY BUT LESS THAN SIZE REGISTER (INDICATED), THEN A
   (1)                                ;*MEMORY REFERENCE TIMEOUT TO VECTOR 114 WILL OCCUR.
   (1)
   (1)  061702  010046                $SIZE:  MOV     R0,-(SP)          ;;SAVE R0 ON THE STACK
   (1)  061704  010146                        MOV     R1,-(SP)          ;;SAVE R1 ON THE STACK
   (1)  061706  010246                        MOV     R2,-(SP)          ;;SAVE R2 ON THE STACK
   (1)  061710  010346                        MOV     R3,-(SP)          ;;SAVE R3 ON THE STACK
   (1)  061712  013746  000004                MOV     @#ERRVEC,-(SP)    ;;SAVE PRESENT ERROR VECTOR PS & PC
   (1)  061716  013746  000006                MOV     @#ERRVEC+2,-(SP)
   (1)  061722  013746  000114                MOV     @#114,-(SP)       ;;SAVE PRESENT PARITY VECTOR PS & PC
   (1)  061726  013746  000116                MOV     @#116,-(SP)
   (1)  061732  010600                        MOV     SP,R0             ;;SAVE THE STACK POINTER
   (1)  061734  013737  177776  000006        MOV     @#PS,@#ERRVEC+2   ;;SET ERRVEC PS TO PRESENT PS
   (1)  061742  012701  003776                MOV     #3776,R1          ;;SETUP ADDRESS
   (1)  061746  105727                        TSTB    (PC)+             ;;USE MEMORY MANAGEMENT?
   (1)  061750  000200                $KT11:  .WORD   200               ;;SET TO USE MEMORY MANAGEMENT
   (1)  061752  100065                        BPL     $CORE             ;;BR IF NO
   (1)  061754  012737  062120  000004        MOV     #$KTNEX,@#ERRVEC  ;;SET FOR TIMEOUT
   (1)  061762  005737  177572                TST     @#SR0             ;;KT11 ARE YOU THERE?
   (1)  061766  052767  100000  177754        BIS     #100000,$KT11     ;;YES--SET KT11 KEY
   (1)  061774  005046                        CLR     -(SP)             ;;INITIALIZE FOR 'PAR' LOADING
   (1)  061776  012702  172340                MOV     #KIPAR0,R2        ;;ADDRESS OF FIRST 'PAR'
   (1)  062002  012703  000010                MOV     #^D8,R3           ;;LOAD EIGHT 'PAR.'S' AND EIGHT 'PDR.'S'
   (1)  062006  012762  077406  177740  1$:   MOV     #77406,-40(R2)    ;;PDR = 4K, UP, READ/WRITE
   (1)  062014  011622                        MOV     (SP),(R2)+        ;;LOAD 'PAR'
   (1)  062016  062716  000200                ADD     #200,(SP)         ;;UPDATE FOR NEXT 'PAR'
   (1)  062022  077307                        SOB     R3,1$             ;;LOOP UNTIL ALL EIGHT ARE LOADED
   (1)  062024  012742  177600                MOV     #177600,-(R2)     ;;SETUP KIPAR7 FOR I/O
   (1)  062030  005042                        CLR     -(R2)             ;;SETUP KIPAR6 FOR TESTING
   (1)  062032  012737  062050  000004        MOV     #2$,@#ERRVEC      ;;CATCH TIMEOUT IF NO SR3
   (1)  062040  012737  000020  172516        MOV     #20,@#SR3         ;;ENABLE 22-BIT ADDRESSING
   (1)  062046  000401                        BR      3$                ;;THIS PDP-11 HAS A SR3 REG.
   (1)  062050  022626                2$:     CMP     (SP)+,(SP)+       ;;CLEAN OFF THE STACK--NO SR3.
   (1)  062052  005237  177572        3$:     INC     @#SR0             ;;TURN ON MEMORY MANAGEMENT
   (1)  062056  012737  062110  000004        MOV     #$KTOUT,@#ERRVEC  ;;SET FOR TIME OUT
```

```
(1)  062064  012737  062232  000114          MOV     #$MTMOUT,@#114  ;;SET FOR MEMORY REF TIMEOUT TO 114
(1)  062072  005737  143776          4$:     TST     @#143776        ;;TRAP ON NON-EX-MEM
(1)  062076  062712  000040                  ADD     #40,(R2)        ;;MAKE A 1K STEP
(1)  062102  023712  172356                  CMP     @#KIPAR7,(R2)   ;;LAST ONE?
(1)  062106  101371                          BHI     4$              ;;NO--TRY IT
(1)  062110  011202                  $KTOUT: MOV     (R2),R2         ;;GET LAST BANK+1
(1)  062112  005037  177572                  CLR     @#SR0           ;;TURN OFF MEMORY MANAGEMENT
(1)  062116  000421                          BR      $SIZEX
(1)  062120  042767  100000  177622  $KTNEX: BIC     #100000,$KT11   ;;KT11 NON-EXISTENT
(1)  062126  012737  062156  000004  $CORE:  MOV     #$CROUT,@#ERRVEC ;;SET FOR TIMEOUT
(1)  062134  005002                          CLR     R2              ;;SET UP BANK
(1)  062136  062701  004000          1$:     ADD     #4000,R1        ;;INCREMENT BY 1K
(1)  062142  062702  000040                  ADD     #40,R2          ;;1K STEP
(1)  062146  005711                          TST     (R1)            ;;TRAP ON TIME OUT
(1)  062150  022701  177776                  CMP     #177776,R1      ;;LAST ONE
(1)  062154  001370                          BNE     1$              ;;NO--TRY AGAIN
(1)  062156  162701  004000  $CROUT: SUB     #4000,R1
(1)  062162  162702  000040  $SIZEX: SUB     #40,R2          ;;DROP BACK
(1)  062166  010006                          MOV     R0,SP           ;;RESTORE THE STACK
(1)  062170  012637  000116                  MOV     (SP)+,@#116     ;;RESTORE PARITY VECTOR
(1)  062174  012637  000114                  MOV     (SP)+,@#114
(1)  062200  012637  000006                  MOV     (SP)+,@#ERRVEC+2 ;;RESTORE ERROR VECTOR
(1)  062204  012637  000004                  MOV     (SP)+,@#ERRVEC
(1)  062210  010167  000050                  MOV     R1,$LSTAD       ;;LAST ADDRESS
(1)  062214  010267  000046                  MOV     R2,$LSTBK       ;;LAST BANK
(1)  062220  012603                          MOV     (SP)+,R3        ;;RESTORE R3
(1)  062222  012602                          MOV     (SP)+,R2        ;;RESTORE R2
(1)  062224  012601                          MOV     (SP)+,R1        ;;RESTORE R1
(1)  062226  012600                          MOV     (SP)+,R0        ;;RESTORE R0
(1)  062230  000207                          RTS     PC
(1)  062232  032737  000001  177744  $MTMOUT: BIT     #BIT0,@#MEMERR  ;;MAKE SURE TRAP TO 114 IS
(1)  062240  001005                          BNE     1$              ;;DUE TO MEMORY REF TIMEOUT
(1)                                                                  ;;IF NOT, IS IT AN ABORT?
(1)  062242  032737  100000  177744          BIT     #BIT15,@#MEMERR ;;CPU ABORT?
(1)  062250  001001                          BNE     1$              ;;IF YES, EXIT
(1)  062252  000002                          RTI                     ;;IF NOT, CONTINUE
(1)  062254  012737  177777  177744  1$:     MOV     #-1,@#MEMERR    ;;CLEAR THE MEM ERROR REG
(1)  062262  000712                          BR      $KTOUT
(1)  062264  000000                  $LSTAD: .WORD   0               ;;CONTAINS THE LAST ADDRESS
(1)  062266  000000                  $LSTBK: .WORD   0               ;;CONTAINS THE LAST BANK
13761                                ;;**************************************************************************
(1)
(1)                                  .SBTTL  TRAP DECODER
(1)
(1)                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE ''TRAP'' INSTRUCTION
(1)                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
(1)                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
(1)                                  ;*GO TO THAT ROUTINE.
(1)
(1)  062270  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
(1)  062272  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
(1)  062276  005740                          TST     -(R0)           ;;BACKUP BY 2
(1)  062300  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
(1)  062302  016000  062310                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
(1)  062306  000200                          RTS     R0              ;;GO TO ROUTINE
(1)
```

```
   (3)
   (3)                                  .SBTTL  TRAP TABLE
   (3)
   (3)                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
   (3)                                  ;*BY THE ''TRAP'' INSTRUCTION.
   (3)
   (3)                                  ;       ROUTINE
   (3)                                  ;       -------
   (3)  062310                          $TRPAD:
   (3)  062310  056410                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
   (3)  062312  057242                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
   (3)  062314  057216                          $TYPOS  ;;CALL=TYPOS    TRAP+4(104404)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
   (3)  062316  057256                          $TYPON  ;;CALL=TYPON    TRAP+6(104406)  TYPE OCTAL NUMBER (AS PER LAST CALL)
   (3)  062320  057444                          $TYPDS  ;;CALL=TYPDS    TRAP+10(104410) TYPE DECIMAL NUMBER (WITH SIGN)
   (3)  062322  061352                          $RDCHR  ;;CALL=RDCHR    TRAP+12(104412) TTY TYPEIN CHARACTER ROUTINE
   (3)  062324  061406                          $RDLIN  ;;CALL=RDLIN    TRAP+14(104414) TTY TYPEIN STRING ROUTINE
   (3)  062326  061524                          $RDDEC  ;;CALL=RDDEC    TRAP+16(104416) READ A DECIMAL NUMBER FROM TTY
   (3)  062330  060026                          $SAVREG ;;CALL=SAVREG   TRAP+20(104420) SAVE R0-R5 ROUTINE
   (3)  062332  060064                          $RESREG ;;CALL=RESREG   TRAP+22(104422) RESTORE R0-R5 ROUTINE
13762   062334  060122                          $FL20   ;;CALL=FL20     TRAP+24(104424)
13763   062336  060410                          $FLD20  ;;CALL=FLD20    TRAP+26(104426)
13764                                ;;*****************************************************************
13765                                .SBTTL  UNIBUS EXERCISER INITIALIZATION ROUTINE
13766                                ;*THIS ROUTINE INITIALIZES THE BASE ADDRESS FOR THE
13767                                ;*UNIBUS EXERCISER AND LOADS UP THE EXERCISER REGISTERS.
13768                                ;;*****************************************************************
13769   062340  012767  002414  117424 UBEINIT:MOV     #$ERRTB,UBESAV          ;BASE ADDRESS OF UBE TRANSFER
13770   062346  005067  117422              CLR     UBESAV+2
13771   062352  012767  002414  117416        MOV     #$ERRTB,UBEADR          ;BASE ADDRESS OF UBE TRANSFER
13772   062360  005067  117414              CLR     UBEADR+2
13773
13774
13775                                ;SET UP THE UBE AND START IT
13776   062364  012702  002340              MOV     #UBETBL,R2      ;GET ADDRESS OF UBE TABLE
13777   062370  005072  000010              CLR     @10(R2)         ;CLEAR ALL ERRORS
13778   062374  012772  053434  000012        MOV     #UBESRV,@12(R2) ;SET UP UBE VECTOR
13779   062402  012772  000340  000014        MOV     #PR7,@14(R2)    ;SET UP UBE VECTOR PSW
13780   062410  012732  172400              MOV     #172400,@(R2)+  ;SET CC FOR 1.3K WORD TRANSFER
13781                                                               ;UBE IS DOING BYTE TRANSFERS
13782   062414  012746  000003              MOV     #3,-(SP)        ;PUT DEVICE ID IN STACK
13783   062420  012746  001776              MOV     #UBEADR,-(SP)   ;PUT ADDRESS OF PHYSICAL BA ON STACK
13784   062424  004737  063022              JSR     PC,@#GETMAP     ;GO GET MAP REGISTER
13785   062430  013732  001776              MOV     @#UBEADR,@(R2)+ ;LOAD UBE BUS ADDRESS
13786   062434  013732  002000              MOV     @#UBEADR+2,@(R2)+;LOAD ADR BITS 16 & 17
13787   062440  052737  000040  172516        BIS     #40,@#SR3       ;ENABLE MAP
13788   062446  000207                        RTS     PC              ;RETURN
13789                                ;;*****************************************************************
13790                                .SBTTL  CONVERT UNIBUS VIRTUAL ADDRESS TO PHYSICAL ADDRESS
13791                                ;*      THIS ROUTINE CONVERTS THE CONTENTS OF LOCATIONS
13792                                ;*      ''ERRBA'' AND ''ERRBA+2'' FROM A VIRTUAL 18-BIT ADDRESS
13793                                ;*      TO A PHYSICAL 22-BIT ADDRESS AS MAPPED BY THE APPROPRIATE
13794                                ;*      MAP REGISTER. THE 22-BIT ADDRESS IS STORED IN LOCATIONS
13795                                ;*      ''PA2116'' AND ''PA1500''.
13796                                ;;*****************************************************************
13797   062450  104420              PHYMAP: SAVREG
13798   062452  013703  002002              MOV     @#ERRBA,R3      ;GET BUS ADDRESS <15:00>
```

```
13799  062456  013702  002004                  MOV     @#ERRBA+2,R2    ;GET BUS ADDRESS <17:16>
13800  062462  042702  177774                  BIC     #177774,R2      ;
13801  062466  032737  000040  172516          BIT     #BIT5,@#MMR3    ;MAP ON?
13802  062474  001005                          BNE     1$              ;BRANCH IF YES
13803  062476  010337  001574                  MOV     R3,@#PA1500     ;PHY ADR=BUS ADR
13804  062502  010237  001576                  MOV     R2,@#PA2116
13805  062506  000421                          BR      MAPEND
13806  062510  010305                  1$:     MOV     R3,R5           ;SAVE ADR BITS <15:00>
13807  062512  073227  000005                  ASHC    #5,R2           ;GET MAP REG SELECT BITS
13808  062516  042702  000003                  BIC     #3,R2           ;
13809  062522  062702  170200                  ADD     #MAPL0,R2       ;FORM ADDRESS OF MAP REG
13810  062526  012237  001574                  MOV     (R2)+,@#PA1500  ;GET CONTENTS OF MAP REG LO
13811  062532  011237  001576                  MOV     (R2),@#PA2116   ;GET CONTENTS OF MAP REG HI
13812  062536  042705  160000                  BIC     #160000,R5      ;FORM PHYSICAL ADDRESS
13813  062542  060537  001574                  ADD     R5,@#PA1500     ;THAT TIMED OUT
13814  062546  005537  001576                  ADC     @#PA2116        ;
13815  062552  104422                  MAPEND: RESREG
13816  062554  000207                          RTS     PC
13817
13818                                  ;;*****************************************************************
13819                                  .SBTTL  CONVERT A VIRTUAL ADDRESS TO A PHYSICAL ADDRESS
13820                                  ;*      THIS ROUTINE CONVERTS A 16-BIT VIRTUAL ADDRESS TO A
13821                                  ;*      22-BIT PHYSICAL ADDRESS. THE VIRTUAL ADDRESS IS
13822                                  ;*      ASSUMED TO BE IN LOCATION 'VADR' AND THE PHYSICAL
13823                                  ;*      ADDRESS IS PLACED IN LOCATIONS 'PA2116' AND 'PA1500'.
13824                                  ;*
13825                                  ;*      IF MEMORY MANAGEMENT IS OFF THE PHYSICAL ADDRESS IS
13826                                  ;*      GENERATED BY SUBTRACTING THE CONTENTS OF LOCATION
13827                                  ;*      'FACTOR' FROM THE VIRTUAL ADDRESS. THIS LOCATION
13828                                  ;*      CONTAINS THE BYTE OFFSET BETWEEN THE RELOCATED CODE
13829                                  ;*      AND THE NON-RELOCATED CODE.
13830                                  ;*
13831                                  ;*      IF MEMORY MANAGEMENT IS ON, THE CONTENTS OF THE
13832                                  ;*      APPROPRIATE PAR REGISTER IS ADDED(AFTER ADJUSTMENT)
13833                                  ;*      TO THE LEAST SIGNIFICANT 13 BITS OF THE VIRTUAL ADDRESS.
13834                                  ;;*****************************************************************
13835  062556  104420                  CNVADR: SAVREG
13836  062560  013703  001572                  MOV     @#VADR,R3               ;GET VIRTUAL ADDRESS TO CONVERT
13837  062564  105737  001601                  TSTB    @#MMON                  ;IS MEMORY MGMT ON?
13838  062570  001426                          BEQ     1$                      ;BRANCH IF NO
13839  062572  005002                          CLR     R2
13840  062574  073227  000003                  ASHC    #3,R2                   ;GET PAR SELECT BITS
13841  062600  072327  177775                  ASH     #-3,R3                  ;RETURN VIR ADDR TO ORIGINAL
13842  062604  042703  160000                  BIC     #160000,R3              ;MAKE SURE SIGN DIDN'T EXTEND
13843  062610  006102                          ROL     R2                      ;MAKE R2 EVEN FOR WORD ADDRESSING
13844  062612  062702  172340                  ADD     #KIPAR0,R2              ;GET ADDRESS OF PAR
13845  062616  011205                          MOV     (R2),R5                 ;GET PAR DATA
13846  062620  005004                          CLR     R4                      ;SETUP R4
13847  062622  073427  000006                  ASHC    #6,R4                   ;SHIFT PAR DATA
13848  062626  060305                          ADD     R3,R5                   ;FORM PHYSICAL ADDRESS
13849  062630  005504                          ADC     R4
13850  062632  010437  001576          2$:     MOV     R4,@#PA2116             ;SAVE PHYSICAL
13851  062636  010537  001574                  MOV     R5,@#PA1500             ;ADDRESS
13852  062642  104422                          RESREG
13853  062644  000207                          RTS     PC                      ;RETURN
13854  062646  163703  001604          1$:     SUB     @#FACTOR,R3             ;FORM PHYSICAL ADDRESS
```

```
13855  062652  005004                        CLR     R4
13856  062654  010305                        MOV     R3,R5
13857  062656  000765                        BR      2$                      ;RETURN
13858
13859                                  ;;*******************************************************************
13860                                  .SBTTL  ROUTINE TO CHECK RELOCATED DATA
13861                                  ;*ROUTINE TO CHECK DATA RELOCATED
13862                                  ;*CALL: R0=     HIGHEST ADDRESS +2 OF SOURCE DATA
13863                                  ;*      R2=     HIGHEST ADDRESS +2 OF DEST DATA
13864                                  ;*      R5=     LOWEST ADDRESS OF THE SOURCE DATA
13865                                  ;*
13866                                  ;*      THIS ROUTINE USES A COMPARE INSTRUCTION TO CHECK
13867                                  ;*      THE DATA THAT WAS RELOCATED. IF A PARITY ERROR OCCURS
13868                                  ;*      DURING THIS CHECK A SPECIAL ERROR MESSAGE IS TYPED
13869                                  ;*      INSTEAD OF THE UNEXPECTED TRAP MESSAGE.
13870                                  ;;*******************************************************************
13871  062660  012703  004000         CHKDAT: MOV     #2048.,R3               ;COUNTER
13872  062664  012737  062756  000114         MOV     #2$,@#CACHVEC           ;SETUP PARITY VECTOR
13873  062672  024042                 7$:     CMP     -(R0),-(R2)             ;CHECK DATA
13874  062674  001026                         BNE     99$
13875  062676  005112                         COM     (R2)                    ;COMPLEMENT DEST DATA
13876  062700  005112                         COM     (R2)                    ;TWICE
13877  062702  021210                         CMP     (R2),(R0)               ;CHECK DATA
13878  062704  001022                         BNE     99$
13879  062706  020005                 1$:     CMP     R0,R5                   ;BRANCH IF ALL DATA CHECKED
13880  062710  001414                         BEQ     3$
13881  062712  105737  001601                 TSTB    @#MMON                  ;MEMORY MANAGMENT ON?
13882  062716  001765                         BEQ     7$                      ;BR BACK IF NOT
13883  062720  077314                         SOB     R3,7$                   ;REPEAT 4096 TIMES
13884  062722  012703  010000                 MOV     #4096.,R3               ;SET UP COUNTER AGAIN
13885  062726  162737  000200  172350         SUB     #200,@#KIPAR4           ;MAP TO NEXT LOWER 4K OF SPACE
13886  062734  012702  120000                 MOV     #120000,R2              ;START AT TOP OF 4K SPACE + 2
13887  062740  000754                         BR      7$                      ;CKECK MORE
13888  062742  012737  063616  000114  3$:     MOV     #.PARSRV,@#CACHVEC      ;RESTORE CACHVEC
13889  062750  000207                         RTS     PC                      ;RETURN
13890  062752  000262                 99$:    SEV
13891  062754  000207                         RTS     PC
13892  062756  013737  177744  001356  2$:     MOV     @#MEMERR,@#$TMP2                ;SAVE ERROR REG
13893  062764  013737  177740  001360         MOV     @#LOADRS,@#$TMP3                ;SAVE ERROR ADR
13894  062772  013737  177742  001362         MOV     @#HIADRS,@#$TMP4
13895  063000  010237  001572                 MOV     R2,@#VADR
13896  063004  010037  001352                 MOV     R0,@#$TMP0
13897  063010  104005                         ERROR   5
13898  063012  012737  177777  177744         MOV     #-1,@#MEMERR                    ;CLEAR ERROR REG
13899  063020  000754                         BR      99$                     ;RETURN
13900
13901                                  ;;*******************************************************************
13902                                  .SBTTL  ROUTINE TO GET A MAP REGISTER
13903                                  ;*THIS ROUTINE TAKES AN 18 BIT RANDOM NUMBER, FINDS TWO
13904                                  ;*CONSECUTIVE MAP REGISTERS THAT ARE NOT IN USE, LOADS THE
13905                                  ;*REGISTERS WITH THE PHYSICAL ADDRESS MINUS THE RANDOM NUMBER
13906                                  ;*AND THE NUMBER + 4K, AND RETURNS A NEW BUS ADDRESS, BASED
13907                                  ;*ON THE RANDOM NUMBER.
13908                                  ;*
13909                                  ;*      MAP REGISTERS 0 THRU 3 ARE NOT USED IF THE PROGRAM IS
13910                                  ;*      RUNNING ON ACT11. THIS ALLOWS 'MOTHER' TO ACCESS THE
```

