PDP11/70

CPU INSTRUCTION EXERCISER

CEQKCC0

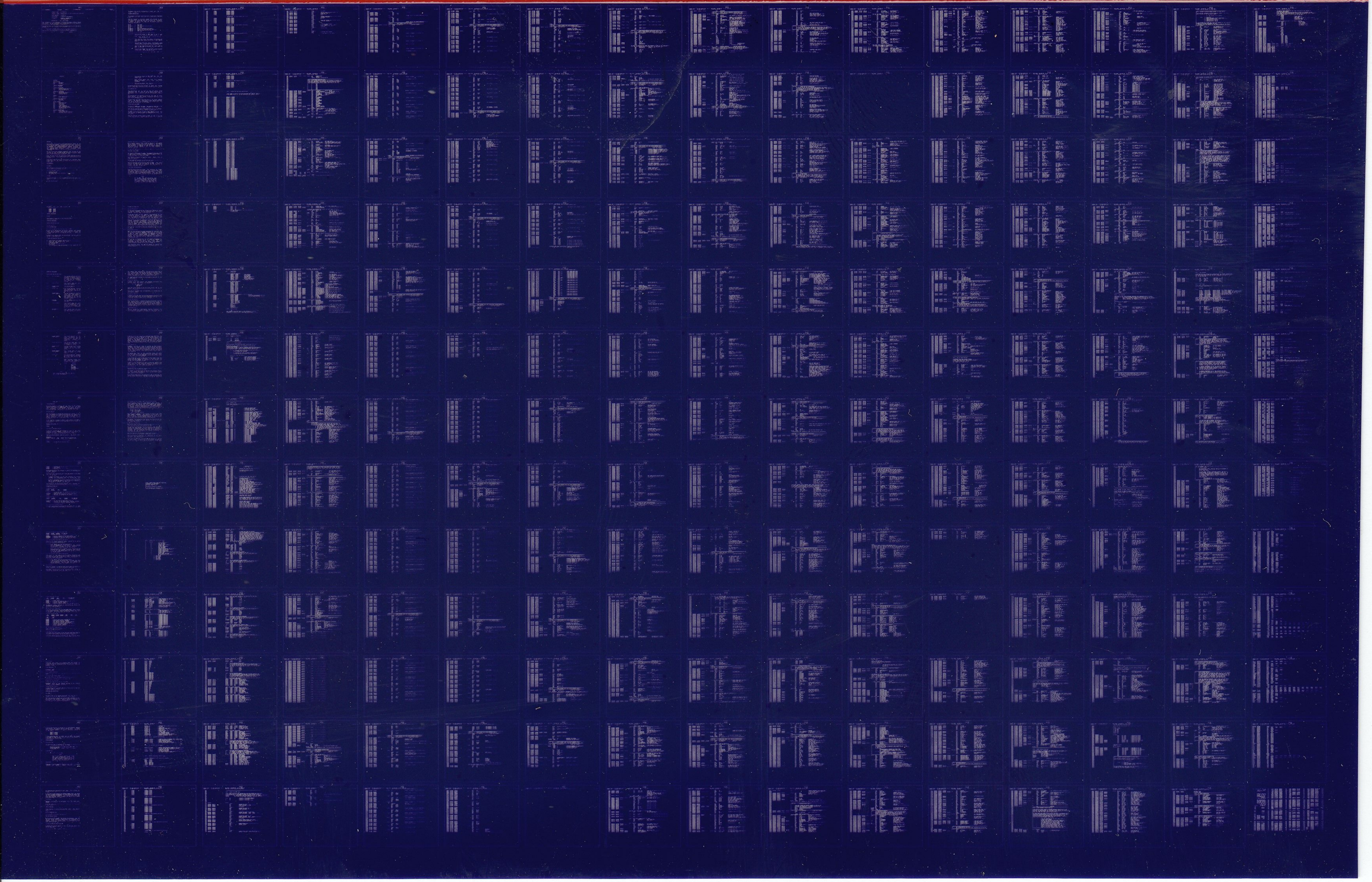
AH-7996C-MC

FICHE 1 OF 2

JUN 1978

digital

MADE IN USA



PDP11/70

CPU INSTRUCTION EXERCISER
CEQKCC0

AH-7996C-MC

FICHE 2 OF 2

JUN 1978

digital

MADE IN USA

```
            PRODUCT CODE:     AC-7994C-MC

            PRODUCT  NAME:    CEQKCC0 PDP11/70 CPU INSTRUCTION EXERCISER

            DATE CREATED:     15-MAR-78

            MAINTAINER:       DIAGNOSTIC ENGINEERING

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

           MODIFIED BY:      BILL SCHLITZKUS
```

CONTENTS

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 not deselected, 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)

Since worst case testing now occurs with all switches down, 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 Central Processor with 16K of memory, a line clock, and an LA30 (or equivalent) console.

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

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.

4.0 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.

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

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 CPU ONLY	This switch when set will cause relocation to be done by the CPU instead of a disk. See section 9.5 for a

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

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.
```

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 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.

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.

5.1.8 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.

5.1.9 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).
```

5.1.10 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.

5.1.11 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

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 "SLPERR". 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.

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

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

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 75 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 initially 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 4K bytes. 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, 776 is loaded as the bus address and the UBE is started again. On the next sequence continues until a memory timeout error occurs.

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 75 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 75 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

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 set, this routine will relocate the section via the CP (see 9.5.3). If SW<8> is not 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 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 set, 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 not 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 set, If SW<8> is not 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 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

```
.TITLE  CEQKCC PDP 11/70 CPU EXERCISOR
;*COPYRIGHT (C) SEPTEMBER 21, 1975,1978
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*PROGRAM BY DONALD W. MONROE
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-A5).
```

```
.SBTTL  OPERATIONAL SWITCH SETTINGS
;*
;*         SWITCH                  USE
;*         ------          --------------------
;*           15            HALT ON ERROR
;*           14            LOOP ON TEST
;*           13            INHIBIT ERROR TYPEOUTS
;*           12            INHIBIT UBE
;*           11            INHIBIT ITERATIONS
;*           10            BELL ON ERROR
;*            9            LOOP ON ERROR
;*            8            INHIBIT RELOCATION VIA I/O DEVICE
;*            7            INHIBIT SYSTEM SIZE TYPEOUT
;*            6            INHIBIT RELOCATION
;*            5            INHIBIT ROUND ROBIN
;*            4            INHIBIT RANDOM DISK ADDRESS
;*            3            INHIBIT MBT
;*            2            THESE THREE SWITCHES
;*            1            ARE ENCODED TO SELECT RELOCATION
;*            0            ON THE FOLLOWING DEVICES:
;*                 0...RP11/RP03
;*                 1...RK11/RK05
;*                 2...NOT USED
;*                 3...NOT USED
;*                 4...RH70/RP04
;*                 5...RH70/RS04
;*                 6...NOT USED
```

```

                                .SBTTL  BASIC DEFINITIONS

                                ;*INITIAL ADDRESS OF THE STACK POINTER *** 1200 ***
        001200                  STACK=  1200                    ;;FIRST ADDRESS OF THE STACK
        001200                  KERSTK= STACK                   ;;KERNEL STACK
        000700                  SUPSTK= STACK-300               ;;SUPERVISOR STACK
        000600                  USESTK= STACK-400               ;;USER STACK
                                .EQUIV  EMT,ERROR               ;;BASIC DEFINITION OF ERROR CALL
                                .EQUIV  IOT,SCOPE               ;;BASIC DEFINITION OF SCOPE CALL
        177776                  PS=     177776                  ;;PROCESSOR STATUS WORD
                                .EQUIV  PS,PSW
        177774                  STKLMT= 177774                  ;;STACK LIMIT REGISTER
        177772                  PIRQ=   177772                  ;;PROGRAM INTERRUPT REQUEST REGISTER
        177570                  SWR=    177570                  ;;SWITCH REGISTER
        177570                  DISPLAY=SWR

                                ;*MISCELLANEOUS DEFINITIONS
        000011                  HT=     11                      ;;CODE FOR HORIZONTAL TAB
        000012                  LF=     12                      ;;CODE LINE FEED
        000015                  CR=     15                      ;;CODE CARRIAGE RETURN
        000200                  CRLF=   200                     ;;CODE FOR CARRIAGE RETURN-LINE FEED

                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
        000000                  R0=     %0                      ;;GENERAL REGISTER
        000001                  R1=     %1                      ;;GENERAL REGISTER
        000002                  R2=     %2                      ;;GENERAL REGISTER
        000003                  R3=     %3                      ;;GENERAL REGISTER
        000004                  R4=     %4                      ;;GENERAL REGISTER
        000005                  R5=     %5                      ;;GENERAL REGISTER
        000006                  R6=     %6                      ;;GENERAL REGISTER
        000007                  R7=     %7                      ;;GENERAL REGISTER
                                .EQUIV  R0,R10                  ;;GENERAL REGISTER
                                .EQUIV  R1,R11                  ;;GENERAL REGISTER
                                .EQUIV  R2,R12                  ;;GENERAL REGISTER
                                .EQUIV  R3,R13                  ;;GENERAL REGISTER
                                .EQUIV  R4,R14                  ;;GENERAL REGISTER
                                .EQUIV  R5,R15                  ;;GENERAL REGISTER
        000006                  SP=%6                   ;;STACK POINTER
                                .EQUIV  SP,KSP                  ;;KERNEL STACK POINTER
                                .EQUIV  SP,SSP                  ;;SUPERVISOR STACK POINTER
                                .EQUIV  SP,USP                  ;;USER STACK POINTER
        000007                  PC=%7                   ;;PROGRAM COUNTER

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

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

                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
        100000          BIT15=  100000
        040000          BIT14=  40000
        020000          BIT13=  20000
        010000          BIT12=  10000
        004000          BIT11=  4000
        002000          BIT10=  2000
        001000          BIT09=  1000
        000400          BIT08=  400
        000200          BIT07=  200
        000100          BIT06=  100
        000040          BIT05=  40
        000020          BIT04=  20
        000010          BIT03=  10
        000004          BIT02=  4
        000002          BIT01=  2
        000001          BIT00=  1
                        .EQUIV  BIT09,BIT9
                        .EQUIV  BIT08,BIT8
                        .EQUIV  BIT07,BIT7
                        .EQUIV  BIT06,BIT6
                        .EQUIV  BIT05,BIT5
                        .EQUIV  BIT04,BIT4
                        .EQUIV  BIT03,BIT3
                        .EQUIV  BIT02,BIT2
                        .EQUIV  BIT01,BIT1
                        .EQUIV  BIT00,BIT0

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
        000010                  RESVEC= 10                  ;;RESERVED AND ILLEGAL INSTRUCTIONS
        000014                  TBITVEC=14                  ;;"T" BIT
        000014                  TRTVEC= 14                  ;;TRACE TRAP
        000014                  BPTVEC= 14                  ;;BREAKPOINT TRAP (BPT)
        000020                  IOTVEC= 20                  ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
        000024                  PWRVEC= 24                  ;;POWER FAIL
        000030                  EMTVEC= 30                  ;;EMULATOR TRAP (EMT) **ERROR**
        000034                  TRAPVEC=34                  ;;"TRAP" TRAP
        000060                  TKVEC=  60                  ;;TTY KEYBOARD VECTOR
        000064                  TPVEC=  64                  ;;TTY PRINTER VECTOR
        000114                  CACHVEC=114                 ;;CACHE ERROR INTERRUPT VECTOR
        000240                  PIRQVEC=240                 ;;PROGRAM INTERRUPT REQUEST VECTOR
        000250                  MMVEC=   250                ;;MEMORY MANAGEMENT VECTOR

                                .SBTTL  CACHE   REGISTER DEFINITIONS


        177740                  LOADRS = 177740             ;;LOWER 16 BITS OF ADDRESS THAT CAUSED ERROR
        177742                  HIADRS = 177742             ;;UPPER SIX BITS OF ADDRESS THAT CAUSED ERROR
        177744                  MEMERR = 177744             ;;CACHE ERROR REGISTER
        177746                  CONTRL = 177746             ;;MEMORY CONTROL REGISTER
        177750                  MAINT  = 177750             ;;MEMORY MAINTENENCE REGISTER
        177752                  HITMIS = 177752             ;;HIT MISS REGISTER "1" IMPLIES HIT IN CACHE


                                .SBTTL  CPU REGISTER DEFINITIONS


        177760                  SIZELO = 177760             ;;MEMORY SIZE REGISTER NUMBER TO PUT INTO A PAR
                                                            ;;TO GET TO THE LAST 32 WORDS OF MEMORY
        177762                  SIZEHI = 177762             ;;HIGH SIZE REGISTER, RESERVED FOR FUTURE USE
                                                            ;;CURRENTLY ALL ZERO
        177764                  SYSTID = 177764             ;;SYSTEM ID REGISTER
        177766                  CPUERR = 177766             ;;CPU ERROR REGISTER HOLDS CONDITION THAT CAUSED
                                                            ;;THE TRAP TO ERRVEC (000004)





                                .SBTTL  MEMORY MANAGEMENT DEFINITIONS


                                ;*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES

        177572                  MMR0=   177572
        177574                  MMR1=   177574
        177576                  MMR2=   177576
        172516                  MMR3=   172516
                                .EQUIV  MMR0,SR0
                                .EQUIV  MMR1,SR1
                                .EQUIV  MMR2,SR2
                                .EQUIV  MMR3,SR3

                                ;*USER "I" PAGE DESCRIPTOR REGISTERS

        177602                  UIPDR1= 177602
        177604                  UIPDR2= 177604
        177606                  UIPDR3= 177606
        177610                  UIPDR4= 177610
        177612                  UIPDR5= 177612
        177614                  UIPDR6= 177614
        177616                  UIPDR7= 177616

                                ;*USER "D" PAGE DESCRIPTOR REGISTORS

        177620                  UDPDR0= 177620
        177622                  UDPDR1= 177622
        177624                  UDPDR2= 177624
        177626                  UDPDR3= 177626
        177630                  UDPDR4= 177630
        177632                  UDPDR5= 177632
        177634                  UDPDR6= 177634
        177636                  UDPDR7= 177636

                                ;*USER "I" PAGE ADDRESS REGISTERS

        177640                  UIPAR0= 177640
        177642                  UIPAR1= 177642
        177644                  UIPAR2= 177644
        177646                  UIPAR3= 177646
        177650                  UIPAR4= 177650
        177652                  UIPAR5= 177652
        177654                  UIPAR6= 177654
        177656                  UIPAR7= 177656

                                ;*USER "D" PAGE ADDRESS REGISTERS

        177660                  UDPAR0= 177660
        177662                  UDPAR1= 177662
        177664                  UDPAR2= 177664
        177666                  UDPAR3= 177666
        177670                  UDPAR4= 177670
        177672                  UDPAR5= 177672
        177674                  UDPAR6= 177674
        177676                  UDPAR7= 177676

                                ;*SUPERVISOR "I" PAGE DESCRIPTOR REGISTERS

        172200                  SIPDR0= 172200
        172202                  SIPDR1= 172202
        172204                  SIPDR2= 172204
        172206                  SIPDR3= 172206
        172210                  SIPDR4= 172210
        172212                  SIPDR5= 172212
        172214                  SIPDR6= 172214
        172216                  SIPDR7= 172216

                                ;*SUPERVISOR "D" PAGE DESCRIPTOR REGISTERS

        172220                  SDPDR0= 172220
        172224          SDPDR2= 172224
        172226          SDPDR3= 172226
        172230          SDPDR4= 172230
        172232          SDPDR5= 172232
        172234          SDPDR6= 172234
        172236          SDPDR7= 172236

                        ;*SUPERVISOR "I" PAGE ADDRESS REGISTERS

        172240          SIPAR0= 172240
        172242          SIPAR1= 172242
        172244          SIPAR2= 172244
        172246          SIPAR3= 172246
        172250          SIPAR4= 172250
        172252          SIPAR5= 172252
        172254          SIPAR6= 172254
        172256          SIPAR7= 172256

                        ;*SUPERVISOR "D" PAGE ADDRESS REGISTERS

        172260          SDPAR0= 172260
        172262          SDPAR1= 172262
        172264          SDPAR2= 172264
        172266          SDPAR3= 172266
        172270          SDPAR4= 172270
        172272          SDPAR5= 172272
        172274          SDPAR6= 172274
        172276          SDPAR7= 172276

                        ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

        172300          KIPDR0= 172300
        172302          KIPDR1= 172302
        172304          KIPDR2= 172304
        172306          KIPDR3= 172306
        172310          KIPDR4= 172310
        172312          KIPDR5= 172312
        172314          KIPDR6= 172314
        172316          KIPDR7= 172316

                        ;*KERNEL "D" PAGE DESCRIPTOR REGISTERS

        172320          KDPDR0= 172320
        172322          KDPDR1= 172322
        172324          KDPDR2= 172324
        172326          KDPDR3= 172326
        172330          KDPDR4= 172330
        172332          KDPDR5= 172332
        172334          KDPDR6= 172334
        172336          KDPDR7= 172336

                        ;*KERNEL "I" PAGE ADDRESS REGISTERS

        172340          KIPAR0= 172340
        172342          KIPAR1= 172342
        172346                  KIPAR3= 172346
        172350                  KIPAR4= 172350
        172352                  KIPAR5= 172352
        172354                  KIPAR6= 172354
        172356                  KIPAR7= 172356

                                ;*KERNEL "D" PAGE ADDRESS REGISTERS

        172360                  KDPAR0= 172360
        172362                  KDPAR1= 172362
        172364                  KDPAR2= 172364
        172366                  KDPAR3= 172366
        172370                  KDPAR4= 172370
        172372                  KDPAR5= 172372
        172374                  KDPAR6= 172374
        172376                  KDPAR7= 172376




                                .SBTTL  UNIBUS MAP REGISTER DEFINITIONS


                                ;*THE LOWER 16 BITS OF THE MAP REGISTERS ARE LABELED 'MAPLXX'
                                ;*THE UPPER 6 BITS OF THE MAP REGISTERS ARE LABELED 'MAPHXX'


        170200                  MAPL00 = 170200
        170202                  MAPH00 = 170202
        170204                  MAPL01 = 170204
        170206                  MAPH01 = 170206
        170210                  MAPL02 = 170210
        170212                  MAPH02 = 170212
        170214                  MAPL03 = 170214
        170216                  MAPH03 = 170216
        170220                  MAPL04 = 170220
        170222                  MAPH04 = 170222
        170224                  MAPL05 = 170224
        170226                  MAPH05 = 170226
        170230                  MAPL06 = 170230
        170232                  MAPH06 = 170232
        170234                  MAPL07 = 170234
        170236                  MAPH07 = 170236
        170240                  MAPL10 = 170240
        170242                  MAPH10 = 170242
        170244                  MAPL11 = 170244
        170246                  MAPH11 = 170246
        170250                  MAPL12 = 170250
        170252                  MAPH12 = 170252
        170254                  MAPL13 = 170254
        170256                  MAPH13 = 170256
        170260                  MAPL14 = 170260
        170262                  MAPH14 = 170262
        170264                  MAPL15 = 170264
        170266                  MAPH15 = 170266
        170272                  MAPH16 = 170272
        170274                  MAPL17 = 170274
        170276                  MAPH17 = 170276
        170300                  MAPL20 = 170300
        170302                  MAPH20 = 170302
        170304                  MAPL21 = 170304
        170306                  MAPH21 = 170306
        170310                  MAPL22 = 170310
        170312                  MAPH22 = 170312
        170314                  MAPL23 = 170314
        170316                  MAPH23 = 170316
        170320                  MAPL24 = 170320
        170320                  MAPH24 = 170320
        170324                  MAPL25 = 170324
        170326                  MAPH25 = 170326
        170330                  MAPL26 = 170330
        170332                  MAPH26 = 170332
        170334                  MAPL27 = 170334
        170336                  MAPH27 = 170336
        170340                  MAPL30 = 170340
        170342                  MAPH30 = 170342
        170344                  MAPL31 = 170344
        170346                  MAPH31 = 170346
        170350                  MAPL32 = 170350
        170352                  MAPH32 = 170352
        170354                  MAPL33 = 170354
        170356                  MAPH33 = 170356
        170360                  MAPL34 = 170360
        170362                  MAPH34 = 170362
        170364                  MAPL35 = 170364
        170366                  MAPH35 = 170366
        170370                  MAPL36 = 170370
        170372                  MAPH36 = 170372
        170374                  MAPL37 = 170374
        170376                  MAPH37 = 170376
                                .EQUIV  MAPL00,MAPL0
                                .EQUIV  MAPH00,MAPH0
                                .EQUIV  MAPL01,MAPL1
                                .EQUIV  MAPH01,MAPH1
                                .EQUIV  MAPL02,MAPL2
                                .EQUIV  MAPH02,MAPH2
                                .EQUIV  MAPL03,MAPL3
                                .EQUIV  MAPH03,MAPH3
                                .EQUIV  MAPL04,MAPL4
                                .EQUIV  MAPH04,MAPH4
                                .EQUIV  MAPL05,MAPL5
                                .EQUIV  MAPH05,MAPH5
                                .EQUIV  MAPL06,MAPL6
                                .EQUIV  MAPH06,MAPH6
                                .EQUIV  MAPL07,MAPL7
                                .EQUIV  MAPH07,MAPH7
```

```
    463          000001                       AC1=     %1
    464          000002                       AC2=     %2
    465          000003                       AC3=     %3
    466          000004                       AC4=     %4
    467          000005                       AC5=     %5
    468                             ;LINE CLOCK AND PROGRAMMABLE LINE CLOCK REGISTERS
    469          172540                       PLKCSR=172540
    470          172542                       PLKCSB=172542
```

```
        177546                          LKS=177546
        000100                          LKVEC=100

                                ;UNIBUS EXERCISOR REGISTER
        170000                          UBEDB=  170000          ;DATA BUFFER
        170002                          UBECC=  170002          ;CYCLE COUNT
        170004                          UBEBA=  170004          ;BUS ADDRESS
        170006                          UBECR1= 170006          ;CRONTROL REGISTER 1
        170010                          UBECLR= 170010          ;ERROR CLEAR
        170014                          UBEGO=  170014          ;MULTI-EXERCISOR GO
        170016                          UBECR2= 170016          ;CONTROL REGISTER 2
        000510                          UBEVEC= 510             ;INTERRUPT VECTOR

                                ;MASS BUS TESTER REGISTERS
        160100                          MBTCS1= 160100
        160102                          MBTWC=  160102
        160104                          MBTBA=  160104
        160106                          MBTMR2= 160106
        160110                          MBTCS2= 160110
        160112                          MBTST=  160112
        160114                          MBTER=  160114
        160116                          MBTAS=  160116
        160120                          MBTDB=  160120
        160124                          MBTMR1= 160124
        160126                          MBTDT=  160126
        160174                          MBTBAE= 160174
        160176                          MBTCS3= 160176
        000774                          MBTVEC= 774
        000776                          MBTPSW= 776

                                ;MISCELLANEOUS BIT ASSIGNMENTS (USED IN OPT.CP)
        100000                          KTOPT=  100000                  ;BELOW BIT ASSIGNMENTS ARE USED
        040000                          EISOPT= 040000                  ;IN THE CPCHK ROUTINE
        020000                          FPOPT=  020000                  ;A BIT FOR EACH OPTION PRESENT
        002000                          MBTOPT= 002000
        001000                          LKOPT=  001000
        000400                          TTOPT=  000400
        000200                          UBEOPT= 000200
                                        .EQUIV  ERROR,HLT
                                        .EQUIV  BIT14,SM
                                        .EQUIV  BIT12,PSM
                                        .EQUIV  BIT11,REG
        000010                          CALLHANDLER=10
        000000                          KM=0
        140000                          UM=140000
        000000                          PKM=0
        030000                          PUM=30000
        177770                          UBREAK=177770

                                .SBTTL  TRAP CATCHER

        000000                          .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
```

```
                                    .SBTTL  STARTING ADDRESS(ES)
        000200                              .=200

000200  000137  003212                      JMP     @#START                 ;;JUMP TO STARTING ADDRESS OF PROGRAM
        000210                      .=210
000210  000137  002464                      JMP     @#START1
000214  000137  002474                      JMP     @#START2
                                    ;;*****************************************************************

                                    .SBTTL          ACT11 HOOKS

                                    ;*THE FOLLOWING LOCATIONS ARE SETUP TO BE USED WITH ACT11
                                    ;*
                                    ;*LOCATION 46 WILL CONTAIN THE ADDRESS OF THE LOCICAL
                                    ;*END OF THE PROGRAM.
                                    ;*LOCATION 52 IS USED TO SPECIFY PROGRAM OPERATING REQUIREMENTS
                                    ;*AND/OR RESTRICTIONS. THIS IS ACCOMPLISHED BY SETTING VARIOUS BITS
                                    ;*TO A ONE OR A ZERO. THE BITS USED AND THERE MEANING ARE:
                                    ;*
                                    ;*      BIT 15=1 PROGRAM SHOULD BE POWER FAILED WHILE RUNNING
                                    ;*             =0 NO POWER FAIL DESIRED
                                    ;*
                                    ;*      BIT 14=1 PROGRAM RUN TIME IS MEMORY SIZE DEPENDENT
                                    ;*             =0 RUN TIME IS NOT MEMORY SIZE DEPENDENT
                                    ;*
                                    ;*      BITS 13-0  MUST BE ZERO'S

        000220                              $SVPC=.                         ;;SAVE LOCATION COUNTER
        000046                              .=46                            ;;SET LOCATION COUNTER
000046  036752                              .WORD   $ENDAD                  ;;SET LOC.46 TO ADDRESS $ENDAD
        000052                              .=52                            ;;SET LOCATION COUNTER
000052  040000                              .WORD   40000                   ;;SET LOC.52 TO 40000
        000220                              .=$SVPC                         ;; RESTORE LOCATION COUNTER
```

```
                                       ;;*******************************************************************

                                       .SBTTL  COMMON TAGS

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

        001200                                 .=1200

001200                                 $CMTAG:                                   ;;START OF COMMON TAGS
001200  000000                         $PASS:  .WORD   0                         ;;CONTAINS PASS COUNT
001202  000000                         $TSTNM: .WORD   0                         ;;CONTAINS THE TEST NUMBER
001204     000                         $ERFLG: .BYTE   0                         ;;CONTAINS ERROR FLAG
        001206                                 .EVEN
001206  000000                         $ICNT:  .WORD   0                         ;;CONTAINS SUBTEST ITERATION COUNT
001210  000000                         $LPADR: .WORD   0                         ;;CONTAINS SCOPE LOOP 1200
001212  000000                         $LPERR: .WORD   0                         ;;CONTAINS SCOPE RETURN FOR ERRORS
001214  000000                         $ERTTL: .WORD   0                         ;;CONTAINS TOTAL ERRORS DETECTED
001216     000                         $ITEMB: .BYTE   0                         ;;CONTAINS ITEM CONTROL BYTE
001217     001                         $ERMAX: .BYTE   1                         ;;CONTAINS MAX. ERRORS PER TEST
001220  000000                         $ERRPC: .WORD   0                         ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001222  000000                         $GDADR: .WORD   0                         ;;CONTAINS 1200 OF 'GOOD' DATA
001224  000000                         $BDADR: .WORD   0                         ;;CONTAINS 1200 OF 'BAD' DATA
001226  000000                         $GDDAT: .WORD   0                         ;;CONTAINS 'GOOD' DATA
001230  000000                         $BDDAT: .WORD   0                         ;;CONTAINS 'BAD' DATA
001232  000000  000000  000000                 .WORD   0,0,0                     ;;RESERVED--NOT TO BE USED
001240  177560                         $TKS:   177560                            ;;TTY KBD STATUS
001242  177562                         $TKB:   177562                            ;;TTY KBD BUFFER
001244  177564                         $TPS:   177564                            ;;TTY PRINTER STATUS REG. 1200
001246  177566                         $TPB:   177566                            ;;TTY PRINTER BUFFER REG. 1200
001250     000                         $NULL:  .BYTE   0                         ;;CONTAINS NULL CHARACTER FOR FILLS
001251     002                         $FILLS: .BYTE   2                         ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001252     012                         $FILLC: .BYTE   12                        ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001253     000                         $TPFLG: .BYTE   0                         ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001254  000000                         $REGAD: .WORD   0                         ;;CONTAINS THE 1200 FROM
                                                                                 ;;WHICH  ($REG0) WAS OBTAINED
001256  000000                         $REG0:  .WORD   0                         ;;CONTAINS (($REGAD)+0)
001260  000000                         $REG1:  .WORD   0                         ;;CONTAINS (($REGAD)+2)
001262  000000                         $REG2:  .WORD   0                         ;;CONTAINS (($REGAD)+4)
001264  000000                         $REG3:  .WORD   0                         ;;CONTAINS (($REGAD)+6)
001266  000000                         $REG4:  .WORD   0                         ;;CONTAINS (($REGAD)+10)
001270  000000                         $REG5:  .WORD   0                         ;;CONTAINS (($REGAD)+12)
001272  000000                         $REG6:  .WORD   0                         ;;CONTAINS (($REGAD)+14)
001274  000000                         $REG7:  .WORD   0                         ;;CONTAINS (($REGAD)+16)
001276  000000                         $TMP0:  .WORD   0                         ;;USER DEFINED
001300  000000                         $TMP1:  .WORD   0                         ;;USER DEFINED
001302  000000                         $TMP2:  .WORD   0                         ;;USER DEFINED
001304  000000                         $TMP3:  .WORD   0                         ;;USER DEFINED
001306  000000                         $TMP4:  .WORD   0                         ;;USER DEFINED
001310  000000                         $TMP5:  .WORD   0                         ;;USER DEFINED
001312  000000                         $TMP6:  .WORD   0                         ;;USER DEFINED
001314  000000                         $TMP7:  .WORD   0                         ;;USER DEFINED
001316  000000                         $TIMES: 0                                 ;;MAX. NUMBER OF ITERATIONS
001320  000000                         $ESCAPE:0                                 ;;ESCAPE ON ERROR 1200
001322  177607  000377                 $BELL:  .ASCIZ  <207><377><377>           ;;CODE FOR BELL
```

```
001327      015                         $CRLF:   .ASCII  <15>        ;;CARRIAGE RETURN
001330   000012                         $LF:     .ASCIZ  <12>        ;;LINE FEED
001332   000000                         ERRRTN:  .WORD
001334   000044                         $FLBUFF:.BLKB    44        ;BUFFER FOR FLOATING POINT CONVERSION
001400   000000                         $BUFF:   .WORD             ;
001402   000000                         $AC0:    .WORD             ;EXTENDED EXPONENT VALUES
001404   000000                         $AC1:    .WORD             ;FOR THE SIX FLOATING POINT
001406   000000                         $AC2:    .WORD             ;ACCUMULATORS
001410   000000                         $AC3:    .WORD
001412   000000                         $AC4:    .WORD
001414   000000                         $AC5:    .WORD
001416   000000                         $STMP4:  .WORD
001420   000000                         $STMP6:  .WORD
001422   000004                         FLTMP0:  .BLKW   4         ;FLOATING POINT DBL PREC BUFFER
001432   000004                         FLTMP1:  .BLKW   4
001442   001444                         TKBFRP:  .WORD   TKBFR     ;POINTER FOR KEYBOARD BUFFER
001444   000011                         TKBFR:   .BLKW   11        ;KEYBOARD BUFFER
001466   000000                         NOTYPE:  .WORD             ;NO TYPEOUT FLAG (INHIBIT WHEN SET)
001470   000000                         OPT.CP:  .WORD             ;CPU OPTION FLAGS
001472   000006                         $RTRN:   RTT               ;RETURN FOR T-BIT TRAP
001474   000000                         VADR:    .WORD             ;BUFFER FOR VIRTUAL ADDRESS
001476   000000                         PA1500:  .WORD             ;BUFFER FOR PHYSICAL ADDRESS BITS<15:00>
001500   000000                         PA2116:  .WORD             ;PHYSICAL ADDRESS BITS<21:16>
001502      000                         NEXEC:   .BYTE             ;NO EXECUTE FLAG(NO TEST EXECUTION WHEN SET)
001503      000                         MMON:    .BYTE             ;MEMORY MGMT FLAG(MGMT IS ON WHEN NON-ZERO)
001504      000                         QV:      .BYTE             ;QV FLAG(QV PASS WHEN SET)
001505      000                         AA:      .BYTE             ;AUTO ACCEPT FLAG (AA PASS WHEN SET)
001506   000000                         FACTOR:  .WORD             ;RELOCATION FACTOR(NUMBER OF
001510   000000                         $FACTOR:.WORD              ;BYTES ABOVE BASE CODE)
001512   000000                         FRSTAD:  .WORD             ;FIRST ADDRESS OF SECTION BEING EXECUTED
001514   000000                         FRSTMEM:.WORD              ;ADDRESS OF FIRST FREE MEMORY
001516   000000                         LSTMEM:.WORD               ;ADDRESS OF LAST FREEE MEMORY(IN 28K)
001520   000000                         NEXPAR:  .WORD             ;NEXT VALUE TO PUT IN PAR0
001522   123456                         $LONUM:  .WORD   123456    ;LOW 16 BITS OF RANDOM NUMBER
001524   065432                         $HINUM:  .WORD   65432     ;HIGH 16 BITS OF RANDOM NUMBER
001526      001        001      001     NULLS:   .BYTE   1,1,1,0   ;BUFFER FOR PRINTER TEST
001531      000
001532   000060                         SUBPASS:.WORD    60        ;SUB-PASS COUNT IN ASCII
001534   000000                         $ERPSW:  .WORD             ;ERROR PSW FOR TYPEOUT
001536   000000                         EXITFL:  .WORD
001540   000000                         OLDBASE:.WORD              ;SOURCE BASE ADDRESS FOR DEVICE RELOCATION
001542   000000                         NWBASL:  .WORD             ;DEST ADDRESS FOR DEVICE RELOC BITS<15:00>
001544   000000                         NWBASH:  .WORD             ;DEST ADDRESS FOR DEVICE RELOC BITS<21:16>
001546   000000                         IOWC:    .WORD             ;TWO'S COMPLIMENT WORD COUNT FOR DEVICE RELOC
001550   000000                         DEVICE:  .WORD
001552   000000                         DEVINDX:.WORD              ;DEVICE INDEX (0 TO 7)
001554   000000                         UNITNO:  .WORD             ;DEVICE UNIT NUMBER
001556   000000                         RNTBINX:.WORD              ;INDEX TO RUN TABLE
001560   000000                         MXMMHI:  .WORD             ;BITS<21:16> OF LAST MEM ADDRESS ON SYSTEM
001562   000000                         MXMMLO:  .WORD             ;BITS<15:00> OF LAST MEM ADDRESS ON SYSTEM
001564   000000                         RP310:   .WORD             ;DATA TO LOAD INTO RP03 CS REGISTER
001566   000000                         RP311:   .WORD             ;RP03 FLAG FOR FIRST 2K OF PROGRAM
001570   000000                         RK10:    .WORD             ;DATA TO LOAD INTO RK05 CS REGISTER
001572   000000                         RK11:    .WORD             ;RK05 FLAG FOR FIRST 2K OF PROGRAM
001574   000000                         RP411:   .WORD             ;RP04 FLAG FOR FIRST 2K OF PROGRAM
```

```
001600  000000          MTICKS: .WORD                   ;ELAPSED RUN TIME IN MINUTES
001602  000000          LTICKS: .WORD                   ;LOW BYTE=NUMBER OF CLOCK INTERRUPTS (0 TO 59)
                                                        ;HIGH BYTE=ELAPSED RUN TIME IN SECONDS(0 TO 59)
001604  000000          $MAINT: .WORD                   ;CURRENT VALUE IN MAINTENANCE REGISTER
001606  000010          SYSSIZE:.BLKW   10              ;SYSTEM SIZE TABLE(ONE ENTRY FOR EACH DEVICE)
001626  000006          RUNTBL: .BLKW   6               ;RUN TIME TABLE(ONE ENTRY FOR EACH 2K BLOCK)
001642  000006          RUNTRAK:.BLKW   6               ;RUN TRACK TABLE(ONE ENTRY FOR EACH 2K BLOCK)
001656  177777          MAPTBL: .WORD   -1              ;MAP TABLE(ONE BYTE FOR EACH UNIBUS DEVICE)
001660  177777                  .WORD   -1              ;UNUSED=377, USED=LOW 5 BITS OF MAP ADDRESS
001662  000002          UBESAV: .BLKW   2               ;BASE ADDRESS OF UBE TRANSFER IN PROGRESS
001666  000002          UBEADR: .BLKW   2               ;ADDRESS THAT GETS LOADED INTO UBE BA REG
001672  000002          ERRBA:  .BLKW   2               ;18 BIT UNIBUS ADDRESS WHEN DEVICE DETECTED AN ERROR
                        .SBTTL  DEVICE HANDLER STATUS WORDS
                        ;#      EACH WORD HAS THE FOLLOWING BIT ASSIGNMENTS:
                        ;#              7               HANDLER READY
                        ;#              8               REPEAT LAST FUNCTION
                        ;#              15              ERROR
001676  000200          RP3HSTAT:.WORD  200             ;RP03
001700  000200          RKHSTAT:.WORD   200             ;RK05
001702  000200          SPARE0: .WORD   200
001704  000200          SPARE1: .WORD   200
001706  000200          RP4HSTAT:.WORD  200             ;RP04
001710  000200          RSHSTAT:.WORD   200             ;RS04
001712  000200                  .WORD   200             ;SPARE
001714  000200                  .WORD   200             ;SPARE

                        .SBTTL  DEVICE HANDLER WORD COUNTS
                        ;#      THIS TABLE GETS LOADED BY THE I/O
                        ;#      RELOCATION ROUTINE WITH THE TWO'S COMPLIMENT WORD
                        ;#      COUNT FOR THE TRANSFER FOR THE PARTICULAR DEVICE.
001716  000000          RP3HWC: .WORD                   ;RP03
001720  000000          RKHWC:  .WORD                   ;RK05
001722  000000                  .WORD                   ;SPARE
001724  000000                  .WORD                   ;SPARE
001726  000000          RP4HWC: .WORD                   ;RP04
001730  000000          RSHWC:  .WORD                   ;RS04

                        .SBTTL  DEVICE HANDLER OLD BASE ADDRESS
                        ;#      THIS TABLE GETS LOADED BY THE I/O RELOCATION ROUTINE
                        ;#      WITH THE BASE ADDRESS OF THE SOURCE DATA FOR THE
                        ;#      DEVICE THAT IS GOING TO TRANSFER THE DATA.
001732  000000          RP3OLD: .WORD                   ;RP03
001734  000000                  .WORD
001736  000000          RKOLD:  .WORD                   ;RK05
001740  000000                  .WORD
001742  000000                  .WORD                   ;SPARE
001744  000000                  .WORD                   ;SPARE
001746  000000                  .WORD
001750  000000                  .WORD                   ;SPARE
001752  000000          RP4OLD: .WORD                   ;RP04
001754  000000                  .WORD
001756  000000          RSOLD:  .WORD                   ;RS04
001760  000000                  .WORD

                        .SBTTL  DEVICE HANDLER NEW BASE ADDRESSES
                        ;*          WITH THE BASE ADDRESS OF THE DESTINATION FOR THE
                        ;*          PARTICULAR DEVICE THAT IS GOING TO DO THE TRANSFER.
001762  000000          RP3NWL: .WORD           ;RP03
001764  000000          RP3NWH: .WORD
001766  000000          RKNEWL: .WORD           ;RK05
001770  000000          RKNEWH: .WORD
001772  000000                  .WORD           ;SPARE
001774  000000                  .WORD
001776  000000                  .WORD           ;SPARE
002000  000000                  .WORD
002002  000000          RP4NWL: .WORD           ;RP04
002004  000000          RP4NWH: .WORD
002006  000000          RSNEWL: .WORD           ;RS04
002010  000000          RSNEWH: .WORD

                        .SBTTL  DEVICE HANDLER UNIT NUMBER
                        ;*      THIS TABLE GETS LOADED BY THE I/O RELOCATION ROUTINE.
                        ;*      IT TELLS THE DEVICE HANDLER WHICH UNIT NUMBER IS
                        ;*      TO DO THE TRANSFER.
002012  000000          RP3UNIT:.WORD           ;RP03
002014  000000          RKUNIT: .WORD           ;RK05
002016  000000                  .WORD           ;SPARE
002020  000000                  .WORD           ;SPARE
002022  000000          RP4UNIT:.WORD           ;RP04
002024  000000          RSUNIT: .WORD           ;RS04

                        .SBTTL  ADDRESS OF THE DEVICE HANDLERS
                        ;*      THIS TABLE CONTAINS THE ADDRESS OF THE DEVICE HANDLER
                        ;*      ROUTINES. IT IS USED BY THE I/O RELOCATION ROUTINE
                        ;*      TO TRANSFER CONTROL TO THE DEVICE HANDLER.
002026  036772          RP3HANA:.WORD   RP3DRV  ;RP03
002030  037410          RKHANA: .WORD   RKDRV   ;RK05
002032  000000                  .WORD           ;SPARE
002034  000000                  .WORD           ;SPARE
002036  040004          RP4HANA:.WORD   RP4DRV  ;RP04
002040  040354          RSHANA: .WORD   RSDRV   ;RS04

                        .SBTTL  DEVICE HANDLER DISK ADDRESS TABLE
                        ;*      THIS TABLE GETS LOADED BY THE DEVICE HANDLER WITH THE
                        ;*      DISK ADDRESS(SECTOR AND CYLINDER) OF THE CURRENT
                        ;*      TRANSFER.
002042  000000          RP3HDA: .WORD           ;RP03 DISK ADDRESS
002044  000000          RP3HDC: .WORD           ;RP03 DESIRED CYLINDER
002046  000000          RKHDA:  .WORD           ;RK05 DISK ADDRESS
002050  000000                  .WORD           ;SPARE
002052  000000          RP4HDA: .WORD
002054  000000          RP4HDC: .WORD           ;RP04 DESIRED CYLINDER
002056  000000          RSHDA:  .WORD           ;RS04 DISK ADDRESS

                        .SBTTL  DEVICE HANDLER FUNCTION TABLE
                        ;*      THIS TABLE GETS LOADED BY THE DEVICE HANDLERS
                        ;*      AND THE DEVICE SERVICE ROUTINES. IT TELLS THE ROUTINES
                        ;*      WHICH FUNCTION TO DO NEXT.
002060  000000          RP3FUN: .WORD           ;RP03
002062  000000          RKFUN:  .WORD           ;RK05
```

```
002066  000000          RP4FUN: .WORD               ;RP04
002070  000000          RSFUN:  .WORD               ;RS04

                        .SBTTL  DEVICE HANDLER RETRY COUNT
                        ;*      THIS TABLE GETS LOADED BY THE DEVICE HANDLERS AND IS USED
                        ;*      BY THE DEVICE SERVICE ROUTINES. IF AN ERROR OCCURS
                        ;*      THE DEVICE SERVICE ROUTINE WILL RETRY THE FUNCTION UNTIL
                        ;*      THE BYTE IN THIS TABLE GOES TO ZERO. IT IS INITIALIZED
                        ;*      TO A -3.
002072     000          RP3TRY: .BYTE               ;RP03
002073     000          RKTRY:  .BYTE               ;RK05
002074     000                  .BYTE               ;SPARE
002075     000          RP4TRY: .BYTE               ;RP04
002076     000          RSTRY:  .BYTE               ;RS04
        002100                  .EVEN

                        .SBTTL  DEVICE REGISTER TABLES
                        ;*      THE FOLLOWING TABLES CONTAIN THE STANDARD ADDRESS FOR
                        ;*      THE DEVICES USED BY THIS PROGRAM. IF A DEVICE IS PLACED
                        ;*      AT A NON-STANDARD ADDRESS THE APPROPRIATE TABLE CAN BE
                        ;*      CHANGED AND THE PROGRAM WILL OPERATE THAT DEVICE.
                        ;*
                        ;*      EXCEPTION--SEE DOCUMENTATION FOR RP03 AND RP04 PROBLEMS.
                        .SBTTL  RP11/RP03 REGISTERS
002100  176710          RP3DS:  .WORD   176710  ;DRIVE STATUS
002102  176712          RP3ER:  .WORD   176712  ;ERROR REGISTER
002104  176714          RP3CS:  .WORD   176714  ;CONTROL AND STATUS
002106  176716          RP3WC:  .WORD   176716  ;WORD COUNT
002110  176720          RP3BA:  .WORD   176720  ;BUS ADDRESS
002112  176724          RP3DA:  .WORD   176724  ;DISK ADDRESS
002114  176722          RP3DC:  .WORD   176722  ;DESIRED CYLINDER
002116  000254          RP3VEC: .WORD   254     ;INTERRUPT VECTOR
002120  000256          RP3PSW: .WORD   256     ;INTERRUPT VECTOR+2

                        .SBTTL RK11/RK05 REGISTERS
002122  177400          RKDS:   .WORD   177400  ;DRIVE STATUS
002124  177402          RKER:   .WORD   177402  ;ERROR REGISTER
002126  177404          RKCS:   .WORD   177404  ;CONTROL AND STATUS
002130  177406          RKWC:   .WORD   177406  ;WORD COUNT
002132  177410          RKBA:   .WORD   177410  ;BUS ADDRESS
002134  177412          RKDA:   .WORD   177412  ;DISK ADDRESS
002136  000220          RKVEC:  .WORD   220     ;INTERRUPT VECTOR
002140  000222          RKPSW:  .WORD   222     ;INTERRUPT VECTOR+2

                        .SBTTL  RH70/RP04 REGISTERS
002142  176700          RP4CS1: .WORD   176700  ;CONTROL AND STATUS #1
002144  176702          RP4WC:  .WORD   176702  ;WORD COUNT
002146  176704          RP4BA:  .WORD   176704  ;BUS ADDRESS
002150  176750          RP4BAE: .WORD   176750  ;BUS ADDRESS EXTENDED
002152  176706          RP4DA:  .WORD   176706  ;DISK ADDRESS
002154  176710          RP4CS2: .WORD   176710  ;CONTROL AND STATUS #2
002156  176752          RP4CS3: .WORD   176752  ;CONTROL AND STATUS #3
002160  176712          RP4DS:  .WORD   176712  ;DRIVE STATUS
002162  176714          RP4ER1: .WORD   176714  ;ERROR REG #1
002164  176734          RP4DC:  .WORD   176734  ;DESIRED CYLINDER
```

```
002170  176742          RP4ER3: .WORD   176742  ;ERROR REG #3
002172  176736          RPCC:   .WORD   176736  ;CURRENT CYLINDER
002174  176732          RP4OF:  .WORD   176732  ;OFFSET REGISTER
002176  000254          RP4VEC: .WORD   254     ;INTERRUPT VECTOR
002200  000256          RP4PSW: .WORD   256     ;INTERRUPT VECTOR+2

                        .SBTTL  RH70/RS04 REGISTERS
002202  172040          RSCS1:  .WORD   172040  ;CONTROL AND STATUS #1
002204  172042          RSWC:   .WORD   172042  ;WORD COUNT
002206  172044          RSBA:   .WORD   172044  ;BUS ADDRESS
002210  172070          RSBAE:  .WORD   172070  ;BUS ADDRESS EXTENDED
002212  172046          RSDA:   .WORD   172046  ;DISK ADDRESS
002214  172050          RSCS2:  .WORD   172050  ;CONTROL AND STATUS #2
002216  172072          RSCS3:  .WORD   172072  ;CONTROL AND STATUS #3
002220  172052          RSDS:   .WORD   172052  ;DRIVE STATUS
002222  172054          RSER:   .WORD   172054  ;ERROR REG
002224  000204          RSVEC:  .WORD   204     ;INTERRUPT VECTOR
002226  000206          RSPSW:  .WORD   206     ;INTERRUPT VECTOR+2

                        .SBTTL  UNIBUS EXERCISER REGISTER ADDRESS TABLE
                        ;*      THIS TABLE IS ASSEMBLED FOR UBE #0. IF THE UBE
                        ;*      ADDRESSES ARE CUT FOR OTHER THAN UNIT #0, THE PROGRAM
                        ;*      WILL CHANGE THIS TABLE. THE PROGRAM LOOKS FOR A
                        ;*      UBE AT ADDRESSES 770002, 770022, 770032, AND 770042.
002230  170002          UBETBL: .WORD   UBECC   ;CYCLE COUNT
002232  170004                  .WORD   UBEBA   ;BUS ADDRESS REG
002234  170016                  .WORD   UBECR2  ;CONTROL REGISTER #2
002236  170006                  .WORD   UBECR1  ;CONTROL REGISTER #1
002240  170010                  .WORD   UBECLR  ;UBE CLEAR ADDRESS
002242  000510                  .WORD   UBEVEC  ;INTERRUPT VECTOR
002244  000512                  .WORD   UBEVEC+2        ;INTERRUPT VECTOR +2

                        .SBTTL  MASS BUS TESTER REGISTER ADDRESSES
                        ;*      THE PROGRAM IS ASSEMBLED WITH ADDRESSES FOR A MBT
                        ;*      AT 770100. IF THE MBT IS AT ANOTHER ADDRESS THE PROGRAM
                        ;*      WILL CHANGE THIS TABLE. THE PROGRAM LOOKS FOR A UBE
                        ;*      AT ADDRESSES 770100, 770200, 770300, AND 770400.
002246  160100          MBTTBL: .WORD   MBTCS1  ;CONTROL AND STATUS #1
002250  160102                  .WORD   MBTWC   ;WORD COUNT
002252  160104                  .WORD   MBTBA   ;BUS ADDRESS
002254  160174                  .WORD   MBTBAE  ;BUS ADDRESS EXTENDED
002256  160106                  .WORD   MBTMR2  ;MAINTENANCE REGISTER #2
002260  160110                  .WORD   MBTCS2  ;CONTROL REGISTER #2
002262  160112                  .WORD   MBTST   ;STATUS REGISTER
002264  160114                  .WORD   MBTER   ;ERROR REGISTER
002266  160176                  .WORD   MBTCS3  ;CONTROL REGISTER #3
002270  000774                  .WORD   MBTVEC  ;INTERRUPT VECTOR
002272  000776                  .WORD   MBTPSW  ;INTERRUPT VECTOR+2
002274  160126                  .WORD   MBTDT   ;DRIVE TYPE REGISTER
002276  160200          MBTN2:  .WORD   160200  ;MASS BUS TESTER #2
002300  160300          MBTN3:  .WORD   160300  ;MASS BUS TESTER #3
002302  160400          MBTN4:  .WORD   160400  ;MASS BUS TESTER #4
```

```
                                ;;*******************************************************************

                                .SBTTL  ERROR POINTER TABLE

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

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


        002304                  $ERRTB:
                                        ;ITEM 1
        002304  054774                  EM1                     ;UNEXPECTED TRAP TO 4
        002306  055021                  DH1                     ;PCOFTP  PHYSC         PSW  CPUERR
        002310  055066                  DT1                     ;VADR,VADR,$TMPO,$TMP2
        002312  055060                  DF1                     ;0,1,0,0,0
                                        ;ITEM 2
        002314  055100                  EM2                     ;UNEXPECTED TRAP TO 10
        002316  055126                  DH2                     ;PCOFTP  PHYSPC  PSW
        002320  055154                  DT2                     ;VADR,VADR,$TMPO
        002322  055060                  DF1
                                        ;ITEM 3
        002324  055164                  EM3                     ;UNEXPECTED TRAP TO 250(MGMT)
        002326  055221                  DH3                     ;PCOFTP  PHYSPC     PSW    MMR0     MMR2
        002330  055270                  DT3                     ;VADR,VADR,$TMPO,,$TMP2,,$TMP3
        002332  055060                  DF1
                                        ;ITEM 4
        002334  055304                  EM4                     ;UNEXPECTED TRAP TO 114
        002336  055333                  DH4                     ;PCOFTP  PHYSPC   PSW    ERADREG MEMERRREG
        002340  055270                  DT3                     ;VADR,VADR,$TMPO,$TMP3,$TMP2
        002342  055411                  DF4                     ;0,1,0,2,0
                                        ;ITEM 5
        002344  055416                  EM5                     ;PARITY ERROR DURING DATA CHECK
        002346  055455                  DH5                     ;SRCADR DSTADR ERRADREG  MEM ERR REG
        002350  055526                  DT5                     ;$TMPO,PA1500,$TMP3,$TMP2
        002352  055522                  DF5
                                        ;ITEM 6
        002354  055540                  EM6                     ;ERROR DURING CHECK OF RELOCATED DATA
        002356  055610                  DH6                     ;SRCADR  DSTADR
        002360  055630                  DT6                     ;$TMPO,PA1500
        002362  055411                  DF4
                                        ;ITEM 7
        002364  000000                  0
        002366  000000                  0
        002370  000000                  0
        002372  000000                  0
                                        ;ITEM 10
        002374  055636                  EM10                    ;ERROR DURING DATA CHECK-RELOC WAS BY I/O
        002376  055707                  DH10                    ;SRCADR   DSTADR   DEVICE THAT DID XFER
```

```
002402  055756                  DF10            ;0,1,3,0
                                ;ITEM 11
002404  055774                  EM11            ;BIT(S) STUCK IN MICRO-BREAK REG
002406  056041                  DH11            ;GOOD DAT  BAD DAT
002410  056064                  DT11            ;$TMP0,$TMP1
002412  056062                  DF11            ;0,0
                                ;ITEM 12
002414  056072                  EM12            ;UNIBUS EXERCISOR NON-EXISTANT MEMOREY
002416  056130                  DH12            ;PHYSICAL ADDRESS
002420  056146                  DT12            ;PA1500
002422  056144                  DF12            ;2
                                ;ITEM 13
002424  056152                  EM13            ;MASS BUS TESTER NON-EXISTANT MEMORY
002426  056210                  DH13            ;PHYSICAL ADDRESS
002430  056146                  DT12
002432  056144                  DF12
                                ;ITEM 14
002434  056225                  EM14            ;FLOATING POINT ERROR
002436  056252                  DH14            ;       DATA1            DATA2
002440  056272                  DT14            ;$TMP4,$REG2,$TMP6,$REG3
002442  056304                  DF14            ;4,0,4,0
                                ;ITEM 15
002444  056310                  EM15            ;DEVICE HUNG
002446  000000                  0
002450  000000                  0
002452  000000                  0
                                ;ITEM 16
002454  056225                  EM14            ;FLOATING POINT ERROR
002456  056324                  DH16
002460  056356                  DT16            ;FLTMP0,$REG2,FLTMP1,$REG3
```

```
002464  013737  177570  177570  START1: MOV     @#SWR,@#SWR
002472  000774                          BR      START1
                                        .SBTTL  PROGRAM INITIALIZATION

                                ;;*****************************************************************
                                ;.SBTTL  MICRO-BREAK REGISTER TEST
                                ;*THIS TEST IS EXECUTED BY STARTING THE PROGRAM AT ADDRESS 214.
                                ;*THIS TEST REQUIRES A MAINTENANCE CARD AND OPERATOR INTERVENTION.
                                ;*THE PROCESSOR SHOULD STOP 8 TIMES. FOLLOWING IS THE DATA
                                ;*THAT SHOULD BE IN THE MICRO-ADRESS DATA LIGHTS EACH TIME:
                                ;*              1               000
                                ;*              2               001
                                ;*              3               002
                                ;*              4               004
                                ;*              5               010
                                ;*              6               020
                                ;*              7               040
                                ;*              8               200
                                ;;*****************************************************************
002474  012706  001100          START2: MOV     #1100,SP        ;SETUP THE SP
002500  012737  051374  000034          MOV     #$TRAP,@#TRAPVEC        ;SETUP TRAP VECTOR
002506  012737  044774  000030          MOV     #$ERROR,@#EMTVEC;SETUP EMT VECTOR
002514  012700  000377                  MOV     #377,R0         ;PUT MICRO-BREAK DATA IN R0
002520  010037  177770                  MOV     R0,@#UBREAK     ;LOAD U BREAK REG
002524  020037  177770                  CMP     R0,@#UBREAK     ;LOAD OK?
002530  001024                          BNE     UBRERR  ;BRANCH IF NO
002532  005000                          CLR     R0
002534  010037  177770                  MOV     R0,@#UBREAK
002540  020037  177770                  CMP     R0,@#UBREAK
002544  001016                          BNE     UBRERR
002546  012700  000125                  MOV     #125,R0
002552  010037  177770                  MOV     R0,@#UBREAK
002556  020037  177770                  CMP     R0,@#UBREAK
002562  001007                          BNE     UBRERR
002564  012700  000252                  MOV     #252,R0
002570  010037  177770                  MOV     R0,@#UBREAK
002574  020037  177770                  CMP     R0,@#UBREAK
002600  001411                          BEQ     UBRK2
002602  010067  176470          UBRERR: MOV     R0,$TMP0
002606  013737  177770  001300          MOV     @#UBREAK,@#$TMP1
002614  012737  002474  001212          MOV     #START2,@#$LPERR
002622  104011                          ERROR   11
                                ;TEST TO ENSURE U BREAK COMPARATORS DO NOT COME ON.
002624  012737  000100  177770  UBRK2:  MOV     #100,@#UBREAK           ;PUT SAFE VALUE IN REG
002632  104400  002640                  TYPE    ,65$                    ;;TYPE ASCIZ STRING
002636  000421                          BR      64$                     ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /SET MAINT TO STOP ON MICRO-BREAK/<CRLF>
002702                          64$:
002702  104400  002710                  TYPE    ,67$                    ;;TYPE ASCIZ STRING
002706  000407                          BR      66$                     ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  /HIT CONTINUE/<CRLF>
002726                          66$:
002726  000000                          HALT
002730  012737  000012  000010          MOV     #12,@#RESVEC
002736  012737  000002  000012          MOV     #2,@#RESVEC+2
```

```
002750  012701  000010                   MOV     #10,R1             ;SET SOB COUNT
002754  012702  003161                   MOV     #UBRTBL+1,R2       ;GET ADRS OF UBREAK DATA TABLE
002760  112237  177770          1$:      MOVB    (R2)+,@#UBREAK     ;LOAD MICRO-BREAK FROM TABLE
002764  000010                           10                         ;EXEC RES INSTR (ROM ADRS 000)
002766  005037  177770                   CLR     @#UBREAK
002772  077106                           SOB     R1,1$              ;CONTINUE
002774  012737  000125  177770           MOV     #125,@#UBREAK      ;SET MICRO-BREAK DATA PATTERN
003002  006400                           MARK    0                  ;EXEC MARK (ROM ADRS 252)
003004  005037  177770          2$:      CLR     @#UBREAK
003010  012706  001100                   MOV     #1100,SP           ;RESTORE SP
003014  012737  000006  000004           MOV     #6,@#ERRVEC
003022  012737  000002  000006           MOV     #2,@#ERRVEC+2
003030  052737  040000  177776           BIS     #BIT14,@#PSW       ;GO TO SUPER MODE
003036  012706  000700                   MOV     #700,SP            ;SET SUPER SP
003042  012746  003064                   MOV     #3$,-(SP)          ;SETUP STACKK FOR JSR INSTR
003046  005000                           CLR     R0                 ;SETUP R0
003050  012701  000007                   MOV     #7,R1              ;SET SOB COUNT
003054  012702  003174                   MOV     #INSTBL+2,R2       ;GET ADRS OF TABLE OF INSTRUCTIONS
003060  012217                  4$:      MOV     (R2)+,(PC)         ;GET INSTRUCTION
003062  000000                           .WORD                      ;EXECUTE INSTRUCTION
003064  077103                  3$:      SOB     R1,4$              ;CONTINUE
003066  012737  000100  177770           MOV     #100,@#UBREAK      ;PUT SAFE VALUE IN UBREAK REG
003074  005000                           CLR     R0
003076  012702  003160                   MOV     #UBRTBL,R2
003102  012703  003172                   MOV     #INSTBL,R3
003106  012701  000010                   MOV     #10,R1
003112  012746  003126                   MOV     #5$,-(SP)
003116  112237  177770          6$:      MOVB    (R2)+,@#UBREAK     ;LOAD UBREAK REG FROM TABLE
003122  012317                           MOV     (R3)+,(PC)         ;GET INSTR FROM TABLE
003124  000000                           .WORD                      ;EXECUTE INSTR. PROCESSOR SHOULD STOP
                                                                    ;WITH THE CORRECT ROM ADR IN THE LIGHTS
003126  077105                  5$:      SOB     R1,6$              ;CONTINUE
003130  111237  177770                   MOVB    (R2),@#UBREAK      ;PUT SAFE VALUE IN UBREAK REG
003134  005037  177776                   CLR     @#PSW              ;GO BACK TO KERNEL MODE
003140  104400  003146                   TYPE    ,69$               ;;TYPE ASCIZ STRING
003144  000403                           BR      68$                ;;GET OVER THE ASCIZ
                                ;;69$:   .ASCIZ  /DONE/<CRLF>
003154                          68$:
003154  000000                           HALT
003156  000415                           BR      START
003160     000     001     002  UBRTBL:  .BYTE   0,1,2,4,10,20,40,200,100
003163     004     010     020
003166     040     200     100
        003172                           .EVEN
003172  000010  005010  005020  INSTBL:  .WORD   10,5010,5020,5040,0,5200,207,5010
003200  005040  000000  005200
```

```
003212  012706  001200          START:  MOV     #KERSTK,SP        ;SET KERNEL STACK PTR
003216  012737  076543  001524          MOV     #76543,@#$HINUM           ;INITIALIZE RANDOM NUM GEN
003224  012737  123456  001522          MOV     #123456,@#$LONUM

                                ;DETERMINE HOW PROGRAM WAS LOADED AND WHAT MODE (IF ACT11)
                                ;AND SET MEMORY PROTECTION.
003232  005037  001504                  CLR     @#QV                      ;SET NOT QV NOR AA MODE
003236  005027                          CLR     (PC)+                     ;SET NOT XXDP
003240     000                  XXDP:   .BYTE   0                         ;XXDP INDICATOR
003241     000                  XXDPC:  .BYTE   0                         ;XXDP CHAIN MODE INDICATOR
003242  005027                          CLR     (PC)+                     ;CLEAR MEMORY PROTECTION LIMIT
003244  000000                  PROT:   .WORD   0                         ;WILL CONTAIN MEM PROT LIMIT
003246  005737  036756                  TST     @#$ENDAD+4                ;BRANCH IF NOT QV
003252  100003                          BPL     1$
003254  110637  001504                  MOVB    SP,@#QV                   ;SET ACT11 QV MODE
003260  000411                          BR      3$

003262  001003                  1$:     BNE     2$
003264  110637  001505                  MOVB    SP,@#AA                   ;SET ACT11 AA MODE
003270  000405                          BR      3$

003272  005737  000042          2$:     TST     @#42                      ;BRANCH IF NOT IN CHAIN MODE
003276  001402                          BEQ     3$
003300  110637  003241                  MOVB    SP,@#XXDPC                ;SET CHAIN MODE INDICATOR

                                ;SET MEMORY PROTECTION LIMITS
003304  005737  001504          3$:     TST     @#QV                      ;BRANCH IF QV OR AA
003310  001006                          BNE     MEMSIZ
003312  005737  003240                  TST     @#XXDP                    ;BRANCH IF NOT VIA XXDP
003316  001403                          BEQ     MEMSIZ
003320  012737  005700  003244          MOV     #5700,@#PROT              ;PROTECT XXDP MONITOR
003326  012737  157776  001516  MEMSIZ: MOV     #157776,@#LSTMEM          ;SET VALUE INTO LSTMEM
003334  163737  003244  001516          SUB     @#PROT,@#LSTMEM           ;SET PROTECTION
003342  012737  056372  001514          MOV     #ENDTAG+2,@#FRSTMEM       ;SET FIRST RELOCATION ADDRESS

                                ;GET ADDRESS OF THE LAST MEMORY LOCATION ON THE SYSTEM
003350  005002                          CLR     R2
003352  013703  177760                  MOV     @#SIZELO,R3       ;GET ADDRESS OF SIZE REG LO
003356  073227  000006                  ASHC    #6,R2             ;SHIFT TO FORM ADDRESS
003362  052703  000077                  BIS     #77,R3            ;ENSURE LOWER SIX BITS SET
003366  062703  000001                  ADD     #1,R3
003372  005502                          ADC     R2
003374  010237  001560                  MOV     R2,@#MXMMHI       ;SAVE UPPER SIX BITS
003400  010337  001562                  MOV     R3,@#MXMMLO       ;SAVE LOWER 16 BITS

003404  012706  001200                  MOV     #KERSTK,SP                ;SET STACK PTR
003410  005037  001200                  CLR     @#$PASS                   ;CLEAR PASS COUNT
003414  105037  001503                  CLRB    @#MMON                    ;SET MEM MGMT ON IND=NOT ON
003420  012737  000600  001520          MOV     #600,@#NEXPAR             ;SET FIRST 'PAR' VALUE
003426  005737  003244                  TST     @#PROT
003432  001403                          BEQ     1$
003434  012737  001600  001520          MOV     #1600,@#NEXPAR
003442                          1$:
003442  012700  000027                  MOV     #27,R0                    ;SET SOB COUNT
003446  005001                          CLR     R1                        ;SETUP INDEX
```