```
13911                                   ;*      END OF PASS HOOKS.
13912                                   ;*
13913                                   ;*      THE MAP TABLE (MAPTBL) CONTAINS 4 BYTES, ONE FOR EACH
13914                                   ;*      UNIBUS DEVICE. IF THE UBE IS PRESENT IT USES THE
13915                                   ;*      4TH BYTE. WHEN A REGISTER IS ASSIGNED TO A DEVICE,
13916                                   ;*      THE LOWER 4 ADDRESS BITS OF THAT REGISTER ARE PLACED
13917                                   ;*      IN THE TABLE. WHEN A DEVICE REQUESTS A REGISTER PAIR
13918                                   ;*      THIS TABLE IS THEN SEARCHED TO SEE IF THE REGISTER
13919                                   ;*      PAIR IS IN USE.
13920                                   ;*      ENTER WITH:
13921                                   ;*              4(SP)=DEVICE ID
13922                                   ;*              2(SP)=ADDRESS OF THE PHYSICAL ADDRESS
13923                                   ;;*****************************************************************
13924  063022  016600  000004  GETMAP: MOV     4(SP),R0                ;GET DIVICE ID
13925  063026  016601  000002          MOV     2(SP),R1                ;GET ADR OF PHY ADR
13926  063032  013746  177776          MOV     @#PSW,-(SP)             ;SAVE CURRENT PRIORITY
13927  063036  005116                  COM     (SP)                    ;MAKE IT READY FOR RESTORE
13928  063040  042716  177437          BIC     #^CPR7,(SP)             ;
13929  063044  000237                  SPL     7                       ;IF THIS IS RK CALL, LOCK OUT UBE
13930  063046  104420                  SAVREG
13931  063050  012137  001276          MOV     (R1)+,@#$GDDAT          ;SAVE PHYSICAL
13932  063054  012137  001300          MOV     (R1)+,@#$BDDAT          ;ADDRESS
13933  063060  004737  060562  2$:     JSR     PC,@#$RAND              ;GET RANDOM NUMBER
13934  063064  013702  001622          MOV     @#$HINUM,R2             ;GET HIGH RANDOM NUMBER
13935  063070  013703  001620          MOV     @#$LONUM,R3             ;GET LOW RANDOM NUMBER
13936  063074  073227  177764          ASHC    #-14,R2                 ;CONVERT TO 20 BIT NUMBER
13937  063100  042702  177760          BIC     #177760,R2              ;GET RID OF 11 BITS OF SIGN EXT
13938  063104  022702  000016          CMP     #16,R2                  ;LEGAL MAP REG SELECT?
13939  063110  100001                  BPL     3$                      ;BRANCH IF YES
13940  063112  000762                  BR      2$                      ;TRY AGAIN
13941  063114  005737  001602  3$:     TST     @#QV                    ;ACT11 (QV OR AUTO)?
13942  063120  001406                  BEQ     4$                      ;BRANCH IF NO
13943  063122  122702  000000          CMPB    #0,R2                   ;MAP SELECT 0?
13944  063126  001754                  BEQ     2$                      ;BRANCH IF YES.(ACT MUST
13945                                                                  ;USE MAP 0 - 3)
13946  063130  122702  000001          CMPB    #1,R2
13947  063134  001751                  BEQ     2$
13948  063136  010204          4$:     MOV     R2,R4                   ;SAVE MAP SELECT BITS
13949  063140  042703  100000          BIC     #BIT15,R3               ;CLEAR SELECT BIT 0
13950  063144  073227  177776          ASHC    #-2,R2                  ;FORM 18 BIT ADDRESS
13951  063150  042703  000001          BIC     #BIT0,R3                ;MAKE SURE ITS EVEN
13952  063154  010241                  MOV     R2,-(R1)                ;RETURN NEW BUS ADDRESS
13953  063156  010341                  MOV     R3,-(R1)                ;TO THE APPROPRIATE HANDLER
13954  063160  012705  000004          MOV     #4,R5                   ;SET SOB COUNT
13955  063164  120465  001765  1$:     CMPB    R4,MAPTBL-1(R5)         ;IS THIS MAP IN USE?
13956  063170  001435                  BEQ     5$                      ;BRANCH IF YES
13957  063172  077504                  SOB     R5,1$                   ;CONTINUE
13958  063174  110460  001766          MOVB    R4,MAPTBL(R0)           ;PUT MAP SELECT BITS IN TABLE
13959  063200  072427  000003          ASH     #3,R4                   ;FORM INDEX TO GET MAP REG ADDR
13960  063204  062704  170200          ADD     #MAPL0,R4               ;GENERATE MAP ADDRESS
13961  063210  042703  160000          BIC     #160000,R3              ;GET LS 13 BITS OF RAND NO.
13962  063214  013701  001276          MOV     @#$GDDAT,R1             ;GET PHYSICAL
13963  063220  013702  001300          MOV     @#$BDDAT,R2             ;ADDRESS
13964  063224  160301                  SUB     R3,R1                   ;GENERATE MAP
13965  063226  005602                  SBC     R2                      ;REGISTER DATA
13966  063230  010124                  MOV     R1,(R4)+                ;LOAD THE
```

```
13967  063232  010224                        MOV     R2,(R4)+                ;FIRST MAP REGISTER
13968  063234  062701  020000                ADD     #20000,R1               ;ADD 4K
13969  063240  005502                        ADC     R2                      ;TO MAP DATA
13970  063242  010124                        MOV     R1,(R4)+                ;LOAD THE
13971  063244  010224                        MOV     R2,(R4)+                ;SECOND MAP REGISTER
13972  063246  104422                        RESREG
13973  063250  042637  177776                BIC     (SP)+,@#PSW             ;RETURN PRIORITY TO ORIGINAL VALUE
13974  063254  011666  000004                MOV     (SP),4(SP)              ;SETUP RETURN PC
13975  063260  022626                        CMP     (SP)+,(SP)+             ;CLEAN UP THE STACK
13976  063262  000207                        RTS     PC                      ;RETURN
13977                                 ;REGISTER PAIR IS IN USE, TRY ANOTHER RANDOM NUMBER
13978  063264  062701  000004          5$:   ADD     #4,R1                   ;RESTORE R1
13979  063270  000137  063060                JMP     @#2$                    ;GET ANOTHER RANDOM NUMBER
13980                                 ;;*****************************************************************
13981                                 .SBTTL  GIVE MAP SUBROUTINE
13982                                 ;*      THIS ROUTINE TAKES THE MAP ADDRESS OUT OF THE MAP TABLE
13983                                 ;*      FOR THE REQUESTING DEVICE AND REPLACES IT WITH 377.
13984                                 ;;*****************************************************************
13985  063274  010046                GIVEMAP:MOV     R0,-(SP)                ;SAVE R0
13986  063276  016600  000004                MOV     4(SP),R0                ;GET DEVICE ID
13987  063302  112760  000377  001766        MOVB    #377,MAPTBL(R0)         ;TAKE IT OUT OF THE TABLE
13988  063310  012600                        MOV     (SP)+,R0                ;RESTORE R0
13989  063312  000207                        RTS     PC                      ;RETURN
13990
13991                                 ;;*****************************************************************
13992                                 .SBTTL  ROUTINE TO CLEAR 'T' BIT
13993                                 ;;*****************************************************************
13994  063314  013746  177776        CLRTBIT:MOV     @#PSW,-(SP)             ;PUSH PSW ONTO STACK
13995  063320  011627                        MOV     (SP),(PC)+              ;SAVE IN RETPSW BELOW
13996  063322  000000                RETPSW: .WORD   0
13997  063324  042716  000020                BIC     #20,(SP)                ;CLEAR T BIT IN PSW ON STACK
13998                                 ;;*****************************************************************
13999                                 .SBTTL  ROUTINE TO RESTORE THE T BIT
14000                                 ;;*****************************************************************
14001  063330  012746  063336        RESPSW: MOV     #1$,-(SP)               ;SET RETURN PC FOR RTI
14002  063334  000002                        RTI                             ;CLEAR 'T' BIT IN PSW
14003  063336  000207                1$:     RTS     PC                      ;RETURN
14004
14005  063340  042737  177400  177776  RESTPS: BIC     #177400,@#PSW           ;SET KERNEL MODE
14006  063346  016746  177750                MOV     RETPSW,-(SP)            ;PUSH ORIG PSW ONTO STACK
14007  063352  000766                        BR      RESPSW
14008
14009                                 ;;*****************************************************************
14010                                 .SBTTL  KEYBOARD INT SERV ROUTINE
14011                                 ;*THIS ROUTINE HANDLES INTERRUPTS FROM THE KEYBOARD
14012                                 ;*
14013                                 ;*TYPING A CONTROL 'C' WILL CAUSE THE PROCESSOR TO  HALT
14014                                 ;*
14015                                 ;*TYPING A CARRAGE RETURN WILL CAUSE A CARRIAGE RETURN-LINE FEED
14016                                 ;*TO BE TYPED.
14017                                 ;*
14018                                 ;*TYPING A CONTROL 'O' WILL INHIBIT ANY FURTHER TYPEOUT. THE SECOND CONTROL 'O'
14019                                 ;*WILL ENABLE TYPEOUT AGAIN AND ECHO A CR-LF.
14020                                 ;*
14021                                 ;*ANY OTHER CHARACTER WILL JUST BE ECHOED.
14022                                 ;;*****************************************************************
```

```
14023
14024           000003                          CNTRLC=3
14025           000017                          CNTRLO=17
14026
14027  063354   017746  115732          TKISR:  MOV     @$TKB,-(SP)             ;GET CHARACTER
14028  063360   042716  177600                  BIC     #177600,(SP)            ;STRIP UNUSED BITS
14029  063364   022716  000003                  CMP     #CNTRLC,(SP)            ;BRANCH IF NOT CONTROL C (^C)
14030  063370   001010                          BNE     1$
14031  063372   012737  001406  001340          MOV     #$CRLF-1,@#$REG5        ;ECHO CR LF
14032  063400   106277  115710                  ASRB    @$TPS
14033  063404   005726                          TST     (SP)+                   ;POP CHARACTER OFF THE STACK
14034  063406   000000                          HALT
14035  063410   000002                          RTI                             ;RETURN
14036
14037  063412   122716  000015          1$:     CMPB    #15,(SP)                ;BRANCH IF NOT <CR>
14038  063416   001007                          BNE     2$
14039  063420   012737  001406  001340          MOV     #$CRLF-1,@#$REG5        ;ECHO CR LF
14040  063426   106277  115662                  ASRB    @$TPS
14041  063432   005726                          TST     (SP)+                   ;POP CHARACTEROFF STACK
14042  063434   000002                          RTI                             ;RETURN
14043
14044  063436   122716  000017          2$:     CMPB    #CNTRLO,(SP)            ;BRANCH IF NOT CONTROL O (^O)
14045  063442   001012                          BNE     3$
14046  063444   005726                          TST     (SP)+
14047  063446   005167  116074                  COM     NOTYPE
14048  063452   100405                          BMI     7$
14049  063454   012737  001406  001340          MOV     #$CRLF-1,@#$REG5        ;ECHO CR LF
14050  063462   106277  115626                  ASRB    @$TPS
14051  063466   000002                  7$:     RTI
14052
14053  063470   104420                  3$:     SAVREG
14054  063472   011605                          MOV     (SP),R5         ;RETRIEVE CHARACTER
14055  063474   004737  064104                  JSR     PC,@#LDKT               ;GO TO LOW CORE
14056  063500   013700  001522                  MOV     @#TKBFRP,R0             ;GET BUFFER PTR
14057  063504   110520                  4$:     MOVB    R5,(R0)+                ;LOAD CHAR INTO BFR
14058  063506   105010                          CLRB    (R0)                    ;CLEAR NEXT LOC
14059  063510   022700  001544          5$:     CMP     #TKBFR+20,R0            ;BRANCH IF NOT END OF BFR
14060  063514   001002                          BNE     6$
14061  063516   012700  001524                  MOV     #TKBFR,R0               ;RESET BUFFER PTR
14062  063522   010037  001522          6$:     MOV     R0,@#TKBFRP             ;RESTORE BFR PTR
14063  063526   004737  064202                  JSR     PC,@#RESKT              ;GO BACK TO ORIGINAL MEMORY
14064  063532   104422                          RESREG
14065  063534   005737  001546          ECHO:   TST     @#NOTYPE                ;TYPEOUT DISABLED?
14066  063540   100004                          BPL     1$                      ;BRANCH IF NO
14067  063542   005726                          TST     (SP)+                   ;FIX UP STACK
14068  063544   105077  115544                  CLRB    @$TPS                   ;CLEAR IE BIT
14069  063550   000002                          RTI                             ;RETURN
14070  063552   105777  115536          1$:     TSTB    @$TPS           ;PRINTER READY?
14071  063556   100375                          BPL     .-4                     ;BRANCH IF NO
14072  063560   112677  115532                  MOVB    (SP)+,@$TPB             ;MOVE CHAR TO PRINTER
14073  063564   000002                          RTI                             ;RETURN
14074
14075
14076                                  ;;*******************************************************************
14077                                  .SBTTL  TELETYPE INTERRUPT SERVICE ROUTINE
14078                                  ;*THIS ROUTINE TYPES A MESSAGE POINTED TO BY THE ADR STORED
```

```
14079                                   ;*IN LOCATION $REG5. THIS ROUTINE IS INTERRUPT DRIVEN.
14080                                   ;;*****************************************************************
14081  063566  005237  001340          TPISR:  INC     @#$REG5         ;STEP MESSAGE ADDRESS PTR
14082  063572  117746  115542                  MOVB    @$REG5,-(SP)            ;GET CHAR TO BE TYPED
14083  063576  001356                          BNE     ECHO                    ;GO TYPE CHAR IF NOT '0'
14084  063600  005726                          TST     (SP)+                   ;POP STACK
14085  063602  005077  115506                  CLR     @$TPS                   ;CLEAR IE BIT
14086  063606  012737  001624  001340          MOV     #NULLS,@#$REG5
14087  063614  000002                          RTI                             ,RETURN
14088
14089                                   ;;*****************************************************************
14090                                   .SBTTL  PARITY ERROR SERVICE
14091                                   ;*      THIS ROUTINE FIELDS UNEXPECTED TRAPS TO 114. IT IS ASSUMED
14092                                   ;*      THAT THE ERROR WAS IN CACHE AND WAS CAUSED BY THE 'OTHER
14093                                   ;*      WORD'' RATHER THAN THE 'WANTED WORD'' WHICH MEANS THAT THE
14094                                   ;*      BAD DATA IS STILL IN THE CACHE. SO, TO CLEAR THE BAD DATA
14095                                   ;*      THE ERROR ADDRESS IS REFERENCED CAUSING THE CACHE TO GO
14096                                   ;*      TO MAIN MEMORY TO GET THE DATA. THIS PREVENTS AN
14097                                   ;*      ARBITRARY REFERENCE TO THE BAD WORD FROM TRAPPING.
14098                                   ;*
14099                                   ;*      AFTER THE ERROR IS REPORTED, BITS 2 AND 3 OF THE MEMORY
14100                                   ;*      ERROR REGISTER ARE TESTED TO SEE IF THE BAD DATA IS IN
14101                                   ;*      MAIN MEMORY. IF IT IS, THE PROGRAM RESTARTS SINCE THE
14102                                   ;*      GOOD DATA IS NOW LOST FOREVER. OTHERWISE THE PROGRAM
14103                                   ;*      RETURNS TO THE ADDRESS POINTED TO BY '$LPERR''.
14104                                   ;;*****************************************************************
14105  063616  012737  064076  000114  .PARSRV:MOV     #RT1,@#CACHVEC  ;PUT NEW ADDRESS IN PARITY VECTOR
14106  063624  016637  000002  001352          MOV     2(SP),@#$TMP0   ;SAVER ERROR PSW
14107  063632  011637  001572                  MOV     (SP),@#VADR     ;SAVE PC
14108  063636  162737  000002  001572          SUB     #2,@#VADR       ;ADJUST ERROR PC
14109  063644  013702  177744                  MOV     @#MEMERR,R2     ;GET ERROR REGISTER
14110  063650  013703  177740                  MOV     @#LOADRS,R3     ;GET LO ADDRESS ERROR REG
14111  063654  010337  001360                  MOV     R3,@#$TMP3      ;PUT LOW ADR IN MEMORY
14112  063660  013737  177742  001362          MOV     @#HIADRS,@#$TMP4        ;GET HI ADDRESS ERROR REG
14113  063666  042703  176000                  BIC     #176000,R3      ;MASK OFF LOWER TEN BITS
14114  063672  013704  172354                  MOV     @#KIPAR6,R4     ;SAVE PAR6
14115  063676  105737  001601                  TSTB    @#MMON          ;IS MEMORY MGMT ON?
14116  063702  001407                          BEQ     1$              ;BRANCH IF NO
14117  063704  005037  172354                  CLR     @#KIPAR6        ;CLEAR PAR6
14118  063710  012737  077406  172314          MOV     #77406,@#KIPDR6 ;ENSURE PDR 6 RESIDENT
14119  063716  052703  140000                  BIS     #140000,R3      ;SETUP R3 TO REFERENCE THRU PAR6
14120  063722  105713                  1$:     TSTB    (R3)            ;REFERENCE ADDRESS THAT TRAPPED
14121                                                                  ;SHOULD CAUSE ABORT
14122  063724  005102                  2$:     COM     R2              ;GET ORIGINAL MEMORY
14123  063726  010237  177744                  MOV     R2,@#MEMERR     ;ERROR REG DATA
14124  063732  013737  177744  001356  PERET:  MOV     @#MEMERR,@#$TMP2        ;SAVE ERROR REG FOR TYPEOUT
14125  063740  013737  001262  001336          MOV     @#$LPERR,@#$REG4        ;SAVE LOOP ADDRESS
14126  063746  012737  063756  001262          MOV     #2$,@#$LPERR            ;SET RETURN ADDRESS IF LOOPING
14127  063754  104004                          ERROR   4
14128  063756  013737  001336  001262  2$:     MOV     @#$REG4,@#$LPERR        ;RESTORE LOOP ADDRESS
14129  063764  010437  172354                  MOV     R4,@#KIPAR6     ;RESTORE PAR6
14130  063770  013704  177744                  MOV     @#MEMERR,R4     ;GET MEM ERR REG
14131  063774  012737  177777  177744          MOV     #-1,@#MEMERR            ;CLEAR ERR REG
14132  064002  012737  063616  000114          MOV     #.PARSRV,@#CACHVEC      ;RESTORE PARITY VECTOR
14133  064010  042704  177763                  BIC     #177763,R4      ;CLEAR ALL BUT BITS 2 & 3
14134  064014  001426                          BEQ     1$              ;BRANCH IF NOT MAIN MEMORY ERROR
```