F04

```
003454  062701  000002                     ADD     #2,R1
003460  077005                             SOB     RO,2$                   ;CONTINUE
003462  012737  177777  001656             MOV     #-1,@#MAPTBL            ;INITIALIZE MAP TABLE
003470  012737  177777  001660             MOV     #-1,@#MAPTBL+2
003476  012700  000010                     MOV     #10,RO                  ;SET SOB COUNT
003502  012701  001676                     MOV     #RP3HSTAT,R1            ;GET ADDRESS OF HANDLER STAT
003506  012721  000200             3$:     MOV     #200,(R1)+              ;INITIALIZE STATUS TABLE
003512  077003                             SOB     RO,3$                   ;CONTINUE
003514  012737  000060  001532             MOV     #60,@#SUBPASS           ;INIT SUBPASS TO ASCII O
003522  012700  047106                     MOV     #TIMEBUF,RO             ;GET ADR OF TIME BUFFER
003526  012701  000012                     MOV     #12,R1                  ;SET SOB COUNT
003532  112720  000060             4$:     MOVB    #60,(RO)+               ;INIT TIME BUFFER
003536  077103                             SOB     R1,4$
003540  105040                             CLRB    -(RO)                   ;INSERT TERMINATOR
003542  112737  000072  047111             MOVB    #72,@#TIMEBUF+3         ;INSERT COLON
003550  112737  000072  047114             MOVB    #72,@#TIMEBUF+6
003556  012737  000340  177776             MOV     #340,@#PS           ;;LOCK OUT ALL INTERRUPTS
003564  012706  001200                     MOV     #$CMTAG,R6          ;;FIRST LOCATION TO BE CLEARED
003570  005026                             CLR     (R6)+               ;;CLEAR MEMORY LOCATION
003572  022706  001240                     CMP     #STKS,R6            ;;DONE?
003576  001374                             BNE     .-6                 ;;LOOP BACK IF NO
003600  012706  001200                     MOV     #STACK,SP           ;;SETUP THE STACK POINTER
003604  012737  044534  000020             MOV     #$SCOPE,@#IOTVEC    ;;IOT VECTOR FOR SCOPE ROUTINE
003612  012737  000340  000022             MOV     #340,@#IOTVEC+2     ;;LEVEL 7
003620  012737  044774  000030             MOV     #$ERROR,@#EMTVEC    ;;EMT VECTOR FOR ERROR ROUTINE
003626  012737  000340  000032             MOV     #340,@#EMTVEC+2     ;;LEVEL 7
003634  012737  051374  000034             MOV     #$TRAP,@#TRAPVEC    ;;TRAP VECTOR FOR TRAP CALLS
003642  012737  000340  000036             MOV     #340,@#TRAPVEC+2;;LEVEL 7
003650  012737  051230  000024             MOV     #$PWRDN,@#PWRVEC    ;;POWER FAILURE VECTOR
003656  012737  000340  000026             MOV     #340,@#PWRVEC+2     ;;LEVEL 7
003664  016767  032726  032716             MOV     $ENDCT,$EOPCT       ;;SETUP END-OF-PROGRAM COUNTER
003672  005067  175420                     CLR     $TIMES              ;;INITIALIZE NUMBER OF ITERATIONS
003676  005067  175416                     CLR     $ESCAPE             ;;CLEAR THE ESCAPE ON ERROR ADDRESS
003702  112767  000001  175307             MOVB    #1,$ERMAX           ;;ALLOW ONE ERROR PER TEST
003710  012767  003710  175272             MOV     #.,$LPADR           ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
003716  012767  003716  175266             MOV     #.,$LPERR           ;;SETUP THE ERROR LOOP ADDRESS


                                   ;CLEAR PROGRAM INDICATORS
003724  052777  000100  175306             BIS     #100,@STKS              ;SET IE BIT IN KEYBOARD STATUS REG
003732  012737  052400  000060             MOV     #TKISR,@#TKVEC          ;SETUP KEYBOARD VECTOR
003740  012737  000200  000062             MOV     #PR4,@#TKVEC+2
003746  012737  052612  000064             MOV     #TPISR,@#TPVEC
003754  012737  000200  000066             MOV     #PR4,@#TPVEC+2
003762  005037  001466                     CLR     @#NOTYPE                ;CLEAR 'NO TYPING' INDICATOR

                                   ;THE BELOW ROUTINE ASCERTAINS WHICH CP & CP OPTIONS THE PROGRAM IS RUN-
                                   ;NING ON AND SETS AN INDICATOR IN OPT.CP ACCORDINGLY.
003766  012737  000006  000004     CPCHK:  MOV     #ERRVEC+2,@#ERRVEC      ;SET UP ERROR TRAP TO RETURN
003774  012737  000002  000006             MOV     #2,@#ERRVEC+2
004002  012737  000012  000010             MOV     #RESVEC+2,@#RESVEC      ;AND ALSO RESERVED INST TRAP
004010  012737  000002  000012             MOV     #2,@#RESVEC+2
004016  012702  144006                     MOV     #144006,R2              ;SET 11/70 NON-OPTION BITS
004022  000261                             SEC
004024  170500                             TSTF    RO                      ;WILL CLEAR CARRY IF 11/45 FLOATING POINT
004026  170000                             CFCC                            ;IS AVAIL. COPY FLOATING CC'S INTO PSW
```

```
004032  052702  020000           BIS     #FPOPT,R2               ;SET FP OPTION AVAIL INDICATOR
004036  000261           6$:     SEC
004040  005737  177546           TST     @#LKS                   ;BRANCH IF NO KW11-L
004044  103402                   BCS     7$
004046  052702  001000           BIS     #LKOPT,R2               ;SET OPTION INDICATOR
004052  000261           7$:     SEC
004054  005777  175164           TST     @$TPS                   ;BRANCH IF NO CONSOLE TTY
004060  103402                   BCS     9$
004062  052702  000400           BIS     #TTOPT,R2
004066  005003           9$:     CLR     R3
004070  000261                   SEC
004072  005737  170000           TST     @#UBEDB         ;IS UBE1 THERE?
004076  103410                   BCS     12$             ;BRANCH IF NO
004100  105037  170006           CLRB    @#UBECR1        ;IS THIS A TESTER OR EXERCISOR?
004104  105737  170006           TSTB    @#UBECR1
004110  100045                   BPL     15$             ;BRANCH IF TESTER
004112  052702  000200   16$:    BIS     #UBEOPT,R2      ;SET INDICATOR
004116  000425                   BR      17$
004120  000261           12$:    SEC
004122  005737  170020           TST     @#UBEDB+20              ;IS UBE2 THERE?
004126  103403                   BCS     13$             ;BRANCH IF NO
004130  012703  000020           MOV     #20,R3          ;SET OFFSET IN R3
004134  000766                   BR      16$
004136  000261           13$:    SEC
004140  005737  170040           TST     @#UBEDB+40      ;IS UBE3 THERE?
004144  103403                   BCS     14$             ;BRANCH IF NO
004146  012703  000040           MOV     #40,R3          ;PUT OFFSET IN R3
004152  000757                   BR      16$
004154  000261           14$:    SEC
004156  005737  170060           TST     @#UBEDB+60      ;IS UBE4 THERE?
004162  103420                   BCS     15$             ;BRANCH IF NO
004164  012703  000060           MOV     #60,R3          ;PUT OFFSET IN R3
004170  000750                   BR      16$
004172  005227  177777   17$:    INC     #-1
004176  001012                   BNE     15$
004200  012704  002230           MOV     #UBETBL,R4      ;GET ADDRESS OF UBE TABLE
004204  012705  000005           MOV     #5,R5           ;SET SOB COUNT
004210  060324           18$:    ADD     R3,(R4)+        ;ADJUST UBE TABLE ENTRIES
004212  077502                   SOB     R5,18$          ;CONTINUE
004214  006003                   ROR     R3
004216  006003                   ROR     R3              ;ADJUST OFFSET FOR UBE VECTOR
004220  060324                   ADD     R3,(R4)+        ;ADJUST UBEVEC ENTRY
004222  060314                   ADD     R3,(R4)         ;ADJUST UBEVEC PSW ENTRY
004224  005003           15$:    CLR     R3                      ;INIT R3
004226  000261                   SEC
004230  005777  176012           TST     @MBTTBL         ;IS MASS BUS TESTER THERE?
004234  103403                   BCS     20$             ;BRANCH IF NO
004236  052702  002000   21$:    BIS     #MBTOPT,R2      ;SET OPTION AVAILABLE
004242  000422                   BR      24$
004244  005777  176026   20$:    TST     @MBTN2          ;IS MBT2 THERE?
004250  103403                   BCS     22$             ;BRANCH IF NO
004252  012703  000100           MOV     #100,R3         ;SETUP R3
004256  000767                   BR      21$
004260  005777  176014   22$:    TST     @MBTN3          ;IS MBT3 THERE?
004264  103403                   BCS     23$             ;BRANCH IF NO
```

```
004272  000761                          BR      21$
004274  005777  176002          23$:    TST     @MBTN4          ;IS MBT4 THERE?
004300  103427                          BCS     30$             ;BRANCH IF NO
004302  012703  000300                  MOV     #300,R3
004306  000753                          BR      21$
004310  005227  177777          24$:    INC     #-1
004314  001021                          BNE     30$
004316  012704  002246                  MOV     #MBTTBL,R4      ;GET ADDRESS OF MBT TABLE
004322  012705  000011                  MOV     #11,R5          ;SET SOB COUNT
004326  060324                  25$:    ADD     R3,(R4)+        ;ADJUST MBT TABLE
004330  077502                          SOB     R5,25$          ;CONTINUE
004332  060337  002274                  ADD     R3,@#MBTTBL+26  ;ADJUST DRIVE TYPE ADDRESS
004336  112777  000007  175714          MOVB    #7,@MBTTBL+12   ;SET UNIT NUMBER
004344  122777  000040  175722          CMPB    #40,@MBTTBL+26  ;IS THIS REALLY A MBT?
004352  001402                          BEQ     30$             ;BRANCH IF YES
004354  042702  002000                  BIC     #MBTOPT,R2      ;CLEAR OPTION AVAILABLE BIT
004360  012737  053442  000004  30$:    MOV     #ERPRT,@#ERRVEC ;RESTORE ERROR TRAP
004366  012737  053370  000010          MOV     #RESERR,@#RESVEC;AND ALSO RESERVED INST TRAP
004374  010237  001470                  MOV     R2,@#OPT.CP     ;LOAD INDICATOR
004400  005227  177777                  INC     #-1             ;;FIRST TIME?
004404  001031                          BNE     64$             ;;BRANCH IF NO
004406  022737  036752  000042          CMP     #$ENDAD,@#42    ;;ACT-11?
004414  001425                          BEQ     64$             ;;BRANCH IF YES
004416  104400  004424                  TYPE    ,65$            ;;TYPE ASCIZ STRING
004422  000422                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF>"CEQKCC...PDP 11/70 CPU EXERCISOR"<CRLF>
004470                          64$:
                                ;;*******************************************************************
                                .SBTTL  SYSTEM SIZER
                                ;       THIS ROUTINE DETERMINES WHAT DRIVES ARE AVAILABLE ON
                                ;       THE FOLLOWING DEVICES: RK05, RP03, RP04, AND RS04. THE
                                ;       INFORMATION IS STORED IN THE TABLE "SYSSIZE" IN THE FOLLOWING FORMAT:
                                ;                       A. EACH DEVICE IS ASSIGNED A WORD
                                ;                       B. THE LOW BYTE OF THIS WORD INDICATES WHICH DRIVES ARE AVAILABLE
                                ;                       C. THE HIGH BYTE INDICATES WHICH DRIVES HAVE BEEN USED
                                ;                          BY THE RELOCATION ROUTINE.
                                ;;*******************************************************************
004470  012737  004602  000004  SIZE:   MOV     #21$,@#ERRVEC          ;SETUP TIMEOUT VECTOR
004476  005037  001276                  CLR     @#$TMPO                ;ENSURE $TMPO CLEAR
004502  005000                          CLR     R0                     ;USED TO SET THE UNIT AVAIL BITS
004504  012701  000010                  MOV     #10,R1                 ;SOB COUNT
004510  013777  001276  175416  9$:     MOV     @#$TMPO,@RKDA          ;SET UNIT NUMBER
004516  012777  000015  175402          MOV     #15,@RKCS              ;SEND DRIVE RESET
004524  032777  000200  175372          BIT     #BIT7,@RKER            ;NON EXISTANT DISK?
004532  001011                          BNE     7$                     ;BRANCH IF YES
004534  017702  175362                  MOV     @RKDS,R2               ;GET DRIVE STATUS
004540  042702  177537                  BIC     #177537,R2             ;GET BITS 5 & 7 ONLY
004544  022702  000200                  CMP     #200,R2                ;IS DRIVE READY?
004550  001002                          BNE     7$                     ;BRANCH IF NO
004552  052700  000400                  BIS     #BIT8,R0               ;SET UNIT AVAILABLE
004556  006000                  7$:     ROR     R0
004560  012777  000001  175340          MOV     #1,@RKCS               ;CLEAR THE ERRORS
004566  062737  020000  001276          ADD     #20000,@#$TMPO         ;SELECT NEXT UNIT
004574  077133                          SOB     R1,9$                  ;CONTINUE
004576  110037  001610                  MOVB    R0,@#SYSSIZE+2         ;STORE IN TABLE
```

```
                                        ;;*****************************************************
                                        ;THIS CODE DETERMINES IF THERE IS AN RP03 OR AN RP04 OR BOTH.
                                        ;IF BOTH ARE ON THE SYSTEM, THE OPERATOR MUST CHANGE THE RP04
                                        ;ADDRESSES IN THE TABLE IN "COMMON TAGS" AND "NOP" THE BRANCH
                                        ;AT "100$".

004602  012737  005042  000004  21$:    MOV     #11$,@#ERRVEC           ;SET THE ERROR VECTOR
004610  005737  176710                  TST     @#176710                ;IS THERE AN RP ON THE SYSTEM?
                                                                        ;STAY HERE IF YES
004614  012737  004630  000004          MOV     #1$,@#ERRVEC
004622  005777  175314                  TST     @RP4CS1         ;IS THERE AN RP04 ON SYSTEM?
004626  000441                  100$:   BR      10$                     ;BRANCH IF YES
                                        ;;*****************************************************


                                        ;;*****************************************************
004630  012737  004732  000004  1$:     MOV     #10$,@#ERRVEC           ;SETUP TIMEOUT VEC FOR RP03 TEST
004636  012737  000001  001276          MOV     #1,@#$TMP0              ;SETUP TEMP0
004644  005000                          CLR     R0                      ;USED TO SET UNIT AVAILABLE BITS
004646  012701  000010                  MOV     #10,R1                  ;SOB COUNT
004652  013777  001276  175224  3$:     MOV     @#$TMP0,@RP3CS          ;SET FUNCTION IDLE WITH UNIT NO
004660  005777  175220                  TST     @RP3CS                  ;WAS THERE AN ERROR?
004664  100006                          BPL     6$                      ;BRANCH IF NO
004666  006000                  4$:     ROR     R0                      ;UNIT NOT AVAILABLE
004670  062737  000400  001276          ADD     #400,@#$TMP0            ;SELECT NEXT UNIT
004676  077113                          SOB     R1,3$                   ;CONTINUE
004700  000412                          BR      5$
004702  017702  175172          6$:     MOV     @RP3DS,R2               ;GET STATUS REGISTER
004706  042702  136377                  BIC     #136377,R2              ;GET BITS 14, 9 & 8 ONLY
004712  022702  041400                  CMP     #41400,R2               ;IS DRIVE READY?
004716  001363                          BNE     4$                      ;BRANCH IF NO
004720  052700  000400                  BIS     #BIT8,R0                ;SET DRIVE AVAILABLE BIT
004724  000760                          BR      4$                      ;CONTINUE
004726  110037  001606          5$:     MOVB    R0,@#SYSSIZE            ;STORE IN TABLE

                                        ;;*****************************************************
004732  012737  005042  000004  10$:    MOV     #11$,@#ERRVEC           ;SETUP ERROR VEC FOR RP04 TEST
004740  005037  001276                  CLR     @#$TMP0
004744  005000                          CLR     R0                      ;UNIT AVAILABLE WORD
004746  012701  000010                  MOV     #10,R1                  ;SOB COUNT
004752  113777  001276  175174  14$:    MOVB    @#$TMP0,@RP4CS2         ;SET UNIT NUMBER
004760  012777  000021  175154          MOV     #21,@RP4CS1             ;TRY READ-IN-PRESET
004766  032777  010000  175160          BIT     #BIT12,@RP4CS2          ;NON EXISTANT DRIVE?
004774  001011                          BNE     12$                     ;BRANCH IF YES
004776  017702  175156                  MOV     @RP4DS,R2               ;GET DRIVE STATUS
005002  042702  163277                  BIC     #163277,R2              ;GET BITS 12, 11, 8, & 6 ONLY
005006  022702  010500                  CMP     #10500,R2               ;IS DRIVE READY?
005012  001002                          BNE     12$                     ;BRANCH IF NO
005014  052700  000400                  BIS     #BIT8,R0                ;SET UNIT AVAILABLE
005020  006000                  12$:    ROR     R0
005022  052777  000040  175124          BIS     #BIT5,@RP4CS2           ;CLEAR ERROR BITS
005030  005237  001276                  INC     @#$TMP0                 ;SELECT NEXT DRIVE
005034  077132                          SOB     R1,14$                  ;CONTINUE
005036  110037  001616                  MOVB    R0,@#SYSSIZE+10         ;STORE IN TABLE

                                        ;;*****************************************************
```

```
005050  005037  001276                   CLR     @#$TMP0
005054  005000                           CLR     R0
005056  012701  000010                   MOV     #10,R1                       ;SOB COUNT
005062  113777  001276  175124   18$:    MOVB    @#$TMP0,@RSCS2               ;SET UNIT NUMBER
005070  012777  000001  175104           MOV     #1,@RSCS1                    ;TRY NOP OPERATION
005076  032777  010000  175110           BIT     #BIT12,@RSCS2                ;NON EXISTANT DRIVE?
005104  001011                           BNE     16$                          ;BRANCH IF YES
005106  017702  175106                   MOV     @RSDS,R2                     ;GET DRIVE STATUS
005112  042702  163577                   BIC     #163577,R2                   ;GET BITS 12, 11, & 7 ONLY
005116  022702  010200                   CMP     #10200,R2                    ;IS DRIVE READY?
005122  001002                           BNE     16$                          ;BRANCH IF NO
005124  052700  000400                   BIS     #BIT8,R0                     ;SET DRIVE AVAILABLE BIT
005130  006000                   16$:    ROR     R0
005132  052777  000040  175054           BIS     #BIT5,@RSCS2                 ;CLEAR ANY ERROR BITS
005140  005237  001276                   INC     @#$TMP0                      ;SELECT NEXT UNIT
005144  077132                           SOB     R1,18$                       ;CONTINUE
005146  110037  001620                   MOVB    R0,@#SYSSIZE+12              ;STORE IN TABLE
                                 ;
                                 ;NEXT, DELETE XXDP UNIT 0 FROM TABLE
005152  122737  000002  000041   15$:    CMPB    #2,@#41                      ;RK?
005160  001004                           BNE     19$                          ;BRANCH IF NO
005162  042737  000001  001610           BIC     #BIT0,@#SYSSIZE+2            ;MAKE UNIT ZERO NOT AVAILABLE
005170  000420                           BR      20$
005172  113700  000041           19$:    MOVB    @#41,R0                      ;GET LOCATION 41
005176  042700  177770                   BIC     #177770,R0                   ;GET LEAST SIG 3 BITS
005202  000241                           CLC                                  ;ENSURE C CLEAR
005204  006100                           ROL     R0                           ;ADJUST
005206  122700  000002                   CMPB    #2,R0                        ;
005212  002404                           BLT     40$                          ;BRANCH IF NO
005214  042737  000001  001606           BIC     #BIT0,@#SYSSIZE
005222  000403                           BR      20$
005224  042760  000001  001612   40$:    BIC     #BIT0,SYSSIZE+4(R0)
005232  005227  177777           20$:    INC     #-1
005236  001055                           BNE     LOOP                 ;;BRANCH IF NOT FIRST TIME
005240  104400  054764                   TYPE    ,MSG25
005244  013746  001470                   MOV     @#OPT.CP,-(SP)
005250  104402                           TYPOC
005252  104400  001327                   TYPE    ,$CRLF
005256  005737  001504                   TST     @#QV                         ;ACT11?
005262  001043                           BNE     LOOP                         ;BRANCH IF YES
005264  105737  003241           50$:    TSTB    @#XXDPC                      ;XXDP CHAIN MODE?
005270  001040                           BNE     LOOP                 ;;BRANCH IF YES
005272  105737  001200                   TSTB    @#$PASS                      ;FIRST PASS?
005276  001035                           BNE     LOOP                 ;;BRANCH IF NO
005300  032737  000200  177570           BIT     #SW7,@#SWR                   ;INHIBIT SIZE TYPEOUT?
005306  001031                           BNE     LOOP                 ;;BRANCH IF YES
005310  004767  041604                   JSR     PC,TYPSIZ                    ;GO TYPE SYSTEM SIZE
005314  104400  005322                   TYPE    ,65$                 ;;TYPE ASCIZ STRING
005320  000417                           BR      64$                  ;;GET OVER THE ASCIZ
                                 ;;65$:   .ASCIZ  /TYPE A CHARACTER TO CONTINUE/<CRLF>
005360                           64$:
005360  005037  177776                   CLR     @#PSW
005364  000001                           WAIT
005366  000137  004470                   JMP     @#SIZE                       ;GO CHECK SYSTEM AGAIN
                                 ;PROGRAM RESTARTS HERE AFTER RELOCATION ABOVE 28K IS COMPLETE.
```

```
005372  012706  000700          LOOP:   MOV     #SUPSTK,SP                      ;SET THE STACK...WILL BE DIFFERENT
                                                                                ;THAN KERN STACK WHEN IN OUTER MODE
005376  012700  000004                  MOV     #ERRVEC,R0
005402  013701  177776                  MOV     @#PSW,R1                        ;GET CURRENT PSW
005406  012720  053442                  MOV     #ERPRT,(R0)+                    ;SET ERROR VEC
005412  052701  000340                  BIS     #PR7,R1                 ;SET PRIORITY 7 IN CURRENT PSW
005416  042701  000020                  BIC     #BIT4,R1                ;CLEAR T BIT
005422  010120                          MOV     R1,(R0)+
005424  012720  053370                  MOV     #RESERR,(R0)+                   ;SET RESERVED INST TRAP VECTOR
005430  010120                          MOV     R1,(R0)+
005432  012720  001472                  MOV     #$RTRN,(R0)+                    ;SET T BIT VEC
005436  042701  000340                  BIC     #PR7,R1
005442  005020                          CLR     (R0)+                           ;SET TBIT VEC+2
005444  005720                          TST     (R0)+                           ;BUMP R0 TO SCOPE VEC+2
005446  005020                          CLR     (R0)+                           ;SET SCOPE VEC+2
005450  062700  000006                  ADD     #6,R0                           ;SET R0 TO ERROR TRAP VEC
005454  012720  000340                  MOV     #PR7,(R0)+                      ;SET ERROR VEC
005460  005720                          TST     (R0)+
005462  012720  000340                  MOV     #PR7,(R0)+                      ;SET TRAP VEC+2
005466  012737  052642  000114          MOV     #.PARSRV,@#CACHVEC              ;SET PARITY ERROR VECTOR
005474  052701  000340                  BIS     #PR7,R1
005500  010137  000116                  MOV     R1,@#CACHVEC+2
005504  012737  053274  000250          MOV     #KTABRT,@#MMVEC                 ;SET KT11 ABORT VECTOR
005512  010137  000252                  MOV     R1,@#MMVEC+2
005516  042737  000340  177776          BIC     #PR7,@#PSW
                                ;;**************************************************
                                ;*TEST 1                MEMORY VERIFICATION TEST
                                ;;**************************************************
005524  112737  000001  001202  TST1:   MOVB    #1,@#$TSTNM                     ;LOAD TEST NUMBER
005532  012767  000001  173556          MOV     #1,$TIMES               ;;DO 1 ITERATION
005540  000004                          SCOPE

                                        .SBTTL  START OF SECTION 0

                                ;000000000000 FIRST ADDRESS TO BE RELOCATED 000000000
005542  010700                  RELO:   MOV     PC,R0                   ;GET PC
005544  005740                          TST     -(R0)                   ;R0 CONTAINS THE ADDRESS OF RELO
005546  010037  001512                  MOV     R0,@#FRSTAD             ;SAVE
005552  010700                          MOV     PC,R0                   ;GET CURRENT PC
005554  162700  005554                  SUB     #.,R0                   ;SUBTRACT RELOCATION FACTOR
005560  010037  001506                  MOV     R0,@#FACTOR             ;SAVE RELOCATION FACTOR
005564  010737  001212                  MOV     PC,@#$LPERR             ;SET LOOP ADDRESS
005570  062737  000030  001212          ADD     #30,@#$LPERR            ;ADJUST
005576  013737  001212  001210          MOV     @#$LPERR,@#$LPADR
005604  105737  001502                  TSTB    @#NEXEC                 ;BR IF TEST CODE TO BE EXECUTED
005610  001402                          BEQ     .+6
005612  000167  000720                  JMP     RELE0

                                ;MEMORY AND DISK (IF SELECTED) VERIFICATION TEST.
005616  000167  000714                  JMP     1$
005622  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
005630  177777  000000  000000
005636  000000  000000
005642  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
005650  177777  000000  000000
005656  000000  000000
005662  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
```

```
  1477  005676  000000  000000
  1478  005702  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1479  005710  177777  000000  000000
  1480  005716  000000  000000
  1481  005722  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1482  005730  177777  000000  000000
  1483  005736  000000  000000
  1484  005742  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1485  005750  177777  000000  000000
  1486  005756  000000  000000
  1487  005762  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1488  005770  177777  000000  000000
  1489  005776  000000  000000
  1490  006002  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1491  006010  177777  000000  000000
  1492  006016  000000  000000
  1493  006022  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1494  006030  177777  000000  000000
  1495  006036  000000  000000
  1496  006042  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1497  006050  177777  000000  000000
  1498  006056  000000  000000
  1499  006062  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1500  006070  177777  000000  000000
  1501  006076  000000  000000
  1502  006102  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1503  006110  177777  000000  000000
  1504  006116  000000  000000
  1505  006122  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1506  006130  177777  000000  000000
  1507  006136  000000  000000
  1508  006142  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1509  006150  177777  000000  000000
  1510  006156  000000  000000
  1511  006162  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1512  006170  177777  000000  000000
  1513  006176  000000  000000
  1514  006202  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1515  006210  177777  000000  000000
  1516  006216  000000  000000
  1517  006222  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1518  006230  177777  000000  000000
  1519  006236  000000  000000
  1520  006242  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1521  006250  177777  000000  000000
  1522  006256  000000  000000
  1523  006262  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1524  006270  177777  000000  000000
  1525  006276  000000  000000
  1526  006302  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1527  006310  177777  000000  000000
  1528  006316  000000  000000
  1529  006322  177777  177777  177777            .WORD   -1,-1,-1,-1,0,0,0,0
  1530  006330  177777  000000  000000
  1531  006336  000000  000000
```

M04

```
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 32
CEQKCC.P11     03-MAR-78 13:13              START OF SECTION 0                                                    SEQ 0051

006350  177777  000000  000000
006356  000000  000000
006362  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006370  177777  000000  000000
006376  000000  000000
006402  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006410  177777  000000  000000
006416  000000  000000
006422  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006430  177777  000000  000000
006436  000000  000000
006442  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006450  177777  000000  000000
006456  000000  000000
006462  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006470  177777  000000  000000
006476  000000  000000
006502  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006510  177777  000000  000000
006516  000000  000000
006522  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0
006530  177777  000000  000000
006536                          1$:
006536  000004                  RELE0:  SCOPE
006540  010702                          MOV     PC,R2
006542  062702  000012                  ADD     #12,R2
006546  012707  034242                  MOV     #RELOC,PC               ;GO RELOCATE PROGRAM CODE
006552  000000                  RELO0:  .WORD   0
                                ;000000000000 LAST ADDRESS OF CODE TO BE RELOCATED 00000000000

                                ;;*****************************************************************
                                ;*TEST 2         CHECK BRANCH INSTRUCTIONS
                                ;;*****************************************************************
006554  112737  000002  001202  TST2:   MOVB    #2,@#$TSTNM             ;LOAD TEST NUMBER
006562  012767  000001  172526          MOV     #1,$TIMES               ;;DO 1 ITERATION
006570  000004                          SCOPE

                                        .SBTTL  START OF SECTION 1
                                ;1111111111111 FIRST ADDRESS TO BE RELOCATED 111111111
006572  010700                  REL1:   MOV     PC,R0                   ;GET PC
006574  005740                          TST     -(R0)                   ;R0 CONTAINS THE ADDRESS OF REL1
006576  010037  001512                  MOV     R0,@#FRSTAD             ;SAVE
006602  010700                          MOV     PC,R0                   ;GET CURRENT PC
006604  162700  006604                  SUB     #.,R0                   ;SUBTRACT RELOCATION FACTOR
006610  010037  001506                  MOV     R0,@#FACTOR             ;SAVE RELOCATION FACTOR
006614  010737  001212                  MOV     PC,@#$LPERR             ;SET LOOP ADDRESS
006620  062737  000030  001212          ADD     #30,@#$LPERR            ;ADJUST
006626  013737  001212  001210          MOV     @#$LPERR,@#$LPADR
006634  105737  001502                  TSTB    @#NEXEC                 ;BR IF TEST CODE TO BE EXECUTED
006640  001402                          BEQ     .+6
006642  000167  004052                  JMP     RELE1

006646  000257                  ;       CCC                             ;CC'S=0000
006650  103407                          BCS     CC0                     ;SAME AS BLO
006652  102406                          BVS     CC0
```

```
  1589 006656  100404                      BMI     CC0
  1590 006660  002403                      BLT     CC0
  1591 006662  003402                      BLE     CC0
  1592 006664  101401                      BLOS    CC0
  1593 006666  101001                      BHI     .+4
  1594 006670  104000              CC0:    HLT                     ;ONE OF THE ABOVE BRANCHES FAILED
  1595
  1596                             ;CONTINUE
  1597 006672  000270                      SEN                     ;CC'S=1000
  1598 006674  100003                      BPL     CC1
  1599 006676  002002                      BGE     CC1
  1600 006700  003001                      BGT     CC1
  1601 006702  002401                      BLT     .+4
  1602 006704  104000              CC1:    HLT                     ;ONE OF THE ABOVE BRANCHES FAILED
  1603
  1604                             ;CONTINUE
  1605 006706  000262                      SEV                     ;CC'S=1010
  1606 006710  102003                      BVC     CC2
  1607 006712  002402                      BLT     CC2
  1608 006714  003401                      BLE     CC2
  1609 006716  002001                      BGE     .+4
```

```


                                    ;CONTINUE
006722  000261                              SEC                     ;CC'S=1011
006724  103002                              BCC     CC3
006726  101001                              BHI     CC3
006730  003001                              BGT     .+4
006732  104000                      CC3:    HLT                     ;ERROR! ONE OF THE ABOVE BRANCHES FAILED

                                    ;CONTINUE
006734  000264                              SEZ                     ;CC'S=1111
006736  001003                              BNE     CC4
006740  003002                              BGT     CC4
006742  101001                              BHI     CC4
006744  003401                              BLE     .+4
006746  104000                      CC4:    HLT                     ;ERROR! ONE OF THE ABOVE BRANCHES FAILED
                                    ;;*****************************************************************
                                    ;*TEST 3        TEST UNIARY CONDITION CODES
                                    ;;*****************************************************************
006750  112737  000003  001202      TST3:   MOVB    #3,@#$TSTNM             ;LOAD TEST NUMBER
006756  000004                              SCOPE
                                    ;CLR    R0
006760  000277                              SCC
006762  000244                              CLZ
006764  005000                              CLR     R0              ;R0=0,CC'S=0100
006766  103404                              BCS     CLR0
006770  102403                              BVS     CLR0
006772  001002                              BNE     CLR0
006774  100401                              BMI     CLR0
006776  003401                              BLE     .+4
007000  104000                      CLR0:   HLT                     ;ERROR! INCORRECT CC'S AFTER CLR

007002  000277                              SCC
007004  000244                              CLZ
007006  005700                              TST     R0              ;R0=0,CC'S=0100
007010  103404                              BCS     TST0
007012  102403                              BVS     TST0
007014  001002                              BNE     TST0
007016  100401                              BMI     TST0
007020  101401                              BLOS    .+4
007022  104000                      TST0:   HLT                     ;ERROR! INCORRECT CC'S AFTER TST

007024  000257                              CCC
007026  000266                              +SEZ!SEV
007030  005100                              COM     R0              ;R0=-1,CC'S=1001
007032  103004                              BCC     COM0
007034  102403                              BVS     COM0
007036  001402                              BEQ     COM0
007040  100001                              BPL     COM0
007042  002401                              BLT     .+4
007044  104000                      COM0:   HLT                     ;ERROR! INCORRECT CC'S AFTER COM

007046  000261                              SEC
007050  005500                              ADC     R0              ;R0=000000,CC'S=0101
007052  103003                              BCC     ADC0
007054  102402                              BVS     ADC0
```

```
007060  002001                  BGE     .+4
007062  104000          ADCO:   HLT                             ;ERROR! INCORRECT CC'S AFTER ADC

007064  000261                  SEC
007066  006000                  ROR     R0                      ;R0=100000,CC'S=1010
007070  103404                  BCS     RORO
007072  102003                  BVC     RORO
007074  001402                  BEQ     RORO
007076  100001                  BPL     RORO
007100  003001                  BGT     .+4
007102  104000          RORO:   HLT                             ;ERROR! INCORRECT CC'S AFTER ROR
007104  000277                  SCC
007106  000242                  CLV
007110  005300                  DEC     R0                      ;R0=077777,CC'S=0011
007112  103004                  BCC     DECO
007114  102003                  BVC     DECO
007116  001402                  BEQ     DECO
007120  100401                  BMI     DECO
007122  003401                  BLE     .+4
007124  104000          DECO:   HLT                             ;ERROR! INCORRECT CC'S AFTER DEC

007126  000257                  CCC
007130  005200                  INC     R0                      ;R0=100000,CC'S=1010
007132  103404                  BCS     INCO
007134  102003                  BVC     INCO
007136  001402                  BEQ     INCO
007140  100001                  BPL     INCO
007142  003001                  BGT     .+4
007144  104000          INCO:   HLT                             ;ERROR! INCORRECT CC'S AFTER INC

007146  000277                  SCC
007150  000242                  CLV
007152  005400                  NEG     R0                      ;R0=100000,CC'S=1011
007154  103003                  BCC     NEGO
007156  102002                  BVC     NEGO
007160  001401                  BEQ     NEGO
007162  002001                  BGE     .+4
007164  104000          NEGO:   HLT                             ;ERROR! INCORRECT CC'S AFTER NEG

007166  000261                  SEC
007170  006300                  ASL     R0                      ;R0=000000,CC'S=0111
007172  103004                  BCC     ASLO
007174  102003                  BVC     ASLO
007176  001002                  BNE     ASLO
007200  100401                  BMI     ASLO
007202  101401                  BLOS    .+4
007204  104000          ASLO:   HLT                             ;ERROR! INCORRECT CC'S AFTER ASL

007206  006100                  ROL     R0                      ;R0=000001,CC'S=0000
007210  103402                  BCS     ROLO
007212  003401                  BLE     ROLO
007214  002001                  BGE     .+4
007216  104000          ROLO:   HLT                             ;ERROR! INCORRECT CC'S AFTER ROL

007220  006200                  ASR     R0                      ;R0=000000,CC'S=0111
```

```
007224  102002                          BVC     ASR0
007226  001001                          BNE     ASR0
007230  002401                          BLT     .+4
007232  104000                  ASR0:   HLT                     ;ERROR! INCORRECT CC'S AFTER ASR

007234  000277                          SCC
007236  005600                          SBC     R0              ;R0=-1,CC'S=1001
007240  103002                          BCC     SBC0
007242  102401                          BVS     SBC0
007244  003401                          BLE     .+4
007246  104000                  SBC0:   HLT                     ;ERROR! INCORRECT CC'S AFTER SBC

007250  005400                          NEG     R0              ;R0=000001,CC'S=00001
007252  000300                          SWAB    R0              ;R0=000400 ,CC'S=0100
007254  103403                          BCS     SWAB0
007256  102402                          BVS     SWAB0
007260  001001                          BNE     SWAB0
007262  002001                          BGE     .+4
007264  104000                  SWAB0:  HLT                     ;ERROR! INCORRECT CC'S AFTER SWAB
                                ;;*************************************************************
                                ;*TEST 4        CHECK REGISTER SELECTION
                                ;;*************************************************************
007266  112737  000004  001202  TST4:   MOVB    #4,@#$TSTNM             ;LOAD TEST NUMBER
007274  000004                          SCOPE
007276  012737  000005  001316          MOV     #5,@#$TIMES     ;SET ITERATION COUNT TO 5
007304  005000                          CLR     R0
007306  000277                          SCC
007310  006100                          ROL     R0              ;R0=1
007312  010002                          MOV     R0,R2
007314  006302                          ASL     R2              ;R2=2
007316  010203                          MOV     R2,R3
007320  006303                          ASL     R3              ;R3=4
007322  010304                          MOV     R3,R4
007324  006304                          ASL     R4              ;R4=10
007326  010405                          MOV     R4,R5
007330  006305                          ASL     R5              ;R5=20
007332  010546                          MOV     R5,-(SP)        ;SET BITS SET IN REGISTERS
007334  050416                          BIS     R4,(SP)         ;INTO STACK ADDRESS
007336  050316                          BIS     R3,(SP)
007340  050216                          BIS     R2,(SP)
007342  050016                          BIS     R0,(SP)
007344  022726  000037                  CMP     #37,(SP)+
007350  001401                          BEQ     .+4             ;WERE SET
007352  104000                          HLT                     ;MISSING BIT(S) REPRESENT
                                                                ;INCORRECT REGISTER SELECTION

                                ;CHECK THAT ALL BITS CAN BE SET & CLEARED IN ALL REGISTERS
007354  000257                          CCC
007356  112700  000377                  MOVB    #377,R0         ;SET ALL BITS (MOVB EXTENDS SIGN)
007362  006100                  1$:     ROL     R0              ;ROTATE A 0 THROUGH ALL BIT
007364  103776                          BCS     1$              ;POSITIONS
007366  005200                          INC     R0              ;FINAL RESULT IS -1
007370  001401                          BEQ     .+4
007372  104000                          HLT                     ;ERROR!
```

```
007400  005002                          CLR     R2
007402  000261                  2$:     SEC
007404  006002                          ROR     R2                      ;ROTATE 1 THROUGH ALL BIT POSITS
007406  005300                          DEC     R0                      ;DECREMENT SHIFT COUNT
007410  001374                          BNE     2$
007412  005102                          COM     R2                      ;R2 SHOULD CONTAIN -1
007414  001401                          BEQ     .+4
007416  104000                          HLT                             ;ERROR! CHECK R2 SHOULD = 0

007420  012703  100000                  MOV     #100000,R3
007424  006203                  3$:     ASR     R3                      ;EXTEND 1 BIT THROUGH ALL POSITIONS
007426  103376                          BCC     3$
007430  005203                          INC     R3
007432  001401                          BEQ     .+4
007434  104000                          HLT                             ;ERROR!

007436  112704  177401                  MOVB    #177401,R4              ;R4=1
007442  060404                  4$:     ADD     R4,R4                   ;HAS THE AFFECT OF SHIFTING A BIT
007444  103376                          BCC     4$                      ;THROUGH ALL POSITIONS
007446  005704                          TST     R4                      ;RESULT SHOULD BE 0
007450  001401                          BEQ     .+4
007452  104000                          HLT

007454  012705  000001                  MOV     #1,R5
007460  006305                  5$:     ASL     R5
007462  102376                          BVC     5$
007464  006305                          ASL     R5
007466  103002                          BCC     6$
007470  005705                          TST     R5
007472  001401                          BEQ     .+4
007474  104000                  6$:     HLT


                                ;CHECK REGISTER VOLITILITY
007476  005002                          CLR     R2
007500  005102                          COM     R2                      ;R2=-1
007502  010203                          MOV     R2,R3
007504  000257                          CCC
007506  006002                          ROR     R2                      ;R2=LOOP COUNT
007510  006202                          ASR     R2
007512  010304                  7$:     MOV     R3,R4
007514  005302                          DEC     R2                      ;DECREMENT LOOP COUNT
007516  001375                          BNE     7$
007520  005203                          INC     R3                      ;CHECK R3
007522  001002                          BNE     8$
007524  005204                          INC     R4                      ;CHECK R4
007526  001401                          BEQ     .+4
007530  104000                  8$:     HLT

                                ;CHECK TRANSFER OF REGISTER DATA BETWEEN THE GS AND GD REGISTERS
007532  032737  000020  177776  GSTST:  BIT     #20,@#PSW               ;CHECK IF 'T' BIT IS SET
007540  001050                          BNE     7$                      ;SKIP TEST IF 'T' BIT SET
007542  010627                          MOV     SP,(PC)+                ;SAVE STACK PTR
007544  000000                  1$:     .WORD   0                       ;CONTAINS SAVED STACK PTR
007546  010727                          MOV     PC,(PC)+                ;LOAD DATA. THE CURRENT PC IS USED AS
007550  000000                  2$:     .WORD   0                       ;DATA. IF THIS TEST FAILS 2$ CON-
```

```
007552  005267  177772                     INC     2$                  ;MAKE ODD TO CHECK BIT 0
007556  016700  177766             3$:     MOV     2$,R0               ;LOAD GD REGISTER 0
007562  010001                             MOV     R0,R1               ;TRANSFER GS REG 0 TO GD REG 1
007564  010102                             MOV     R1,R2               ;AND GS REG 1 TO GD REG 2
007566  010203                             MOV     R2,R3               ;ETC...
007570  010304                             MOV     R3,R4
007572  010405                             MOV     R4,R5
007574  152737  000340  177776             BISB    #340,@#PSW          ;SET PRIORITY LEVEL 7
007602  010506                             MOV     R5,SP               ;TRANSFER GS REG 5 TO GD STK PTR
007604  010627                             MOV     SP,(PC)+            ;TRANSFER GS STK PTR TO MEMORY
007606  000000                     4$:     .WORD   0                   ;CONTAINS GS STACK PTR
007610  016706  177730                     MOV     1$,SP               ;RESTORE STK PTR NEEDED FOR HLT/SCOPE
007614  142737  000340  177776             BICB    #340,@#PSW          ;SET PRIORITY LEVEL 0
007622  026700  177760                     CMP     4$,R0               ;COMPARE GS/GD STACK WITH GS REG 0
007626  001004                             BNE     5$                  ;BRANCH IF THEY WERE NOT =
007630  006367  177714                     ASL     2$                  ;SHIFT TEST DATA UNTIL = 000000
007634  001350                             BNE     3$
007636  000411                             BR      6$
007640  010046                     5$:     MOV     R0,-(SP)            ;GET GS REG 0
007642  010146                             MOV     R1,-(SP)            ;ETC...
007644  010246                             MOV     R2,-(SP)
007646  010346                             MOV     R3,-(SP)
007650  010446                             MOV     R4,-(SP)
007652  010546                             MOV     R5,-(SP)
007654  104000                             HLT                         ;ERROR! DATA IN GS STK PTR NOT = GS REG 0
                                                                       ;GS REG 0-GS REG 5 ARE ON THE STACK
007656  016706  177662                     MOV     1$,SP               ;RESTORE STACK PTR
007662                             6$:
007662                             7$:
                                   ;;*****************************************************************
                                   ;;*TEST 5          TEST UNIARY WORD INSTRUCTIONS USING ADDRESS MODE 1
                                   ;;*****************************************************************
007662  112737  000005  001202     TST5:   MOVB    #5,@#$TSTNM                  ;LOAD TEST NUMBER
007670  000004                             SCOPE
007672  012737  000005  001316             MOV     #5,@#$TIMES
007700  000401                             BR      .+4
007702  000000                             .WORD   0                   ;RESERVE ADDRESS FOR TESTS
007704  010702                             MOV     PC,R2
007706  162702  000004                     SUB     #4,R2               ;R2 POINTS TO RESERVED WORD
007712  005012                             CLR     (R2)                ;PRESET (R2)

007714  000261                             SEC
007716  006012                             ROR     (R2)                ;(R2)=100000,CC=1010
007720  101402                             BLOS    ROR1
007722  100001                             BPL     ROR1
007724  002001                             BGE     .+4
007726  104000                     ROR1:   HLT                         ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

007730  000257                             CCC
007732  000261                             SEC
007734  005312                             DEC     (R2)                ;(R2)=077777,CC=0011
007736  103001                             BCC     DEC1
007740  003401                             BLE     .+4
007742  104000                     DEC1:   HLT                         ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
```

```
007746  000261                  SEC
007750  005512                  ADC     (R2)            ;(R2)=100000,CC=1010
007752  103403                  BCS     ADC1
007754  102002                  BVC     ADC1
007756  100001                  BPL     ADC1
007760  001001                  BNE     .+4
007762  104000          ADC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

007764  006112                  ROL     (R2)            ;(R2)=000000,CC=0111
007766  103003                  BCC     ROL1
007770  102002                  BVC     ROL1
007772  001001                  BNE     ROL1
007774  100001                  BPL     .+4
007776  104000          ROL1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010000  006112                  ROL     (R2)            ;(R2)=000001,CC=0000
010002  101402                  BLOS    ROL1A           ;BRANCH IF C OR Z IS SET
010004  102401                  BVS     ROL1A
010006  100001                  BPL     .+4
010010  104000          ROL1A:  HLT

010012  006212                  ASR     (R2)            ;(R2)=000000,CC=0111
010014  103003                  BCC     ASR1
010016  102002                  BVC     ASR1
010020  001001                  BNE     ASR1
010022  100001                  BPL     .+4
010024  104000          ASR1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010026  006012                  ROR     (R2)            ;(R2)=100000,CC=1010
010030  103403                  BCS     ROR1A
010032  102002                  BVC     ROR1A
010034  001401                  BEQ     ROR1A
010036  100401                  BMI     .+4
010040  104000          ROR1A:  HLT

010042  000261                  SEC
010044  005212                  INC     (R2)            ;(R2)=100001,CC=1001
010046  103003                  BCC     INC1
010050  102402                  BVS     INC1
010052  001401                  BEQ     INC1
010054  100401                  BMI     .+4
010056  104000          INC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010060  005612                  SBC     (R2)            ;(R2)=100000,CC=1000
010062  103403                  BCS     SBC1
010064  102402                  BVS     SBC1
010066  001401                  BEQ     SBC1
010070  100401                  BMI     .+4
010072  104000          SBC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010074  000261                  SEC
010076  005612                  SBC     (R2)            ;(R2)=077777,CC=0010
010100  103403                  BCS     SBC1A
010102  102002                  BVC     SBC1A
010104  001401                  BEQ     SBC1A
```

```
010110  104000                      SBC1A:  HLT                     ;ERROR! INCORRECT CC'S AS SHOEN ABOVE

010112  000261                              SEC
010114  005512                              ADC     (R2)            ;(R2)=100000,CC=1010
010116  100401                              BMI     .+4
010120  104000                              HLT

010122  000261                              SEC
010124  006312                              ASL     (R2)            ;(R2)=000000,CC=0111
010126  103003                              BCC     ASL1
010130  102002                              BVC     ASL1
010132  001001                              BNE     ASL1
010134  100001                              BPL     .+4
010136  104000                      ASL1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010140  005112                              COM     (R2)            ;(R2)=177777,CC=1001
010142  103002                              BCC     COM1
010144  102401                              BVS     COM1
010146  100401                              BMI     .+4
010150  104000                      COM1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010152  000250                              CLN
010154  005712                              TST     (R2)            ;(R2)=177777,CC=1000
010156  103403                              BCS     TEST1
010160  102402                              BVS     TEST1
010162  100001                              BPL     TEST1
010164  001001                              BNE     .+4
010166  104000                      TEST1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010170  000262                              SEV
010172  005412                              NEG     (R2)            ;(R2)=000001,CC=0000
010174  103002                              BCC     NEG1
010176  102401                              BVS     NEG1
010200  001001                              BNE     .+4
010202  104000                      NEG1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010204  005312                              DEC     (R2)            ;(R2)=000000,CC=0101
010206  103001                              BCC     DEC1A
010210  001401                              BEQ     .+4
010212  104000                      DEC1A:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
                                    ;;*****************************************************************
                                    ;*TEST 6         CHECK UNIARY BYTE INSTRUCTIONS USING ADDRESS MODE 1
                                    ;;*****************************************************************
010214  112737  000006  001202      TST6:   MOVB    #6,@#$TSTNM             ;LOAD TEST NUMBER
010222  000004                              SCOPE
010224  000401                              BR      .+4             ;RESERVE A WORD
010226  000000                              .WORD   0               ;ADDRESS RESERVED FOR TESTS
010230  010703                              MOV     PC,R3
010232  162703  000004                      SUB     #4,R3           ;R3 POINTS TO EVEN BYTE OF WORD
010236  010304                              MOV     R3,R4           ;R4 POINTS TO ODD BYTE OF WORD
010240  005204                              INC     R4
010242  005013                              CLR     (R3)            ;PRESET DATA

010244  000261                      1$:     SEC
010246  105513                              ADCB    (R3)            ;ADD CARRY TO EVEN BYTE
```

```
010252  105214                  INCB    (R4)            ;INCREMENT ODD BYTE
010254  000773                  BR      1$
010256  102401          2$:     BVS     .+4             ;(R3)=077600=[0774][200],CC=1010
010260  104000                  HLT
010262  000242                  CLV
010264  105214                  INCB    (R4)            ;(R3)=100200=[1000][200],CC=1010
010266  103402                  BCS     INCB1
010270  102001                  BVC     INCB1
010272  100401                  BMI     .+4
010274  104000          INCB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010276  106114                  ROLB    (R4)            ;(R3)=000200=[0000][200],CC=0111
010300  103002                  BCC     ROLB1
010302  102001                  BVC     ROLB1
010304  001401                  BEQ     .+4
010306  104000          ROLB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010310  105614                  SBCB    (R4)            ;(R3)=177600=[1774][200], CC=1001
010312  103002                  BCC     SBCB1
010314  102401                  BVS     SBCB1
010316  100401                  BMI     .+4
010320  104000          SBCB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010322  106313                  ASLB    (R3)            ;(R3)=177400,CC=0111
010324  103002                  BCC     ASLB1
010326  102001                  BVC     ASLB1
010330  001401                  BEQ     .+4
010332  104000          ASLB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010334  105413                  NEGB    (R3)            ;(R3)=177400,CC=0100
010336  103402                  BCS     NEGB1
010340  102401                  BVS     NEGB1
010342  001401                  BEQ     .+4
010344  104000          NEGB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010346  000277                  SCC
010350  105313                  DECB    (R3)            ;(R3)=177777,CC=1001
010352  103002                  BCC     DECB1
010354  102401                  BVS     DECB1
010356  001001                  BNE     .+4
010360  104000          DECB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010362  000241                  CLC
010364  106013                  RORB    (R3)            ;(R3)=177577,CC=0011
010366  103002                  BCC     RORB1
010370  102001                  BVC     RORB1
010372  100001                  BPL     .+4
010374  104000          RORB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010376  000241                  CLC
010400  105114                  COMB    (R4)            ;(R3)=000177,CC=0101
010402  103002                  BCC     COMB1
010404  102401                  BVS     COMB1
010406  001401                  BEQ     .+4
010410  104000          COMB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
```

```
010412  106213          1$:      ASRB    (R3)            ;SHIFT EVEN BYTE UNTIL V CLEARS
010414  102002                   BVC     2$
010416  105514                   ADCB    (R4)            ;AND ADD CARRY TO ODD BYTE
010420  000774                   BR      1$
010422  103401          2$:      BCS     ASRB1
010424  001401                   BEQ     .+4
010426  104000          ASRB1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010430  106214                   ASRB    (R4)
010432  106214                   ASRB    (R4)            ;(R3)=000400,CC=0011
010434  103002                   BCC     ASRB1A
010436  102001                   BVC     ASRB1A
010440  001001                   BNE     .+4
010442  104000          ASRB1A:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010444  105314                   DECB    (R4)            ;(R3)=000000,CC=0100
010446  001401                   BEQ     .+4
010450  104000                   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010452  000261                   SEC
010454  106014                   RORB    (R4)            ;(R3)=100000,CC=1010
010456  103402                   BCS     RORB1A
010460  102001                   BVC     RORB1A
010462  100401                   BMI     .+4
010464  104000          RORB1A:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010466  000242                   CLV
010470  105314                   DECB    (R4)            ;(R3)=077400,CC=0100
010472  102401                   BVS     .+4
010474  104000                   HLT

010476  000261                   SEC
010500  105313                   DECB    (R3)            ;(R3)=077777,CC=1001
010502  103002                   BCC     DECB1A
010504  102401                   BVS     DECB1A
010506  100401                   BMI     .+4
010510  104000          DECB1A:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010512  000277                   SCC
010514  000313                   SWAB    (R3)            ;(R3)=177577=[1774][177],CC=0000
010516  103402                   BCS     SWAB1
010520  102401                   BVS     SWAB1
010522  100001                   BPL     .+4
010524  104000          SWAB1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010526  105714                   TSTB    (R4)            ;(R3)=177577=[1774][177],CC=1000
010530  103402                   BCS     TSTB1
010532  102401                   BVS     TSTB1
010534  100401                   BMI     .+4
010536  104000          TSTB1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010540  105014                   CLRB    (R4)            ;(R3)=000177=[0000][177],CC=0100
010542  001401                   BEQ     .+4
010544  104000                   HLT
010546  106313                   ASLB    (R3)            ;(R3)=000376 ,CC=1010
```

```
010552  102001                              BVC     ASLB1A
010554  100401                              BMI     .+4
010556  104000                    ASLB1A:   HLT                             ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010560  105113                              COMB    (R3)                    ;(R3)=000001,CC=0001
010562  103002                              BCC     COMB1A
010564  102401                              BVS     COMB1A
010566  100001                              BPL     .+4
010570  104000                    COMB1A:   HLT                             ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010572  000313                              SWAB    (R3)                    ;(R3)=000400, CC=0100
010574  001401                              BEQ     .+4
010576  104000                              HLT

010600  105213                              INCB    (R3)
010602  000261                              SEC
010604  105613                              SBCB    (R3)                    ;(R3)=000400,CC=0100
010606  001401                              BEQ     .+4
010610  104000                              HLT
010612  022713  000400                      CMP     #400,(R3)               ;CHECK REMAINING RESULT
010616  001401                              BEQ     .+4
010620  104000                              HLT
                                  ;;*****************************************************************
                                  ;*TEST 7          CHECK UNIARY WORD OPS USING ADDRESS MODES 2 & 4
                                  ;;*****************************************************************
010622  112737  000007  001202    TST7:     MOVB    #7,@#$TSTNM                       ;LOAD TEST NUMBER
010630  000004                              SCOPE
010632  000401                              BR      .+4
010634  000000                              .WORD   0                       ;ADDRESS RESERVED FOR TESTS
010636  010704                              MOV     PC,R4
010640  162704  000004                      SUB     #4,R4                   ;R4 AND R5 POINT TO
010644  010405                              MOV     R4,R5                   ;RESERVED WORD
010646  005015                              CLR     (R5)                    ;PRESET DATA=0

010650  000277                              SCC
010652  000244                              CLZ
010654  005725                              TST     (R5)+                   ;(R5)=000000,CC=0100
010656  103402                              BCS     TEST2
010660  102401                              BVS     TEST2
010662  001401                              BEQ     .+4
010664  104000                    TEST2:    HLT                             ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010666  005145                              COM     -(R5)                   ;(R5)=177777,CC=1001
010670  103001                              BCC     COM4
010672  100401                              BMI     .+4
010674  104000                    COM4:     HLT                             ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010676  000241                              CLC
010700  006024                              ROR     (R4)+                   ;(R4)=077777,CC=0011
010702  103002                              BCC     ROR2
010704  102001                              BVC     ROR2
010706  100001                              BPL     .+4
010710  104000                    ROR2:     HLT                             ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010712  000257                              CCC
```

```
010716  102002                  BVC     INC4
010720  001401                  BEQ     INC4
010722  100401                  BMI     .+4
010724  104000          INC4:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010726  000261                  SEC
010730  000324                  SWAB    (R4)+           ;(R4)=000200,CC=1000
010732  103401                  BCS     SWAB2
010734  100401                  BMI     .+4
010736  104000          SWAB2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010740  005425                  NEG     (R5)+           ;(R5)=177600,CC=1001
010742  103001                  BCC     NEG2
010744  100401                  BMI     .+4
010746  104000          NEG2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010750  005044                  CLR     -(R4)           ;(R4)=000000,CC=0100
010752  001401                  BEQ     .+4
010754  104000                  HLT

010756  000261                  SEC
010760  006045                  ROR     -(R5)           ;(R5)=100000,CC=1010
010762  000261                  SEC
010764  005525                  ADC     (R5)+           ;(R5)=100001,CC=1000
010766  102401                  BVS     ADC2
010770  100401                  BMI     .+4
010772  104000          ADC2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010774  000262                  SEV
010776  006224                  ASR     (R4)+           ;(R4)=140000,CC=1001
011000  103002                  BCC     ASR2
011002  102401                  BVS     ASR2
011004  100401                  BMI     .+4
011006  104000          ASR2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011010  000262                  SEV
011012  006144                  ROL     -(R4)           ;(R4)=100001, CC=1001
011014  103002                  BCC     ROL4
011016  102401                  BVS     ROL4
011020  100401                  BMI     .+4
011022  104000          ROL4:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011024  005645                  SBC     -(R5)           ;(R5)=100000,CC=1000
011026  103001                  BCC     .+4
011030  104000                  HLT                     ;ERROR! 'C' BIT FAILED TO CLEAR

011032  005325                  DEC     (R5)+           ;(R5)=077777,CC=0010
011034  103402                  BCS     DEC2
011036  102001                  BVC     DEC2
011040  100001                  BPL     .+4
011042  104000          DEC2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011044  006324                  ASL     (R4)+           ;(R4)=177776,CC=1010
011046  102401                  BVS     .+4
011050  104000                  HLT
```

```
011054  103003                                BCC     ASL4
011056  102402                                BVS     ASL4
011060  001401                                BEQ     ASL4
011062  100401                                BMI     .+4
011064  104000                        ASL4:   HLT                             ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011066  022724  177774                        CMP     #177774,(R4)+
011072  001401                                BEQ     .+4
011074  104000                                HLT
011076  020405                                CMP     R4,R5
011100  001401                                BEQ     .+4
011102  104000                                HLT
                                      ;;*********************************************************************
                                      ;*TEST 10        CHECK UNIARY BYTE OPS USING ADDRESS MODES 2 & 4
                                      ;;*********************************************************************
011104  112737  000010  001202        TST10:  MOVB    #10,@#STSTNM            ;LOAD TEST NUMBER
011112  000004                                SCOPE
011114  000401                                BR      .+4                     ;RESERVE A WORD
011116  000000                                .WORD   0                       ;RESERVED WORD
011120  010705                                MOV     PC,R5
011122  162705  000004                        SUB     #4,R5                   ;R5 POINTS TO EVEN BYTE OF RESERVED WORD
011126  010500                                MOV     R5,R0
011130  010002                                MOV     R0,R2
011132  005202                                INC     R2                      ;R2 POINTS TO ODD BYTE OF RESERVED WORD
011134  005010                                CLR     (R0)                    ;PRESET

011136  000277                                SCC
011140  000241                                CLC
011142  105125                                COMB    (R5)+                   ;(R0)=000377,CC=1001
011144  103002                                BCC     COMB2
011146  102401                                BVS     COMB2
011150  100401                                BMI     .+4
011152  104000                        COMB2:  HLT                             ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011154  105542                                ADCB    -(R2)                   ;(R0)=000000,CC=0101
011156  001401                                BEQ     .+4
011160  104000                                HLT                             ;ERROR! INCORRECT RESULT AS SHOWN ABOVE
011162  105525                                ADCB    (R5)+                   ;(R0)=000400,CC=0000
011164  103401                                BCS     ADCB2
011166  001001                                BNE     .+4
011170  104000                        ADCB2:  HLT                             ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011172  000263                                +SEC!SEV
011174  106045                                RORB    -(R5)                   ;(R0)=100000,CC=1001
011176  103003                                BCC     RORB4
011200  102402                                BVS     RORB4
011202  001401                                BEQ     RORB4
011204  100401                                BMI     .+4
011206  104000                        RORB4:  HLT                             ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011210  000277                                SCC
011212  106122                                ROLB    (R2)+                   ;(R0)=100001,CC=0000
011214  103403                                BCS     ROLB2
011216  102402                                BVS     ROLB2
011220  001401                                BEQ     ROLB2
```

```
011224  104000          ROLB2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011226  000257                  CCC
011230  106225                  ASRB    (R5)+           ;(R0)=140001, CC=1010
011232  103402                  BCS     ASRB2
011234  102001                  BVC     ASRB2
011236  100401                  BMI     .+4
011240  104000          ASRB2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011242  105242                  INCB    -(R2)           ;(R0)=140002,CC=0000
011244  000277                  SCC
011246  106222                  ASRB    (R2)+           ;(R0)=140001,CC=0000
011250  103402                  BCS     ASRB2A
011252  102401                  BVS     ASRB2A
011254  100001                  BPL     .+4
011256  104000          ASRB2A: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011260  000266                  +SEZ!SEV                ;SET Z,V
011262  106345                  ASLB    -(R5)           ;(R0)=100001,CC=1001
011264  103003                  BCC     ASLB4
011266  102402                  BVS     ASLB4
011270  001401                  BEQ     ASLB4
011272  100401                  BMI     .+4
011274  104000          ASLB4:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011276  105322                  DECB    (R2)+           ;(R0)=077401=[0774][001] ,CC=0010
011300  103002                  BCC     DECB2
011302  102001                  BVC     DECB2
011304  100001                  BPL     .+4
011306  104000          DECB2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWNABOVE

011310  105645                  SBCB    -(R5)           ;(R0)=077400, CC=0100
011312  103402                  BCS     SBCB4
011314  102401                  BVS     SBCB4
011316  001401                  BEQ     .+4
011320  104000          SBCB4:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011322  105442                  NEGB    -(R2)           ;(R0)=10400,CC=1001
011324  103002                  BCC     NEGB4
011326  102401                  BVS     NEGB4
011330  100401                  BMI     .+4
011332  104000          NEGB4:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011334  105725                  TSTB    (R5)+           ;(R0)=100400,CC=0100
011336  103401                  BCS     TSTB2
011340  001401                  BEQ     .+4
011342  104000          TSTB2:  HLT

011344  105722                  TSTB    (R2)+           ;(R0)=100400,CC=1000
011346  001401                  BEQ     TSTB2A
011350  100401                  BMI     .+4
011352  104000          TSTB2A: HLT

011354  000261                  SEC
011356  000342                  SWAB    -(R2)           ;(R0)=000201,CC=1000
```

```
011362  100401                           BMI     .+4
011364  104000                  SWAB4:   HLT

011366  000277                           SCC
011370  105225                           INCB    (R5)+            ;(R0)=000601=[0004][201],CC=0000
011372  103003                           BCC     INCB2
011374  102402                           BVS     INCB2
011376  001401                           BEQ     INCB2
011400  100001                           BPL     .+4
011402  104000                  INCB2:   HLT

011404  022227  000601                   CMP     (R2)+,#000601    ;CHECK END RESULT
011410  001401                           BEQ     .+4
011412  104000                           HLT
011414  020205                           CMP     R2,R5            ;CHECK REGISTERS
011416  001401                           BEQ     .+4
011420  104000                           HLT
                                ;;*************************************************************
                                ;*TEST 11        CHECK UNIARY WORD OPS USING ADDRESS MODES 3 & 5
                                ;;*************************************************************
011422  112737  000011  001202  TST11:   MOVB    #11,@#STSTNM              ;LOAD TEST NUMBER
011430  000004                           SCOPE
011432  000402                           BR      .+6              ;RESERVE 2 WORDS
011434  000000                           .WORD   0                ;1 FOR THE ADDRESS
011436  000000                           .WORD   0                ;AND 1 FOR DATA
011440  010703                           MOV     PC,R3
011442  162703  000004                   SUB     #4,R3
011446  005013                           CLR     (R3)             ;PRESET DATA
011450  010300                           MOV     R3,R0            ;R0 POINTS TO DATA WORD
011452  005743                           TST     -(R3)
011454  010013                           MOV     R0,(R3)
011456  010304                           MOV     R3,R4

011460  000257                           CCC
011462  005733                           TST     @(R3)+           ;(R0)=000000,CC=0100
011464  001401                           BEQ     .+4
011466  104000                           HLT

011470  000261                           SEC
011472  006053                           ROR     @-(R3)           ;(R0)=100000,CC=1010
011474  103402                           BCS     ROR5
011476  102001                           BVC     ROR5
011500  100401                           BMI     .+4
011502  104000                  ROR5:    HLT

011504  000257                           CCC
011506  006234                           ASR     @(R4)+           ;(R0)=140000,CC=1010
011510  102001                           BVC     ASR3
011512  100401                           BMI     .+4
011514  104000                  ASR3:    HLT

011516  000250                           CLN
011520  006333                           ASL     @(R3)+           ;(R0)=100000,CC=1001
011522  103002                           BCC     ASL3
011524  102401                           BVS     ASL3
```

```
 011530  104000                      ASL3:   HLT

 011532  000277                              SCC
 011534  005354                              DEC     @-(R4)              ;(R0)=077777, CC=0010
 011536  103003                              BCC     DEC5
 011540  102002                              BVC     DEC5
 011542  001401                              BEQ     DEC5
 011544  100001                              BPL     .+4
 011546  104000                      DEC5:   HLT

 011550  005453                              NEG     @-(R3)              ;(R0)=100001, CC=1001
 011552  103002                              BCC     NEG5
 011554  102401                              BVS     NEG5
 011556  100401                              BMI     .+4
 011560  104000                      NEG5:   HLT

 011562  000262                              SEV
 011564  005134                              COM     @(R4)+              ;(R0)=077776, CC=0001
 011566  103001                              BCC     COM3
 011570  102001                              BVC     .+4
 011572  104000                      COM3:   HLT

 011574  005233                              INC     @(R3)+              ;(R0)=077777, CC=0001
 011576  103001                              BCC     INC3
 011600  100001                              BPL     .+4
 011602  104000                      INC3:   HLT

 011604  005554                              ADC     @-(R4)              ;(R0)=100000, CC=1010
 011606  103402                              BCS     ADC5
 011610  102001                              BVC     ADC5
 011612  100401                              BMI     .+4
 011614  104000                      ADC5:   HLT

 011616  000257                              CCC
 011620  006134                              ROL     @(R4)+              ;(R0)=000000,CC=0111
 011622  103002                              BCC     ROL3
 011624  102001                              BVC     ROL3
 011626  001401                              BEQ     .+4
 011630  104000                      ROL3:   HLT

 011632  005253                              INC     @-(R3)              ;(R0)=000001, CC=0001
 011634  005654                              SBC     @-(R4)              ;(R0)=000000, CC=0100
 011636  103401                              BCS     SBC5
 011640  001401                              BEQ     .+4
 011642  104000                      SBC5:   HLT
                                     ;;*************************************************************
                                     ;*TEST 12        CHECK UNIARY BYTE OPS USING ADDRESS MODES 3 & 5
                                     ;;*************************************************************
 011644  112737  000012  001202      TST12:  MOVB    #12,@#$TSTNM            ;LOAD TEST NUMBER
 011652  000004                              SCOPE
 011654  000403                              BR      .+10                ;RESERVE 3 WORDS
 011656  000000                              .WORD   0                   ;1 FOR EVEN BYTE ADDRESS
 011660  000000                              .WORD   0                   ;1 FOR ODD BYTE ADDRESS
 011662  000000                              .WORD   0                   ;AND 1 FOR DATA
 011664  010702                              MOV     PC,R2
```

```
011670  005742                  TST     -(R2)           ;DATA WORD
011672  010200                  MOV     R2,R0           ;R0 POINTS TO THE DATA WORD
011674  005010                  CLR     (R0)            ;PRESET DATA
011676  005742                  TST     -(R2)           ;BACK R2 UP TO
011700  005742                  TST     -(R2)           ;EVEN BYTE ADDRESS WORD
011702  010022                  MOV     R0,(R2)+        ;LOAD ADDRESS
011704  005200                  INC     R0              ;ODD BYTE ADDRESS
011706  010022                  MOV     R0,(R2)+        ;LOAD ODD BYTE ADDRESS
011710  010200                  MOV     R2,R0           ;RESET R0
011712  010205                  MOV     R2,R5
011714  105152                  COMB    @-(R2)          ;(R0)=177400,CC=1001
011716  103001                  BCC     COMB5
011720  100401                  BMI     .+4
011722  104000          COMB5:  HLT
011724  105752                  TSTB    @-(R2)          ;(R0)=177400, CC=0100
011726  001401                  BEQ     .+4
011730  104000                  HLT
011732  000262                  SEV
011734  106255                  ASRB    @-(R5)          ;(R0)=177400, CC=1001
011736  103002                  BCC     ASRB5
011740  102401                  BVS     ASRB5
011742  100401                  BMI     .+4
011744  104000          ASRB5:  HLT

011746  105232                  INCB    @(R2)+          ;(R0)=177401, CC=000
011750  103001                  BCC     INCB3
011752  100001                  BPL     .+4
011754  104000          INCB3:  HLT

011756  000241                  CLC
011760  106055                  RORB    @-(R5)          ;(R0)=177400, CC=0111
011762  103003                  BCC     RORB5
011764  102002                  BVC     RORB5
011766  001001                  BNE     RORB5
011770  100001                  BPL     .+4
011772  104000          RORB5:  HLT

011774  106332                  ASLB    @(R2)+          ;(R0)=177000, CC=1001
011776  103002                  BCC     ASLB3
012000  102401                  BVS     ASLB3
012002  100401                  BMI     .+4
012004  104000          ASLB3:  HLT

012006  105552                  ADCB    @-(R2)          ;(R0)=177400, CC=1000
012010  103401                  BCS     ADCB5
012012  100401                  BMI     .+4
012014  104000          ADCB5:  HLT

012016  000277                  SCC
012020  106135                  ROLB    @(R5)+          ;(R0)=177401, CC=0000
012022  101402                  BLOS    ROLB3           ;BRANCH IF C OR Z IS SET
012024  102401                  BVS     ROLB3
012026  100001                  BPL     .+4
012030  104000          ROLB3:  HLT

```

```
012034  100401                                  BMI     .+4
012036  104000                                  HLT

012040  000261                                  SEC
012042  105635                                  SBCB    @(R5)+          ;(R0)=000377, CC=0100
012044  103401                                  BCS     SBCB3
012046  001401                                  BEQ     .+4
012050  104000                          SBCB3:  HLT

012052  105432                                  NEGB    @(R2)+          ;(R0)=000001
012054  105352                                  DECB    @-(R2)          ;(R0)=000000, CC=0101
012056  103001                                  BCC     DECB5
012060  001401                                  BEQ     .+4
012062  104000                          DECB5:  HLT
                                        ;;*******************************************************************
                                        ;*TEST 13        CHECK UNIARY WORD OPS USING ADDRESS MODE 6 (PC)
                                        ;;*******************************************************************
012064  112737  000013  001202          TST13:  MOVB    #13,@#$TSTNM            ;LOAD TEST NUMBER
012072  000004                                  SCOPE
012074  005027                                  CLR     (PC)+           ;PRESET DATA = 0
012076  000000                          UWM6:   .WORD   0               ;RESERVED FOR DATA
012100  010700                                  MOV     PC,R0
012102  024040                                  CMP     -(R0),-(R0)     ;R0 POINTS TO DATA WORD
012104  000277                                  SCC
012106  006167  177764                          ROL     UWM6            ;(R0)=000001,CC=0000
012112  103403                                  BCS     ROL6
012114  102402                                  BVS     ROL6
012116  001401                                  BEQ     ROL6
012120  100001                                  BPL     .+4
012122  104000                          ROL6:   HLT

012124  005167  177746                          COM     UWM6            ;(R0)=177776, CC=1001
012130  103002                                  BCC     COM6
012132  102401                                  BVS     COM6
012134  100401                                  BMI     .+4
012136  104000                          COM6:   HLT
012140  006267  177732                          ASR     UWM6            ;(R0)=177777, CC=1010
012144  103402                                  BCS     ASR6
012146  102001                                  BVC     ASR6
012150  100401                                  BMI     .+4
012152  104000                          ASR6:   HLT

012154  000277                                  SCC
012156  005467  177714                          NEG     UWM6            ;(R0)=000001, CC=0001
012162  103003                                  BCC     NEG6
012164  102402                                  BVS     NEG6
012166  001401                                  BEQ     NEG6
012170  100001                                  BPL     .+4
012172  104000                          NEG6:   HLT

012174  000277                                  SCC
012176  006067  177674                          ROR     UWM6            ;(R0)=100000, CC=1001
012202  103003                                  BCC     ROR6
012204  102402                                  BVS     ROR6
012206  001401                                  BEQ     ROR6
```

```
012212  104000                          ROR6:   HLT

012214  005667  177656                          SBC     UWM6                    ;(R0)=077777, CC=0010
012220  103402                                  BCS     SBC6
012222  102001                                  BVC     SBC6
012224  100001                                  BPL     .+4
012226  104000                          SBC6:   HLT

012230  000242                                  CLV
012232  005267  177640                          INC     UWM6                    ;(R0)=100000, CC=1011
012236  103403                                  BCS     INC6
012240  102002                                  BVC     INC6
012242  001401                                  BEQ     INC6
012244  100401                                  BMI     .+4
012246  104000                          INC6:   HLT

012250  006267  177622                          ASR     UWM6                    ;(R0)=140000, CC=1010
012254  000261                                  SEC
012256  006367  177614                          ASL     UWM6                    ;(R0)=100000, CC=1001
012262  103002                                  BCC     ASL6
012264  102401                                  BVS     ASL6
012266  100401                                  BMI     .+4
012270  104000                          ASL6:   HLT

012272  005367  177600                          DEC     UWM6                    ;(R0)=077777, CC=0011
012276  103002                                  BCC     DEC6
012300  102001                                  BVC     DEC6
012302  100001                                  BPL     .+4
012304  104000                          DEC6:   HLT

012306  005567  177564                          ADC     UWM6                    ;(R0)=100000, CC=1010
012312  103402                                  BCS     ADC6
012314  102001                                  BVC     ADC6
012316  100401                                  BMI     .+4
012320  104000                          ADC6:   HLT
012322  000242                                  CLV
012324  000367  177546                          SWAB    UWM6
012330  100401                                  BMI     .+4
012332  104000                                  HLT
012334  022710  000200                          CMP     #200,(R0)
012340  001401                                  BEQ     .+4
012342  104000                                  HLT
                                        ;;*****************************************************************
                                        ;*TEST 14       CHECK UNIARY BYTE OPS (EVEN/ODD) USING ADDRESS MODE 6 (PC)
                                        ;;*****************************************************************
012344  112737  000014  001202          TST14:  MOVB    #14,@#$TSTNM            ;LOAD TEST NUMBER
012352  000004                                  SCOPE
012354  012700  012716                          MOV     #UBM6,R0
012360  063700  001506                          ADD     @#FACTOR,R0     ;R0 POINTS TO ADDRESS OF DATA
012364  005067  000326                          CLR     UBM6            ;CLEAR DATA
012370  000277                                  SCC
012372  000244                                  CLZ
012374  105767  000316                          TSTB    UBM6
012400  103403                                  BCS     TSTB6
012402  102402                                  BVS     TSTB6
```

```
012406  100001                              BPL     .+4
012410  104000                  TSTB6:      HLT

012412  000257                              CCC
012414  105767  000277                      TSTB    UBM6+1          ;TEST ODD BYTE
012420  001401                              BEQ     .+4
012422  104000                              HLT

012424  105667  000266                      SBCB    UBM6            ;(R0)=000000, CC=0100
012430  103402                              BCS     SBCB6
012432  102401                              BVS     SBCB6
012434  001401                              BEQ     .+4
012436  104000                  SBCB6:      HLT

012440  000261                  1$:         SEC
012442  105267  000250                      INCB    UBM6            ;LOOP UNTIL (R0)=077600, CC=1011
012446  100403                              BMI     2$
012450  105567  000243                      ADCB    UBM6+1          ;INCB INST INCREMENTS EVEN BYTE
012454  000771                              BR      1$              ;ADCB INCREMENTS ODD BYTE
012456  103001                  2$:         BCC     INCB6
012460  102401                              BVS     .+4
012462  104000                  INCB6:      HLT

012464  106367  000226                      ASLB    UBM6            ;(R0)=077400, CC=0111
012470  103003                              BCC     ASLB6
012472  102002                              BVC     ASLB6
012474  001001                              BNE     ASLB6
012476  100001                              BPL     .+4
012500  104000                  ASLB6:      HLT

```

```
012504  105567  000207             ADCB    UBM6+1          ;(R0)=100000, CC=1010
012510  103402                     BCS     ADCB6
012512  102001                     BVC     ADCB6
012514  100401                     BMI     .+4
012516  104000           ADCB6:    HLT

012520  000261                     SEC
012522  106067  000171             RORB    UBM6+1          ;(R0)=140000, CC=1010
012526  103402                     BCS     RORB6
012530  102001                     BVC     RORB6
012532  100401                     BMI     .+4
012534  104000           RORB6:    HLT

012536  105167  000154             COMB    UBM6            ;(R0)=140377 CC=1001
012542  103002                     BCC     COMB6
012544  102401                     BVS     COMB6
012546  100401                     BMI     .+4
012550  104000           COMB6:    HLT

012552  000262                     SEV
012554  105467  000137             NEGB    UBM6+1          ;(R0)=040377, CC=0001
012560  103002                     BCC     NEGB6
012562  102401                     BVS     NEGB6
012564  100001                     BPL     .+4
012566  104000           NEGB6:    HLT

012570  106167  000123             ROLB    UBM6+1          ;(R0)=100777, CC=1010
012574  103402                     BCS     ROLB6
012576  102001                     BVC     ROLB6
012600  100401                     BMI     .+4
012602  104000           ROLB6:    HLT

012604  106267  000106             ASRB    UBM6            ;(R0)=100777, CC=1001
012610  103002                     BCC     ASRB6
012612  102401                     BVS     ASRB6
012614  100401                     BMI     .+4
012616  104000           ASRB6:    HLT

012620  105267  000072             INCB    UBM6            ;(R0)=100400, CC=0101
012624  103002                     BCC     INCB6A
012626  102401                     BVS     INCB6A
012630  001401                     BEQ     .+4
012632  104000           INCB6A:   HLT

012634  105367  000057             DECB    UBM6+1          ;(R0)=100000, CC=1001
012640  103003                     BCC     DECB6A
012642  102402                     BVS     DECB6A
012644  001401                     BEQ     DECB6A
012646  100401                     BMI     .+4
012650  104000           DECB6A:   HLT

012652  000367  000040             SWAB    UBM6            ;(R0)=000200, CC=1000
012656  103401                     BCS     SWAB6
012660  100401                     BMI     .+4
012662  104000           SWAB6:    HLT
```

```
012664  106167  000026                    ROLB    UBM6                ;(R0)=000000, CC=0111
012670  103002                            BCC     ROLB6A
012672  102001                            BVC     ROLB6A
012674  001401                            BEQ     .+4
012676  104000                  ROLB6A:   HLT

012700  005767  000012                    TST     UBM6                ;(R0)=000000, CC=0100
012704  103402                            BCS     TEST6
012706  102401                            BVS     TEST6
012710  001401                            BEQ     .+4
012712  104000                  TEST6:    HLT

012714  000401                            BR      .+4                 ;RESERVE A WORD
012716  000000                  UBM6:     .WORD   0                   ;WORD RESERVED FOR DATA
012720  000004                  RELE1:    SCOPE
012722  010702                            MOV     PC,R2
012724  062702  000012                    ADD     #12,R2
012730  012707  034242                    MOV     #RELOC,PC           ;GO RELOCATE PROGRAM CODE
012734  000000                  REL11:    .WORD   0
                                ;1111111111111 LAST ADDRESS OF CODE TO BE RELOCATED 1111111111


                                ;;*****************************************************************
                                ;*TEST 15         CHECK UNIARY WORD OPS USING ADDRESS MODE 7
                                ;;*****************************************************************
012736  112737  000015  001202  TST15:    MOVB    #15,@#$TSTNM              ;LOAD TEST NUMBER
012744  012767  000001  166344            MOV     #1,$TIMES           ;;DO 1 ITERATION
012752  000004                            SCOPE


                                          .SBTTL  START OF SECTION 2
                                ;2222222222222 FIRST ADDRESS TO BE RELOCATED 222222222
012754  010700                  REL2:     MOV     PC,R0               ;GET PC
012756  005740                            TST     -(R0)               ;R0 CONTAINS THE ADDRESS OF REL2
012760  010037  001512                    MOV     R0,@#FRSTAD         ;SAVE
012764  010700                            MOV     PC,R0               ;GET CURRENT PC
012766  162700  012766                    SUB     #.,R0               ;SUBTRACT RELOCATION FACTOR
012772  010037  001506                    MOV     R0,@#FACTOR         ;SAVE RELOCATION FACTOR
012776  010737  001212                    MOV     PC,@#$LPERR         ;SET LOOP ADDRESS
013002  062737  000030  001212            ADD     #30,@#$LPERR        ;ADJUST
013010  013737  001212  001210            MOV     @#$LPERR,@#$LPADR
013016  105737  001502                    TSTB    @#NEXEC             ;BR IF TEST CODE TO BE EXECUTED
013022  001402                            BEQ     .+6
013024  000167  004060                    JMP     RELE2
013030  000403                            BR      UW7                 ;RESERVE 3 WORDS FOR ADDRESSES & DATA
013032  000000                            .WORD   0                   ;CONTAINS ADDRESS OF UWM7
013034  000000                  UWM7:     .WORD   0                   ;CONTAINS DATA
013036  000000                            .WORD   0                   ;CONTAINS ADDRESS OF UWM7

013040  010700                  UW7:      MOV     PC,R0
013042  005740                            TST     -(R0)
013044  005740                            TST     -(R0)
013046  005040                            CLR     -(R0)               ;CLEAR TEST DATA
013050  010002                            MOV     R0,R2
013052  010240                            MOV     R2,-(R0)            ;SET UP ADDRESS
013054  005720                            TST     (R0)+               ;MOVE R0 TO NEXT ADDRESS
013056  005720                            TST     (R0)+
013062  010200                          MOV     R2,R0           ;SET R0 POINTING TO DATA
013064  000277                          SCC
013066  000244                          CLZ
013070  005772  000002                  TST     @2(2)           ;(R0)=000000, CC=0100
013074  001401                          BEQ     .+4
013076  104000                          HLT

013100  000277                          SCC
013102  005672  177776                  SBC     @-2(2)          ;(R0)=177777, CC=1001
013106  103002                          BCC     SBC7
013110  102401                          BVS     SBC7
013112  100401                          BMI     .+4
013114  104000                  SBC7:   HLT

013116  000277                          SCC
013120  000241                          CLC
013122  006372  000002                  ASL     @2(2)           ;(R0)=177776, CC=1001
013126  103002                          BCC     ASL7
013130  102401                          BVS     ASL7
013132  100401                          BMI     .+4
013134  104000                  ASL7:   HLT

013136  000257                          CCC
013140  005372  000002                  DEC     @2(2)           ;(R0)=177775, CC=1000
013144  103402                          BCS     DEC7
013146  102401                          BVS     DEC7
013150  100401                          BMI     .+4
013152  104000                  DEC7:   HLT

013154  000262                          SEV
013156  006272  177776                  ASR     @-2(2)          ;(R0)=177776, CC=1001
013162  103002                          BCC     ASR7
013164  102401                          BVS     ASR7
013166  100401                          BMI     .+4
013170  104000                  ASR7:   HLT

013172  000241                          CLC
013174  000262                          SEV
013176  006072  177776                  ROR     @-2(2)          ;(R0)=077777, CC=0000
013202  101402                          BLOS    ROR7            ;BRANCH IF C OR Z IS SET
013204  102401                          BVS     ROR7
013206  100001                          BPL     .+4
013210  104000                  ROR7:   HLT

013212  000262                          SEV
013214  005472  000002                  NEG     @2(2)           ;(R0)=100001, CC=1001
013220  103002                          BCC     NEG7
013222  102401                          BVS     NEG7
013224  100401                          BMI     .+4
013226  104000                  NEG7:   HLT

013230  000250                          CLN
013232  000372  177776                  SWAB    @-2(2)          ;(R0)=000600, CC=1000
013236  103401                          BCS     SWAB7
013240  100401                          BMI     .+4
```

```

013244  000262                          SEV                                     
013246  005172  000002                  COM     @2(2)           ;(R0)=177177, CC=1001
013252  103002                          BCC     COM7
013254  102401                          BVS     COM7
013256  100401                          BMI     .+4
013260  104000                  COM7:   HLT

013262  000372  000002                  SWAB    @2(2)           ;(R0)=077776, CC=1000
013266  100401                          BMI     .+4
013270  104000                          HLT

013272  000277                          SCC
013274  005572  177776                  ADC     @-2(2)          ;(R0)=077777, CC=0000
013300  103402                          BCS     ADC7
013302  102401                          BVS     ADC7
013304  100001                          BPL     .+4
013306  104000                  ADC7:   HLT

013310  005272  000002                  INC     @2(2)           ;(R0)=100000, CC=1010
013314  102001                          BVC     INC7
013316  100401                          BMI     .+4
013320  104000                  INC7:   HLT

013322  000257                          CCC
013324  006172  177776                  ROL     @-2(2)          ;(R0)=000000, CC=0111
013330  103002                          BCC     ROL7
013332  102001                          BVC     ROL7
013334  001401                          BEQ     .+4
013336  104000                  ROL7:   HLT
                                ;;*************************************************************
                                ;*TEST 16        CHECK UNIARY BYTE OPS USING ADDRESS MODE 7
                                ;;*************************************************************
013340  112737  000016  001202  TST16:  MOVB    #16,@#$TSTNM            ;LOAD TEST NUMBER
013346  000004                          SCOPE
013350  012700  013034                  MOV     #UWM7,R0
013354  063700  001506                  ADD     @#FACTOR,R0
013360  010002                          MOV     R0,R2
013362  010067  177450                  MOV     R0,UWM7+2
013366  005720                          TST     (R0)+
013370  005210                          INC     (R0)            ;WORD FOLLOWING UWM7 CONTAINS ADDRESS
013372  005740                          TST     -(R0)           ;OF ODD BYTE, R0 POINTS TO DATA WORD
013374  005010                          CLR     (R0)            ;PRESET DATA
013376  010067  177430                  MOV     R0,UWM7-2
                                ;NOTE:  @2(2) REFERENCES THE ODD BYTE, AND @-2(2) REFERENCES THE EVEN BYTE.

013402  000263                          +SEC!SEV                ;SET C AND V
013404  105672  000002                  SBCB    @2(2)           ;(R0)=177400, CC=1001
013410  103003                          BCC     SBCB7
013412  102402                          BVS     SBCB7
013414  001401                          BEQ     SBCB7
013416  100401                          BMI     .+4
013420  104000                  SBCB7:  HLT

013422  000277                          SCC                     ;SET CONDITION CODES
```

```
 013430  103403                           BCS     ADCB7
 013432  102402                           BVS     ADCB7
 013434  001401                           BEQ     ADCB7
 013436  100001                           BPL     .+4
 013440  104000                   ADCB7:  HLT

 013442  105172  177776                   COMB    @-2(2)                ;(R0)=177776, CC=1001
 013446  103002                           BCC     COMB7
 013450  102401                           BVS     COMB7
 013452  100401                           BMI     .+4
 013454  104000                   COMB7:  HLT

 013456  000241                           CLC                           ;CLEAR CARRY
 013460  106072  000002                   RORB    @2(2)                 ;(R0)=077776, CC=0011
 013464  103002                           BCC     RORB7
 013466  102001                           BVC     RORB7
 013470  100001                           BPL     .+4
 013472  104000                   RORB7:  HLT

 013474  105272  000002                   INCB    @2(2)                 ;(R0)=100376, CC=1011
 013500  103002                           BCC     INCB7
 013502  102001                           BVC     INCB7
 013504  100401                           BMI     .+4
 013506  104000                   INCB7:  HLT

 013510  105372  177776                   DECB    @-2(2)                ;(R0)=100375, CC=1001
 013514  103002                           BCC     DECB7
 013516  102401                           BVS     DECB7
 013520  100401                           BMI     .+4
 013522  104000                   DECB7:  HLT

 013524  106372  000002                   ASLB    @2(2)                 ;(R0)=000375, CC=0111
 013530  103002                           BCC     ASLB7
 013532  102001                           BVC     ASLB7
 013534  001401                           BEQ     .+4
 013536  104000                   ASLB7:  HLT

 013540  000241                           CLC                           ;CLEAR CARRY
 013542  106272  177776                   ASRB    @-2(2)                ;(R0)=000376, CC=1001
 013546  103002                           BCC     ASRB7
 013550  102401                           BVS     ASRB7
 013552  100401                           BMI     .+4
 013554  104000                   ASRB7:  HLT

 013556  105472  000002                   NEGB    @2(2)                 ;(R0)=000376, CC=0100
 013562  103402                           BCS     NEGB7
 013564  102401                           BVS     NEGB7
 013566  001401                           BEQ     .+4
 013570  104000                   NEGB7:  HLT

 013572  000262                           SEV
 013574  106172  177776                   ROLB    @-2(2)                ;(R0)=00374, CC=1001
 013600  103002                           BCC     ROLB7
 013602  102401                           BVS     ROLB7
 013604  100401                           BMI     .+4
```

```

013610  105272  177776                  INCB    @-2(2)                  ;(R0)=000375, CC=1001
013614  105272  177776                  INCB    @-2(2)                  ;(R0)=000376, CC=1001
013620  105572  177776                  ADCB    @-2(2)                  ;(R0)=000377, CC=1000
013624  105172  177776                  COMB    @-2(2)                  ;(R0)=000000, CC=0100
013630  001401                          BEQ     .+4
013632  104000                          HLT
                                ;;*************************************************************
                                ;*TEST 17       CHECK BINARY OPS USING ADDRESS MODE 0
                                ;;*************************************************************
013634  112737  000017  001202  TST17:  MOVB    #17,@#$TSTNM                    ;LOAD TEST NUMBER
013642  000004                          SCOPE
013644  000277                          SCC                             ;SET CONDITION CODES
013646  010700                          MOV     PC,R0                   ;R0=PC, CC=X001
013650  103002                          BCC     MOV0
013652  102401                          BVS     MOV0
013654  001001                          BNE     .+4
013656  104000                  MOV0:   HLT

013660  010002                          MOV     R0,R2                   ;R2=R0
013662  000262                          SEV                             ;SET V
013664  160002                          SUB     R0,R2                   ;R2=000000, CC=0100
013666  103402                          BCS     SUB0
013670  102401                          BVS     SUB0
013672  001401                          BEQ     .+4
013674  104000                  SUB0:   HLT

013676  000244                          CLZ
013700  010203                          MOV     R2,R3                   ;R2=R3=000000, CC=0100
013702  103401                          BCS     MOV0A
013704  001401                          BEQ     .+4
013706  104000                  MOV0A:  HLT

013710  000257                          CCC
013712  000272                          +SEV!SEN                        ;SET V & N
013714  020203                          CMP     R2,R3                   ;R2=R3=000000, CC=0100
013716  103403                          BCS     CMP0
013720  102402                          BVS     CMP0
013722  001001                          BNE     CMP0
013724  100001                          BPL     .+4
013726  104000                  CMP0:   HLT

013730  010002                          MOV     R0,R2                   ;R0=R2
013732  010203                          MOV     R2,R3                   ;R0=R2=R3
013734  060203                          ADD     R2,R3                   ;R3=2*R0
013736  006302                          ASL     R2                      ;R2=2*R0
013740  020203                          CMP     R2,R3                   ;R2=R3=2*R0
013742  001401                          BEQ     .+4
013744  104000                          HLT                             ;ERROR! CHECK ADD INSTRUCTION

                                ;THE FOLLOWING SUBTEST SHIFTS A BIT THROUGH R2 AND R5 AND DOES A
                                ;BIT TEST (BIT) USING R2 AND R5.
013746  005002                          CLR     R2
013750  005202                          INC     R2
013752  000402                          BR      2$
```

```
 013756  100407                           BMI     4$
 013760  010205                   2$:     MOV     R2,R5
 013762  000277                           SCC
 013764  030205                           BIT     R2,R5           ;R2=R5
 013766  103002                           BCC     3$
 013770  102401                           BVS     3$
 013772  001370                           BNE     1$
 013774  104000                   3$:     HLT
 013776  010205                   4$:     MOV     R2,R5
 014000  000257                           CCC
 014002  030205                           BIT     R2,R5
 014004  100401                           BMI     .+4
 014006  104000                           HLT

 014010  005002                           CLR     R2
 014012  000277                           SCC
 014014  050002                           BIS     R0,R2
 014016  103002                           BCC     BIS0
 014020  102401                           BVS     BIS0
 014022  001001                           BNE     .+4
 014024  104000                   BIS0:   HLT

 014026  010003                           MOV     R0,R3
 014030  000277                           SCC
 014032  000244                           CLZ
 014034  040003                           BIC     R0,R3
 014036  103003                           BCC     BIC0
 014040  102402                           BVS     BIC0
 014042  001001                           BNE     BIC0
 014044  100001                           BPL     .+4
 014046  104000                   BIC0:   HLT

 014050  010004                           MOV     R0,R4
 014052  005104                           COM     R4
 014054  040004                           BIC     R0,R4
 014056  005104                           COM     R4
 014060  020004                           CMP     R0,R4
 014062  001401                           BEQ     .+4
 014064  104000                           HLT

 014066  010004                           MOV     R0,R4
 014070  005104                           COM     R4
 014072  010403                           MOV     R4,R3
 014074  050003                           BIS     R0,R3
 014076  103001                           BCC     BIS0A
 014100  100401                           BMI     .+4
 014102  104000                   BIS0A:  HLT
 014104  005203                           INC     R3
 014106  001401                           BEQ     .+4
 014110  104000                           HLT
 014112  010304                           MOV     R3,R4           ;R3=R4=0
 014114  005103                           COM     R3              ;R3=177777
 014116  000261                           SEC                     ;SET C
 014120  006004                           ROR     R4              ;R4=100000
 014122  060304                           ADD     R3,R4           ;R3=177777,R4=077777, CC=0011
```

```
014126  102002                            BVC     ADD0
014130  001401                            BEQ     ADD0
014132  100001                            BPL     .+4
014134  104000                    ADD0:   HLT
014136  010700                            MOV     PC,R0
014140  022020                            CMP     (R0)+,(R0)+
014142  020007                            CMP     R0,PC
014144  001401                            BEQ     .+4
014146  104000                            HLT

014150  010700                            MOV     PC,R0
014152  062700  000010                    ADD     #10,R0
014156  010002                            MOV     R0,R2
014160  020700                            CMP     PC,R0
014162  001002                            BNE     CMP0A
014164  020200                            CMP     R2,R0
014166  001401                            BEQ     .+4
014170  104000                    CMP0A:  HLT
                                  ;;*******************************************************************
                                  ;*TEST 20       CHECK BINARY OPS USING ADDRESS MODE 1
                                  ;;*******************************************************************
014172  112737  000020  001202    TST20:  MOVB    #20,@#$TSTNM            ;LOAD TEST NUMBER
014200  000004                            SCOPE
014202  000402                            BR      .+6             ;RESERVE TWO WORDS
014204  000000                            .WORD   0               ;RESERVED FOR SOURCE DATA
014206  000000                            .WORD   0               ;RESERVED FOR DESTINATION DATA
014210  010704                            MOV     PC,R4
014212  005744                            TST     -(R4)
014214  005044                            CLR     -(R4)           ;R4 POINTS TO DESTINATION DATA
014216  010403                            MOV     R4,R3
014220  005043                            CLR     -(R3)           ;R3 POINTS TO SOUCE DATA

014222  005113                            COM     (R3)            ;(R3)=177777
014224  005214                            INC     (R4)            ;(R4)=000001
014226  000262                            SEV                     ;SET V
014230  061314                            ADD     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=0101
014232  103002                            BCC     ADD1
014234  102401                            BVS     ADD1
014236  001401                            BEQ     .+4
014240  104000                    ADD1:   HLT

014242  000277                            SCC
014244  000250                            CLN
014246  021314                            CMP     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=1000
014250  103403                            BCS     CMP1
014252  102402                            BVS     CMP1
014254  001401                            BEQ     CMP1
014256  100401                            BMI     .+4
014260  104000                    CMP1:   HLT

014262  000277                            SCC
014264  000244                            CLZ
014266  031314                            BIT     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=0101
014270  103002                            BCC     BITT1
014272  102401                            BVS     BITT1
```

```
014276  104000          BITT1:  HLT

014300  000277                  SCC
014302  000245                  +CLC!CLZ
014304  005114                  COM     (R4)            ;(R4)=177777
014306  161314                  SUB     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=0100
014310  103402                  BCS     SUB1
014312  102401                  BVS     SUB1
014314  001401                  BEQ     .+4
014316  104000          SUB1:   HLT

014320  105013                  CLRB    (R3)            ;(R3)=177400
014322  000313                  SWAB    (R3)            ;(R3)=000377
014324  000270                  SEN
014326  011314                  MOV     (R3),(R4)       ;(R3)=(R4)=000377
014330  100001                  BPL     .+4
014332  104000                  HLT
014334  000314                  SWAB    (R4)            ;(R3)=000377,(R4)=177400
014336  000263                  +SEC!SEV                ;SET C & V
014340  051314                  BIS     (R3),(R4)       ;(R3)=000377,(R4)=177777, CC=1001
014342  103002                  BCC     BIS1
014344  102401                  BVS     BIS1
014346  100401                  BMI     .+4
014350  104000          BIS1:   HLT

014352  041314                  BIC     (R3),(R4)       ;(R3)=000377,(R4)=177400, CC=1001
014354  103002                  BCC     BIC1
014356  102401                  BVS     BIC1
014360  100401                  BMI     .+4
014362  104000          BIC1:   HLT

014364  000262                  SEV                     ;SET V
014366  021314                  CMP     (R3),(R4)       ;(R3)=000377,(R4)=177400, CC=0001
014370  103003                  BCC     CMP1A
014372  102402                  BVS     CMP1A
014374  001401                  BEQ     CMP1A
014376  100001                  BPL     .+4
014400  104000          CMP1A:  HLT

014402  005013                  CLR     (R3)            ;(R3)=000000
014404  000261                  SEC
014406  006013                  ROR     (R3)            ;(R3)=100000
014410  011314                  MOV     (R3),(R4)       ;(R3)=(R4)=100000
014412  005114                  COM     (R4)            ;(R4)=077777
014414  161314                  SUB     (R3),(R4)       ;(R3)=100000,(R4)=177777, CC=1011
014416  103002                  BCC     SUB1A
014420  102001                  BVC     SUB1A
014422  100401                  BMI     .+4
014424  104000          SUB1A:  HLT

014426  000277                  SCC
014430  161314                  SUB     (R3),(R4)       ;(R3)=100000,(R4)=077777, CC=0000
014432  101402                  BLOS    SUB1B           ;BRANCH IF C OR Z IS SET
014434  102401                  BVS     SUB1B
014436  100001                  BPL     .+4
```

```

014442  011314                           MOV     (R3),(R4)          ;(R3)=100000,(R4)=100000, CC=1000
014444  001401                           BEQ     MOV1
014446  100401                           BMI     .+4
014450  104000                   MOV1:   HLT

014452  061314                           ADD     (R3),(R4)          ;(R3)=100000,(R4)=000000, CC=0111
014454  103003                           BCC     ADD1A
014456  102002                           BVC     ADD1A
014460  001001                           BNE     ADD1A
014462  100001                           BPL     .+4
014464  104000                   ADD1A:  HLT

014466  005113                           COM     (R3)               ;(R3)=077777
014470  011314                           MOV     (R3),(R4)          ;(R4)=077777
014472  061314                           ADD     (R3),(R4)          ;(R3)=077777,(R4)=177776, CC=1010
014474  103402                           BCS     ADD1B
014476  102001                           BVC     ADD1B
014500  100401                           BMI     .+4
014502  104000                   ADD1B:  HLT

014504  062714  000002                   ADD     #2,(R4)
014510  005714                           TST     (R4)               ;CHECK FINAL RESULT
014512  001401                           BEQ     .+4
014514  104000                           HLT
                                 ;;*******************************************************************
                                 ;*TEST 21        CHECK BINARY BYTE OPS USING ADDRESS MODE 1
                                 ;;*******************************************************************
014516  112737  000021  001202   TST21:  MOVB    #21,@#$TSTNM                 ;LOAD TEST NUMBER
014524  000004                           SCOPE
014526  000402                           BR      .+6
014530  000000                           .WORD   0
014532  000000                           .WORD   0
014534  010705                           MOV     PC,R5
014536  005745                           TST     -(R5)
014540  005045                           CLR     -(R5)              ;(R5)=000000
014542  010502                           MOV     R5,R2
014544  005042                           CLR     -(R2)              ;(R2)=000000
014546  005202                           INC     R2                 ;R2 POINTS TO ODD BYTE
014550  105112                           COMB    (R2)               ;(R2)=177400

014552  000277                           SCC
014554  111215                           MOVB    (R2),(R5)          ;(R2)=177400,(R5)=000377,CC=1001
014556  103005                           BCC     MOVB1
014560  102404                           BVS     MOVB1
014562  001403                           BEQ     MOVB1
014564  100002                           BPL     MOVB1
014566  105215                           INCB    (R5)               ;CHECK RESULT
014570  001401                           BEQ     .+4
014572  104000                   MOVB1:  HLT

014574  106312                           ASLB    (R2)               ;SHIFT (R2) UNTIL
014576  102376                           BVC     .-2                ;(R2)=000000
014600  106012                           RORB    (R2)               ;(R2)=100000
014602  105315                           DECB    (R5)               ;(R5)=00377
```

```
 014606  000257                    CCC
 014610  121512                    CMPB    (R5),(R2)               ;(R5)=000177,(R2)=100000, CC=1010
 014612  102001                    BVC     CMPB1
 014614  100401                    BMI     .+4
 014616  104000            CMPB1:  HLT

 014620  005003                    CLR     R3
 014622  000261                    SEC
 014624  006003                    ROR     R3                      ;R3=100000
 014626  050315                    BIS     R3,(R5)                 ;(R5)=100177
 014630  000273                    +SEC!SEV!SEN                    ;SET C,V, & N
 014632  131215                    BITB    (R2),(R5)               ;(R2)=100000,(R5)=100177, CC=0101
 014634  103002                    BCC     BITB1
 014636  102401                    BVS     BITB1
 014640  001401                    BEQ     .+4
 014642  104000            BITB1:  HLT

 014644  151215                    BISB    (R2),(R5)               ;(R2)=100000,(R5)=100377, CC=1001
 014646  103001                    BCC     BISB1
 014650  100401                    BMI     .+4
 014652  104000            BISB1:  HLT

 014654  141215                    BICB    (R2),(R5)               ;(R2)=100000,(R5)=100177, CC=0001
 014656  103002                    BCC     BICB1
 014660  001401                    BEQ     BICB1
 014662  100001                    BPL     .+4
 014664  104000            BICB1:  HLT

 014666  105112                    COMB    (R2)                    ;(R2)=077400,(R5)=100177
 014670  121215                    CMPB    (R2),(R5)
 014672  001401                    BEQ     .+4
 014674  104000                    HLT

 014676  141512                    BICB    (R5),(R2)               ;(R5)=100177,(R2)=000000, CC=0100
 014700  001002                    BNE     BICB1A
 014702  105712                    TSTB    (R2)
 014704  001401                    BEQ     .+4
 014706  104000            BICB1A: HLT

 014710  000402                    BR      .+6                     ;RESERVE TWO WORDS FOR DATA
 014712  000000                    .WORD   0                       ;SOURCE DATA
 014714  000000                    .WORD   0                       ;DEST DATA
 014716  010705                    MOV     PC,R5
 014720  005745                    TST     -(R5)
 014722  105045                    CLRB    -(R5)                   ;R5 POINTS TO DEST ODD BYTE
 014724  010504                    MOV     R5,R4
 014726  105044                    CLRB    -(R4)                   ;R4 POINTS TO DEST EVEN BYTE
 014730  010403                    MOV     R4,R3
 014732  105043                    CLRB    -(R3)                   ;R3 POINTS TO SOURCE ODD BYTE
 014734  010302                    MOV     R3,R2
 014736  105042                    CLRB    -(R2)                   ;R2 POINTS TO SOURCE EVEN BYTE

                           ;COMMENTS ARE LEAST SIGNIFICANT 4 BITS OF BYTES POINTED TO BY R2,R3
                           ;R4, AND R5 RESPECTIVELY AND THE REMAINING BITS ARE 0'S.
 014740  000261                    SEC                             ;SET CARRY
```

```
014742  106112                          ROLB    (R2)            ;0001,0000,0000,0000
014744  111214                          MOVB    (R2),(R4)       ;0001,0000,0001,0000
014746  106112                          ROLB    (R2)            ;0010,0000,0001,0000
014750  111213                          MOVB    (R2),(R3)       ;0010,0010,0001,0000
014752  106112                          ROLB    (R2)            ;0100,0010,0001,0000
014754  111315                          MOVB    (R3),(R5)       ;0100,0010,0001,0010
014756  106112                          ROLB    (R2)            ;1000,0010,0001,0010
014760  106113                          ROLB    (R3)            ;1000,0100,0001,0010
014762  151215                          BISB    (R2),(R5)       ;1000,0100,0001,1010
014764  131512                          BITB    (R5),(R2)       ;1000,0100,0001,1010
014766  001426                          BEQ     BIN1
014770  151314                          BISB    (R3),(R4)       ;1000,0100,0101,1010
014772  131413                          BITB    (R4),(R3)       ;1000,0100,0101,1010
014774  001423                          BEQ     BIN1
014776  105213                          INCB    (R3)            ;1000,0101,0101,1010
015000  121314                          CMPB    (R3),(R4)       ;1000,0101,0101,1010
015002  001020                          BNE     BIN1
015004  106113                          ROLB    (R3)            ;1000,1010,0101,1010
015006  121315                          CMPB    (R3),(R5)       ;1000,1010,0101,1010
015010  001015                          BNE     BIN1
015012  106212                          ASRB    (R2)            ;0100,1010,0101,1010
015014  131214                          BITB    (R2),(R4)       ;0100,1010,0101,1010
015016  001412                          BEQ     BIN1
015020  106015                          RORB    (R5)            ;0100,1010,0101,0101
015022  121415                          CMPB    (R4),(R5)       ;0100,1010,0101,0101
015024  001007                          BNE     BIN1
015026  105314                          DECB    (R4)            ;0100,1010,0100,0101
015030  141214                          BICB    (R2),(R4)       ;0100,1010,0000,0101
015032  001004                          BNE     BIN1
015034  111314                          MOVB    (R3),(R4)       ;0100,1010,1010,0101
015036  106213                          ASRB    (R3)            ;0100,0101,1010,0101
015040  141315                          BICB    (R3),(R5)       ;0100,0101,1010,0101
015042  001401                          BEQ     .+4
015044  104000                  BIN1:   HLT
                                ;;*****************************************************************
                                ;*TEST 22       CHECK BINARY WORD OPS USING ADDRESS MODE 2 & 4
                                ;;*****************************************************************
015046  112737  000022  001202  TST22:  MOVB    #22,@#$TSTNM            ;LOAD TEST NUMBER
015054  000004                          SCOPE
015056  012704  014714                  MOV     #BICB1A+6,R4
015062  012702  014712                  MOV     #BICB1A+4,R2
015066  063702  001506                  ADD     @#FACTOR,R2
015072  063704  001506                  ADD     @#FACTOR,R4
015076  010405                          MOV     R4,R5           ;SET DESTINATION REGISTER
015100  012715  000001                  MOV     #1,(R5)
015104  012712  177777                  MOV     #-1,(R2)
015110  000257                          CCC
015112  000262                          SEV
015114  062225                          ADD     (R2)+,(R5)+     ;(R2)=177777,(R5)=000000, CC=0101
015116  103002                          BCC     ADD2
015120  102401                          BVS     ADD2
015122  001401                          BEQ     .+4
015124  104000                  ADD2:   HLT

015126  000262                          SEV                     ;SET V
```

```
015134  103002                          BCC     CMP2
015136  102401                          BVS     CMP2
015140  100401                          BMI     .+4
015142  104000                  CMP2:   HLT

015144  054225                          BIS     -(R2),(R5)+             ;(R2)=177777,(R5)=177777,CC=1001
015146  103001                          BCC     BIS2
015150  100401                          BMI     .+4
015152  104000                  BIS2:   HLT
015154  000277                          SCC
015156  000244                          CLZ
015160  162245                          SUB     (R2)+,-(R5)             ;(R2)=177777,(R5)=000000, CC=0100
015162  103402                          BCS     SUB2
015164  102401                          BVS     SUB2
015166  001401                          BEQ     .+4
015170  104000                  SUB2:   HLT

015172  005442                          NEG     -(R2)                   ;(R2)=000001
015174  005115                          COM     (R5)                    ;(R5)=177777
015176  000277                          SCC
015200  000250                          CLN
015202  042225                          BIC     (R2)+,(R5)+             ;(R2)=000001,(R5)=177776, CC=1001
015204  103003                          BCC     BIC2
015206  102402                          BVS     BIC2
015210  001401                          BEQ     BIC2
015212  100401                          BMI     .+4
015214  104000                  BIC2:   HLT

015216  012742  125252                  MOV     #125252,-(R2)
015222  012245                          MOV     (R2)+,-(R5)
015224  005125                          COM     (R5)+                   ;(R5)=052525
015226  000262                          SEV
015230  034245                          BIT     -(R2),-(R5)             ;(R2)=125252,(R5)=052525, CC=0101
015232  103002                          BCC     BITT2
015234  102401                          BVS     BITT2
015236  001401                          BEQ     .+4
015240  104000                  BITT2:  HLT

015242  000262                          SEV
015244  052225                          BIS     (R2)+,(R5)+             ;(R2)=125252,(R5)=177777, CC=1001
015246  103002                          BCC     BIS2A
015250  102401                          BVS     BIS2A
015252  100401                          BMI     .+4
015254  104000                  BIS2A:  HLT

015256  042745  125252                  BIC     #125252,-(R5)           ;(R5)=052525
015262  005125                          COM     (R5)+                   ;(R5)=125252
015264  024245                          CMP     -(R2),-(R5)
015266  001401                          BEQ     .+4
015270  104000                          HLT

015272  005012                          CLR     (R2)
015274  005122                          COM     (R2)+                   ;(R2)=177777
015276  162742  000001                  SUB     #1,-(R2)                ;(R2)=177776, CC=1000
015302  103402                          BCS     SUB2A
```

```
015306  100401                          BMI     .+4
015310  104000                  SUB2A:  HLT
015312  010702                          MOV     PC,R2           ;GET CURRENT PC
015314  010205                          MOV     R2,R5           ;MOVE TO R5
015316  124245                  1$:     CMPB    -(R2),-(R5)     ;COMPARE ALL PREVIOUS MEMORY ADDRESSES
015320  001401                          BEQ     .+4
015322  104000                          HLT                     ;ERROR!
015324  020237  001512                  CMP     R2,@#FRSTAD     ;CHECK FOR LOW LIMIT
015330  001372                          BNE     1$
                                ;;*****************************************************************
                                ;*TEST 23        CHECK BINARY BYTE OPS USING ADDRESS MODE 2 & 4
                                ;;*****************************************************************
015332  112737  000023  001202  TST23:  MOVB    #23,@#$TSTNM            ;LOAD TEST NUMBER
015340  000004                          SCOPE
015342  000402                          BR      .+6             ;RESERVE TWO WORDS
015344  000000                          .WORD   0               ;SOURCE DATA
015346  000000                          .WORD   0               ;DESTINATION DATA
015350  010703                          MOV     PC,R3
015352  005743                          TST     -(R3)

                                ;FIRST CHECK AUTO INCREMENT/DECREMENT
015354  010300                          MOV     R3,R0
015356  010002                          MOV     R0,R2
015360  005302                          DEC     R2
015362  010604                          MOV     SP,R4
015364  010605                          MOV     SP,R5
015366  005745                          TST     -(R5)

015370  114046                          MOVB    -(R0),-(SP)
015372  020506                          CMP     R5,SP
015374  001021                          BNE     BINB
015376  020200                          CMP     R2,R0
015400  001017                          BNE     BINB
015402  122026                          CMPB    (R0)+,(SP)+
015404  020406                          CMP     R4,SP
015406  001014                          BNE     BINB
015410  020003                          CMP     R0,R3
015412  001012                          BNE     BINB
015414  154640                          BISB    -(SP),-(R0)
015416  020506                          CMP     R5,SP
015420  001007                          BNE     BINB
015422  020200                          CMP     R2,R0
015424  001005                          BNE     BINB
015426  142620                          BICB    (SP)+,(R0)+
015430  020406                          CMP     R4,SP
015432  001002                          BNE     BINB
015434  020003                          CMP     R0,R3
015436  001401                          BEQ     .+4
015440  104000                  BINB:   HLT
015442  010003                          MOV     R0,R3
015444  112743  000200                  MOVB    #200,-(R3)
015450  112743  000377                  MOVB    #377,-(R3)      ;(R3)=100377
015454  010304                          MOV     R3,R4
015456  112744  000177                  MOVB    #177,-(R4)
015462  112744  000000                  MOVB    #0,-(R4)        ;(R4)=077400
```

```
015470  104000                          HLT

015472  152324                          BISB    (R3)+,(R4)+             ;(R3)=100377,(R4)=077777
015474  100401                          BMI     .+4
015476  104000                          HLT

015500  122324                          CMPB    (R3)+,(R4)+
015502  103402                          BCS     CMPB2
015504  102001                          BVC     CMPB2
015506  100001                          BPL     .+4
015510  104000                  CMPB2:  HLT

015512  000261                          SEC
015514  134344                          BITB    -(R3),-(R4)
015516  103002                          BCC     BITB2
015520  102401                          BVS     BITB2
015522  001401                          BEQ     .+4
015524  104000                  BITB2:  HLT

015526  000244                          CLZ
015530  144344                          BICB    -(R3),-(R4)             ;(R3)=100377,(R4)=077400
015532  001401                          BEQ     .+4
015534  104000                          HLT
                                ;;*************************************************************
                                ;*TEST 24        CHECK BINARY WORD OPS USING ADDRESS MODES 3 & 5
                                ;;*************************************************************
015536  112737  000024  001202  TST24:  MOVB    #24,@#$TSTNM                    ;LOAD TEST NUMBER
015544  000004                          SCOPE
015546  000404                          BR      2$                      ;RESERVE SPACE FOR DATA AND ADDRESSES
015550  000000                          .WORD   0                       ;CONTAINS ADDRESS OF SOURCE DATA
015552  000000                          .WORD   0                       ;CONTAINS ADDRESS OF DEST DATA
015554  000000                          .WORD   0                       ;CONTAINS SOURCE DATA
015556  000000                          .WORD   0                       ;CONTAINS DEST DATA
015560  010701                  2$:     MOV     PC,R1
015562  010100                          MOV     R1,R0                   ;SET SCOPE PTR
015564  024040                          CMP     -(R0),-(R0)             ;ADJUST R0
015566  010005                          MOV     R0,R5                   ;R5 POINTS TO DEST DATA
015570  024545                          CMP     -(R5),-(R5)             ;SUB 4 FROM R5
015572  010015                          MOV     R0,(R5)                 ;R5 POINTS TO ADDRESS OF DEST DATA
015574  010502                          MOV     R5,R2
015576  010004                          MOV     R0,R4                   ;R4 POINTS TO DEST DATA
015600  005740                          TST     -(R0)
015602  010003                          MOV     R0,R3                   ;R3 POINTS TO SOURCE DATA
015604  010042                          MOV     R0,-(R2)                ;R2 POINTS TO ADDRESS OF SOURCE DATA
015606  005013                          CLR     (R3)                    ;PRESET SOURCE DATA
015610  005014                          CLR     (R4)                    ;PRESET DEST DATA

015612  000277                          SCC
015614  000244                          CLZ
015616  163235                          SUB     @(R2)+,@(R5)+           ;(R3)=000000,(R4)=000000, CC=0100
015620  103402                          BCS     SUB3
015622  102401                          BVS     SUB3
015624  001401                          BEQ     .+4
015626  104000                  SUB3:   HLT
```

```
015634  062755  000001                     ADD     #1,@-(R5)          ;(R4)=000001
015640  163235                             SUB     @(R2)+,@(R5)+      ;(R3)=100000,(R4)=100001, CC=1011
015642  103002                             BCC     SUB3A
015644  102001                             BVC     SUB3A
015646  100401                             BMI     .+4
015650  104000                   SUB3A:    HLT

015652  005414                             NEG     (R4)               ;(R4)=077777
015654  035255                             BIT     @-(R2),@-(R5)      ;(R3)=100000,(R4)=077777
015656  001401                             BEQ     .+4
015660  104000                             HLT
015662  023235                             CMP     @(R2)+,@(R5)+
015664  102401                             BVS     .+4
015666  104000                             HLT
015670  005152                             COM     @-(R2)
015672  000257                             CCC
015674  063255                             ADD     @(R2)+,@-(R5)
015676  102001                             BVC     ADD3
015700  100401                             BMI     .+4
015702  104000                   ADD3:     HLT
015704  000261                             SEC
015706  045235                             BIC     @-(R2),@(R5)+      ;(R3)=077777,(R4)=100000
015710  103001                             BCC     BIC3
015712  100401                             BMI     .+4
015714  104000                   BIC3:     HLT

015716  005155                             COM     @-(R5)             ;(R4)=077777
015720  023235                             CMP     @(R2)+,@(R5)+      ;(R3)=077777,(R4)=077777
015722  001401                             BEQ     .+4
015724  104000                             HLT
                                 ;;*****************************************************
                                 ;*TEST 25          CHECK BINARY BYTE OPS USING ADDRESS MODES 3 & 5
                                 ;;*****************************************************
015726  112737  000025  001202   TST25:    MOVB    #25,@#$TSTNM               ;LOAD TEST NUMBER
015734  000004                             SCOPE
015736  000406                             BR      1$                 ;RESERVE SPACE FOR ADDRESS AND DATA
015740  000000                             .WORD   0                  ;CONTAINS ADDRESS OF SOURCE DATA (EVEN BYTE)
015742  000000                             .WORD   0                  ;CONTAINS ADDRESS OF SOURCE DATA (ODD BYTE)
015744  000000                             .WORD   0                  ;CONTAINS ADDRESS OF DEST DATA (EVEN BYTE)
015746  000000                             .WORD   0                  ;CONTAINS ADDRESS OF DEST DATA (ODD BYTE)
015750  000000                             .WORD   0                  ;CONTAINS SOURCE DATA
015752  000000                             .WORD   0                  ;CONTAINS DEST DATA

015754  010700                   1$:       MOV     PC,R0
015756  024040                             CMP     -(R0),-(R0)        ;R0=ADDRESS OF DEST DATA
015760  010003                             MOV     R0,R3              ;R3    "       "
015762  010305                             MOV     R3,R5              ;R5    "       "
015764  005743                             TST     -(R3)              ;SUB 2 FROM R3
015766  010043                             MOV     R0,-(R3)           ;R3 POINTS TO ADDRESS OF DEST DATA
015770  005213                             INC     (R3)               ;ODD BYTE
015772  010043                             MOV     R0,-(R3)           ;EVEN BYTE
015774  010304                             MOV     R3,R4
015776  005740                             TST     -(R0)              ;R0=ADDRESS OF SOURCE DATA
016000  010044                             MOV     R0,-(R4)           ;R4 POINTS TO ADDRESS OF SOURCE DATA
016002  005214                             INC     (R4)               ;ODD BYTE
```

```

016006  000261                                  SEC                             ;SET CARRY
016010  012734  177001                          MOV     #177001,@(R4)+
016014  112734  000200                          MOVB    #200,@(R4)+             ;SOURCE DATA=100001
016020  115433                                  MOVB    @-(R4),@(R3)+
016022  115433                                  MOVB    @-(R4),@(R3)+           ;DEST DATA=000600
016024  103401                                  BCS     .+4
016026  104000                                  HLT                             ;ERROR! MOV DOES AFFECT C BIT IN PSW
016030  022715  000600                          CMP     #600,(R5)               ;CHECK DEST DATA
016034  001401                                  BEQ     .+4
016036  104000                                  HLT                             ;ERROR! INCORRECT RESULT
016040  024343                                  CMP     -(R3),-(R3)             ;POINT R4 BACK TO EVEN BYTE
016042  153433                                  BISB    @(R4)+,@(R3)+
016044  153433                                  BISB    @(R4)+,@(R3)+           ;DEST DATA=100601
016046  022715  100601                          CMP     #100601,(R5)            ;CHECK RESULT
016052  001401                                  BEQ     .+4
016054  104000                                  HLT                             ;ERROR! INCORRECT DEST DATA AFTER BISB
016056  145453                                  BICB    @-(R4),@-(R3)
016060  145453                                  BICB    @-(R4),@-(R3)
016062  133433                                  BITB    @(R4)+,@(R3)+
016064  001002                                  BNE     BITB3
016066  135433                                  BITB    @-(R4),@(R3)+
016070  001001                                  BNE     .+4
016072  104000                          BITB3:  HLT

016074  123453                                  CMPB    @(R4)+,@-(R3)
016076  001002                                  BNE     CMPB3
016100  123453                                  CMPB    @(R4)+,@-(R3)
016102  001401                                  BEQ     .+4
016104  104000                          CMPB3:  HLT
                                        ;;**********************************************************
                                        ;*TEST 26       CHECK BINARY OPS USING ADDRESS MODE 6
                                        ;;**********************************************************
016106  112737  000026  001202          TST26:  MOVB    #26,@#$TSTNM            ;LOAD TEST NUMBER
016114  000004                                  SCOPE
016116  000402                                  BR      .+6                     ;RESERVE TWO LOCATIONS
016120  000000                          SDATA:  .WORD   0                       ;RESERVED FOR SOURCE DATA
016122  000000                          DDATA:  .WORD   0                       ;RESERVED FOR DESTINATION DATA

016124  013702  001506                          MOV     @#FACTOR,R2             ;GET RELOCATION FACTOR AND USE AS AN
016130  010205                                  MOV     R2,R5                   ;INDEX VALUE TO POINT TO DATA
016132  005065  016122                          CLR     DDATA(5)                ;PRESET DESTINATION DATA
016136  012762  000001  016120                  MOV     #1,SDATA(2)             ;THIS ROUTSINE PUT A 1 BIT INTO EVERY
016144  056265  016120  016122          1$:     BIS     SDATA(2),DDATA(5)               ;OTHER BIT POSITION IN THE DEST-
016152  006362  016120                          ASL     SDATA(2)                        ;INATION ADDRESS (52525)
016156  006362  016120                          ASL     SDATA(2)
016162  103370                                  BCC     1$
016164  022765  052525  016122                  CMP     #52525,DDATA(5) ;CHECK RESULT
016172  001401                                  BEQ     .+4
016174  104000                                  HLT                             ;ERROR! INCORRECT RESULT
016176  012762  177777  016120                  MOV     #-1,SDATA(2)
016204  046562  016122  016120                  BIC     DDATA(5),SDATA(2)               ;SOURCE DATA=125252
016212  036265  016120  016122                  BIT     SDATA(2),DDATA(5)
016220  001401                                  BEQ     .+4
016222  104000                                  HLT                             ;ERROR! BIT INST FAILED
```

```
016230  026265  016120  016122          CMP     SDATA(2),DDATA(5)
016236  001401                          BEQ     .+4

016240  104000                          HLT                             ;ERROR! CMP INST FAILED
016242  000257                          CCC
016244  066265  016120  016122          ADD     SDATA(2),DDATA(5)
016252  103002                          BCC     ADD6
016254  102001                          BVC     ADD6
016256  100001                          BPL     .+4
016260  104000                  ADD6:   HLT

016262  006362  016120                  ASL     SDATA(2)                ;SDATA=52524
016266  166265  016120  016122          SUB     SDATA(2),DDATA(5)
016274  103401                          BCS     SUB6
016276  001401                          BEQ     .+4
016300  104000                  SUB6:   HLT

016302  112700  000377                  MOVB    #377,R0                 ;R0=177777 (MOVB %R EXTENDS SIGN)
016306  010062  016120                  MOV     R0,SDATA(2)
016312  012765  177777  016122          MOV     #-1,DDATA(5)
016320  166500  016122                  SUB     DDATA(5),R0
016324  001401                          BEQ     .+4
016326  104000                          HLT
016330  066265  016120  016122  1$:     ADD     SDATA(2),DDATA(5)
016336  006362  016120                  ASL     SDATA(2)
016342  005162  016120                  COM     SDATA(2)
016346  036265  016120  016122          BIT     SDATA(2),DDATA(5)
016354  001401                          BEQ     .+4
016356  104000                          HLT
016360  005162  016120                  COM     SDATA(2)
016364  026265  016120  016122          CMP     SDATA(2),DDATA(5)
016372  001401                          BEQ     .+4
016374  104000                          HLT
016376  026200  016120                  CMP     SDATA(2),R0
016402  001352                          BNE     1$
                                ;*****************************************************************
                                ;*TEST 27       CHECK BINARY BYTE OPS USING ADDRESS MODE 6
                                ;*****************************************************************
016404  112737  000027  001202  TST27:  MOVB    #27,@#$TSTNM            ;LOAD TEST NUMBER
016412  000004                          SCOPE
                                ;NOTE: SDATAB(2), AND DDATAB(4) REFERENCE EVEN BYTE OF SOURCE & DEST DATA
                                ;AND SDATAB(3), AND DDATAB(5) FEFERENCE ODD BYTE OF SOURCE & DEST DATA

016414  013702  001506                  MOV     @#FACTOR,R2             ;GET INDEX VALUE
016420  010204                          MOV     R2,R4                   ;R2 FOR SOURCE EVEN BYTE INDEX, R4 FOR
016422  010403                          MOV     R4,R3                   ;DEST ODD BYTE, R3 FOR SOURCE EVEN
016424  005203                          INC     R3                      ;AND R5 FOR DEST ODD BYTE
016426  010305                          MOV     R3,R5
016430  000261                          SEC                                     ;SET CARRY
016432  012762  125252  016554          MOV     #125252,SDATAB(2)
016440  112763  177125  016554          MOVB    #177125,SDATAB(3)               ;SOURCE DATA = 052652
016446  016264  016554  016556          MOV     SDATAB(2),DDATAB(4)
016454  052764  125125  016556          BIS     #125125,DDATAB(4)               ;DEST DATA = 177777
016462  136263  016554  016554          BITB    SDATAB(2),SDATAB(3)
016470  001401                          BEQ     .+4
```

```

016474  146264  016554  016556                BICB    SDATAB(2),DDATAB(4)
016502  103401                                BCS     .+4
016504  104000                                HLT                             ;ERROR MOV,BIS,BIT;BIC DO NOT AFFECT 'C'
016506  126364  016554  016556                CMPB    SDATAB(3),DDATAB(4)
016514  001401                                BEQ     .+4
016516  104000                                HLT

016520  146365  016554  016556                BICB    SDATAB(3),DDATAB(5)
016526  126265  016554  016556                CMPB    SDATAB(2),DDATAB(5)
016534  001401                                BEQ     .+4
016536  104000                                HLT

016540  136564  016556  016556                BITB    DDATAB(5),DDATAB(4)
016546  001401                                BEQ     .+4
016550  104000                                HLT
016552  000412                                BR      UB7                     ;RESERVE TWO WORDS
016554  000000                       SDATAB:  .WORD   0                       ;RESERVED FOR SOURCE DATA
016556  000000                       DDATAB:  .WORD   0                       ;RESERVED FOR DEST DATA

                                     ;;*************************************************************
                                     ;*TEST 30        CHECK BINARY WORD OPS USING ADDRESS MODE 7
                                     ;*        R2=ADDRESS OF SOURCE DATA, AND R3= ADDRESS OF DEST DATA
                                     ;;*************************************************************
016560  112737  000030  001202       TST30:   MOVB    #30,@#$TSTNM            ;LOAD TEST NUMBER
016566  000004                                SCOPE
016570  000000                       SBIN7:   .WORD   0                       ;CONTAINS ADDRESS OF SOURCE DATA
016572  000000                       DBIN7:   .WORD   0                       ;CONTAINS ADDRESS OF DEST DATA
016574  000000                                .WORD   0                       ;CONTAINS SOURCE DATA
016576  000000                                .WORD   0                       ;CONTAINS DEST DATA

016600  010700                       UB7:     MOV     PC,R0
016602  024040                                CMP     -(R0),-(R0)
016604  010002                                MOV     R0,R2
016606  024242                                CMP     -(R2),-(R2)
016610  010012                                MOV     R0,(R2)
016612  010203                                MOV     R2,R3
016614  024043                                CMP     -(R0),-(R3)
016616  010013                                MOV     R0,(R3)

016620  000261                                SEC
016622  012777  100000  177740                MOV     #100000,@SBIN7          ;SOURCE DATA = 100000
016630  017777  177734  177734                MOV     @SBIN7,@DBIN7           ;DEST DATA = 100000
016636  103001                                BCC     MOV7
016640  100401                                BMI     .+4
016642  104000                       MOV7:    HLT
016644  006377  177722                        ASL     @DBIN7                  ;DEST DATA = 000000
016650  102001                                BVC     .+4
016652  001401                                BEQ     .+4
016654  104000                                HLT

016656  027777  177706  177706                CMP     @SBIN7,@DBIN7           ;(R2)=100000,(R3)=000000
016664  103402                                BCS     CMP7
016666  102401                                BVS     CMP7
016670  100401                                BMI     .+4
```

```
3714
```

```
016702  103003                                  BCC     SUB7
016704  102002                                  BVC     SUB7
016706  001401                                  BEQ     SUB7
016710  100401                                  BMI     .+4
016712  104000                          SUB7:   HLT

016714  006277  177650                          ASR     @SBIN7                  ;(R2)=140000
016720  067777  177644  177644                  ADD     @SBIN7,@DBIN7           ;(R2)=140000,(R3)=040000
016726  103003                                  BCC     ADD7
016730  102002                                  BVC     ADD7
016732  001401                                  BEQ     ADD7
016734  100001                                  BPL     .+4
016736  104000                          ADD7:   HLT

016740  047777  177624  177624                  BIC     @SBIN7,@DBIN7           ;(R2)=140000,(R3)=000000
016746  001401                                  BEQ     .+4
016750  104000                                  HLT

016752  057777  177612  177612                  BIS     @SBIN7,@DBIN7           ;(R2)=140000,(R3)=140000
016760  100401                                  BMI     .+4
016762  104000                                  HLT

016764  027777  177600  177600                  CMP     @SBIN7,@DBIN7
016772  001401                                  BEQ     .+4
016774  104000                                  HLT
                                        ;;*****************************************************************
                                        ;*TEST 31        SOME MISCELLANEOUS OPERATIONS INVOLVING THE PC
                                        ;*               NOTE: NONE OF THESE OPERATIONS SHOULD AFFECT THE PC
                                        ;;*****************************************************************
016776  112737  000031  001202          TST31:  MOVB    #31,@#$TSTNM                    ;LOAD TEST NUMBER
017004  000004                                  SCOPE
017006  005000                                  CLR     R0
017010  005067  000072                          CLR     1$
017014  010707                                  MOV     PC,PC
017016  120707                                  CMPB    PC,PC
017020  030707                                  BIT     PC,PC
017022  060007                                  ADD     R0,PC
017024  105707                                  TSTB    PC
017026  005507                                  ADC     PC
017030  021007                                  CMP     (R0),PC
017032  131007                                  BITB    (R0),PC
017034  062707  000000                          ADD     #0,PC
017040  023707  001506                          CMP     @#FACTOR,PC
017044  133707  001506                          BITB    @#FACTOR,PC
017050  000240                                  NOP
                                        ;THE NEXT TWO INSTRUCTION CAUSE THE PROGRAM TO JUMP TO THE UNRELOCATED
                                        ;CODE AND TO RETURN ON THE FOLLOWING INST (IF THE CODE IS RELOCATED)
017052  163707  001506                          SUB     @#FACTOR,PC     ;JUMPS TO UNRELOCATED CODE
017056  063707  001506                          ADD     @#FACTOR,PC     ;RETURNS
017062  000240                                  NOP
017064  024607                                  CMP     -(SP),PC
017066  132607                                  BITB    (SP)+,PC
017070  026707  000012                          CMP     1$,PC
017074  166707  000006                          SUB     1$,PC
017100  046707  000002                          BIC     1$,PC
```

```
017106  000000                          1$:     0
017110  000004                          RELE2:  SCOPE
017112  010702                                  MOV     PC,R2
017114  062702  000012                          ADD     #12,R2
017120  012707  034242                          MOV     #RELOC,PC               ;GO RELOCATE PROGRAM CODE
017124  000000                          REL22:  .WORD   0
                                        ;2222222222222 LAST ADDRESS OF CODE TO BE RELOCATED 22222222222

                                        ;;****************************************************************
                                        ;;*TEST 32      CHECK BINARY BYTE OPS USING ADDRESS MODE 0
                                        ;;****************************************************************
017126  112737  000032  001202          TST32:  MOVB    #32,@#$TSTNM            ;LOAD TEST NUMBER
017134  012767  000001  162154                  MOV     #1,$TIMES               ;;DO 1 ITERATION
017142  000004                                  SCOPE

                                                .SBTTL  START OF SECTION 3
                                        ;3333333333333 FIRST ADDRESS TO BE RELOCATED 333333333
017144  010700                          REL3:   MOV     PC,R0                   ;GET PC
017146  005740                                  TST     -(R0)                   ;R0 CONTAINS THE ADDRESS OF REL3
017150  010037  001512                          MOV     R0,@#FRSTAD             ;SAVE
017154  010700                                  MOV     PC,R0                   ;GET CURRENT PC
017156  162700  017156                          SUB     #.,R0                   ;SUBTRACT RELOCATION FACTOR
017162  010037  001506                          MOV     R0,@#FACTOR             ;SAVE RELOCATION FACTOR
017166  010737  001212                          MOV     PC,@#$LPERR             ;SET LOOP ADDRESS
017172  062737  000030  001212                  ADD     #30,@#$LPERR            ;ADJUST
017200  013737  001212  001210                  MOV     @#$LPERR,@#$LPADR
017206  105737  001502                          TSTB    @#NEXEC                 ;BR IF TEST CODE TO BE EXECUTED
017212  001402                                  BEQ     .+6
017214  000167  002250                          JMP     RELE3
017220  012703  125252                          MOV     #125252,R3
017224  010304                                  MOV     R3,R4                   ;R3=R4=125252
017226  140304                                  BICB    R3,R4                   ;R3=125252,R4=125000
017230  022704  125000                          CMP     #125000,R4              ;CHECK RESULT
017234  001401                                  BEQ     .+4
017236  104000                                  HLT

017240  005004                                  CLR     R4                      ;R3=125252,R4=0
017242  150304                                  BISB    R3,R4                   ;R3=125252,R4=000252
017244  022704  000252                          CMP     #252,R4                 ;CHECK RESULT
017250  001401                                  BEQ     .+4
017252  104000                                  HLT

017254  110404                                  MOVB    R4,R4                   ;R4=177652
017256  022704  177652                          CMP     #177652,R4              ;CHECK RESULT
017262  001401                                  BEQ     .+4
017264  104000                                  HLT

017266  132704  177525                          BITB    #177525,R4
017272  001401                                  BEQ     .+4
017274  104000                                  HLT

017276  105104                                  COMB    R4                      ;R4=177525
017300  110404                                  MOVB    R4,R4                   ;R4=000125
017302  022704  000125                          CMP     #125,R4                 ;CHECK RESULT
017306  001401                                  BEQ     .+4
```

```
 017312  150304                           BISB    R3,R4             ;R3=125252,R4=000377
 017314  105204                           INCB    R4
 017316  001401                           BEQ     .+4
 017320  104000                           HLT
                                  ;;*************************************************************
                                  ;*TEST 33        CHECK BINARY BYTE OPS USING ADDRESS MODE 7
                                  ;;*************************************************************
 017322  112737  000033  001202   TST33:  MOVB    #33,@#$TSTNM            ;LOAD TEST NUMBER
 017330  000004                           SCOPE
 017332  000406                           BR      BINB7             ;RESERVE SPACE FOR ADDRESSES & DATA
 017334  000000                   SBINB7: .WORD   0                 ;CONTAINS ADDRESS OF SOURCE EVEN BYTE
 017336  000000                           .WORD   0                 ;CONTAINS ADDRESS OF SOURCE ODD BYTE
 017340  000000                           .WORD   0                 ;CONTAINS ADDRESS OF DEST EVEN BYTE
 017342  000000                           .WORD   0                 ;CONTAINS ADDRESS OF DEST ODD BYTE
 017344  000000                   DBINB7: .WORD   0                 ;CONTAINS SOURCE DATA
 017346  000000                           .WORD   0                 ;CONTAINS DEST DATA

 017350  010700                   BINB7:  MOV     PC,R0
 017352  024040                           CMP     -(R0),-(R0)       ;R0 = ADDRESS OF DEST DATA
 017354  010060  177772                   MOV     R0,-6(R0)         ;LOAD ADDRES OF DEST EVEN BYTE DATA
 017360  010060  177774                   MOV     R0,-4(R0)
 017364  005260  177774                   INC     -4(R0)            ;LOAD ADDRESS OF DEST ODD BYTE DATA
 017370  005740                           TST     -(R0)             ;R0=ADDRESS OF SOURCE DATA
 017372  010060  177770                   MOV     R0,-10(R0)        ;LOAD ADDRESS OF SOURCE EVEN BYTE DATA
 017376  010060  177772                   MOV     R0,-6(R0)
 017402  005260  177772                   INC     -6(R0)            ;LOAD ADDRESS OF SOURCE ODD BYTE DATA

 017406  005002                           CLR     R2                ;SET INDEX REGISTERS
 017410  012703  000002                   MOV     #2,R3             ;@SBINB7(2);@SBINB7(3) REFERENCE EVEN &
 017414  012704  177774                   MOV     #-4,R4            ;ODD BYTE SOURCE DATA; @DBINB7(4);@DBINB7(5)
 017420  012705  177776                   MOV     #-2,R5            ;REFERENCE DEST EVEN& ODD BYTE DATA


 017424  005020                           CLR     (R0)+             ;PRESET SOURCE DATA
 017426  005010                           CLR     (R0)              ;PRESET DEST DATA
 017430  013746  001506                   MOV     @#FACTOR,-(SP)    ;GET RELOCATION FACTOR
 017434  061602                           ADD     (SP),R2           ;AND ADD TO INDEX VALUES
 017436  061603                           ADD     (SP),R3
 017440  061604                           ADD     (SP),R4
 017442  062605                           ADD     (SP)+,R5

 017444  112773  177777  017334           MOVB    #-1,@SBINB7(3)    ;SRC DATA = 177400
 017452  132772  000377  017334           BITB    #377,@SBINB7(2)   ;CHECK THAT EVEN BYTE WAS NOT AFFECTED
 017460  001401                           BEQ     .+4               ;BY MOVB INSTRUCTION
 017462  104000                           HLT

 017464  157374  017334  017344           BISB    @SBINB7(3),@DBINB7(4)
 017472  105274  017344                   INCB    @DBINB7(4)        ;CHECK THAT BIS SET ALL BITS
 017476  001401                           BEQ     .+4
 017500  104000                           HLT

 017502  105375  017344                   DECB    @DBINB7(5)        ;DEST DATA = 177400
 017506  005274  017344                   INC     @DBINB7(4)        ;DEST DATA = 177401
 017512  127375  017334  017344           CMPB    @SBINB7(3),@DBINB7(5)
```

```
017522  104000                                  HLT

017524  147375  017334  017344                  BICB    @SBINB7(3),@DBINB7(5)
017532  001401                                  BEQ     .+4
017534  104000                                  HLT

017536  105073  017334                          CLRB    @SBINB7(3)          ;SRC DATA = 000000
                                        ;THIS ROUTINE SETS ALL BITS IN THE SOURCE ODD BYTE BY BISING A BIT FROM
                                        ;THE DEST EVEN BYTE INTO THE SOURCE ODD BYTE
017542  157473  017344  017334  BIS7:   BISB    @DBINB7(4),@SBINB7(3)
017550  106174  017344                  ROLB    @DBINB7(4)
017554  103372                          BCC     BIS7
017556  022772  177400  017334          CMP     #177400,@SBINB7(2)          ;CHECK RESULT
017564  001401                          BEQ     .+4
017566  104000                          HLT

017570  000372  017334                  SWAB    @SBINB7(2)          ;SRC DATA = 000377
017574  112775  000200  017344          MOVB    #200,@DBINB7(5) ;DEST DATA = 100000

017602  147572  017344  017334  BIC7:   BICB    @DBINB7(5),@SBINB7(2)
017610  106075  017344                  RORB    @DBINB7(5)
017614  103372                          BCC     BIC7
017616  005772  017334                  TST     @SBINB7(2)
017622  001401                          BEQ     .+4
017624  104000                          HLT

017626  012702  000001          OAERR:  MOV     #1,R2               ;LOAD R2 WITH ODD #
017632  010703                          MOV     PC,R3
017634  000401                          BR      .+4                 ;RESERVE SPACE FOR A WORD
017636  000000                          .WORD   0                   ;WILL CONTAIN AN ODD ADDRESS
017640  005723                          TST     (R3)+               ;STEP R3 TO POINT TO WORD ABOVE
017642  010313                          MOV     R3,(R3)
017644  005213                          INC     (R3)                ;AND MAKE ODD
017646  012737  017774  000004          MOV     #1$,@#ERRVEC        ;SET ODD ADDRESS & RESERVED INSTRUCTION
017654  063737  001506  000004          ADD     @#FACTOR,@#ERRVEC
017662  013737  000004  000010          MOV     @#ERRVEC,@#RESVEC           ;TO TRAP TO 1$ BELOW

017670  000277                          SCC                         ;SET ALL CC'S
017672  160212                          SUB     R2,(R2)
017674  104000                          HLT
017676  060222                          ADD     R2,(R2)+
017700  104000                          HLT
017702  006342                          ASL     -(R2)
017704  104000                          HLT
017706  106512                          MFPD    (R2)                ;NOTE: MAY BE RESERVED
017710  104000                          HLT
017712  170412                          CLRF    (R2)
017714  104000                          HLT
017716  042202                          BIC     (R2)+,R2
017720  104000                          HLT
017722  164202                          SUB     -(R2),R2
017724  104000                          HLT
017726  155202                          BISB    @-(R2),R2
017730  104000                          HLT
017732  105532                          ADCB    @(R2)+
017736  163302                          SUB     @(R3)+,R2
017740  104000                          HLT
017742  005733                          TST     @(R3)+
017744  104000                          HLT
017746  106533                          MFPD    @(R3)+
017750  104000                          HLT
017752  170453                          CLRD    @-(R3)
017754  104000                          HLT
017756  137702  177775                  BITB    @.+1,R2
017762  104000                          HLT
017764  105477  177773                  NEGB    @.-1
017770  104000                          HLT
017772  000406                          BR      2$

017774  062716  000002          1$:     ADD     #2,(SP)                 ;ADJUST RETURN PC
020000  052766  000017  000002          BIS     #17,2(SP)               ;SET CONDITION CODES ON RETURN
020006  000002                          RTI

020010  012706  000700          2$:     MOV     #SUPSTK,SP              ;RESET STACK PTR
020014  012737  053442  000004          MOV     #ERPRT,@#ERRVEC         ;RESET TIME OUT VECTOR
020022  012737  053370  000010          MOV     #RESERR,@#RESVEC
                                ;;*************************************************************
                                ;*TEST 34          CHECK JUMP INSTRUCTIONS
                                ;;*************************************************************
020030  112737  000034  001202  TST34:  MOVB    #34,@#$TSTNM                    ;LOAD TEST NUMBER
020036  000004                          SCOPE
020040  010700                          MOV     PC,R0
020042  062700  000012                  ADD     #12,R0                  ;SET ADDRESS FOR JMP INST
020046  000277                          SCC                             ;SET CC'S
020050  000110                          JMP     (R0)
020052  000402                          BR      .+6
020054  000250                          CLN                             ;JMP INST JUMPS HERE
020056  000775                          BR      .-4

020060  103003                          BCC     JMP1
020062  102002                          BVC     JMP1
020064  001001                          BNE     JMP1
020066  100001                          BPL     .+4
020070  104000                  JMP1:   HLT                             ;ERROR! INCORRECT CC'S AFTER JMP

020072  005002                          CLR     R2                      ;SET INDICATOR
020074  010703                          MOV     PC,R3
020076  000401                          BR      .+4                     ;RESERVE WORD FOR JMP ADDRESS
020100  000000                          .WORD   0                       ;CONTAINS ADDRESS FOR JMP INST
020102  005723                          TST     (R3)+
020104  010313                          MOV     R3,(R3)
020106  010300                          MOV     R3,R0
020110  062713  000022                  ADD     #22,(R3)                ;(R3) IS JMP ADDRESS
020114  010300                          MOV     R3,R0
020116  000133                          JMP     @(R3)+                  ;JUMP TO ADDRESS CONTAINED IN R3
020120  000402                          BR      .+6
020122  005102                          COM     R2                      ;COMPLEMENT INDICATOR
020124  000775                          BR      .-4
020126  005202                          INC     R2                      ;CHECK INDICATOR
020130  001003                          BNE     JMP3
```

```
020134  020003                             CMP    R0,R3             ;CHECK AUTO-INC R3
020136  001401                             BEQ    .+4
020140  104000                    JMP3:    HLT

020142  005002                             CLR    R2                ;SET INDICATOR
020144  010704                             MOV    PC,R4             ;SET UP JMP REGISTER
020146  010400                             MOV    R4,R0             ;SET UP CHECK REGISTER
020150  000402                             BR     1$
020152  005102                             COM    R2                ;COMPLEMENT INDICATOR
020154  000403                             BR     2$
020156  022424                    1$:      CMP    (R4)+,(R4)+
020160  005724                             TST    (R4)+             ;R4=JMP ADDRESS
020162  000144                             JMP    -(R4)             ;USE R4 AS ADDRESS
020164  005202                    2$:      INC    R2                ;CHECK INDICATOR
020166  001003                             BNE    JMP4
020170  022020                             CMP    (R0)+,(R0)+
020172  020004                             CMP    R0,R4             ;CHECK AUTO-DEC R4
020174  001401                             BEQ    .+4
020176  104000                    JMP4:    HLT

020200  010703                             MOV    PC,R3
020202  000401                             BR     .+4               ;RESERVE WORD FOR JMP ADDRESS
020204  000000                    1$:      .WORD  0                 ;CONTAINS JUMP ADDRESS
020206  005723                             TST    (R3)+
020210  010313                             MOV    R3,(R3)
020212  062723  000016                     ADD    #16,(R3)+
020216  010300                             MOV    R3,R0             ;LOAD CHECK REGISTER
020220  000402                             BR     3$
020222  005102                    2$:      COM    R2
020224  000401                             BR     4$
020226  000153                    3$:      JMP    @-(R3)            ;JUMP TO 2$ VIA 1$ ABOVE
020230  005202                    4$:      INC    R2                ;CHECK INDICATOR
020232  001003                             BNE    JMP5
020234  005740                             TST    -(R0)
020236  020003                             CMP    R0,R3             ;CHECK AUTO-DEC R3
020240  001401                             BEQ    .+4
020242  104000                    JMP5:    HLT

020244  000402                             BR     2$
020246  005102                    1$:      COM    R2                ;COMPLEMENT INDICATOR
020250  000402                             BR     3$
020252  000167  177770            2$:      JMP    1$
020256  005202                    3$:      INC    R2
020260  001401                             BEQ    .+4
020262  104000                    JMP6:    HLT

020264  012767  020302  000020             MOV    #1$,7$            ;SET UP JMP ADDRESS
020272  063767  001506  000012             ADD    @#FACTOR,7$       ;ADD RELOCATION FACTOR
020300  000402                             BR     2$                ;GO TO JMP  @7$ INST
020302  005102                    1$:      COM    R2                ;COMPLEMENT INDICATOR
020304  000403                             BR     3$                ;GO TO CHECK ROUTINE
020306  000177  000000            2$:      JMP    @7$               ;JMP TO 1$ ABOVE VIA 7$
020312  000000                    7$:      .WORD  0                 ;CONTAINS JMP ADDRESS
020314  005202                    3$:      INC    R2                ;CHECK INDICATOR
020316  001401                             BEQ    .+4
```

```
                                        ;;*********************************************************************
                                        ;*TEST 35        CHECK JSR INSTRUCTIONS
                                        ;;*********************************************************************
020322  112737  000035  001202          TST35:  MOVB    #35,@#$TSTNM            ;LOAD TEST NUMBER
020330  000004                                  SCOPE
020332  013705  001506                  JSR1:   MOV     @#FACTOR,R5         ;GET RELOCATION FACTOR
020336  012702  020370                          MOV     #3$,R2              ;FORM DEST ADRS
020342  060502                                  ADD     R5,R2               ;ADD RELOCATION FACTOR
020344  000277                                  SCC                         ;PRESET CC'S
020346  000242                                  CLV
020350  004512                                  JSR     R5,(R2)             ;GO TO 3$ VIA R2
020352  005702                          1$:     TST     R2                  ;CHECK INDICATOR
020354  001017                                  BNE     4$                  ;R2 SHOULD=0
020356  023705  001506                          CMP     @#FACTOR,R5         ;CHECK THAT RTS  R5 RESTORED R5
020362  001014                                  BNE     4$
020364  000414                                  BR      JSR3                ;GO TO NEXT TEST
020366  000205                          2$:     RTS     R5                  ;RETURN FROM SUBROUTINE
020370  103011                          3$:     BCC     4$                  ;CHECK THAT JSR DID NOT
020372  102410                                  BVS     4$
020374  001007                                  BNE     4$                  ;AFFECT CC'S
020376  100006                                  BPL     4$
020400  005002                                  CLR     R2                  ;CLEAR INDICATOR
020402  012704  020352                          MOV     #1$,R4              ;GET UNRELOCATED RETURN ADDRESS
020406  061604                                  ADD     (SP),R4             ;ADD RELOCATION FACTOR (OLD R5)
020410  020405                                  CMP     R4,R5               ;CHECK THAT OLD R5 WAS PLACED ON THE
020412  001765                                  BEQ     2$                  ;STACK & THAT NEW R5 CONTAINS RETURN PC
020414  104000                          4$:     HLT                         ;ERROR! ABOVE

                                        ;CHECK JSR INSTRUCTION ADDRESS MODE 3
020416  013704  001506                  JSR3:   MOV     @#FACTOR,R4         ;GET RELOCATION FACTOR
020422  005000                                  CLR     R0                  ;SET INDICATOR
020424  012705  020444                          MOV     #1$,R5
020430  060405                                  ADD     R4,R5               ;SET UP JSR DEFERRED ADRS
020432  010502                                  MOV     R5,R2
020434  012715  020462                          MOV     #5$,(R5)
020440  060415                                  ADD     R4,(R5)             ;(R5)=DEST ADRS
020442  000401                                  BR      2$                  ;RESERVE WORD FOR ADDRESS
020444  000000                          1$:     .WORD   0                   ;CONTAINS DEST ADRS FOR JSR
020446  004435                          2$:     JSR     R4,@(R5)+           ;JSR TO 5$ VIA 1$ ABOVE
020450  005200                          3$:     INC     R0                  ;CHECK INDICATOR
020452  001013                                  BNE     6$
020454  000413                                  BR      JSR4
020456  005100                          4$:     COM     R0                  ;COMPLEMENT INDICATOR
020460  000204                                  RTS     4                   ;RETURN FROM SUBROUTINE
020462  012703  020450                  5$:     MOV     #3$,R3              ;GET UNRELOCATED RETURN ADDRESS
020466  061603                                  ADD     (SP),R3             ;ADD RELOCATION FACTOR (OLD R4)
020470  020403                                  CMP     R4,R3
020472  001003                                  BNE     6$
020474  005722                                  TST     (R2)+
020476  020205                                  CMP     R2,R5               ;CHECK AUTO-INC R5
020500  001766                                  BEQ     4$                  ;GO TO RTS
020502  104000                          6$:     HLT                         ;ERROR ABOVE

                                        ;CHECK JSR INST ADDRESS MODE 4
020504  013704  001506                  JSR4:   MOV     @#FACTOR,R4
```

```
020512  010703                             MOV     PC,R3
020514  000401                             BR      2$
020516  000405                     1$:     BR      4$
020520  022323                     2$:     CMP     (R3)+,(R3)+
020522  000277                             SCC
020524  004443                             JSR     R4,-(R3)                ;GO TO 2$
020526  104000                     3$:     HLT
020530  000414                             BR      JSR6                    ;GO TO NEXT TEST
020532  103012                     4$:     BCC     5$
020534  102011                             BVC     5$
020536  001010                             BNE     5$
020540  100007                             BPL     5$
020542  012702  020526                     MOV     #3$,R2                  ;GET UNRELOCATED RETURN ADDRESS
020546  061602                             ADD     (SP),R2                 ;ADD RELOCATION FACTOR (OLD R4)
020550  020204                             CMP     R2,R4                   ;CHECK THAT CALCULATED RETURN
020552  001002                             BNE     5$                      ;PC = NEW R4
020554  005724                             TST     (R4)+
020556  000204                             RTS     R4
020560  104000                     5$:     HLT


                                   ;TEST JSR INST ADDRESS MODE 6
020562  000401                     JSR6:   BR      2$
020564  000405                     1$:     BR      3$
020566  010700                     2$:     MOV     PC,R0
020570  004767  177770                     JSR     PC,1$
020574  100407                             BMI     JSR7                    ;GO TO NEXT TEST
020576  104000                             HLT                             ;ERROR ON CC'S
020600  022020                     3$:     CMP     (R0)+,(R0)+
020602  020016                             CMP     R0,(SP)                 ;CHECK THAT RETURN ADDRESS IS ON THE
020604  001401                             BEQ     .+4                     ;STACK
020606  104000                             HLT
020610  000270                             SEN                             ;SET N
020612  000207                             RTS     PC


                                   ;TEST JSR INST ADDRESS MODE 7
020614  013746  001506             JSR7:   MOV     @#FACTOR,-(SP)          ;GET RELOCATION FACTOR
020620  062716  020640                     ADD     #1$,(SP)                ;FORM ADDRESS OF 1$ BELOW
020624  000277                             SCC                             ;SET ALL CC'S
020626  004076  000000                     JSR     R0,@(SP)                ;JSR TO 1$
020632  003003                             BGT     3$
020634  102002                             BVC     3$
020636  000402                             BR      4$

020640  000200                     1$:     RTS     R0                      ;RETURN
020642  104000                     3$:     HLT                             ;ERROR!! INCORRECT CC'S
020644                             4$:
                                   ;;*****************************************************************
                                   ;*TEST 36        CHECK IOT TRAP (AND ROLB/ASLB)
                                   ;*               THIS TEST CHECKS THAT THE PSW IS CORRECT AFTER THE IOT AND THAT THE
                                   ;*               'NEW'PSW (FROM IOTVEC+2) IS CORRECT.
                                   ;;*****************************************************************
020644  112737  000036  001202     TST36:  MOVB    #36,@#$TSTNM            ;LOAD TEST NUMBER
020652  000004                             SCOPE
020654  012705  000022             IOTTST: MOV     #IOTVEC+2,R5
020660  005000                             CLR     R0
```

```
020666  011015                          MOV     (R0),(R5)               ;SET IOTVEC+2 = PSW
020670  011504                          MOV     (R5),R4                 ;SAVE IN R4
020672  010746                          MOV     PC,-(SP)
020674  062716  000036                  ADD     #1$-.,(SP)
020700  012645                          MOV     (SP)+,-(R5)             ;LOAD IOT TRAP VECTOR
020702  042710  000357                  BIC     #PR7+17,(R0)
020706  052710  000244                  BIS     #PR5+4,(R0)             ;PSW=X XXX X00 101 1X1 000
020712  012003                          MOV     (R0)+,R3                ;R3 = PSW ABOVE
020714  010340                          MOV     R3,-(R0)                ;RESTORE PSW (MOV CHANGED IT)
020716  000004                          IOT
020720  012737  044534  000020  10$:    MOV     #$SCOPE,@#IOTVEC        ;RESTORE IOT VECTOR
020726  104000                          HLT                             ;ERROR! IOT FAILED TO TRAP
020730  000457                          BR      TST37           ;;GO TO NEXT TEST

020732  012002                  1$:     MOV     (R0)+,R2                ;GET PSW AFTER IOT TRAP
                                                                        ;NOTE: R0=0
020734  012725  044534                  MOV     #$SCOPE,(R5)+           ;RESTORE IOTVEC
020740  012715  000200                  MOV     #PR4,(R5)               ;AND IOTVEC+2
020744  010746                          MOV     PC,-(SP)                ;FORM PC OF 10$ ABOVE
020746  062716  177752                  ADD     #10$-.,(SP)
020752  022626                          CMP     (SP)+,(SP)+             ;CHECK RETURN PC ON STACK
020754  001036                          BNE     99$
020756  022603                          CMP     (SP)+,R3                ;CHECK SAVED PSW
020760  001034                          BNE     99$
020762  032703  140000                  BIT     #UM,R3                  ;BRANCH TO 3$ IF IN USER MODE
020766  100413                          BMI     3$
020770  001003                          BNE     2$                      ;BRANCH TO 2$ IF IN SUPER MODE
020772  020204                          CMP     R2,R4                   ;CHECK PSW AFTER IOT
020774  001026                          BNE     99$
020776  000413                          BR      4$

021000  042704  030000          2$:     BIC     #PUM,R4                 ;CLEAR PREV MODE BITS
021004  052704  010000                  BIS     #PSM,R4                 ;SET PREV SUPER MODE
021010  020204                          CMP     R2,R4                   ;CHECK PSW AFTER IOT
021012  001017                          BNE     99$
021014  000404                          BR      4$

021016  052704  030000          3$:     BIS     #PUM,R4                 ;SET PREV USER MODE
021022  020204                          CMP     R2,R4                   ;CHECK PSW AFTER IOT
021024  001012                          BNE     99$

021026  005002                  4$:     CLR     R2
021030  000261                          SEC
021032  106100                          ROLB    R0                      ;ROTATE R0
021034  102376                          BVC     .-2                     ;UNTIL V SETS (R0=200)

021036  106300                          ASLB    R0                      ;SHIFT SHOULD SET CARRY
021040  103004                          BCC     99$
021042  102003                          BVC     99$
021044  001002                          BNE     99$
021046  005700                          TST     R0
021050  001401                          BEQ     .+4
021052  104000                  99$:    HLT                             ;ERROR! ROL/ASL FAILED TO SET
                                                                        ;CC'S PROPERLY (IF R2=0) OR IN-
                                                                        ;CORRECT PSW AFTER IOT (IF R2 NOT 0)
```

```
 021060  010437  177776                    MOV     R4,@#PSW                       ;RESTORE PSW
 021064  012706  000700                    MOV     #SUPSTK,SP                     ;RESTORE STACK PTR
                                   ;;**************************************************************
                                   ;*TEST 37        CHECK EMT TRAP SEQUENCE
                                   ;;**************************************************************
 021070  112737  000037  001202   TST37:   MOVB    #37,@#$TSTNM                   ;LOAD TEST NUMBER
 021076  000004                            SCOPE
                                           .EQUIV  IOT,HLT                        ;REDEFINE HLT CALL
 021100  012737  044774  000020            MOV     #$ERROR,@#IOTVEC               ;SETUP VECTOR
 021106  012737  000340  000022            MOV     #PR7,@#IOTVEC+2
 021114  005000                            CLR     R0
 021116  010746                            MOV     PC,-(SP)
 021120  062716  000030                    ADD     #EMT1-.,(SP)
 021124  012637  000030                    MOV     (SP)+,@#EMTVEC
 021130  000262                            SEV                                    ;SET V
 021132  013737  177776  000032            MOV     @#PSW,@#EMTVEC+2               ;RETAIN CURRENT PSW ON TRAP
 021140  000265                            +SEZ!SEC
 021142  104000                            EMT                          ;TRAP TO EMT1
 021144  001433                            BEQ     EMT1C                ;GO TO EMT1C
 021146  000004                            HLT                          ;ERROR! INCORRECT CC'S WERE SET ON RETURN
 021150  102027                   EMT1:    BVC     EMT1B                ;'V' SHOULD'VE SET ON EMT TRAP
 021152  105100                            COMB    R0                   ;R0=000377,CC'S=1001
 021154  105500                            ADCB    R0                   ;R0=000000,CC'S=0101
 021156  106000                            RORB    R0                   ;R0=000200,CC'S=1010
 021160  102023                            BVC     EMT1B
 021162  100022                            BPL     EMT1B
 021164  000257                            CCC
 021166  105400                            NEGB    R0                   ;R0=000200,CC'S=1010
 021170  102017                            BVC     EMT1B
 021172  100016                            BPL     EMT1B
 021174  000242                            CLV                          ;CLEAR 'V'
 021176  000261                            SEC                          ;AND SET 'C'
 021200  105300                            DECB    R0                   ;R0=000177,CC'S=0011
 021202  102012                            BVC     EMT1B
 021204  100411                            BMI     EMT1B
 021206  000242                            CLV                          ;CLEAR 'V'
 021210  105200                            INCB    R0                   ;R0=000200,CC'S=1011
 021212  103006                            BCC     EMT1B
 021214  102005                            BVC     EMT1B
 021216  100004                            BPL     EMT1B
 021220  000242                            CLV                          ;CLEAR 'V'
 021222  106200                            ASRB    R0                   ;SHIFT R0 UNTIL 'V' CLEARS
 021224  102776                            BVS     .-2
 021226  000401                            BR      .+4
 021230  000004                   EMT1B:   HLT                          ;ERROR!
 021232  000002                            RTI                          ;EXIT WITH R0=000377
 021234  105500                   EMT1C:   ADCB    R0                   ;R0=000000
 021236  103003                            BCC     EMT1D
 021240  001002                            BNE     EMT1D
 021242  005700                            TST     R0
 021244  001401                            BEQ     .+4
 021246  000004                   EMT1D:   HLT
 021250  012737  044774  000030            MOV     #$ERROR,@#EMTVEC               ;RESTORE EMT TO ERROR
 021256  012737  000340  000032            MOV     #PR7,@#EMTVEC+2                ;SET PRIORITY 7 ON ERROR
 021264  012737  044534  000020            MOV     #$SCOPE,@#IOTVEC               ;RESTORE IOT VECTOR
```

```
                                              .EQUIV  ERROR,HLT                       ;REDEFINE HLT CALL
                                       ;;*************************************************************
                                       ;*TEST 40        CHECK TRAP INSTRUCTION TRAP SEQUENCE
                                       ;;*************************************************************
021276  112737  000040  001202  TST40:  MOVB    #40,@#$TSTNM                    ;LOAD TEST NUMBER
021304  000004                          SCOPE
021306  052737  000340  177776          BIS     #PR7,@#PSW                      ;LOCK OUT LINE CLOCK
021314  052737  000340  000016          BIS     #PR7,@#TBITVEC+2
021322  010746                          MOV     PC,-(SP)
021324  062716  000056                  ADD     #TRAP1-.,(SP)
021330  012637  000034                  MOV     (SP)+,@#TRAPVEC
021334  000270                          SEN                                     ;SET N
021336  013737  177776  000036          MOV     @#PSW,@#TRAPVEC+2               ;RETAIN CURRENT PSW ON TRAP
021344  000261                          SEC                                     ;SET CARRY
021346  010700                          MOV     PC,R0
021350  000264                          SEZ                             ;SET Z BIT
021352  104400                          TRAP                            ;TRAP TO TRAP1
021354  103404                          BCS     .+12
021356  012737  051374  000034          MOV     #$TRAP,@#TRAPVEC                ;RESTORE TRAP VECTOR
021364  104000                          HLT
021366  001404                          BEQ     .+12
021370  012737  051374  000034          MOV     #$TRAP,@#TRAPVEC                ;RESTORE TRAP VECTOR
021376  104000                          HLT
021400  000420                          BR      TRAP1C
021402  100404                  TRAP1:  BMI     .+12                    ;N BIT GOT SET ON TRAP
021404  012737  051374  000034          MOV     #$TRAP,@#TRAPVEC                ;RESTORE TRAP VECTOR
021412  104000                          HLT
021414  062700  000004                  ADD     #4,R0
021420  020016                          CMP     R0,(SP)                 ;CHECK LOW BYTE OF RETURN PC ON
021422  001404                          BEQ     .+12                    ;STACK
021424  012737  051374  000034          MOV     #$TRAP,@#TRAPVEC                ;RESTORE TRAP VECTOR
021432  104000                          HLT
021434  124646                          CMPB    -(SP),-(SP)
021436  032626                          BIT     (SP)+,(SP)+
021440  000002                          RTI                             ;RETURN TO INST FOLLOWING TRAP (1$)

021442  012702  000036          TRAP1C: MOV     #TRAPVEC+2,R2                   ;RESTORE VECTORS
021446  012712  000340                  MOV     #PR7,(R2)
021452  012742  051374                  MOV     #$TRAP,-(R2)
021456  042737  000340  000016          BIC     #PR7,@#TBITVEC+2
021464  105037  177776                  CLRB    @#PSW                           ;GO BACK TO PRIORITY 0

021470  000004                  RELE3:  SCOPE
021472  010702                          MOV     PC,R2
021474  062702  000012                  ADD     #12,R2
021500  012707  034242                  MOV     #RELOC,PC               ;GO RELOCATE PROGRAM CODE
021504  000000                  REL33:  .WORD   0
                                ;3333333333333 LAST ADDRESS OF CODE TO BE RELOCATED 33333333333


                                ;;*************************************************************
                                ;*TEST 41        CHECK STACK OVERFLOW
                                ;;*************************************************************
021506  112737  000041  001202  TST41:  MOVB    #41,@#$TSTNM                    ;LOAD TEST NUMBER
021514  012767  000001  157574          MOV     #1,$TIMES               ;;DO 1 ITERATION
021522  000004                          SCOPE
```

```
                                                  .SBTTL  START OF SECTION 4
                                          ;444444444444 FIRST ADDRESS TO BE RELOCATED 444444444
021524  010700                            REL4:   MOV     PC,R0                   ;GET PC
021526  005740                                    TST     -(R0)                   ;R0 CONTAINS THE ADDRESS OF REL4
021530  010037  001512                            MOV     R0,@#FRSTAD             ;SAVE
021534  010700                                    MOV     PC,R0                   ;GET CURRENT PC
021536  162700  021536                            SUB     #.,R0                   ;SUBTRACT RELOCATION FACTOR
021542  010037  001506                            MOV     R0,@#FACTOR             ;SAVE RELOCATION FACTOR
021546  010737  001212                            MOV     PC,@#$LPERR             ;SET LOOP ADDRESS
021552  062737  000030  001212                    ADD     #30,@#$LPERR            ;ADJUST
021560  013737  001212  001210                    MOV     @#$LPERR,@#$LPADR
021566  105737  001502                            TSTB    @#NEXEC                 ;BR IF TEST CODE TO BE EXECUTED
021572  001402                                    BEQ     .+6
021574  000167  001324                            JMP     RELE4

021600  013767  177776  000334    OVFLW:  MOV     @#PSW,7$                ;SAVE STATUS IN 7$ BELOW
021606  005037  177776                            CLR     @#PSW                   ;SET KERNEL MODE
021612  004737  052340                            JSR     PC,@#CLRTBIT            ;GO CLEAR 'T' BIT IF SET
021616  052737  000340  177776                    BIS     #PR7,@#PSW              ;SET PRIORITY LEVEL 7 TO BLOCK CLOCK
021624  010746                                    MOV     PC,-(SP)                ;PUSH CURRENT PC ONTO STACK
021626  062716  000152                            ADD     #2$-.,(SP)              ;FORM ADDRESS OF 2$ BELOW
021632  011637  000004                            MOV     (SP),@#ERRVEC           ;SET ERROR VECTOR
021636  012737  000340  000006                    MOV     #340,@#ERRVEC+2         ;SET PRIORITY LEVEL 7 ON TRAP
021644  013727  000014                            MOV     @#BPTVEC,(PC)+          ;SAVE BPT VECTOR ADRS
021650  000000                            43$:    .WORD   0
021652  062716  000100                            ADD     #41$-2$,(SP)            ;FORM ADDRESS OF 41$ BELOW
021656  012637  000014                            MOV     (SP)+,@#BPTVEC          ;SET BPT TRAP VECTOR TO 41$
021662  012737  000340  000016                    MOV     #340,@#BPTVEC+2

021670  012703  000376                            MOV     #376,R3
021674  010313                                    MOV     R3,(R3)                 ;LOAD 376 INTO ADDRESS 376
021676  010306                                    MOV     R3,SP                   ;SET STACK PTR AT BOUNDARY
021700  032767  140000  000234                    BIT     #UM,7$                  ;CHECK IF ENTERED TEST IN KERNEL
021706  001015                                    BNE     1$                      ;MODE. BRANCH IF NOT IN KERNEL

                                          ;THE BELOW INSTRUCTIONS SHOULD NOT CAUSE AN OVERFLOW TRAP
021710  005716                                    TST     (SP)                    ;BECAUSE TST IS A NON MODIFYING INST
021712  021666  177776                            CMP     (SP),-2(SP)             ;SO IS COMPARE
021716  012656                                    MOV     (SP)+,@-(SP)            ;BECAUSE OF ADDRESS MODE 5
021720  057636  000000                            BIS     @(SP),@(SP)+            ;BECAUSE OF ADDRESS MODE 3
021724  054676  000000                            BIS     -(SP),@(SP)             ;BECAUSE OF ADDRESS MODE 7
021730  005006                                    CLR     SP
021732  013766  020000  020000                    MOV     @#20000,20000(SP)
021740  000425                                    BR      3$                      ;BRANCH OVER NON KERNEL MODE TESTS

                                          ;NOTE:  NO OVEFLOW TRAP WILL OCCUR IF NOT IN KERNEL MODE!!!
021742  156737  000175  177777    1$:     BISB    7$+1,@#PSW+1            ;RESTORE MODE BITS IN PSW
021750  012706  000376                            MOV     #376,SP                 ;SET STACK PTR
021754  016646  177776                            MOV     -2(SP),-(SP)            ;SHOULD NOT TRAP
021760  051616                                    BIS     (SP),(SP)
021762  061666  177776                            ADD     (SP),-2(SP)
021766  105037  177777                            CLRB    @#PSW+1                 ;SET KERNEL MODE
021772  012706  000700                            MOV     #SUPSTK,SP              ;RESTORE THE STACK
021776  000451                                    BR      6$                      ;EXIT TEST
```

```
022000  012600                          2$:     MOV     (SP)+,R0        ;SAVE PC OF INSTRUCTION THAT TRAPPED
022002  012602                                  MOV     (SP)+,R2        ;SAVE PSW
022004  012706  000700                          MOV     #SUPSTK,SP      ;SET STACK PTR
022010  104000                                  HLT                     ;ERROR! AN INSTRUCTION THAT WAS NOT
                                                                        ;SUPPOESED TO TRAP TRAPPED
                                                                        ;R0 CONTAINS PC, R2 CONTAINS PSW
022012  000443                                  BR      6$              ;EXIT TEST
                                        ;THE BELOW INSTRUCTIONS WILL CAUSE A STACK OVERFLOW
                                        ;STACK PTR IS AT 376
022014  062737  000066  000004          3$:     ADD     #4$-2$,@#ERRVEC ;SET ERROR VECTOR TO 4$
022022  010306                                  MOV     R3,SP           ;SET STACK PTR AT 376
022024  112702  000001                          MOVB    #1,R2
022030  005000                                  CLR     R0
022032  005016                                  CLR     (SP)            ;SETS BIT 0 IN R0
022034  006302                                  ASL     R2              ;SHIFT INDICATOR BIT
022036  105226                                  INCB    (SP)+           ;SETS BIT 1 IN R0
022040  006302                                  ASL     R2
022042  060746                                  ADD     PC,-(SP)        ;SETS BIT 2 IN R0
022044  006302                                  ASL     R2
022046  000003                                  BPT                     ;SETS BIT 3 IN R0
022050  006302                                  ASL     R2
022052  004767  000014                          JSR     PC,40$          ;SETS BIT 4 IN R0
022056  006302                                  ASL     R2
022060  050666  177776                          BIS     SP,-2(SP)       ;SETS BIT 5 IN R0
022064  000410                                  BR      5$

                                        ;PROGRAM WILL TRAP HERE ON OVERFLOW TRAP
022066  050200                          4$:     BIS     R2,R0           ;SET APPROPRIATE BIT IN R0
022070  000002                                  RTI                     ;RETURN FROM TRAP

022072  052700  001000                  40$:    BIS     #1000,R0                        ;SET IND THAT JSR WAS EXECUTED
022076  000207                                  RTS     PC

022100  052700  000400                  41$:    BIS     #400,R0                         ;SET IND THAT BPT WAS EXECUTED
022104  000002                                  RTI

                                        ;CHECK THAT ABOVE INSRUCTIONS DID TRAP
022106  012706  000700                  5$:     MOV     #SUPSTK,SP      ;SET STACK PTR
022112  022700  001477                          CMP     #1477,R0        ;EACH INSTRUCTION SET A BIT IN R0
022116  001401                                  BEQ     .+4             ;R0= 1477
022120  104000                                  HLT

                                        ;EXIT ROUTINE
022122  012706  001200                  6$:     MOV     #KERSTK,SP      ;SET KERNEL STACK PTR
022126  016737  177516  000014                  MOV     43$,@#BPTVEC    ;RESTORE BPT VECTOR
022134  005037  000016                          CLR     @#BPTVEC+2
022140  012746                                  MOV     (PC)+,-(SP)     ;PUSH OLD PSW ONTO STACK
022142  000000                          7$:     .WORD   0               ;CONTAINS SAVED PSW
022144  010746                                  MOV     PC,-(SP)        ;PUSH CURRENT PC ONTO STACK
022146  062716  000006                          ADD     #6,(SP)         ;ADD OFFSET
022152  000002                                  RTI
022154  012706  000700                          MOV     #SUPSTK,SP      ;SET STACK PTR
022160  012737  053442  000004                  MOV     #ERPRT,@#ERRVEC ;RESET TIME OUT VECTOR
022166  013737  177776  000006                  MOV     @#PSW,@#ERRVEC+2
022174  052737  000340  000006                  BIS     #PR7,@#ERRVEC+2
```

```
022210  005037  177766                    CLR     @#CPUERR
                                  ;;*************************************************************
                                  ;*TEST 42        CHECK THAT ALL RESERVED INSTRUCTIONS TRAP
                                  ;;*************************************************************
022214  112737  000042  001202    TST42:  MOVB    #42,@#$TSTNM            ;LOAD TEST NUMBER
022222  000004                            SCOPE
022224  012702  022330            RESTRP: MOV     #5$,R2          ;GET ADDRESS OF RESERVED INSTRUCTION TABLE
022230  063702  001506                    ADD     @#FACTOR,R2
022234  132737  000040  001471            BITB    #40,@#OPT.CP+1  ;CHECK IF 11/45 FLOATING POINT IS AVAIL.
022242  001402                            BEQ     .+6             ;BRANCH IF NOT AVAILABLE
022244  005067  000110                    CLR     50$             ;SET TABLE TERMINATOR AT GROUP 7
022250  012737  022306  000010            MOV     #4$,@#RESVEC    ;SET RESERVED INSTRUCTION TRAP
022256  063737  001506  000010            ADD     @#FACTOR,@#RESVEC
022264  012203                    1$:     MOV     (R2)+,R3        ;GET FIRST RESERVED INSTRUCTION
022266  001437                            BEQ     7$              ;0 TERMINATES THE TABLE
022270  012204                            MOV     (R2)+,R4        ;GET LAST RESERVED INSTRUCTION IN GROUP
022272  010317                    2$:     MOV     R3,(PC)         ;EXECUTE RESERVED INSTRUCTION
022274  000000                    3$:     .WORD   0               ;CONTAINS RESERVED INSTRUCTION
022276  104000                            HLT                     ;ERROR! INSTRUCTION IN R3
022300  104000                            HLT                     ;(2$) ABOVE FAILED TO CAUSE A
022302  104000                            HLT                     ;RESERVED INSTRUCTION TRAP
022304  000405                            BR      41$
022306  012716  022320            4$:     MOV     #41$,(SP)       ;ADJUST RETURN PC
022312  063716  001506                    ADD     @#FACTOR,(SP)   ;TO RETURN TO 41$
022316  000002                            RTI                     ;RETURN TO 41$
022320  020304                    41$:    CMP     R3,R4           ;HAS GROUP OF RESERVED INSTRUCTIONS
022322  001760                            BEQ     1$              ;BEEN EXECUTED
022324  005203                            INC     R3              ;INCREMENT THIS RESERVED INSTRUCTION
022326  000761                            BR      2$              ;TO NEXT ONE AND EXECUTE
                                  ;TABLE OF 11/40,11/45 RESERVED INSTRUCTIONS (0 TERMINATES THE TABLE)
022330  000007                    5$:     7                       ;GROUP 1
022332  000077                            77                      ;     "
022334  000210                            210                     ;GROUP 2
022336  000227                            227                     ;     "
022340  007000                            7000                    ;GROUP 3
022342  007777                            7777                    ;     "
022344  075040                            75040                   ;GROUP 4
022346  076777                            76777                   ;     "
022350  106400                            106400                  ;GROUP 5
022352  106477                            106477                  ;     "
022354  106700                            106700                  ;GROUP 6
022356  107777                            107777                  ;     "
022360  170000                    50$:    170000                  ;GROUP 7        FLOATING POINT
022362  177777                            177777                  ;     "         INSTRUCTIONS
022364  000000                            0                       ;0 TERMINATES THE TABLE

022366  012737  053370  000010    7$:     MOV     #RESERR,@#RESVEC        ;RESTORE RESERVED TRAP
                                  ;;*************************************************************
                                  ;*TEST 43        CHECK THAT ALL BITS IN THE PSW CAN BE SET AND CLEARED
                                  ;;*************************************************************
022374  112737  000043  001202    TST43:  MOVB    #43,@#$TSTNM            ;LOAD TEST NUMBER
022402  000004                            SCOPE
022404  105737  001503            PSWCHK: TSTB    @#MMON          ;IF MEM MGMT IS ON SKIP THIS TEST
022410  001065                            BNE     4$
022412  013767  177776  000132            MOV     @#PSW,3$        ;SAVE STATUS
```

```
022424  004737  052340                   JSR     PC,@#CLRTBIT      ;GO CLEAR 'T' BIT IF SET
022430  013746  000016                   MOV     @#TBITVEC+2,-(SP)
022434  012704  177776                   MOV     #PSW,R4           ;LOAD ADDRESS OF PSW INTO R4
022440  000250                           CLN
022442  005714                           TST     (R4)              ;CHECK THAT PSW WAS CLEARED
022444  001401                           BEQ     .+4
022446  104000                           HLT                       ;ERROR! PSW FAILED TO CLEAR
022450  012700  170357                   MOV     #170357,R0
022454  052700  170000                   BIS     #170000,R0        ;SET BITS 15-12 IF MEM MGMT
022460  012702  000001          10$:     MOV     #1,R2             ;R2 = TEST BIT
022464  030200                  1$:      BIT     R2,R0             ;CHECK IF BIT CAN BE SET/CLEARED
022466  001423                           BEQ     2$
022470  005037  000016                   CLR     @#TBITVEC+2
022474  030227  000020                   BIT     R2,#20            ;CHECK IF TEST WILL SET 'T' BIT
022500  001403                           BEQ     20$
022502  012737  000002  000016           MOV     #RTI,@#TBITVEC+2;SET RTI INTO RETURN
022510  005014                  20$:     CLR     (R4)              ;CLEAR PSW
022512  050214                           BIS     R2,(R4)           ;SET R2 INTO PSW
022514  011403                           MOV     (R4),R3           ;GET BIT
022516  020203                           CMP     R2,R3             ;CHECK THAT BIT WAS SET IN PSW
022520  001401                           BEQ     .+4
022522  104000                           HLT                       ;ERROR! BIT IN R2 FAILED TO SET IN PSW
022524  000244                           CLZ                       ;CLEAR Z BIT
022526  040214                           BIC     R2,(R4)           ;CLEAR BIT IN PSW
022530  011403                           MOV     (R4),R3           ;GET PSW RESULT
022532  001401                           BEQ     2$                ;BRANCH IF BIC ABOVE CLEARED BIT IN PSW
022534  104000                           HLT                       ;ERROR! BIT IN R2 FAILED TO CLEAR IN PSW
022536  006302                  2$:      ASL     R2                ;SHIFT TEST BIT
022540  103351                           BCC     1$                ;BRANCH IF ALL BITS NOT TESTED
022542  005014                           CLR     (R4)              ;CLEAR STATUS
022544  012637  000016                   MOV     (SP)+,@#TBITVEC+2      ;RESTORE T BIT RETURN
022550  012746                           MOV     (PC)+,-(SP)       ;PUSH ORIGINAL STATUS ON STACK
022552  000000                  3$:      .WORD   0                 ;CONTAINS ORIGINAL PSW
022554  010746                           MOV     PC,-(SP)          ;SET RETURN PC
022556  062716  000006                   ADD     #6,(SP)
022562  000002                           RTI                       ;RETURN
022564  013704  177776          4$:      MOV     @#PSW,R4          ;SAVE PSW IN R4
022570  112737  000340  177776           MOVB    #340,@#PSW        ;SET PRIORITY LEVEL 7
022576  004737  052340                   JSR     PC,@#CLRTBIT      ;GO CLEAR 'T' BIT IF SET
                                ;;*****************************************************************
                                ;*TEST 44        CHECK THAT ALL BITS IN THE CURRENT STACK PTR CAN BE SET CLEARED
                                ;;*****************************************************************
022602  112737  000044  001202  TST44:   MOVB    #44,@#$TSTNM               ;LOAD TEST NUMBER
022610  000004                           SCOPE
022612  010603                  CHKSP:   MOV     SP,R3             ;SAVE STACK PTR
022614  000257                           CCC
022616  112706  000377                   MOVB    #377,SP           ;SET STACK PTR = -1
022622  006006                  1$:      ROR     SP                ;ROTATE 0 BIT THROUGH ALL BIT
022624  103776                           BCS     1$                ;BIT POSITIONS
022626  005206                           INC     SP                ;SHOULD INCREMENT SP TO 0
022630  001403                           BEQ     2$
022632  010602                           MOV     SP,R2             ;SAVE ERROR STACK PTR
022634  010306                           MOV     R3,SP             ;SET STACK PTR FOR TRAP
022636  104000                           HLT                       ;ERROR!
```

```

                                        ;CHECK BYTE OPERATIONS USING THE STACK
022642  010600                          SPCHK:  MOV     SP,R0           ;SAVE STACK PTR
022644  010003                                  MOV     R0,R3

022646  005043                                  CLR     -(R3)
022650  112746  177777                          MOVB    #-1,-(SP)       ;(SP) = 377
022654  022713  000377                          CMP     #377,(R3)       ;CHECK THAT ONLY EVEN BYTE WAS AFFECTED
022660  001002                                  BNE     1$
022662  020306                                  CMP     R3,SP           ;CHECK AUTO-DEC
022664  001401                                  BEQ     .+4
022666  104000                          1$:     HLT

022670  105226                                  INCB    (SP)+
022672  005723                                  TST     (R3)+           ;CHECK RESULT
022674  001002                                  BNE     2$
022676  020006                                  CMP     R0,SP           ;CHECK AUTO-INC
022700  001401                                  BEQ     .+4
022702  104000                          2$:     HLT

022704  005143                                  COM     -(R3)           ;(R3)=177777
022706  144613                                  BICB    -(SP),(R3)
022710  022713  177400                          CMP     #177400,(R3)    ;CHECK RESULT
022714  001002                                  BNE     3$
022716  020603                                  CMP     SP,R3
022720  001401                                  BEQ     .+4
022722  104000                          3$:     HLT

022724  132627  000377                          BITB    (SP)+,#377
022730  001002                                  BNE     4$
022732  020600                                  CMP     SP,R0
022734  001401                                  BEQ     .+4
022736  104000                          4$:     HLT

022740  012746  000001                          MOV     #1,-(SP)
022744  062706  000002                          ADD     #2,SP
022750  012702  177401                          MOV     #177401,R2
022754  120246                                  CMPB    R2,-(SP)
022756  001004                                  BNE     5$
022760  122602                                  CMPB    (SP)+,R2
022762  001002                                  BNE     5$
022764  020006                                  CMP     R0,SP
022766  001401                                  BEQ     .+4
022770  104000                          5$:     HLT
022772  105037  177776                          CLRB    @#PSW
022776  010446                                  MOV     R4,-(SP)        ;RESTORE ORIGINAL PSW TO STACK
023000  010746                                  MOV     PC,-(SP)
023002  062716  000006                          ADD     #6,(SP)
023006  000002                                  RTI
                                        ;;*****************************************************************
                                        ;*TEST 45         CHECK THAT 'C' BIT SETS/CLEARS PROPERLY
                                        ;;*****************************************************************
023010  112737  000045  001202          TST45:  MOVB    #45,@#$TSTNM            ;LOAD TEST NUMBER
023016  000004                                  SCOPE
023020  012727  177776                  CBIT:   MOV     #177776,(PC)+   ;LOAD CONSTANT
```

```
023026  010700                          MOV     PC,R0           ;GET CURRENT PC
023030  162700  000004                  SUB     #4,R0           ;POINT R0 TO 1$ ABOVE
023034  005520                  2$:     ADC     (R0)+           ;ADD 'C' BIT TO 1$ ABOVE
023036  006340                          ASL     -(R0)           ;SHIFT 1$
023040  102375                          BVC     2$              ;UNTIL 'V' BIT SETS
023042  022767  077776  177754          CMP     #077776,1$      ;CHECK RESULT
023050  001401                          BEQ     .+4
023052  104000                          HLT                     ;ERROR! INCORRECT RESULT IN 1$ ABOVE
                                                                ;R0=ADDRESS OF DATA

                                ;CHECK THAT CONDITION CODES ARE SET PROPERLY WHEN A NUMBER (CURRENT PC)
                                ;AND THAT NUMBER +1 ARE COMPARED, AND VICE VERSA.
023054  010700                  CMPN:   MOV     PC,R0           ;GET CURRENT PC
023056  010002                          MOV     R0,R2           ;SAVE IN R2
023060  005202                          INC     R2              ;MAKE R2 = R0+1
023062  000277                          SCC
023064  000251                          +CLC!CLN                ;CLEAR C & N BITS
023066  020002                          CMP     R0,R2           ;COMPARE # WITH #+1
023070  103003                          BCC     1$              ;CARRY BIT SHOULD SET
023072  102402                          BVS     1$              ;V BIT SHOULD CLEAR
023074  001401                          BEQ     1$              ;Z BIT SHOULD CLEAR
023076  100401                          BMI     .+4             ;N BIT SHOULD SET
023100  104000                  1$:     HLT                     ;ERROR! COMPARE # WITH #+1 FAILED TO
                                                                ;SET CONDITION CODES IN PSW CORRECTLY

023102  000277                          SCC                     ;SET CONDITION CODES IN PSW
023104  120200                          CMPB    R2,R0           ;COMPARE #+1 WITH #
023106  103403                          BCS     2$              ;C BIT SHOULD CLEAR
023110  102402                          BVS     2$              ;V BIT SHOULD CLEAR
023112  001401                          BEQ     2$              ;Z BIT SHOULD CLEAR
023114  100001                          BPL     .+4             ;N BIT SHOLD CLEAR
023116  104000                  2$:     HLT                     ;ERROR! COMPARE #+1 WITH # FAILED TO SET
                                                                ;CONDITION CODES IN PSW CORRECTLY
023120  105037  177776                  CLRB    @#PSW           ;ENSURE PRIORITY 0
023124  000004                  RELE4:  SCOPE
023126  010702                          MOV     PC,R2
023130  062702  000012                  ADD     #12,R2
023134  012707  034242                  MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
023140  000000                  REL44:  .WORD   0
                                ;444444444444 LAST ADDRESS OF CODE TO BE RELOCATED 44444444444

                                ;;*****************************************************************
                                ;*TEST 46        CHECK EXTENDED INSTRUCTION SET
                                ;;*****************************************************************
023142  112737  000046  001202  TST46:  MOVB    #46,@#$TSTNM            ;LOAD TEST NUMBER
023150  012767  000001  156140          MOV     #1,$TIMES       ;;DO 1 ITERATION
023156  000004                          SCOPE

                                        .SBTTL  START OF SECTION 5
                                ;555555555555 FIRST ADDRESS TO BE RELOCATED 555555555
023160  010700                  REL5:   MOV     PC,R0           ;GET PC
023162  005740                          TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL5
023164  010037  001512                  MOV     R0,@#FRSTAD     ;SAVE
023170  010700                          MOV     PC,R0           ;GET CURRENT PC
023172  162700  023172                  SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
```

```
023202  010737  001212                          MOV     PC,@#$LPERR             ;SET LOOP ADDRESS
023206  062737  000030  001212                  ADD     #30,@#$LPERR            ;ADJUST
023214  013737  001212  001210                  MOV     @#$LPERR,@#$LPADR
023222  105737  001502                          TSTB    @#NEXEC                 ;BR IF TEST CODE TO BE EXECUTED
023226  001402                                  BEQ     .+6
023230  000167  001454                          JMP     RELES
023234  005000                        EXTINST:CLR     R0
023236  000277                                  SCC                             ;PRESET CC'S
023240  006700                                  SXT     R0                      ;EXTEND SIGN (1) INTO R0
023242  103005                                  BCC     SXT0                    ;CHECK RESULT CC'S
023244  102404                                  BVS     SXT0
023246  001403                                  BEQ     SXT0
023250  100002                                  BPL     SXT0
023252  005200                                  INC     R0                      ;CHECK RESULT
023254  001401                                  BEQ     .+4
023256  104000                        SXT0:     HLT

023260  010700                                  MOV     PC,R0
023262  010002                                  MOV     R0,R2
023264  012703  177777                          MOV     #-1,R3
023270  005102                                  COM     R2
023272  000243                                  +CLV!CLC                        ;CLEAR C AND V BITS
023274  074003                                  XOR     R0,R3                   ;R3 SHOULD CONTAIN COMPLEMENT OF R0
023276  103404                                  BCS     XOR0                    ;CHECK THAT C WAS NOT AFFECTED
023300  102403                                  BVS     XOR0                    ;AND THAT V WAS CLEARED
023302  001402                                  BEQ     XOR0
023304  020203                                  CMP     R2,R3                   ;CHECK RESULT
023306  001401                                  BEQ     .+4
023310  104000                        XOR0:     HLT                             ;ERROR!   XOR FAILED

023312  010700                                  MOV     PC,R0
023314  022020                                  CMP     (R0)+,(R0)+             ;SET ADDRESS REGISTER
023316  000401                                  BR      1$                      ;RESERVE WORD FOR TEST DATA
023320  000000                                  .WORD   0                       ;CONTAINS TEST DATA
023322  005700                        1$:       TST     R0                      ;EXTEND SIGN OF ADDRESS INTO
023324  006710                                  SXT     (R0)                    ;ADDRESS (R0)=-1 IF MSB R0=1
023326  005002                                  CLR     R2                      ;OTHERWISE, (R0)=0
023330  005700                                  TST     R0                      ;CHECK SIGN OF ADDRESS
023332  100001                                  BPL     .+4
023334  005102                                  COM     R2                      ;COMPLEMENT CHECK REG IF NEG
023336  021002                                  CMP     (R0),R2                 ;CHECK RESULT OF SXT
023340  001401                                  BEQ     .+4
023342  104000                        SXT1:     HLT                             ;ERROR!   SXT FAILED TO EXTEND SIGN PROPERLY

023344  012710  100000                          MOV     #100000,(R0)            ;PRESET DATA
023350  011002                                  MOV     (R0),R2
023352  000277                                  SCC                             ;PRESET CC'S
023354  074210                                  XOR     R2,(R0)                 ;XOR 100000 WITH 100000 RESULT = 0
023356  103007                                  BCC     XOR1                    ;CHECK CC'S AFTER XOR
023360  102406                                  BVS     XOR1
023362  001005                                  BNE     XOR1
023364  100404                                  BMI     XOR1
023366  005710                                  TST     (R0)                    ;CHECK RESULT (0)
023370  001002                                  BNE     XOR1
023372  005402                                  NEG     R2                      ;CHECK THAT REG WAS NOT AFFECTED
```

```
023376  104000                  XOR1:   HLT

023400  010702                          MOV     PC,R2
023402  022222                          CMP     (R2)+,(R2)+
023404  000401                          BR      SXT4                    ;PRESERVE WORD FOR DATA
023406  000000                          .WORD   0                       ;RESERVED FOR DATA
023410  012722  125252          SXT4:   MOV     #125252,(R2)+           ;PRESET DATA
023414  006742                          SXT     -(R2)                   ;EXTEND SIGN
023416  074722                          XOR     PC,(R2)+
023420  010700                          MOV     PC,R0                   ;GET PC
023422  005740                          TST     -(R0)                   ;SUBTRACT 2 FROM PC
023424  005100                          COM     R0                      ;R0=RESULT OF XOR PC-1 ABOVE
023426  074042                          XOR     R0,-(R2)                ;CHECK RESULT OF SXT AND XOR ABOVE
023430  001401                          BEQ     .+4
023432  104000                  XOR24:  HLT                             ;ERROR!  SXT & XOR ABOVE INCORRECT

023434  012704  000001                  MOV     #1,R4                   ;SET R4
023440  006767  000060                  SXT     XOR6A                   ;PRESET DATA=0
023444  074467  000054          2$:     XOR     R4,XOR6A
023450  100423                          BMI     XOR6
023452  006304                          ASL     R4                      ;SHIFT R4
023454  102373                          BVC     2$                      ;UNTIL V SETS (R4=100000)
023456  100020                          BPL     XOR6                    ;BRANCH IF 'N' IS CLEAR
023460  074467  000040                  XOR     R4,XOR6A                ;XOR6A=177777
023464  100015                          BPL     XOR6
023466  074767  000032                  XOR     PC,XOR6A                ;XOR PC WITH XOR6A (177777)
023472  010767  000030                  MOV     PC,XOR6B                ;FORM PC AS USED IN XOR ABOVE
023476  162767  000004  000022          SUB     #4,XOR6B
023504  005167  000016                  COM     XOR6B
023510  026767  000012  000006          CMP     XOR6B,XOR6A             ;XOR6A SHOULD = COMPLEMENT OF PC
023516  001401                          BEQ     .+4
023520  104000                  XOR6:   HLT                             ;ERROR!  XOR TESTS ABOVE FAILED

023522  000402                          BR      .+6

023524  000000                  XOR6A:  .WORD   0                       ;CONTAINS DATA USED BY TEST ABOVE
023526  000000                  XOR6B:  .WORD   0


023530  012700  077777                  MOV     #077777,R0              ;SET SOURCE OPERAND FOR ADD
023534  006767  177764                  SXT     XOR6A                   ;CLEAR XOR6A
023540  001004                          BNE     SXT6                    ;CHECK CC'S AFTER EXTENDING ZERO'S
023542  100403                          BMI     SXT6
023544  103402                          BCS     SXT6
023546  102401                          BVS     SXT6
023550  000401                          BR      .+4
023552  104000                  SXT6:   HLT                             ;ERROR!  SXT FAILED

023554  012702  000001                  MOV     #1,R2                   ;SET DEST OPERAND FOR ADD
023560  013703  001506                  MOV     @#FACTOR,R3             ;LOAD INDEX REGISTER
023564  060002                          ADD     R0,R2                   ;RESULT OF ADD=100000
023566  006763  023524                  SXT     XOR6A(3)                ;EXTEND SIGN OF ADD ABOVE
023572  001403                          BEQ     SXT6A
023574  005267  177724                  INC     XOR6A                   ;CHECK RESULT OF SXT
023600  001401                          BEQ     .+4
```

```
                                                                  ;SIGN
023604  010703                          MOV     PC,R3
023606  000402                          BR      .+6               ;PRESERVE 2 WORDS FOR DATA
023610  000000                 SXRA:    .WORD   0                 ;RESERVED WORD FOR DATA
023612  000000                 SXRB:    .WORD   0                 ;RESERVED WORD FOR DATA
023614  005723                          TST     (R3)+
023616  010304                          MOV     R3,R4             ;R3 = ADDRESS OF SXRA
023620  000250                          CLN                       ;CLEAR N BIT
023622  006724                          SXT     (R4)+             ;EXTEND ZEROS INTO SXRA
023624  001401                          BEQ     .+4
023626  104000                 SXT2:    HLT                       ;ERROR!  SXT FAILED

023630  010467  177754                  MOV     R4,SXRA           ;SXRA = ADDRESS OF SXRB
023634  000257                          CCC                       ;CLEAR CONDITION CODES
023636  006733                          SXT     @(R3)+            ;EXTEND ZEROS INTO SXRB
023640  001401                          BEQ     .+4
023642  104000                 SXT3:    HLT                       ;ERROR!

023644  000270                          SEN                       ;SET N BIT
023646  006753                          SXT     @-(R3)            ;EXTEND ONES INTO SXRB
023650  100401                          BMI     .+4
023652  104000                 SXT5:    HLT                       ;ERROR!

023654  012704  025252                  MOV     #025252,R4        ;R4 = 025252
023660  074433                          XOR     R4,@(R3)+         ;SXRB = 152525 (COMPLEMENT OF R4)
023662  005002                          CLR     R2
023664  074253                          XOR     R2,@-(R3)         ;SXRB REMAINS UNCHANGED
023666  001405                          BEQ     XOR35             ;CHECK CONDITION CODES
023670  100004                          BPL     XOR35
023672  005104                          COM     R4                ;R4 = 152525
023674  020467  177712                  CMP     R4,SXRB           ;CHECK XOR
023700  001401                          BEQ     .+4
023702  104000                 XOR35:   HLT                       ;ERROR!  XOR FAILED

023704  005743                          TST     -(R3)             ;R3 = ADDRESS OF SXRA-2
023706  000250                          CLN                       ;CLEAR N BIT
023710  006773  000002                  SXT     @2(R3)            ;SXRB = 0
023714  001401                          BEQ     .+4
023716  104000                 SXT7:    HLT                       ;ERROR!  SXT FAILED

023720  074473  000002                  XOR     R4,@2(R3)         ;SXRB = R4
023724  020473  000002                  CMP     R4,@2(R3)         ;CHECK XOR
023730  001401                          BEQ     .+4
023732  104000                 XOR7:    HLT                       ;ERROR!  XOR FAILED
                               ;;*******************************************************************
                               ;;*TEST 47           SOB TEST
                               ;;*       NOTE:      DO NOT INSERT ANY CODE IN FOLLOWING SOB TESTS
                               ;                    SINCE IT TESTS THE MAXIMUM BRANCH WIDTH OF THE INSTRUCTION.
                               ;;*******************************************************************
023734  112737  000047  001202 TST47:   MOVB    #47,@#$TSTNM              ;LOAD TEST NUMBER
023742  000004                          SCOPE