```
14135  064016  104400  064024                  TYPE    ,65$            ;;TYPE ASCIZ STRING
  (1)  064022  000420                          BR      64$             ;;GET OVER THE ASCIZ
  (1)                                  ;;65$:  .ASCIZ  /FATAL PARITY ERROR-RESTARTING/<CRLF>
  (1)  064064                          64$:
14136  064064  000005                          RESET                   ;CLEAR THE WORLD
14137  064066  000137  003612                  JMP     @#START
14138  064072  012716  064100          1$:     MOV     #X,(SP)         ;PUT ADDRESS ON STACK TO GET ORIGINAL
14139                                                                  ;PSW BACK
14140  064076  000002                  RT1:    RTI                     ;GET OLD PSW
14141  064100  000177  115156          X:      JMP     @$LPERR         ;JUMP TO START OF TEST THAT HAD THE PE
14142                                  ;;*****************************************************************
14143                                  .SBTTL  CONTEXT SWITCH DOWN SUBROUTINE
14144                                  ;*      SUBROUTINE TO SAVE & LOAD KIPAR'S 0,1,2 AND 3 (IF MEM MGMT ENABLED)
14145                                  ;*      THIS ROUTINE IS CALLED BY THE KEYBOARD INTERRUPT, LINE CLOCK
14146                                  ;*      INTERRUPT, UBE SERVICE ROUTINE, MBT SERVICE ROUTINE, AND TYPE TIME ROUTINE.
14147                                  ;;*****************************************************************
14148  064104  105737  001601          LDKT:   TSTB    @#MMON                  ;BRANCH IF MEM MGMT DISABLED
14149  064110  001433                          BEQ     1$
14150  064112  012604                          MOV     (SP)+,R4        ;SAVE RETURN PC
14151  064114  013737  177776  001566          MOV     @#PSW,@#$SAVPSW ;SAVE THE CURRENT PSW
14152  064122  042737  140000  177776          BIC     #140000,@#PSW   ;GO TO KERNEL MODE
14153  064130  012700  172340                  MOV     #KIPAR0,R0      ;GET ADDRESS OF PAR0
14154  064134  012001                          MOV     (R0)+,R1        ;GET PAR0
14155  064136  012002                          MOV     (R0)+,R2        ;GET PAR1
14156  064140  012003                          MOV     (R0)+,R3        ;GET PAR2
14157  064142  012005                          MOV     (R0)+,R5        ;GET PAR3
14158  064144  012740  000600                  MOV     #600,-(R0)      ;BACK TO LOW CORE
14159  064150  012740  000400                  MOV     #400,-(R0)      ;RELOC BACK TO LOW CORE
14160  064154  012740  000200                  MOV     #200,-(R0)
14161  064160  005040                          CLR     -(R0)
14162  064162  012700  001556                  MOV     #$SAVPAR,R0     ;GET ADDRESS OF SAVE BUFFER
14163  064166  010120                          MOV     R1,(R0)+        ;PUT PAR DATA IN MEMORY
14164  064170  010220                          MOV     R2,(R0)+
14165  064172  010320                          MOV     R3,(R0)+
14166  064174  010510                          MOV     R5,(R0)
14167  064176  010446                          MOV     R4,-(SP)        ;PUT RETURN PC ON STACK
14168  064200  000207                  1$:     RTS     PC
14169
14170                                  ;;*****************************************************************
14171                                  .SBTTL  CONTEXT SWITCH UP SUBROUTINE
14172                                  ;SUBROUTINE TO RESTORE KIPAR0, 1,2 AND 3 (IF MGMT ENABLED)
14173                                  ;;*****************************************************************
14174  064202  105737  001601          RESKT:  TSTB    @#MMON          ;BRANCH IF MEM MGMT DISABLED
14175  064206  001421                          BEQ     1$
14176  064210  012604                          MOV     (SP)+,R4        ;GET RETURN PC
14177  064212  012700  001556                  MOV     #$SAVPAR,R0     ;GET ADDRESS OF SAVE BUFF
14178  064216  012001                          MOV     (R0)+,R1        ;GET OLD PAR DATA
14179  064220  012002                          MOV     (R0)+,R2
14180  064222  012003                          MOV     (R0)+,R3
14181  064224  012005                          MOV     (R0)+,R5
14182  064226  012700  172340                  MOV     #KIPAR0,R0      ;GET ADDRESS OF PAR0
14183  064232  010120                          MOV     R1,(R0)+        ;RELOCATE BACK
14184  064234  010220                          MOV     R2,(R0)+
14185  064236  010320                          MOV     R3,(R0)+
14186  064240  010510                          MOV     R5,(R0)
14187  064242  013737  001566  177776          MOV     @#$SAVPSW,@#PSW
```

```
14188  064250  010446                          MOV     R4,-(SP)
14189  064252  000207                  1$:     RTS     PC
14190                                  ;;*****************************************************************
14191                                  .SBTTL  KT ABORT SUBROUTINE
14192                                  ;;*****************************************************************
14193  064254  016637  000002  001352  KTABRT: MOV     2(SP),@#$TMP0           ;SAVE ERROR PSW
14194  064262  011637  001572                  MOV     (SP),@#VADR             ;SAVE ERROR PC
14195  064266  162737  000002  001572          SUB     #2,@#VADR
14196  064274  013737  177572  001356          MOV     @#MMR0,@#$TMP2          ;SAVE MMR0
14197  064302  013737  177576  001360          MOV     @#MMR2,@#$TMP3          ;SAVE MMR2
14198  064310  013737  001262  001336          MOV     @#$LPERR,@#$REG4        ;SAVE LOOP ADDRESS
14199  064316  012737  064326  001262          MOV     #1$,@#$LPERR            ;SET RETURN ADR IF LOOPING
14200  064324  104003                          ERROR   3
14201  064326  013737  001336  001262  1$:     MOV     @#$REG4,@#$LPERR        ;RESTORE LOOP ADR
14202  064334  042737  170000  177572          BIC     #170000,@#MMR0          ;CLEAR ERRORS
14203  064342  013716  001262                  MOV     @#$LPERR,(SP)           ;GET LOOP ADDRESS
14204  064346  000002                          RTI                             ;RETURN
14205
14206                                  ;;*****************************************************************
14207                                  .SBTTL  RESERVED INSTRUCTION ROUTINE
14208                                  ;;*****************************************************************
14209  064350  016637  000002  001352  RESERR: MOV     2(SP),@#$TMP0           ;SAVE PSW
14210  064356  011637  001572                  MOV     (SP),@#VADR             ;SAVE ERROR PC
14211  064362  162737  000002  001572          SUB     #2,@#VADR
14212  064370  013737  001262  001336          MOV     @#$LPERR,@#$REG4        ;SAVE LOOP ADR
14213  064376  012737  064406  001262          MOV     #1$,@#$LPERR            ;SET RETURN ADR IF LOOPING
14214  064404  104002                          ERROR   2
14215  064406  013737  001336  001262  1$:     MOV     @#$REG4,@#$LPERR        ;RESTORE LOOP ADR
14216  064414  013716  001262                  MOV     @#$LPERR,(SP)           ;GET LOOP ADDRESS
14217  064420  000002                          RTI                             ;RETURN
14218
14219                                  ;;*****************************************************************
14220                                  .SBTTL  TRAP TO 4 SERVICE ROUTINE
14221                                  ;;*****************************************************************
14222  064422  016637  000002  001352  ERPRT:  MOV     2(SP),@#$TMP0           ;SAVE ERROR PSW
14223
14224  064430  011637  001572                  MOV     (SP),@#VADR             ;SAVE ERROR PC
14225  064434  162737  000002  001572          SUB     #2,@#VADR
14226  064442  012706  000700                  MOV     #SUPSTK,SP              ;RESTORE SP
14227  064446  013737  177766  001356          MOV     @#CPUERR,@#$TMP2        ;GET ERROR REG
14228  064454  013737  001262  001336          MOV     @#$LPERR,@#$REG4        ;SAVE LOOP ADR
14229  064462  012737  064472  001262          MOV     #1$,@#$LPERR            ;SET RETURN ADR IF LOOPING
14230  064470  104001                          ERROR   1
14231  064472  013737  001336  001262  1$:     MOV     @#$REG4,@#$LPERR        ;SET LOOP ADR
14232  064500  005037  177766                  CLR     @#CPUERR
14233  064504  013746  001352                  MOV     @#$TMP0,-(SP)   ;SETUP STACK TO RETURN
14234  064510  013746  001262                  MOV     @#$LPERR,-(SP)
14235  064514  000002                          RTI                             ;RETURN
14236
```

```
14238                                   ;THE BELOW TABLE REPRESENTS THE 'NEW' PSW SET BY THE PROGRAM ON
14239                                   ;SUCCESSIVE SUB-PASSES.
14240                                   ;NOTE THE BELOW TABLE MAY BE MODIFIED TO CAUSE THE PROGRAM TO RUN
14241                                   ;UNDER USER DEFINED PARAMETERS BY PATCHING IN THE DESIRED PASS PARAMETER
14242                                   ;FOR EXAMPLE TO CAUSE THE PROGRAM TO RUN WITHOUT SETTING THE 'T' BIT
14243                                   ;IN ALL PASSES PATCH OUT THE 'T' BIT IN THE TABLE.
14244  064516  000000                   PSWTAB: 000000                  ;
14245  064520  000020                           000020                  ;T-BIT TRAPPING
14246  064522  140000                           140000                  ;USER MODE
14247  064524  144020                           144020                  ;USER MODE, REG SET #1, T-BIT TRAPPING
14248  064526  040000                           040000                  ;SUPERVISOR MODE
14249  064530  044020                           044020                  ;SUPERVISOR MODE, REG SET #1, T-BIT TRAPPING
14250
14251                                   ;THE BELOW TABLE IS USED TO SET MEMORY MARGINS
14252  064532  000000                   MRGTAB: .WORD   0                       ;NO MARGINS
14253  064534  000004                           .WORD   4                       ;EARLY STROBE
14254  064536  000006                           .WORD   6                       ;LATE STROBE
14255  064540  000010                           .WORD   10                      ;LOW DRIVE CURRENT
14256  064542  000000                           .WORD   0                       ;NO MARGINS
14257  064544  000012                           .WORD   12                      ;HIGH DRIVE CURRENT
14258                                   ;MESSAGES
14259                                           .EVEN
14260  064546  002210                   REGINX: RP3DS
14261  064550  002232                           RKDS
14262  064552  000000                           .WORD
14263  064554  000000                           .WORD
14264  064556  002252                           RP4CS1
14265  064560  002312                           RSCS1
14266  064562  000000                           .WORD
14267  064564  000000                           .WORD
14268  064566  002356                           MBTTBL
14269  064570  002340                           UBETBL
14270  064572  064731                   MSGINX: .WORD   MSG5
14271  064574  064737                           .WORD   MSG6
14272  064576  065546                           .WORD   MSG21
14273  064600  065546                           .WORD   MSG21
14274  064602  064745                           .WORD   MSG10
14275  064604  064753                           .WORD   MSG11
14276  064606  065546                           .WORD   MSG21
14277  064610  065546                           .WORD   MSG21
14278  064612  065242                           .WORD   MSG15
14279  064614  065605                           .WORD   MSG24
14280  064616  046200  053517  046040   MSG1:   .ASCIZ  <CRLF>'LOW LIM?'
       064624  046511  000077
14281  064630  044510  044107  046040   MSG2:   .ASCIZ  'HIGH LIM?'
       064636  046511  000077
14282  064642  051105  047522  050122   MSG3:   .ASCIZ  /ERRORPC PHYSC PC    PSW   MAINT   TEST NO SUB-PASS CNT/
       064650  020103  044120  051531
       064656  020103  041520  020040
       064664  020040  051520  020127
       064672  020040  040515  047111
       064700  020124  020040  042524
       064706  052123  047040  020117
       064714  052523  026502  040520
       064722  051523  041440  052116
       064730     000
```

```
14283                                   ;MSG4 HAS BEEN MOVED TO END OF PROGRAM
14284
14285  064731     122  030120  004463  MSG5:   .ASCIZ  ?RP03   ?
       064736     000
14286  064737     122  030113  004465  MSG6:   .ASCIZ  ?RK05   ?
       064744     000
14287  064745     122  030120  004464  MSG10:  .ASCIZ  ?RP04   ?
       064752     000
14288  064753     122  030123  004464  MSG11:  .ASCIZ  ?RS04   ?
       064760     000
14289  064761     104  053122  052123  MSG12:  .ASCIZ  /DRVSTA  ERRREG  CSREG   WRDCNT  BUSADR  DSKADR  CYLADR(RP03)  PHYS BUSA
       064766  020101  042440  051122
       064774  042522  020107  041440
       065002  051123  043505  020040
       065010  053440  042122  047103
       065016  020124  041040  051525
       065024  042101  020122  042040
       065032  045523  042101  020122
       065040  041440  046131  042101
       065046  024122  050122  031460
       065054  020051  050040  054510
       065062  020123  052502  040523
       065070  051104  000200
14290  065074  041440  030523  020040  MSG13:  .ASCIZ  / CS1    WRDCNT  BUSADR  BADREX  DSKADR   CS2     CS3    DRVSTA  ERRREG/
       065102  020040  051127  041504
       065110  052116  020040  052502
       065116  040523  051104  020040
       065124  040502  051104  054105
       065132  020040  051504  040513
       065140  051104  020040  041440
       065146  031123  020040  020040
       065154  041440  031523  020040
       065162  020040  051104  051526
       065170  040524  020040  051105
       065176  051122  043505  000200
14291  065204  042504  041523  046131  MSG14:  .ASCIZ  /DESCYL   ER2     ER3    RPCC/<CRLF>
       065212  020040  042440  031122
       065220  020040  020040  042440
       065226  031522  020040  020040
       065234  050122  041503  000200
14292  065242  040515  051523  041040  MSG15:  .ASCIZ  /MASS BUS TESTER /
       065250  051525  052040  051505
       065256  042524  020122     000
14293  065263     040  051503  020061  MSG16:  .ASCIZ  / CS1    WRDCNT  BUSADR  BADREX    MR2     CS2      ST      ER      CS3/<
       065270  020040  053440  042122
       065276  047103  020124  041040
       065304  051525  042101  020122
       065312  041040  042101  042522
       065320  020130  020040  046440
       065326  031122  020040  020040
       065334  041440  031123  020040
       065342  020040  020040  052123
       065350  020040  020040  042440
       065356  020122  020040  020040
       065364  041440  031523  000200
14294  065372  020040  041503  020040  MSG17:  .ASCIZ  /  CC    BUSADR    CR2     CR1  PHYS BUSADR/<CRLF>
```

```
                                                                G 2
                065400  020040  052502  040523
                065406  051104  020040  020040
                065414  051103  020062  020040
                065422  020040  051103  020061
                065430  050040  054510  020123
                065436  052502  040523  051104
                065444  000200
14295           065446  044124  020105  052521  MSG20:  .ASCIZ  /THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789/<15><12>
                065454  041511  020113  051102
                065462  053517  020116  047506
                065470  020130  052512  050115
                065476  042105  047440  042526
                065504  020122  044124  020105
                065512  040514  054532  042040
                065520  043517  020123  040502
                065526  045503  030040  031061
                065534  032063  033065  034067
                065542  006471  000012
14296           065546  046111  042514  040507  MSG21:  .ASCIZ  /ILLEGAL DEVICE/<CRLF>
                065554  020114  042504  044526
                065562  042503  000200
14297           065566  020040  020040  020040  MSG22:  .ASCII  /       /
                065574  020040
14298           065576  020040  020040  020040  MSG23:  .ASCIZ  /      /
                065604     000
14299           065605     125  044516  052502  MSG24:  .ASCIZ  /UNIBUS EXERCISER /
                065612  020123  054105  051105
                065620  044503  042523  020122
                065626     000
14300           065627     117  052120  041456  MSG25:  .ASCIZ  /OPT.CP=/
                065634  036520     000
14301
14302                                              ;MSG30 THROUGH MSG38 ARE AT END OF LISTING
14303
14304           065637     125  042516  050130  EM1:    .ASCIZ  /UNEXPECTED TRAP TO 4/
                065644  041505  042524  020104
                065652  051124  050101  052040
                065660  020117  000064
14305           065664  041520  043117  050124  DH1:    .ASCIZ  /PCOFTP  PHYSPC    PSW   CPUERR/
                065672  020040  044120  051531
                065700  041520  020040  020040
                065706  051520  020127  020040
                065714  050103  042525  051122
                065722     000
14306           065723     000     001     000  DF1:    .BYTE   0,1,0,0,0
                065726     000     000
14307                                              .EVEN
14308           065730  001572  001572  001352  DT1:    .WORD   VADR,VADR,$TMP0,$TMP2,0
                065736  001356  000000
14309           065742  047125  054105  042520  EM2:    .ASCIZ  /UNEXPECTED TRAP TO 10/
                065750  052103  042105  052040
                065756  040522  020120  047524
                065764  030440  000060
14310           065770  041520  043117  050124  DH2:    .ASCIZ  /PCOFTP  PHYSPC    PSW/
                065776  020040  044120  051531
                066004  041520  020040  020040
```

```
        066012  051520  000127
14311                                   .EVEN
14312   066016  001572  001572  001352  DT2:    .WORD   VADR,VADR,$TMP0,0
        066024  000000
14313   066026  047125  054105  042520  EM3:    .ASCIZ  /UNEXPECTED TRAP TO 250(MGMT)/
        066034  052103  042105  052040
        066042  040522  020120  047524
        066050  031040  030065  046450
        066056  046507  024524     000
14314   066063     120  047503  052106  DH3:    .ASCIZ  /PCOFTP  PHYSPC    PSW    MMR0    MMR2/
        066070  020120  050040  054510
        066076  050123  020103  020040
        066104  050040  053523  020040
        066112  020040  046515  030122
        066120  020040  020040  046515
        066126  031122     000
14315           066132                  .EVEN
14316   066132  001572  001572  001352  DT3:    .WORD   VADR,VADR,$TMP0,$TMP2,$TMP3,0
        066140  001356  001360  000000
14317   066146  047125  054105  042520  EM4:    .ASCIZ  /UNEXPECTED TRAP TO 114/
        066154  052103  042105  052040
        066162  040522  020120  047524
        066170  030440  032061     000
14318   066175     120  047503  052106  DH4:    .ASCIZ  /PCOFTP  PHYSC PC    PSW   ERRREG  ERR ADR REG/
        066202  020120  050040  054510
        066210  041523  050040  020103
        066216  020040  050040  053523
        066224  020040  042440  051122
        066232  042522  020107  042440
        066240  051122  040440  051104
        066246  051040  043505     000
14319   066253     000     001     000  DF4:    .BYTE   0,1,0,0,2
        066256     000     002
14320   066260  040520  044522  054524  EM5:    .ASCIZ  /PARITY ERROR DURING DATA CHECK/
        066266  042440  051122  051117
        066274  042040  051125  047111
        066302  020107  040504  040524
        066310  041440  042510  045503
        066316     000
14321   066317     123  041522  042101  DH5:    .ASCIZ  /SRCADR  DSTADR  EADRREG  MEM ERR REG/
        066324  020122  042040  052123
        066332  042101  020122  042440
        066340  042101  051122  043505
        066346  020040  042515  020115
        066354  051105  020122  042522
        066362  000107
14322   066364     000     001     002  DF5:    .BYTE   0,1,2,0
        066367     000
14323                                   .EVEN
14324   066370  001352  001572  001360  DT5:    .WORD   $TMP0,VADR,$TMP3,$TMP2,0
        066376  001356  000000
14325   066402  051105  047522  020122  EM6:    .ASCIZ  /ERROR DURING DATA CHECK-RELOC WAS BY CP/
        066410  052504  044522  043516
        066416  042040  052101  020101
        066424  044103  041505  026513
        066432  042522  047514  020103
```

```
                                                                    I 2
                066440  040527  020123  054502
                066446  041440  000120
14326           066452  051123  040503  051104  DH6:    .ASCIZ  /SRCADR  DSTADR/
                066460  020040  051504  040524
                066466  051104     000
14327                   066472                  .EVEN
14328           066472  001352  001572  000000  DT6:    .WORD   $TMP0,VADR,0
14329           066500  051105  047522  020122  EM10:   .ASCIZ  ?ERROR DURING DATA CHECK-RELOC WAS BY I/O?
                066506  052504  044522  043516
                066514  042040  052101  020101
                066522  044103  041505  026513
                066530  042522  047514  020103
                066536  040527  020123  054502
                066544  044440  047457     000
14330           066551     123  041522  042101  DH10:   .ASCIZ  /SRCADR   DSTADR   DEVICE THAT DID XFER/
                066556  020122  020040  051504
                066564  040524  051104  020040
                066572  042040  053105  041511
                066600  020105  044124  052101
                066606  042040  042111  054040
                066614  042506  000122
14331           066620     000     001     003  DF10:   .BYTE   0,1,3,0
                066623     000
14332                                           .EVEN
14333           066624  001352  001572  001356  DT10:   .WORD   $TMP0,VADR,$TMP2,$TMP3,0
                066632  001360  000000
14334           066636  044502  024124  024523  EM11:   .ASCIZ  /BIT(S) STUCK IN MICRO-BREAK REGISTER/
                066644  051440  052524  045503
                066652  044440  020116  044515
                066660  051103  026517  051102
                066666  040505  020113  042522
                066674  044507  052123  051105
                066702     000
14335           066703     107  047517  042104  DH11:   .ASCIZ  /GOODDAT BAD DATA/
                066710  052101  041040  042101
                066716  042040  052101  000101
14336           066724     000     000          DF11:   .BYTE   0,0
14337                                           .EVEN
14338           066726  001352  001354  000000  DT11:   .WORD   $TMP0,$TMP1,0
14339           066734  041125  020105  047516  EM12:   .ASCIZ  /UBE NON-EXISTANT MEMORY ERROR/
                066742  026516  054105  051511
                066750  040524  052116  046440
                066756  046505  051117  020131
                066764  051105  047522  000122
14340           066772  044120  051531  041040  DH12:   .ASCIZ  /PHYS BUSADR/
                067000  051525  042101  000122
14341           067006     002          DF12:   .BYTE   2
14342                   067010                  .EVEN
14343           067010  001276  000000  DT12:   .WORD   $GDDAT,0
14344           067014  041115  020124  047516  EM13:   .ASCIZ  /MBT NON-EXISTANT MEMORY ERROR/
                067022  026516  054105  051511
                067030  040524  052116  046440
                067036  046505  051117  020131
                067044  051105  047522  000122
14345           067052  044120  051531  040440  DH13:   .ASCIZ  /PHYS ADDRESS/
                067060  042104  042522  051523
```

```
                067066   000
14346           067067   106   047514  052101  EM14:   .ASCIZ  /FLOATING POINT ERROR/
                067074  047111 020107  047520
                067102  047111 020124  051105
                067110  047522 000122
14347           067114  042011 040524  030524  DH14:   .ASCIZ  /      DTAT1                                DATA2/
                067122  004411 004411  040504
                067130  040524 000062
14348                                           .EVEN
14349           067134  001362 001332  001366  DT14:   .WORD   $TMP4,$REG2,$TMP6,$REG3,0
                067142  001334 000000
14350           067146    004    000     004   DF14:   .BYTE   4,0,4,0
                067151    000
14351           067152  042504 044526  042503  EM15:   .ASCIZ  /DEVICE HUNG/
                067160  044040 047125  000107
14352           067166  004411 040504  040524  DH16:   .ASCIZ  /              DATA1                          DATA2/
                067174  004461 004411  020011
                067202  020040 042040  052101
                067210  031101 000
14353           067213    005    000     005   DF16:   .BYTE   5,0,5,0
                067216    000
14354                   067220                          .EVEN
14355           067220  001502 001332  001512  DT16:   .WORD   FLTMP0,$REG2,FLTMP1,$REG3,0
                067226  001334 000000
14356           067232  030122 043040  044501  EM17:   .ASCIZ  /R0 FAILED TO LOAD CORRECTLY ON MFPT/
                067240  042514 020104  047524
                067246  046040 040517  020104
                067254  047503 051122  041505
                067262  046124 020131  047117
                067270  046440 050106  000124
14357           067276  044503 020123  047111  EM20:   .ASCIZ  /CIS INSTRUCTION FAILURE/
                067304  052123 052522  052103
                067312  047511 020116  040506
                067320  046111 051125  000105
14358           067326  000000                  ENDTAG: .WORD   0
14359                                           ;;*****************************************************************
14360                                           ;THE FOLLOWING ASCII GETS OVERLAYED WHEN THE PROGRAM RUNS.
14361           067330  050117 051105  052101  SWITCH: .ASCII  /OPERATIONAL SWITCH SETTINGS/<CRLF>
                067336  047511 040516  020114
                067344  053523 052111  044103
                067352  051440 052105  044524
                067360  043516 100123
14362           067364  053523 052111  044103          .ASCII  /SWITCH                 USE/<CRLF>
                067372  004411 052411  042523
                067400    200
14363           067401    040  030440  004465          .ASCII  /  15           HALT ON ERROR/<CRLF>
                067406  044011 046101  020124
                067414  047117 042440  051122
                067422  051117   200
14364           067425    040  030440  004464          .ASCII  /  14           LOOP ON TEST/<CRLF>
                067432  046011 047517  020120
                067440  047117 052040  051505
                067446  100124
14365           067450  020040 031461  004411          .ASCII  /  13           INHIBIT ERROR TYPEOUTS/<CRLF>
                067456  047111 044510  044502
                067464  020124 051105  047522
```

```
                067472  020122  054524  042520
                067500  052517  051524     200
14366           067505     040  030440  004462          .ASCII  /  12          INHIBIT UBE/<CRLF>
                067512  044411  044116  041111
                067520  052111  052440  042502
                067526     200
14367           067527     040  030440  004461          .ASCII  /  11          INHIBIT ITTERATIONS/<CRLF>
                067534  044411  044116  041111
                067542  052111  044440  052124
                067550  051105  052101  047511
                067556  051516     200
14368           067561     040  030440  004460          .ASCII  /  10          BELL ON ERROR/<CRLF>
                067566  041011  046105  020114
                067574  047117  042440  051122
                067602  051117     200
14369           067605     040  020040  004471          .ASCII  /   9          LOOP ON ERROR/<CRLF>
                067612  046011  047517  020120
                067620  047117  042440  051122
                067626  051117     200
14370           067631     040  020040  004470          .ASCII  ?   8          ALLOW RELOCATION VIA I/O DEVICE (NOTE CHANGE)?<CRLF>
                067636  040411  046114  053517
                067644  051040  046105  041517
                067652  052101  047511  020116
                067660  044526  020101  027511
                067666  020117  042504  044526
                067674  042503  024040  047516
                067702  042524  041440  040510
                067710  043516  024505     200
14371           067715     040  020040  004467          .ASCII  /   7          INHIBIT TYPEOUT OF THIS TEXT AND SYS SIZE/<CRLF>
                067722  044411  044116  041111
                067730  052111  052040  050131
                067736  047505  052125  047440
                067744  020106  044124  051511
                067752  052040  054105  020124
                067760  047101  020104  054523
                067766  020123  044523  042532
                067774     200
14372           067775     040  020040  004466          .ASCII  /   6          INHIBIT RELOCATION/<CRLF>
                070002  044411  044116  041111
                070010  052111  051040  046105
                070016  041517  052101  047511
                070024  100116
14373           070026  020040  032440  004411          .ASCII  /   5          INHIBIT ROUND ROBIN RELOCATION/<CRLF>
                070034  047111  044510  044502
                070042  020124  047522  047125
                070050  020104  047522  044502
                070056  020116  042522  047514
                070064  040503  044524  047117
                070072     200
14374           070073     040  020040  004464          .ASCII  /   4          INHIBIT RANDOM DISK ADDRESS/<CRLF>
                070100  044411  044116  041111
                070106  052111  051040  047101
                070114  047504  020115  044504
                070122  045523  040440  042104
                070130  042522  051523     200
14375           070135     040  020040  004463          .ASCII  /   3          INHIBIT MBT/<CRLF>
```

```
                070142  044411  044116  041111
                070150  052111  046440  052102
                070156     200
14376           070157     040  020040  004462          .ASCII  /   2           THESE THREE SWITCHES/<CRLF>
                070164  052011  042510  042523
                070172  052040  051110  042505
                070200  051440  044527  041524
                070206  042510  100123
14377           070212  020040  030440  004411          .ASCII  /   1           ARE ENCODED TO SELECT RELOCATION/<CRLF>
                070220  051101  020105  047105
                070226  047503  042504  020104
                070234  047524  051440  046105
                070242  041505  020124  042522
                070250  047514  040503  044524
                070256  047117     200
14378           070261     040  020040  004460          .ASCII  /   0           ON THE FOLLOWING DEVICES:/<CRLF>
                070266  047411  020116  044124
                070274  020105  047506  046114
                070302  053517  047111  020107
                070310  042504  044526  042503
                070316  035123     200
14379           070321     011  027060  027056          .ASCII  ?       0...RP11/RP03?<CRLF>
                070326  050122  030461  051057
                070334  030120  100063
14380
14381           070340  030411  027056  051056          .ASCII  ?       1...RK11/RK05?<CRLF>
                070346  030513  027461  045522
                070354  032460     200
14382           070357     011  027062  027056          .ASCII  ?       2...NOT USED?<CRLF>
                070364  047516  020124  051525
                070372  042105     200
14383           070375     011  027063  027056          .ASCII  ?       3...NOT USED?<CRLF>
                070402  047516  020124  051525
                070410  042105     200
14384           070413     011  027064  027056          .ASCII  ?       4...RH70/RP04?<CRLF>
                070420  044122  030067  051057
                070426  030120  100064
14385           070432  032411  027056  051056          .ASCII  ?       5...RH70/RS04 OR RS03?<CRLF>
                070440  033510  027460  051522
                070446  032060  047440  020122
                070454  051522  031460     200
14386           070461     011  027066  027056          .ASCII  ?       6...NOT USED?<CRLF>
                070466  047516  020124  051525
                070474  042105     200
14387           070477     011  027067  027056          .ASCIZ  ?       7...NOT USED?<CRLF>
                070504  047516  020124  051525
                070512  042105  000200
14388           070516  052200  042510  043040  MSG4:   .ASCII  <CRLF>/THE FOLLOWING DEVICES AND DRIVES WILL BE USED FOR RELOCATION IF B
                070524  046117  047514  044527
                070532  043516  042040  053105
                070540  041511  051505  040440
                070546  042116  042040  044522
                070554  042526  020123  044527
                070562  046114  041040  020105
                070570  051525  042105  043040
                070576  051117  051040  046105
```

```
                070604  041517  052101  047511
                070612  020116  043111  041040
                070620  052111  034040  051440
                070626  052105  100072
14389           070632  042504  044526  042503          .ASCIZ  /DEVICE DRIVES/<CRLF>
                070640  042011  044522  042526
                070646  100123     000
14390           070651     015  025012  047052  MSG30:  .ASCII  <15><12>/**NOTE** SWITCH REG BIT 8 HAS BEEN REVERSED IN REV D/<CRLF>
                070656  052117  025105  020052
                070664  053523  052111  044103
                070672  051040  043505  041040
                070700  052111  034040  044040
                070706  051501  041040  042505
                070714  020116  042522  042526
                070722  051522  042105  044440
                070730  020116  042522  020126
                070736  100104
14391           070740  047516  042524  052040          .ASCII  'NOTE THAT SWR BIT 8 SET NOW ALLOWS I/O RELOCATION'<CRLF><CRLF>
                070746  040510  020124  053523
                070754  020122  044502  020124
                070762  020070  042523  020124
                070770  047516  020127  046101
                070776  047514  051527  044440
                071004  047457  051040  046105
                071012  041517  052101  047511
                071020  100116     200
14392           071023     124  044510  020123          .ASCII  'THIS PROGRAM SUPPORTS I/O RELOCATION ONLY WITH THE FOLLOWING DEVICES:'
                071030  051120  043517  040522
                071036  020115  052523  050120
                071044  051117  051524  044440
                071052  047457  051040  046105
                071060  041517  052101  047511
                071066  020116  047117  054514
                071074  053440  052111  020110
                071102  044124  020105  047506
                071110  046114  053517  047111
                071116  020107  042504  044526
                071124  042503  035123
14393           071130  051200  030120  026063           .ASCIZ <CRLF>'RP03,RK05,RP04/5/6,RS03/4'
                071136  045522  032460  051054
                071144  030120  027464  027465
                071152  026066  051522  031460
                071160  032057     000
14394           071163     113  030502  026461  MSG31:  .ASCIZ  'KB11-EM'<15><12>
                071170  046505  005015     000
14395           071175     061  027461  032067  MSG32:  .ASCIZ  '11/74          (KB11CM)'<15><12>
                071202  020040  020040  020040
                071210  020040  020040  045450
                071216  030502  041461  024515
                071224  005015     000
14396           071227     015  041412  052520  MSG34:  .ASCIZ  <15><12>'CPU UNDER TEST FOUND TO BE A '
                071234  052440  042116  051105
                071242  052040  051505  020124
                071250  047506  047125  020104
                071256  047524  041040  020105
                071264  020101     000
```

```
14397  071267     113  030502  026461  MSG35:  .ASCIZ  'KB11-B/C''<15><12>
       071274  027502  006503  000012
14398  071302  041113  030461  042455  MSG36:  .ASCIZ  'KB11-E''<15><12>
       071310  005015     000
14399  071313     103  051511  020120  MSG37:  .ASCIZ  /CISP OPTION NOT FOUND/<15><12>
       071320  050117  044524  047117
       071326  047040  052117  043040
       071334  052517  042116  005015
       071342     000
14400  071343     103  051511  020120  MSG38:  .ASCIZ  /CISP OPTION FOUND/<15><12>
       071350  050117  044524  047117
       071356  043040  052517  042116
       071364  005015     000
14401          000001                          .END
```

```
A       041254          11469* 11575  11576  11632# 11671
AA      001603           5946# 6180*  6191*
ADCB2   013656           7356  7358#
ADCB5   014516           7582  7584#
ADCB6   015234           7730  7731   7733#
ADCB7   016172           7924  7925   7926   7928#
ADC0    011512           6774  6775   6776   6778#
ADC1    012426           6995  6996   6997   6999#
ADC2    013452           7290  7292#
ADC5    014310           7514  7515   7517#
ADC6    015030           7677  7678   7680#
ADC7    016032           7886  7887   7889#
ADDN  = 076050           5462# 11411
ADDNI = 076150           5494# 11389
ADDP  = 076070           5477# 11522
ADDPI = 076170           5502#
ADD0    016674           8087  8088   8089   8091#
ADD1    017006           8120  8121   8123#
ADD1A   017232           8203  8204   8205   8207#
ADD1B   017250           8212  8213   8215#
ADD2    017706           8350  8351   8353#
ADD3    020500           8536  8538#
ADD6    021072           8629  8630   8632#
ADD7    021564           8739  8740   8741   8743#
ARBEX   043116          11705  11762#
ARBFIN  043072          11690  11712  11742  11744  11755#
ASHCL0  030402          10052#
ASHCR0  030460          10073#
ASHL0   030172          10002#
ASHL1   031004          10180#
ASHN  = 076056           5468# 11416
ASHNI = 076156           5500# 11381  11385  11393  11447
ASHP  = 076076           5483# 11516  11527
ASHPI = 076176           5508# 11499
ASHR0   030306          10029#
ASHR1   031072          10203#
ASLB1   013004           7125  7126   7128#
ASLB1A  013230           7212  7213   7215#
ASLB3   014506           7576  7577   7579#
ASLB4   013762           7394  7395   7396   7398#
ASLB6   015216           7722  7723   7724   7726#
ASLB7   016270           7956  7957   7959#
ASL0    011634           6818  6819   6820   6821   6823#
ASL1    012602           7058  7059   7060   7062#
ASL3    014224           7483  7484   7486#
ASL4    013544           7322  7323   7324   7326#
ASL6    015000           7665  7666   7668#
ASL7    015660           7833  7834   7836#
ASRB1   013100           7161  7163#
ASRB1A  013114           7167  7168   7170#
ASRB2   013726           7378  7379   7382#
ASRB2A  013744           7387  7388   7390#
ASRB5   014446           7557  7558   7560#
ASRB6   015334           7762  7763   7765#
ASRB7   016306           7963  7964   7966#
ASR0    011662           6832  6833   6834   6836#
```

```
ASR1     012470          7015    7016    7017    7019#
ASR2     013466          7296    7297    7299#
ASR3     014210          7477    7479#
ASR6     014662          7627    7628    7630#
ASR7     015714          7847    7848    7850#
A.DSC    041252         11363   11374   11375   11382   11403   11442   11448   11459   11481   11482   11631#
A1       040026         11374#  11662
A2       040030         11375#  11662
A3       040044         11382#  11662
A4       040114         11403#  11662
A5       040234         11442#  11662
A6       040250         11448#  11662
A7       040430         11481#  11662
A8       040432         11482#  11662
B        041260         11470*  11634#  11671
BICB1    017440          8272    8273    8275#
BICB1A   017462          8283    8286#   8340    8341
BIC0     016606          8058    8059    8060    8062#
BIC1     017130          8166    8167    8169#
BIC2     017776          8379    8381    8382    8384#
BIC3     020512          8541    8543#
BIC7     022452          8887#   8889
BINB     020230          8440    8442    8445    8448    8451    8453    8456    8459#
BINB7    022220          8822    8830#
BIN1     017620          8315    8318    8321    8324    8327    8330    8333    8338#
BISB1    017426          8267    8269#
BIS0     016564          8049    8050    8052#
BIS0A    016642          8076    8078#
BIS1     017116          8160    8161    8163#
BIS2     017734          8363    8365#
BIS2A    020036          8398    8399    8401#
BIS7     022412          8877#   8879
BITB1    017416          8261    8262    8264#
BITB2    020314          8481    8482    8484#
BITB3    020676          8591    8594#
BITB6    021312          8674#
BITT1    017044          8137    8138    8140#
BITT2    020022          8391    8392    8394#
BIT0   = 000001          5373#   6399    6400    6407    6626    6634    6636   12586   12624   12665   13760   13951
BIT00  = 000001          5373#
BIT01  = 000002          5373#
BIT02  = 000004          5373#
BIT03  = 000010          5373#
BIT04  = 000020          5373#
BIT05  = 000040          5373#
BIT06  = 000100          5373#
BIT07  = 000200          5373#   6212
BIT08  = 000400          5373#
BIT09  = 001000          5373#  13163   13164
BIT1   = 000002          5373#
BIT10  = 002000          5373#  13164
BIT11  = 004000          5373#   5428    6402    6403    6405   13094   13163
BIT12  = 010000          5373#   5427    6582    6610   12499
BIT13  = 020000          5373#  12479   13164
BIT14  = 040000          5373#   5426    6126    6394    6395    6397   12752   12824   12847   12862   12873   12892   12923
                        12941   12956   12968   12987   13087   13733
```

```
BIT15 = 100000          5373#  6408  6409  6411  6413  6414  6416  6461  6470  12150 12596 12603 12715
                       12722  12842 12849 12936 12943 13684 13760 13949
BIT2  = 000004          5373#
BIT3  = 000010          5373#  12631
BIT4  = 000020          5373#  6667  9385  9944  10498 12233 12401 12405
BIT5  = 000040          5373#  6596  6618  10653 11775 12159 12161 12334 12416 12484 12636 12672 12757
                       12793  12828 12867 12897 12927 12961 12992 13801
BIT6  = 000100          5373#  12428 13160
BIT7  = 000200          5373#  6528
BIT8  = 000400          5373#  6430  6534  6571  6594  6616  12165 12293 12370 12444 12513 12580 12615
                       12658  12698 12735 12777 12822 12833 12860 12890 12921 12954 12985
BIT9  = 001000          5373#  6585  12175
BPTVEC= 000014          5373#  9296  9299* 9300* 9373* 9374*
BUFF    041752         11626  11628 11630 11632 11634 11636 11638 11640 11642 11652 11681#
BUFFAD  041330         11162  11163 11343 11344 11472 11557 11559 11652# 11671
B.DSC   041256         11386  11408 11487 11488 11496 11501 11519 11633#
B1      040054         11386# 11663
B2      040126         11408# 11663
B3      040442         11487# 11663
B4      040444         11488# 11663
B5      040510         11506# 11663
B6      040540         11519# 11663
B7      040476         11501# 11663
C       041264         11471* 11636# 11671
CACHVE= 000114          5373#  6680*  6682*  11872 11873* 11874* 11881* 11882* 12251* 12252* 13872* 13888* 14105*
                       14132*
CALLHA= 000010          5429#  12058 12167
CBIT    026122          9580#
CC0     011312          6700   6701  6702  6703  6704  6705  6706  6708#
CC1     011326          6712   6713  6714  6716#
CC2     011342          6720   6721  6722  6724#
CC3     011354          6728   6729  6731#
CC4     011370          6735   6736  6737  6739#
CHAR    041320         11186  11234 11239 11255 11336 11648#
CHAR1   037120         11186# 11661
CHAR2   037306         11234# 11661
CHAR3   037320         11239# 11661
CHAR4   037370         11255# 11661
CHAR5   037664         11336# 11661
CHKDAT  062660         11931  12105 12198 13871#
CHKSP   025706          9518#
CISER   040706         11179  11193 11215 11242 11258 11277 11296 11317 11339 11533 11566#
CISER1  040724         11438  11548 11569#
CISER2  040742         11423  11456 11572#
CISER3  040760         11377  11575#
CISOP   006012          6455*  6457  6458* 6460  6462* 6464  6473#
CISOPT= 010000          5420#  6451  6469
CISP    001555          5946#  6454* 6466  6468* 6477* 6501  9392  11129 11141
CISTRP  006030          6475   6478#
CISTST= 006600          5441#  6463
CLRTBI  063314          9290   9476  9516  11818 12254 13994#
CLR0    011430          6745   6746  6747  6748  6750#
CMPB1   017372          8251   8253#
CMPB2   020300          8474   8475  8477#
CMPB3   020710          8597   8600#
CMPC  = 076044          5460#  11240 11256 11275 11337
```

```
CMPCI = 076144          5492#  11174   11188   11210   11291   11312
CMPN  = 076052          5464#  11421
CMPNI = 076152          5496#  11373   11451
CMPNUM  026156          9594#
CMPP  = 076072          5479#
CMPPAK  040426         11480#
CMPPI = 076172          5504#  11480   11529
CMP0    016466          8012    8013    8014    8016#
CMP0A   016730          8102    8105#
CMP1    017026          8128    8129    8130    8132#
CMP1A   017146          8173    8174    8175    8177#
CMP2    017724          8357    8358    8360#
CMP7    021520          8725    8726    8728#
CNTRLC= 000003         14024#  14029
CNTRLO= 000017         14025#  14044
CNVADR  062556         11935   13207   13284   13835#
COMB1   013062          7152    7153    7155#
COMB1A  013242          7219    7220    7222#
COMB2   013640          7347    7348    7350#
COMB5   014424          7549    7551#
COMB6   015266          7743    7744    7746#
COMB7   016206          7931    7932    7934#
COMP    037062         11173#
COM0    011474          6765    6766    6767    6768    6770#
COM1    012614          7065    7066    7068#
COM3    014266          7504    7506#
COM4    013354          7253    7255#
COM6    014646          7622    7623    7625#
COM7    016004          7875    7876    7878#
CONNUM  040170         11422   11428#
CONPAK  040600         11532   11538#
CONTRL= 177746          5373#   6391    6421*   6422
CPCHK   004772          6283#
CPUERR= 177766          5373#   9386*  10416*  10499*  14227   14232*
CR    = 000015          5373#  13439
CRLF  = 000200          5373#   6108    6109    6148    6484    6652   12007   12013   13336   13439   14135   14280   14289
                       14290   14291   14293   14294   14296   14361   14362   14363   14364   14365   14366   14367   14368
                       14369   14370   14371   14372   14373   14374   14375   14376   14377   14378   14379   14381   14382
                       14383   14384   14385   14386   14387   14388   14389   14390   14391   14393
CVTLN = 076057          5469#
CVTLNI= 076157          5501#  11428
CVTLP = 076077          5484#
CVTLPI= 076177          5509#  11538
CVTNL = 076053          5465#
CVTNLI= 076153          5497#  11431
CVTNP = 076055          5467#
CVTNPI= 076155          5499#  11441
CVTPL = 076073          5480#
CVTPLI= 076173          5505#  11541
CVTPN = 076054          5466#
CVTPNI= 076154          5498#  11444
C.DSC   041262         11394   11404   11491   11492   11495   11500   11520   11635#
C1      040074         11394#  11664
C2      040116         11404#  11664
C3      040452         11491#  11664
C4      040454         11492#  11664
```