```

```
023746  000407                          BR      SOB0            ;BRANCH TO SOB TEST

023750  005004                  SOB10:  CLR     R4              ;R4 = 0
023752  005705                          TST     R5              ;CHECK ERROR INDICATOR
023754  001401                          BEQ     .+4             ;SOB BRANCHED CORRECTLY
023756  104000                          HLT                     ;ERROR!

023760  005005                  SOB9:   CLR     R5              ;CLEAR INDICATOR (R5)
023762  006004                          ROR     R4              ;ROTATE RIGHT R4
023764  000467                          BR      SOB8

023766  012700  000010          SOB0:   MOV     #10,R0          ;R0=10
023772  000277                          SCC                     ;SET CONDITION CODES
023774  001012                  SOB1:   BNE     SOB2            ;CHECK CONDITION CODES AFTER SOB
023776  100011                          BPL     SOB2            ;SOB SHOULD NOT EFFECT THE
024000  102010                          BVC     SOB2            ;CONDITION CODES.
024002  103007                          BCC     SOB2
024004  077005                          SOB     R0,SOB1
024006  001005                          BNE     SOB2            ;CHECK CONDITION CODES AFTER
024010  100004                          BPL     SOB2            ;SOB FALLS THROUGH,
024012  102003                          BVC     SOB2            ;SOB SHOULD NOT EFFECT
024014  103002                          BCC     SOB2            ;CONDITION CODES.
024016  005700                          TST     R0              ;CHECK IF R0=0
024020  001401                          BEQ     .+4
024022  104000                  SOB2:   HLT                     ;ERROR!

024024  012702  000100                  MOV     #100,R2         ;R2=100
024030  012700  000101                  MOV     #101,R0         ;SET CHECK REGISTER, R0=101
024034  001414                  SOB3:   BEQ     SOB4            ;CHECK CONDITION CODES AFTER
024036  100413                          BMI     SOB4            ;SOB BRANCH,
024040  102412                          BVS     SOB4            ;SOB SHOULD NOT EFFECT
024042  103411                          BCS     SOB4            ;CONDITION CODES.
024044  005300                          DEC     R0              ;DECREMENT CHECK REGISTER
024046  020002                          CMP     R0,R2           ;CHECK THAT SOB DECREMENTS
024050  001006                          BNE     SOB4
024052  000257                          CCC                     ;SET CONDITION CODES BEFORE SOB
024054  077211                          SOB     R2,SOB3         ;BRANCH TO SOB3 UNTIL R2=0
024056  001403                          BEQ     SOB4            ;CHECK CONDITION CODES AFTER
024060  100402                          BMI     SOB4            ;SOB FALLS THROUGH
024062  005702                          TST     R2              ;CHECK IF R2=0
024064  001401                          BEQ     .+4
024066  104000                  SOB4:   HLT                     ;ERROR!

024070  012700  000001          SOB5:   MOV     #1,R0           ;R0=1
024074  000401                          BR      .+4
024076  104000                          HLT                     ;ERROR!
024100  077002                          SOB     R0,.-2          ;SOB SHOULD NOT BRANCH

024102  005700                          TST     R0              ;CHECK IF R0=0 AFTER SOB
024104  001401                          BEQ     .+4
024106  104000                          HLT                     ;ERROR!

024110  012704  100000          SOB5A:  MOV     #100000,R4      ;R4=100000
024114  000403                          BR      1$
024116  005204                  3$:     INC     R4              ;R4=100000
```

```
024122  104000                           HLT                         ;ERROR!   SOB DID NOT
                                                                     ;INCREMENT PROPERLY

024124  077404                   1$:     SOB     R4,3$               ;SOB SHOULD BRANCH
024126  104000                           HLT                         ;ERROR!  SOB DID NOT BRANCH

024130  012703  000100           2$:     MOV     #100,R3             ;R3=100
024134  077301                   SOB6:   SOB     R3,SOB6             ;USE SOB TO BRANCH TO ITSELF
024136  005703                           TST     R3                  ;CHECK IF R3=0
024140  001703                           BEQ     SOB10
024142  104000                   SOB7:   HLT                         ;ERROR!

024144  005705                   SOB8:   TST     R5                  ;CHECK INDICATOR (R5)
                                                                     ;IF SOB BRANCHES INCORRECTLY
                                                                     ;WHEN CHECKING MAX. BRANCH,
                                                                     ;R5 WILL NOT BE CLEARED AT
                                                                     ;THIS POINT INDICATING AN ERROR.

024146  001401                           BEQ     .+4                 ;BRANCH IF SOB BRANCHES CORRECTLY
024150  104000                           HLT                         ;ERROR!

024152  005205                           INC     R5                  ;SET INDICATOR (R5)
024154  077477                           SOB     R4,SOB9             ;TEST MAX. BRANCH OF SOB
024156  005704                           TST     R4                  ;CHECK IF R4=0
024160  001401                           BEQ     .+4
024162  104000                           HLT                         ;ERROR!
                                 ;;*************************************************************
                                 ;*TEST 50       CHECK THE MARK INSTRUCTION
                                 ;;*************************************************************
024164  112737  000050  001202   TST50:  MOVB    #50,@#$TSTNM             ;LOAD TEST NUMBER
024172  000004                           SCOPE
024174  010602                   MRKTST: MOV     SP,R2
024176  010705                           MOV     PC,R5               ;THE STACK LOOKS LIKE THIS AFTER
024200  010500                           MOV     R5,R0               ;THE JSR INSTRUCTION
024202  010546                           MOV     R5,-(SP)            ; -2(SP)= R0     THIS IS A
024204  010746                           MOV     PC,-(SP)            ; -4(SP)= PC     STRING
024206  010746                           MOV     PC,-(SP)            ; -6(SP)= PC+2   OF
024210  010746                           MOV     PC,-(SP)            ;-10(SP)= PC+4   FIVE
024212  010746                           MOV     PC,-(SP)            ;-12(SP)= PC+6   DUMMY
024214  010746                           MOV     PC,-(SP)            ;-14(SP)= PC+10 ARGUMENTS
024216  012746  006405                   MOV     #MARK+5,-(SP)       ;-16(SP)= MARK 5
024222  010605                           MOV     SP,R5               ;-20(SP)= PC PUSHED BY JSR
024224  004767  000002                   JSR     PC,MARK1
024230  000403                           BR      .+10
024232  000205                   MARK1:  RTS     R5
024234  104000                           HLT                         ;ERROR! SHOULD BE DOING MARK 5 INST.
024236  000407                           BR      MARKEX
024240  020602                           CMP     SP,R2
024242  001402                           BEQ     .+6
024244  104000                           HLT                         ;ERROR! SP NOT RETURNED TO PROPER
024246  000403                           BR      MARKEX              ;VALUE BY MARK INSTRUCTION
024250  020005                           CMP     R0,R5
024252  001401                           BEQ     .+4
024254  104000                           HLT                         ;ERROR! DID NOT RESTORE R5 FROM STACK
024256  010206                   MARKEX: MOV     R2,SP               ;RESTORE SP
```

```
                                        ;*TEST 51          RTT/RTI TEST
                                        ;*       RTT/RTI TEST INSURES THAT CP DOES THE INSTRUCTION FOLLOWING
                                        ;*       AN RTT IF THE "T"BIT IS SET IN THE PSW,BUT DOES HONOR
                                        ;*       THE TRAP IMMEDIATELY IF IT EXECUTES AN RTI
                                        ;*       INSTRUCTION SEQUENCE-RTT
                                        ;*       2$:      RTT                      ;NO 'T' TRAP AFTER RTT
                                        ;*                INC       R0             ;R0=000001
                                        ;*                                         ;'T' TRAP TO 5$ AFTER INC
                                        ;*       5$:      COM       R0             ;R0=177776
                                        ;*                MOV       SAVPSW,2(SP)   ;CLEAR 'T' BIT IN RETURN PSW
                                        ;*                RTI                      ;RETURN TO INSTRUCTION FOLLOWING INC
                                        ;*                CMP       #RTT,2$        ;CHECK
                                        ;*                          ETC
                                        ;*
                                        ;*       INSTRUCTION SQUENCE-RTI
                                        ;*       2$:      RTI                      ;'T' TRAP AFTER RTI
                                        ;*       5$:      COM       R0             ;R0=177777
                                        ;*                MOV       SAVPSW,2(SP)   ;CLEAR 'T' BIT IN RETURN PSW
                                        ;*                RTI                      ;RETURN TO INC INSTRUCTION
                                        ;*                INC       R0             ;R0=000000
                                        ;*                CMP       #RTT,2$        ;CHECK
                                        ;*                          ETC
                                        ;*************************************************************************
 024260  112737  000051  001202  TST51:  MOVB     #51,@#$TSTNM                     ;LOAD TEST NUMBER
 024266  000004                          SCOPE
 024270  013767  177776  000202  RTT1:   MOV      @#PSW,SAVPSW                     ;SAVE PSW
 024276  032767  000020  000174          BIT      #20,SAVPSW                       ;CHECK IF "T"BIT SET
 024304  001176                          BNE      RTT2EX                           ;BRANCH TO EXIT
 024306  010746                  1$:     MOV      PC,-(SP)                         ;GET CURRENT PC
 024310  062716  000116                  ADD      #5$-.,(SP)                       ;FORM RELOCATED PC
 024314  012637  000014                  MOV      (SP)+,@#TBITVEC                  ;LOAD INTO TRAP VECTOR
 024320  016746  000154                  MOV      SAVPSW,-(SP)                     ;GET CURRENT PSW
 024324  011637  000016                  MOV      (SP),@#TBITVEC+2
 024330  052737  000340  177776          BIS      #PR7,@#PSW                       ;SET PRIORITY LEVEL 7
 024336  005000                          CLR      R0
 024340  052716  000360                  BIS      #PR7+20,(SP)                     ;SET "T"BIT IN PSW ON STACK
 024344  010746                          MOV      PC,-(SP)                         ;PUT THE PC ON THE STACK
 024346  062716  000006                  ADD      #6,(SP)                          ;ADJUST PC FOR NEXT INSTRUCTION
 024352  000006                  2$:     RTT
 024354  005200                          INC      R0                               ;DONE TO SEE IF INSTR. FOLLOWING
                                                                                   ;RTT IS EXECUTED IF T-BIT SET
 024356  042737  000340  177776          BIC      #PR7,@#PSW                       ;SET PRIORITY LEVEL 0
 024364  022767  000006  177760          CMP      #RTT,2$
 024372  001005                          BNE      3$                               ;CHECK IF INC WAS EXECUTED
 024374  022700  177776                  CMP      #177776,R0                       ;CHECK IF COM-R0 EXECUTED
 024400  001406                          BEQ      4$
 024402  104000                          HLT                                       ;ERROR!R0 NOT COMPLIMENTED
 024404  000415                          BR       6$                               ;EXIT TEST
 024406  005700                  3$:     TST      R0                               ;TEST IF TRAPED BEFORE INC INST.
                                                                                   ;WAS EXECUTED
 024410  001413                          BEQ      6$
 024412  104000                          HLT                                       ;ERROR!
 024414  000411                          BR       6$                               ;EXIT TEST
 024416  012767  000002  177726  4$:     MOV      #RTI,2$
 024424  000730                          BR       1$
```

```
024430  016766  000044  000002             MOV     SAVPSW,2(SP)
024436  000002                             RTI
024440  012767  000006  177704    6$:      MOV     #RTT,2$
024446  012737  001472  000014             MOV     #$RTRN,@#TBITVEC                      ;RESTORE 'T' TRAP VECTOR
024454  005037  000016                     CLR     @#TBITVEC+2
024460  042737  000360  000016             BIC     #PR7+BIT4,@#TBITVEC+2
024466                            RTT1EX:
                                  ;;*****************************************************************
                                  ;*TEST 52          SECOND RTT TEST
                                  ;;*****************************************************************
024466  112737  000052  001202    TST52:   MOVB    #52,@#$TSTNM              ;LOAD TEST NUMBER
024474  000004                             SCOPE
024476  000401                             BR      RTT2A
024500  000000                    SAVPSW:  .WORD   0
024502  016700  177772            RTT2A:   MOV     SAVPSW,R0                 ;GET SAVED PSW
024506  105000                             CLRB    R0                        ;CLEAR PRIOITY LEVEL,T, AND COND CODES
024510  012702  144000                     MOV     #UM+REG,R2
024514  074002                             XOR     R0,R2
024516  001435                             BEQ     2$                        ;USER MODE REG. SET #1 ON
024520  012702  044000                     MOV     #SM+REG,R2
024524  074002                             XOR     R0,R2
024526  001447                             BEQ     3$                        ;SUPER MODE REG. SET #1 ON
024530  032700  140000                     BIT     #UM,R0
024534  001062                             BNE     RTT2EX

                                  ;TEST THAT RTT CLEARS BITS 11,12,13 & PRIORITY LEVEL BITS IN KERNEL MODE
024536  012702  177777                     MOV     #-1,R2                    ;KERNEL MODE REG. SET 0 ON
024542  012737  034240  177776             MOV     #PUM+REG+PR5,@#PSW        ;SELECT REG. SET #1
024550  005002                             CLR     R12                       ;SHOULD CLEAR REG #12
024552  012746  000100                     MOV     #PR2,-(SP)
024556  010746                             MOV     PC,-(SP)
024560  062716  000006                     ADD     #1$-.,(SP)                ;FORM NEW PC
024564  000006                             RTT
024566  013700  177776            1$:      MOV     @#PSW,R0                  ;NOW USING REG SET 0
024572  005702                             TST     R2                        ;SHOULD TEST R2 NOT R12
024574  001001                             BNE     4$
024576  104000                             HLT                               ;ERROR!DID NOT CLEAR BIT #11 OF PSW
024600  022700  000100            4$:      CMP     #PR2,R0          ;TESTS THE PSW AFTER THE RTT
024604  001436                             BEQ     RTT2EX
024606  104000                             HLT                               ;ERROR! INCORRECT PSW AFTER THE RTT
024610  000434                             BR      RTT2EX

                                  ;TEST TO INSURE THAT RTI DOES NOT CLEAR BITS 11-15 IN USER MODE
024612  052737  030340  177776    2$:      BIS     #PUM+PR7,@#PSW   ;PSW<15-5>=144X
024620  005046                             CLR     -(SP)
024622  010746                             MOV     PC,-(SP)
024624  062716  000006                     ADD     #5$-.,(SP)
024630  000002                             RTI                               ;ATTEMPS TO INSERT A PSW OF 0
024632  022737  174340  177776    5$:      CMP     #UM+PUM+REG+PR7,@#PSW     ;SHOULD CHECK AGAINST REG #0
024640  001420                             BEQ     RTT2EX
024642  104000                             HLT                               ;ERROR! RTI CLEARED BITS IN PSW
024644  000416                             BR      RTT2EX

                                  ;TEST THAT BITS 11-15 AND PRIORTY BITS ARE NOT ALTERED IN SUPER MODE
024646  052737  030200  177776    3$:      BIS     #PUM+PR4,@#PSW   ;PSW<15-5>=044X
```

```
024660  010746                            MOV     PC,-(SP)
024662  062716  000006                    ADD     #6$-.,(SP)
024666  000006                            RTT                                     ;ATTEMPTS TO CLEAR 11-15 AND ALTER PR

024670  022737  074200  177776  6$:       CMP     #SM+PUM+REG+PR4,@#PSW
024676  001401                            BEQ     RTT2EX
024700  104000                            HLT                                     ;ERROR! RTT ALTERED PR IN
                                                                                  ;SUPER MODE OR BITS 11-15.
024702  016737  177572  177776  RTT2EX:   MOV     SAVPSW,@#PSW
024710  000004                  RELES:    SCOPE
024712  010702                            MOV     PC,R2
024714  062702  000012                    ADD     #12,R2
024720  012707  034242                    MOV     #RELOC,PC         ;GO RELOCATE PROGRAM CODE
024724  000000                  REL55:    .WORD   0
                                ;555555555555 LAST ADDRESS OF CODE TO BE RELOCATED 5555555555

                                ;;*****************************************************************
                                ;*TEST 53         CHECK ASH, ASHC, MUL, AND DIV INSTRUCTIONS
                                ;;*****************************************************************
024726  112737  000053  001202  TST53:    MOVB    #53,@#$TSTNM              ;LOAD TEST NUMBER
024734  012767  000001  154354            MOV     #1,$TIMES         ;;DO 1 ITERATION
024742  000004                            SCOPE

                                          .SBTTL  START OF SECTION 6
                                ;6666666666666 FIRST ADDRESS TO BE RELOCATED 666666666
024744  010700                  REL6:     MOV     PC,R0             ;GET PC
024746  005740                            TST     -(R0)             ;R0 CONTAINS THE ADDRESS OF REL6
024750  010037  001512                    MOV     R0,@#FRSTAD       ;SAVE
024754  010700                            MOV     PC,R0             ;GET CURRENT PC
024756  162700  024756                    SUB     #.,R0             ;SUBTRACT RELOCATION FACTOR
024762  010037  001506                    MOV     R0,@#FACTOR       ;SAVE RELOCATION FACTOR
024766  010737  001212                    MOV     PC,@#$LPERR       ;SET LOOP ADDRESS
024772  062737  000030  001212            ADD     #30,@#$LPERR      ;ADJUST
025000  013737  001212  001210            MOV     @#$LPERR,@#$LPADR
025006  105737  001502                    TSTB    @#NEXEC           ;BR IF TEST CODE TO BE EXECUTED
025012  001402                            BEQ     .+6
025014  000167  002016                    JMP     RELE6
025020  012700  000001          ASHLO:    MOV     #1,R0             ;R0 WILL BE THE SHIFT COUNT
025024  012703  000021                    MOV     #17.,R3           ;MAX SHIFT COUNT
025030  005067  000014          1$:       CLR     2$                ;PRESET SAVED CC'S LOCATION=0
025034  010002                            MOV     R0,R2             ;GET SHIFT COUNT FOR PASS
025036  010705                            MOV     PC,R5             ;R5 & R4 WILL BE DATA SHIFTED BY
025040  010504                            MOV     R5,R4             ;ASH & ASL INSTRUCTIONS
025042  072502                            ASH     R2,R5             ;SHIFT R5
025044  113727  177776                    MOVB    @#PSW,(PC)+       ;SAVE CC'S
025050  000000                  2$:       .WORD   0                 ;CONTAINS ASH CC'S IN EVEN BYTE
                                                                    ;ASL CC'S IN ODD BYTE
025052  006304                  3$:       ASL     R4                ;SHIFT R4
025054  113746  177776                    MOVB    @#PSW,-(SP)       ;SAVE PSW ON STACK
025060  132716  000002                    BITB    #2,(SP)           ;CHECK IF ASL SET V BIT
025064  001403                            BEQ     30$
025066  152767  000002  177755            BISB    #2,2$+1           ;IF ASL SET V THEN SET V IN 2$+1
025074  112637  177776          30$:      MOVB    (SP)+,@#PSW       ;RESTORE ORIGINAL PSW
025100  077214                            SOB     R2,3$             ;SHIFT R4 R2 TIMES
025102  153767  177776  177741            BISB    @#PSW,2$+1        ;SAVE CC'S AFTER ASL
```

```
025112  001004                           BNE     4$
025114  126767  177730  177727           CMPB    2$,2$+1         ;CHECK ASH & ASL CC'S
025122  001401                           BEQ     .+4
025124  104000                  4$:      HLT                     ;ERROR! INCORRECT RESULT OR CC'S
025126  005200                           INC     R0              ;INCREMENT PASS SHIFT COUNT
025130  020003                           CMP     R0,R3
025132  001336                           BNE     1$

025134  012700  177777          ASHR0:   MOV     #-1,R0          ;R0 = RIGHT SHIFT COUNT FOR PASS
025140  012703  177757                   MOV     #-17.,R3        ;MAX SHIFT COUNT
025144  010002                  1$:      MOV     R0,R2           ;GET SHIFT COUNT FOR PASS
025146  010705                           MOV     PC,R5           ;R5 & R4 = DATA TO BE SHIFTED
025150  010504                           MOV     R5,R4           ;BY ASH & ASR INSTRUCTIONS
025152  072502                           ASH     R2,R5           ;SHIFT R5 R2 TIMES
025154  113727  177776                   MOVB    @#PSW,(PC)+     ;SAVE CC'S IN EVEN BYTE
025160  000000                  2$:      .WORD   0               ;CONTAINS ASH CC'S IN EVEN BYTE
                                                                 ;ASR CC'S IN ODD BYTE
025162  005402                           NEG     R2
025164  006204                  3$:      ASR     R4              ;SHIFT R4
025166  077202                           SOB     R2,3$           ;SHIFT R4 R2 TIMES
025170  113767  177776  177763           MOVB    @#PSW,2$+1      ;SAVE CC'S AFTER ASR
025176  142767  000002  177755           BICB    #2,2$+1         ;ASH RIGHT WILL NOT SET V ASR MAY SET V
025204  020504                           CMP     R5,R4           ;CHECK ASH & ASR RESULTS
025206  001004                           BNE     4$
025210  126767  177744  177743           CMPB    2$,2$+1         ;CHECK ASH & ASR CC'S
025216  001401                           BEQ     .+4
025220  104000                  4$:      HLT
025222  005300                           DEC     R0              ;DECREMENT PASS SHIFT COUNT
025224  020003                           CMP     R0,R3
025226  001346                           BNE     1$

025230  012746  000037          ASHCL0:  MOV     #31.,-(SP)      ;PUT MAX SHIFT COUNT ON STACK
025234  012746  000001                   MOV     #1,-(SP)        ;PUT LEFT SHIFT COUNT ON STACK
025240  011600                  1$:      MOV     (SP),R0         ;GET PASS SHIFT COUNT
025242  010705                           MOV     PC,R5           ;CURRENT PC IS DATA TO BE SHIFTED
025244  010503                           MOV     R5,R3           ;ASHC SHIFTS R4,R5;ASL,ROL SHIFTS R2,R3
025246  005004                           CLR     R4
025250  005002                           CLR     R2
025252  073400                           ASHC    R0,R4           ;SHIFT R4 LEFT AS SPECIFIED BY R0
025254  006303                  2$:      ASL     R3              ;SHIFT R2,R3 LEFT
025256  006102                           ROL     R2              ;AS SPECIFIED BY R0
025260  077003                           SOB     R0,2$
025262  020402                           CMP     R4,R2           ;CHECK RESULTS
025264  001002                           BNE     3$
025266  020503                           CMP     R5,R3
025270  001401                           BEQ     .+4
025272  104000                  3$:      HLT
025274  005216                           INC     (SP)            ;INCREMENT NEXT PASS SHIFT COUNT
025276  021666  000002                   CMP     (SP),2(SP)      ;REACHED MAX COUNT (31.)
025302  001356                           BNE     1$
025304  022626                           CMP     (SP)+,(SP)+     ;RESTORE STACK PTR

025306  012746  177740          ASHCR0:  MOV     #-32.,-(SP)     ;PUT MAX RIGHT SHIFT COUNT ON STACK
025312  012746  177777                   MOV     #-1,-(SP)       ;PUT PASS SHIFT COUNT ON STACK
025316  011600                  1$:      MOV     (SP),R0         ;GET PASS SHIFT COUNT
```

```
 025322  010204                           MOV     R2,R4           ;TO BE SHIFTED BY TEST
 025324  005003                           CLR     R3
 025326  005005                           CLR     R5
 025330  000262                           SEV                     ;SET V BIT IN PSW
 025332  073200                           ASHC    R0,R2           ;SHIFT R2,R3 RIGHT R0 TIMES
 025334  102410                           BVS     3$              ;SHIFT RIGHT CLEARS V
 025336  005400                           NEG     R0              ;NEGATE SHIFT COUNT FOR SOB
 025340  006204                   2$:     ASR     R4              ;SHIFT R4,R5 RIGHT R0 TIMES
 025342  006005                           ROR     R5
 025344  077003                           SOB     R0,2$
 025346  020204                           CMP     R2,R4           ;CHECK RESULT
 025350  001002                           BNE     3$
 025352  020305                           CMP     R3,R5
 025354  001401                           BEQ     .+4
 025356  104000                   3$:     HLT
 025360  005316                           DEC     (SP)            ;SET SHIFT COUNT FOR NEXT PASS
 025362  021666  000002                   CMP     (SP),2(SP)      ;CHECK IF MAX SHIFT COUNT
 025366  001353                           BNE     1$
 025370  022626                           CMP     (SP)+,(SP)+     ;RESTORE STACK PTR
                                  ;;*****************************************************************
                                  ;*TEST 54         CHECK MUL
                                  ;*       THE BELOW TEST OF THE MUL INSTRUCTION MULTIPLIES THE CURRENT PC
                                  ;*       BY 1,2,4,8 ETC AND SHIFTS THE SAME PC VALUE USING AN ASHC LEFT BY
                                  ;*       0,1,2,3,ETC AND COMPARES THE RESULTS. CONDITION CODE RESULTS ARE NOT CHECKED.
                                  ;;*****************************************************************
 025372  112737  000054  001202   TST54:  MOVB    #54,@#STSTNM            ;LOAD TEST NUMBER
 025400  000004                           SCOPE
 025402  012700  000001           MUL0:   MOV     #1,R0           ;R0 CONTAINS MULTIPLIER FOR MUL
 025406  012706  000700                   MOV     #SUPSTK,SP      ;SETUP THE STACK
 025412  005016                           CLR     (SP)            ;(SP) CONTAINS SHIFT VALUE FOR ASHC
 025414  010702                   1$:     MOV     PC,R2           ;R3,R2 & R5,R4 ARE DATA REGISTERS
 025416  010227                           MOV     R2,(PC)+        ;SAVE MULTIPICAND
 025420  000000                           .WORD   0               ;CONTAINS ORIGINAL MULTIPICAND
 025422  005003                           CLR     R3
 025424  005004                           CLR     R4
 025426  010205                           MOV     R2,R5           ;FOR MUL AND ASHC
 025430  100001                           BPL     .+4             ;IF MULTPICAND IS NEG THEN SET R4 = -1
 025432  005104                           COM     R4              ;FOR ASHC
 025434  000277                           SCC                     ;PRESET CC'S
 025436  070200                           MUL     R0,R2           ;MULTIPLY R2 BY R0 LEAVE PRODUCT
                                                                  ;IN R2,R3 MSH IN R2,LSH IN R3
 025440  102406                           BVS     2$
 025442  001405                           BEQ     2$              ;PRODUCT WILL NEVER BE = 0
 025444  073416                           ASHC    (SP),R4         ;'MULTIPLY' R4,R5 BY (SP) LEAVE PRODUCT
                                                                  ;IN R4,R5 MSH IN R4,LSH IN R5
 025446  020204                           CMP     R2,R4           ;CHECK MSH RESULT
 025450  001002                           BNE     2$
 025452  020305                           CMP     R3,R5           ;CHECK LSH RESULT
 025454  001401                           BEQ     .+4
 025456  104000                   2$:     HLT
 025460  005216                           INC     (SP)            ;INCREMENT ASHC SHIFT COUNT
 025462  006300                           ASL     R0              ;SHIFT MUL MULTIPLIER
 025464  102353                           BVC     1$
                                  ;CHECK MUL INST WITH MULTIPIER (R0) = 100000
 025466  010702                           MOV     PC,R2           ;R2 = MULTIPICAND
```

```
025472  010227                                  MOV     R2,(PC)+                ;SAVE MULTIPICAND
025474  000000                                  .WORD   0                       ;CONTAINS ORIGINAL MULTIPICAND
025476  005103                                  COM     R3
025500  010204                                  MOV     R2,R4                   ;R4 WILL BE MSH 'PRODUCT'
025502  006204                                  ASR     R4                      ;FORM 'PRODUCT'
025504  005104                                  COM     R4                      ;COMPLEMENT MSH 'PRODUCT'
025506  070200                                  MUL     R0,R2                   ;MULTIPLY R2 BY 100000 LEAVING
                                                                                ;R2 = MSH, R3 = LSH PRODUCT
025510  020204                                  CMP     R2,R4                   ;COMPARE MSH PRODUCTS
025512  001002                                  BNE     3$
025514  020003                                  CMP     R0,R3                   ;CHECK LSH PRODUCT
025516  001401                                  BEQ     .+4
025520  104000                          3$:     HLT
                                        ;;*************************************************************
                                        ;*TEST 55       CHECK THE DIV INSTRUCTION
                                        ;*              THE BELOW TEST OF THE DIV INSTRUCTION DIVIDES THE CURRENT PC BY
                                        ;*              1,2,4,8,ETC LEAVING THE QUOTIENT/REMAINDER IN R2/R3. NEXT THE QUOTIENT
                                        ;*              IS MULTPLIED BY 1,2,4,8,ETC AND THE REMAINDER ADDED. THE RESULT IS
                                        ;*              THEN COMPARED WITH THE ORIGINAL CURRENT PC.
                                        ;;*************************************************************
025522  112737  000055  001202          TST55:  MOVB    #55,@#$TSTNM                    ;LOAD TEST NUMBER
025530  000004                                  SCOPE
025532  012700  000001                  DIV0:   MOV     #1,R0                   ;R0=DIVISOR
025536  010716                                  MOV     PC,(SP)                 ;SAVE DATA ON STACK
025540  011603                          1$:     MOV     (SP),R3                 ;GET DATA
025542  005002                                  CLR     R2                      ;CLEAR MSH DIVIDEND
025544  000277                                  SCC
025546  071200                                  DIV     R0,R2                   ;DIVIDE R2 BY R0 LEAVING QUOTIENT IN R2
                                                                                ;AND REMAINDER IN R3
025550  103417                                  BCS     2$
025552  100416                                  BMI     2$
025554  102007                                  BVC     20$                     ;BRANCH IF DIVIDE WORKED
025556  022700  000001                          CMP     #1,R0                   ;V BIT SHOULD ONLY SET IF DIVIDING BY 1
025562  001012                                  BNE     2$                      ;AND THE LSH OF DIVIDEND
025564  032716  100000                          BIT     #100000,(SP)            ;IS NEGATIVE
025570  001407                                  BEQ     2$
025572  000407                                  BR      3$
025574  010204                          20$:    MOV     R2,R4                   ;GET QUOTIENT
025576  070400                                  MUL     R0,R4                   ;MULTIPLY QUOTIENT BY DIVISOR
025600  060305                                  ADD     R3,R5                   ;ADD REMAINDER TO LSH PRODUCT
025602  103402                                  BCS     2$                      ;SHOULD BE NO CARRY
025604  021605                                  CMP     (SP),R5                 ;CHECK RESULT
025606  001401                                  BEQ     .+4
025610  104000                          2$:     HLT                             ;ERROR! DIVIDE FAILED
                                                                                ;QUOTIENT IS IN R2,REMAINDER IN R3
                                                                                ;ORIGINAL PC IS ON STACK AND FINAL
                                                                                ;PRODUCT IN R4,R5 [MSH][LSH]
025612  006300                          3$:     ASL     R0                      ;GET NEXT DIVISOR
025614  102351                                  BVC     1$

                                        ;CHECK ASH,ASHC,MUL, AND DIV INSTRUCTIONS USING ADDRESS MODE 1
025616  005016                          ASHL1:  CLR     (SP)                    ;(SP) = SHIFT COUNT
025620  005000                                  CLR     R0                      ;R0 = SHIFT COUNT FOR CHECK ASH
025622  012702  000020                          MOV     #16.,R2                 ;R2 = MAX LEFT SHIFT COUNT
025626  005067  000012                  1$:     CLR     2$                      ;CLEAR CC'S HOLDING ADDRESS
```

```
025634  010304                          MOV     R3,R4
025636  072316                          ASH     (SP),R3                 ;SHIFT R3 LEFT (SP) TIMES
025640  013727  177776                  MOV     @#PSW,(PC)+             ;SAVE CC'S
025644  000000                  2$:     .WORD   0                       ;CONTAINS ASH (SP),R3 CC'S IN EVEN BYTE
                                                                        ;AND ASH R0,R4 CC'S IN ODD BYTE
025646  072400                          ASH     R0,R4                   ;SHIFT R4 LEFT R0 TIMES
025650  113767  177776  177767          MOVB    @#PSW,2$+1              ;SAVE CC'S IN ODD BYTE OF 2$
025656  020304                          CMP     R3,R4                   ;COMPARE RESULTS
025660  001004                          BNE     3$                      ;BRANCH IF THEY DO NOT COMPARE
025662  126767  177756  177755          CMPB    2$,2$+1                 ;CHECK CC'S AFTER ASH INSTRUCTIONS
025670  001401                          BEQ     .+4
025672  104000                  3$:     HLT                             ;ERROR! EITHER RESULTS OF SHIFT OR
                                                                        ;RESULT CC'S ARE INCORRECT
025674  005200                          INC     R0                      ;INCREMENT SHIFT COUNT FOR ASH R0,R4
025676  005216                          INC     (SP)                    ;INCREMENT SHIFT COUNT FOR ASH (SP),R3
025700  020200                          CMP     R2,R0                   ;CHECK FOR MAX SHIFT COUNT
025702  001351                          BNE     1$

025704  005016                  ASHR1:  CLR     (SP)                    ;(SP) = SHIFT COUNT FOR ASH (SP),R4
025706  005000                          CLR     R0                      ;R0 = SHIFT COUNT FOR ASH R0,R5
025710  005402                          NEG     R2                      ;R2 = MAX RIGHT SHIFT COUNT (SET BY
                                                                        ;ABOVE TEST TO 16. NOW = -16.
025712  005067  000012          1$:     CLR     2$                      ;CLEAR CC'S HOLDING ADDRESS
025716  010704                          MOV     PC,R4                   ;R4,R5 = DATA TO BE SHIFTED RIGHT
025720  010405                          MOV     R4,R5
025722  072416                          ASH     (SP),R4                 ;SHIFT R4 RIGHT (SP) TIMES
025724  013727  177776                  MOV     @#PSW,(PC)+             ;SAVE CC'S
025730  000000                  2$:     .WORD   0                       ;CONTAINS ASH (SP),R4 CC'S IN EVEN BYTE
                                                                        ;AND ASH R0,R5 CC'S IN ODD BYTE
025732  072500                          ASH     R0,R5                   ;SHIFT R5 RIGHT R0 TIMES
025734  113767  177776  177767          MOVB    @#PSW,2$+1              ;SAVE CC'S IN ODD BYTE 2$
025742  020405                          CMP     R4,R5                   ;CHECK RESULTS
025744  001004                          BNE     3$
025746  126767  177756  177755          CMPB    2$,2$+1                 ;CHECK RESULT CC'S
025754  001401                          BEQ     .+4
025756  104000                  3$:     HLT                             ;ERROR! EITHER RESULTS OR RESULT CC'S
                                                                        ;DID NOT COMPARE
025760  005300                          DEC     R0                      ;DECREMENT SHIFT COUNT
025762  005316                          DEC     (SP)                    ;DECREMENT SHIFT COUNT FOR ASH (SP),R4
025764  020002                          CMP     R0,R2                   ;CHECK FOR MAX RIGHT SHIFT
025766  001351                          BNE     1$
                                ;;*****************************************************************
                                ;*TEST 56               DIVIDE AGAIN
                                ;*      THE BELOW TEST CHECKS THE DIVIDE INSTRUCTION BY DIVIDING
                                ;*      THE CURRENT PC BY ITSELF+1. THE QUOTIENT (IN R2) ALWAYS = 0,
                                ;*      AND THE REMAINDER (IN R3) ALWAYS = THE CURRENT PC.
                                ;;*****************************************************************
025770  112737  000056  001202  TST56:  MOVB    #56,@#$TSTNM            ;LOAD TEST NUMBER
025776  000004                          SCOPE
026000  010703                  DIV1:   MOV     PC,R3                   ;CURRENT PC IS LSH DIVIDEND
026002  006702                          SXT     R2                      ;EXTEND SIGN TO R2 (MSH DIVIDEND)
026004  010304                          MOV     R3,R4                   ;SAVE ORIGINAL DIVIDEND
026006  010316                          MOV     R3,(SP)                 ;PUT ON STACK
026010  005216                          INC     (SP)                    ;ADD 1 (WILL BE DIVISOR)
026012  100002                          BPL     1$                      ;BRANCH IF POSITIVE
```

```
026020  071216                      1$:     DIV     (SP),R2         ;DIVIDE R2 BY (SP)
026022  103410                              BCS     2$              ;CHECK CONDITION CODES
026024  102407                              BVS     2$
026026  001006                              BNE     2$
026030  100405                              BMI     2$
026032  005702                              TST     R2              ;CHECK QUOTIENT (R2 = 0)
026034  001361                              BNE     DIV1
026036  010416                              MOV     R4,(SP)         ;GET ORIGINAL DIVISOR
026040  020316                              CMP     R3,(SP)         ;CHECK REMAINDER
026042  001401                              BEQ     .+4
026044  104000                      2$:     HLT                     ;REPORT ERROR
                                    ;;*****************************************************************
                                    ;*TEST 57       CHECK SPL INSTRUCTION
                                    ;;*****************************************************************
026046  112737  000057  001202      TST57:  MOVB    #57,@#$TSTNM            ;LOAD TEST NUMBER
026054  000004                              SCOPE
026056  012702                      SPLTST: MOV     (PC)+,R2        ;R2 CONTAINS OP CODE FOR SPL  7
026060  000237                              SPL     7
026062  005004                              CLR     R4
026064  042744  000340                      BIC     #PR7,-(R4)      ;CLEAR PRIORITY LEVEL BITS IN PSW
026070  011403                              MOV     (R4),R3         ;GET CURRENT PSW
026072  042703  177757                      BIC     #177757,R3      ;R3 CONTAINS CORRECT PSW AFTER SPL

026076  012767  000230  000010              MOV     #SPL+0,2$       ;INITIALIZE SPL INSTRUCTIONS
026104  012767  000237  000050              MOV     #SPL+7,5$
026112  000257                      1$:     CCC                     ;CLEAR CONDITION CODES
026114  000230                      2$:     SPL     0               ;SET PRIORITY LEVEL (NOTE: SPL=NOP IF USER/SUPER MODE)
026116  121403                              CMPB    (R4),R3         ;CHECK RESULT OF SPL ABOVE
026120  001401                              BEQ     .+4
026122  104000                              HLT                     ;ERROR! SPL ABOVE FAILED
026124  032714  140000                      BIT     #UM,(R4)        ;IF NOT IN KERNEL MODE THEN SPL
026130  001002                              BNE     3$              ;ACTS AS A NOP
026132  062703  000040                      ADD     #40,R3          ;SET NEXT CORRECT PSW RESULT
026136  005267  177752              3$:     INC     2$              ;SET NEXT SPL INSTRUCTION
026142  026702  177746                      CMP     2$,R2           ;CHECK IF DONE
026146  002761                              BLT     1$              ;LOOP UNTIL DONE CHANGING SPL EACH PASS
026150  012702                              MOV     (PC)+,R2        ;R2 CONTAINS SPL INSTRUCTION BELOW
026152  000230                              SPL     0
026154  052703  000017                      BIS     #17,R3          ;SET CONDITION CODE RESULT INTO R3
026160  000277                      4$:     SCC                     ;SET CONDITION CODES
026162  000237                      5$:     SPL     7               ;SET PRIORITY LEVEL
026164  121403                              CMPB    (R4),R3         ;CHECK RESULT OF SPL ABOVE
026166  001401                              BEQ     .+4
026170  104000                              HLT                     ;ERROR! SPL ABOVE FAILED
026172  032714  140000                      BIT     #UM,(R4)        ;CHECK IF IN KERNEL MODE
026176  001002                              BNE     6$
026200  162703  000040                      SUB     #40,R3          ;SET NEXT CORRECT PSW RESULT
026204  005367  177752              6$:     DEC     5$              ;SET NEXT SPL
026210  026702  177746                      CMP     5$,R2           ;CHECK IF DONE ALL SPL'S
026214  002361                              BGE     4$
                                    ;;*****************************************************************
                                    ;*TEST 60       CHECK PIRQ LOGIC
                                    ;*      THIS TEST CHECKS THAT WHEN A REQUEST IS MADE AT A LEVEL = TO THE
                                    ;*      CURRENT PROCESSER PRIORITY LEVEL THAT NO INTERRUPT TAKES PLACE, AND
                                    ;*      THAT WHEN A REQUEST IS MADE AT A LEVEL 1 GREATER THAN THE CURRENT PRO-
```

```
                                        ;;************************************************************
026216  112737  000060  001202  TST60:  MOVB    #60,@#$TSTNM                    ;LOAD TEST NUMBER
026224  000004                          SCOPE
026226  012700  026372          PIRQ0:  MOV     #4$,R0                  ;R0 POINTS TO A TABLE OF CORRECT PIRQ
                                                                        ;CONTENTS AFTER AN INTERRUPT
026232  012702  000400                  MOV     #400,R2                 ;R2 CONTAINS INTERRUPT REQUEST LEVEL
026236  005003                          CLR     R3                      ;R3 CONTAINS PROCESSER PRIORITY LEVEL
026240  012704  177772                  MOV     #PIRQ,R4                ;R4 CONTAINS ADDRESS OF PIRQ REGISTER
026244  005014                          CLR     (R4)                    ;INITIALZE REQUEST LEVEL TO 0
026246  013737  177776  000242          MOV     @#PSW,@#PIRQVEC+2          ;RETAIN MODE & REG SET ON TRAP
026254  112737  000340  000242          MOVB    #PR7,@#PIRQVEC+2           ;ASSUME LEVEL 7 ON INTERRUPT
026262  112737  000340  000016          MOVB    #PR7,@#TBITVEC+2           ;PRIORITY LEVEL 7 ON TRAP
026270  012737  026330  000240  1$:     MOV     #2$,@#PIRQVEC              ;SET PIRQ ERROR INTERRUPT VECTOR
026276  063737  001506  000240          ADD     @#FACTOR,@#PIRQVEC         ;ADD RELOCATION FACTOR
026304  110337  177776                  MOVB    R3,@#PSW                   ;SET CP PRIORITY LEVEL
026310  050214                          BIS     R2,(R4)                    ;MAKE REQUEST AT LEVEL = TO CP LEVEL
026312  100436                          BMI     5$                         ;BRANCH WHEN DONE
026314  062737  000002  000240          ADD     #3$-2$,@#PIRQVEC                   ;SET PIRQ INTERRUPT VECTOR TO 3$
026322  006302                          ASL     R2
026324  050214                          BIS     R2,(R4)                    ;MAKE REQUEST AT LEVEL 1 HIGHER
026326  000240                          NOP
026330  104000                  2$:     HLT                                ;ERROR! EITHER AN INTERRUPT OCCURED
                                                                           ;WHEN RQST LEVEL = CP LEVEL (PIRQVEC)=2$
                                                                           ;OR INTERRUPT FAILED (PIRQVEC)=3$
026332  022014                  3$:     CMP     (R0)+,(R4)                 ;CHECK CONTENTS OF PIRQ REGISTER
026334  001406                          BEQ     6$
026336  013737  177772  001276          MOV     @#PIRQ,@#$TMP0             ;SAVE PIRQ
026344  005037  177772                  CLR     @#PIRQ
026350  104000                          HLT                                ;ERROR! INCORRECT PIRQ CONTENTS
026352  062703  000040          6$:     ADD     #40,R3                     ;SET NEXT CP PRIORITY LEVEL
026356  040214                          BIC     R2,(R4)                    ;LOWER LEVEL BY 1
026360  012716  026270                  MOV     #1$,(SP)                   ;ADJUST RETURN ADDRESS
026364  063716  001506                  ADD     @#FACTOR,(SP)              ;TO RETURN TO 1$
026370  000006                  30$:    RTT

                                ;TABLE OF CORRECT PIRQ REGISTER CONTENTS ON INTERRUPT
026372  001042                  4$:     1042                       ;PIR1+PIA1
026374  003104                          3104                       ;PIR2+PIR1+PIA2
026376  007146                          7146                       ;PIR3+PIR2+PIR1+PIA3
026400  017210                          17210                      ;PIR4+PIR3+PIR2+PIR1+PIA4
026402  037252                          37252                      ;PIR5+PIR4+PIR3+PIR2+PIR1+PIA5
026404  077314                          77314                      ;PIR6+PIR5+PIR4+PIR3+PIR2+PIR1+PIA6
026406  177356                          177356                     ;PIR7+PIR6+PIR5+PIR4+PIR3+PIR2+PIR1+PIA7

026410  005014                  5$:     CLR     (R4)                       ;CLEAR PIRQ REGISTER
026412  012737  000242  000240          MOV     #PIRQVEC+2,@#PIRQVEC       ;RESET PIRQVEC TO HALT AT PIRQVEC+2
026420  005037  000242                  CLR     @#PIRQVEC+2
026424  105037  177776                  CLRB    @#PSW
026430  042737  000340  000016          BIC     #PR7,@#TBITVEC+2
                                ;;************************************************************
                                ;*TEST 61        CHECK MICRO-BREAK REGISTER
                                ;*       THIS TEST SHIFTS A '0' BIT THRU ALL BIT POSITIONS.
                                ;
                                ;;************************************************************
026436  112737  000061  001202  TST61:  MOVB    #61,@#$TSTNM               ;LOAD TEST NUMBER
026444  000004                          SCOPE
```

```
026452  011246                            MOV     (R2),-(SP)         ;SAVE ORIG CONTENTS
026454  012700  000376                    MOV     #376,R0            ;SET DATA PATTERN
026460  010012                   1$:      MOV     R0,(R2)            ;LOAD REGISTER WITH PATTERN
026462  021200                            CMP     (R2),R0            ;AND CHECK
026464  001004                            BNE     3$                 ;BRANCH IF INCORRECT
026466  000261                   2$:      SEC                        ;SET 'C'
026470  006100                            ROL     R0                 ;SHIFT DATA
026472  103772                            BCS     1$
026474  000402                            BR      4$
026476  104000                   3$:      HLT                        ;ERROR DATA IN R0 NOT IN UBREAK REG
026500  000772                            BR      2$                 ;CONTINUE TEST
026502  012612                   4$:      MOV     (SP)+,(R2)         ;RESTORE ORIG UBREAK CONTENTS
                                 ;;*****************************************************************
                                 ;*TEST 62         CHECK MFPI/MTPI INSTRUCTIONS
                                 ;;*****************************************************************
026504  112737  000062  001202   TST62:   MOVB    #62,@#STSTNM                 ;LOAD TEST NUMBER
026512  000004                            SCOPE
026514  032737  140000  177776   MPI:     BIT     #UM,@#PSW          ;KERNEL MODE?
026522  001545                            BEQ     ENDCP              ;YES EXIT TEST
026524  010746                            MOV     PC,-(SP)
026526  062716  000134                    ADD     #5$-.,(SP)
026532  012637  000250                    MOV     (SP)+,@#MMVEC      ;SET MEM MGMT ABORT VECTOR
026536  005046                            CLR     -(SP)              ;CLEAR CHECK WORD
026540  010603                            MOV     SP,R3
026542  010346                            MOV     R3,-(SP)           ;PUT ADDRESS OF CHECK WORD ON THE STACK
026544  105737  001503                    TSTB    @#MMON             ;CHECK IF MEM MGMT IS ENABLED
026550  001417                            BEQ     1$                 ;BRANCH IF OFF
026552  013737  177640  177654            MOV     @#UIPAR0,@#UIPAR6        ;SET UP USER PAGE ADDR. REG.
026560  012737  006006  177614            MOV     #6006,@#UIPDR6           ;SET USER PAGE DESC REG R/W UP 6 PAGES
026566  013737  172240  172254            MOV     @#SIPAR0,@#SIPAR6
026574  012737  006006  172214            MOV     #6006,@#SIPDR6     ;SET SUPER PAGE DESC. REG.
026602  062706  140000           10$:     ADD     #140000,SP         ;SET CURRENT MODE'S STACK POINTER
026606  000240                            NOP
026610  010746                   1$:      MOV     PC,-(SP)
026612  062716  000024                    ADD     #3$-.,(SP)
026616  012637  000020                    MOV     (SP)+,@#IOTVEC     ;SET IOT TRAP VECTOR
026622  000004                            IOT                        ;TRAP TO 3$ BELOW
026624  005266  000002                    INC     2(SP)              ;INCREMENT CHECK WORD
026630  001417                            BEQ     6$
026632  104000                   4$:      HLT                        ;ERROR! MFPI,MTPI FAILURE-FOR BETTER
026634  000415                            BR      6$                 ;ISOLATION SUGGEST RUNNING MFPI DIAG. DCKTD/E
026636  000240                   3$:      NOP                        ;PSW=KERNEL MODE,PREV USER OR SUPER MODE
026640  006506                            MFPI    SP                 ;GET PREV. MODES STACK POINTER
026642  006536                            MFPI    @(SP)+             ;GET DATA (AN ADDRESS) ON PREV MODE'S STACK
026644  006576  000000                    MFPI    @(SP)              ;GET DATA (=0) FROM PREV MODES ADDRESS
026650  000240                            NOP                        ;SPACE AND PUSH ONTO KERNEL STACK
026652  001367                            BNE     4$                 ;ERROR IF BRANCH TAKEN! SHOULD HAVE A ZERO ON THE STACK
026654  005116                            COM     (SP)               ;COMPLEMENT OPERAND
026656  006636                            MTPI    @(SP)+             ;POP OPERAND OFF KERNEL STACK AND MOVE
                                                                     ;IT TO PREV MODE'S SPACE
026660  000002                            RTI                        ;RETURN TO INST FOLLOWING IOT ABOVE
026662  104000                   5$:      HLT                        ;ERROR! MEMORY MANG. ABORT
026664  105037  177776                    CLRB    @#PSW              ;SET PRIORITY LEVEL BACK TO 0
026670  012737  053274  000250   6$:      MOV     #KTABRT,@#MMVEC    ;RESTORE VECTOR
026676  012737  044534  000020            MOV     #$SCOPE,@#IOTVEC
```

```
                                        ;;*********************************************************************
                                        ;*TEST 63         CHECK ILLEGAL HALT
                                        ;;*********************************************************************
026710  112737  000063  001202          TST63:  MOVB    #63,@#STSTNM            ;LOAD TEST NUMBER
026716  000004                                  SCOPE
026720  010746                          HALT1:  MOV     PC,-(SP)        ;GET CURRENT PC
026722  062716  000022                          ADD     #2$-.,(SP)
026726  011637  000004                          MOV     (SP),@#ERRVEC   ;SET ERROR TRAP VECTOR TO 2$ BELOW
026732  012637  000010                          MOV     (SP)+,@#RESVEC  ;LOAD RESERVED INST TRAP VECTOR (11/40)
026736  000000                                  HALT                    ;SHOULD TRAP TO 4 IN USER/SUPER MODE
026740  104000                          1$:     HLT                     ;ERROR! HALT ABOVE FAILED IN USER/SUPER MODE
026742  000404                                  BR      3$
026744  010716                          2$:     MOV     PC,(SP)         ;REPLACE RETURN PC WITH
026746  062716  000006                          ADD     #3$-.,(SP)      ;ADDRESS OF 3$ BELOW
026752  000002                                  RTI                     ;RETURN (TO 3$)

026754  012737  053442  000004          3$:     MOV     #ERPRT,@#ERRVEC ;RESTORE ERROR TRAP VECTOR
026762  012737  053370  000010                  MOV     #RESERR,@#RESVEC
026770  105037  177776                          CLRB    @#PSW
026774  005037  177766                          CLR     @#CPUERR
                                        ;;*********************************************************************
                                        ;*TEST 64         CHECK RESET IN SUPER/USER MODE
                                        ;;*********************************************************************
027000  112737  000064  001202          TST64:  MOVB    #64,@#STSTNM            ;LOAD TEST NUMBER
027006  000004                                  SCOPE
027010  000277                          RESET1: SCC
027012  013700  177776                          MOV     @#PSW,R0        ;GET CURRENT PSW
027016  000277                                  SCC
027020  000005                                  RESET
027022  023700  177776                          CMP     @#PSW,R0        ;CHECK THAT PSW UNCHANGED BY RESET ABOVE
027026  001401                                  BEQ     .+4
027030  104000                                  HLT                     ;ERROR! RESET CLEARED MODE BITS IN PSW
027032  010037  177776                          MOV     R0,@#PSW        ;RESTORE PSW (FOR ERROR)
027036                                  ENDCP:
027036  000004                          RELE6:  SCOPE
027040  010702                                  MOV     PC,R2
027042  062702  000012                          ADD     #12,R2
027046  012707  034242                          MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
027052  000000                          REL66:  .WORD   0
                                        ;666666666666 LAST ADDRESS OF CODE TO BE RELOCATED 6666666666

                                        ;;*********************************************************************
                                        ;*TEST 65         TEST STACK LIMIT REGISTER
                                        ;;*********************************************************************
027054  112737  000065  001202          TST65:  MOVB    #65,@#STSTNM            ;LOAD TEST NUMBER
027062  012767  000001  152226                  MOV     #1,$TIMES       ;;DO 1 ITERATION
027070  000004                                  SCOPE

                                                .SBTTL  START OF SECTION 7
                                        ;777777777777 FIRST ADDRESS TO BE RELOCATED 777777777
027072  010700                          REL7:   MOV     PC,R0           ;GET PC
027074  005740                                  TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL7
027076  010037  001512                          MOV     R0,@#FRSTAD     ;SAVE
027102  010700                                  MOV     PC,R0           ;GET CURRENT PC
027104  162700  027104                          SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
```

```
027114  010737  001212                          MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
027120  062737  000030  001212                  ADD     #30,@#$LPERR    ;ADJUST
027126  013737  001212  001210                  MOV     @#$LPERR,@#$LPADR
027134  105737  001502                          TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
027140  001402                                  BEQ     .+6
027142  000167  001206                          JMP     RELE7
                                        ;THIS TEST SHIFTS A '1' BIT THROUGH ALL BIT POSITIONS
027146  012702  177774                          MOV     #STKLMT,R2                      ;GET ADDRESS OF STACK LIM REG
027152  005022                                  CLR     (R2)+                   ;CLEAR STACK LIMIT REG
027154  032712  000020                          BIT     #20,(R2)                        ;EXIT TEST IF 'T' BIT IS SET
027160  001116                                  BNE     101$
027162  052712  000340                          BIS     #340,(R2)               ;SET PRIORITY LEVEL 7 TO PREVENT
                                                                        ;ANY INTERRUPTS FROM OCCURRING
027166  012700  000400                          MOV     #400,R0                 ;SET CHECK DATA
027172  010042                          1$:     MOV     R0,-(R2)                ;MOVE TO STACK LIMIT REG
027174  022200                                  CMP     (R2)+,R0                ;AND CHECK RESULT
027176  001401                                  BEQ     2$
027200  104000                                  HLT                             ;ERROR! STACK LIMIT DID NOT
                                                                        ;LOAD CORRECTLY. CORRECT RESULT
                                                                        ;IS IN R0
027202  006300                          2$:     ASL     R0                      ;SHIFT '1' BIT LEFT
027204  103372                                  BCC     1$                      ;LOOP UNTIL 1 BIT SHIFTS OUT
027206  005042                                  CLR     -(R2)                   ;CLEAR STACK LIMIT REG

                                        ;THIS TEST CHECKS THAT A PROPER 'RED' ZONE VIOLATION OCCURS, NOTE THAT
                                        ;NO 'RED ZONE' VIOLATION WILL OCCUR IF IN USER/SUPER MODES.
                                        ;A RED ZONE VIOLATION PUSHES THE CURRENT PSW,PC ON A STACK AT 2 AND 0
                                        ;AND TAKES THE NEXT INSTRUCTION FROM THE PC IN LOCATION4. THE INST-
                                        ;RUCTION CAUSING THE RED ZONE VIOLATION IS 'ABORTED'.
027210  010746                                  MOV     PC,-(SP)                ;GET CURRENT PC
027212  062716  000060                          ADD     #4$-.,(SP)              ;FORM ADDRESS OF 4$ BELOW
027216  012637  000004                          MOV     (SP)+,@#ERRVEC          ;SET ERROR TRAP VECTOR TO 4$ BELOW
027222  013737  177776  000006                  MOV     @#PSW,@#ERRVEC+2        ;RETAIN CURRENT STATUS ON TRAP
027230  010712                                  MOV     PC,(R2)                 ;SET STACK LIMIT TO CURRENT PC
                                                                        ;+400
027232  011206                                  MOV     (R2),SP                 ;AND STACK PTR = STACK LIMIT REG
027234  010603                                  MOV     SP,R3                   ;SAVE STACK PTR
027236  016304  000336                          MOV     336(R3),R4              ;SAVE MEMORY LOC CONTENTS
                                                                        ;AT 'RED ZONE' BOUNDARY
027242  032737  140000  177776                  BIT     #UM,@#PSW               ;BRANCH IF IN KERNEL MODE
027250  001403                                  BEQ     20$
027252  010466  000336                          MOV     R4,336(SP)              ;SHOULD NOT CAUSE TRAP
027256  000432                                  BR      100$

027260  005066  000336                  20$:    CLR     336(SP)                 ;SHOULD CAUSE 'RED ZONE' TRAP
027264  012706  000700                  3$:     MOV     #SUPSTK,SP              ;RESTORE THE STACK
027270  104000                                  HLT                             ;ERROR! FAILED TO TRAP

027272  032737  140000  000002  4$:     BIT     #UM,@#2                 ;CHECK IF TRAPPED WHEN IN USER
                                                                        ;/SUPER MODES (2 CONTAINS OLD PSW)
027300  001013                                  BNE     99$                     ;GO TO ERROR CALL
027302  010600                                  MOV     SP,R0                   ;STACK PTR SHOULD = 0
027304  001011                                  BNE     99$                     ;GO TO ERROR CALL IF NOT 0
027306  026304  000336                          CMP     336(R3),R4              ;CHECK THAT INST WAS ABORTED
027312  001006                                  BNE     99$                     ;GO REPORT ERRPR
```

```
 027316  010705                             MOV     PC,R5                   ;GET CURRENT PC
 027320  062705  177744                     ADD     #3$-.,R5                ;FORM ADDRESS OF  3$ ABOVE
 027324  020516                             CMP     R5,(SP)                 ;CHECK THAT RETURN PC IS ON
                                                                            ;THE STACK (AT 0)
 027326  001406                             BEQ     100$                    ;EXIT TEST

                                    ;ERROR
 027330  005012                     99$:    CLR     (R2)                    ;CLEAR STACK LIMIT REG
 027332  010463  000336                     MOV     R4,336(R3)              ;RESTORE MEM LOCATION
 027336  012706  000700                     MOV     #SUPSTK,SP              ;SET STACK PTR
 027342  104000                             HLT                             ;ERROR!
 027344  010463  000336             100$:   MOV     R4,336(R3)              ;RESTORE MEM LOCATION
 027350  005022                             CLR     (R2)+                   ;CLEAR STACK LIM REG
 027352  012706  000700                     MOV     #SUPSTK,SP              ;SET STACK PTR
 027356  042712  000340                     BIC     #340,(R2)               ;SET PRIORITY LEVEL BACK TO 0
 027362  012737  053442  000004             MOV     #ERPRT,@#ERRVEC         ;RESTORE ERROR TRAP VECTOR
 027370  013737  177776  000006             MOV     @#PSW,@#ERRVEC+2
 027376  112737  000340  000006             MOVB    #PR7,@#ERRVEC+2
 027404  042737  000020  000006             BIC     #BIT4,@#ERRVEC+2
 027412  005037  177766                     CLR     @#CPUERR                        ;CLEAR ERROR REG
 027416                             101$:
                                    ;;*************************************************************
                                    ;*TEST 66       MEMORY MANAGEMENT REGISTER TESTS
                                    ;*      PDR TEST - THIS TEST WRITES 64. RANDOM #'S INTO EACH PDR REGISTER
                                    ;*      NOTE:  IF MEM MGMT IS ENABLED ONLY PDR/PAR PAIRS 3-5 ARE TESTED.
                                    ;;*************************************************************
 027416  112737  000066  001202     TST66:  MOVB    #66,@#$TSTNM            ;LOAD TEST NUMBER
 027424  000004                             SCOPE

 027426  012702  027652             KTPDR:  MOV     #PDRTBL,R2      ;SET TABLE ADDRESS OF PDR'S
 027432  012705  100360                     MOV     #100360,R5      ;SET BIT MASK (11/45)
 027436  012200                     1$:     MOV     (R2)+,R0        ;GET PDR ADDRESS
 027440  001435                             BEQ     100$            ;EXIT ON '0' TERMINATOR
 027442  012716  000010             2$:     MOV     #8.,(SP)        ;SET LOOP COUNT (FOR 8 REGS)
 027446  105737  001503                     TSTB    @#MMON          ;BRANCH IF MEM MGMT DISABLED
 027452  001404                             BEQ     3$
 027454  062700  000006                     ADD     #6,R0           ;SET R0 TO PDR3
 027460  012716  000003                     MOV     #3,(SP)         ;AND LIMIT TO TEST 3 PDRS
 027464  012703  000040             3$:     MOV     #32.,R3         ;SET DATA COUNT
 027470  005004                             CLR     R4              ;INITIALIZE DATA TO BE WRITTEN
 027472  040504                     4$:     BIC     R5,R4           ;CLEAR NON-SETTABLE BITS
 027474  010410                             MOV     R4,(R0)         ;WRITE INTO PDR
 027476  021004                             CMP     (R0),R4         ;AND CHECK DATA READ BACK
 027500  001013                             BNE     99$             ;GO TO ERROR CALL
 027502  005104                             COM     R4              ;COMPLEMENT DATA
 027504  040504                             BIC     R5,R4           ;CLEAR NON-SETTABLE BITS
 027506  010410                             MOV     R4,(R0)         ;WRITE COMPLEMENT DATA INTO PDR
 027510  021004                             CMP     (R0),R4         ;AND CHECK
 027512  001006                             BNE     99$             ;GO TO ERROR CALL
 027514  060104                             ADD     R1,R4           ;STEP DATA
 027516  077313                             SOB     R3,4$
 027520  005020                     5$:     CLR     (R0)+           ;STEP TO NEXT REGISTER
 027522  005316                             DEC     (SP)            ;DECREMENT REGISTER COUNT
 027524  001357                             BNE     3$
 027526  000743                             BR      1$              ;GET NEXT SET OF 8 REGISTERS
```

```
027530  104000                    99$:    HLT                             ;ERROR! INCORRECT DATA READ
                                                                          ;BACK FROM PDR. ADDRESS OF
                                                                          ;PDR IS IN RO, DATA IS IN R4
027532  000772                            BR      5$                      ;STEP TO NEXT REGISTER
027534                            100$:
                                  ;;*************************************************************
                                  ;*TEST 67        PAR TEST
                                  ;*      PAR TEST - THIS TEST WRITES 64. COMPLEMENTING RANDOM #'S INTO EACH PAR.
                                  ;;*************************************************************
027534  112737  000067  001202    TST67:  MOVB    #67,@#$TSTNM            ;LOAD TEST NUMBER
027542  000004                            SCOPE
027544  012702  027670            KTPAR:  MOV     #PARTBL,R2              ;GET TABLE ADDRESS OF PAR'S
027550  005005                            CLR     R5
027552  012200                    1$:     MOV     (R2)+,RO                ;GET PAR ADDRESS
027554  001435                            BEQ     100$                    ;EXIT ON 'O' TERMINATOR
027556  012716  000010            2$:     MOV     #8.,(SP)                ;SET LOOP COUNT (FOR 8 REGS.)
027562  105737  001503                    TSTB    @#MMON                  ;BRANCH IF MEM MGMT DISABLED
027566  001404                            BEQ     3$
027570  062700  000006                    ADD     #6,RO                   ;SET RO TO PAR3
027574  012716  000003                    MOV     #3,(SP)                 ;AND LIMIT TEST TO 3 PARS
027600  012703  000040            3$:     MOV     #32.,R3                 ;SET DATA COUNT
027604  005004                            CLR     R4                      ;INITIALIZE DATA
027606  040504                    4$:     BIC     R5,R4                   ;CLEAR NON-SETTABLE BITS
027610  010410                            MOV     R4,(RO)                 ;WRITE INTO PAR
027612  021004                            CMP     (RO),R4                 ;AND CHECK
027614  001013                            BNE     99$                     ;TAKE ERROR EXIT
027616  005104                            COM     R4                      ;COMPLEMENT DATA
027620  040504                            BIC     R5,R4                   ;CLEAR NON-SETTABLE BITS
027622  010410                            MOV     R4,(RO)                 ;WRITE COMPLEMENT DATA
027624  021004                            CMP     (RO),R4                 ;AND CHECK
027626  001006                            BNE     99$                     ;TAKE ERROR EXIT
027630  060104                            ADD     R1,R4                   ;STEP DATA
027632  077313                            SOB     R3,4$                   ;LOOP UNTIL FINISHED

027634  005020                    5$:     CLR     (RO)+
027636  005316                            DEC     (SP)                    ;DECREMENT REGISTER COUNT
027640  001357                            BNE     3$                      ;BRANCH IF 8 REGS NOT DONE
027642  000743                            BR      1$

027644  104000                    99$:    HLT                             ;ERROR! INCORRECT DATA READ BACK
                                                                          ;FROM PAR. ADDRESS OF PAR IS IN
                                                                          ;RO, DATA IS IN R4
027646  000772                            BR      5$                      ;DO NEXT REGISTER
027650                            100$:
027650  000416                            BR      TST70                   ;;GO TO NEXT TEST
                                  ;TABLES FOR PDR & PAR TESTS ABOVE
027652  172300                    PDRTBL: .WORD   KIPDRO