```
C5      040506          11505#  11664
C6      040542          11520#  11664
C7      040474          11500#  11664
D       041300          11642#  11671
DBINB7  022214           8827#   8859*  8860*  8864*  8865*  8866   8870*  8877   8878*  8885*  8887   8888*
DBIN7   021420           8700#   8715*  8719*  8724   8730*  8738*  8745*  8749*  8753
DDATA   020734           8604#   8608*  8610*  8614   8618   8619   8622*  8623   8628*  8635*  8642*  8643   8646*
                         8649    8653
DDATAB  021376           8670*   8671*  8676*  8679   8683*  8684   8688   8693#
DECB1   013032           7138    7139   7141#
DECB1A  013162           7190    7191   7193#
DECB2   013774           7401    7402   7404#
DECB5   014564           7605    7607#
DECB6A  015366           7774    7775   7776   7778#
DECB7   016254           7950    7951   7953#
DECDAT  037700          11338   11343#
DECO    011554           6791    6792   6793   6794   6796#
DEC1    012406           6988    6990#
DEC1A   012656           7086    7088#
DEC2    013522           7313    7314   7316#
DEC5    014242           7490    7491   7492   7494#
DEC6    015014           7671    7672   7674#
DEC7    015676           7840    7841   7843#
DEVICE  001646           5946#
DEVIND  001650           5946#  11967*  11968*  11969*  11973   11975*  11976*  11977
DF1     065723           5951    5956    5961   14306#
DF10    066620           5986   14331#
DF11    066724           5991    6021    6027   14336#
DF12    067006           5996    6001   14341#
DF14    067146           6006   14350#
DF16    067213           6016   14353#
DF4     066253           5966    5976   14319#
DF5     066364           5971   14322#
DH1     065664           5949   14305#
DH10    066551           5984   14330#
DH11    066703           5989    6019   14335#
DH12    066772           5994   14340#
DH13    067052           5999   14345#
DH14    067114           6004   14347#
DH16    067166           6014   14352#
DH2     065770           5954   14310#
DH3     066063           5959   14314#
DH4     066175           5964   14318#
DH5     066317           5969   14321#
DH6     066452           5974    6025   14326#
DISPLA= 177570           5373#   6055*  6686*  6696*  6740*  6852*  6970*  7089*  7236*  7334*  7451*  7531*  7608*
                         7688*   7800*  7902*  7987*  8106*  8222*  8339*  8423*  8490*  8549*  8601*  8658*  8698*
                         8759*   8787*  8821*  8945*  9032*  9133*  9195*  9247*  9285*  9387*  9471*  9517*  9579*
                         9617*   9773*  9858*  9906*  9945*  10000* 10101* 10150* 10232* 10251* 10292* 10342* 10360*
                        10401*  10417*  10427*  10504*  10540*  10596*  10652*  10721*  10883*  11126*  11145*  11685*  11762*
                        11803*  13164*
DIVP   = 076075          5482#
DIVPI  = 076175          5507#  11512
DIV0    030720          10151#
DIV1    031174          10233#  10246
DOCIS   036714          11142   11144#
```

```
DOMFPT  036530          11114   11116#
DONE    040642          11547   11550#
DONE7   036702          11137   11141#
DST.1A  041244          11169   11308   11559*  11560*  11567   11628#  11670
DST.1D  041242          11176   11185   11190   11208   11212   11223   11233   11238   11264   11267   11268   11273   11283
                        11287   11288   11293   11309   11314   11331   11335   11558*  11627#
DST1    037072          11176#  11658
DST10   037414          11264#  11658
DST11   037422          11267#  11659
DST12   037424          11268#  11659
DST13   037436          11273#  11659
DST14   037464          11283#  11659
DST15   037474          11287#  11659
DST16   037476          11288#  11659
DST17   037510          11293#  11659
DST18   037600          11314#  11659
DST19   037652          11331#  11659
DST2    037116          11185#  11658
DST20   037662          11335#  11659
DST3    037130          11190#  11658
DST4    037210          11208#  11658
DST5    037220          11212#  11658
DST6    037252          11223#  11658
DST7    037304          11233#  11658
DST8    037316          11238#  11658
DST9    037366          11254#  11658
DT1     065730           5950   14308#
DT10    066624           5985   14333#
DT11    066726           5990    6020    6026    14338#
DT12    067010           5995    6000   14343#
DT14    067134           6005   14349#
DT16    067220           6015   14355#
DT2     066016           5955   14312#
DT3     066132           5960    5965   14316#
DT5     066370           5970   14324#
DT6     066472           5975   14328#
DUMMY   042676          11698#
D.DSC   041276          11365   11383   11390   11395   11398   11410   11413   11461   11489   11506   11509   11515   11521
                        11524   11641#
D1      040046          11383#  11665
D10     040552          11524#  11665
D2      040064          11390#  11665
D3      040076          11395#  11665
D4      040106          11399#  11665
D5      040132          11410#  11665
D6      040140          11413#  11665
D7      040532          11515#  11665
D8      040516          11509#  11665
D9      040544          11521#  11665
E       041270          11572   11638#  11671
ECHO    063534          14065#  14083
EISOPT= 040000           5418#
EMTVEC= 000030           5373#   6076*   6271*   9202*   9204*   9242*   9243*
EMT1    024050           9201    9209#
EMT1B   024130           9209    9213    9214    9217    9219    9223    9224    9227    9228    9229    9234#
EMT1C   024134           9207    9236#
```

```
EMT1D   024146          9237    9238    9241#
EM1     065637          5948   14304#
EM10    066500          5983   14329#
EM11    066636          5988   14334#
EM12    066734          5993   14339#
EM13    067014          5998   14344#
EM14    067067          6003    6013   14346#
EM15    067152          6008   14351#
EM17    067232          6018   14356#
EM2     065742          5953   14309#
EM20    067276          6024   14357#
EM3     066026          5958   14313#
EM4     066146          5963   14317#
EM5     066260          5968   14320#
EM6     066402          5973   14325#
END     046244         12249#
ENDCIS  042610         11115   11684#
ENDCP   032312         10362   10426#
ENDKB   005660          6437    6443#
ENDM    046204         11808   12232#
ENDMEM  046202         11871   11874   12231#
ENDTAG  067326          6206   14358#
END1    046244         12250#
ENTER2  044316         11926   11959#  12008   12147
ERPRT   064422          6366    6665    8943    9382   10413   10495   11880   12250   14222#
ERRBA   002002          5946#  13032*  13033*  13034*  13396*  13399*  13400*  13401*  13413*  13414*  13415*  13416*  13417*
                       13798   13799
ERRRTN  001412          5946#
ERRVEC= 000004          5373#   6124*   6125*   6283*   6284*   6366*   6522*   6547*   6550*   6556*   6576*   6604*   6663
                        8901*   8902*   8903    8943*   9294*   9295*   9338*   9382*   9383*   9384*   9385*  10404*  10413*
                       10455*  10456*  10495*  10496*  10497*  10498*  11870*  11871*  11880*  12250*  13163*  13760*
ERR4    037516         11296#
EXIT    046062         12109   12199   12205#
EXITFL  001634          5946#
EXITRE  044230         11899   11901   11932   11938#  12207
EXPEXT  061020         11050   11061   11106   13695   13717#
EXTINS  026344          9619#
E.DSC   041266         11387   11391   11400   11419   11429   11432   11443   11445   11449   11453   11493   11505   11507
                       11510   11514   11530   11539   11542   11637#
E1      040056         11387#  11666
E10     040446         11489#  11666
E11     040464         11496#  11667
E12     040466         11497#  11667
E13     040530         11514#  11667
E14     040520         11510#  11667
E15     040564         11530#  11667
E16     040602         11539#  11667
E17     040610         11542#  11667
E18     040200         11432#  11667
E2      040066         11391#  11666
E3      040110         11400#  11666
E4      040152         11419#  11666
E5      040172         11429#  11666
E6      040236         11443#  11666
E7      040242         11445#  11666
E8      040252         11449#  11666
```

```
E9      040260          11453# 11666
F       041274          11573  11640# 11671
FACTOR  001604           5946#  6687*  6697*  7690   7801*  7904   8342   8343   8606   8662   8771   8772   8776
                         8777   8788*  8848   8902   9023   9033   9041   9057   9082   9119   9286*  9395   9401
                         9412   9618*  9718   10001* 10303  10322  10428* 10723* 11127* 11153  11157  11159  11348
                        11364  11366  11460  11462  11686* 11909  11917* 13854
FLD20 = 104426          13323  13763#
FLTADD  036342          10733  10773  10776  10806  10891  10931  10934  10964  11072#
FLTDBL  060660          10885  13673#
FLTDIV  036314          10866  11024  11058#
FLTMPY  036272          10736  10746  10755  10765  10768  10818  10826  10834  10867  10894  10904  10913  10923
                        10926  10976  10984  10992  11025  11047#
FLTMP0  001502           5946#  10895* 10947  10977* 11008  11026* 14355
FLTMP1  001512           5946#  10935* 10998* 11029* 14355
FLTSGL  060666          10727  13674#
FLTSUB  036336          10791  10814  10839  10852  10874  10949  10972  10997  11010  11032  11071#
FL20  = 104424          13313  13632  13762#
FNDLOC  037546          11303  11306#
FNDSCN  037174          11201  11204#
FNDSKP  037636          11324  11327#
FNDSPN  037270          11226  11229#
FPOPT = 020000           5419#  6292  10724
FRSTAD  001610           5946#  6687*  6697*  7801*  8421   8788*  9286*  9618*  10001* 10428* 10723* 11127* 11902
FRSTME  001612           5946#  6206*  11912
F.DSC   041272          11392  11399  11405  11409  11414  11420  11446  11454  11497  11513  11525  11531  11639#
F1      040070          11392# 11668
F10     040462          11495# 11668
F11     040512          11507# 11669
F12     040526          11513# 11669
F13     040554          11525# 11669
F14     040566          11531# 11669
F2      040104          11398# 11668
F3      040120          11405# 11668
F4      040130          11409# 11668
F5      040142          11414# 11668
F6      040154          11420# 11668
F7      040244          11446# 11668
F8      040262          11454# 11668
F9      040456          11493# 11668
GETMAP  063022          12338  12420  12641  12762  13016  13784  13924#
GIVEMA  063274          12639  12675  12760  12796  13010  13045  13985#
GNS   = ****** U         5512   6108   6109   6148   6157   6158   6160   6220   6221   6222   6225   6227   6484
                         6506   6508   6510   6587   6589   6652   12007  12013  12286  13336  13761  13762  13763
                        14135
GSTST   012170           6934#
HALT1   032166          10402#
HIADRS= 177742           5373#  13894  14112
HIMASK  041226          11469  11470  11471  11616#
HITMIS= 177752           5373#
HT    = 000011           5373#  13439
INCB1   012746           7107   7108   7110#
INCB2   014070           7436   7437   7438   7441#
INCB3   014456           7563   7565#
INCB6   015200           7717   7719#
INCB6A  015350           7768   7769   7771#
INCB7   016240           7944   7945   7947#
```

```
INC0    011574          6800    6801    6802    6803    6805#
INC1    012522          7030    7031    7032    7034#
INC3    014276          7509    7511#
INC4    013404          7266    7267    7269#
INC6    014756          7656    7657    7658    7660#
INC7    016044          7892    7894#
INSTBL  003360          6131    6138    6153#
IOMON   044236         11883   11948#  12148
IOTTST  023546          9134#
IOTVEC= 000020          5373#   6271*   9134    9147*   9197*   9198*   9244*   9245*   10379*  10399*
IOWC    001644          5946#  11925*  11958*  12042
JMP1    022746          8954    8955    8956    8958#
JMP3    023016          8974    8978#
JMP4    023054          8990    8994#
JMP5    023120          9008    9012#
JMP6    023140          9020#
JMP7    023176          9031#
JSR1    023216          9033#
JSR3    023302          9043    9057#
JSR4    023370          9069    9082#
JSR6    023446          9091    9105#
JSR7    023500          9109    9119#
KB11CM  001554          5946#   6382*   6433*   6490
KB11E   001552          5946#   6078    6089    6096    6383*   6388*   6427    6430*   6434    6488    6496    9388    9484
                       10347   11113
KB11EM  001553          5946#
KDPAR0= 172360          5373#  10586
KDPAR1= 172362          5373#
KDPAR2= 172364          5373#
KDPAR3= 172366          5373#
KDPAR4= 172370          5373#
KDPAR5= 172372          5373#
KDPAR6= 172374          5373#
KDPAR7= 172376          5373#
KDPDR0= 172320          5373#  10578
KDPDR1= 172322          5373#
KDPDR2= 172324          5373#
KDPDR3= 172326          5373#
KDPDR4= 172330          5373#
KDPDR5= 172332          5373#
KDPDR6= 172334          5373#
KDPDR7= 172336          5373#
KERSTK= 001200          5373#   6166    6244    9372    12220   12255
KIPAR0= 172340          5373#  10583   11831*  12209*  12217   13760   13844   14153   14182
KIPAR1= 172342          5373#  11832*  12210*
KIPAR2= 172344          5373#  11833*  12211*
KIPAR3= 172346          5373#  11834*  12212*
KIPAR4= 172350          5373#  10599*  11835*  11836   12103*  12189   12191*  12193   12203*  12209   12210   12211   12212
                       13885*
KIPAR5= 172352          5373#  11836*  11837*  12103   12189   12193
KIPAR6= 172354          5373#  14114   14117*  14129*
KIPAR7= 172356          5373#  11838*  11852   11866   13760
KIPDR0= 172300          5373#   6408*   6409    6411*  10575   11822*
KIPDR1= 172302          5373#  11823*
KIPDR2= 172304          5373#  11824*
KIPDR3= 172306          5373#  11825*
```

```
KIPDR4= 172310          5373#  10600*  11826*
KIPDR5= 172312          5373#  11827*
KIPDR6= 172314          5373#  11828*  14118*
KIPDR7= 172316          5373#  11829*
KM    = 000000          5430#
KTABRT  064254          6683   10398   14193#
KTABT   033222         10597#
KTEX    033462         10598   10652#
KTOPT = 100000          5417#
KTPAR   033042         10541#
KTPDR   032716         10506#
LDKT    064104         13005   13084   13147   13448   14055   14148#
LDRP3   047256         12352   12358#  12610
LDRP4   050200         12491   12498#  12829   12855   12868   12883   12898
LDRS    050504         12546   12555#  12928   12949   12962   12978   12993
LD2PNT  001702          5946#   6456    6459
LD2PT1  001704          5946#
LD3PNT  001706          5946#
LF    = 000012          5373#  13439
LKOPT = 001000          5422#   6296   11711   11763
LKS   = 177546          5386#   6294   11729*  11730   11767*  13160*
LKSRV   054362         11751   11765   13146#
LKVEC = 000100          5387#  11715*  11716*  11751*  11765*  11766*
LOADRS= 177740          5373#  13893   14110
LOCATE  037524         11295   11299#
LOCC  = 076040          5456#
LOCCHR  037534         11302#  11304*
LOCCI = 076140          5488#  11300
LOMASK  041230         11473   11475   11477   11618#
LONG.1  041306         11430   11434   11436   11540   11544   11546   11569   11645#  11650   11672
LONG.2  041312         11433   11434   11436   11543   11544   11546   11570   11646#  11651   11672
LONG1   040174         11430#  11672
LONG1P  041324         11650#
LONG2   040604         11540#  11672
LONG2P  041326         11651#
LONG3   040612         11543#  11672
LONG4   040202         11433#  11672
LOOP    010000          6657    6661#  12224   12274   12286
LOOP1   007440          6638    6644    6646    6648    6657#
LSTMEM  001614          5946#   6204*   6205*  11914
LTICKS  001700          5946#  12067   12084   13148*  13149   13151*  13153*  13154   13156*  13449
L2D0  = 076020          5438#   6455
L2D1  = 076021          5446#
L2D2  = 076022          5447#
L2D3  = 076023          5448#
L2D4  = 076024          5449#
L2D5  = 076025          5450#
L2D6  = 076026          5451#
L2D7  = 076027          5452#  11246   11418
L3D0  = 076060          5470#
L3D1  = 076061          5439#   6458
L3D2  = 076062          5471#
L3D3  = 076063          5472#
L3D4  = 076064          5473#
L3D5  = 076065          5474#
L3D6  = 076066          5475#
```

```
L3D7  = 076067          5476#  11183  11221  11231  11236  11252  11266  11271  11333  11402  11407  11412  11508
                       11518  11523
MAINT = 177750          5373#   6389*  6392  12264* 12276*
MAPEND  062552         13805   13815#
MAPH0 = 170202          5373#  10710*
MAPH00= 170202          5373#   6393
MAPH01= 170206          5373#
MAPH02= 170212          5373#
MAPH03= 170216          5373#
MAPH04= 170222          5373#
MAPH05= 170226          5373#
MAPH06= 170232          5373#
MAPH07= 170236          5373#
MAPH1 = 170206          5373#  10708*
MAPH10= 170242          5373#
MAPH11= 170246          5373#
MAPH12= 170252          5373#
MAPH13= 170256          5373#
MAPH14= 170262          5373#
MAPH15= 170266          5373#
MAPH16= 170272          5373#
MAPH17= 170276          5373#
MAPH2 = 170212          5373#  10706*
MAPH20= 170302          5373#
MAPH21= 170306          5373#
MAPH22= 170312          5373#
MAPH23= 170316          5373#
MAPH24= 170320          5373#
MAPH25= 170326          5373#
MAPH26= 170332          5373#
MAPH27= 170336          5373#
MAPH3 = 170216          5373#
MAPH30= 170342          5373#
MAPH31= 170346          5373#
MAPH32= 170352          5373#
MAPH33= 170356          5373#
MAPH34= 170362          5373#
MAPH35= 170366          5373#
MAPH36= 170372          5373#
MAPH37= 170376          5373#
MAPH4 = 170222          5373#
MAPH5 = 170226          5373#
MAPH6 = 170232          5373#
MAPH7 = 170236          5373#
MAPL0 = 170200          5373#  10655  10696  10709* 13809  13960
MAPL00= 170200          5373#
MAPL01= 170204          5373#
MAPL02= 170210          5373#
MAPL03= 170214          5373#
MAPL04= 170220          5373#
MAPL05= 170224          5373#
MAPL06= 170230          5373#
MAPL07= 170234          5373#
MAPL1 = 170204          5373#  10707*
MAPL10= 170240          5373#
MAPL11= 170244          5373#
```

```
MAPL12= 170250          5373#
MAPL13= 170254          5373#
MAPL14= 170260          5373#
MAPL15= 170264          5373#
MAPL16= 170270          5373#
MAPL17= 170274          5373#
MAPL2 = 170210          5373#  10705*
MAPL20= 170300          5373#
MAPL21= 170304          5373#
MAPL22= 170310          5373#
MAPL23= 170314          5373#
MAPL24= 170320          5373#
MAPL25= 170324          5373#
MAPL26= 170330          5373#
MAPL27= 170334          5373#
MAPL3 = 170214          5373#
MAPL30= 170340          5373#
MAPL31= 170344          5373#
MAPL32= 170350          5373#
MAPL33= 170354          5373#
MAPL34= 170360          5373#
MAPL35= 170364          5373#
MAPL36= 170370          5373#
MAPL37= 170374          5373#
MAPL4 = 170220          5373#
MAPL5 = 170224          5373#
MAPL6 = 170230          5373#
MAPL7 = 170234          5373#
MAPTBL  001766          5946#  6257*  6258*  13955  13958*  13987*
MAPTST  033510         10655#
MAPTWO  033670         10654   10698   10703#
MARKEX  027402          9874    9878    9882#
MARK1   027356          9870    9872#
MATC  = 076045          5461#  11249
MATCH   037334         11241   11245#
MATCI = 076145          5493#
MBRK    031664         10343#
MBTAS = 160116          5407#
MBTBA = 160104          5402#   5946
MBTBAE= 160174          5411#   5946
MBTCS1= 160100          5400#   5946
MBTCS2= 160110          5404#   5946
MBTCS3= 160176          5412#   5946
MBTDB = 160120          5408#
MBTDT = 160126          5410#   5946
MBTER = 160114          5406#   5946
MBTERR  054222         13095   13112#
MBTMR1= 160124          5409#
MBTMR2= 160106          5403#   5946
MBTN2   002406          5946#   6341
MBTN3   002410          5946#   6345
MBTN4   002412          5946#   6349
MBTOPT= 002000          5421#   6339    6365   11782
MBTPSW= 000776          5414#   5946
MBTSET  043226         11772   11774   11782#
MBTSRV  054050         11793   13083#
```

```
MBTST = 160112          5405#  5946
MBTTBL  002356          5946#  6337   6357   6361*  6362*  6363   11790* 11791* 11792* 11793* 11794* 11795* 13086
                       13091*  13123  13137* 14268
MBTVEC= 000774          5413#  5946
MBTWC = 160102          5401#  5946
MBT1    043264         11790#
MED6X = 076600          5510#
MED74C= 076601          5440#  6462
MEMERR= 177744          5373#  13760* 13892  13898* 14109  14123* 14124  14130  14131*
MEMHOL  054216         13102   13104  13111#
MEMSIZ  003756          6200   6202   6204#
MFPT  = 000007          5442#  6385   11135
MFPTTR  005652          6384   6440#
MHOLE   053720         13040   13043  13053#
MMON    001601          5946#  6246*  9472   10369  10511  10546  10597  11788  11869* 11900  12014  12059  12091
                       12118   12145  12162  12205  12242* 13837  13881  14115  14148  14174
MMR0  = 177572          5373#  14196  14202*
MMR1  = 177574          5373#
MMR2  = 177576          5373#  14197
MMR3  = 172516          5373#  10653  11775  12233* 12334  12416  12636  12672  12757  12793  13801
MMVEC = 000250          5373#   6683*  6684*  10365* 10398* 10605  10606  10609* 10610* 10646* 10647*
MOVB1   017346          8236   8237   8238   8239   8242#
MOVC  = 076030          5453#  11187  11235  11311
MOVCI = 076130          5485#  11206  11329
MOVE    037106         11178   11182#
MOVER   037404         11257   11261#
MOVRC = 076031          5454#  11270
MOVRCI= 076131          5486#  11262
MOVT    037454         11276   11280#
MOVTC = 076032          5455#
MOVTCI= 076132          5487#  11281  11286
MOV0    016416          7990   7991   7993#
MOV0A   016446          8005   8007#
MOV1    017216          8198   8200#
MOV7    021470          8716   8718#
MPI     031754         10361#
MRGTAB  064532         12275   12276  14252#
MRKTST  027320          9859#
MSGINX  064572         12003   13304  13332  13518  14270#
MSG1    064616         14280#
MSG10   064745         14274   14287#
MSG11   064753         14275   14288#
MSG12   064761         13340   14289#
MSG13   065074         13347   14290#
MSG14   065204         13431   14291#
MSG15   065242         14278   14292#
MSG16   065263         13365   14293#
MSG17   065372         13371   14294#
MSG2    064630         14281#
MSG20   065446         12281   14295#
MSG21   065546         14272   14273  14276  14277  14296#
MSG22   065566         13387   14297#
MSG23   065576         13392   14298#
MSG24   065605         14279   14299#
MSG25   065627          6639   14300#
MSG3    064642         13197   14282#
```

```
MSG30   070651          6656   13508   14390#
MSG31   071163          6500   14394#
MSG32   071175          6494   14395#
MSG34   071227          6487   14396#
MSG35   071267          6492   14397#
MSG36   071302          6498   14398#
MSG37   071313          6503   14399#
MSG38   071343          6505   14400#
MSG4    070516         13509   14388#
MSG5    064731         14270   14285#
MSG6    064737         14271   14286#
MSKTAB  041176         11347   11604#
MTICKS  001676          5946#   6254*  13157*  13459
MULP  = 076074          5481#
MULPI = 076174          5506#  11486   11490
MULO    030562         10102#
MXMPHI  001656          5946#   6241*  13038   13103
MXMPLO  001660          5946#   6242*  13041   13101
NEGB1   013016          7131    7132    7134#
NEGB4   014020          7413    7414    7416#
NEGB6   015304          7750    7751    7753#
NEGB7   016322          7969    7970    7972#
NEGO    011614          6810    6811    6812    6814#
NEG1    012646          7080    7081    7083#
NEG2    013426          7278    7280#
NEG5    014254          7497    7498    7500#
NEG6    014702          7634    7635    7636    7638#
NEG7    015752          7862    7863    7865#
NEXEC   001600          5946#   6687    6697    7801    8788    9286    9618   10001   10428   10723   11127   12225
NEXPAR  001616          5946#   6247*  6250*  11835   11845   11859   11950   12203   12208*  12238*  12241*
NINTAB  041206         11608#
NOMEM   044232         11915   11937   11939#
NOTRAP  006024          6474    6477#
NOTYPE  001546          5946#   6279*  13439   14047*  14065
NOZERO  040504         11498   11504#
NULLS   001624          5946#  11693   14086
NUMPAC  040232         11437   11441#
NUMRIC  040042         11376   11381#
NWBASH  001642          5946#  11922*  11954*  12056   12065*
NWBASL  001640          5946#  11921*  11953*  12017   12055   12064*  12097
NXLOC   037530         11300#  11305
NXSCAN  037156         11198#  11203
NXSKIP  037620         11321#  11326
NXSPAN  037246         11221#  11228
OAERR   022476          8894#
OBJ.1A  041250         11630#  11670
OBJ.1D  041246         11248   11254   11629#
OBJ1    037344         11248#  11660
OFFTAB  041332         11152   11656#
OKSIZ   004354          6218    6233#
OLDBAS  001636          5946#  11920*  11949*  12016   12054   12063*  12093
OLDPSW  043362         11816#  12223
ONE     041316         11415   11511   11526   11647#
ONE1    040144         11415#  11673
ONE2    040556         11526#  11673
ONE3    040522         11511#  11673
```

```
OPT.CP  001550          5946#  6483*  6640   9396  10724  11132  11688  11711  11763  11771  11782
OVFLW   024514          9288#
PACDAT  040274         11455  11459#
PACKED  040440         11483  11486#  11503
PARTBL  033166         10541  10583#
PA1500  001574          5946#  13036*  13041  13061  13097*  13099*  13101  13119  13210*  13213  13285  13297*  13403
                       13803*  13810*  13813*  13851*
PA2116  001576          5946#  13037*  13038  13062  13098*  13100*  13103  13120  13211*  13298*  13804*  13811*  13814*
                       13850*
PDRTBL  033150         10506  10575#
PERET   063732         14124#
PHYMAP  062450         13035  13402  13797#
PIRQ  = 177772          5373#  10297  10316  10317*
PIRQVE= 000240          5373#  10299*  10300*  10302*  10303*  10307*  10335*  10336*
PIRQ0   031436         10293#
PKM   = 000000          5432#
PLKCSB= 172542          5383#
PLKCSR= 172540          5382#
PLKVEC= 000104          5384#
PROT    003644          6177#  6203*  6205   6248  12239
PR0   = 000000          5373#
PR1   = 000040          5373#
PR2   = 000100          5373#  9963   9971
PR3   = 000140          5373#
PR4   = 000200          5373#  6276   6278   9136   9154   9988   9994  11817
PR5   = 000240          5373#  9143   9961  12051
PR6   = 000300          5373#
PR7   = 000340          5373#  6666   6672   6677   6679   6681   6685   9142   9192   9198   9243   9248   9249
                        9279   9281   9291   9384   9916   9918   9924   9944   9977   9982   9989  10255  10300
                       10301  10338  10497  11714  11716  11740  11766  11794  13779  13928
PS    = 177776          5373#  6271*  13760
PSM   = 010000          5427#  9169
PSW   = 177776          5373#  6126*  6147*  6653*  6664  6685*  6934  6948*  6953*  9193*  9204  9248*  9254
                        9282*  9288  9289*  9291*  9319*  9324*  9383  9474  9475*  9478  9514  9515*  9574*
                        9615*  9907  9916*  9924*  9961*  9967  9977*  9982  9988*  9994  9998*  10009  10013
                       10017*  10019  10035  10041  10187  10191  10211  10215  10299  10304*  10337*  10361  10397*
                       10415*  10419  10422  10425*  10456  10463  10496  10610  10620  11704  11708*  11735*  11740*
                       11757*  11815  11817*  12222*  12253*  12590  12628  12669  12709  12749  12790  12831  12870
                       12900  12930  12964  12995  13926  13973*  13994  14005*  14151  14152*  14187*
PSWCHK  025464          9472#
PSWTAB  064516         12273  14244#
PUM   = 030000          5433#  9168   9174   9961   9977   9982   9988   9994
PWRVEC= 000024          5373#  6271*  13757*
QV      001602          5946#  6172*  6187*  6199  6643  10703  12009  12277  13941
RDCHR = 104412         13758  13761#
RDDEC = 104416          6159  13761#
RDLIN = 104414         13759  13761#
REG   = 004000          5428#  9950   9953   9961   9982   9994
REGINX  064546         13352  13367  13421  14260#
RELE0   011152          6687   6695#
RELE1   015436          6697   7799#
RELE2   021744          7801   8786#
RELE3   024376          8788   9284#
RELE4   026226          9286   9616#
RELE5   030054          9618   9999#
RELE6   032312         10001  10426#
```

```
RELE7   033732          10428   10704   10711#
RELE8   036500          10723   11037   11039   11110#
RELE9   042572          11127   11143   11550   11683#
RELNIO  044174          11919   11928#  11930   12021
RELOC   044042           6695    7799    8786    9284    9616    9999   10426   10711   11110   11683   11898#
RELOCP  045754          11957   12024   12186#  12194
REL0    010156           6687#
REL00   011166           6695#
REL1    011214           6697#
REL11   015452           7799#
REL2    015500           7801#
REL22   021760           8786#
REL3    022006           8788#
REL33   024412           9284#
REL4    024440           9286#
REL44   026242           9616#
REL5    026270           9618#
REL55   030070           9999#
REL6    030116          10001#
REL66   032326          10426#
REL7    032354          10428#
REL77   033746          10711#
REL8    034002          10723#
REL88   036514          10726   11110#
REL9    036554          11127#
REL99   042606          11683#
RESERR  064350           6482    6669    8944    9470   10414   14209#
RESET1  032264          10418#
RESKT   064202          13020   13048   13075   13089   13135   13158   13495   14063   14174#
RESPSW  063330          14001#  14007
RESREG= 104422           6436    6465    6480   12295   12348   12372   12426   12446   12490   12515   12545   13021   13049
                        13076   13090   13136   13159   13258   13496   13535   13604   13657   13761#  13815   13852   13972
                        14064
RESTOR  006032           6435    6480#
RESTPS  063340          14005#
RESTRP  025146           9388#
RESVEC= 000010           5373#   6111*   6112*   6285*   6286*   6384*   6453*   6482*   8903*   8944*   9400*   9401*   9470*
                        10405*  10414*  12051*  12052*  12166*
RETPC   044234          11911*  11940#
RETPSW  063322          13996#  14006
RETRY   043750          11870#  12204
RKBA    002242           5946#  12431*  12730*  12772*
RKCS    002236           5946#   6527*   6536*  12155*  12435*  12700   12704*  12706   12720   12731*  12737   12744*  12746
                        12752   12773*  12779   12785*  12787   13388
RKDA    002244           5946#   6526*  12427*  12728*  12770*
RKDRV   047310           5946   12368#
RKDS    002232           5946#   6530    12428   14261
RKER    002234           5946#   6528
RKERR   051520          12703   12712#  12742   12783
RKFUN   002172           5946#  12434*  12684*  12685   12690*  12692   12784*
RKHANA  002140           5946#
RKHDA   002156           5946#  12414*  12427   12728   12770
RKHSTA  002010           5946#  12369*  12370   12375*  12698   12712*  12719*  12722*  12735   12777   12798*
RKHWC   002030           5946#  12430   12729   12771
RKLOOP  051546          12699   12719#  12736   12778
RKNEWH  002100           5946#  12764   12766*  12769
```

```
RKNEWL  002076          5946#  12761  12772
RKOLD   002046          5946#  12419  12422  12425*  12431  12727  12730
RKPSW   002250          5946#  12433*
RKREAD  052076         12695  12777#
RKRPT   051362         12373  12683#
RKSRV   051412         12432  12690#
RKTRY   002203          5946#  12415*  12702  12708*  12725*  12741  12748*  12756*  12781  12789*
RKUNIT  002124          5946#  12411
RKVEC   002246          5946#  12432*
RKWC    002240          5946#  12430*  12729*  12771*
RKWRCK  051646         12693  12735#
RKWTRY  047624         12427#  12688  12710
RK1     051572         12686  12701  12725#
RK10    001666          5946#  12421*  12424*  12435  12726*  12727*  12731  12768*  12769*  12773
RK11    001670          5946#  12374*  12393  12714  12739
RK2     051614         12728#  12750
RK3     052044         12689  12770#  12791
RNTBIN  001654          5946#  11960*  12047*  12048  12061  12297  12374  12448  12517
ROLB1   012760          7113    7114    7116#
ROLB2   013712          7370    7371    7372    7374#
ROLB3   014532          7588    7589    7591#
ROLB6   015320          7756    7757    7759#
ROLB6A  015414          7786    7787    7789#
ROLB7   016340          7976    7977    7979#
ROL0    011646          6826    6827    6829#
ROL1    012442          7002    7003    7004    7006#
ROL1A   012454          7009    7010    7012#
ROL3    014324          7521    7522    7524#
ROL4    013502          7303    7304    7306#
ROL6    014632          7615    7616    7617    7619#
ROL7    016062          7898    7899    7901#
RORB1   013046          7145    7146    7148#
RORB1A  013136          7178    7179    7181#
RORB4   013674          7362    7363    7364    7366#
RORB5   014474          7569    7570    7571    7573#
RORB6   015252          7737    7738    7740#
RORB7   016224          7938    7939    7941#
RORO    011532          6782    6783    6784    6785    6787#
RORI    012372          6980    6981    6983#
RORIA   012504          7022    7023    7024    7026#
RORR2   013370          7259    7260    7262#
RORR5   014176          7470    7471    7473#
RORR6   014722          7642    7643    7644    7646#
RORR7   015734          7855    7856    7858#
RPCC    002302          5946#
RP3BA   002220          5946#  12361*  12653*
RP3CS   002214          5946#   6560*   6561  12351*  12356*  12582  12586*  12587  12601  12611*  12617  12624*  12625
                       12654*  12660  12665*  12666  13391
RP3DA   002222          5946#  12332*  12358*  12650*
RP3DC   002224          5946#  12359*  12651*
RP3DRV  046672          5946   12291#
RP3DS   002210          5946#   6567  12349  14260
RP3ER   002212          5946#  12631
RP3ERR  050674         12585  12593#  12622  12663
RP3FUN  002170          5946#  12355*  12567*  12568  12573*  12574  12623*  12664*
RP3HAN  002136          5946#  12052  12166
```

```
RP3HDA  002152   5946#  12358  12650
RP3HDC  002154   5946#  12331* 12359  12651
RP3HST  002006   5946#   6260  12025  12030  12076  12150* 12165* 12168  12170  12197  12292* 12293  12580
                12593* 12600* 12603* 12615  12658  12677*
RP3HWC  002026   5946#  12042* 12360  12652
RP3LOO  050722  12581   12600# 12616  12659
RP3NWH  002074   5946#  12056* 12643  12645* 12648
RP3NWL  002072   5946#  12055* 12640  12653
RP3OLD  002042   5946#  12054* 12057* 12337  12340  12343* 12361  12608
RP3PSW  002230   5946#  12354*
RP3REA  051250  12577   12658#
RP3RPT  050544  12296   12566#
RP3SRV  050574  12353   12573#
RP3TRY  002202   5946#  12333* 12584  12589* 12606* 12621  12627* 12635* 12662  12668*
RP3UNI  002122   5946#  12050* 12344  12347* 12351  12609  12649
RP3VEC  002226   5946#  12353*
RP3WC   002216   5946#  12360* 12652*
RP3WCK  051012  12575   12615#
RP3WTR  047210  12349#  12350  12571  12591
RP31    050746  12569   12583  12606#
RP310   001662   5946#  12339* 12342* 12346* 12356  12607* 12608* 12609* 12611  12647* 12648* 12649* 12654
RP311   001664   5946#  12297* 12312  12595  12619
RP32    050776  12610#  12629
RP33    051210  12572   12650# 12670
RP4BA   002256   5946#  12504* 12885*
RP4BAE  002260   5946#  12503* 12884*
RP4CS1  002252   5946#   6551   6581* 12495* 12835* 12856* 12886* 14264
RP4CS2  002264   5946#   6580*  6582   6596* 12161* 12498* 12828* 12867* 12873  12897*
RP4CS3  002266   5946#
RP4DA   002262   5946#  12501*
RP4DC   002274   5946#  12500*
RP4DRV  047704   5946   12442#
RP4DS   002270   5946#   6584  12824  12836  12847  12853  12862  12881  12892
RP4ERR  052404  12827   12839# 12865  12895
RP4ER1  002272   5946#
RP4ER2  002276   5946#
RP4ER3  002300   5946#
RP4FUN  002176   5946#  12494* 12743* 12811* 12812  12816* 12817  12866* 12896*
RP4HAN  002146   5946#
RP4HDA  002162   5946#  12488* 12501
RP4HDC  002164   5946#  12487* 12500
RP4HST  002016   5946#  12443* 12444  12449* 12822  12833  12839* 12846* 12849* 12860  12890  12903*
RP4HWC  002036   5946#  12502
RP4LOO  052432  12823   12846# 12861  12891
RP4NWH  002114   5946#  12884
RP4NWL  002112   5946#  12885
RP4OF   002304   5946#  12499*
RP4OLD  002062   5946#  12503  12504
RP4PSW  002310   5946#  12493*
RP4REA  052656  12819   12890#
RP4RPT  052240  12447   12810#
RP4SRV  052264  12492   12816#
RP4TRY  002205   5946#  12489* 12826  12830* 12852* 12864  12869* 12880* 12894  12899*
RP4UNI  002132   5946#  12498
RP4VEC  002306   5946#  12492*
RP4WC   002254   5946#  12502*
```

```
RP4WCK  052510          12818  12860#
RP4WTR  050146          12491# 12815  12832
RP41    052460          12813  12825  12852#
RP411   001672           5946# 12448* 12466  12841  12875
RP42    052466          12853# 12854  12871
RP43    052620          12814  12881# 12882  12901
RSBA    002316           5946# 12559* 12980*
RSBAE   002320           5946# 12558* 12979*
RSCS1   002312           5946#  6609* 12552* 12950* 12981* 14265
RSCS2   002324           5946#  6608*  6610  6618* 12159* 12555* 12927* 12961* 12968  12992*
RSCS3   002326           5946#
RSDA    002322           5946# 12556*
RSDRV   050254           5946  12511#
RSDS    002330           5946#  6612  12550  12923  12941  12947  12956  12976  12987
RSER    002332           5946#
RSERR   053070          12926  12933# 12959  12990
RSFUN   002200           5946# 12549* 12910* 12911  12915* 12916  12960* 12991*
RSHANA  002150           5946#
RSHDA   002166           5946# 12543* 12556
RSHSTA  002020           5946# 12512* 12513  12921  12933* 12940* 12943* 12954  12985  12998*
RSHWC   002040           5946# 12557
RSLOOP  053116          12922  12940# 12955  12986
RSNEWH  002120           5946# 12979
RSNEWL  002116           5946# 12980
RSOLD   002066           5946# 12558  12559
RSPSW   002336           5946# 12548*
RSREAD  053342          12918  12985#
RSRPT   052750          12516  12909#
RSSRV   052774          12547  12915#
RSTRY   002206           5946# 12544* 12925  12929* 12946* 12958  12963* 12975* 12989  12994*
RSUNIT  002134           5946# 12555
RSVEC   002334           5946# 12547*
RSWC    002314           5946# 12557*
RSWCK   053174          12917  12954#
RSWTRY  050444          12546# 12914  12931
RS11    001674           5946# 12517* 12532  12935  12970
RS41    053144          12912  12924  12946#
RS42    053152          12947# 12948  12965
RS43    053304          12913  12976# 12977  12996
RTI1    046456          12278  12283#
RTT1    027422           9907#
RTT1EX  027624           9945#
RTT2A   027646           9946   9948#
RTT2EX  030046           9910   9957   9972   9974   9983   9985   9995   9998#
RT1     064076          14105  14140#
RUNTBL  001732           5946# 12049* 12125  12313  12316* 12394  12397* 12468  12471* 12534*
RUNTRA  001750           5946# 12181  12330* 12408* 12486* 12540* 12596* 12715* 12842* 12936*
SAV   = 000224           5519# 5523
SAVPSW  027644           9907* 9908   9914   9939   9947#  9948   9998
SAVREG= 104420           6381  6452   12291  12368  12442  12511  13004  13083  13146  13194  13447  13535  13554
                        13629  13761# 13797  13835  13930  14053
SAVRNH  041234          11147* 11151  11621#
SAVRNL  041232          11146* 11150  11620#
SBCB1   012772           7119  7120   7122#
SBCB3   014552           7599  7601#
SBCB4   014006           7407  7408   7410#
```

```
SBCB6    015154          7707    7708    7710#
SBCB7    016152          7916    7917    7918    7920#
SBC0     011676          6840    6841    6843#
SBC1     012536          7037    7038    7039    7041#
SBC1A    012554          7045    7046    7047    7049#
SBC5     014336          7528    7530#
SBC6     014736          7649    7650    7652#
SBC7     015640          7825    7826    7828#
SBINB7   022204          8823#   8854*   8855    8859    8866    8870    8874*   8877*   8880    8884*   8887*   8890
SBIN7    021416          8699#   8714*   8715    8724    8730    8737*   8738    8745    8749    8753
SCAN     037144         11192   11196#
SCANC  = 076042          5458#
SCANCI= 076142          5490#   11198
SDATA    020732          8603#   8609*   8610    8611*   8612*   8617*   8618*   8619    8623    8628    8634*   8635    8641*
                         8646    8647*   8648*   8649    8652*   8653    8656
SDATAB   021374          8668*   8669*   8670    8672    8676    8679    8683    8684    8692#
SDPAR0= 172260          5373#   10588
SDPAR1= 172262          5373#
SDPAR2= 172264          5373#
SDPAR3= 172266          5373#
SDPAR4= 172270          5373#
SDPAR5= 172272          5373#
SDPAR6= 172274          5373#
SDPAR7= 172276          5373#
SDPDR0= 172220          5373#   10580
SDPDR1= 172222          5373#
SDPDR2= 172224          5373#
SDPDR3= 172226          5373#
SDPDR4= 172230          5373#
SDPDR5= 172232          5373#
SDPDR6= 172234          5373#
SDPDR7= 172236          5373#
SECT2    034466         10797   10800#
SECT2D   035600         10955   10958#
SECT3    035032         10858   10862#
SECT3D   036144         11016   11020#
SETOP    006034          6471    6481#
SETUP    040646         11173   11182   11196   11218   11245   11261   11280   11299   11320   11556#
SET.1D   041302         11197*  11200   11202*  11219*  11224   11227*  11643#
SET1     037162         11200#  11660
SET2     037254         11224#  11660
SIGN     041227         11474   11476   11478   11617#
SIPAR0= 172240          5373#   10373   10585   11859*  11860   11862   11864
SIPAR1= 172242          5373#   11860*  11861*
SIPAR2= 172244          5373#   11862*  11863*
SIPAR3= 172246          5373#   11864*  11865*
SIPAR4= 172250          5373#   10603*
SIPAR5= 172252          5373#
SIPAR6= 172254          5373#   10373*
SIPAR7= 172256          5373#   11866*
SIPDR0= 172200          5373#   10577   11854*
SIPDR1= 172202          5373#   11855*
SIPDR2= 172204          5373#   11856*
SIPDR3= 172206          5373#   11857*
SIPDR4= 172210          5373#   10604*
SIPDR5= 172212          5373#
```