027654  177600                            .WORD   UIPDRO
027656  172200                            .WORD   SIPDRO                  ;CHANGED TO 'O' IF 11/40
027660  172320                            .WORD   KDPDRO
027662  177620                            .WORD   UDPDRO
027664  172220                            .WORD   SDPDRO
027666  000000                            .WORD   O                       ;TERMINATOR

027670  172340                    PARTBL: .WORD   KIPARO
```

```
027674  172240                          .WORD   SIPAR0                  ;CHANGED TO '0' IF 11/40
027676  172360                          .WORD   KDPAR0
027700  177660                          .WORD   UDPAR0
027702  172260                          .WORD   SDPAR0
027704  000000                          .WORD   0                       ;TERMINATOR

                                ;;*************************************************************
                                ;*TEST 70         CHECK KT ABORT LOGIC
                                ;*        THIS TEST CHECKS KT ABORT LOGIC. TEST CREATES AN ABORT CONDITION
                                ;*        AND INSURES THAT ABORT IS TAKEN PROPERLY. NOTE: TEST IS EXECUTED ONLY
                                ;*        IF TEST IS ENTERED WITH MEM MGMT ENABLED.
                                ;;*************************************************************
027706  112737  000070  001202  TST70:  MOVB    #70,@#$TSTNM            ;LOAD TEST NUMBER
027714  000004                          SCOPE
027716  105737  001503          KTABT:  TSTB    @#MMON                  ;BRANCH IF MEM MGMT DISABLED
027722  001515                          BEQ     KTEX
027724  005037  172350                  CLR     @#KIPAR4                ;SET UP MEM MGMT REGISTERS
027730  005037  172310                  CLR     @#KIPDR4                ;TO ABORT IF A MEMORY
027734  005037  177650                  CLR     @#UIPAR4                ;REFERENCE IS MADE TO
027740  005037  177610                  CLR     @#UIPDR4                ;ADDRESSES (VIRTUAL) BETWEEN
027744  005037  172250                  CLR     @#SIPAR4
027750  005037  172210                  CLR     @#SIPDR4
027754  013746  000250          1$:     MOV     @#MMVEC,-(SP)           ;SAVE MEM MGMT VECTOR
027760  013746  000252                  MOV     @#MMVEC+2,-(SP)         ;AND PRIORITY
027764  010746                          MOV     PC,-(SP)                ;SET MEM MGMT
027766  062716  000040                  ADD     #4$-.,(SP)              ;VECTOR TO 4$ BELOW
027772  012637  000250                  MOV     (SP)+,@#MMVEC
027776  013737  177776  000252          MOV     @#PSW,@#MMVEC+2
030004  005000                          CLR     R0                      ;CLEAR ABORT INDICATOR
030006  010702                          MOV     PC,R2                   ;SET R2 AND R3 NOTE:
030010  012703  100000                  MOV     #100000,R3              ;THE REF VIA R3 CAUSES THE
030014  014223                  2$:     MOV     -(R2),(R3)+             ;ABORT
030016  005700                  3$:     TST     R0                      ;BRANCH IF THE ABORT OCCURRED
030020  001001                          BNE     .+4
030022  104000                          HLT                             ;REPORT ERROR
030024  000445                          BR      100$
                                ;ABORT HERE
030026  013700  177776          4$:     MOV     @#PSW,R0                ;SR0 SHOULD CONTAIN
030032  000300                          SWAB    R0                      ;CAUSE FOR ABORT AND
030034  006200                          ASR     R0                      ;ALSO WHICH SEGMENT
030036  042700  177637                  BIC     #177637,R0              ;WAS IN USE WHEN ABORT
030042  062700  100011                  ADD     #100011,R0              ;OCCURRED.
030046  020037  177572                  CMP     R0,@#SR0
030052  001025                          BNE     99$
030054  012700  030014                  MOV     #2$,R0                  ;GET ADDRESS OF INST THAT ABORTED
030060  020037  177576                  CMP     R0,@#SR2                ;THAT ABORTED
030064  001020                          BNE     99$
030066  012700  000362                  MOV     #362,R0
030072  120037  177574                  CMPB    R0,@#SR1                ;SR1 (11/45) CONTAINS REGISTER
030076  001013                          BNE     99$                     ;MODIFICATIONS MADE
030100  012700  000023                  MOV     #23,R0
030104  120037  177575                  CMPB    R0,@#SR1+1
030110  001006                          BNE     99$
030112  012700  030014                  MOV     #2$,R0
030116  005720                  5$:     TST     (R0)+                   ;R0=ADDRESS OF INST FOLLOWING ABORT
```

```
030122  001001                           BNE     99$
030124  000002                           RTI                         ;RETURN
                                 ;ENTER HERE ON ERROR
030126  104000                   99$:    HLT                         ;REPORT ERROR
030130  010716                           MOV     PC,(SP)
030132  062716  177664                   ADD     #3$-.,(SP)
030136  000002                           RTI                         ;RETURN
030140  012637  000252           100$:   MOV     (SP)+,@#MMVEC+2     ;RESTORE ABORT VECTOR
030144  012637  000250                   MOV     (SP)+,@#MMVEC       ;& PRIORITY.
030150  012737  000001  177572           MOV     #1,@#SR0            ;CLEAR ERROR CONDITIONS
030156                           KTEX:
                                 ;;*************************************************************
                                 ;*TEST 71        MAPPING REGISTER TESTS
                                 ;*       THIS TEST LOADS RANDOM #'S INTO EACH MAPPING REGISTER
                                 ;;*************************************************************
030156  112737  000071  001202   TST71:  MOVB    #71,@#$TSTNM                ;LOAD TEST NUMBER
030164  000004                           SCOPE
030166  032737  000040  172516           BIT     #BIT5,@#MMR3        ;IS MAP ON?
030174  001053                           BNE     MAPTWO              ;BRANCH IF YES
030176  012700  170200           MAPTST: MOV     #MAPLO,R0           ;SET ADRS OF FIRST MAP REGISTER
030202  012706  000700                   MOV     #SUPSTK,SP          ;SETUP THE SP
030206  012716  000001                   MOV     #1,(SP)             ;SET BIT MASK FOR MAPLO <15-01>
030212  012702  177700                   MOV     #177700,R2          ;AND ALSO FOR MAPHO <21-16>
030216  012703  000040           1$:     MOV     #32.,R3             ;SET DATA COUNT
030222  005005                           CLR     R5                  ;SET INITIAL DATA
030224  010504                   2$:     MOV     R5,R4               ;GET DATA
030226  041604                           BIC     (SP),R4             ;CLEAR UNUSED BITS
030230  010410                           MOV     R4,(R0)             ;LOAD DATA INTO MAPLO <15-01>
030232  021004                           CMP     (R0),R4             ;CHECK DATA
030234  001032                           BNE     99$                 ;BRANCH IF INCORRECT
030236  005105                           COM     R5                  ;COMPLEMENT TEST DATA
030240  010504                           MOV     R5,R4               ;GET TEST DATA
030242  041604                           BIC     (SP),R4             ;CLEAR UNUSED BITS
030244  010410                           MOV     R4,(R0)             ;LOAD COMPLEMENT DATA
030246  021004                           CMP     (R0),R4             ;AND CHECK
030250  001024                           BNE     99$
030252  005720                           TST     (R0)+               ;STEP TO NEXT REGISTER
030254  010504                           MOV     R5,R4               ;GET COMPLEMENT TEST DATA
030256  040204                           BIC     R2,R4               ;CLEAR UNUSED BITS
030260  010410                           MOV     R4,(R0)             ;LOAD TEST DATA INTO MAPHO <21-16>
030262  021004                           CMP     (R0),R4             ;AND CHECK
030264  001016                           BNE     99$
030266  005105                           COM     R5                  ;COMPLEMENT TEST DATA
030270  010504                           MOV     R5,R4               ;GET TEST DATA
030272  040204                           BIC     R2,R4               ;CLEAR UNUSED BITS
030274  010410                           MOV     R4,(R0)             ;LOAD TEST DATA
030276  021004                           CMP     (R0),R4             ;AND CHECK
030300  001010                           BNE     99$
030302  060705                           ADD     PC,R5               ;FORM NEXT TEST DATA
030304  005740                           TST     -(R0)               ;RESET PTR TO REGISTER <15-01>
030306  077332                           SOB     R3,2$               ;AND TEST UNTIL ALL #'S USED
030310  022020                           CMP     (R0)+,(R0)+         ;STEP TO NEXT REGISTER PAIR
030312  022700  170400                   CMP     #MAPLO+128.,R0      ;BRANCH IF NOT LAST PAIR
030316  001337                           BNE     1$
030320  000401                           BR      MAPTWO
```

```
030322  104000                         99$:    HLT                             ;ERROR! INCORRECT DATA READ BACK
                                                                               ;FROM MAP REG. ADRS OF REGISTER 15
                                                                               ;IN R0, GOOD DATA IS IN R4
030324  005737  001504                 MAPTWO: TST     @#QV                    ;QV OR AUTO-ACCEPT?
030330  001411                                 BEQ     RELE7                   ;BRANCH IF NO
030332  012737  020000  170204                 MOV     #20000,@#MAPL1          ;SET MAP 1 INCASE ACT11
030340  005037  170206                         CLR     @#MAPH1
030344  005037  170200                         CLR     @#MAPL0
030350  005037  170202                         CLR     @#MAPH0
030354  000004                         RELE7:  SCOPE
030356  010702                                 MOV     PC,R2
030360  062702  000012                         ADD     #12,R2
030364  012707  034242                         MOV     #RELOC,PC               ;GO RELOCATE PROGRAM CODE
030370  000000                         REL77:  .WORD   0
                                       ;7777777777777 LAST ADDRESS OF CODE TO BE RELOCATED 77777777777


                                       ;;**************************************************************
                                       ;*TEST 72        FLOATING POINT TEST 1
                                       ;*
                                       ;*      THIS TEST TAKES TWO RANDOM NUMBERS (A AND B) AND
                                       ;*      COMPARES THE RESULTS OF TWO EQUAL CALCULATIONS.
                                       ;*      EACH SECTION EVALUATES A DIFFERENT EQUATION AS DESCRIBED BELOW:
                                       ;*      SECT1   (A+B)**2=A**2+2*A*B+B**2
                                       ;*      SECT2   (A+B)*(A-B)=A**2-B**2
                                       ;*      SECT3   A/B*B=A
                                       ;;**************************************************************
030372  112737  000072  001202         TST72:  MOVB    #72,@#$TSTNM                    ;LOAD TEST NUMBER
030400  000004                                 SCOPE
030402  012737  000001  001316                 MOV     #1,@#$TIMES             ;SET ITTERATIONS TO 1

                                               .SBTTL  START OF SECTION 8
                                       ;888888888888 FIRST ADDRESS TO BE RELOCATED 888888888
030410  010700                         REL8:   MOV     PC,R0                   ;GET PC
030412  005740                                 TST     -(R0)                   ;R0 CONTAINS THE ADDRESS OF REL8
030414  010037  001512                         MOV     R0,@#FRSTAD             ;SAVE
030420  010700                                 MOV     PC,R0                   ;GET CURRENT PC
030422  162700  030422                         SUB     #.,R0                   ;SUBTRACT RELOCATION FACTOR
030426  010037  001506                         MOV     R0,@#FACTOR             ;SAVE RELOCATION FACTOR
030432  010737  001212                         MOV     PC,@#$LPERR             ;SET LOOP ADDRESS
030436  062737  000030  001212                 ADD     #30,@#$LPERR            ;ADJUST
030444  013737  001212  001210                 MOV     @#$LPERR,@#$LPADR
030452  105737  001502                         TSTB    @#NEXEC                 ;BR IF TEST CODE TO BE EXECUTED
030456  001402                                 BEQ     .+6
030460  000167  002414                         JMP     RELE8
030464  032737  020000  001470                 BIT     #FPOPT,@#OPT.CP         ;FLOATING POINT AVAILABLE?
030472  001002                                 BNE     100$                    ;BRANCH IF YES
030474  000167  002416                         JMP     REL88+2
030500  004737  050710                 100$:   JSR     PC,@#FLTSGL             ;GET RANDOM OPERANDS
030504  170127  000000                         LDFPS   #0                      ;INIT FPS
030510  172537  001276                         LDF     @#$TMP0,AC1             ;LOAD A OPERAND
030514  172437  001302                         LDF     @#$TMP2,AC0             ;LOAD B OPERAND
030520  013737  001256  001404                 MOV     @#$REG0,@#$AC1          ;SETUP EXTENDED
030526  013737  001260  001402                 MOV     @#$REG1,@#$AC0          ;EXPONENTS
030534  004767  002202                         JSR     PC,FLTADD               ;PERFORM THE ADD
030540  174100                                 STF     AC1,AC0                 ;SETUP AC0 TO
```

```
030550  004767  002116                  JSR     PC,FLTMPY               ;DO THE MULTIPLY
030554  174137  001306                  STF     AC1,@#STMP4             ;SAVE RESULT
030560  013737  001404  001262          MOV     @#SAC1,@#SREG2          ;AND SOFTWARE EXP

                                ;NOW DO THE RIGHT HAND SIDE OF THE EQUATION
                                ;DO THE A*A FIRST
030566  013737  001256  001402          MOV     @#SREG0,@#SAC0          ;GET EXT EXPONENT
030574  172437  001276                  LDF     @#STMP0,AC0             ;LOAD OPERAND A
030600  013737  001402  001404          MOV     @#SAC0,@#SAC1           ;SET OPERAND B EXT EXPONENT
030606  172500                          LDF     AC0,AC1                 ;LOAD B OPERAND
030610  004767  002056                  JSR     PC,FLTMPY               ;EXECUTE THE MULTIPLY
030614  174102                          STF     AC1,AC2                 ;SAVE RESULT
030616  013737  001404  001406          MOV     @#SAC1,@#SAC2

                                ;NOW DO THE B*B
030624  172437  001302                  LDF     @#STMP2,AC0             ;LOAD B OPERAND
030630  172500                          LDF     AC0,AC1
030632  013737  001260  001402          MOV     @#SREG1,@#SAC0          ;AND EXT EXPONENT
030640  013737  001402  001404          MOV     @#SAC0,@#SAC1
030646  004767  002020                  JSR     PC,FLTMPY               ;DO THE MULTIPLY
030652  174103                          STF     AC1,AC3                 ;SAVE THE RESULT
030654  013737  001404  001410          MOV     @#SAC1,@#SAC3

                                ;NOW DO THE 2*B*A
030662  012701  001302                  MOV     #STMP2,R1
030666  172411                          LDF     (R1),AC0                ;LOAD THE B OPERAND
030670  172541                          LDF     -(R1),AC1               ;LOAD THE A OPERAND
030672  013737  001260  001402          MOV     @#SREG1,@#SAC0          ;AND THE EXT EXPONENTS
030700  013737  001256  001404          MOV     @#SREG0,@#SAC1
030706  004767  001760                  JSR     PC,FLTMPY               ;DO THE MULTIPLY
030712  172427  040000                  LDF     #↑040000,AC0            ;SETUP TO MULTIPLY BY TWO
030716  012737  000002  001402          MOV     #2,@#SAC0
030724  004767  001742                  JSR     PC,FLTMPY               ;DO THE MULTIPLY

                                ;NOW SUM THE RESULTS
030730  013737  001410  001402          MOV     @#SAC3,@#SAC0
030736  172403                          LDF     AC3,AC0                 ;GET RESULT OF B*B
030740  004767  001776                  JSR     PC,FLTADD               ;ADD THE RESULT
030744  172402                          LDF     AC2,AC0                 ;GET RESULT OF A*A
030746  013737  001406  001402          MOV     @#SAC2,@#SAC0
030754  004767  001762                  JSR     PC,FLTADD               ;ADD THIS RESULT
030760  174137  001312                  STF     AC1,@#STMP6             ;SAVE FINAL RESULT
030764  013737  001404  001264          MOV     @#SAC1,@#SREG3

                                ;NOW CHECK BOTH SIDES OF THE EQUATION
                                ;CALCULATE THE NUMBER OF CORRECT BITS
                                ;PUT LARGEST EXPONENT OF A**2 OR B**2 IN SAC2
030772  023737  001406  001410          CMP     @#SAC2,@#SAC3
031000  002003                          BGE     1$                      ;BRANCH IF SAC2 ALREADY HAS LARGEST
031002  013737  001410  001406          MOV     @#SAC3,@#SAC2           ;SAC3 WAS LARGER
031010  163737  001404  001406  1$:     SUB     @#SAC1,@#SAC2           ;NOW CALCULATE NUMBER
031016  162737  000024  001406          SUB     #20.,@#SAC2             ;OF CORRECT BITS WITHIN 2
031024  005437  001406                  NEG     @#SAC2                  ;MAKE RESULT POSITIVE
031030  172437  001306                  LDF     @#STMP4,AC0             ;LOAD RESULT OF LEFT HAND SIDE
031034  013737  001262  001402          MOV     @#SREG2,@#SAC0          ;AND EXTENDED EXPONENT
```

```
                                                                        ;ACTUAL EXP'S ARE EQUAL TO 200
031054  100002                          BPL     3$                      ;ENSURE RESULT IS POSITIVE
031056  005437  001404                  NEG     @#$AC1
031062  023737  001406  001404  3$:     CMP     @#$AC2,@#$AC1           ;ANSWERS WITHIN ALLOWABLE NUMBER?
031070  003401                          BLE     SECT2                   ;BRANCH IF YES
031072  104014                  4$:     ERROR   14                      ;RESULTS ARE WRONG
                                ;;*****************************************************************
031074  170127  000000          SECT2:  LDFPS   #0
                                ;DO A+B
031100  172537  001276                  LDF     @#STMP0,AC1             ;LOAD A OPERAND
031104  172437  001302                  LDF     @#STMP2,AC0             ;LOAD B OPERAND
031110  013737  001256  001404          MOV     @#SREG0,@#$AC1
031116  013737  001260  001402          MOV     @#SREG1,@#$AC0
031124  004767  001612                  JSR     PC,FLTADD               ;ADD THEM
031130  174102                          STF     AC1,AC2                 ;SAVE IN AC2
031132  013737  001404  001406          MOV     @#$AC1,@#$AC2           ;AND EXT EXPONENT
                                ;NOW DO THE A-B
031140  172537  001276                  LDF     @#STMP0,AC1             ;LOAD OPERAND A
031144  013737  001256  001404          MOV     @#SREG0,@#$AC1          ;AND EXT EXPONENT
031152  172437  001302                  LDF     @#STMP2,AC0             ;LOAD OPERAND B
031156  013737  001260  001402          MOV     @#SREG1,@#$AC0
031164  004767  001546                  JSR     PC,FLTSUB               ;SUBTRACT THEM
                                ;NOW DO (A+B)*(A-B)
031170  172402                          LDF     AC2,AC0                 ;GET RESULT OF (A+B)
031172  013737  001406  001402          MOV     @#$AC2,@#$AC0
031200  004767  001466                  JSR     PC,FLTMPY               ;FORM THE PRODUCT
031204  174137  001306                  STF     AC1,@#STMP4             ;SAVE RESULT
031210  013737  001404  001262          MOV     @#$AC1,@#SREG2          ;AND EXT EXPONENT
                                ;NOW DO THE B*B
031216  172437  001302                  LDF     @#STMP2,AC0             ;LOAD OPERAND B
031222  013737  001260  001402          MOV     @#SREG1,@#$AC0
031230  172500                          LDF     AC0,AC1                 ;B OPERAND IS IN AC0
031232  013737  001402  001404          MOV     @#$AC0,@#$AC1           ;AND EXT EXPONENT
031240  004767  001426                  JSR     PC,FLTMPY               ;
031244  174102                          STF     AC1,AC2                 ;SAVE RESULT IN AC2
031246  013737  001404  001406          MOV     @#$AC1,@#$AC2
                                ;NOW DO THE A*A
031254  172437  001276                  LDF     @#STMP0,AC0             ;LOAD OPERAND A
031260  013737  001256  001402          MOV     @#SREG0,@#$AC0
031266  172500                          LDF     AC0,AC1
031270  013737  001402  001404          MOV     @#$AC0,@#$AC1
031276  004767  001370                  JSR     PC,FLTMPY               ;EXECUTE THE MULTIPLY
031302  013737  001404  001410          MOV     @#$AC1,@#$AC3           ;SAVE EXT EXPO OF A*A
                                ;NOW DO A**2-B**2
031310  172402                          LDF     AC2,AC0                 ;GET B*B
031312  013737  001406  001402          MOV     @#$AC2,@#$AC0           ;A*A IN AC1
031320  004767  001412                  JSR     PC,FLTSUB
031324  174137  001312                  STF     AC1,@#STMP6             ;SAVE IN MEMORY
031330  013737  001404  001264          MOV     @#$AC1,@#SREG3
                                ;NOW COMPUTE THE RESULTS
                                ;CALCULATE THE NUMBER OF CORRECT BITS
031336  023737  001406  001410          CMP     @#$AC2,@#$AC3           ;DETERMINE WHICH EXP IS LARGER
031344  002003                          BGE     2$                      ;BRANCH IF AC2 LARGER
031346  013737  001410  001406          MOV     @#$AC3,@#$AC2           ;PUT LARGEST IN AC2
031354  163737  001404  001406  2$:     SUB     @#$AC1,@#$AC2
```

```
031370  005437  001406                   NEG     @#$AC2
031374  172437  001306                   LDF     @#$TMP4,AC0             ;GET LEFT HAND SIDE
031400  013737  001262  001402           MOV     @#$REG2,@#$AC0
031406  004767  001324                   JSR     PC,FLTSUB               ;SUBTRACT TO SEE HOW CLOSE THEY ARE
031412  163737  001264  001404           SUB     @#$REG3,@#$AC1          ;SUB EXT EXPONENTS
                                                                         ;ACTUAL EXPONENTS ARE EQUAL
031420  100002                           BPL     1$                      ;MAKE SURE RESULT IS POSITIVE
031422  005437  001404                   NEG     @#$AC1
031426  023737  001406  001404   1$:     CMP     @#$AC2,@#$AC1           ;RESULTS WITHIN RANGE ALLOWED?
031434  003401                           BLE     SECT3                   ;BRANCH IF YES
031436  104014                           ERROR   14                      ;RESULTS WRONG

                                 ;;*************************************************************
031440  172537  001276           SECT3:  LDF     @#$TMP0,AC1             ;LOAD OPERAND A
031444  172437  001302                   LDF     @#$TMP2,AC0             ;AND OPERAND B
031450  013737  001256  001404           MOV     @#$REG0,@#$AC1
031456  013737  001260  001402           MOV     @#$REG1,@#$AC0
031464  004767  001224                   JSR     PC,FLTDIV               ;GO DIVIDE THEM
031470  004767  001176                   JSR     PC,FLTMPY               ;MULTIPLY RESULT BY B
031474  174137  001306                   STF     AC1,@#$TMP4             ;SAVE RESULT
031500  013737  001404  001262           MOV     @#$AC1,@#$REG2
031506  172437  001276                   LDF     @#$TMP0,AC0             ;LOAD OPERAND A
031512  174037  001312                   STF     AC0,@#$TMP6             ;SAVE INCASE TYPE OUT
031516  013737  001256  001402           MOV     @#$REG0,@#$AC0
031524  013737  001256  001264           MOV     @#$REG0,@#$REG3
031532  004767  001200                   JSR     PC,FLTSUB               ;SUBTRACT RIGHT AND LEFT HAND SIDES
031536  163737  001256  001404           SUB     @#$REG0,@#$AC1          ;SEE IF RESULT OK
031544  100002                           BPL     1$                      ;ENSURE DIFFERANCE IS POSITIVE
031546  005437  001404                   NEG     @#$AC1
031552  022737  000027  001404   1$:     CMP     #23.,@#$AC1             ;RESULTS WITHIN 2 BITS?
031560  003001                           BGT     2$                      ;BRANCH IF NO
031562  000401                           BR      TST73                   ;;GO TO NEXT TEST
031564  104014                   2$:     ERROR   14                      ;RESULTS WRONG

                                 ;;*************************************************************
                                 ;*TEST 73        FLOATING POINT TEST 2
                                 ;*
                                 ;*       THIS TEST TAKES TWO RANDOM NUMBERS (A AND B) AND
                                 ;*       COMPARES THE RESULTS OF TWO EQUAL CALCULATIONS.
                                 ;*       EACH SECTION EVALUATES A DIFFERENT EQUATION AS DESCRIBED BELOW:
                                 ;*       SECT1   (A+B)**2=A**2+2*A*B+B**2
                                 ;*       SECT2   (A+B)*(A-B)=A**2-B**2
                                 ;*       SECT3   A/B*B=A
                                 ;;*************************************************************
031566  112737  000073  001202   TST73:  MOVB    #73,@#$TSTNM                    ;LOAD TEST NUMBER
031574  000004                           SCOPE
031576  012737  000001  001316           MOV     #1,@#$TIMES
031604  004737  050702           100$:   JSR     PC,@#FLTDBL             ;GET RANDOM OPERANDS
031610  170127  000200                   LDFPS   #200                    ;INIT FPS
031614  172537  001276                   LDF     @#$TMP0,AC1             ;LOAD A OPERAND
031620  172437  001306                   LDF     @#$TMP4,AC0             ;LOAD B OPERAND
031624  013737  001256  001404           MOV     @#$REG0,@#$AC1          ;SETUP EXTENDED
031632  013737  001260  001402           MOV     @#$REG1,@#$AC0          ;EXPONENTS
031640  004767  001076                   JSR     PC,FLTADD               ;PERFORM THE ADD
031644  174100                           STF     AC1,AC0                 ;SETUP AC0 TO
```

```
031654  004767  001012                  JSR     PC,FLTMPY               ;DO THE MULTIPLY
031660  174137  001422                  STF     AC1,@#FLTMP0            ;SAVE RESULT
031664  013737  001404  001262          MOV     @#$AC1,@#$REG2          ;AND SOFTWARE EXP

                                ;NOW DO THE RIGHT HAND SIDE OF THE EQUATION
                                ;DO THE A*A FIRST
031672  013737  001256  001402          MOV     @#$REG0,@#$AC0          ;GET EXT EXPONENT
031700  172437  001276                  LDF     @#$TMP0,AC0             ;LOAD OPERAND A
031704  013737  001402  001404          MOV     @#$AC0,@#$AC1           ;SET OPERAND B EXT EXPONENT
031712  172500                          LDF     AC0,AC1                 ;LOAD B OPERAND
031714  004767  000752                  JSR     PC,FLTMPY               ;EXECUTE THE MULTIPLY
031720  174102                          STF     AC1,AC2                 ;SAVE RESULT
031722  013737  001404  001406          MOV     @#$AC1,@#$AC2
                                ;
                                ;NOW DO THE B*B
031730  172437  001306                  LDF     @#$TMP4,AC0             ;LOAD B OPERAND
031734  172500                          LDF     AC0,AC1
031736  013737  001260  001402          MOV     @#$REG1,@#$AC0          ;AND EXT EXPONENT
031744  013737  001402  001404          MOV     @#$AC0,@#$AC1
031752  004767  000714                  JSR     PC,FLTMPY               ;DO THE MULTIPLY
031756  174103                          STF     AC1,AC3                 ;SAVE THE RESULT
031760  013737  001404  001410          MOV     @#$AC1,@#$AC3
                                ;
                                ;NOW DO THE 2*B*A
031766  012701  001306                  MOV     #$TMP4,R1
031772  172411                          LDF     (R1),AC0                ;LOAD THE B OPERAND
031774  172541                          LDF     -(R1),AC1               ;LOAD THE A OPERAND
031776  013737  001260  001402          MOV     @#$REG1,@#$AC0          ;AND THE EXT EXPONENTS
032004  013737  001256  001404          MOV     @#$REG0,@#$AC1
032012  004767  000654                  JSR     PC,FLTMPY               ;DO THE MULTIPLY
032016  172427  040000                  LDF     #↑O40000,AC0            ;SETUP TO MULTIPLY BY TWO
032022  012737  000002  001402          MOV     #2,@#$AC0
032030  004767  000636                  JSR     PC,FLTMPY               ;DO THE MULTIPLY
                                ;
                                ;NOW SUM THE RESULTS
032034  013737  001410  001402          MOV     @#$AC3,@#$AC0
032042  172403                          LDF     AC3,AC0                 ;GET RESULT OF B*B
032044  004767  000672                  JSR     PC,FLTADD               ;ADD THE RESULT
032050  172402                          LDF     AC2,AC0                 ;GET RESULT OF A*A
032052  013737  001406  001402          MOV     @#$AC2,@#$AC0
032060  004767  000656                  JSR     PC,FLTADD               ;ADD THIS RESULT
032064  174137  001432                  STF     AC1,@#FLTMP1            ;SAVE FINAL RESULT
032070  013737  001404  001264          MOV     @#$AC1,@#$REG3
                                ;
                                ;NOW CHECK BOTH SIDES OF THE EQUATION
                                ;CALCULATE THE NUMBER OF CORRECT BITS
                                ;PUT LARGEST EXPONENT OF A**2 OR B**2 IN $AC2
032076  023737  001406  001410          CMP     @#$AC2,@#$AC3
032104  002003                          BGE     1$                      ;BRANCH IF $AC2 ALREADY HAS LARGEST
032106  013737  001410  001406          MOV     @#$AC3,@#$AC2           ;$AC3 WAS LARGER
032114  163737  001404  001406  1$:     SUB     @#$AC1,@#$AC2           ;NOW CALCULATE NUMBER
032122  162737  000065  001406          SUB     #53.,@#$AC2             ;OF CORRECT BITS WITHIN 2
032130  005437  001406                  NEG     @#$AC2                  ;MAKE RESULT POSITIVE
032134  172437  001422                  LDF     @#FLTMP0,AC0            ;LOAD RESULT OF LEFT HAND SIDE
032140  013737  001262  001402          MOV     @#$REG2,@#$AC0          ;AND EXTENDED EXPONENT
```

```
032152  163737  001264  001404           SUB     @#$REG3,@#$AC1      ;GET DIFFERENCE IN EXT EXPONENTS
                                                                     ;ACTUAL EXP'S ARE EQUAL TO 200
032160  100002                           BPL     3$                  ;ENSURE RESULT IS POSITIVE
032162  005437  001404                   NEG     @#$AC1
032166  023737  001406  001404  3$:      CMP     @#$AC2,@#$AC1       ;ANSWERS WITHIN ALLOWABLE NUMBER?
032174  003401                           BLE     SECT2D              ;BRANCH IF YES
032176  104016                  4$:      ERROR   16                  ;RESULTS ARE WRONG
                                ;;*************************************************************
032200  170127  000200          SECT2D:  LDFPS   #200
                                ;DO A+B
032204  172537  001276                   LDF     @#$TMP0,AC1         ;LOAD A OPERAND
032210  172437  001306                   LDF     @#$TMP4,AC0         ;LOAD B OPERAND
032214  013737  001256  001404           MOV     @#$REG0,@#$AC1
032222  013737  001260  001402           MOV     @#$REG1,@#$AC0
032230  004767  000506                   JSR     PC,FLTADD           ;ADD THEM
032234  174102                           STF     AC1,AC2             ;SAVE IN AC2
032236  013737  001404  001406           MOV     @#$AC1,@#$AC2       ;AND EXT EXPONENT
                                ;NOW DO  THE A-B
032244  172537  001276                   LDF     @#$TMP0,AC1         ;LOAD OPERAND A
032250  013737  001256  001404           MOV     @#$REG0,@#$AC1      ;AND EXT EXPONENT
032256  172437  001306                   LDF     @#$TMP4,AC0         ;LOAD OPERAND B
032262  013737  001260  001402           MOV     @#$REG1,@#$AC0
032270  004767  000442                   JSR     PC,FLTSUB           ;SUBTRACT THEM
                                ;NOW DO  (A+B)*(A-B)
032274  172402                           LDF     AC2,AC0             ;GET RESULT OF (A+B)
032276  013737  001406  001402           MOV     @#$AC2,@#$AC0
032304  004767  000362                   JSR     PC,FLTMPY           ;FORM THE PRODUCT
032310  174137  001422                   STF     AC1,@#FLTMP0        ;SAVE RESULT
032314  013737  001404  001262           MOV     @#$AC1,@#$REG2      ;AND EXT EXPONENT
                                ;NOW DO  THE B*B
032322  172437  001306                   LDF     @#$TMP4,AC0         ;LOAD OPERAND B
032326  013737  001260  001402           MOV     @#$REG1,@#$AC0
032334  172500                           LDF     AC0,AC1             ;B OPERAND IS IN AC0
032336  013737  001402  001404           MOV     @#$AC0,@#$AC1       ;AND EXT EXPONENT
032344  004767  000322                   JSR     PC,FLTMPY           ;
032350  174102                           STF     AC1,AC2             ;SAVE RESULT IN AC2
032352  013737  001404  001406           MOV     @#$AC1,@#$AC2
                                ;NOW DO  THE A*A
032360  172437  001276                   LDF     @#$TMP0,AC0         ;LOAD OPERAND A
032364  013737  001256  001402           MOV     @#$REG0,@#$AC0
032372  172500                           LDF     AC0,AC1
032374  013737  001402  001404           MOV     @#$AC0,@#$AC1
032402  004767  000264                   JSR     PC,FLTMPY           ;EXECUTE THE MULTIPLY
032406  013737  001404  001410           MOV     @#$AC1,@#$AC3       ;SAVE EXT EXPO OF A*A
                                ;NOW DO  A**2-B**2
032414  172402                           LDF     AC2,AC0             ;GET B*B
032416  013737  001406  001402           MOV     @#$AC2,@#$AC0       ;A*A IN AC1
032424  004767  000306                   JSR     PC,FLTSUB
032430  174137  001432                   STF     AC1,@#FLTMP1        ;SAVE IN MEMORY
032434  013737  001404  001264           MOV     @#$AC1,@#$REG3
                                ;NOW COMPUTE THE RESULTS
                                ;CALCULATE THE NUMBER OF CORRECT BITS
032442  023737  001406  001410           CMP     @#$AC2,@#$AC3       ;DETERMINE WHICH EXP IS LARGER
032450  002003                           BGE     2$                  ;BRANCH IF AC2 LARGER
032452  013737  001410  001406           MOV     @#$AC3,@#$AC2       ;PUT LARGEST IN AC2
```

```
032466  162737  000066  001406           SUB     #54.,@#$AC2
032474  005437  001406                   NEG     @#$AC2
032500  172437  001422                   LDF     @#FLTMP0,AC0      ;GET LEFT HAND SIDE
032504  013737  001262  001402           MOV     @#$REG2,@#$AC0
032512  004767  000220                   JSR     PC,FLTSUB         ;SUBTRACT TO SEE HOW CLOSE THEY ARE
032516  163737  001264  001404           SUB     @#$REG3,@#$AC1    ;SUB EXT EXPONENTS
                                                                   ;ACTUAL EXPONENTS ARE EQUAL
032524  100002                           BPL     1$                ;MAKE SURE RESULT IS POSITIVE
032526  005437  001404                   NEG     @#$AC1
032532  023737  001406  001404   1$:     CMP     @#$AC2,@#$AC1     ;RESULTS WITHIN RANGE ALLOWED?
032540  003401                           BLE     SECT3D            ;BRANCH IF YES
032542  104016                           ERROR   16                ;RESULTS WRONG

                                 ;;*************************************************************
032544  172537  001276           SECT3D: LDF     @#$TMP0,AC1       ;LOAD OPERAND A
032550  172437  001306                   LDF     @#$TMP4,AC0       ;AND OPERAND B
032554  013737  001256  001404           MOV     @#$REG0,@#$AC1
032562  013737  001260  001402           MOV     @#$REG1,@#$AC0
032570  004767  000120                   JSR     PC,FLTDIV         ;GO DIVIDE THEM
032574  004767  000072                   JSR     PC,FLTMPY         ;MULTIPLY RESULT BY B
032600  174137  001422                   STF     AC1,@#FLTMP0      ;SAVE RESULT
032604  013737  001404  001262           MOV     @#$AC1,@#$REG2
032612  172437  001276                   LDF     @#$TMP0,AC0       ;LOAD OPERAND A
032616  174037  001432                   STF     AC0,@#FLTMP1      ;SAVE INCASE TYPE OUT
032622  013737  001256  001402           MOV     @#$REG0,@#$AC0
032630  013737  001256  001264           MOV     @#$REG0,@#$REG3
032636  004767  000074                   JSR     PC,FLTSUB         ;SUBTRACT RIGHT AND LEFT HAND SIDES
032642  163737  001256  001404           SUB     @#$REG0,@#$AC1    ;SEE IF RESULT OK
032650  100002                           BPL     1$                ;ENSURE DIFFERANCE IS POSITIVE
032652  005437  001404                   NEG     @#$AC1
032656  022737  000067  001404   1$:     CMP     #55.,@#$AC1       ;RESULTS WITHIN 2 BITS?
032664  003505                           BLE     RELEB             ;BRANCH IF YES
032666  104016                           ERROR   16                ;RESULTS WRONG
032670  000503                           BR      RELEB

                                 ;;*************************************************************
                                 .SBTTL  FLOATING POINT MULTIPLY ROUTINE
                                 ;*      THIS ROUTINE MULTIPLIES THE CONTENTS OF AC0 AND AC1
                                 ;*      AND LEAVES THE RESULT IN AC1. IT ALSO TAKES CARE OF
                                 ;*      THE SOFTWARE EXPONENTS THAT ARE KEPT IN $AC0 AND $AC1.
                                 ;;*************************************************************
032672  063737  001402  001404   FLTMPY: ADD     @#$AC0,@#$AC1     ;ADD SOFTWARE EXPONENTS
032700  171100                           MULF    AC0,AC1           ;DO THE MULTIPLY
032702  012746  100400                   MOV     #100400,-(SP)     ;PUT CONTROL WORD ON STACK
032706  004737  051042                   JSR     PC,@#EXPEXT       ;CALCULATE EXT EXPONENT
032712  000207                   1$:     RTS     PC                ;RETURN

                                 ;;*************************************************************
                                 .SBTTL  FLOATING POINT DIVIDE ROUTINE
                                 ;*      THIS ROUTINE DIVIDES THE CONTENTS OF AC1 BY AC0
                                 ;*      AND LEAVES THE RESULT IN AC1.
                                 ;;*************************************************************
032714  163737  001402  001404   FLTDIV: SUB     @#$AC0,@#$AC1     ;ADJUST SOFTWARE EXPONENTS
032722  174500                           DIVF    AC0,AC1           ;EXECUTE THE DIVIDE
032724  012746  100400                   MOV     #100400,-(SP)     ;PUT CONTROL WORD ON STACK
```

```
032734  000207                          1$:     RTS     PC                      ;RETURN

                                        ;;*************************************************************
                                        .SBTTL  FLOATING POINT ADD ROUTINE
                                        ;*      THIS ROUTINE ADDS THE CONTENTS OF AC0 TO AC1.
                                        ;*      THIS CAN ONLY BE DONE IF THE SOFTWARE EXPONENTS
                                        ;*      ARE CLOSE ENOUGH TOGETHER SUCH THAT AN ADJUSTMENT
                                        ;*      OF THE REAL EXPONENT LEAVES A NON-ZERO NUMBER.
                                        ;;*************************************************************
032736  010667  000134                  FLTSUB: MOV     SP,SUBFLG               ;SET SUBTRACT FLAG
032742  023737  001402  001404          FLTADD: CMP     @#$AC0,@#$AC1           ;CHECK SOFTWARE EXPONENTS
032750  003016                                  BGT     1$
032752  001434                                  BEQ     2$
                                        ;ACCUMULATOR 1 IS LARGER THAN ACCUMULATOR 0
032754  013702  001404                          MOV     @#$AC1,R2               ;GET OPERAND B SOFTWARE EXP
032760  163702  001402                          SUB     @#$AC0,R2               ;GET DIFFERENCE IN SOFTWARE EXP'S
032764  020227  000071                          CMP     R2,#57.                 ;EXP WITHIN DBL PREC RANGE?
032770  002003                                  BGE     7$                      ;BRANCH IF ADD NOT REQUIRED
                                                                                ;RESULT IS OPERAND B
032772  005402                                  NEG     R2
032774  176402                                  LDEXP   R2,AC0                  ;RELOAD THE EXPONENT
032776  000422                                  BR      2$
033000  176427  177703                  7$:     LDEXP   #-75,AC0                ;FAKE EXPONENT SO HARDWARE
033004  000417                                  BR      2$                      ;WILL DETECT OUT OF RANGE

                                        ;
                                        ;ACCUMULATOR 0 IS LARGER THAN ACCUMULATOR 1
033006  013702  001402                  1$:     MOV     @#$AC0,R2               ;GET SOFTWARE EXP OF OPERAND A
033012  163702  001404                          SUB     @#$AC1,R2               ;GET DIFFERENCE IN EXP'S
033016  013737  001402  001404                  MOV     @#$AC0,@#$AC1           ;MAKE SOFTWARE EXP'S EQUAL
033024  020227  000071                          CMP     R2,#57. ;EXP WITHIN DBL PREC RANGE?
033030  002003                                  BGE     4$                      ;BRANCH IF NO
033032  005402                                  NEG     R2
033034  176502                                  LDEXP   R2,AC1                  ;RELOAD THE EXPONENT
033036  000402                                  BR      2$

                                        ;
                                        ;ACCUMULATOR 0 IS MUCH LARGER THAN ACCUMULATOR 1 SO RESULT IS 0
033040  176527  177703                  4$:     LDEXP   #-75,AC1                ;FAKE EXPONENT SO HARDWARE
                                                                                ;WILL DETECT OUT OF RANGE
033044  005767  000026                  2$:     TST     SUBFLG                  ;ADD OR SUBTRACT?
033050  001402                                  BEQ     5$                      ;BRANCH IF ADD
033052  173100                                  SUBF    AC0,AC1
033054  000401                                  BR      6$
033056  172100                          5$:     ADDF    AC0,AC1                 ;EXECUTE THE ADD
033060  012746  100400                  6$:     MOV     #100400,-(SP)           ;PUT CONTROL WORD ON STACK
033064  004737  051042                          JSR     PC,@#EXPEXT             ;CALCULATE EXT EXPONENT
033070  005067  000002                  3$:     CLR     SUBFLG                  ;INIT SUBTRACT FLAG
033074  000207                                  RTS     PC                      ;RETURN
033076  000000                          SUBFLG: .WORD
033100  000004                          RELEB:  SCOPE
033102  010702                                  MOV     PC,R2
033104  062702  000012                          ADD     #12,R2
033110  012707  034242                          MOV     #RELOC,PC               ;GO RELOCATE PROGRAM CODE
033114  000000                          RELBB:  .WORD   0
                                        ;8888888888888 LAST ADDRESS OF CODE TO BE RELOCATED 88888888888

                                        ;*TEST 74          TELETYPE AND CLOCK TESTS
                                        ;;**********************************************************************
033116  112737  000074  001202  TST74:  MOVB    #74,@#$TSTNM                    ;LOAD TEST NUMBER
033124  000240                          NOP
033126  113737  001202  177570          MOVB    @#$TSTNM,@#SWR
033134  005037  001506          TTYCHK: CLR     @#FACTOR
033140  012704  000100                  MOV     #100,R4         ;SET R4 = CONSTANT 100
033144  032737  000400  001470          BIT     #TTOPT,@#OPT.CP ;BRANCH IF TTY
033152  001002                          BNE     1$              ;ON SYSTEM
033154  000167  000204                  JMP     ARBFIN          ;JUMP IF NOT
033160  122777  000200  146056  1$:     CMPB    #200,@$TPS      ;CHECK IF TTY IS READY
033166  001374                          BNE     1$
033170  012737  001525  001270          MOV     #NULLS-1,@#$REGS;SET ADDRESS OF ASCII STRING TO TYPE
033176  106277  146042                  ASRB    @$TPS           ;SET IE BIT. SEE TPISR FOR INT SERVICE.
033202  000001                          WAIT                    ;WAIT FOR INTERRUPT


033204                          DUMMY:
                                ;ROUTINE TO CHECK PRIORITY ARBITRATION LOGIC
                                ;THE BELOW TEST WILL INHIBIT INTERRUPTS ON LEVEL 6 AND ABOVE (LOCKING
                                ;OUT THE LINE CLOCK) AND THEN SET UP THE TTY TO INTERRUPT. NEXT THE
                                ;PRIORITY LEVEL WILL BE SET TO 0 ALLOWING INTERRUPTS IN WHICH CASE
                                ;THE LINE CLOCK (AT LEVEL 6) SHOULD INTERRUPT BEFORE THE TTY (AT LEVEL 4).
033204  132737  000020  177776  1$:     BITB    #20,@#PSW
033212  001071                          BNE     ARBEX                   ;EXIT TEST IF 'T' BIT SET
033214  030477  146024          2$:     BIT     R4,@$TPS                ;WAIT FOR TTY TO BE NOT
033220  001375                          BNE     2$                      ;BUSY
033222  112737  000300  177776          MOVB    #300,@#PSW              ;SET PRIORITY LEVEL 6
033230  150477  146010          3$:     BISB    R4,@$TPS                ;SET IE BIT
033234  100375                          BPL     3$                      ;AND WAIT FOR READY
033236  032737  001000  001470          BIT     #LKOPT,@#OPT.CP         ;LINE CLOCK AVAILABLE?
033244  001447                          BEQ     ARBFIN                  ;BRANCH IF NO
033246  012737  033336  000064          MOV     #7$,@#TPVEC             ;SET TTY VECTOR
033254  012737  033350  000100          MOV     #8$,@#LKVEC             ;SET CLOCK VECTORS
033262  012737  000340  000102          MOV     #PR7,@#LKVEC+2
033270  005027                          CLR     (PC)+                   ;CLEAR CHECK WORD
033272  000000                  4$:     .WORD   0
033274  000240                          NOP
033276  000240                          NOP
033300  000240                          NOP
033302  010437  177546                  MOV     R4,@#LKS
033306  113700                  5$:     MOVB    @(PC)+,R0               ;GET CLOCK STATUS & BRANCH IF READY
033310  177546                  6$:     .WORD   LKS                     ;CONTAINS ADDRESS OF L CLOCK STAT
033312  100375                          BPL     5$
033314  000240                          NOP                             ;AT THIS TIME BOTH THE CLOCK
                                                                        ;ARE READY TO INTERRUPT
033316  105037  177776                  CLRB    @#PSW                   ;SET PRIORITY LEVEL 0
                                ;A CLOCK INTERRUPT WILL OCCUR (8$) AND LOC 4$ WILL BE INCREMENTED
                                ;AFTER THE CLOCK SERVICE A TTY INTERRUPT WILL OCCUR. THE TTY INT SERV
                                ;ICE WILL SHIFT LEFT 4$.

033322  022767  000002  177742          CMP     #2,4$                   ;CHECK THAT THE CLOCK
033330  001415                          BEQ     ARBFIN                  ;& TTY INTERRUPTED IN
033332  104000                          HLT                             ;THE PROPER SEQUENCE
033334  000413                          BR      ARBFIN
```

```
033336  005077  145702          7$:     CLR     @$TPS                   ;CLEAR IE BIT
033342  006367  177724                  ASL     4$                      ;SHIFT INDICATOR
033346  000002                          RTI                             ;RETURN

033350  005267  177716          8$:     INC     4$
033354  012737  044444  000100          MOV     #LKSRV,@#LKVEC          ;SET CLOCK VECTORS
033362  000002                          RTI


033364  012737  052612  000064  ARBFIN: MOV     #TPISR,@#TPVEC          ;RESTORE TTY VECTOR
033372  005077  145646                  CLR     @$TPS                   ;CLEAR IE BIT
033376                          ARBEX:
                                ;;*****************************************************************
                                ;*TEST 75        TURN ON UBE AND MBT
                                ;*       TURN ON THE MASS BUS TESTER AND UNIBUS EXERCISER IF PRESENT
                                ;;*****************************************************************
033376  112737  000075  001202  TST75:  MOVB    #75,@#$TSTNM            ;LOAD TEST NUMBER
033404  000240                          NOP
033406  113737  001202  177570          MOVB    @#$TSTNM,@#SWR
033414  032737  001000  001470          BIT     #LKOPT,@#OPT.CP         ;BRANCH IF NOT AVAIL
033422  001411                          BEQ     UBESET
033424  012737  044444  000100          MOV     #LKSRV,@#LKVEC
033432  012737  000340  000102          MOV     #PR7,@#LKVEC+2
033440  052737  000100  177546          BIS     #100,@#LKS              ;SET IE BIT

                                ;;*****************************************************************
                                ;TURN ON THE UNIBUS EXERCISER IF PRESENT
033446  105737  001470          UBESET: TSTB    @#OPT.CP        ;IS UBE OPTION AVAILABLE?
033452  100015                          BPL     MBTSET          ;BRANCH IF NO
033454  032737  010000  177570          BIT     #SW12,@#SWR     ;INHIBIT UBE?
033462  001011                          BNE     MBTSET          ;BRANCH IF YES
033464  032737  000040  172516          BIT     #BIT5,@#MMR3    ;IS MAP ON?
033472  001050                          BNE     STMM            ;BRANCH IF YES
033474  004737  051436                  JSR     PC,@#UBEINIT    ;INITIALIZE UBE
033500  012772  064545  000000          MOV     #64545,@(R2)    ;START UBE

                                ;;*****************************************************************
                                ;TURN ON THE MASS BUS TESTER IF PRESENT
033506  032737  002000  001470  MBTSET: BIT     #MBTOPT,@#OPT.CP        ;IS MBT AVAILABLE?
033514  001437                          BEQ     STMM            ;BRANCH IF NO
033516  032737  000010  177570          BIT     #SW3,@#SWR      ;INHIBIT MBT?
033524  001033                          BNE     STMM            ;BRANCH IF YES
033526  122737  000060  001532          CMPB    #60,@#SUBPASS   ;FIRST SUB-PASS?
033534  001027                          BNE     STMM            ;BRANCH IF NO
033536  105737  001200                  TSTB    @#$PASS         ;FIRST PASS?
033542  001024                          BNE     STMM            ;BRANCH IF NO
033544  105737  001503                  TSTB    @#MMON          ;MEM MGMT ON?
033550  001021                          BNE     STMM            ;BRANCH IF YES
033552  052777  000047  146500  MBT1:   BIS     #47,@MBTTBL+12          ;CLEAR THE MBT
033560  012777  000007  146472          MOV     #7,@MBTTBL+12           ;SELECT UNIT 7
033566  005077  146456                  CLR     @MBTTBL+2               ;CLEAR THE WORD COUNT
033572  012777  044132  146470          MOV     #MBTSRV,@MBTTBL+22      ;SETUP INTERRUPT VECTOR
033600  012777  000240  146464          MOV     #PR5,@MBTTBL+24         ;SET VECTOR PSW
033606  112777  000161  146432          MOVB    #161,@MBTTBL            ;START MBT
```

```
                                         .SBTTL  STMM ROUTINE
                                         ;ROUTINE TO SET UP MEMORY MANAGEMENT TO RELOCATE PROGRAM CODE ABOVE 16K
                                         ;CHECK IF PROGRAM IS TO BE RELOCATED.
                                         ;SW6=1=NO RELOCATION
                                         ;
                                         ;*****PLEASE NOTE:     THE RELOCATION ROUTINES WILL BE REFERED TO
                                         ;                      AS TEST 76 IN ERROR TYPEOUTS DURING
                                         ;                      RELOCATION.
                                         ;
                                         ;;************************************************************
033614  112737  000076  001202           STMM:    MOVB    #76,@#STSTNM        ;LOAD TEST NUMBER
033622  000240                                    NOP
033624  113737  001202  177570                    MOVB    @#STSTNM,@#SWR
033632  032737  000100  177570                    BIT     #SW6,@#SWR          ;RELOCATION DISABLED?
033640  001402                                    BEQ     3$                  ;BRANCH IF NO
033642  000167  002450                            JMP     ENDM

                                         ;THE PROGRAM IS GOING TO RELOCATE.
                                         ;RELOCATION WILL BE PERFORMED IN KERNEL MODE WITH PSW SET AT PRIORITY
                                         ;LEVEL 4 (TO PREVENT TTY INTERRUPT-WHICH CHANGES DATA IN PROGRAM)
                                         ;THE 'T' BIT IS CLEARED (IF SET). AFTER THE DATA HAS BEEN WRITTEN IT IS
                                         ;VERIFIED BEFORE EXECUTION.
033646  013727  177776                   3$:      MOV     @#PSW,(PC)+         ;SAVE CURRENT PSW
033652  000000                           OLDPSW:  .WORD   0
033654  012737  000200  177776                    MOV     #PR4,@#PSW
033662  004767  016452                            JSR     PC,CLRTBIT          ;CO CLEAR 'T' BIT IF SET

                                         ;NOW SETUP MEMORY MANAGEMENT REGISTERS.
033666  012700  077406                            MOV     #77406,R0           ;SET CONSTANT=R/W UP 4K WORDS
033672  010037  172300                            MOV     R0,@#KIPDR0         ;SET KIPDR0,1,2,3,& 7 R/W UP 4K WORDS
033676  010037  172302                            MOV     R0,@#KIPDR1
033702  010037  172304                            MOV     R0,@#KIPDR2
033706  010037  172306                            MOV     R0,@#KIPDR3
033712  010037  172310                            MOV     R0,@#KIPDR4
033716  010037  172312                            MOV     R0,@#KIPDR5
033722  010037  172316                            MOV     R0,@#KIPDR7

033726  005037  172340                            CLR     @#KIPAR0            ;NOTE: THESE 2 INSTRUCIONS EFFECTIVELY
033732  012737  000200  172342                    MOV     #200,@#KIPAR1       ;RELOCATE PROGRAM EXECUTION
033740  012737  000400  172344                    MOV     #400,@#KIPAR2
033746  013737  001520  172346                    MOV     @#NEXPAR,@#KIPAR3           ;SET UP KIPAR3 & KIPAR4 & 5
033754  013737  172346  172350                    MOV     @#KIPAR3,@#KIPAR4
033762  062737  000200  172350                    ADD     #200,@#KIPAR4
033770  013737  172346  172352                    MOV     @#KIPAR3,@#KIPAR5
033776  062737  000400  172352                    ADD     #400,@#KIPAR5
034004  012737  177600  172356                    MOV     #177600,@#KIPAR7;AND OF COUSE THE I/O PAGE
                                         ;NOW SETUP USER MEM MGMT REGISTERS
034012  010037  177600                   1$:      MOV     R0,@#UIPDR0         ;SET UP USER MEM MGMT REGS
034016  010037  177602                            MOV     R0,@#UIPDR1
034022  010037  177604                            MOV     R0,@#UIPDR2
034026  010037  177616                            MOV     R0,@#UIPDR7
034032  016737  145462  177640                    MOV     NEXPAR,@#UIPAR0
034040  013737  177640  177642                    MOV     @#UIPAR0,@#UIPAR1
034046  062737  000200  177642                    ADD     #200,@#UIPAR1
034054  013737  177640  177644                    MOV     @#UIPAR0,@#UIPAR2
```

```
034070  013737  172356  177656             MOV     @#KIPAR7,@#UIPAR7

034076  010037  172200                     MOV     R0,@#SIPDR0      ;SET UP SUPERVISOR MEM MGMT REGS
034102  010037  172202                     MOV     R0,@#SIPDR1
034106  010037  172204                     MOV     R0,@#SIPDR2
034112  010037  172216                     MOV     R0,@#SIPDR7
034116  016737  145376  172240             MOV     NEXPAR,@#SIPAR0
034124  013737  172240  172242             MOV     @#SIPAR0,@#SIPAR1
034132  062737  000200  172242             ADD     #200,@#SIPAR1
034140  013737  172240  172244             MOV     @#SIPAR0,@#SIPAR2
034146  062737  000400  172244             ADD     #400,@#SIPAR2
034154  013737  172356  172256             MOV     @#KIPAR7,@#SIPAR7
034162  012737  000001  177572             MOV     #1,@#SR0         ;ENABLE MEM MGMT
034170  012737  000060  172516             MOV     #60,@#SR3        ;SETUP SR3
034176  110637  001503                     MOVB    SP,@#MMON        ;SET MEM MGMT ON IND = ON
034202  005037  000006             RETRY:  CLR     @#ERRVEC+2
034206  012737  036314  000004             MOV     #ENDMEM,@#ERRVEC;SET TIME OUT TRAP VECTOR
034214  012702  060000                     MOV     #60000,R2        ;SETUP GENERAL REGISTERS
034220  005000                             CLR     R0               ;DATA WILL BE RELOCATED FROM
                                                                    ;ADDRESS IN R0 TO ADDRESS IN R2
034222  012703  137776                     MOV     #137776,R3       ;GET 12K WORDS TO RELOCATE
034226  010013                             MOV     R0,(R3)          ;TRAP TO ENDMEM IF INSUFFICIENT MEMORY
034230  012737  053442  000004             MOV     #ERPRT,@#ERRVEC  ;RESTORE ERROR TRAP VECTOR
034236  000137  034436                     JMP     @#IOMON
                                   ;;*********************************************************
                                   .SBTTL  RELOCATION ROUTINE
                                   ;*      THIS ROUTINE IS USED TO RELOCATE THE 9 SUBTESTS UP TO 28K.
                                   ;*      IF RELOCATION BY AN I/O DEVICE IS SELECTED, CONTROL IS PASSED
                                   ;*      TO THE I/O MONITOR.
                                   ;*      ENTER WITH:
                                   ;*              FRSTAD=PHYSICAL ADDRESS OF FIRST CODE
                                   ;*              FACTOR=NUMBER OF BYTES ABOVE BASE CODE
                                   ;*              R2    =LAST PHYSICAL ADDRESS OF THE SECTION
                                   ;*      EXIT TO I/O MONITOR WITH:
                                   ;*              OLDBASE=FIRST PHYSICAL ADDRESS TO BE RELOCATED
                                   ;*              NWBASL =FIRST PHYSICAL ADDRESS TO RELOCATE TO
                                   ;*              IOWC   =TWO'S COMPLIMENT WORD COUNT
                                   ;;*********************************************************
034242  032737  000100  177570     RELOC:  BIT     #SW6,@#SWR                ;IS RELOCATION DISABLED?
034250  001067                             BNE     EXITRE                    ;BRANCH IF YES
034252  105737  001503                     TSTB    @#MMON                    ;IS MEMORY MGMT ON?
034256  001064                             BNE     EXITRE                    ;BRANCH IF YES
034260  013700  001512                     MOV     @#FRSTAD,R0               ;GET FIRST ADDRESS TO BE RELOCATED
034264  010005                             MOV     R0,R5
                                   ;LAST ADDRESS IS IN R2
034266  010203                             MOV     R2,R3                     ;SAVE LAST ADDRESS
034270  010204                             MOV     R2,R4
034272  160004                             SUB     R0,R4                     ;R4 NOW HAS BYTE COUNT
034274  010437  001510                     MOV     R4,@#$FACTOR              ;SAVE BYTE COUNT
034300  005737  001506                     TST     @#FACTOR                  ;FIRST RELOC IS TO ENDTAG+2
034304  001004                             BNE     1$                        ;BRANCH IF NOT EXECUTING BASE CODE
034306  010237  034434                     MOV     R2,@#RETPC                ;SAVE RETURN PC TO NEXT SECTION
034312  013702  001514                     MOV     @#FRSTMEM,R2              ;GET FIRST ADDRESS TO RELOCATE TO
034316  060204                     1$:     ADD     R2,R4                     ;R4 NOW CONTAINS LAST MEM ADDRESS
034320  020437  001516                     CMP     R4,@#LSTMEM               ;ENOUGH MEMORY?
```

```
034326  160204                                  SUB     R2,R4                   ;R4 NOW HAS BYTE COUNT
034330  005037  001506                          CLR     @#FACTOR
034334  032737  000400  177570                  BIT     #SW8,@#SWR              ;INHIBIT RELOC BY I/O DEVICE?
034342  001014                                  BNE     RELNIO                  ;BRANCH IF YES
034344  010037  001540                          MOV     R0,@#OLDBASE            ;SAVE START ADDRESS
034350  010237  001542                          MOV     R2,@#NWBASL             ;SAVE NEW BASE ADDRESS
034354  005037  001544                          CLR     @#NWBASH                ;
034360  006204                                  ASR     R4                      ;MAKE IT A WORD COUNT
034362  005404                                  NEG     R4                      ;GET TWO'S COMPLIMENT
034364  010437  001546                          MOV     R4,@#IOWC               ;SAVE R4 AS WORDCOUNT
034370  000167  000120                          JMP     ENTER2                  ;GO TO I/O MONITOR

                                        ;RELOCATE BY CPU-MEMORY MANAGEMENT OFF
034374  012022                          RELNIO: MOV     (R0)+,(R2)+             ;RELOCATE CODE
034376  020003                                  CMP     R0,R3                   ;DONE YET?
034400  001375                                  BNE     RELNIO                  ;BRANCH IF NO
034402  004737  051746                          JSR     PC,@#CHKDAT             ;GO CHECK DATA
034406  102010                                  BVC     EXITRE
034410  010037  001276                          MOV     R0,@#$TMP0              ;SAVE R0 FOR TYPEOUT
034414  010237  001474                          MOV     R2,@#VADR               ;SAVE R2
034420  004737  051644                          JSR     PC,@#CNVADR             ;CONVERT R2 TO A PHYSICAL ADR
034424  104006                                  ERROR   6
034426  000401                                  BR      NOMEM
034430  010207                          EXITRE: MOV     R2,PC                   ;GO EXECUTE RELOCATED CODE
034432  011707                          NOMEM:  MOV     (PC),PC                 ;GO TO NEXT SECTION
034434  000000                          RETPC:  .WORD   0                       ;CONTAINS PC OF NEXT SECTION
                                        ;;**********************************************************
                                        .SBTTL  I/O RELOCATION MONITOR
                                        ;*      THIS ROUTINE IS USED TO SCHEDULE I/O DEVICES FOR SUBTEST
                                        ;*      RELOCATION AND PROGRAM RELOCATION. THE I/O DEVICE UNIT
                                        ;*      NUMBER IS DETERMINED, THE BUS ADDRESS CALCULATED, THE WORD
                                        ;*      COUNT CALCULATED AND PASSED TO THE DEVICE HANDLER.
                                        ;;**********************************************************
034436  012737  034444  001212          IOMON:  MOV     #1$,@#$LPERR    ;SETUP ERROR LOOP
034444  005037  001540                  1$:     CLR     @#OLDBASE
034450  013705  172346                          MOV     @#KIPAR3,R5             ;SETUP R2 AND R3
034454  005004                                  CLR     R4                      ;TO FORM 22 BIT ADDRESS
034456  073427  000006                          ASHC    #6,R4                   ;FORM 22 BIT ADDRESS
034462  010537  001542                          MOV     R5,@#NWBASL             ;SAVE LOWER 16 BITS
034466  010437  001544                          MOV     R4,@#NWBASH             ;SAVE UPPER 6 BITS
034472  032737  000400  177570                  BIT     #SW8,@#SWR              ;RELOCATE VIA I/O?
034500  001402                                  BEQ     2$                      ;BRANCH IF YES
034502  000167  001436                          JMP     RELOCP                  ;GO RELOCATE VIA CP
034506  012737  174000  001546          2$:     MOV     #174000,@#IOWC          ;SET WORD COUNT TO 2K
034514  005037  001302                  ENTER2: CLR     @#$TMP2
034520  012737  177776  001556                  MOV     #-2,@#RNTBINX           ;SETUP RUN TABLE INDEX
034526  005037  001276                          CLR     @#$TMP0
034532  005002                                  CLR     R2                      ;CLEAR LEGAL DEV FLAG
034534  032737  000040  177570          41$:    BIT     #SW5,@#SWR              ;INHIBIT ROUND ROBIN?
034542  001416                                  BEQ     50$                     ;BRANCH IF NO
034544  005737  001276                          TST     @#$TMP0                 ;FLAG SET?
034550  001027                                  BNE     43$                     ;BRANCH IF YES
034552  113737  177570  001552                  MOVB    @#SWR,@#DEVINDX         ;GET DEVICE FROM SWITCHES
034560  042737  177770  001552                  BIC     #177770,@#DEVINDX       ;MASK LOWER 3 BITS
034566  006337  001552                          ASL     @#DEVINDX               ;ADJUST FOR WORD INDEX
034572  005237  001276                          INC     @#$TMP0                 ;SET FLAG
```

```
034600  012705  000010          50$:    MOV     #10,%5                          ;SET SOB COUNT
034604  022737  000016  001552  40$:    CMP     #16,@#DEVINDX                   ;LAST DEVICE YET?
034612  001003                          BNE     42$                             ;BRANCH IF NO
034614  012737  177776  001552  48$:    MOV     #-2,@#DEVINDX                   ;INIT DEVICE INDEX
034622  062737  000002  001552  42$:    ADD     #2,@#DEVINDX                    ;INCREMENT INDEX
034630  013703  001552          43$:    MOV     @#DEVINDX,R3                    ;GET INDEX
034634  012737  000401  001300          MOV     #401,@#STMP1                    ;INIT UNIT MASK
034642  012704  000010                  MOV     #10,R4                  ;SET SOB COUNT
034646  133763  001300  001606  44$:    BITB    @#STMP1,SYSSIZE(R3)             ;IS THIS UNIT EXISTENT?
034654  001405                          BEQ     52$                             ;BRANCH IF NO
034656  005202                          INC     R2                              ;SET LEGAL DEVICE FLAG
034660  133763  001301  001607          BITB    @#STMP1+1,SYSSIZE+1(R3)         ;HAS IT BEEN USED?
034666  001520                          BEQ     11$                             ;BRANCH IF NO
034670  006337  001300          52$:    ASL     @#STMP1                         ;SELECT NEXT UNIT
034674  077414                          SOB     R4,44$                          ;CONTINUE
034676  005737  001276                  TST     @#STMP0                         ;INHIBIT ROUND ROBIN?
034702  001013                          BNE     45$                             ;BRANCH IF YES
034704  077541                          SOB     R5,40$                          ;CONTINUE
034706  005702                          TST     R2                              ;ANY DEVICES AT ALL?
034710  001442                          BEQ     46$                             ;BRANCH IF NO
034712  012704  000010                  MOV     #10,R4                          ;SET SOB COUNT
034716  012701  001607                  MOV     #SYSSIZE+1,R1                   ;GET ADR OF SIZE TABLE
034722  105021                  47$:    CLRB    (R1)+                           ;CLEAR ALL USED BITS
034724  005201                          INC     R1                              ;IN ALL DEVICES
034726  077403                          SOB     R4,47$                          ;CONTINUE
034730  000701                          BR      41$
034732  005702                  45$:    TST     R2                              ;WAS IT A LEGAL DEVICE?
034734  001403                          BEQ     49$                             ;BRANCH IF NO
034736  105063  001607                  CLRB    SYSSIZE+1(R3)                   ;CLEAR ALL USED BITS THIS DEV
034742  000732                          BR      43$
034744  010367  000016          49$:    MOV     R3,60$                          ;
034750  062767  053612  000010          ADD     #MSGINX,60$                     ;GEN MESSAGE ADR
034756  017767  000004  000002          MOV     @60$,60$
034764  104400                          TYPE
034766  000000                  60$:    .WORD
034770  104400  034776                  TYPE    ,65$                    ;;TYPE ASCIZ STRING
034774  000407                          BR      64$                     ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /UNAVAILABLE/<CRLF>
035014                          64$:
035014  000637                          BR      ENTER2
035016  105737  001504          46$:    TSTB    @#QV                            ;ACT11?
035022  001016                          BNE     51$                             ;BRANCH IF YES
035024  005227  177777                  INC     #-1
035030  001013                          BNE     51$
035032  104400  035040                  TYPE    ,67$                    ;;TYPE ASCIZ STRING
035036  000410                          BR      66$                     ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  ?NO I/O DEVICES?<CRLF>
035060                          66$:
035060  105737  001503          51$:    TSTB    @#MMON                          ;MGMT ON?
035064  001012                          BNE     61$                             ;BRANCH IF YES
035066  013700  001540                  MOV     @#OLDBASE,R0                    ;RESTORE R0
035072  013702  001542                  MOV     @#NWBASL,R2                     ;RESTORE R2
035076  013703  001510                  MOV     @#SFACTOR,R3                    ;GET RELOCATION FACTOR
035102  060003                          ADD     R0,R3                           ;FORM LAST ADDRESS
035104  010005                          MOV     R0,R5                           ;SETUP R5
```

```
035112  012702  060000          61$:    MOV     #60000,R2                       ;SETUP REGISTERS
035116  012703  137776                  MOV     #137776,R3                      ;WITH FROM
035122  005000                          CLR     R0                              ;AND TOO ADDRESS
035124  000137  036144                  JMP     @#RELOCP                        ;RELOCATE VIA CP
035130  105763  001676          11$:    TSTB    RP3HSTAT(R3)                    ;IS HANDLER BUSY?
035134  100405                          BMI     8$                              ;BRANCH IF NO
035136  005737  001276                  TST     @#$TMP0                         ;ROUND ROBIN?
035142  001372                          BNE     11$                             ;BRANCH IF NO
035144  000167  177364                  JMP     41$
035150  005763  001676          8$:     TST     RP3HSTAT(R3)                    ;DID HANDLER FAIL?
035154  100005                          BPL     62$                             ;BRANCH IF NO
035156  005737  001276                  TST     @#$TMP0                         ;ROUND ROBIN
035162  001402                          BEQ     62$                             ;BRANCH IF YES
035164  000137  036124                  JMP     @#15$
035170  153763  001301  001607  62$:    BISB    @#$TMP1+1,SYSSIZE+1(R3) ;SET UNIT USED BIT
035176  005002                          CLR     R2
035200  006037  001300          30$:    ROR     @#$TMP1                         ;ENCODE THE BIT POSITION
035204  005202                          INC     R2                              ;INTO A UNIT NUMBER
035206  103374                          BCC     30$
035210  005302                          DEC     R2
035212  010237  001554                  MOV     R2,@#UNITNO                     ;SAVE UNIT NUMBER
035216  013763  001546  001716  10$:    MOV     @#IOWC,RP3HWC(R3)               ;GIVE WORD COUNT TO HANDLER
035224  010304                          MOV     R3,R4                           ;
035226  072427  000003                  ASH     #3,R4                           ;ENCODE DEVICE FOR RUNTABLE
035232  053704  001554                  BIS     @#UNITNO,R4                     ;ENCODE UNIT NUMBER
035236  006304                          ASL     R4
035240  062737  000002  001556          ADD     #2,@#RNTBINX                    ;INCREMENT RUN TABLE INDEX
035246  013702  001556                  MOV     @#RNTBINX,R2                    ;GET RUN TABLE INDEX
035252  110462  001627                  MOVB    R4,RUNTBL+1(R2)                 ;ENTER DEV & UNIT IN TABLE
035256  013763  001554  002012          MOV     @#UNITNO,RP3UNIT(R3)            ;GIVE HANDLER UNIT NUMBER
035264  012737  000240  000012          MOV     #PR5,@#RESVEC+2                 ;SETUP RESERVED VECTOR PSW
035272  016337  002026  000010          MOV     RP3HANA(R3),@#RESVEC            ;SETUP RESERVED VECTOR
035300  006303                          ASL     R3                              ;ADJUST INDEX
035302  013763  001540  001732          MOV     @#OLDBASE,RP3OLD(R3)            ;GIVE HANDLER OLD BASE ADDRESS
035310  013763  001542  001762          MOV     @#NWBASL,RP3NWL(R3)             ;GIVE HANDLER
035316  013763  001544  001764          MOV     @#NWBASH,RP3NWH(R3)             ;NEW BASE ADDRESS
035324  005063  001734                  CLR     RP3OLD+2(R3)                    ;ENSURE OLD BASE HIGH IS CLR
035330  004710                          CALLHANDLER
035332  105737  001503                  TSTB    @#MMON                          ;IS MEMORY MANAGEMENT ON?
035336  001416                          BEQ     13$                             ;BRANCH IF NO
035340  022737  000012  001556          CMP     #12,@#RNTBINX                   ;TRANSFERED 12K YET?
035346  001412                          BEQ     13$                             ;BRANCH IF YES
035350  062737  010000  001540          ADD     #10000,@#OLDBASE                ;ADD 2K
035356  062737  010000  001542          ADD     #10000,@#NWBASL                 ;TO BASE
035364  005537  001544                  ADC     @#NWBASH                        ;ADDRESSES
035370  000137  034534                  JMP     @#41$
035374  113705  001603          13$:    MOVB    @#LTICKS+1,R5                   ;GET SECOND COUNT
035400  062705  000002                  ADD     #2,R5                           ;INCREMENT BY TWO
035404  162705  000074                  SUB     #60.,R5                         ;ENSURE RESULT IS 59 OR LESS
035410  100002                          BPL     31$                             ;
035412  062705  000074                  ADD     #60.,R5                         ;COUNT WAS LESS THAN 58-RESTORE
035416  012700  000010          31$:    MOV     #10,R0                          ;SET SOB COUNT
035422  005002                          CLR     R2
035424  005003                          CLR     R3
035426  005004                          CLR     R4
```

```
035434  005504                                  ADC     R4                              ;STATUS WORDS. WHEN ALL
035436  062702  000002                          ADD     #2,R2                           ;TRANSFERS ARE FINISHED
035442  077006                                  SOB     R0,14$                          ;RESULT WILL BE 2000
035444  006103                                  ROL     R3                              ;(WITHOUT ROTATE)
035446  005504                                  ADC     R4
035450  022703  004000                          CMP     #4000,R3                        ;ALL DONE?
035454  001406                                  BEQ     32$                             ;BRANCH IF YES
035456  123705  001603                          CMPB    @#LTICKS+1,R5                   ;TWO SECONDS ELAPSED YET?
035462  001355                                  BNE     31$                             ;BRANCH IF NO
035464  104015                                  ERROR   15                              ;DEVICE HUNG
035466  000177  143520                          JMP     @$LPERR                         ;RESTART RELOCATION
035472  005704                          32$:    TST     R4                              ;ANY DEVICE ERRORS?
035474  001402                                  BEQ     82$                             ;BRANCH IF NO
035476  000167  000422                          JMP     15$                             ;ERROR
035502  105737  001503                  82$:    TSTB    @#MMON                          ;MEM MGMT ON?
035506  001012                                  BNE     25$                             ;BRANCH IF YES
035510  013705  001540                          MOV     @#OLDBASE,R5                    ;SETUP R5 FOR DATA CHECK
035514  010500                                  MOV     R5,R0
035516  063700  001510                          ADD     @#$FACTOR,R0                    ;GET LAST ADDRESS
                                                                                        ;OF GOOD DATA
035522  013702  001542                          MOV     @#NWBASL,R2
035526  063702  001510                          ADD     @#$FACTOR,R2                    ;GET LAST ADDRESS
                                                                                        ;OF DATA TO BE CHECKED
035532  000406                                  BR      22$                             ;CONTINUE
035534  012700  057776                  25$:    MOV     #57776,R0                       ;GET LAST ADR OF GOOD DATA
035540  012702  137776                          MOV     #137776,R2                      ;GET LAST ADR OF DATA TO BE CHECKED
035544  012705  002100                          MOV     #2100,R5                        ;DON'T CHECK FIRST 2100 LOCATIONS
035550  004737  051746                  22$:    JSR     PC,@#CHKDAT                     ;GO CHECK DATA
035554  102413                                  BVS     81$                             ;BRANCH IF ERROR
035556  105737  001204                          TSTB    @#$ERFLG                        ;ANY ERRORS?
035562  001002                                  BNE     83$                             ;BRANCH IF YES
035564  000167  000412                          JMP     EXIT                            ;RETURN
035570  032737  001000  177570          83$:    BIT     #SW9,@#SWR                      ;LOOP ON ERROR?
035576  001473                                  BEQ     100$+2                          ;BRANCH IF NO
035600  000167  000244                          JMP     20$                             ;GO DO FUNCTION AGAIN
035604  005001                          81$:    CLR     R1
035606  010037  001276                          MOV     R0,@#$TMP0
035612  010237  001474                          MOV     R2,@#VADR
035616  010203                                  MOV     R2,R3                           ;SAVE ERROR ADDRESS
035620  005004                                  CLR     R4
035622  105737  001503                          TSTB    @#MMON                          ;IS MEM MGMT ON?
035626  001406                                  BEQ     16$                             ;BRANCH IF NO
035630  162703  010000                  17$:    SUB     #10000,R3                       ;SUBTRACT 2K FROM ERROR ADDRESS
035634  100403                                  BMI     16$                             ;BRANCH IF BLOCK IS FOUND
035636  062704  000002                          ADD     #2,R4                           ;COUNT ONE MORE BLOCK
035642  000772                                  BR      17$                             ;CONTINUE
                                        ;R4 NOW CONTAINS INDEX OF ERROR FOR RUN TIME TABLE
035644  116404  001627                  16$:    MOVB    RUNTBL+1(R4),R4                 ;GET DEVICE THAT FAILED
035650  042704  177400                          BIC     #177400,R4                      ;ENSURE HIGH BYTE CLEAR
035654  006004                                  ROR     R4                              ;THROW AWAY LSB
035656  005005                                  CLR     R5                              ;ENSURE R5 CLEAR
035660  073427  177775                          ASHC    #-3,R4                          ;GET UNIT NUMBER IN R5
035664  010500                                  MOV     R5,R0
035666  072027  177764                          ASH     #-14,R0
035672  042700  177770                          BIC     #177770,R0
```

```
035702  010403                          MOV     R4,R3                   ;AND DEVICE INDEX IN R4 & R3
035704  010337  001302                  MOV     R3,@#$TMP2
035710  012737  000001  001300          MOV     #1,@#$TMP1                      ;ENCODE 3 BIT UNIT NO INTO
035716  162705  020000          19$:    SUB     #20000,R5               ;ONE BIT IN THE LOW BYTE OF $TMP1
035722  103403                          BCS     18$                     ;BRANCH IF DONE
035724  006137  001300                  ROL     @#$TMP1                 ;SELECT NEXT UNIT
035730  000772                          BR      19$                     ;CONTINUE
035732  012737  036050  001212  18$:    MOV     #20$,@#$LPERR           ;SETUP LOOP RETURN
035740  005701                          TST     R1                      ;DEVICE ERROR?
035742  001010                          BNE     100$                    ;BRANCH IF YES
035744  104010                          ERROR   10                      ;DATA CHECK ERROR
035746  105737  001503                  TSTB    @#MMON                  ;MGMT ON?
035752  001002                          BNE     70$                     ;BRANCH IF YES
035754  000137  034514          71$:    JMP     @#ENTER2
035760  000137  034436          70$:    JMP     @#IOMON
035764  104007                  100$:   ERROR   7
035766  042763  100000  001676          BIC     #BIT15,RP3HSTAT(R3)     ;CLEAR THE ERROR
035774  022703  000002                  CMP     #2,R3                   ;RK05 ERROR?
036000  002405                          BLT     90$                     ;BRANCH IF RH70
036002  003016                          BGT     92$                     ;BRANCH IF RP03
036004  112777  000001  144114          MOVB    #1,@RKCS                ;RK CONTROLLER CLEAR
036012  000412                          BR      92$
036014  022703  000012          90$:    CMP     #12,R3                  ;RS04?
036020  001004                          BNE     91$                     ;BRANCH IF NO
036022  052777  000040  144164          BIS     #BIT5,@RSCS2            ;CLEAR RS CONTROLLER
036030  000403                          BR      92$
036032  052777  000040  144114  91$:    BIS     #BIT5,@RP4CS2           ;CLEAR RP04 CONTROLLER
036040  105737  001503          92$:    TSTB    @#MMON                  ;MGMT ON?
036044  001345                          BNE     70$                     ;BRANCH IF YES
036046  000742                          BR      71$
036050  052763  000400  001676  20$:    BIS     #BIT8,RP3HSTAT(R3)      ;SET REPEAT FLAG IN HANDLER
036056  016337  002026  000010          MOV     RP3HANA(R3),@#RESVEC    ;SETUP RESERVED INSTRUCTION VECTOR
036064  000010                          CALLHANDLER
036066  105763  001676          21$:    TSTB    RP3HSTAT(R3)            ;HANDLER FINISHED?
036072  100375                          BPL     21$                     ;BRANCH IF NO
036074  005763  001676                  TST     RP3HSTAT(R3)            ;ANY ERROR?
036100  100714                          BMI     18$                     ;BRANCH IF YES
036102  005701                          TST     R1                      ;DEVICE ERROR?
036104  001002                          BNE     80$                     ;BRANCH IF YES
036106  000167  177364                  JMP     32$+4                   ;GO CHECK DATA
036112  032737  001000  177570  80$:    BIT     #BIT9,@#SWR             ;STILL LOOPING?
036120  001353                          BNE     20$                     ;BRANCH IF YES
036122  000721                          BR      100$+2                  ;CONTINUE TEST
                                ;THE FOLLOWING CODE HANDLES DEVICE ERROR ON RELOCATION
036124  005004                  15$:    CLR     R4                      ;SET INDEX
036126  010601                          MOV     SP,R1
036130  005764  001642          24$:    TST     RUNTRAK(R4)             ;SEARCH FOR DEVICE ERROR
036134  100643                          BMI     16$                     ;BRANCH IF ERROR
036136  062704  000002                  ADD     #2,R4                   ;INCREMENT INDEX
036142  000772                          BR      24$                     ;CONTINUE SEARCH
                                ;RELOCATE BY CPU-MEMORY MANAGEMENT ON
036144  012022                  RELOCP: MOV     (R0)+,(R2)+             ;RELOCATE CODE
036146  020302                          CMP     R3,R2                   ;DONE YET?
036150  001375                          BNE     RELOCP                  ;BRANCH IF NO
036152  012705  001700                  MOV     #1700,R5
```

```
036162  102007                                  BVC     EXIT
036164  010037  001276                          MOV     R0,@#$TMP0
036170  010237  001474                          MOV     R2,@#VADR
036174  104006                                  ERROR   6
036176  000167  176000                          JMP     RETRY
036202  105737  001503                  EXIT:   TSTB    @#MMON                  ;MEM MGMT ON?
036206  001002                                  BNE     .+6                     ;BRANCH IF YES
036210  000137  034430                          JMP     @#EXITRE
036214  062737  000077  001520                  ADD     #77,@#NEXPAR            ;SET VALUE FOR NEXT RELOCATION
036222  013737  172346  172340                  MOV     @#KIPAR3,@#KIPAR0
036230  013737  172350  172342                  MOV     @#KIPAR4,@#KIPAR1
036236  013737  172352  172344                  MOV     @#KIPAR5,@#KIPAR2
                                        ;*****************************************************************
                                        ;PROGRAM IS NOW EXECUTING IN KERNEL MODE RELOCATED TO ADDRESS AS SPEC-
                                        ;IFIED IN KIPAR0. FOR EX. IF KIPAR0=1600 THEN PROGRAM EXECUTING AT
                                        ;ADDRESS 160000+(PC)
036244  013700  172340                          MOV     @#KIPAR0,R0             ;GET PAR0
036250  072027  177771                          ASH     #-7,R0                  ;GET BITS <14:7> IN LOW BYTE
036254  110037  001203                          MOVB    R0,@#$TSTNM+1           ;PUT IN DSPLAY REG HIGH BYTE
036260  012706  001200                          MOV     #KERSTK,SP              ;SET KERNEL STACK PTR
036264  005037  177776                          CLR     @#PSW
036270  016746  175356                          MOV     OLDPSW,-(SP)            ;RESTORE OLD PSW
036274  012746  005372                          MOV     #LOOP,-(SP)
036300  105737  001502                          TSTB    @#NEXEC                 ;BRANCH IF TEST CODE TO
036304  001402                                  BEQ     1$                      ;BE EXECUTED
036306  012716  033614                          MOV     #STMM,(SP)
036312  000002                          1$:     RTI                             ;RESTART PROGRAM AT LOOP

                                        ;WHEN RELOCATION ABOVE 28K IS COMPLETE PROGRAM TRAPS TO ENDMEM.
036314  022626                          ENDMEM: CMP     (SP)+,(SP)+             ;POP STACK TWICE
036316  005037  177572                  ENDM:   CLR     @#SR0                   ;DISABLE MEM MGMT
036322  042737  000020  172516                  BIC     #BIT4,@#MMR3            ;CLEAR 22 BIT MODE

                                        ;*****************************************************************
                                        ;AT THIS TIME A 'SUB-PASS' HAS BEEN COMPLETED.
                                        ;PROGRAM NOW EXECUTING IN KERNEL MODE AT PC AS SHOWN (NO RELOCATION)
036330  012737  000600  001520                  MOV     #600,@#NEXPAR           ;RESET NEXT VALUE FOR PAR REGISTERS
036336  005737  003244                          TST     @#PROT
036342  001403                                  BEQ     2$
036344  012737  001600  001520                  MOV     #1600,@#NEXPAR
036352  105037  001503                  2$:     CLRB    @#MMON                  ;SET MEM MGMT ON IND = OFF
                                        ;;***********************************************************
                                        .SBTTL  END OF SUB-PASS ROUTINE
                                        ;*      THIS ROUTINE SETSUP THE PSW AND MAINTENANCE REGISTERS
                                        ;*      FOR THE NEXT SUB-PASS. IT THEN STARTS THE PRINTER
                                        ;*      (IF NOT ON ACT11) FOR TYPING THE END OF SUB-PASS MESSAGE.
                                        ;;***********************************************************
036356                                  END:
036356  012737  053442  000004          END1:   MOV     #ERPRT,@#ERRVEC
036364  005037  177776                          CLR     @#PSW                   ;CLEAR MODE BITS IN PSW
036370  004767  013744                          JSR     PC,CLRTBIT              ;GO CLEAR 'T' BIT IF SET
036374  012706  001200                          MOV     #KERSTK,SP              ;SET KERNEL STACK PTR
036400  032777  000100  142636                  BIT     #100,@$TPS              ;CHECK IF OUTPUT DEVICE IS BUSY
036406  001374                                  BNE     .-6                     ;IS AVAILABLE
036410  105237  001532                  1$:     INCB    @#SUBPASS
```

```
036420  162702  000060                  SUB     #60,R2
036424  022702  000006                  CMP     #6,R2                   ;END OF TEST?
036430  001013                          BNE     2$                      ;BRANCH IF NOT AT END
036432  012737  000060  001532          MOV     #60,@#SUBPASS           ;INIT SUBPASS COUNT TO ASCII 0
036440  005037  177750                  CLR     @#MAINT                 ;CLEAR MAINTENANCE REG
036444  005037  001604                  CLR     @#$MAINT                ;CLEAR SOFTWARE VALUE
036450  005046                          CLR     -(SP)
036452  012746  036560                  MOV     #$EOP,-(SP)
036456  000002                          RTI
036460  006302                  2$:     ASL     R2
036462  012737  001472  000014          MOV     #$RTRN,@#TBITVEC        ;SET 'T' TRAP VECTOR
036470  012737  001531  001270          MOV     #SUBPASS-1,@#$REG5
036476  106277  142542                  ASRB    @$TPS
036502  016246  053536                  MOV     PSWTAB(2),-(SP) ;PUSH NEXT PASS PSW ON STACK
036506  012746  005372                  MOV     #LOOP,-(SP)     ;RESART PROGRAM AT LOOP
036512  016237  053552  001604          MOV     MRGTAB(R2),@#$MAINT
036520  016237  053552  177750          MOV     MRGTAB(R2),@#MAINT
036526  105737  001504          3$:     TSTB    @#QV            ;QV PASS?
036532  001011                          BNE     RTI1            ;BRANCH IF YES
036534  122777  000200  142502          CMPB    #200,@$TPS      ;IS PRINTER READY?
036542  001371                          BNE     3$              ;BRANCH IF NO
036544  012737  054602  001270          MOV     #MSG20-1,@#$REG5
036552  106277  142466                  ASRB    @$TPS           ;TYPE END SUBPASS MESSAGE
036556  000002                  RTI1:   RTI                     ;RESTART PROGRAM AT LOOP WITH NEW PSW
                                                                ;(FROM TABLE BELOW)

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

                                .SBTTL  END OF PASS ROUTINE

                                ;*INCREMENT THE PASS NUMBER ($PASS)
                                ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
                                ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
                                ;*IF THERES A MONITOR GO TO IT
                                ;*IF THERE ISN'T JUMP TO LOOP

036560                          $EOP:
036560  004737  046660                  JSR     PC,@#TYPTIME
036564  005067  142412                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
036570  005067  142522                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
036574  005267  142400                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
036600  042767  100000  142372          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
036606  005327                          DEC     (PC)+           ;;LOOP?
036610  000001                  $EOPCT: .WORD   1
036612  003063                          BGT     $DOAGN          ;;YES
036614  012737                          MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
036616  000001                  $ENDCT: .WORD   1
036620  036610                          $EOPCT
036622  104400  036630                  TYPE    ,65$            ;;TYPE ASCIZ STRING
036626  000407                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <12><15>/END PASS #/
036646                          64$:
036646  016746  142326                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                                                ;;TYPE PASS NUMBER
036652  104410                          TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
```

```
036660  000421                           BR      66$                     ;;GET OVER THE ASCIZ
                                 ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
036724                           66$:
036724  016746  142264                   MOV     $ERTTL,-(SP)            ;;SAVE $ERTTL FOR TYPEOUT
                                                                         ;;TOTAL NUMBER OF ERRORS
036730  104410                           TYPDS                           ;;GO TYPE--DECIMAL ASCII WITH SIGN
036732  104400  001327                   TYPE    ,$CRLF                  ;;TYPE CARRIAGE RETURN, LINE FEED
036736  005067  142252                   CLR     $ERTTL                  ;;CLEAR ERROR TOTAL
036742  013700  000042           $GET42: MOV     @#42,R0                 ;;GET MONITOR ADDRESS
036746  001405                           BEQ     $DOAGN                  ;;BRANCH IF NO MONITOR
036750  000005                           RESET                           ;;CLEAR THE WORLD
036752  004710                   $ENDAD: JSR     PC,(R0)                 ;;GO TO MONITOR
036754  000240                           NOP                             ;;SAVE ROOM
036756  000240                           NOP                             ;;FOR
036760  000240                           NOP                             ;;ACT11
036762                           $DOAGN:
036762  000137  005372                   JMP     @#LOOP                  ;;RETURN
036766     377     377     000   $ENULL: .BYTE   -1,-1,0                 ;;NULL CHARACTER STRING
        036772                           .EVEN
                                 ;;*******************************************************************
                                 .SBTTL  RP11/RP03 HANDLER
                                 ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
                                 ;;*******************************************************************
036772  104412                   RP3DRV: SAVREG
036774  105037  001676                   CLRB    @#RP3HSTA                       ;CLEAR DONE FLAG
037000  032737  000400  001676           BIT     #BIT8,@#RP3HSTA                 ;REPEAT FLAG SET?
037006  001403                           BEQ     8$                              ;BRANCH IF NO
037010  104414                           RESREG
037012  000137  040644                   JMP     @#RP3RPT
037016  013737  001556  001566   8$:     MOV     @#RNTBINX,@#RP311               ;SAVE RUN TABLE INDEX
037024  032737  000020  177570           BIT     #SW4,@#SWR                      ;INHIBIT RND DSK ADR?
037032  001403                           BEQ     1$                              ;BRANCH IF NO
037034  005000                           CLR     R0
037036  005001                           CLR     R1
037040  000410                           BR      4$
037042  004737  050604           1$:     JSR     PC,@#$RAND                      ;GO GET RANDOM NUMBER
037046  013700  001524                   MOV     @#$HINUM,R0                     ;GET HI NUMBER
037052  013701  001522                   MOV     @#$LONUM,R1                     ;GET LO NUMBER
037056  073027  177771                   ASHC    #-7,R0                          ;ADJUST TO FORM CYL ADR
037062  042700  177000           4$:     BIC     #177000,R0                      ;GET RID OF UNUSED BITS
037066  022700  000624                   CMP     #624,R0                         ;LEGAL CYL?
037072  100003                           BPL     5$                              ;BRANCH IF YES
037074  062700  000624                   ADD     #624,R0                         ;MAKE IT LEGAL
037100  000770                           BR      4$
037102  013702  001566           5$:     MOV     @#RP311,R2                      ;GET RUN TABLE INDEX
037106  016203  001626                   MOV     RUNTBL(R2),R3                   ;GET DEVICE ID
037112  042703  000777                   BIC     #777,R3                         ;ID ONLY
037116  050300                           BIS     R3,R0                           ;COMBINE WITH CYL ADR
037120  010062  001626                   MOV     R0,RUNTBL(R2)                   ;PUT BACK IN TABLE
037124  072127  177775                   ASH     #-3,R1                          ;GEN TRK-SECT ADR
037130  010103                           MOV     R1,R3                           ;SAVE
037132  042701  160377           6$:     BIC     #160377,R1                      ;GET RID OF ALL BUT TRK
037136  022701  011400                   CMP     #11400,R1                       ;LEGAL TRAK?
037142  100003                           BPL     2$                              ;BRANCH IF YES
037144  062701  011400                   ADD     #11400,R1                       ;MAKE IT LEGAL
```

```
037152  042703  177760          2$:      BIC     #177760,R3              ;GET SECTOR ADR
037156  022703  000011                   CMP     #11,R3                  ;IS IT LEGAL?
037162  100003                           BPL     3$                      ;BRANCH IF YES
037164  062703  000011                   ADD     #11,R3                  ;MAKE IT LEGAL
037170  000770                           BR      2$
037172  050301                  3$:      BIS     R3,R1                   ;COMBINE TRK-SECT
037174  010162  001642                   MOV     R1,RUNTRAK(R2)          ;PUT IN TABLE
037200  010037  002044                   MOV     R0,@#RP3HDC             ;SAVE DESIRED CYL
037204  010137  002112                   MOV     R1,@#RP3DA              ;SAVE DSK ADR
037210  112737  177775  002072           MOVB    #-3,@#RP3TRY            ;INIT TRY COUNT
037216  032737  000040  172516           BIT     #BIT5,@#MMR3            ;MAP ON?
037224  001405                           BEQ     7$                      ;BRANCH IF NO
037226  005046                           CLR     -(SP)                   ;PUT DEVICE ID ON STACK
037230  013746  001732                   MOV     @#RP3OLD,-(SP)          ;PUT ADR OF BUS ADR ON STK
037234  004737  052054                   JSR     PC,@#GETMAP             ;GET MAP REGISTER
037240  012737  000103  001564  7$:      MOV     #103,@#RP310            ;GET FUNCTION
037246  013700  001734                   MOV     @#RP3OLD+2,R0           ;GET BAE BITS
037252  072027  000004                   ASH     #4,R0                   ;SHIFT TO BITS 4 & 5
037256  050037  001564                   BIS     R0,@#RP310              ;COMBINE WITH FUNCTION
037262  010037  001734                   MOV     R0,@#RP3OLD+2
037266  013700  002012                   MOV     @#RP3UNIT,R0
037272  072027  000010                   ASH     #10,R0                  ;SHIFT UNIT NO TO RIGHT BITS
037276  050037  001564                   BIS     R0,@#RP310              ;COMBINE WITH FUNC & BAE
037302  010037  002012                   MOV     R0,@#RP3UNIT
037306  104414                           RESREG
037310  005777  142564          RP3WTRY: TST     @RP3DS                          ;IS DRIVE READY?
037314  100375                           BPL     RP3WTRY                         ;BRANCH IF NO
037316  053777  002012  142560           BIS     @#RP3UNIT,@RP3CS                ;SET UNIT BITS
037324  004737  037356                   JSR     PC,@#LDRP3              ;LOAD RP3 REGISTERS
037330  012777  040674  142560           MOV     #RP3SRV,@RP3VEC         ;SET VECTOR
037336  005077  142556                   CLR     @RP3PSW
037342  005037  002060                   CLR     @#RP3FUN                ;SET FUNCTION TO WRITE
037346  013777  001564  142530           MOV     @#RP310,@RP3CS          ;LOAD FUNCT AND GO
037354  000002                           RTI                             ;RETURN
037356  013777  002042  142526  LDRP3:   MOV     @#RP3HDA,@RP3DA         ;LOAD DSK ADR
037364  013777  002044  142522           MOV     @#RP3HDC,@RP3DC         ;LOAD CYL ADR
037372  013777  001716  142506           MOV     @#RP3HWC,@RP3WC         ;LOAD WORD COUNT
037400  013777  001732  142502           MOV     @#RP3OLD,@RP3BA         ;LOAD BUS ADR
037406  000207                           RTS     PC                      ;RETURN

                                ;;**************************************************************
                                .SBTTL   RK11/RK05 HANDLER
                                ;*       SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
                                ;;**************************************************************
037410  104412                  RKDRV:   SAVREG
037412  105037  001700                   CLRB    @#RKHSTAT               ;CLEAR DONE FLAG IN HANDLER STAT
037416  032737  000400  001700           BIT     #BIT8,@#RKHSTAT         ;REPEAT FLAG SET?
037424  001403                           BEQ     5$                      ;BRANCH IF NO
037426  104414                           RESREG
037430  000137  041462                   JMP     @#RKRPT
037434  013737  001556  001572  5$:      MOV     @#RNTBINX,@#RK11        ;SAVE RUN TABLE INDEX
037442  105037  001700                   CLRB    @#RKHSTAT               ;CLEAR DONE FLAG IN HANDLER STAT
037446  032737  000020  177570           BIT     #SW4,@#SWR              ;RANDOM DSK ADDRESS?
037454  001403                           BEQ     6$                      ;BRANCH IF YES
037456  005000                           CLR     R0                      ;CLEAR REGISTERS
```

```
 7018  037462  000404                         BR      7$                      ;FOR ADDRESS CHECKING
 7019  037464  004737  050604         6$:     JSR     PC,@#$RAND              ;GET RANDOM NUMBER
 7020  037470  013700  001524                 MOV     @#$HINUM,R0             ;GET HIGH NUMBER
 7021  037474  072027  177775         7$:     ASH     #-3,R0                  ;ADJUST TO FORM
 7022                                                                         ;CYLINDER ADDRESS
 7023  037500  010001                         MOV     R0,R1                   ;SAVE IN R1
 7024  037502  042701  160037         4$:     BIC     #160037,R1              ;GET RID OF SURF-SECT BITS
 7025  037506  022701  014300                 CMP     #14300,R1               ;IS IT A LEGAL CYLINDER?
 7026  037512  100003                         BPL     3$                      ;BRANCH IF YES
 7027  037514  062701  014340                 ADD     #14340,R1               ;ADD MAXIMUM CYLINDER
 7028  037520  000770                         BR      4$                      ;TRY AGAIN
 7029  037522  072127  177773         3$:     ASH     #-5,R1                  ;ADJUST CYLINDER ADDRESS
 7031  037532  016203  001626          MOV     RUNTBL(R2),R3          ;GET RUN TABLE ENTRY
 7032  037536  042703  000777          BIC     #777,R3                ;SAVE ID AND UNIT NO.
 7033  037542  050103                  BIS     R1,R3                  ;INSERT CYLINDER ADDR
 7034  037544  010362  001626          MOV     R3,RUNTBL(R2)          ;ENTER CYLINDER ADR IN RUN TABLE
 7035  037550  072027  177770          ASH     #-10,R0                ;GENER SECTOR-SURF ADDRESS
 7036  037554  042700  177740          BIC     #177740,R0             ;GET RID OF EXTRA BITS
 7037  037560  010003                  MOV     R0,R3                  ;SAVE
```

```
037566  022700  000012                            CMP     #12,R0                          ;IS SECTOR ADDRESS LEGAL?
037572  100004                                    BPL     1$                              ;BRANCH IF YES
037574  062700  000012                            ADD     #12,R0                          ;MAKE IT LEGAL
037600  042700  000020                            BIC     #BIT4,R0                        ;GET RID OF CARRY FROM ADD
037604  042703  000017                  1$:       BIC     #17,R3                          ;GET SURFACE ADDRESS
037610  050300                                    BIS     R3,R0                           ;GENER COMP SECT-SURF ADDRESS
037612  010062  001642                            MOV     R0,RUNTRAK(R2)                  ;SAVE IN RUN TRAK TABLE
037616  072127  000005                            ASH     #5,R1                           ;ADJUST CYLINDER ADDRESS
037622  050100                                    BIS     R1,R0                           ;CONCATINATE TRK & SECT ADDR
037624  013701  002014                            MOV     @#RKUNIT,R1                     ;GET UNIT NUMBER
037630  072127  000015                            ASH     #15,R1              ;ADJUST
037634  050100                                    BIS     R1,R0                           ;CONCATINATE UNIT,TRK,SURF,SECT
037636  010037  002046                            MOV     R0,@#RKHDA                      ;SAVE
037642  112737  177775  002073                    MOVB    #-3,@#RKTRY                     ;SET RETRY COUNT
037650  032737  000040  172516                    BIT     #BIT5,@#MMR3                    ;MAP ON?
037656  001406                                    BEQ     2$                              ;BRANCH IF NO
037660  012746  000001                            MOV     #1,-(SP)                        ;PUT DEVICE ID ON STACK
037664  012746  001736                            MOV     #RKOLD,-(SP)                    ;PUT ADDRESS OF ADR ON STACK
037670  004737  052054                            JSR     PC,@#GETMAP                     ;GET MAP REG
037674  012767  000103  141666  2$:       MOV     #103,RK10                       ;SET FUNCTION
037702  013700  001740                            MOV     @#RKOLD+2,R0                    ;GET BA EXTENDED
037706  072027  000004                            ASH     #4,R0                           ;ADJUST
037712  050037  001570                            BIS     R0,@#RK10                       ;PUT IN WITH FUNCTION
037716  010037  001740                            MOV     R0,@#RKOLD+2                    ;SAVE IN MEMORY
037722  104414                                    RESREG
037724  013777  002046  142202  RKWTRY:   MOV     @#RKHDA,@RKDA                   ;LOAD DISK ADDRESS
037732  032777  000100  142162            BIT     #BIT6,@RKDS                     ;UNIT READY?
037740  001774                                    BEQ     .-6                             ;BRANCH IF NO
037742  013777  001720  142160            MOV     @#RKHWC,@RKWC                   ;LOAD WORD COUNT
037750  013777  001736  142154            MOV     @#RKOLD,@RKBA                   ;LOAD BUS ADDRESS
037756  012777  041512  142152            MOV     #RKSRV,@RKVEC                   ;LOAD INTERRUPT VECTOR
037764  005077  142150                            CLR     @RKPSW
037770  005037  002062                            CLR     @#RKFUN                         ;SET FUNCTION TO WRITE
037774  013777  001570  142124            MOV     @#RK10,@RKCS                    ;LOAD FUNCTION AND GO
040002  000006                                    RTT                                     ;RETURN

                                ;;*************************************************************
                                .SBTTL    RH70/RP04 HANDLER
                                ;*        SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
                                ;;*************************************************************
040004  104412                          RP4DRV:   SAVREG
040006  105037  001706                            CLRB    @#RP4HSTA                       ;CLEAR DONE FLAG
040012  032737  000400  001706                    BIT     #BIT8,@#RP4HST                  ;REPEAT FLAG SET?
040020  001403                                    BEQ     6$                              ;BRANCH IF NO
040022  104414                                    RESREG
040024  000137  042340                            JMP     @#RP4RPT
040030  013737  001556  001574  6$:       MOV     @#RNTBINX,@#RP411               ;SAVE RUN TABLE INDEX
040036  105037  001706                            CLRB    @#RP4HSTA                       ;CLEAR DONE FLAG
040042  032737  000020  177570                    BIT     #SW4,@#SWR                      ;RANDOM DSK ADDRESS?
040050  001403                                    BEQ     1$                              ;BRANCH IF YES
040052  005000                                    CLR     R0
040054  005001                                    CLR     R1
040056  000410                                    BR      4$
040060  004737  050604                  1$:       JSR     PC,@#$RAND                      ;GET RANDOM NUMBER
040064  013700  001524                            MOV     @#$HINUM,R0                     ;GET HI NUMBER
```

```
040074  073027  177771                    ASHC    #-7,R0                          ;ADJUST TO FORM CYL. ADR.
040100  042700  177000            4$:     BIC     #177000,R0                      ;GET RID OF UNUSED BITS
040104  022700  000631                    CMP     #631,R0                         ;LEGAL CYLINDER
040110  100003                            BPL     5$                              ;BRANCH IF YES
040112  062700  000631                    ADD     #631,R0                         ;MAKE IT LEGAL
040116  000770                            BR      4$

040120  013702  001574            5$:     MOV     @#RP411,R2                      ;GET RUN TABLE INDEX
040124  016203  001626                    MOV     RUNTBL(R2),R3                   ;GET DEVICE ID
040130  042703  000777                    BIC     #777,R3                         ;SAVE ID ONLY
040134  050003                            BIS     R0,R3                           ;COMBINE WITH CYL ADR
040136  010362  001626                    MOV     R3,RUNTBL(R2)                   ;PUT IN RUN TABLE
040142  072127  177775                    ASH     #-3,R1                          ;GEN TRAK-SECT ADR
040146  042701  160340                    BIC     #160340,R1                      ;GET RID OF UNUSED BITS
040152  010103                            MOV     R1,R3                           ;SAVE
040154  042701  000037                    BIC     #37,R1                          ;GET RID OF SECT BITS
040160  022701  011000                    CMP     #11000,R1                       ;LEGAL TRAK?
040164  100004                            BPL     2$                              ;BRANCH IF YES
040166  062701  011000                    ADD     #11000,R1                       ;MAKE IT LEGAL
040172  042701  020000                    BIC     #BIT13,R1                       ;GET RID OF ADD CARRY
040176  042703  177740            2$:     BIC     #177740,R3                      ;GET SECTOR ADR
040202  022703  000025                    CMP     #25,R3                          ;LEGAL SECTOR
040206  100004                            BPL     3$                              ;BRANCH IF YES
040210  062703  000025                    ADD     #25,R3                          ;MAKE IT LEGAL
040214  042703  000040                    BIC     #BIT5,R3                        ;GET RID OF ADD CARRY
040220  050301                    3$:     BIS     R3,R1                           ;COMBINE TRAK-SECTOR
040222  010162  001642                    MOV     R1,RUNTRAK(R2)                  ;PUT TRAK-SECT IN TABLE
040226  010037  002054                    MOV     R0,@#RP4HDC                     ;SAVE CYLINDER ADR
040232  010137  002052                    MOV     R1,@#RP4HDA                     ;SAVE TRAK-SECTOR ADR
040236  112737  177775  002075            MOVB    #-3,@#RP4TRY                    ;SET TRY COUNT
040244  104414                            RESREG
040246  004767  000026            RP4WTRY:JSR     PC,LDRP4                        ;LOAD RP4 REGISTERS
040252  012777  042364  141716            MOV     #RP4SRV,@RP4VEC ;LOAD INTERRUPT VECTOR
040260  005077  141714                    CLR     @RP4PSW                         ;
040264  005037  002066                    CLR     @#RP4FUN                        ;SET FUNCTION TO WRITE
040270  112777  000161  141644            MOVB    #161,@RP4CS1                    ;LOAD FUNCTION AND GO
040276  000002                            RTI                                     ;RETURN

040300  013777  002022  141646    LDRP4:  MOV     @#RP4UNIT,@RP4CS2               ;LOAD UNIT NUMBER
040306  012777  010000  141660            MOV     #BIT12,@RP4OF                   ;SET FORMAT TO 16 BIT
040314  013777  002054  141642            MOV     @#RP4HDC,@RP4DC                 ;LOAD CYLINDER ADR
040322  013777  002052  141622            MOV     @#RP4HDA,@RP4DA                 ;LOAD TRAK-SECTOR
040330  013777  001726  141606            MOV     @#RP4HWC,@RP4WC                 ;LOAD WORD COUNT
040336  013777  001754  141604            MOV     @#RP4OLD+2,@RP4BAE              ;LOAD EXTENDED ADR BITS
040344  013777  001752  141574            MOV     @#RP4OLD,@RP4BA                 ;LOAD BUS ADR
040352  000207                            RTS     PC                              ;RETURN

                                  ;;*************************************************************
                                  .SBTTL  RH70/RS04 HANDLER
                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
                                  ;;*************************************************************
040354  104412                    RSDRV:  SAVREG
040356  105037  001710                    CLRB    @#RSHSTAT                       ;CLEAR DONE FLAG
040362  032737  000400  001710            BIT     #BIT8,@#RSHSTAT                 ;REPEAT FLAG SET?
040370  001403                            BEQ     3$                              ;BRANCH IF NO
```

```
040374  000137  043050                   JMP     @#RSRPT
040400  013737  001556  001576   3$:     MOV     @#RNTBINX,@#RS11          ;SAVE RUN TABLE INDEX
040406  032737  000020  177570           BIT     #SW4,@#SWR                ;RANDOM DSK ADR?
040414  001403                           BEQ     1$                        ;BRANCH IF YES
040416  005000                           CLR     R0
040420  005001                           CLR     R1
040422  000407                           BR      4$
040424  004737  050604           1$:     JSR     PC,@#$RAND                ;GET RANDOM NUMBER
040430  013700  001524                   MOV     @#$HINUM,R0
040434  072027  177774                   ASH     #-4,R0
040440  010001                           MOV     R0,R1                     ;SAVE RANDOM NUMBER
040442  042700  170077           4$:     BIC     #170077,R0                ;GET TRACK ADR
040446  022700  007600                   CMP     #7600,R0                  ;IS IT LEGAL?
040452  100003                           BPL     5$                        ;BRANCH IF YES
040454  062700  007600                   ADD     #7600,R0                  ;MAKE IT LEGAL
040460  000770                           BR      4$
040462  013702  001576           5$:     MOV     @#RS11,R2                 ;GET RUN TABLE INDEX
040466  072027  177772                   ASH     #-6,R0                    ;ADJUST TRACK ADR
040472  110062  001626                   MOVB    R0,RUNTBL(R2)             ;SAVE TRAK ADR IN RUN TBL
040476  042701  177700           6$:     BIC     #177700,R1                ;GET SECTOR ADR
040502  022701  000077                   CMP     #77,R1                    ;IS IT LEGAL?
040506  100003                           BPL     2$                        ;BRANCH IF YES
040510  062701  000077                   ADD     #77,R1                    ;MAKE IT LEGAL
040514  000770                           BR      6$
040516  010162  001642           2$:     MOV     R1,RUNTRAK(R2)            ;SAVE IN RUN TRAK TABLE
040522  072027  000006                   ASH     #6,R0                     ;ADJUST TRACK ADDR
040526  050100                           BIS     R1,R0                     ;COMBINE SECTOR TRAK
040530  010037  002056                   MOV     R0,@#RSHDA                ;SAVE AS DSK ADR
040534  112737  177775  002076           MOVB    #-3,@#RSTRY               ;SET TRY COUNT
040542  104414                           RESREG
040544  004737  040604           RSWTRY: JSR     PC,@#LDRS                 ;GO LOAD REGISTERS
040550  012777  043074  141446           MOV     #RSSRV,@RSVEC             ;SET INTERRUPT VECTOR
040556  005077  141444                   CLR     @RSPSW
040562  005037  002070                   CLR     @#RSFUN                   ;SET FUNCTION TO WRITE
040566  105777  141426           1$:     TSTB    @RSDS                     ;IS DRIVE READY?
040572  001775                           BEQ     1$                        ;BRANCH IF NO
040574  112777  000161  141400           MOVB    #161,@RSCS1               ;LOAD FUNCTION AND GO
040602  000002                           RTI

040604  013777  002024  141402   LDRS:   MOV     @#RSUNIT,@RSCS2           ;LOAD UNIT NUMBER
040612  013777  002056  141372           MOV     @#RSHDA,@RSDA             ;LOAD DSK ADR
040620  013777  001730  141356           MOV     @#RSHWC,@RSWC             ;LOAD WORD COUNT
040626  013777  001760  141354           MOV     @#RSOLD+2,@RSBAE          ;LOAD EXTENDED ADDRESS
040634  013777  001756  141344           MOV     @#RSOLD,@RSBA             ;LOAD BUS ADDRESS
040642  000207                           RTS     PC                        ;RETURN


                                 ;;*****************************************************************
                                 .SBTTL  RP11/RP03 SERVICE ROUTINE
                                 ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
                                 ;;*****************************************************************
040644  000005                   RP3RPT: RESET
040646  005337  002060                   DEC     @#RP3FUN                  ;RESTORE FUNCTION
040652  022737  000001  002060           CMP     #1,@#RP3FUN               ;WHAT IS IT?
040660  001472                           BEQ     RP31                      ;BRANCH IF WC
040662  002402                           BLT     1$                        ;BRANCH IF WRITE
```

```
040670  000167  000414          1$:     JMP     RP33
040674  005237  002060          RP3SRV: INC     @#RP3FUN                ;INCREMENT FUNCTION
040700  022737  000002  002060          CMP     #2,@#RP3FUN             ;WHAT IS IT?
040706  001501                          BEQ     RP3WCK                  ;BRANCH TO WRITE CHECK
040710  100002                          BPL     .+6
040712  000137  041350                  JMP     @#RP3READ

                                ;FUNCTION JUST EXECUTED WAS A WRITE
040716  032737  000400  001676          BIT     #BIT8,@#RP3HSTAT        ;REPEAT FLAG SET?
040724  001036                          BNE     RP3LOOP                 ;BRANCH IF YES
040726  005777  141152                  TST     @RP3CS                  ;ANY ERRORS?
040732  100045                          BPL     RP31                    ;BRANCH IF NO
040734  105737  002072                  TSTB    @#RP3TRY                ;TRIED 3 TIMES?
040740  001415                          BEQ     RP3ERR                  ;BRANCH IF YES
040742  112777  000001  141134          MOVB    #BIT0,@RP3CS            ;CLEAR THE DRIVE
040750  105777  141130                  TSTB    @RP3CS                  ;CONTROLLER READY?
040754  100375                          BPL     .-4                     ;BRANCH IF NO
040756  105237  002072                  INCB    @#RP3TRY                ;INCREMENT TRY COUNT
040762  013746  177776                  MOV     @#PSW,-(SP)             ;MAINTAIN SAME PSW
040766  012746  037310                  MOV     #RP3WTRY,-(SP)          ;SET RETRY ADDRESS
040772  000002                          RTI                             ;RETURN
040774  012737  100200  001676  RP3ERR: MOV     #100200,@#RP3HSTA       ;SET ERROR BIT IN HAND. STA
041002  010046                          MOV     R0,-(SP)                ;SAVE R0
041004  013700  001566                  MOV     @#RP311,R0              ;GET RUNTABLE INDEX
041010  052760  100000  001642          BIS     #BIT15,RUNTRAK(R0)      ;SET ERROR BIT
041016  012600                          MOV     (SP)+,R0                ;RESTORE R0
041020  000002                          RTI                             ;RETURN

041022  012737  100200  001676  RP3LOOP:MOV     #100200,@#RP3HSTAT      ;SET DONE AND ERROR
041030  005777  141050                  TST     @RP3CS                  ;ANY ERRORS?
041034  100403                          BMI     1$                      ;BRANCH IF YES
041036  042737  100000  001676          BIC     #BIT15,@#RP3HSTAT       ;CLEAR ERROR BIT
041044  000002                  1$:     RTI                             ;RETURN
                                ;WRITE WAS OK- NOW DO A WRITE CHECK
041046  112737  177775  002072  RP31:   MOVB    #-3,@#RP3TRY            ;INIT TRY COUNT
041054  012737  000107  001564          MOV     #107,@#RP310            ;SET FUNCTION
041062  053737  001734  001564          BIS     @#RP3OLD+2,@#RP310      ;SET BAE BITS
041070  053737  002012  001564          BIS     @#RP3UNIT,@#RP310       ;SET UNIT BITS
041076  004737  037356          RP32:   JSR     PC,@#LDRP3              ;LOAD RP3 REGISTERS
041102  013777  001564  140774          MOV     @#RP310,@RP3CS          ;LOAD FUNCTION AND GO
041110  000002                          RTI                             ;RETURN

                                ;FUNCTION JUST EXECUTED WAS A WRITE CHECK
041112  032737  000400  001676  RP3WCK: BIT     #BIT8,@#RP3HSTAT                ;REPEAT FLAG SET?
041120  001340                          BNE     RP3LOOP                 ;BRANCH IF YES
041122  005777  140756                  TST     @RP3CS                  ;ANY ERRORS?
041126  100031                          BPL     1$                      ;BRANCH IF NO
041130  005737  001566                  TST     @#RP311                 ;FIRST 2K?
041134  001422                          BEQ     4$                      ;BRANCH IF YES
041136  105737  002072          5$:     TSTB    @#RP3TRY                ;TRIED 3 TIMES?
041142  001714                          BEQ     RP3ERR                  ;BRANCH IF YES
041144  005337  002060                  DEC     @#RP3FUN                ;RESTORE FUNCTION
041150  112777  000001  140726          MOVB    #BIT0,@RP3CS            ;CLEAR THE DRIVE
041156  105777  140722                  TSTB    @RP3CS                  ;CONTROLLER READY?
041162  100375                          BPL     .-4                     ;BRANCH IF NO
```

```
041170  013746  177776                  MOV     @#PSW,-(SP)
041174  012746  041076                  MOV     #RP32,-(SP)
041200  000002                          RTI                                     ;GO TRY AGAIN
041202  032777  000010  140672  4$:     BIT     #BIT3,@RP3ER                    ;WRITE CHECK ERROR?
041210  001752                          BEQ     5$                              ;BRANCH IF NO


                                ;WRITE CHECK OK- NOW DO A READ
041212  112737  177775  002072  1$:     MOVB    #-3,@#RP3TRY                    ;RESTORE TRY COUNT
041220  032737  000040  172516          BIT     #BIT5,@#MMR3                    ;MAP ON?
041226  001407                          BEQ     2$                              ;BRANCH IF NO
041230  005046                          CLR     -(SP)                           ;PUT DEVICE ID ON STACK
041232  004737  052320                  JSR     PC,@#GIVEMAP                    ;RETURN MAP REGISTER
041236  012746  001762                  MOV     #RP3NWL,-(SP)                   ;PUT ADR OF BUS ADR ON STK
041242  004737  052054                  JSR     PC,@#GETMAP                     ;GET MAP REGISTERS
041246  010046                  2$:     MOV     R0,-(SP)                        ;SAVE R0
041250  013700  001764                  MOV     @#RP3NWH,R0                     ;GET BAE BITS
041254  072027  000004                  ASH     #4,R0                           ;ADJUST
041260  010037  001764                  MOV     R0,@#RP3NWH                     ;SAVE
041264  012600                          MOV     (SP)+,R0                        ;RESTORE R0
041266  012737  000105  001564          MOV     #105,@#RP310                    ;SET FUNCTION
041274  053737  001764  001564          BIS     @#RP3NWH,@#RP310                ;SET BAE BITS
041302  053737  002012  001564          BIS     @#RP3UNIT,@#RP310               ;SET UNIT NUMBER
041310  013777  002042  140574  RP33:   MOV     @#RP3HDA,@RP3DA                 ;LOAD DSK ADR
041316  013777  002044  140570          MOV     @#RP3HDC,@RP3DC                 ;LOAD CYL
041324  013777  001716  140554          MOV     @#RP3HWC,@RP3WC                 ;LOAD WORD COUNT
041332  013777  001762  140550          MOV     @#RP3NWL,@RP3BA                 ;LOAD BUS ADR
041340  013777  001564  140536          MOV     @#RP310,@RP3CS                  ;LOAD FUNCTION AND GO
041346  000002                          RTI                                     ;RETURN


                                ;FUNCTION JUST EXECUTED WAS A READ
041350  032737  000400  001676  RP3READ:BIT     #BIT8,@#RP3HSTAT                        ;REPEAT FLAG SET?
041356  001221                          BNE     RP3LOOP                         ;BRANCH IF YES
041360  005777  140520                  TST     @RP3CS                          ;ANY ERRORS?
041364  100022                          BPL     1$                              ;BRANCH IF NO
041366  105737  002072                  TSTB    @#RP3TRY                        ;TRIED 3 TIMES?
041372  001600                          BEQ     RP3ERR                          ;BRANCH IF YES
041374  005337  002060                  DEC     @#RP3FUN                        ;RESTORE FUNCTION
041400  112777  000001  140476          MOVB    #BIT0,@RP3CS                    ;CLEAR THE DRIVE
041406  105777  140472                  TSTB    @RP3CS                          ;CONTROLLER READY?
041412  100375                          BPL     .-4                             ;BRANCH OF NO
041414  105237  002072                  INCB    @#RP3TRY                        ;INCREMENT TRY COUNT
041420  013746  177776                  MOV     @#PSW,-(SP)
041424  012746  041310                  MOV     #RP33,-(SP)
041430  000002                          RTI                                     ;GO TRY AGAIN
041432  032737  000040  172516  1$:     BIT     #BIT5,@#MMR3                    ;MAP ON?
041440  001404                          BEQ     2$                              ;BRANCH IF NO
041442  005046                          CLR     -(SP)                           ;PUT DEVICE ID IN STK
041444  004737  052320                  JSR     PC,@#GIVEMAP                    ;RETURN MAP REGISTERS
041450  005726                          TST     (SP)+                           ;RESTORE STACK
041452  112737  000200  001676  2$:     MOVB    #200,@#RP3HSTA                  ;SET DONE FLAG
041460  000002                          RTI                                     ;RETURN
                                ;;*************************************************************
                                .SBTTL  RK11/RK05 SERVICE ROUTINE
                                ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
                                ;;*************************************************************
```

```
041464  005337  002062                   DEC     @#RKFUN                 ;RESTORE FUNCTION
041470  022737  000001  002062           CMP     #1,@#RKFUN              ;WHAT IS IT?
041476  001475                           BEQ     RK1                     ;BRANCH IF WC
041500  002402                           BLT     1$                      ;BRANCH IF WRITE
041502  000137  037724                   JMP     @#RKWTRY                ;IT WAS A WRITE
041506  000137  042144           1$:     JMP     @#RK3
041512  062737  000001  002062   RKSRV:  ADD     #1,@#RKFUN              ;FIND OUT WHAT FUNCTION
                                                                         ;WAS EXECUTED
041520  022737  000002  002062           CMP     #2,@#RKFUN              ;WAS IT A WRITE CHECK?
041526  001507                           BEQ     RKWRCK                  ;BRANCH IF YES
041530  100002                           BPL     .+6                     ;BRANCH IF IT WAS A WRITE
041532  000137  042176                   JMP     @#RKREAD

                                 ;FUNCTION JUST EXECUTED WAS A WRITE. ANY ERRORS?
041536  032737  000400  001700           BIT     #BIT8,@#RKHSTAT         ;REPEAT FLAG SET?
041544  001040                           BNE     RKLOOP                  ;BRANCH IF YES
041546  005777  140354                   TST     @RKCS                   ;ANY ERRORS?
041552  100047                           BPL     RK1                     ;BRANCH IF NO
041554  105737  002073                   TSTB    @#RKTRY                 ;TRYED 3 TIMES?
041560  001417                           BEQ     RKERR                   ;BRANCH IF YES
041562  012777  000001  140336           MOV     #1,@RKCS                ;CLEAR THE ERROR
041570  004737  042322                   JSR     PC,@#TIMER              ;WAIT A LITTLE
041574  105777  140326                   TSTB    @RKCS                   ;WAIT FOR CONT CLR TO FINISH
041600  100375                           BPL     .-4
041602  105237  002073                   INCB    @#RKTRY                 ;INCREMENT TRY COUNT
041606  013746  177776                   MOV     @#PSW,-(SP)
041612  012746  037724                   MOV     #RKWTRY,-(SP)
041616  000002                           RTI
041620  012737  100200  001700   RKERR:  MOV     #100200,@#RKHSTAT       ;SET ERROR & DONE FLAG
041626  010046                           MOV     R0,-(SP)                ;SAVE R0
041630  013700  001572                   MOV     @#RK11,R0               ;GET SAVED RUN TABLE INDEX
041634  052760  100000  001642           BIS     #BIT15,RUNTRAK(R0)      ;SET ERROR BIT IN RUN TABLE
041642  012600                           MOV     (SP)+,R0                ;RESTORE R0
041644  000002                           RTI                             ;RETURN

041646  012737  100200  001700   RKLOOP:MOV      #100200,@#RKHSTAT       ;SET DONE AND ERROR BITS
041654  005777  140246                   TST     @RKCS                   ;ANY ERRORS?
041660  100403                           BMI     1$                      ;BRANCH IF YES
041662  042737  100000  001700           BIC     #BIT15,@#RKHSTAT        ;CLEAR ERROR BIT
041670  000002                   1$:     RTI                             ;RETURN
                                 ;WRITE WAS OK, NOW DO A WRITE CHECK
041672  112737  177775  002073   RK1:    MOVB    #-3,@#RKTRY             ;RESTORE TRY COUNT
041700  012767  000507  137662           MOV     #507,RK10               ;SET FUNCTION TO WRITE
041706  053767  001740  137654           BIS     @#RKOLD+2,RK10          ;SET BA EXT BITS
041714  013777  002046  140212   RK2:    MOV     @#RKHDA,@RKDA           ;LOAD DISK ADDRESS
041722  013777  001720  140200           MOV     @#RKHWC,@RKWC           ;LOAD WORD COUNT
041730  013777  001736  140174           MOV     @#RKOLD,@RKBA           ;LOAD BUS ADDRESS
041736  016777  137626  140162           MOV     RK10,@RKCS              ;START FUNCTION
041744  000002                           RTI                             ;RETURN

                                 ;FUNCTION JUST EXECUTED WAS A WRITE CHECK. ANY ERRORS?
041746  032737  000400  001700   RKWRCK:BIT      #BIT8,@#RKHSTAT                 ;REPEAT FLAG SET?
041754  001334                           BNE     RKLOOP                  ;BRANCH IF YES
041756  005777  140144                   TST     @RKCS                   ;ANY ERRORS?
041762  100033                           BPL     1$                      ;BRANCH IF NO
```

```
041770  001424                           BEQ     4$                              ;BRANCH IF YES
041772  105737  002073           5$:     TSTB    @#RKTRY                         ;TRYED 3 TIMES?
041776  001710                           BEQ     RKERR                           ;BRANCH IF YES
042000  005337  002062                   DEC     @#RKFUN                         ;SET FUNCTION BACK TO WC
042004  012777  000001  140114           MOV     #1,@RKCS                        ;CLEAR THE ERROR
042012  004737  042322                   JSR     PC,@#TIMER                      ;WAIT A LITTLE
042016  105777  140104                   TSTB    @RKCS                           ;WAIT FOR CLR TO FINISH
042022  100375                           BPL     .-4
042024  105237  002073                   INCB    @#RKTRY                         ;INCREMENT TRY COUNT
042030  013746  177776                   MOV     @#PSW,-(SP)
042034  012746  041714                   MOV     #RK2,-(SP)
042040  000002                           RTI
042042  032777  040000  140056   4$:     BIT     #BIT14,@RKCS                    ;HARD ERROR?
042050  001350                           BNE     5$                              ;BRANCH IF YES


                                 ;WRITE CHECK WAS OK, NOW DO A READ.
042052  112737  177775  002073   1$:     MOVB    #-3,@#RKTRY                     ;RESTORE TRY COUNT
042060  032737  000040  172516           BIT     #BIT5,@#MMR3                    ;MAP ON?
042066  001410                           BEQ     2$                              ;BRANCH IF NO
042070  012746  000001                   MOV     #1,-(SP)                        ;PUT DEVICE ID ON STACK
042074  004767  010220                   JSR     PC,GIVEMAP                      ;RELINQUISH MAP REG
042100  012746  001766                   MOV     #RKNEWL,-(SP)                   ;PUT ADR OF BADR ON STACK
042104  004737  052054                   JSR     PC,@#GETMAP                     ;GET MAPREGISTER
042110  010046                   2$:     MOV     R0,-(SP)                        ;SAVE R0
042112  013700  001770                   MOV     @#RKNEWH,R0                     ;GET BA EXT
042116  072027  000004                   ASH     #4,R0                           ;ADJUST
042122  010037  001770                   MOV     R0,@#RKNEWH                     ;SAVE
042126  012600                           MOV     (SP)+,R0                        ;RESTORE R0
042130  012767  000105  137432           MOV     #105,RK10                       ;SET FUNCTION
042136  053767  001770  137424           BIS     @#RKNEWH,RK10                   ;SET BA EXT BITS IN FUNCTION
042144  013777  002046  137762   RK3:    MOV     @#RKHDA,@RKDA                   ;LOAD DISK ADDRESS
042152  013777  001720  137750           MOV     @#RKHWC,@RKWC                   ;LOAD WORD COUNT
042160  013777  001766  137744           MOV     @#RKNEWL,@RKBA                  ;LOAD BUS ADDRESS
042166  016777  137376  137732           MOV     RK10,@RKCS                      ;LOAD FUNCTION AND GO
042174  000002                           RTI                                     ;RETURN


                                 ;FUNCTION JUST EXECUTED WAS A READ. ANY ERRORS?
042176  032737  000400  001700   RKREAD: BIT     #BIT8,@#RKHSTAT                         ;REPEAT FLAG SET?
042204  001220                           BNE     RKLOOP                          ;BRANCH IF YES
042206  005777  137714                   TST     @RKCS                           ;ANY ERRORS?
042212  100026                           BPL     1$                              ;BRANCH IF NO
042214  105737  002073                   TSTB    @#RKTRY                         ;TRYED 3 TIMES?
042220  001002                           BNE     3$                              ;BRANCH IF NO
042222  000167  177372                   JMP     RKERR
042226  005337  002062           3$:     DEC     @#RKFUN                         ;SET FUNCTION BACK TO READ
042232  012777  000001  137666           MOV     #1,@RKCS                        ;CLEAR THE ERROR
042240  004737  042322                   JSR     PC,@#TIMER                      ;WAIT A LITTLE
042244  105777  137656                   TSTB    @RKCS                           ;WAIT FOR CLR TO FINISH
042250  100375                           BPL     .-4
042252  105237  002073                   INCB    @#RKTRY                         ;INCREMENT TRY COUNT
042256  013746  177776                   MOV     @#PSW,-(SP)
042262  012746  042144                   MOV     #RK3,-(SP)
042266  000002                           RTI
042270  032737  000040  172516   1$:     BIT     #BIT5,@#MMR3                    ;MAP ON?
042276  001405                           BEQ     2$                              ;BRANCH IF NO
```

```
042304  004737  052320                  JSR     PC,@#GIVEMAP            ;RELINQUISH MAP REGSTER
042310  005726                          TST     (SP)+                   ;POP THE STACK
042312  112737  000200  001700  2$:     MOVB    #200,@#RKHSTAT          ;SET DON E FLAG
042320  000002                          RTI                             ;RETURN
042322  005067  000010          TIMER:  CLR     1$
042326  105267  000004          2$:     INCB    1$
042332  001375                          BNE     2$
042334  000207                          RTS     PC
042336  000000                  1$:     .WORD

                                ;;****************************************************************
                                .SBTTL  RH70/RP04 SERVICE ROUTINE
                                ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
                                ;;****************************************************************
042340  000005                  RP4RPT: RESET
042342  005337  002066                  DEC     @#RP4FUN                ;RESTORE FUNCTION
042346  022737  000001  002066          CMP     #1,@#RP4FUN             ;WHAT IS IT?
042354  001501                          BEQ     RP41                    ;BRANCH IF WC
042356  002560                          BLT     RP43                    ;BRANCH IF READ
042360  000137  040246                  JMP     @#RP4WTRY               ;GO TO WRITE
042364  005237  002066          RP4SRV: INC     @#RP4FUN                ;FIND OUT WHAT FUNCTION
042370  022737  000002  002066          CMP     #2,@#RP4FUN             ;WAS JUST EXECUTED
042376  001504                          BEQ     RP4WCK
042400  100566                          BMI     RP4READ


                                ;WRITE FUNCTION WAS JUST EXECUTED.
042402  032737  000400  001706          BIT     #BIT8,@#RP4HSTAT        ;REPEAT FLAG SET?
042410  001050                          BNE     RP4LOOP                 ;BRANCH IF YES
042412  032777  040000  137540          BIT     #BIT14,@RP4DS           ;ANY ERRORS
042420  001457                          BEQ     RP41                    ;BRANCH IF NO
042422  105737  002075                  TSTB    @#RP4TRY                ;TRIED 3 TIMES?
042426  001426                          BEQ     RP4ERR                  ;BRANCH IF YES
042430  052777  000040  137516          BIS     #BIT5,@RP4CS2           ;CLEAR ALL ERRORS
042436  004737  040300                  JSR     PC,@#LDRP4              ;RELOAD THE UNIT NO
042442  105237  002075                  INCB    @#RP4TRY                ;INCREMENT TRY COUNT
042446  013746  177776                  MOV     @#PSW,-(SP)             ;SETUP THE STACK TO
042452  012746  040246                  MOV     #RP4WTRY,-(SP)          ;TRY WRITE AGAIN
042456  032737  000400  001706          BIT     #BIT8,@#RP4HSTAT        ;REPEAT FLAG SET?
042464  001006                          BNE     2$                      ;BRANCH IF YES
042466  012777  000007  137446          MOV     #7,@RP4CS1              ;RECALIBRATE
042474  105777  137460          1$:     TSTB    @RP4DS                  ;DRIVE READY?
042500  100375                          BPL     1$                      ;BRANCH IF NO
042502  000002                  2$:     RTI
042504  012737  100200  001706  RP4ERR: MOV     #100200,@#RP4HSTA       ;SET ERROR & DONE BIT
042512  010046                          MOV     R0,-(SP)                ;SAVE R0
042514  013700  001574                  MOV     @#RP411,R0              ;GET RUN TABLE INDEX
042520  052760  100000  001642          BIS     #BIT15,RUNTRAK(R0)      ;SET ERROR BIT
042526  012600                          MOV     (SP)+,R0                ;RESTORE R0
042530  000002                          RTI                             ;RETURN

042532  012737  100200  001706  RP4LOOP:MOV     #100200,@#RP4HSTAT      ;SET DONE AND ERROR BITS
042540  032777  040000  137412          BIT     #BIT14,@RP4DS           ;ANY ERRORS?
042546  001003                          BNE     1$                              ;BRANCH IF YES
042550  042737  100000  001706          BIC     #BIT15,@#RP4HSTAT       ;CLEAR ERROR BIT
042556  000002                  1$:     RTI                             ;RETURN
```

```
042560  112737  177775  002075  RP41:   MOVB    #-3,@#RP4TRY                    ;INITIALIZE TRY COUNT
042566  105777  137366          RP42:   TSTB    @RP4DS                          ;IS DRIVE READY?
042572  001775                          BEQ     RP42                            ;BRANCH IF NO
042574  004737  040300                  JSR     PC,@#LDRP4
042600  112777  000151  137334          MOVB    #151,@RP4CS1                    ;LOAD FUNCTION AND GO
042606  000002                          RTI

                                ;FUNCTION JUST EXECUTED WAS A WRITE CHECK
042610  032737  000400  001706  RP4WCK: BIT     #BIT8,@#RP4HSTAT                   ;REPEAT FLAG SET?
042616  001345                          BNE     RP4LOOP                         ;BRANCH IF YES
042620  032777  040000  137332          BIT     #BIT14,@RP4DS                   ;ANY ERRORS?
042626  001421                          BEQ     1$                              ;BRANCH IF NO
042630  105737  002075          3$:     TSTB    @#RP4TRY                        ;TRIED 3 TIMES?
042634  001723                          BEQ     RP4ERR                          ;BRANCH IF YES
042636  005337  002062                  DEC     @#RKFUN                         ;SET FUNCTION TO WC
042642  052777  000040  137304          BIS     #BIT5,@RP4CS2                   ;CLEAR ALL ERRORS
042650  004737  040300                  JSR     PC,@#LDRP4                      ;RELOAD THE UNIT NO
042654  105237  002075                  INCB    @#RP4TRY                        ;INCREMENT TRY COUNT
042660  013746  177776                  MOV     @#PSW,-(SP)