```
SIPDR6= 172214          5373#  10374*
SIPDR7= 172216          5373#  11858*
SIXTAB  041216         11612#
SIZE    006364          6522#   6655
SIZEHI= 177762          5373#   6224
SIZELO= 177760          5373#   6217   6226
SKIP    037614         11316   11320#
SKPC  = 076041          5457#
SKPCHR  037624         11323#  11325*
SKPCI = 076141          5489#  11321
SM    = 040000          5426#   9953   9994
SOBDBL  061016         13673*  13674  13680  13692  13696*  13698#
SOB0    027104          9776   9787#
SOB1    027112          9789#   9793
SOB10   027066          9778#   9841
SOB2    027140          9789    9790   9791   9792   9794   9795   9796   9797   9800#
SOB3    027152          9804#   9812
SOB4    027204          9804    9805   9806   9807   9810   9813   9814   9817#
SOB5    027206          9819#
SOB5A   027226          9828#
SOB6    027252          9839#
SOB7    027260          9842#
SOB8    027262          9785    9844#
SOB9    027076          9783#   9854
SPAN    037234         11214   11218#
SPANC = 076043          5459#  11225
SPANCI= 076143          5491#
SPARE0  002012          5946#
SPARE1  002014          5946#
SPCHK   025736          9532#
SPLTST  031260         10252#
SRC.1A  041240         11205*  11230*  11250*  11307*  11328*  11557*  11566  11626#  11670
SRC.1D  041236         11175   11184   11189   11199   11204*  11207   11211  11222   11229*  11232  11237  11247  11251*
                       11253   11263   11272   11282   11292   11301   11306* 11313   11322   11327* 11330  11334  11556*
                       11625#
SRC1    037070         11175#  11656
SRC10   037342         11247#  11656
SRC11   037364         11253#  11657
SRC12   037412         11263#  11657
SRC13   037434         11272#  11657
SRC14   037462         11282#  11657
SRC15   037506         11292#  11657
SRC16   037532         11301#  11657
SRC17   037576         11313#  11657
SRC18   037622         11322#  11657
SRC19   037650         11330#  11657
SRC2    037114         11184#  11656
SRC20   037660         11334#  11657
SRC3    037126         11189#  11656
SRC4    037160         11199#  11656
SRC5    037206         11207#  11656
SRC6    037216         11211#  11656
SRC7    037250         11222#  11656
SRC8    037302         11232#  11656
SRC9    037314         11237#  11656
SR0   = 177572          5373#  10625  10648*  11867*  12232*  13760*
```

```
SR1    = 177574          5373#  10631  10634
SR2    = 177576          5373#  10628
SR3    = 172516          5373#  11868*  13760*  13787*
STACK  = 001200          5373#   6271
START    003612          5512    6150    6166#  13757  14137
START1   002614          5514    6055#   6056
START2   002624          5515    6074#   6104
START3   003400          5517    6155#   6162
STKLMT= 177774           5373#  10430
STMM     043326         11776   11783   11785  11787  11789  11802#  12227
SUBFLG   036476         11071*  11100   11107*  11109#
SUBN   = 076051          5463#  11406
SUBNI  = 076151          5495#  11397
SUBP   = 076071          5478#
SUBPAS   001630          5946#   6263*  11786  12258*  12259  12263*  12271  13230
SUBPI  = 076171          5503#  11494   11504
SUB0     016434          7998    7999    8001#
SUB1     017064          8146    8147    8149#
SUB1A    017172          8185    8186    8188#
SUB1B    017206          8192    8193    8195#
SUB2     017752          8369    8370    8372#
SUB2A    020072          8412    8413    8415#
SUB3     020424          8513    8514    8516#
SUB3A    020446          8521    8522    8524#
SUB6     021112          8636    8638#
SUB7     021540          8731    8732    8733   8735#
SUPSTK= 000700           5373#   6661    8942   9194   9325   9331   9366   9381  10103  10400  10469  10489  10493
                        10660   14226
SWAB0    011714          6847    6848    6849   6851#
SWAB1    013176          7197    7198    7200#
SWAB2    013416          7273    7275#
SWAB4    014052          7430    7432#
SWAB6    015400          7781    7783#
SWAB7    015766          7869    7871#
SWITCH   067330         13507   14361#
SWR    = 177570          5373#   6055
SW0    = 000001          5373#
SW00   = 000001          5373#
SW01   = 000002          5373#
SW02   = 000004          5373#
SW03   = 000010          5373#
SW04   = 000020          5373#
SW05   = 000040          5373#
SW06   = 000100          5373#
SW07   = 000200          5373#
SW08   = 000400          5373#
SW09   = 001000          5373#
SW1    = 000002          5373#
SW10   = 002000          5373#
SW11   = 004000          5373#
SW12   = 010000          5373#  11773
SW13   = 020000          5373#
SW14   = 040000          5373#  13163
SW15   = 100000          5373#
SW2    = 000004          5373#
SW3    = 000010          5373#  11784
```

```
SW4     = 000020          5373#  12298  12376  12450  12518
SW5     = 000040          5373#  11963
SW6     = 000100          5373#  11806  11898
SW7     = 000200          5373#   6649
SW8     = 000400          5373#  11918  11955
SW8MSG    007434          6650   6656#
SW9     = 001000          5373#  12110
SXRA      026720          9728#  9737*
SXRB      026722          9729#  9755
SXT0      026366          9622   9623   9624   9625   9628#
SXT1      026452          9655#
SXT2      026736          9735#
SXT3      026752          9741#
SXT4      026520          9673   9675#
SXT5      026762          9746#
SXT6      026662          9710   9711   9712   9713   9715#
SXT6A     026712          9721   9724#
SXT7      027026          9763#
SYSSIZ    001712          5946#  6539*  6573*  6601*  6621*  6626*  6634*  6636*  11980  11983  11993  12000*  12035*
                         13512  13523
SYSTID= 177764            5373#
TBITVE= 000014            5373#  9249*  9281*  9477   9490*  9493*  9508*  9913*  9915*  9942*  9943*  9944*  10301*
                         10338*  12270*
TEST1     012632          7072   7073   7074   7076#
TEST2     013344          7247   7248   7250#
TEST6     015430          7792   7793   7795#
TIMEBU    057060          6264   6269*  6270*  13453*  13458*  13463*  13469*  13478*  13485*  13492*  13493  13498#
TIMER     052222         12705  12745  12786  12800#
TKBFR     001524          5946#  14059  14061
TKBFRP    001522          5946#  14056  14062*
TKISR     063354          6275   14027#
TKVEC   = 000060          5373#  6275*  6276*
TPISR     063566          6277   11755  14081#
TPVEC   = 000064          5373#  6277*  6278*  11713*  11714*  11755*  11756*
TRANS     040776         11285  11290  11584#  11644  11649
TRANS1    037470         11285#  11660
TRANS2    037502         11290#  11660
TRANS3    041322         11649#  11660
TRANS4    041304         11644#  11671
TRAPVE= 000034            5373#  6075*  6156*  6219*  6271*  9252*  9254*  9260*  9263*  9267*  9272*  9278
TRAP1     024310          9251   9266#
TRAP1C    024350          9265   9278#
TRPRTN    006016          6453   6475#
TRTVEC= 000014            5373#
TSTB1     013210          7203   7204   7206#
TSTB2     014030          7419   7421#
TSTB2A    014040          7424   7426#
TSTB6     015126          7695   7696   7697   7699#
TST0      011452          6755   6756   6757   6758   6760#
TST1      010140          6686#
TST10     013564          7334#
TST11     014110          7451#
TST12     014340          7531#
TST13     014566          7608#
TST14     015054          7688#
TST15     015462          7800#
```

TST16	016064	7902#	
TST17	016366	7987#	
TST2	011176	6696#	
TST20	016732	8106#	
TST21	017264	8222#	
TST22	017622	8339#	
TST23	020114	8423#	
TST24	020326	8490#	
TST25	020524	8549#	
TST26	020712	8601#	
TST27	021216	8658#	
TST3	011372	6740#	
TST30	021400	8698#	
TST31	021624	8759#	
TST32	021770	8787#	
TST33	022164	8821#	
TST34	022700	8945#	
TST35	023200	9032#	
TST36	023530	9133#	
TST37	023762	9149	9195#
TST4	011716	6852#	
TST40	024176	9247#	
TST41	024422	9285#	
TST42	025130	9387#	
TST43	025446	9471#	
TST44	025670	9517#	
TST45	026104	9579#	
TST46	026252	9617#	
TST47	027044	9773#	
TST5	012320	6970#	
TST50	027302	9858#	
TST51	027404	9906#	
TST52	027624	9945#	
TST53	030100	10000#	
TST54	030544	10101#	
TST55	030702	10150#	
TST56	031156	10232#	
TST57	031242	10251#	
TST6	012660	7089#	
TST60	031420	10292#	
TST61	031646	10342#	
TST62	031736	10360#	
TST63	032150	10401#	
TST64	032246	10417#	
TST65	032336	10427#	
TST66	032700	10504#	
TST67	033024	10540#	
TST7	013274	7236#	
TST70	033204	10573	10596#
TST71	033462	10652#	
TST72	033756	10721#	
TST73	035160	10880	10883#
TST74	036536	11126#	
TST75	036714	11145#	
TST76	042610	11685#	
TST77	043116	11762#	

```
TTOPT = 000400          5423#   6300   11688
TTYCHK  042626         11686#
TYPDS = 104410          6509   12286   13761#
TYPE  = 104400          6108    6109    6148    6157    6158    6160    6220    6221    6222    6223    6225    6227    6484
                        6487    6492    6494    6498    6500    6503    6505    6506    6508    6510    6511    6587    6589
                        6639    6642    6652    6656   12005   12007   12013   12286   13164   13195   13197   13198   13200
                       13217   13219   13223   13225   13229   13233   13235   13248   13250   13253   13255   13259   13279
                       13289   13306   13316   13326   13334   13336   13340   13347   13365   13371   13376   13387   13392
                       13407   13409   13426   13428   13429   13431   13434   13439   13493   13494   13507   13508   13509
                       13520   13527   13530   13533   13534   13757   13758   13759   13761#  14135
TYPOC = 104402          6161    6507    6588    6641   13199   13221   13224   13228   13232   13234   13276   13375   13425
                       13433   13761#
TYPON = 104406         13761#
TYPOS = 104404          6224    6226    6228   13761#
TYPSIZ  057072          6651   13507#
TYPTIM  056632         12286   13196   13447#
T.END   005570          6415    6418#
T1      005422          6389#   6441
T2      005464          6396    6399#
T3      005516          6401    6404    6407#
T4      005550          6410    6413#
UBEADR  001776          5946#  13013*  13014*  13015   13017   13018   13070   13072   13771*  13772*  13783   13785   13786
UBEBA = 170004          5392#   5946
UBECC = 170002          5391#   5946
UBECLR= 170010          5394#   5946
UBECR1= 170006          5393#   5946    6305*   6306
UBECR2= 170016          5396#   5946
UBEDB = 170000          5390#   6303    6311    6316    6321
UBEERR  053724         13031   13054#
UBEGO = 170014          5395#
UBEINI  062340         11777   13047   13769#
UBEOPT= 000200          5424#   6308
UBESAV  001772          5946#  13011*  13012*  13769*  13770*
UBESET  043166         11764   11771#
UBESRV  053434         13004#  13778
UBETBL  002340          5946#   6327   13006   13022*  13050*  13068   13412   13776   14269
UBEVEC= 000510          5397#   5946
UBE2    053552         13008   13026#
UBE3    053774         13055   13060   13066#
UBM6    015434          7689    7691*   7694    7702    7706*   7713*   7715*   7721*   7729*   7736*   7742*   7749*   7755*
                        7761*   7767*   7773*   7780*   7785*   7791    7798#
UBREAK= 177770          5434#   6081*   6082    6085*   6086    6092*   6093    6099*   6100    6103    6107*   6116*   6118*
                        6120*   6122*   6135*   6141*   6146*   6461*   6470*  10343
UBRERR  002770          6083    6087    6094    6102#
UBRK2   003012          6101    6107#
UBRTBL  003346          6115    6137    6151#
UB7     021426          8691    8704#
UDPAR0= 177660          5373#  10587
UDPAR1= 177662          5373#
UDPAR2= 177664          5373#
UDPAR3= 177666          5373#
UDPAR4= 177670          5373#
UDPAR5= 177672          5373#
UDPAR6= 177674          5373#
UDPAR7= 177676          5373#
UDPDR0= 177620          5373#  10579
```

```
UDPDR1= 177622          5373#
UDPDR2= 177624          5373#
UDPDR3= 177626          5373#
UDPDR4= 177630          5373#
UDPDR5= 177632          5373#
UDPDR6= 177634          5373#
UDPDR7= 177636          5373#
UIPAR0= 177640          5373#  10371   10584   11845*  11846   11848   11850
UIPAR1= 177642          5373#  11846*  11847*
UIPAR2= 177644          5373#  11848*  11849*
UIPAR3= 177646          5373#  11850*  11851*
UIPAR4= 177650          5373#  10601*
UIPAR5= 177652          5373#
UIPAR6= 177654          5373#  10371*
UIPAR7= 177656          5373#  11852*
UIPDR0= 177600          5373#  10576   11840*
UIPDR1= 177602          5373#  11841*
UIPDR2= 177604          5373#  11842*
UIPDR3= 177606          5373#  11843*
UIPDR4= 177610          5373#  10602*
UIPDR5= 177612          5373#
UIPDR6= 177614          5373#  10372*
UIPDR7= 177616          5373#  11844*
UM    = 140000          5431#   9161    9305    9950    9956    9982   10266   10280   10361   10463   10472
UNITNO  001652          5946#  12041*  12045   12050
USESTK= 000600          5373#
UWM6    014606          7610#   7614*   7621*   7626*   7633*   7641*   7648*   7655*   7662*   7664*   7670*   7676*   7682*
UWM7    015560          7804#   7903    7906*   7911*
UW7     015564          7802    7807#
VADR    001572          5946#  11934*  12115*  12201*  13201   13202*  13208*  13210   13212*  13836   13895*  14107*  14108*
                       14194*  14195*  14210*  14211*  14224*  14225*  14308   14312   14316   14324   14328   14333
X       064100         14138   14141#
XOR0    026420          9636    9637    9638    9641#
XOR1    026506          9661    9662    9663    9664    9666    9669#
XOR24   026542          9683#
XOR35   027012          9752    9753    9757#
XOR6    026630          9688    9691    9693    9700#
XOR6A   026634          9686*   9687*   9692*   9694*   9698    9704#   9709*   9720*   9722*
XOR6B   026636          9695*   9696*   9697*   9698    9705#
XOR7    027042          9768#
XXDP    003640          6174#   6201
XXDPC   003641          6175#   6196*   6645
$AC0    001462          5946#  10732*  10735*  10742*  10744   10753*  10754   10763*  10767*  10771*  10775*  10790*  10805*
                       10813*  10817*  10823*  10825   10831*  10833   10838*  10851*  10865*  10872*  10890*  10893*  10900*
                       10902   10911*  10912   10921*  10925*  10929*  10933*  10948*  10963*  10971*  10975*  10981*  10983
                       10989*  10991   10996*  11009*  11023*  11030*  11047   11058   11072   11077   11088   11090   13746
$AC1    001464          5946#  10731*  10735   10738   10744*  10748   10754*  10757   10764*  10778   10786   10792*  10795*
                       10796   10804*  10808   10811*  10820   10825*  10828   10833*  10835   10841   10847   10853*  10856*
                       10857   10864*  10869   10875*  10877*  10878   10889*  10893   10896   10902*  10906   10912*  10915
                       10922*  10936   10944   10950*  10953*  10954   10962*  10966   10969*  10978   10983*  10986   10991*
                       10993   10999   11005   11011*  11014*  11015   11022*  11027   11033*  11035*  11036   11047*  11058*
                       11072   11076   11089   11090*
$AC2    001466          5946#  10748*  107.5   10783   10785*  10786*  10787*  10788*  10796   10808*  10817   10828*  10838
                       10844   10846*  10847*  10848*  10849*  10857   10906*  10933   10941   10943*  10944*  10945*  10946*
                       10954   10966*  10975   10986*  10996   11002   11004*  11005*  11006*  11007*  11015
$AC3    001470          5946#  10757*  10771   10783   10735   10835*  10844   10846   10915*  10929   10941   10943   10993*
```

```
                        11002   11004
$AC4    001472           5946#
$AC5    001474           5946#
$APTHE  001200           5522    5923#
$APTSW  001236           5942#   6183
$BDADR  001274           5946#
$BDDAT  001300           5946#  13062*  13120*  13932*  13963
$BELL   001402           5946#  13164
$BUFF   001460           5946#  13635*  13660
$CHARC  056626          13439#*
$CMTAG  001250           5946#   6271
$CM1  = 000012           5946#
$CM2  = 000024           5946#
$CM3  = 000012           5946#
$CM4  = 000012           5946#
$CORE   062126          13760#
$CPUOP  001242           5944#
$CRLF   001407           5946#   6223    6511    6642   12286   13164   13195   13198   13235   13250   13255   13259   13409
                        13428   13429   13439   13494   13530   13758   13759   14031   14039   14049
$CROUT  062156          13760#
$DBLK   057660          13534#
$DB20   057670          13214   13286   13404   13535#
$DEVCT  001224           5936#  13152*
$DOAGN  046662          12286#
$DTBL   057650          13534#
$ENDAD  046652           5524    6185    6484   12286#  13164
$ENDCT  046516           6271   12286#
$ENULL  046666          12286#
$ENV    001234           5940#   6178   13164
$ENVM   001235           5941#   6181
$EOP    046460          12267   12286#
$EOPCT  046510           6271*  12286#
$ERFLG  001254           5946#  12107   12151*  13163*  13164*
$ERMAX  001267           5946#   6271*  13163*
$ERPSW  001632           5946#
$ERROR  054710           6076    6271    9197    9242   13164#
$ERRPC  001270           5946#  13164*  13199   13202
$ERRTB  002414           5946#  12104   13011   13013   13245   13769   13771
$ERRTY  055132          13164   13194#
$ERTTL  001264           5946#  12286*  13164*
$ESCAP  001400           5946#   6271*  13163*  13164
$FACTO  001606           5946#  11908*  12018   12095   12098
$FATAL  001216           5933#  13164*
$FILLC  001322           5946#  13439
$FILLS  001321           5946#  13439
$FLBUF  001414           5946#  13559   13607
$FLD20  060410          13629#  13763
$FL20   060122          13554#  13762
$GDADR  001272           5946#
$GDDAT  001276           5946#  13061*  13119*  13931*  13962   14343
$GET42  046642          12286#
$HD   = 000000           5348
$HINUM  001622           5946#   6167*  11147   11151*  12304   12383   12456   12524   13663*  13682   13686   13934
$ICNT   001256           5946#  13163*
$ILLUP  061334          13757#
$ITEMB  001266           5946#  13164*  13203   13205   13237
```