042664  012746  042566                  MOV     #RP42,-(SP)
042670  000002                          RTI                                     ;TRY AGAIN
042672  032777  040000  137254  1$:     BIT     #BIT14,@RP4CS2                  ;WRITE CHECK ERROR?
042700  001404                          BEQ     2$                              ;BRANCH IF NO
042702  005737  001574                  TST     @#RP411                         ;FIRST 2K?
042706  001401                          BEQ     2$
042710  000747                          BR      3$

                                ;WRITE CHECK WAS OK...NOW DO A READ.
042712  112737  177775  002075  2$:     MOVB    #-3,@#RP4TRY                    ;INITIALIZE TRY COUNT
042720  105777  137234          RP43:   TSTB    @RP4DS                          ;IS DRIVE READY?
042724  001775                          BEQ     RP43                            ;BRANCH IF NO
042726  004737  040300                  JSR     PC,@#LDRP4                      ;LOAD REGISTERS
042732  013777  002004  137210          MOV     @#RP4NWH,@RP4BAE                ;LOAD EXTENDED ADR BITS
042740  013777  002002  137200          MOV     @#RP4NWL,@RP4BA                 ;LOAD BUS ADR
042746  112777  000171  137166          MOVB    #171,@RP4CS1                    ;LOAD FUNCTION AND GO
042754  000002                          RTI                                     ;RETURN

                                ;FUNCTION JUST EXECUTED WAS A READ.
042756  032737  000400  001706  RP4READ:BIT     #BIT8,@#RP4HSTAT                ;REPEAT FLAG SET?
042764  001262                          BNE     RP4LOOP                         ;BRANCH IF YES
042766  032777  040000  137164          BIT     #BIT14,@RP4DS                   ;ANY ERRORS?
042774  001421                          BEQ     1$                              ;BRANCH IF NO
042776  105737  002075                  TSTB    @#RP4TRY                        ;TRIED 3 TIMES?
043002  001640                          BEQ     RP4ERR                          ;BRANCH IF YES
043004  005337  002066                  DEC     @#RP4FUN                        ;SET FUNCTION TO A READ
043010  052777  000040  137136          BIS     #BIT5,@RP4CS2                   ;CLEAR ALL ERRORS
043016  004737  040300                  JSR     PC,@#LDRP4                      ;RELOAD THE UNIT NO
043022  105237  002075                  INCB    @#RP4TRY                        ;INCREMENT TRY COUNT
043026  013746  177776                  MOV     @#PSW,-(SP)
043032  012746  042720                  MOV     #RP43,-(SP)
043036  000002                          RTI                                     ;TRY AGAIN
043040  112737  000200  001706  1$:     MOVB    #200,@#RP4HSTA                  ;SET DONE FLAG
043046  000002                          RTI                                     ;RETURN
                                ;;*******************************************************************
                                .SBTTL  RH70/RS04 SERVICE ROUTINE
```

```
                                        ;;*************************************************************
043050  000005                          RSRPT:  RESET
043052  005337  002070                          DEC     @#RSFUN                 ;RESTORE FUNCTION
043056  022737  000001  002070                  CMP     #1,@#RSFUN              ;WHAT IS IT?
043064  001465                                  BEQ     RS41                    ;BRANCH IF WC
043066  002542                                  BLT     RS43                    ;BRANCH IF WRITE
043070  000137  040544                          JMP     @#RSWTRY
043074  005237  002070                  RSSRV:  INC     @#RSFUN                 ;FIND OUT WHAT FUNCTION
043100  022737  000002  002070                  CMP     #2,@#RSFUN              ;WAS JUST EXECUTED
043106  001470                                  BEQ     RSWCK
043110  100550                                  BMI     RSREAD

                                        ;WRITE FUNCTION WAS JUST EXECUTED
043112  032737  000400  001710                  BIT     #BIT8,@#RSHSTAT ;REPEAT FLAG SET?
043120  001034                                  BNE     RSLOOP                  ;BRANCH IF YES
043122  032777  040000  137070                  BIT     #BIT14,@RSDS            ;ANY ERRORS?
043130  001443                                  BEQ     RS41                    ;BRANCH IF NO
043132  105737  002076                          TSTB    @#RSTRY                 ;TRIED 3 TIMES?
043136  001412                                  BEQ     RSERR                   ;BRANCH IF YES
043140  052777  000040  137046                  BIS     #BIT5,@RSCS2            ;CLEAR ALL ERRORS
043146  105237  002076                          INCB    @#RSTRY                 ;INCREMENT TRY COUNT
043152  013746  177776                          MOV     @#PSW,-(SP)             ;SETUP THE STACK TO
043156  012746  040544                          MOV     #RSWTRY,-(SP)           ;TRY THE WRITE AGAIN
043162  000002                                  RTI
043164  012737  100200  001710          RSERR:  MOV     #100200,@#RSHSTAT       ;SET ERROR AND DONE BIT
043172  010046                                  MOV     R0,-(SP)                ;SAVE R0
043174  013700  001576                          MOV     @#RS11,R0               ;GET RUN TBL INDEX
043200  052760  100000  001642                  BIS     #BIT15,RUNTRAK(R0)      ;SET ERROR BIT
043206  012600                                  MOV     (SP)+,R0                ;RESTORE R0
043210  000002                                  RTI

043212  012737  100200  001710          RSLOOP: MOV     #100200,@#RSHSTAT       ;SET DONE AND ERROR BITS
043220  032777  040000  136772                  BIT     #BIT14,@RSDS            ;ANY ERRORS?
043226  001003                                  BNE     1$                      ;BRANCH IF YES
043230  042737  100000  001710                  BIC     #BIT15,@#RSHSTAT        ;CLEAR ERROR BIT
043236  000002                          1$:     RTI                             ;RETURN
                                        ;WRITE OK...NOW DO A WRITE CHECK
043240  112737  177775  002076          RS41:   MOVB    #-3,@#RSTRY             ;INIT TRY COUNT
043246  105777  136746                  RS42:   TSTB    @RSDS                   ;IS DRIVE READY?
043252  001775                                  BEQ     RS42                    ;BRANCH IF NO
043254  004737  040604                          JSR     PC,@#LDRS               ;LOAD RS REGISTERS
043260  112777  000151  136714                  MOVB    #151,@RSCS1             ;LOAD FUNCTION AND GO
043266  000002                                  RTI                             ;RETURN

                                        ;FUNCTION JUST EXECUTED WAS A WRITE CHECK
043270  032737  000400  001710          RSWCK:  BIT     #BIT8,@#RSHSTAT         ;REPEAT FLAG SET?
043276  001345                                  BNE     RSLOOP                  ;BRANCH IF YES
043300  032777  040000  136712                  BIT     #BIT14,@RSDS            ;ANY ERRORS?
043306  001417                                  BEQ     1$                      ;BRANCH IF NO
043310  105737  002076                  3$:     TSTB    @#RSTRY                 ;TRIED 3 TIMES?
043314  001723                                  BEQ     RSERR                   ;BRANCH IF YES
043316  005337  002070                          DEC     @#RSFUN                 ;SET FUNCTION BACK TO WC
043322  052777  000040  136664                  BIS     #BIT5,@RSCS2            ;CLEAR THE ERROR
043330  105237  002076                          INCB    @#RSTRY                 ;INCREMENT THE TRY COUNT
043334  013746  177776                          MOV     @#PSW,-(SP)
```

```
043344  000002                                  RTI                                     ;TRY AGAIN

043346  032777  040000  136640  1$:             BIT     #BIT14,@RSCS2                   ;WRITE CHECK ERROR?
043354  001404                                  BEQ     2$                              ;BRANCH IF NO
043356  005737  001576                          TST     @#RS11                          ;FIRST 2K?
043362  001401                                  BEQ     2$                              ;BRANCH IF YES
043364  000751                                  BR      3$

                                        ;WRITE CHECK WAS OK...NOW DO A READ.
043366  112737  177775  002076  2$:             MOVB    #-3,@#RSTRY                     ;INIT TRY COUNT
043374  105777  136620          RS43:           TSTB    @RSDS                           ;IS DRIVE READY?
043400  001775                                  BEQ     RS43                            ;BRANCH IF NO
043402  004737  040604                          JSR     PC,@#LDRS                       ;LOAD RS REGISTERS
043406  013777  002010  136574                  MOV     @#RSNEWH,@RSBAE                 ;LOAD BAE
043414  013777  002006  136564                  MOV     @#RSNEWL,@RSBA                  ;LOAD BUS ADR
043422  112777  000171  136552                  MOVB    #171,@RSCS1                     ;LOAD FUNCTION AND GO
043430  000002                                  RTI                                     ;RETURN

                                        ;FUNCTION JUST EXECUTED WAS A READ.
043432  032737  000400  001710  RSREAD: BIT     #BIT8,@#RSHSTAT                 ;REPEAT FLAG SET?
043440  001264                                  BNE     RSLOOP                          ;BRANCH IF YES
043442  032777  040000  136550                  BIT     #BIT14,@RSDS                    ;ANY ERRORS?
043450  001417                                  BEQ     1$                              ;BRANCH IF NO
043452  105737  002076                          TSTB    @#RSTRY                         ;TRIED 3 TIMES?
043456  001642                                  BEQ     RSERR                           ;BRANCH IF YES
043460  005337  002070                          DEC     @#RSFUN                         ;RESTORE FUN TO READ
043464  052777  000040  136522                  BIS     #BIT5,@RSCS2                    ;CLEAR ALL ERRORS
043472  105237  002076                          INCB    @#RSTRY                         ;INCREMENT TRY COUNT
043476  013746  177776                          MOV     @#PSW,-(SP)
043502  012746  043374                          MOV     #RS43,-(SP)
043506  000002                                  RTI                                     ;TRY AGAIN
043510  112737  000200  001710  1$:             MOVB    #200,@#RSHSTAT                  ;SET DONE FLAG
043516  000002                                  RTI                                     ;RETURN
                                        ;;*************************************************************************
                                        .SBTTL  UNIBUS EXERCISER SERVICE ROUTINE
                                        ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
                                        ;;*************************************************************************
043520  104412                          UBESRV: SAVREG
043522  004737  053130                          JSR     PC,@#LDKT                       ;GO TO LOW CORE
043526  012704  002236                          MOV     #UBETBL+6,R4                    ;GET ADDRESS OF UBECR1
043532  005774  000000                          TST     @(R4)                           ;WAS THERE AN ERROR?
043536  100440                                  BMI     UBE2                            ;BRANCH IF YES
043540  012746  000003                          MOV     #3,-(SP)                        ;PUT DEVICE ID IN STACK
043544  004737  052320                          JSR     PC,@#GIVEMAP                    ;GIVE UP MAP REG
043550  062737  000776  001662                  ADD     #776,@#UBESAV                   ;INCREMENT UBE BUS ADR
043556  005537  001664                          ADC     @#UBESAV+2
043562  013737  001662  001666                  MOV     @#UBESAV,@#UBEADR
043570  013737  001664  001670                  MOV     @#UBESAV+2,@#UBEADR+2
043576  012746  001666                          MOV     #UBEADR,-(SP)
043602  004737  052054                          JSR     PC,@#GETMAP
043606  013754  001670                          MOV     @#UBEADR+2,@-(R4)               ;LOAD UBECR2
043612  013754  001666                          MOV     @#UBEADR,@-(R4)                 ;LOAD UBEBA
043616  012754  170000                          MOV     #170000,@-(R4)                  ;LOAD UBECC
043622  004737  053216                          JSR     PC,@#RESKT                      ;GO BACK TO ORIGINAL CORE
043626  104414                                  RESREG
```

```
043636  000002                                  RTI                                     ;RETURN

                                        ;UBE ERROR-IS IT LAST MEMORY?
043640  005037  001276                  UBE2:   CLR     @#STMP0
043644  162704  000004                          SUB     #4,R4                           ;ADJUST R4
043650  017403  000002                          MOV     @2(R4),R3                       ;GET BECR2
043654  042703  000003                          BIC     #3,R3                           ;GET RID OF ADDRESS BITS
043660  022703  000400                          CMP     #400,R3                         ;WAS ERROR A TIMEOUT?
043664  001050                                  BNE     UBEERR                          ;BRANCH IF NO
043666  017437  000000  001672                  MOV     @(R4),@#ERRBA                   ;SAVE BUS ADR OF ERROR
043674  017437  000002  001674                  MOV     @2(R4),@#ERRBA+2
043702  042737  177774  001674                  BIC     #177774,@#ERRBA+2
043710  004737  051536                          JSR     PC,@#PHYMAP                     ;GET PHYSICAL ADDRESS THAT TIMED OUT
043714  162737  000004  001476                  SUB     #4,@#PA1500                     ;ADJUST PHYSICAL ADR THAT FAILED
043722  005637  001500                          SBC     @#PA2116                        ;UBE STOPS AT ADR+4
043726  023737  001476  001562                  CMP     @#PA1500,@#MXMMLO               ;AT MAXIMUM MEMORY LO?
043734  001022                                  BNE     MHOLE                           ;BRANCH IF NO
043736  023737  001500  001560                  CMP     @#PA2116,@#MXMMHI               ;AT MAX MEMORY HI?
043744  001016                                  BNE     MHOLE                           ;BRANCH IF NO
043746  012746  000003                          MOV     #3,-(SP)                        ;PUT DEVICE ID ON STACK
043752  004737  052320                          JSR     PC,@#GIVEMAP
043756  005726                                  TST     (SP)+
043760  004737  051436                          JSR     PC,@#UBEINIT
043764  004737  053216                          JSR     PC,@#RESKT
043770  104414                                  RESREG
043772  012777  064545  136236                  MOV     #64545,@UBETBL+6
044000  000002                                  RTI

044002  010637  001276                  MHOLE:  MOV     SP,@#STMP0
044006  013737  001212  001302          UBEERR: MOV     @#$LPERR,@#STMP2                ;SAVE LOOP ARROR ADR
044014  012737  044056  001212                  MOV     #UBE3,@#$LPERR                  ;SET LOOP ADR
044022  012703  000022                          MOV     #22,R3
044026  005737  001276                          TST     @#STMP0
044032  001002                                  BNE     1$
044034  104007                                  ERROR   7
044036  000407                                  BR      UBE3
044040  013737  001476  001226          1$:     MOV     @#PA1500,@#$GDDAT
044046  013737  001500  001230                  MOV     @#PA2116,@#$BDDAT
044054  104012                                  ERROR   12

                                        ;RESTART UBE IN SAME MEMORY
044056  013737  001302  001212          UBE3:   MOV     @#STMP2,@#$LPERR                ;RESTORE ERROR LOOP ADR
044064  010446                                  MOV     R4,-(SP)                        ;SAVE R4
044066  012704  002230                          MOV     #UBETBL,R4                      ;GET ADDRESS OF UBE TABLE
044072  012734  170000                          MOV     #170000,@(R4)+                  ;SET UBECC
044076  013734  001666                          MOV     @#UBEADR,@(R4)+                 ;SET UBEBA <15:00>
044102  005074  000004                          CLR     @4(R4)                          ;CLEAR ALL ERRORS
044106  013734  001670                          MOV     @#UBEADR+2,@(R4)+               ;SET EXT ADR BITS
044112  012774  064545  000000                  MOV     #64545,@(R4)                    ;START UBE
044120  012604                                  MOV     (SP)+,R4                        ;RESTORE R4
044122  004737  053216                          JSR     PC,@#RESKT
044126  104414                                  RESREG
044130  000002                                  RTI                                     ;RETURN


                                        ;;*****************************************************************
```

```
                                     ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
                                     ;;************************************************************
044132  104412                       MBTSRV: SAVREG
044134  004737  053130                       JSR     PC,@#LDKT               ;GO TO LOW CORE
044140  005037  001276                       CLR     @#$TMP0
044144  012704  002246                       MOV     #MBTTBL,R4              ;GET ADDRESS OF ADDRESS OF CS1 REG
044150  032734  040000                       BIT     #BIT14,@(R4)+           ;ANY ERRORS?
044154  001007                               BNE     1$                      ;BRANCH IF YES
044156  004737  053216               2$:     JSR     PC,@#RESKT              ;GO BACK TO ORIGINAL CORE
044162  104414                               RESREG
044164  112777  000161  136054               MOVB    #161,@MBTTBL
044172  000002                               RTI                             ;RESTART MBT AND RETURN
044174  062704  000010               1$:     ADD     #10,R4                  ;ADJUST R4
044200  032774  004000  000000               BIT     #BIT11,@(R4)            ;NON-EXISTANT MEMORY ERROR?
044206  001436                               BEQ     MBTERR                  ;BRANCH IF NO
044210  162704  000006                       SUB     #6,R4                   ;ADJUST R4
044214  013437  001476                       MOV     @(R4)+,@#PA1500         ;GET BUS ADR
044220  013437  001500                       MOV     @(R4)+,@#PA2116         ;GET BUS ADR EXT
044224  162737  000004  001476               SUB     #4,@#PA1500             ;ADJUST BUS ADR
044232  005637  001500                       SBC     @#PA2116
044236  023737  001476  001562               CMP     @#PA1500,@#MXMMLO       ;IS IT LAST MEMORY?
044244  001015                               BNE     MEMHOLE                 ;BRANCH IF NO
044246  023737  001500  001560               CMP     @#PA2116,@#MXMMHI       ;CHECK EXT ADR BITS
044254  001011                               BNE     MEMHOLE
044256  005724                               TST     (R4)+                   ;INCREMENT R4
044260  052774  000047  000000               BIS     #47,@(R4)               ;CLEAR THE ERROR
044266  012734  000007                       MOV     #7,@(R4)+               ;SELECT UNIT 7
044272  005074  177766                       CLR     @-12(R4)                ;CLEAR WORD COUNT
044276  000727                               BR      2$                      ;CONTINUE

044300  010637  001276               MEMHOLE:MOV     SP,@#$TMP0
044304  013737  001212  001302       MBTERR: MOV     @#$LPERR,@#$TMP2        ;SAVE LOOP ADDRESS
044312  012737  044354  001212               MOV     #1$,@#$LPERR            ;SET NEW LOOP ADR
044320  012703  000020                       MOV     #20,R3                  ;PUT DEVICE ID IN R3
044324  005737  001276                       TST     @#$TMP0
044330  001002                               BNE     2$
044332  104007                               ERROR   7
044334  000407                               BR      1$
044336  013737  001476  001226       2$:     MOV     @#PA1500,@#$GDDAT
044344  013737  001500  001230               MOV     @#PA2116,@#$BDDAT
044352  104013                               ERROR   13
044354  013737  001302  001212       1$:     MOV     @#$TMP2,@#$LPERR        ;RESTORE LOOP ADR
044362  012704  002256                       MOV     #MBTTBL+10,R4           ;GET ADR OF MBTTBL+10
044366  015400                               MOV     @-(R4),R0               ;GET BUS ADR EXTENDED
044370  015401                               MOV     @-(R4),R1               ;GET BUS ADR
044372  015402                               MOV     @-(R4),R2               ;GET WORD COUNT
044374  006302                               ASL     R2                      ;ADJUST WORD COUNT
044376  160201                               SUB     R2,R1                   ;FORM START ADR OF THIS XFER
044400  005600                               SBC     R0
044402  052774  000047  000010               BIS     #47,@10(R4)             ;CLEAR THE WORLD
044410  012774  000007  000010               MOV     #7,@10(R4)              ;SELECT UNIT 7
044416  005724                               TST     (R4)+                   ;ADJUST R4
044420  010134                               MOV     R1,@(R4)+               ;RESTORE BUS ADR
044422  010074  000000                       MOV     R0,@(R4)
044426  004737  053216                       JSR     PC,@#RESKT              ;GO BACK TO ORIGINAL CORE
```

```
044434  112777  000161  135604          MOVB    #161,@MBTTBL                    ;START MBT AGAIN
044442  000002                          RTI                                     ;RETURN
                                ;;*****************************************************************
                                .SBTTL  LINE CLOCK SERVICE ROUTINE
                                ;*      THIS ROUTINE FIRST REMAPS PROGRAM EXECUTION TO LOW
                                ;*      MEMORY. IT THEN INCREMENTS AND KEEPS TRACK OF THE
                                ;*      SECOND AND MINUTE COUNTS KEPT IN LOCATIONS "LTICKS"
                                ;*      AND "MTICKS" RESPECTIVELY.
                                ;;*****************************************************************
044444  104412                  LKSRV:  SAVREG
044446  004737  053130                  JSR     PC,@#LDKT                       ;GO TO LOW CORE
044452  105237  001602                  INCB    @#LTICKS                        ;INCREMENT TICK COUNT
044456  122737  000074  001602          CMPB    #60.,@#LTICKS                   ;ONE SECOND YET?
044464  001014                          BNE     1$                              ;BRANCH IF NO
044466  105237  001603                  INCB    @#LTICKS+1                      ;INCREMENT SECOND COUNT
044472  105037  001602                  CLRB    @#LTICKS                        ;CLEAR SECOND COUNT
044476  122737  000074  001603          CMPB    #60.,@#LTICKS+1                 ;ONE MINUTE YET?
044504  001004                          BNE     1$                              ;BRANCH IF NO
044506  105037  001603                  CLRB    @#LTICKS+1
044512  005237  001600                  INC     @#MTICKS                        ;INCREMENT MINUTE COUNT
044516  004737  053216          1$:     JSR     PC,@#RESKT                      ;RESTORE THE KT
044522  104414                          RESREG
044524  012737  000100  177546          MOV     #BIT6,@#LKS                     ;CLEAR READY BIT IN CLOCK
044532  000002                          RTI                                     ;RETURN

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

                                .SBTTL  SCOPE HANDLER ROUTINE

                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW14=1        LOOP ON TEST
                                ;*SW11=1        INHIBIT ITERATIONS
                                ;*SW09=1        LOOP ON ERROR
                                ;*CALL
                                ;*      SCOPE           ;;SCOPE=IOT

044534                          $SCOPE:
044534  032737  040000  177570          BIT     #SW14,@#SWR             ;;LOOP ON PRESENT TEST?
044542  001077                          BNE     $OVER                   ;;YES IF SW14=1
                                ;#####START OF CODE FOR THE XOR TESTER#####
044544  000416                  $XTSTR: BR      6$                      ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                        ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
044546  013746  000004                  MOV     @#ERRVEC,-(SP)          ;;SAVE THE CONTENTS OF THE ERROR VECTOR
044552  012737  044572  000004          MOV     #5$,@#ERRVEC            ;;SET FOR TIMEOUT
044560  005737  177060                  TST     @#177060                ;;TIME OUT ON XOR?
044564  012637  000004                  MOV     (SP)+,@#ERRVEC          ;;RESTORE THE ERROR VECTOR
044570  000453                          BR      $SVLAD                  ;;GO TO THE NEXT TEST
044572  022626                  5$:     CMP     (SP)+,(SP)+             ;;CLEAR THE STACK AFTER A TIME OUT
044574  012637  000004                  MOV     (SP)+,@#ERRVEC          ;;RESTORE THE ERROR VECTOR
044600  000413                          BR      7$                      ;;LOOP ON THE PRESENT TEST
044602                          6$:;#####END OF CODE FOR THE XOR TESTER#####
044602  105767  134376          2$:     TSTB    $ERFLG                  ;;HAS AN ERROR OCCURRED?
044606  001421                          BEQ     3$                      ;;BR IF NO
```

```
044616  101015                           BHI     3$                      ;;BR IF NO
044620  032737  001000  177570           BIT     #BIT09,@#SWR            ;;LOOP ON ERROR?
044626  001404                           BEQ     4$                      ;;BR IF NO
044630  016767  134356  134352   7$:     MOV     $LPERR,$LPADR           ;;SET LOOP ADDRESS TO LAST SCOPE
044636  000441                           BR      $OVER
044640  105067  134340           4$:     CLRB    $ERFLG                  ;;ZERO THE ERROR FLAG
044644  005067  134446                   CLR     $TIMES                  ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
044650  000415                           BR      1$                      ;;ESCAPE TO THE NEXT TEST
044652  032737  004000  177570   3$:     BIT     #BIT11,@#SWR            ;;INHIBIT ITERATIONS?
044660  001011                           BNE     1$                      ;;BR IF YES
044662  105767  134312                   TSTB    $PASS                   ;;IF FIRST PASS OF PROGRAM
044666  001406                           BEQ     1$                      ;;      INHIBIT ITERATIONS
044670  005267  134312                   INC     $ICNT                   ;;INCREMENT ITERATION COUNT
044674  026767  134416  134304           CMP     $TIMES,$ICNT            ;;CHECK THE NUMBER OF ITERATIONS MADE
044702  002017                           BGE     $OVER                   ;;BR IF MORE ITERATION REQUIRED
044704  012767  000001  134274   1$:     MOV     #1,$ICNT                ;;REINITIALIZE THE ITERATION COUNTER
044712  016767  000054  134376           MOV     $MXCNT,$TIMES           ;;SET NUMBER OF ITERATIONS TO DO
044720  011667  134264          $SVLAD:  MOV     (SP),$LPADR             ;;SAVE SCOPE LOOP ADDRESS
044724  011667  134262                   MOV     (SP),$LPERR             ;;SAVE ERROR LOOP ADDRESS
044730  005067  134364                   CLR     $ESCAPE                 ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
044734  112767  000001  134255           MOVB    #1,$ERMAX               ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
044742  105767  134236          $OVER:   TSTB    $ERFLG                  ;;ANY ERRORS?
044746  001403                           BEQ     1$                      ;;BRANCH IF NO
044750  116737  134230  001203           MOVB    $ERFLG,@#$TSTNM+1
044756  016737  134220  177570   1$:     MOV     $TSTNM,@#DISPLAY        ;;DISPLAY TEST NUMBER
044764  016716  134220                   MOV     $LPADR,(SP)             ;;FUDGE RETURN ADDRESS
044770  000002                           RTI                             ;;FIXES PS
044772  000010                  $MXCNT:  10                              ;;MAX. NUMBER OF ITERATIONS
                                ;;*******************************************************************

                                .SBTTL  ERROR HANDLER ROUTINE

                                ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                ;*AND GO TO $ERRTYP ON ERROR
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW15=1         HALT ON ERROR
                                ;*               HALT CAN OCCUR BEFORE AND AFTER THE ERROR TYPEOUT
                                ;*SW13=1         INHIBIT ERROR TYPEOUTS
                                ;*SW10=1         BELL ON ERROR
                                ;*SW09=1         LOOP ON ERROR
                                ;*CALL
                                ;*       ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

044774                          $ERROR:
044774  116737  134204  001203           MOVB    $ERFLG,@#$TSTNM+1
045002  105267  134176           7$:     INCB    $ERFLG                  ;;SET THE ERROR FLAG
045006  001775                           BEQ     7$                      ;;DON'T LET THE FLAG GO TO ZERO
045010  016737  134166  177570           MOV     $TSTNM,@#DISPLAY        ;;DISPLAY TEST NUMBER AND ERROR FLAG
045016  005737  177570                   TST     @#SWR                   ;;HALT ON ERROR = 1?
045022  100001                           BPL     8$                      ;;BRANCH IF NO
045024  000000                           HALT                            ;;YES--HALT
045026  032737  002000  177570   8$:     BIT     #BIT10,@#SWR            ;;BELL ON ERROR?
045034  001402                           BEQ     1$                      ;;NO - SKIP
045036  104400  001322                   TYPE    ,$BELL                  ;;RING BELL
```

```
045046  011667  134146                  MOV     (SP),$ERRPC         ;;GET ADDRESS OF ERROR INSTRUCTION
045052  162767  000002  134140          SUB     #2,$ERRPC
045060  117767  134134  134130          MOVB    @$ERRPC,$ITEMB      ;;STRIP AND SAVE THE ERROR ITEM CODE
045066  032737  020000  177570          BIT     #BIT13,@#SWR        ;;SKIP TYPEOUT IF SET
045074  001004                          BNE     2$                  ;;SKIP TYPEOUTS
045076  004767  000056                  JSR     PC,$ERRTYP          ;;GO TO USER ERROR ROUTINE
045102  104400  001327                  TYPE    ,$CRLF
045106  005737  177570          2$:     TST     @#SWR               ;;HALT ON ERROR
045112  100001                          BPL     9$                  ;;SKIP IF CONTINUE
045114  000000                          HALT                        ;;HALT ON ERROR!
045116  022767  036752  132716  9$:     CMP     #$ENDAD,42          ;;ACT-11?
045124  001001                          BNE     3$                  ;;BRANCH IF NO
045126  000000                          HALT                        ;;YES
045130  032737  001000  177570  3$:     BIT     #BIT09,@#SWR        ;;LOOP ON ERROR SWITCH SET?
045136  001402                          BEQ     4$                  ;;BR IF NO
045140  016716  134046                  MOV     $LPERR,(SP)         ;;FUDGE RETURN FOR LOOPING
045144  005767  134150          4$:     TST     $ESCAPE             ;;CHECK FOR AN ESCAPE ADDRESS
045150  001402                          BEQ     5$                  ;;BR IF NONE
045152  016716  134142                  MOV     $ESCAPE,(SP)        ;;FUDGE RETURN ADDRESS FOR ESCAPE
045156                          5$:
045156  000002                          RTI                         ;;RETURN
                                ;;*************************************************************************

                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

                                ;*THIS ROUTINE FIRST TYPES A STANDARD MESSAGE CONSISTING OF THE
                                ;*VIRTUAL PC, THE PHYSICAL PC, THE PSW AT THE TIME OF THE ERROR CALL,
                                ;*AND THE  SUB-PASS COUNT. THE SUB-PASS COUNT CONSISTS OF THE SUB PASS COUNT IN THE
                                ;*HIGH BYTE AND THE PASS COUNT IN THE LOW BYTE.
                                ;*
                                ;*IT THEN USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE"
                                ;*THE ERROR MESSAGE POINTER AND TYPES THE ERROR MESSAGE. THE DATA
                                ;*HEADER POINTER IS THEN OBTAINED AND A DATA HEADER IS TYPED.
                                ;*THE DATA POINTER AND DATA FORMAT ARE THEN OBTAINED. THERE ARE
                                ;*FOUR TYPES OF DATA FORMAT, AS FOLLOWS:
                                ;*
                                ;*      0       TYPE THE CONTENTS OF THE DATA TABLE WORD IN
                                ;*              6 DIGIT OCTAL FORMAT
                                ;*      1       CONVERT THE CONTENTS OF THE DATA TABLE WORD TO
                                ;*              22 BITS AND TYPE AN 8 DIGIT OCTAL NUMBER
                                ;*      2       TYPE THE CONTENTS OF THE DATA TABLE WORD AND
                                ;*              THE WORD+2 IN 8 DIGIT OCTAL FORMAT
                                ;*      3       USE THE CONTENTS OF THE DATA TABLE WORD AS A
                                ;*              DEVICE ID AND TYPE THE DEVICES NAME
                                ;*      4       CONVERT THE TWO WORDS POINTED TO BY THE DATA
                                ;*              TABLE TO FLOATING POINT FORMAT AND TYPE.
                                ;*      5       CONVERT THE FOUR WORDS POINTED TO BY THE DATA
                                ;*              TABLE TO FLOATING  DOUBLE FORMAT AND TYPE
                                ;;*************************************************************************

045160  104412                  $ERRTYP:SAVREG
045162  104400  001327                  TYPE    ,$CRLF              ;;"CARRIAGE RETURN" & "LINE FEED"
045166  004737  046660                  JSR     PC,@#TYPTIME        ;GO TYPE THE TIME
045172  104400  053662                  TYPE    ,MSG3
045176  104400  001327                  TYPE    ,$CRLF
```

```
                                                          ;;TYPE THE VIRTUAL PC
 045206  104402                          TYPOC            ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 045210  104400  046432                  TYPE     ,8$
 045214  013700  001474                  MOV      @#VADR,R0          ;SAVE VADR
 045220  013737  001220  001474          MOV      @#$ERRPC,@#VADR    ;SAVE THE VIR PC FOR CONVERSION
 045226  122737  000014  001216          CMPB     #14,@#$ITEMB
 045234  003403                          BLE      51$
 045236  105737  001216                  TSTB     @#$ITEMB           ;ERROR ZERO?
 045242  001005                          BNE      42$                ;BRANCH IF NO
 045244  004737  051644          51$:    JSR      PC,@#CNVADR        ;CONVERT TO 22 BITS
 045250  010037  001474                  MOV      R0,@#VADR
 045254  000407                          BR       41$
 045256  013737  001474  001476  42$:    MOV      @#VADR,@#PA1500
 045264  005037  001500                  CLR      @#PA2116
 045270  010037  001474                  MOV      R0,@#VADR
 045274  012746  001476          41$:    MOV      #PA1500,-(SP)      ;PUT ADDRESS OFPC ON STACK
 045300  004737  047712                  JSR      PC,@#$DB20         ;CONVERT TO ASCII
 045304  062716  000003                  ADD      #3,(SP)            ;GET RID OF 3 MS DIGITS
 045310  012667  000002                  MOV      (SP)+,30$          ;SAVE POINTER TO ASCII
 045314  104400                          TYPE                        ;TYPE IT
 045316  000000                  30$:    .WORD
 045320  104400  046432                  TYPE     ,8$
 045324  016646  000030                  MOV      30(SP),-(SP)       ;GET PSW AT TIME OF ERROR
 045330  104402                          TYPOC                       ;TYPE IT
 045332  104400  046432                  TYPE     ,8$
 045336  016746  134242                  MOV      $MAINT,-(SP)    ;;SAVE $MAINT FOR TYPEOUT
                                                                  ;;TYPE THE MAINTENANCE REG
 045342  104402                          TYPOC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 045344  104400  046432                  TYPE     ,8$
 045350  116746  133626                  MOVB     $TSTNM,-(SP)
 045354  105066  000001                  CLRB     1(SP)
 045360  104402                          TYPOC                    ;TYPE THE TEST NUMBER
 045362  104400  046432                  TYPE     ,8$
 045366  013746  001532                  MOV      @#SUBPASS,-(SP)
 045372  162716  000060                  SUB      #60,(SP)
 045376  104402                          TYPOC
 045400  104400  046432                  TYPE     ,8$
 045404  016746  133570                  MOV      $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                                                  ;;TYPE THE PASS COUNT
 045410  104402                          TYPOC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
 045412  104400  001327                  TYPE     ,$CRLF
 045416  005000                          CLR      R0
 045420  153700  001216                  BISB     @#$ITEMB,R0          ;PICK UP THE INDEX
 045424  001431                          BEQ      6$                   ;EXIT IF ZERO
 045426  022700  000007          1$:     CMP      #7,R0           ;IS THIS ERROR 7?
 045432  001551                          BEQ      15$             ;BRANCH IF YES
 045434  005300                          DEC      R0              ;;ADJUST THE INDEX SO THAT IT WILL
 045436  006300                          ASL      R0              ;;        WORK FOR THE ERROR TABLE
 045440  006300                          ASL      R0
 045442  006300                          ASL      R0
 045444  062700  002304                  ADD      #$ERRTB,R0      ;;FORM TABLE POINTER
 045450  012067  000004                  MOV      (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
 045454  001404                          BEQ      3$              ;;SKIP TYPEOUT IF NO POINTER
 045456  104400                          TYPE                     ;;TYPE THE "ERROR MESSAGE"
 045460  000000                  2$:     .WORD    0               ;;"ERROR MESSAGE" POINTER GOES HERE
```

```
045466  012067  000004          3$:     MOV     (R0)+,4$                ;;PICKUP "DATA HEADER" POINTER
045472  001404                          BEQ     5$                      ;;SKIP TYPEOUT IF 0
045474  104400                          TYPE                            ;;TYPE THE "DATA HEADER"
045476  000000                  4$:     .WORD   0                       ;;"DATA HEADER" POINTER GOES HERE
045500  104400  001327                  TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
045504  012001                  5$:     MOV     (R0)+,R1                        ;;PICKUP "DATA TABLE" POINTER
045506  001004                          BNE     7$                      ;;GO TYPE THE DATA
045510  104414                  6$:     RESREG
045512  104400  001327                  TYPE    ,$CRLF                  ;;"CARRIAGE RETURN" & "LINE FEED"
045516  000207                          RTS     PC                      ;;RETURN
045520  011002                  7$:     MOV     (R0),R2                 ;GET "DATA FORMAT" POINTER
045522  122712  000001          10$:    CMPB    #1,(R2)                 ;DATA FORMAT 1?
045526  001424                          BEQ     9$                      ;BRANCH IF YES
045530  122712  000002                  CMPB    #2,(R2)                 ;DATA FORMAT 2?
045534  001441                          BEQ     11$                     ;BRANCH IF YES
045536  122712  000003                  CMPB    #3,(R2)                 ;DATA FORMAT 3?
045542  001445                          BEQ     24$                     ;BRANCH IF YES
045544  122712  000004                  CMPB    #4,(R2)                 ;DATA FORMAT 4?
045550  001456                          BEQ     40$                     ;BRANCH IF YES
045552  122712  000005                  CMPB    #5,(R2)                 ;DATA FORMAT 5?
045556  001465                          BEQ     60$                     ;BRANCH IF YES
                                ;;*************************************************************************
                                ;DATA FORMAT 0
045560  005202                          INC     R2                      ;INCREMENT FORMAT POINTER
045562  013146                          MOV     @(R1)+,-(SP)            ;PUSH DATA TO BE TYPED
045564  104402                          TYPOC
045566  005711                  13$:    TST     (R1)                    ;ANY MORE DATA?
045570  001747                          BEQ     6$                      ;BRANCH IF NO
045572  104400  046432                  TYPE    ,B$                     ;TYPE TWO SPACES
045576  000751                          BR      10$
                                ;;*************************************************************************
                                ;DATA FORMAT 1
045600  005202                  9$:     INC     R2                      ;INCREMENT FORMAT POINTER
045602  004737  051644                  JSR     PC,@#CNVADR             ;GET 22 BIT ADR
045606  012746  001476          14$:    MOV     #PA1500,-(SP)           ;PUSH ADR OF 22 BIT ADR
045612  004737  047712                  JSR     PC,@#$DB20              ;CONVERT TO ASCII
045616  062716  000003                  ADD     #3,(SP)                 ;DELETE LEADING ZEROS
045622  012667  000002                  MOV     (SP)+,12$               ;GET ADR OF ASCII STRING
045626  104400                          TYPE
045630  000000                  12$:    .WORD
045632  062701  000002                  ADD     #2,R1                   ;INCREMENT R1
045636  000753                          BR      13$
                                ;;*************************************************************************
                                ;DATA FORMAT 2
045640  005202                  11$:    INC     R2                      ;INCREMENT FORMAT POINTER
045642  011100                          MOV     (R1),R0
045644  012037  001476                  MOV     (R0)+,@#PA1500
045650  011037  001500                  MOV     (R0),@#PA2116
045654  000754                          BR      14$
                                ;;*************************************************************************
                                ;DATA FORMAT 3
045656  005202                  24$:    INC     R2                      ;INCREMENT FORMAT POINTER
045660  013167  000016                  MOV     @(R1)+,25$              ;GET DEVICE ID
045664  062767  053612  000010          ADD     #MSGINX,25$             ;FORM ADR OF ASCIZ ADR
045672  017767  000004  000002          MOV     @25$,25$                ;GET ADR OF ASCIZ
```

```
 045702  000000                  25$:    .WORD
 045704  000730                          BR      13$                     ;CONTINUE
                                 ;;*******************************************************************
                                 ;DATA FORMAT 4
 045706  005202                  40$:    INC     R2
 045710  012167  000002                  MOV     (R1)+,44$               ;GET ADDRESS OF DATA
 045714  104416                          FL20                            ;CONVERT TO FLOATING FORMAT
 045716  000000                  44$:    .WORD
 045720  012667  000002                  MOV     (SP)+,45$               ;GET ADDRESS OF ASCIZ STRING
 045724  104400                          TYPE                            ;TYPE THE DATA
 045726  000000                  45$:    .WORD
 045730  000716                          BR      13$
                                 ;;*******************************************************************
                                 ;DATA FORMAT 5
 045732  005202                  60$:    INC     R2                      ;INCREMENT FORMAT POINTER
 045734  012167  000002                  MOV     (R1)+,61$               ;GET ADDRESS OF DATA
 045740  104420                          FLD20                           ;CONVERT TO FLOATING ASCIZ
 045742  000000                  61$:    .WORD
 045744  012667  000002                  MOV     (SP)+,62$               ;GET ADDRESS OF ASCIZ STRING
 045750  104400                          TYPE                            ;TYPE THE DATA
 045752  000000                  62$:    .WORD
 045754  000704                          BR      13$
                                 ;;*******************************************************************
                                 ;ERROR 7 DECODE
 045756  010300                  15$:    MOV     R3,R0                   ;SAVE R3
 045760  062700  053612                  ADD     #MSGINX,R0              ;GEN ADRS OF ASCIZ
 045764  011067  000002                  MOV     (R0),16$
 045770  104400                          TYPE
 045772  000000                  16$:    .WORD
 045774  104400  046002                  TYPE    ,65$                    ;;TYPE ASCIZ STRING
 046000  000404                          BR      64$                     ;;GET OVER THE ASCIZ
                                 ;;65$:   .ASCIZ  /FAILED/<CRLF>
 046012                          64$:
 046012  010300                          MOV     R3,R0                   ;SAVE DEVICE ID
 046014  022700  000010                  CMP     #10,R0                  ;MASS BUS DEVICE?
 046020  003403                          BLE     17$                     ;BRANCH IF YES
 046022  104400  054116                  TYPE    ,MSG12
 046026  000411                          BR      18$
                                 ;;*******************************************************************
                                 ;MASS BUS ERR
 046030  022703  000020          17$:    CMP     #20,R3                  ;MBT ERROR?
 046034  001426                          BEQ     26$                     ;BRANCH IF MBT ERROR
 046036  002435                          BLT     27$                     ;BRANCH IF UBE ERROR
 046040  104400  054231                  TYPE    ,MSG13
 046044  022700  000012                  CMP     #12,R0                  ;WAS IT RS?
 046050  001140                          BNE     29$                     ;BRANCH IF NO
                                 ;;*******************************************************************
                                 ;UNIBUS ERROR OR RS04 ERROR
 046052  062700  053566          18$:    ADD     #REGINX,R0              ;FORM ADR OF REG TABLE
 046056  011000                          MOV     (R0),R0                 ;GET ADR OF REG TABLE
 046060  022703  000002                  CMP     #2,R3                   ;RP3 OR RK?
 046064  001404                          BEQ     20$                     ;BRANCH IF RK
 046066  100406                          BMI     21$                     ;BRANCH IF NOT RP03
 046070  012704  000007                  MOV     #7,R4                   ;SET RP03 SOB COUNT
 046074  000423                          BR      22$
```

```
046102  000420                                  BR      22$
046104  012704  000011                  21$:    MOV     #11,R4          ;SET RS04 SOB COUNT
046110  000415                                  BR      22$
                                        ;;*******************************************************************
                                        ;MBT ERROR
046112  104400  054420                  26$:    TYPE    ,MSG16
046116  012704  000011                          MOV     #11,R4          ;SET MBT SOB COUNT
046122  062700  053566                  28$:    ADD     #REGINX,R0
046126  011000                                  MOV     (R0),R0         ;GET ADR OF MBT TABLE
046130  000405                                  BR      22$             ;GO TYPE REGISTERS
                                        ;UNIBUS  EXERCISER ERROR
046132  104400  054527                  27$:    TYPE    ,MSG17
046136  012704  000004                          MOV     #4,R4           ;SET UBE SOB COUNT
046142  000767                                  BR      28$             ;GO TYPE UBE REGISTERS
046144  013046                          22$:    MOV     @(R0)+,-(SP)    ;GET DATA IN REG
046146  104402                                  TYPOC                   ;TYPE IT
046150  104400  046432                          TYPE    ,BS             ;TYPE TWO SPACES
046154  077405                                  SOB     R4,22$          ;CONTINUE
                                        ;;*******************************************************************
                                        ;THIS CODE TYPES A PHYSICAL BUS ADDRESS IF THE ERROR WAS AN RP03, RK05, OR UBE
046156  022703  000022                          CMP     #22,R3          ;UBE ERROR?
046162  001454                                  BEQ     73$             ;BRANCH IF YES
046164  022703  000002                          CMP     #2,R3           ;RK05?
046170  002445                                  BLT     32$             ;BRANCH IF NOT RK OR RP03
046172  001005                                  BNE     70$             ;BRANCH IF RP03
                                        ;RK05 ERROR
046174  104400  054723                          TYPE    ,MSG22
046200  012700  002126                          MOV     #RKCS,R0        ;GET ADR OF ADR OF RKCS REG
046204  000404                                  BR      71$
                                        ;RP03 ERROR
046206  012700  002104                  70$:    MOV     #RP3CS,R0       ;GET ADR OF ADR OF RP3CS REG
046212  104400  054733                          TYPE    ,MSG23
                                        ;GET, CALCULATE, & TYPE PHYSICAL BUS ADDRESS
046216  013001                          71$:    MOV     @(R0)+,R1       ;GET BUS ADR EXTENDED BITS
046220  005720                                  TST     (R0)+           ;ADJUST R0
046222  013037  001672                          MOV     @(R0)+,@#ERRBA  ;GET BUS ADRESS THAT FAILED
046226  072127  177774                          ASH     #-4,R1          ;GET BITS 4&5 INTO BITS 0&1
046232  042701  177774                          BIC     #177774,R1      ;GET RID OF UNUSED BITS
046236  010137  001674                          MOV     R1,@#ERRBA+2    ;SAVE EXTENDED BITS
046242  162737  000002  001672  74$:    SUB     #2,@#ERRBA      ;DECREMENT BUS ADR
046250  005637  001674                          SBC     @#ERRBA+2
046254  004737  051536                          JSR     PC,@#PHYMAP     ;GO CONVERT TO 22 BIT PHYSICAL
046260  012746  001476                          MOV     #PA1500,-(SP)
046264  004737  047712                          JSR     PC,@#$DB20      ;CONVERT TO ASCIZ STRING
046270  062716  000003                          ADD     #3,(SP)         ;GET RID OF LEADING ZEROS
046274  012667  000002                          MOV     (SP)+,72$
046300  104400                                  TYPE
046302  000000                          72$:    .WORD
046304  104400  001327                  32$:    TYPE    ,$CRLF
```

```
                                        ;GET UBE VIRTUAL ADDRESS
046314  012700  002232                  73$:    MOV     #UBETBL+2,R0        ;GET ADR OF UBE TABLE +2
046320  013037  001672                          MOV     @(R0)+,@#ERRBA      ;GET BUS ADR THAT FAILED
046324  013037  001674                          MOV     @(R0)+,@#ERRBA+2;GET BAE BITS
046330  042737  177774  001674                  BIC     #177774,@#ERRBA+2;MASK OFF ADR BITS
046336  162737  000002  001672                  SUB     #2,@#ERRBA
046344  005637  001674                          SBC     @#ERRBA+2
046350  000734                                  BR      74$                 ;GO CONVERT & TYPE PHYSICAL ADR
                                        ;;*******************************************************************
                                        ;RP04 ERROR
046352  062700  053566                  29$:    ADD     #REGINX,R0
046356  011000                                  MOV     (R0),R0             ;FORM ADR OF RP04 TABLE
046360  012704  000011                          MOV     #11,R4              ;SET SOB COUNT
046364  013046                          31$:    MOV     @(R0)+,-(SP)        ;GET DATA TO BE TYPED
046366  104402                                  TYPOC                       ;TYPE DATA
046370  104400  046432                          TYPE    ,8$
046374  077405                                  SOB     R4,31$              ;CONTINUE
046376  104400  001327                          TYPE    ,$CRLF
046402  104400  001327                          TYPE    ,$CRLF
046406  012704  000004                          MOV     #4,R4               ;SET SOB COUNT
046412  104400  054341                          TYPE    ,MSG14
046416  013046                          50$:    MOV     @(R0)+,-(SP)        ;GET DTA TO BE TYPED
046420  104402                                  TYPOC                       ;TYPE IT
046422  104400  046432                          TYPE    ,8$
046426  077405                                  SOB     R4,50$              ;CONTINUE
046430  000725                                  BR      32$
046432  020040     000                  8$:     .ASCIZ  /  /                ;;TWO(2) SPACES
        046436                                  .EVEN
                                        ;;*******************************************************************

                                        .SBTTL  TYPE ROUTINE

                                        ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                        ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                        ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                        ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                        ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                        ;*
                                        ;*CALL:
                                        ;*1) USING A TRAP INSTRUCTION
                                        ;*              TYPE    ,MESADR             ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                        ;*OR
                                        ;*              TYPE
                                        ;*              MESADR
                                        ;*
                                        ;*2) USING A JSR INSTRUCTION
                                        ;*              MOV     PS,-(SP)            ;;PUSH PROCESSOR STATUS WORD ON THE STACK
                                        ;*              JSR     PC,$TYPE            ;;CALL TYPE ROUTINE
                                        ;*              MESADDR                     ;;FIRST ADRESS OF MESSAGE

046436  105767  132611                  $TYPE:  TSTB    $TPFLG              ;;IS THERE A TERMINAL?
046442  100002                                  BPL     1$                  ;;BR IF YES
046444  000000                                  HALT                        ;;HALT HERE IF NO TERMINAL
046446  000407                                  BR      3$                  ;;LEAVE
046450  010046                          1$:     MOV     R0,-(SP)            ;;SAVE R0
```

```
046456  112046                  2$:     MOVB    (R0)+,-(SP)             ;;PUSH CHARACTER TO BE TYPED ONTO STACK
046460  001005                          BNE     4$                      ;;BR IF IT ISN'T THE TERMINATOR
046462  005726                          TST     (SP)+                   ;;IF TERMINATOR POP IT OFF THE STACK
046464  012600                          MOV     (SP)+,R0                ;;RESTORE R0
046466  062716  000002          3$:     ADD     #2,(SP)                 ;;ADJUST RETURN PC
046472  000002                          RTI                             ;;RETURN
046474  122716  000011          4$:     CMPB    #HT,(SP)                ;;BRANCH IF <HT>
046500  001426                          BEQ     8$
046502  122716  000200                  CMPB    #CRLF,(SP)              ;;BRANCH IF NOT
046506  001004                          BNE     5$
046510  005726                          TST     (SP)+                   ;;POP  <CR><LF> EQUIV
046512  104400  001327                  TYPE    ,$CRLF
046516  000757                          BR      2$                      ;;GET NEXT CHARACTER
046520  004767  000056          5$:     JSR     PC,$TYPEC               ;;GO TYPE THIS CHARACTER
046524  126726  132522          6$:     CMPB    $FILLC,(SP)+            ;;IS IT TIME FOR FILLER CHARS.?
046530  001352                          BNE     2$                      ;;IF NO GO GET NEXT CHAR.
046532  016746  132512                  MOV     $NULL,-(SP)             ;;GET # OF FILLER CHARS. NEEDED
                                                                        ;;AND THE NULL CHAR.
046536  105366  000001          7$:     DECB    1(SP)                   ;;DOES A NULL NEED TO BE TYPED?
046542  002770                          BLT     6$                      ;;BR IF NO--GO POP THE NULL OFF OF STACK
046544  004767  000032                  JSR     PC,$TYPEC               ;;GO TYPE A NULL
046550  105367  000100                  DECB    $CHARCNT                ;;DON'T COUNT THE NULL AS A CHARACTER
046554  000770                          BR      7$                      ;;LOOP

                                ;;HORIZONTAL TAB PROCESSOR

046556  112716  000040          8$:     MOVB    #' ,(SP)                ;;REPLACE TAB WITH SPACE
046562  004767  000014          9$:     JSR     PC,$TYPEC               ;;TYPE A SPACE
046566  132767  000007  000060          BITB    #7,$CHARCNT             ;;BRANCH IF NOT AT
046574  001372                          BNE     9$                      ;;TAB STOP
046576  005726                          TST     (SP)+                   ;;POP SPACE OFF STACK
046600  000726                          BR      2$                      ;;GET NEXT CHARACTER
046602  005737  001466          $TYPEC: TST     @#NOTYPE                ;;INHIBIT TYPING?
046606  100423                          BMI     $TYPEX                  ;;BRANCH IF YES
046610  105777  132430                  TSTB    @$TPS                   ;WAIT UNTIL PRINTER IS READY
046614  100372                          BPL     $TYPEC
046616  116677  000002  132422          MOVB    2(SP),@$TPB             ;;LOAD CHAR TO BE TYPED INTO DATA REG.
046624  122766  000015  000002          CMPB    #CR,2(SP)               ;;BRANCH IF
046632  001003                          BNE     1$                      ;;NOT <CR>
046634  105067  000014                  CLRB    $CHARCNT                ;;
046640  000406                          BR      $TYPEX                  ;;EXIT
046642  122766  000012  000002  1$:     CMPB    #LF,2(SP)               ;;BRANCH IF
046650  001402                          BEQ     $TYPEX                  ;;<LF>
046652  105227                          INCB    (PC)+                   ;;INC SPACE
046654  000000                  $CHARCNT:.WORD  0                       ;;COUNT
046656  000207                  $TYPEX: RTS     PC


                                ;;*****************************************************************
                                .SBTTL  ROUTINE TO TYPE THE ELAPSED RUN TIME OF THE PROGRAM
                                ;*      THIS ROUTINE CONVERTS THE CONTENTS OF LOCATIONS "LTICKS"
                                ;*      AND "MTICKS" TO SECONDS AND MINUTES/HOURS RESPECTIVELY
                                ;*      AND TYPES THEM IN THE FOLLOWING FORMAT:
                                ;*              HHH:MM:SS
                                ;;*****************************************************************
046660  104412                  TYPTIME:SAVREG
```

```
046666  113701  001603                  MOVB    @#LTICKS+1,R1   ;GET SECOND COUNT
046672  005000                          CLR     R0
046674  071027  000012                  DIV     #10.,R0
046700  062701  000060                  ADD     #60,R1
046704  110137  047116                  MOVB    R1,@#TIMEBUF+10
046710  010001                          MOV     R0,R1
046712  005000                          CLR     R0
046714  071027  000006                  DIV     #6,R0
046720  062701  000060                  ADD     #60,R1
046724  110137  047115                  MOVB    R1,@#TIMEBUF+7
046730  013701  001600                  MOV     @#MTICKS,R1     ;GET MINUTE COUNT
046734  005000                          CLR     R0
046736  071027  000012                  DIV     #10.,R0         ;GET HOURS AND MINUTES
046742  062701  00006C                  ADD     #60,R1          ;MAKE REMAINDER ASCII
046746  110167  000141                  MOVB    R1,TIMEBUF+5    ;PUT IN BUFFER
046752  010001                          MOV     R0,R1
046754  005000                          CLR     R0
046756  071027  000006                  DIV     #6.,R0
046762  062701  000060                  ADD     #60,R1
046766  110167  000120                  MOVB    R1,TIMEBUF+4
046772  005700                          TST     R0
046774  001434                          BEQ     2$
046776  010001                          MOV     R0,R1
047000  005000                          CLR     R0
047002  071027  000012                  DIV     #10.,R0
047006  062701  000060                  ADD     #60,R1
047012  110167  000072                  MOVB    R1,TIMEBUF+2
047016  005700                          TST     R0
047020  001422                          BEQ     2$
047022  010001                          MOV     R0,R1
047024  005000                          CLR     R0
047026  071027  000010                  DIV     #10,R0
047032  062701  000060                  ADD     #60,R1
047036  110167  000045                  MOVB    R1,TIMEBUF+1
047042  005700                          TST     R0
047044  001410                          BEQ     2$
047046  010001                          MOV     R0,R1
047050  005000                          CLR     R0
047052  071027  000012                  DIV     #10.,R0
047056  062701  000060                  ADD     #60,R1
047062  110167  000020                  MOVB    R1,TIMEBUF
047066  104400  047106          2$:     TYPE    ,TIMEBUF
047072  104400  001327                  TYPE    ,$CRLF
047076  004737  053216                  JSR     PC,@#RESKT      ;GO BACK TO ORIGINAL MEMORY
047102  104414                          RESREG
047104  000207                          RTS     PC
047106     001     001     001  TIMEBUF:.BYTE   1,1,1,72,1,1,72,60,60,0
047111     072     001     001
047114     072     060     060
047117     000

                                        .EVEN
                                ;;******************************************************
                                .SBTTL  ROUTINE TO TYPE THE AVAILABLE DEVICES AND UNIT NUMBERS
                                ;*      THIS ROUTINE SEARCHES THE SYSTEM SIZE TABLE FOR NON-
                                ;*      ZERO ENTRIES.WHEN IT FINDS ONE, IT TYPES THE NAME OF THE
```

```
                                        ;*      AVAILABLE FOR THAT DEVICE.
                                        ;;*********************************************************************
047120  104400  056372                  TYPSIZ: TYPE    ,SWITCH
047124  104400  053751                          TYPE    ,MSG4
047130  012700  000010                          MOV     #10,R0          ;SET SOB COUNT
047134  005001                                  CLR     R1
047136  105761  001606                  1$:     TSTB    SYSSIZE(R1)     ;DEVICE AVAILABLE?
047142  001004                                  BNE     2$              ;BRANCH IF YES
047144  062701  000002                  7$:     ADD     #2,R1           ;INCREMENT INDEX
047150  077006                                  SOB     R0,1$           ;CONTINUE
047152  000207                                  RTS     PC              ;RETURN
047154  010102                          2$:     MOV     R1,R2           ;GET INDEX
047156  062702  053612                          ADD     #MSGINX,R2      ;GET ADR OF MESSAGE ADR
047162  011267  000002                          MOV     (R2),3$         ;GET ADDRESS OF MESSAGE
047166  104400                                  TYPE
047170  000000                          3$:     .WORD
047172  112767  000060  000034                  MOVB    #60,4$          ;INIT UNIT NO. BUFFER (ASCII)
047200  116102  001606                          MOVB    SYSSIZE(R1),R2  ;GET WORD WITH AVAILABLE UNITS
047204  012703  000010                          MOV     #10,R3          ;SET SOB COUNT
047210  006002                          6$:     ROR     R2              ;GET UNITS
047212  103002                                  BCC     5$              ;BRANCH IF NOT A UNIT
047214  104400  047234                          TYPE    ,4$
047220  005267  000010                  5$:     INC     4$
047224  077307                                  SOB     R3,6$           ;CONTINUE
047226  104400  001327                          TYPE    ,$CRLF
047232  000744                                  BR      7$
047234     000     054     040          4$:     .BYTE   0,54,40,0       ;NUMBER,COMMA,SPACE,TERMINATOR
047237     000

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

                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                        ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)               ;;NUMBER TO BE TYPED
                                        ;*      TYPOS                           ;;CALL FOR TYPEOUT
                                        ;*      .BYTE   N                       ;;N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
                                        ;*      .BYTE   M                       ;;M=1 OR 0
                                        ;*                                              ;;1=TYPE LEADING ZEROS
                                        ;*                                              ;;0=SUPPRESS LEADING ZEROS
                                        ;*
                                        ;*$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
                                        ;*$TYPOS OR $TYPOC
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)               ;;NUMBER TO BE TYPED
                                        ;*      TYPON                           ;;CALL FOR TYPEOUT
                                        ;*
                                        ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                        ;*CALL:
                                        ;*      MOV     NUM,-(SP)               ;;NUMBER TO BE TYPED
                                        ;*      TYPOC                           ;;CALL FOR TYPEOUT

047240  017646  000000                  $TYPOS: MOV     @(SP),-(SP)             ;;PICKUP THE MODE
```

```
047252  112667  000207                     MOVB    (SP)+,$OMODE+1      ;;NUMBER OF DIGITS TO TYPE
047256  062716  000002                     ADD     #2,(SP)             ;;ADJUST RETURN ADDRESS
047262  000406                             BR      $TYPON
047264  112767  000001  000171   $TYPOC:   MOVB    #1,$OFILL           ;;SET THE ZERO FILL SWITCH
047272  112767  000006  000165             MOVB    #6,$OMODE+1         ;;SET FOR SIX(6) DIGITS
047300  112767  000005  000154   $TYPON:   MOVB    #5,$OCNT            ;;SET THE ITERATION COUNT
047306  010346                             MOV     R3,-(SP)            ;;SAVE R3
047310  010446                             MOV     R4,-(SP)            ;;SAVE R4
047312  010546                             MOV     R5,-(SP)            ;;SAVE R5
047314  116704  000145                     MOVB    $OMODE+1,R4         ;;GET THE NUMBER OF DIGITS TO TYPE
047320  005404                             NEG     R4
047322  062704  000006                     ADD     #6,R4               ;;SUBTRACT IT FOR MAX. ALLOWED
047326  110467  000132                     MOVB    R4,$OMODE           ;;SAVE IT FOR USE
047332  116704  000125                     MOVB    $OFILL,R4           ;;GET THE ZERO FILL SWITCH
047336  016605  000012                     MOV     12(SP),R5           ;;PICKUP THE INPUT NUMBER
047342  005003                             CLR     R3                  ;;CLEAR THE OUTPUT WORD
047344  006105                   1$:       ROL     R5                  ;;ROTATE MSB INTO "C"
047346  000404                             BR      3$                  ;;GO DO MSB
047350  006105                   2$:       ROL     R5                  ;;FORM THIS DIGIT
047352  006105                             ROL     R5
047354  006105                             ROL     R5
047356  010503                             MOV     R5,R3
047360  006103                   3$:       ROL     R3                  ;;GET LSB OF THIS DIGIT
047362  105367  000076                     DECB    $OMODE              ;;TYPE THIS DIGIT?
047366  100016                             BPL     7$                  ;;BR IF NO
047370  042703  177770                     BIC     #177770,R3          ;;GET RID OF JUNK
047374  001002                             BNE     4$                  ;;TEST FOR 0
047376  005704                             TST     R4                  ;;SUPPRESS THIS 0?
047400  001403                             BEQ     5$                  ;;BR IF YES
047402  005204                   4$:       INC     R4                  ;;DON'T SUPPRESS ANYMORE 0'S
047404  052703  000060                     BIS     #'0,R3              ;;MAKE THIS DIGIT ASCII
047410  052703  000040           5$:       BIS     #' ,R3              ;;MAKE ASCII IF NOT ALREADY
047414  110367  000040                     MOVB    R3,8$               ;;SAVE FOR TYPING
047420  104400  047460                     TYPE    ,8$                 ;;GO TYPE THIS DIGIT
047424  105367  000032           7$:       DECB    $OCNT               ;;COUNT BY 1
047430  003347                             BGT     2$                  ;;BR IF MORE TO DO
047432  002402                             BLT     6$                  ;;BR IF DONE
047434  005204                             INC     R4                  ;;INSURE LAST DIGIT ISN'T A BLANK
047436  000744                             BR      2$                  ;;GO DO THE LAST DIGIT
047440  012605                   6$:       MOV     (SP)+,R5            ;;RESTORE R5
047442  012604                             MOV     (SP)+,R4            ;;RESTORE R4
047444  012603                             MOV     (SP)+,R3            ;;RESTORE R3
047446  016666  000002  000004             MOV     2(SP),4(SP)         ;;SET THE STACK FOR RETURNING
047454  012616                             MOV     (SP)+,(SP)
047456  000002                             RTI                         ;;RETURN
047460     000                   8$:       .BYTE   0                   ;;STORAGE FOR ASCII DIGIT
047461     000                             .BYTE   0                   ;;TERMINATOR FOR TYPE ROUTINE
047462     000                   $OCNT:    .BYTE   0                   ;;OCTAL DIGIT COUNTER
047463     000                   $OFILL:   .BYTE   0                   ;;ZERO FILL SWITCH
047464  000000                   $OMODE:   .WORD   0                   ;;NUMBER OF DIGITS TO TYPE
                                 ;;************************************************************************



                                 .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

                                 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
```