```
$KTNEX  062120          13760#
$KTOUT  062110          13760#
$KT11   061750           6212*  13760#*
$LF     001410           5946#  13164   13439   13758   13759
$LONUM  001620           5946#   6168*  11146   11150*  11166   12305   12457   13663*  13687   13935
$LPADR  001260           5946#   6271*   6687*   6697*   7801*   8788*   9286*   9618*  10001*  10428*  10723*  11127*  13163*
$LPERR  001262           5946#   6104*   6271*   6687*   6697*   7801*   8788*   9286*   9618*  10001*  10428*  10723*  11127*
                        11148*  11948*  12087   12141*  13054   13055*  13066*  13112   13113*  13122*  13163*  13164   14125
                        14126*  14128*  14141   14198   14199*  14201*  14203   14212   14213*  14215*  14216   14228   14229*
                        14231*  14234
$LSTAD  062264          13760#*
$LSTBK  062266           6214*   6216    6228    6235   13760#*
$MAIL   001214           5924    5931#
$MAINT  001710           5946#  12265*  12275*  13224
$MBADR  001202           5924#
$MBLTH  001212           5928#
$MSGAD  001230           5938#
$MSGLG  001232           5939#
$MSGTY  001214           5932#  13164*
$MTMOU  062232          13760#
$MXCNT  054706          13163#
$NULL   001320           5946#  13439
$NWTST= 000001           6686#   6696#   6740#   6852#   6970#   7089#   7236#   7334#   7451#   7531#   7608#   7688#   7800#
                         7902#   7987#   8106#   8222#   8339#   8423#   8490#   8549#   8601#   8658#   8698#   8759#   8787#
                         8821#   8945#   9032#   9133#   9195#   9247#   9285#   9387#   9471#   9517#   9579#   9617#   9773#
                         9858#   9906#   9945#  10000#  10101#  10150#  10232#  10251#  10292#  10342#  10360#  10401#  10417#
                        10427#  10504#  10540#  10596#  10652#  10721#  10883#  11126#  11145#  11685#  11762#
$OCNT   057440          13533#*
$OCTVL  060010          13535#
$OMODE  057442          13533#*
$OVER   054664          13163#
$PAS    001222           5935#
$PASS   001250           5946#   6245*   6647   12286*  13163   13234
$PASTM  001206           5926#
$POWER  061342          13757#
$PWRAD  061330          13757#
$PWRDN  061206           6271   13757#
$PWRMG  061324          13757#
$PWRUP  061254          13757#
$QUES   001406           5946#  13164   13758   13759
$RAND   060562          11165   12303   12382   12455   12523   13663#  13677   13933
$RDCHR  061352          13758#  13761
$RDDEC  061524          13759#  13761
$RDLIN  061406          13758#  13761
$RDOCT= ******  U       13761
$RDSZ = 000010          13758#
$REGAD  001324           5946#
$REG0   001326           5946#  10731   10742   10764   10804   10811   10831   10864   10872   10873   10875   10889   10900
                        10922   10962   10969   10989   11022   11030   11031   11033   13726
$REG1   001330           5946#  10732   10753   10763   10805   10813   10823   10865   10890   10911   10921   10963   10971
                        10981   11023
$REG10  001346           5946#
$REG11  001350           5946#
$REG2   001332           5946#  10738*  10790   10820*  10851   10869*  10896*  10948   10978*  11009   11027*  14349   14355
$REG3   001334           5946#  10778*  10792   10841*  10853   10873*  10936*  10950   10999*  11011   11031*  14349   14355
$REG4   001336           5946#  14125*  14128   14198*  14201   14212*  14215   14228*  14231
```

```
$REG5   001340          5946# 11693* 12271* 12281* 14031* 14039* 14049* 14081* 14082  14086*
$REG6   001342          5946#
$REG7   001344          5946#
$RESRE  060064         13536# 13761
$RTRN   001570          5946#  6671   9942  12270
$SAVPA  001556          5946# 14162  14177
$SAVPS  001566          5946# 14151* 14187
$SAVRE  060026         13536# 13761
$SAVR6  061340         13757#*
$SCOPE  054456          6271   9147   9153   9244  10399  13163#
$SETUP= 000037          6251#  6271   6484  12286  13164
$SIZE   061702          6213  13760#
$SIZEX  062162         13760#
$STUP = 177777          6251#
$SVLAD  054642         13163#
$SVPC = 000224          5524#
$SWR  = 167377          4391#  5348   5360   5946   6271   6686   6696   6740   6852   6970   7089   7236   7334
                        7451   7531   7608   7688   7800   7902   7987   8106   8222   8339   8423   8490   8549
                        8601   8658   8698   8759   8787   8821   8945   9032   9133   9195   9247   9285   9387
                        9471   9517   9579   9617   9773   9858   9906   9945  10000  10101  10150  10232  10251
                       10292  10342  10360  10401  10417  10427  10504  10540  10596  10652  10721  10883  11126
                       11145  11685  11762  12286  13163  13164
$SWRMK= 000000         13163
$SWRP   001244          5945#  6183*  6649  11773  11784  11806  11898  11918  11955  11963  11967  12110  12175
                       12298  12376  12450  12518  13163  13164
$TESTN  001220          5934#  13164*
$TIMES  001376          5946#  6271*  6686*  6696*  6853*  6971*  7800*  8787*  9285*  9617*  10000*  10427*  10721*
                       10722*  10884*  11126*  12286*  13163*
$TKB    001312          5946#  13758  14027
$TKS    001310          5946#  6271   6274*  13758
$TMP0   001352          5946#  6102*  6523*  6526   6537*  6557*  6560   6564*  6577*  6580   6588   6597*  6605*
                        6608   6619*  10316*  10729  10743  10802  10810  10830  10862  10870  10887  10901  10960
                       10968  10988  11020  11028  11138*  11566*  11569*  11572*  11575*  11933*  11961*  11965  11970*
                       11987  12027  12032  12114*  12200*  13675  13690  13722  13896*  14106*  14193*  14209*  14222*
                       14233  14308  14312  14316  14324  14328  14333  14338
$TMP1   001354          5946#  6103*  11139*  11567*  11570*  11573*  11576*  11978*  11980  11983  11985*  12035  12037*
                       12136*  12139*  14338
$TMP10  001372          5946#  13026*  13053*  13057  13085*  13111*  13115
$TMP11  001374          5946#  13054*  13066  13112*  13122
$TMP2   001356          5946#  10730  10751  10760  10803  10812  10822  10863  11959*  12135*  13892*  14124*  14196*
                       14227*  14308  14316  14324  14333
$TMP3   001360          5946#  12133*  13893*  14111*  14197*  14316  14324  14333
$TMP4   001362          5946#  10737*  10789  10819*  10850  10868*  10888  10909  10918  10961  10970  10980  11021
                       13894*  14112*  14349
$TMP5   001364          5946#
$TMP6   001366          5946#  10777*  10840*  10871*  14349
$TMP7   001370          5946#
$TN   = 000100          4390#  5348   6686#  6696#  6740#  6852#  6970#  7089#  7236#  7334#  7451#  7531#  7608#
                        7688#  7800#  7902#  7987#  8106#  8222#  8339#  8423#  8490#  8549#  8601#  8658#  8698#
                        8759#  8787#  8821#  8945#  9032#  9133#  9149  9195#  9247#  9285#  9387#  9471#  9517#
                        9579#  9617#  9773#  9858#  9906#  9945# 10000# 10101# 10150# 10232# 10251# 10292# 10342#
                       10360# 10401# 10417# 10427# 10504# 10540# 10573 10596# 10652# 10721# 10880  10883# 11126#
                       11145# 11685# 11762#
$TPB    001316          5946#  13439*  14072*
$TPFLG  001323          5946#  13439
$TPS    001314          5946#  6298   11691  11694*  11706  11709*  11746*  11758*  12256  12272*  12279  12282*  13439
```

```
                    14032* 14040* 14050* 14068* 14070  14085*
$TRAP   062270       6075   6156   6219   6271   9260   9263   9267   9272   9280  13761#
$TRP  = 000030      13761# 13762# 13763#
$TRPAD  062310      13761#
$TSTM   001204       5925#
$TSTNM  001252       5946#  6686*  6696*  6740*  6852*  6970*  7089*  7236*  7334*  7451*  7531*  7608*  7688*
                     7800*  7902*  7987*  8106*  8222*  8339*  8423*  8490*  8549*  8601*  8658*  8698*  8759*
                     8787*  8821*  8945*  9032*  9133*  9195*  9247*  9285*  9387*  9471*  9517*  9579*  9617*
                     9773*  9858*  9906*  9945* 10000* 10101* 10150* 10232* 10251* 10292* 10342* 10360* 10401*
                    10417* 10427* 10504* 10540* 10596* 10652* 10721* 10883* 11126* 11145* 11685* 11762* 11802*
                    12219* 12286* 13163* 13164* 13226
$TTYIN  061514      13758#
$TYPBN= ****** U    13761
$TYPDS  057444      13534#  13761
$TYPE   056410      13439#  13761
$TYPEC  056554      13439#
$TYPEX  056630      13439#
$TYPOC  057242      13533#  13761
$TYPON  057256      13533#  13761
$TYPOS  057216      13533#  13761
$UNIT   001226       5937#
$UNITM  001210       5927#
$USWR   001240       5943#
$XTSTR  054466      13163#
$$GET4= 000000      12286#
$$TMP4  001476       5946#
$$TMP6  001500       5946#
$$TRP = 000002      13761#  13762   13763
$OFILL  057441      13533#*
.     = 071367       5512#  5513#  5519   5521#  5523#  5524#  5920#  5946#  6152#  6157#  6160#  6220#  6221#
                     6225#  6227#  6271   6508#  6587#  6589#  6599   6658#  6687   6697   6707   6715   6723
                     6730   6738   6749   6759   6769   6777   6786   6795   6804   6813   6822   6828   6835
                     6842   6850   6871   6881   6891   6898   6905   6914   6930   6972   6982   6989   6998
                     7005   7011   7018   7025   7033   7040   7048   7053   7061   7067   7075   7082   7087
                     7090   7103   7109   7115   7121   7127   7133   7140   7147   7154   7162   7169   7173
                     7180   7185   7192   7199   7205   7209   7214   7221   7225   7231   7234   7237   7249
                     7254   7261   7268   7274   7279   7283   7291   7298   7305   7309   7315   7319   7325
                     7329   7332   7335   7349   7353   7357   7365   7373   7381   7389   7397   7403   7409
                     7415   7420   7425   7431   7439   7446   7449   7452   7465   7472   7478   7485   7493
                     7499   7505   7510   7516   7523   7529   7532   7550   7553   7559   7564   7572   7578
                     7583   7590   7594   7600   7606   7618   7624   7629   7637   7645   7651   7659   7667
                     7673   7679   7683   7686   7698   7703   7709   7718   7725   7732   7739   7745   7752
                     7758   7764   7770   7777   7782   7788   7794   7797   7801   7820   7827   7835   7842
                     7849   7857   7864   7870   7877   7881   7888   7893   7900   7919   7927   7933   7940
                     7946   7952   7958   7965   7971   7978   7985   7992   8000   8006   8015   8023   8043
                     8051   8061   8069   8077   8080   8090   8095   8104   8107   8122   8131   8139   8148
                     8155   8162   8168   8176   8187   8194   8199   8206   8214   8220   8223   8241   8245
                     8252   8263   8268   8274   8279   8285   8288   8337   8352   8359   8364   8371   8383
                     8393   8400   8406   8414   8419   8424   8458   8466   8470   8476   8483   8488   8515
                     8523   8528   8531   8537   8542   8547   8577   8580   8586   8593   8599   8602   8615
                     8620   8624   8631   8637   8644   8650   8654   8673   8677   8680   8685   8689   8717
                     8720   8721   8727   8734   8742   8746   8750   8754   8784   8788   8793   8799   8804
                     8808   8814   8819   8856   8861   8867   8871   8881   8891   8896   8932   8934*  8950
                     8952   8957   8962   8970   8972   8977   8993   8997   9011   9019   9030   9113   9140
                     9156   9181   9188   9201   9232   9233   9240   9251   9259   9262   9266   9271   9286
                     9293   9368   9481   9498   9540   9547   9555   9561   9572   9588   9603   9612   9618
```

		9627	9640	9651	9654	9668	9682	9699	9702	9714	9723	9727	9734	9740
		9745	9756	9762	9767	9780	9799	9816	9820	9822	9825	9850	9856	9871
		9876	9880	9912	9965	9980	9991	10001	10023	10046	10066	10090	10111	10123
		10142	10171	10195	10220	10249	10264	10278	10364	10378	10403	10410	10423	10428
		10454	10481	10608	10616	10644	10723	11127	11619#	11646#	11681#	12007#	12206	12257
		12286#	12429	12576	12588	12626	12667	12694	12707	12747	12788	13163	13164	13438#
		13439	13534#	13535#	13757	13758#	13759	14071	14135#	14315#	14327#	14342#	14354#	
.PARSR	063616	6680	11882	12251	13888	14105#	14132							

```
COMMEN  1335#  5373#
ENDCOM  1347#  5373#
ERROR   5373#  5425   6105   9246   10798  10859  10881  10956  11017  11038  11140  11180  11194  11216  11243
        11259  11278  11297  11318  11340  11378  11424  11439  11457  11484  11534  11549  11936  12086  12144
        12149  12202  13059  13063  13117  13121  13897  14127  14200  14214  14230
ESCAPE  1449#  5373#
FLTST1  10712# 10721  10883
HLT     5425#  6708   6716   6724   6731   6739   6750   6760   6770   6778   6787   6796   6805   6814   6823
        6829   6836   6843   6851   6872   6882   6892   6899   6906   6915   6931   6965   6983   6990   6999
        7006   7012   7019   7026   7034   7041   7049   7054   7062   7068   7076   7083   7088   7104   7110
        7116   7122   7128   7134   7141   7148   7155   7163   7170   7174   7181   7186   7193   7200   7206
        7210   7215   7222   7226   7232   7235   7250   7255   7262   7269   7275   7280   7284   7292   7299
        7306   7310   7316   7320   7326   7330   7333   7350   7354   7358   7366   7374   7382   7390   7398
        7404   7410   7416   7421   7426   7432   7441   7447   7450   7466   7473   7479   7486   7494   7500
        7506   7511   7517   7524   7530   7551   7554   7560   7565   7573   7579   7584   7591   7595   7601
        7607   7619   7625   7630   7638   7646   7652   7660   7668   7674   7680   7684   7687   7699   7704
        7710   7719   7726   7733   7740   7746   7753   7759   7765   7771   7778   7783   7789   7795   7821
        7828   7836   7843   7850   7858   7865   7871   7878   7882   7889   7894   7901   7920   7928   7934
        7941   7947   7953   7959   7966   7972   7979   7986   7993   8001   8007   8016   8024   8039   8044
        8052   8062   8070   8078   8081   8091   8096   8105   8123   8132   8140   8149   8156   8163   8169
        8177   8188   8195   8200   8207   8215   8221   8242   8253   8264   8269   8275   8280   8286   8338
        8353   8360   8365   8372   8384   8394   8401   8407   8415   8420   8459   8467   8471   8477   8484
        8489   8516   8524   8529   8532   8538   8543   8548   8578   8581   8587   8594   8600   8616   8621
        8626   8632   8638   8645   8651   8655   8674   8678   8681   8686   8690   8718   8722   8728   8735
        8743   8747   8751   8755   8794   8800   8805   8809   8815   8820   8857   8862   8868   8872   8882
        8892   8907   8909   8911   8913   8915   8917   8919   8921   8923   8925   8927   8929   8931   8933
        8935   8958   8978   8994   9012   9020   9031   9054   9079   9090   9102   9110   9114   9128   9148
        9189   9196#  9208   9234   9241   9246#  9261   9264   9268   9273   9332   9369   9409   9482   9499
        9504   9527   9541   9548   9556   9562   9573   9589   9604   9613   9628   9641   9655   9669   9683
        9700   9715   9724   9735   9741   9746   9757   9763   9768   9781   9800   9817   9821   9826   9832
        9836   9842   9851   9857   9873   9877   9881   9929   9934   9970   9973   9984   9996   10024  10047
        10067  10091  10124  10143  10172  10196  10221  10250  10265  10279  10311  10318  10357  10384  10396
        10407  10424  10440  10470  10490  10533  10569  10617  10642  10700  11743
MSGA    8695#  8698
MSGB    8756#  8759
MSGC    9129#  9133
MSGD    9769#  9773
MSGE    9883#  9906
MSGF    10096# 10101
MSGG    10144# 10150
MSGH    10227# 10232
MSGI    10286# 10292
MSGJ    10339# 10342
MSGK    10500# 10504
MSGL    10537# 10540
MSGM    10591# 10596
MSGN    10649# 10652
MSGO    11759# 11762
MSGP    11117# 11126
MULT    3869#  5373#
MYTAGS  5635#  5946
NEWTST  1387#  5176#  5373#  6686   6696   6740   6852   6970   7089   7236   7334   7451   7531   7608   7688
        7800   7902   7987   8106   8222   8339   8423   8490   8549   8601   8658   8698   8759   8787   8821
        8945   9032   9133   9195   9247   9285   9387   9471   9517   9579   9617   9773   9858   9906   9945
        10000  10101  10150  10232  10251  10292  10342  10360  10401  10417  10427  10504  10540  10596  10652
        10721  10883  11126  11145  11685  11762
```

```
POP      1854#  5373#  13534  13536  13663  13757  13759
PUSH     1846#  5373#  13534  13536  13663  13757  13759
RELOCA   6028#  6687   6697   7801   8788   9286   9618   10001  10428  10723  11127
RELOCB   6045#  6695   7799   8786   9284   9616   9999   10426  10711  11110  11683
SCOPE    5373#  6686   6695   6696   6740   6852   6970   7089   7236   7334   7451   7531   7608   7688   7799
         7800   7902   7987   8106   8222   8339   8423   8490   8549   8601   8658   8698   8759   8786   8787
         8821   8945   9032   9133   9195   9247   9284   9285   9387   9471   9517   9579   9616   9617   9773
         9858   9906   9945   9999   10000  10101  10150  10232  10251  10292  10342  10360  10401  10417  10426
        10427  10504  10540  10596  10652  10711  10721  10883  11110  11126  11145  11683
SETTRA   13761#  13762  13763
SETUP    1169#  5373#  6271
SKIP     1483#  5373#  6638   6646   6648   6650   9149   10573  10880
SLASH    1287#  5373#
SPACE    5373#
STARS    1256#  5373#  5524   5946   6059   6073   6164   6512   6521   6541   6553   6555   6575   6603   6686
         6696   6740   6852   6970   7089   7236   7334   7451   7531   7608   7688   7800   7902   7987   8106
         8222   8339   8423   8490   8549   8601   8658   8698   8759   8787   8821   8945   9032   9133   9195
         9247   9285   9387   9471   9517   9579   9617   9773   9858   9906   9945   10000  10101  10150  10232
        10251  10292  10342  10360  10401  10417  10427  10504  10540  10596  10652  10721  10799  10861  10883
        10957  11019  11041  11046  11053  11057  11064  11070  11126  11145  11685  11762  11769  11780  11796
        11801  11884  11897  11941  11947  12243  12248  12286  12287  12290  12364  12367  12438  12441  12507
        12510  12562  12565  12679  12682  12806  12809  12905  12908  13000  13003  13079  13082  13139  13145
        13163  13164  13165  13192  13272  13281  13293  13300  13309  13319  13329  13342  13350  13363  13378
        13419  13439  13440  13446  13500  13506  13533  13534  13535  13536  13537  13553  13610  13628  13663
        13664  13672  13699  13716  13757  13758  13759  13760  13761  13764  13768  13789  13796  13818  13834
        13859  13870  13901  13923  13980  13984  13991  13993  13998  14000  14009  14022  14076  14080  14089
        14104  14142  14147  14170  14173  14190  14192  14206  14208  14219  14221  14359
TRMTRP   13761#
TYPBIN   1790#  5373#
TYPDEC   1760#  5373#  6509   12286
TYPNAM   1536#  4387#  5373#  6484
TYPNUM   1727#  5373#
TYPOCS   1680#  5373#  6224   6226   6228
TYPOCT   1643#  5373#  6507   6588   13199  13224  13234
TYPTXT   1597#  5373#  6108   6109   6148   6157   6158   6160   6220   6221   6222   6225   6227   6506   6508
         6510   6587   6589   6652   12007  12013  12286  13336  14135
$$CMRE   5946#
$$CMTM   5946#
$$ESCA   1462#  5373#
$$NEWT   1416#  5200#  5373#  6686   6696   6740   6852   6970   7089   7236   7334   7451   7531   7608   7688
         7800   7902   7987   8106   8222   8339   8423   8490   8549   8601   8658   8698   8759   8787   8821
         8945   9032   9133   9195   9247   9285   9387   9471   9517   9579   9617   9773   9858   9906   9945
        10000  10101  10150  10232  10251  10292  10342  10360  10401  10417  10427  10504  10540  10596  10652
        10721  10883  11126  11145  11685  11762
$$SET    13761#  13762  13763
$$SKIP   1496#  5373#  9149   10573  10880
.EQUAT    168#  4386#
.HEADE     43#  4386#  5348
.KT11     311#
.SETUP   1115#  4387#  6251
.SWRHI     85#  4386#  5360
.SWRLO   4386#  5360#  5361   5362   5363
.$ACT1   4332#  4387#  5524
.$APTB   4389#
.$CATC    914#  4386#  5512
.$CMTA    971#  4386#  5031#  5946
```

```
.$DB2D   4069#
.$DB2O   4194#   5245#  13535
.$DIV    3971#
.$EOP    1913#   4387#  12286
.$ERRO   2329#   4388#  13164
.$ERRT   2510#
.$MULT   3907#
.$POWE   3615#   4387#  13757
.$RAND   3675#   5300#  13663
.$RDDE   3311#   4388#  13759
.$RDOC   3219#
.$READ   2983#   4388#  13758
.$SAVE   3387#   4388#  13536
.$SB2D   4154#
.$SB2O   4257#
.$SCOP   2131#   4847#  13163
.$SIZE   3729#   4388#  13760
.$SUPR   4296#
.$TRAP   3488#   4387#  13761
.$TYPB   2886#
.$TYPD   2808#   4387#  13534
.$TYPE   2598#   5537#  13439
.$TYPO   2711#   4388#  13533
.1170     494#   4408#   5373

. ABS.  071367     000

ERRORS DETECTED:  0

CEQKCE.BIN,CEQKCE.LST/CRF=CEQKCE.SML,CEQKCE.P11
RUN-TIME: 77 113 13 SECONDS
RUN-TIME RATIO: 1070/205=5.2
CORE USED:  33K  (65 PAGES)
```