K14

```
                                      ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                      ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                      ;*REPLACED WITH SPACES.
                                      ;*CALL:
                                      ;*      MOV     NUM,-(SP)               ;;PUT THE BINARY NUMBER ON THE STACK
                                      ;*      TYPDS                           ;;GO TO THE ROUTINE

047466                                $TYPDS:
047466  010046                                MOV     R0,-(SP)                ;;PUSH R0 ON STACK
047470  010146                                MOV     R1,-(SP)                ;;PUSH R1 ON STACK
047472  010246                                MOV     R2,-(SP)                ;;PUSH R2 ON STACK
047474  010346                                MOV     R3,-(SP)                ;;PUSH R3 ON STACK
047476  010546                                MOV     R5,-(SP)                ;;PUSH R5 ON STACK
047500  012746  020200                        MOV     #20200,-(SP)            ;;SET BLANK SWITCH AND SIGN
047504  016605  000020                        MOV     20(SP),R5               ;;GET THE INPUT NUMBER
047510  100004                                BPL     1$                      ;;BR IF INPUT IS POS.
047512  005405                                NEG     R5                      ;;MAKE THE BINARY NUMBER POS.
047514  112766  000055  000001                MOVB    #'-,1(SP)               ;;MAKE THE ASCII NUMBER NEG.
047522  005000                        1$:     CLR     R0                      ;;ZERO THE CONSTANTS INDEX
047524  012703  047702                        MOV     #$DBLK,R3               ;;SETUP THE OUTPUT POINTER
047530  112723  000040                        MOVB    #' ,(R3)+               ;;SET THE FIRST CHARACTER TO A BLANK
047534  005002                        2$:     CLR     R2                      ;;CLEAR THE BCD NUMBER
047536  016001  047672                        MOV     $DTBL(R0),R1            ;;GET THE CONSTANT
047542  160105                        3$:     SUB     R1,R5                   ;;FORM THIS BCD DIGIT
047544  002402                                BLT     4$                      ;;BR IF DONE
047546  005202                                INC     R2                      ;;INCREASE THE BCD DIGIT BY 1
047550  000774                                BR      3$
047552  060105                        4$:     ADD     R1,R5                   ;;ADD BACK THE CONSTANT
047554  005702                                TST     R2                      ;;CHECK IF BCD DIGIT=0
047556  001002                                BNE     5$                      ;;FALL THROUGH IF 0
047560  105716                                TSTB    (SP)                    ;;STILL DOING LEADING 0'S?
047562  100407                                BMI     7$                      ;;BR IF YES
047564  106316                        5$:     ASLB    (SP)                    ;;MSD?
047566  103003                                BCC     6$                      ;;BR IF NO
047570  116663  000001  177777                MOVB    1(SP),-1(R3)            ;;YES--SET THE SIGN
047576  052702  000060                6$:     BIS     #'0,R2                  ;;MAKE THE BCD DIGIT ASCII
047602  052702  000040                7$:     BIS     #' ,R2                  ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
047606  110223                                MOVB    R2,(R3)+                ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
047610  005720                                TST     (R0)+                   ;;JUST INCREMENTING
047612  020027  000010                        CMP     R0,#10                  ;;CHECK THE TABLE INDEX
047616  002746                                BLT     2$                      ;;GO DO THE NEXT DIGIT
047620  003002                                BGT     8$                      ;;GO TO EXIT
047622  010502                                MOV     R5,R2                   ;;GET THE LSD
047624  000764                                BR      6$                      ;;GO CHANGE TO ASCII
047626  105726                        8$:     TSTB    (SP)+                   ;;WAS THE LSD THE FIRST NON-ZERO?
047630  100003                                BPL     9$                      ;;BR IF NO
047632  116663  177777  177776                MOVB    -1(SP),-2(R3)           ;;YES--SET THE SIGN FOR TYPING
047640  105013                        9$:     CLRB    (R3)                    ;;SET THE TERMINATOR
047642  012605                                MOV     (SP)+,R5                ;;POP STACK INTO R5
047644  012603                                MOV     (SP)+,R3                ;;POP STACK INTO R3
047646  012602                                MOV     (SP)+,R2                ;;POP STACK INTO R2
047650  012601                                MOV     (SP)+,R1                ;;POP STACK INTO R1
047652  012600                                MOV     (SP)+,R0                ;;POP STACK INTO R0
047654  104400  047702                        TYPE    ,$DBLK                  ;;NOW TYPE THE NUMBER
047660  016666  000002  000004                MOV     2(SP),4(SP)             ;;ADJUST THE STACK
```

```
047670  000002                               RTI                                 ;;RETURN TO USER
047672  023420                       $DTBL:  10000.
047674  001750                               1000.
047676  000144                               100.
047700  000012                               10.
047702  000004                       $DBLK:  .BLKW   4
                                     ;;***********************************************************


                                     .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE

                                     ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
                                     ;*UNSIGNED OCTAL ASCIZ NUMBER.
                                     ;*CALL
                                     ;*      MOV     #PNTR,-(SP)         ;;POINTER TO LOW WORD OF BINARY NUMBER
                                     ;*      JSR     PC,@#$DB20          ;;CALL THE ROUTINE
                                     ;*      RETURN                      ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK


047712  104412                       $DB20:  SAVREG                      ;;SAVE ALL REGISTERS
047714  016601  000002                       MOV     2(SP),R1            ;;PICKUP THE POINTER TO LOW WORD
047720  012705  050047                       MOV     #$OCTVL+13.,R5      ;;POINTER TO DATA TABLE
047724  012704  000014                       MOV     #12.,R4             ;;DO ELEVEN CHARACTERS
047730  012703  177770                       MOV     #↑C7,R3             ;;MASK
047734  012100                               MOV     (R1)+,R0            ;;LOWER WORD
047736  012101                               MOV     (R1)+,R1            ;;HIGH WORD
047740  005002                               CLR     R2                  ;;TERMINATOR
047742  110245                       1$:     MOVB    R2,-(R5)            ;;PUT CHARACTER IN DATA TABLE
047744  010002                               MOV     R0,R2               ;;GET THIS DIGIT
047746  005304                               DEC     R4                  ;;COUNT THIS CHARACTER
047750  003016                               BGT     3$                  ;;BR IF NOT THE LAST DIGIT
047752  001414                               BEQ     2$                  ;;BR IF IT IS THE LAST DIGIT
047754  005205                               INC     R5                  ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
047756  010566  000002                       MOV     R5,2(SP)            ;;ASCIZ CHAR. & PUT IT ON THE STACK
047762  122765  000061  000003               CMPB    #61,3(R5)           ;LAST NUBER LEGAL?
047770  002003                               BGE     4$                  ;BRANCH IF YES
047772  112765  000060  000003               MOVB    #60,3(R5)           ;MAKE IT ZERO
050000  104414                       4$:     RESREG                      ;;RESTORE ALL REGISTERS
050002  000207                               RTS     PC                  ;;RETURN TO USER
050004  006203                       2$:     ASR     R3                  ;;POSITION THE MASK FOR THE LAST DIGIT
050006  006001                       3$:     ROR     R1                  ;;POSITION THE BINARY NUMBER FOR
050010  006000                               ROR     R0                  ;;        THE NEXT OCTAL DIGIT
050012  006001                               ROR     R1
050014  006000                               ROR     R0
050016  006001                               ROR     R1
050020  006000                               ROR     R0
050022  040302                               BIC     R3,R2               ;;MASK OUT ALL JUNK
050024  062702  000060                       ADD     #'0,R2              ;;MAKE THIS CHAR. ASCII
050030  000744                               BR      1$                  ;;GO PUT IT IN THE DATA TABLE
050032  000016                       $OCTVL: .BLKB   14.                 ;;RESERVE DATA TABLE
                                     ;;***********************************************************


                                     .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES

                                     ;*SAVE R0-R5
                                     ;*CALL:
```

```
                                  ;*UPON RETURN FROM $SAVREG THE STACK WILL LOOK LIKE:
                                  ;*
                                  ;*TOP---(+16)
                                  ;* +2---(+18)
                                  ;* +4---R5
                                  ;* +6---R4
                                  ;* +8---R3
                                  ;*+10---R2
                                  ;*+12---R1
                                  ;*+14---R0

050050                            $SAVREG:
050050  010046                            MOV     R0,-(SP)            ;;PUSH R0 ON STACK
050052  010146                            MOV     R1,-(SP)            ;;PUSH R1 ON STACK
050054  010246                            MOV     R2,-(SP)            ;;PUSH R2 ON STACK
050056  010346                            MOV     R3,-(SP)            ;;PUSH R3 ON STACK
050060  010446                            MOV     R4,-(SP)            ;;PUSH R4 ON STACK
050062  010546                            MOV     R5,-(SP)            ;;PUSH R5 ON STACK
050064  016646  000022                    MOV     22(SP),-(SP)        ;;SAVE PS OF MAIN FLOW
050070  016646  000022                    MOV     22(SP),-(SP)        ;;SAVE PC OF MAIN FLOW
050074  016646  000022                    MOV     22(SP),-(SP)        ;;SAVE PS OF CALL
050100  016646  000022                    MOV     22(SP),-(SP)        ;;SAVE PC OF CALL
050104  000002                            RTI

                                  ;*RESTORE R0-R5
                                  ;*CALL:
                                  ;*        RESREG
050106                            $RESREG:
050106  012666  000022                    MOV     (SP)+,22(SP)        ;;RESTORE PC OF CALL
050112  012666  000022                    MOV     (SP)+,22(SP)        ;;RESTORE PS OF CALL
050116  012666  000022                    MOV     (SP)+,22(SP)        ;;RESTORE PC OF MAIN FLOW
050122  012666  000022                    MOV     (SP)+,22(SP)        ;;RESTORE PS OF MAIN FLOW
050126  012605                            MOV     (SP)+,R5            ;;POP STACK INTO R5
050130  012604                            MOV     (SP)+,R4            ;;POP STACK INTO R4
050132  012603                            MOV     (SP)+,R3            ;;POP STACK INTO R3
050134  012602                            MOV     (SP)+,R2            ;;POP STACK INTO R2
050136  012601                            MOV     (SP)+,R1            ;;POP STACK INTO R1
050140  012600                            MOV     (SP)+,R0            ;;POP STACK INTO R0
050142  000002                            RTI
                                  ;;*************************************************************
                                  .SBTTL  CONVERT FLOATING BINARY TO OCTAL ASCIZ
                                  ;*
                                  ;*THIS ROUTINE CONVERTS A 32 BIT FLOATING NUMBER TO AN OCTAL
                                  ;*ASCIZ STRING IN THE FOLLOWING FORMAT:
                                  ;*
                                  ;*                  W  XXX  YYY ZZZZZZ
                                  ;*
                                  ;*        WHERE     W = SIGN BIT
                                  ;*                  X = 8-BIT EXPONENT (RIGHT JUSTIFIED)
                                  ;*                  Y = FRACTION BITS <57:51> (RIGHT JUSTIFIED)
                                  ;*                  Z = FRACTION BITS <50:35>
                                  ;*
                                  ;*IT IS ENTERED BY A TRAP CALL WITH THE ADDRESS OF THE FLOATING
                                  ;*NUMBER IN THE WORD FOLLOWING THE CALL.
                                  ;*IT RETURNS WITH THE ADDRESS OF THE ASCIZ STRING ON THE STACK.
```

```
050144  104412                          $FL20:  SAVREG
050146  017600  000000                          MOV     @(SP),R0                ;GET ADDRESS OF DATA
050152  062716  000002                          ADD     #2,(SP)                 ;ADJUST RETURN PC
050156  016001  000002                          MOV     2(R0),R1                ;PUT SECOND DATA WORD IN R1
050162  011000                                  MOV     (R0),R0                 ;PUT FIRST DATA WORD IN R0
050164  012704  001357                          MOV     #$FLBUFF+23,R4          ;GET ADDRESS OF BUFFER END IN R4
050170  112744  000000                          MOVB    #0,-(R4)                ;PUT TERMINATOR IN BUFFER
050174  012705  000005                          MOV     #5,R5                   ;SET SOB COUNT FOR FRACTION DIGITS
050200  010103                          1$:     MOV     R1,R3                   ;GET LSB'S OF FRACTION
050202  042703  177770                          BIC     #^C7,R3                 ;SAVE LS 3 BITS
050206  062703  000060                          ADD     #60,R3                  ;MAKE THEM ASCII
050212  110344                                  MOVB    R3,-(R4)                ;STORE IN BUFFER
050214  073027  177775                          ASHC    #-3,R0                  ;SHIFT NUMBER TO NEXT 3 BITS
050220  077511                                  SOB     R5,1$                   ;CONTINUE FOR 7 DIGITS
050222  010103                                  MOV     R1,R3                   ;GET NEXT DIGITS
050224  042703  177776                          BIC     #^C1,R3                 ;ONLY WANT 1 BIT
050230  062703  000060                          ADD     #60,R3                  ;MAKE THEM ASCII
050234  110344                                  MOVB    R3,-(R4)                ;STORE IN BUFFER
050236  112744  000040                          MOVB    #40,-(R4)               ;PUT SPACE IN BUFFER
050242  073027  177777                          ASHC    #-1,R0
050246  012705  000002                          MOV     #2,R5                   ;SET SOB COUNT
050252  010103                          3$:     MOV     R1,R3                   ;GET LOW WORD
050254  042703  177770                          BIC     #^C7,R3                 ;MASK 3 BITS
050260  062703  000060                          ADD     #60,R3                  ;MAKE THEM ASCII
050264  110344                                  MOVB    R3,-(R4)                ;PUT IN BUFFER
050266  073027  177775                          ASHC    #-3,R0                  ;GET NEXT 3 BITS
050272  077511                                  SOB     R5,3$                   ;CONVERT THEM
050274  010103                                  MOV     R1,R3                   ;
050276  042703  177776                          BIC     #^C1,R3                 ;ONLY WANT 1 BIT
050302  062703  000060                          ADD     #60,R3                  ;MAKE IT ASCII
050306  110344                                  MOVB    R3,-(R4)                ;PUT IN BUFFER
050310  112744  000040                          MOVB    #40,-(R4)               ;PUT SPACE IN BUFFER
050314  112744  000040                          MOVB    #40,-(R4)
050320  072127  177777                          ASH     #-1,R1                  ;GET FIRST 3 BITS OF EXPONENT
050324  012705  000002                          MOV     #2,R5                   ;SET SOB COUNT FOR 2 DIGITS
050330  010103                          2$:     MOV     R1,R3                   ;GET LSB'S OF EXPONENT
050332  042703  177770                          BIC     #^C7,R3                 ;SAVE 3 BITS
050336  062703  000060                          ADD     #60,R3                  ;MAKE THEM ASCII
050342  110344                                  MOVB    R3,-(R4)                ;STORE IN BUFFER
050344  072127  177775                          ASH     #-3,R1                  ;GET NEXT 3 BITS
050350  077511                                  SOB     R5,2$                   ;CONTINUE
050352  010103                                  MOV     R1,R3                   ;GET LAST 2 BITS OF EXPONENT
050354  042703  177774                          BIC     #^C3,R3                 ;MAKE SURE ONLY 2 BITS
050360  062703  000060                          ADD     #60,R3                  ;MAKE THEM ASCII
050364  110344                                  MOVB    R3,-(R4)                ;STORE IN BUFFER
050366  112744  000040                          MOVB    #40,-(R4)               ;PUT SPACE IN BUFFER
050372  112744  000040                          MOVB    #40,-(R4)
050376  042700  177776                          BIC     #^C1,R0                 ;GET SIGN BIT (IT WAS EXTENDED)
050402  062700  000060                          ADD     #60,R0                  ;MAKE IT ASCII
050406  110044                                  MOVB    R0,-(R4)                ;PUT IT IN THE BUFFER
050410  104414                                  RESREG
050412  011646                                  MOV     (SP),-(SP)              ;SAVE RETURN PC
050414  016666  000004  000002                  MOV     4(SP),2(SP)             ;AND RETURN PSW
050422  012766  001334  000004                  MOV     #$FLBUFF,4(SP)          ;PUT BUFFER ADDRESS ON STACK
050430  000006                                  RTT                             ;RETURN
```

```
                                        ;;*************************************************************
                                        .SBTTL  CONVERT FLOATING DOUBLE BINARY TO OCTAL ASCIZ
                                        ;*
                                        ;*THIS ROUTINE CONVERTS A 64 BIT FLOATING NUMBER TO AN OCTAL
                                        ;*ASCIZ STRING IN THE FOLLOWING FORMAT:
                                        ;*
                                        ;*                  U  VVV  WWW XXXXXX YYYYYY ZZZZZZ
                                        ;*
                                        ;*      WHERE       U = SIGN BIT
                                        ;*                  V = 8-BIT EXPONENT (RIGHT JUSTIFIED)
                                        ;*                  W = FRACTION BITS<57:51> (RIGHT JUSTIFIED)
                                        ;*                  X = FRACTION BITS <50:35>
                                        ;*                  Y = FRACTION BITS <34:19>
                                        ;*                  Z = FRACTION BITS <18:03>
                                        ;*
                                        ;*IT IS ENTERED BY A TRAP CALL WITH THE ADDRESS OF THE FLOATING
                                        ;*NUMBER IN THE WORD FOLLOWING THE CALL.
                                        ;*IT RETURNS WITH THE ADDRESS OF THE ASCIZ STRING ON THE STACK.
                                        ;;*************************************************************
050432  104412                          $FLD20: SAVREG
050434  017667  000000  000006                  MOV     @(SP),1$        ;GET ADDRESS OF DATA TO CONVERT
050442  062716  000002                          ADD     #2,(SP)         ;ADJUST RETURN PC
050446  104416                                  FL20                    ;CONVERT MS 32 BITS
050450  000000                          1$:     .WORD
050452  012600                                  MOV     (SP)+,R0        ;GET ADDRESS OF CONVERTED DATA
050454  010067  130720                          MOV     R0,$BUFF        ;SAVE IT
050460  062700  000041                          ADD     #41,R0          ;ADJUST TO END OF BUFFER
050464  105040                                  CLRB    -(R0)           ;PUT TERMINATOR IN BUFFER
050466  016701  177756                          MOV     1$,R1           ;GET ADDRESS OF DATA TO CONVERT
050472  062701  000004                          ADD     #4,R1           ;ADJUST TO LOWER 32 BITS
050476  012102                                  MOV     (R1)+,R2        ;SAVE THE DATA
050500  012103                                  MOV     (R1)+,R3
050502  012701  000002                          MOV     #2,R1           ;SET LOOP COUNT
050506  012704  000005                  3$:     MOV     #5,R4           ;SET LOOP COUNT
050512  010305                          4$:     MOV     R3,R5           ;GET LS 32 BITS OF DATA
050514  042705  177770                          BIC     #^C7,R5         ;MASK 3 BITS
050520  062705  000060                          ADD     #60,R5          ;MAKE THEM ASCII
050524  110540                                  MOVB    R5,-(R0)        ;PUT IN BUFFER
050526  073227  177775                          ASHC    #-3,R2          ;GET NEXT 3 BITS
050532  077411                                  SOB     R4,4$           ;CONTINUE
050534  010305                                  MOV     R3,R5           ;GET LS 32 BITS
050536  042705  177776                          BIC     #^C1,R5         ;ONLY WANT 1 BIT
050542  062705  000060                          ADD     #60,R5          ;MAKE IT ASCII
050546  110540                                  MOVB    R5,-(R0)        ;PUT IN TABLE
050550  112740  000040                          MOVB    #40,-(R0)       ;PUT SPACE IN TABLE
050554  073227  177777                          ASHC    #-1,R2
050560  077126                                  SOB     R1,3$           ;CONVERT NEXT 16 BITS
050562  104414                                  RESREG
050564  011646                                  MOV     (SP),-(SP)      ;ADJUST STACK
050566  016666  000004  000002                  MOV     4(SP),2(SP)     ;TO RETURN WITH ADDRESS
050574  016766  130600  000004                  MOV     $BUFF,4(SP)     ;OF BUFFER ON STACK
050602  000006                                  RTT                     ;RETURN

                                        ;;*************************************************************
```

```

                                       ;*THIS ROUTINE IS A DOUBLE PRECISION PSEUDO RANDOM NUMBER GENERATOR
                                       ;*WITH A RANGE OF 0 TO 2(+33)-1.
                                       ;*CALL:
                                       ;*      JSR     PC,$RAND                ;;CALL THE ROUTINE
                                       ;*      RETURN                          ;;RETURN HERE THE RANDOM
                                       ;*                                      ;;NUMBER WILL BE IN
                                       ;*                                      ;;$HINUM,$LONUM

        050604                         $RAND:
        050604  010046                         MOV     R0,-(SP)                ;;PUSH R0 ON STACK
        050606  010146                         MOV     R1,-(SP)                ;;PUSH R1 ON STACK
        050610  010246                         MOV     R2,-(SP)                ;;PUSH R2 ON STACK
        050612  016700  130704                 MOV     $LONUM,R0               ;;SET R0 WITH LOW
        050616  016701  130702                 MOV     $HINUM,R1               ;;SET R1 WITH HIGH
        050622  012702  177771                 MOV     #-7,R2                  ;;SET SHIFT COUNT
        050626  006300                 1$:     ASL     R0                      ;;SHIFT R0 LEFT AND
        050630  006101                         ROL     R1                      ;;ROTATE CARRY INTO R1 AND
        050632  005202                         INC     R2                      ;;CHECK FOR DONE
        050634  001374                         BNE     1$                      ;;CONTINUE SHIFT LOOP
        050636  066700  130660                 ADD     $LONUM,R0               ;;ADD NUMBER TO MAKE X 129
        050642  005501                         ADC     R1                      ;;PROPOGATE CARRY
        050644  066701  130654                 ADD     $HINUM,R1               ;;ADD NUMBER TO MAKE X 129
        050650  062700  001057                 ADD     #1057,R0                ;;ADD LOW CONSTANT
        050654  005501                         ADC     R1                      ;;PROPOGATE CARRY
        050656  062701  047401                 ADD     #47401,R1               ;;ADD HIGH CONSTANT
        050662  010067  130634                 MOV     R0,$LONUM               ;;SAVE R0
        050666  010167  130632                 MOV     R1,$HINUM               ;;SAVE R1
        050672  012602                         MOV     (SP)+,R2                ;;POP STACK INTO R2
        050674  012601                         MOV     (SP)+,R1                ;;POP STACK INTO R1
        050676  012600                         MOV     (SP)+,R0                ;;POP STACK INTO R0
        050700  000207                         RTS     PC                      ;;RETURN
                                       ;;*****************************************************************
                                       .SBTTL  FLOATING POINT NUMBER GENERATOR
                                       ;*      THIS ROUTINE GENERATES TWO RANDOM FLOATING POINT NUMBERS
                                       ;*      IN EITHER SINGLE OR DOUBLE PRECISION. FOR SINGLE PRECISION
                                       ;*      THE NUMBERS ARE STORED IN $TMP0 AND $TMP2. DOUBLE PRECISION
                                       ;*      NUMBERS ARE STORED IN $TMP0 AND $TMP4.
                                       ;*      IN EITHER SINGLE OR DOUBLE THE EXTENDED EXPONENT IS STORED
                                       ;*      IN $REG0 AND $REG1.
                                       ;;*****************************************************************
        050702  012767  000002  000130 FLTDBL: MOV     #2,SOBDBL               ;SET LOOP FOR 2, FOUR WORD NUMBERS
        050710  016700  000124         FLTSGL: MOV     SOBDBL,R0               ;SET WORD LENGTH LOOP
        050714  012702  001276                 MOV     #$TMP0,R2               ;GET ADDRESS TO STORE WORDS IN
        050720  012701  000002         2$:     MOV     #2,R1                   ;SET NUMBER OF WORDS TO 2
        050724  004767  177654         1$:     JSR     PC,$RAND                ;GET RANDOM NUMBER
        050730  022701  000002                 CMP     #2,R1                   ;FIRST TIME?
        050734  001404                         BEQ     3$                      ;BRANCH IF YES
        050736  022767  000002  000074         CMP     #2,SOBDBL               ;DOUBLE PRECISION?
        050744  001407                         BEQ     4$                      ;BRANCH IF YES
        050746  016703  130552         3$:     MOV     $HINUM,R3               ;GET EXPONENT PART
        050752  042703  000177                 BIC     #177,R3                 ;CHECK FOR MINUS ZERO
        050756  022703  100000                 CMP     #BIT15,R3
        050762  001760                         BEQ     1$                      ;BRANCH IF MINUS ZERO
        050764  016722  130534         4$:     MOV     $HINUM,(R2)+            ;SAVE HINUM
```

```
050774  077125                           SOB     R1,1$           ;CONTINUE
050776  077030                           SOB     R0,2$           ;CONTINUE FOR DOUBLE PREC
051000  012746  001276                   MOV     #$TMP0,-(SP)    ;PUT ADDRESS OF NUMBER ON STACK
051004  012746  001002                   MOV     #1002,-(SP)     ;PUT CONTROL WORD ON STACK
051010  022767  000002  000022           CMP     #2,SOBDBL       ;DOUBLE PREC?
051016  001002                           BNE     5$              ;BRANCH IF NO
051020  012716  001004                   MOV     #1004,(SP)      ;CHANGE CONTROL WORD
051024  004767  000012          5$:      JSR     PC,EXPEXT       ;CALCULATE EXT EXPONENTS
051030  012767  000001  000002           MOV     #1,SOBDBL       ;INIT SOBDBL FOR SINGLE PREC
051036  000207                           RTS     PC              ;RETURN
051040  000001                  SOBDBL: .WORD    1
                                ;*******************************************************
                                .SBTTL  FLOATING POINT EXPONENT EXTENSION
                                ;*      THIS ROUTINE CONVERTS THE ACTUAL EXPONENT OF A FLOATING POINT
                                ;*      NUMBER INTO AN ACTUAL EXPONENT OF 200 AND AN EXTENDED
                                ;*      EXPONENT EQUAL TO THE DIFFERANCE BETWEEN THE ORIGINAL
                                ;*      ACTUAL EXPONENT AND 200.
                                ;*
                                ;*      THE ROUTINE IS ENTERED WITH A CONTROL WORD ON THE STACK.
                                ;*      BIT 15 OF THE CONTROL WORD INDICATES WHETHER THE NUMBER
                                ;*      IS IN MEMORY (<15>=0) OR IN AN ACCUMULATOR (<15>=1).
                                ;*      IF THE NUMBER IS IN AN ACCUMULATOR, BITS <9:8> INDICATE
                                ;*      THE ACCUMULATOR NUMBER. IF THE NUMBER(S) IS IN MEMORY,
                                ;*      BITS <9:8> INDICATE THE NUMBER OF NUMBERS TO CONVERT AND
                                ;*      BITS <2:0> INDICATE THE WORD LENGTH OF THE NUMBER(S).
                                ;*      IN THE CASE OF A MEMORY CONVERSION, THE ADDRESS OF THE
                                ;*      FIRST WORD TO CONVERT IS ALSO ON THE STACK (PRECEDING
                                ;*      THE CONTROL WORD).
                                ;*******************************************************
051042  012605                  EXPEXT: MOV     (SP)+,R5        ;SAVE RETURN PC
051044  012600                          MOV     (SP)+,R0        ;GET CONTROL WORD
051046  100437                          BMI     1$              ;BRANCH IF ACC CONVERSION
051050  012601                          MOV     (SP)+,R1        ;GET START ADDRESS
051052  162700  000400                  SUB     #400,R0
051056  012702  001276                  MOV     #$TMP0,R2       ;GET OFFSET FROM $TMP0
051062  160102                          SUB     R1,R2
051064  005402                          NEG     R2
051066  006202                          ASR     R2
051070  062702  001256                  ADD     #$REG0,R2       ;GEN ADDRESS OF EXT WORD
051074  011103                  3$:     MOV     (R1),R3         ;GET DATA
051076  042703  100177                  BIC     #100177,R3      ;GET EXPONENT
051102  072327  177771                  ASH     #-7,R3          ;RIGHT JUSTIFY EXPONENT
051106  162703  000200                  SUB     #200,R3         ;CONVERT TO 2'S COMPLIMENT
051112  010312                          MOV     R3,(R2)         ;ADD TO EXTENDED EXPONENT
051114  042711  077600                  BIC     #77600,(R1)     ;MAKE ACTUAL
051120  052711  040000                  BIS     #BIT14,(R1)     ;EXPONENT 200
051124  162700  000400                  SUB     #400,R0         ;ANY MORE WORDS?
051130  100435                          BMI     2$              ;BRANCH IF NO
051132  110003                          MOVB    R0,R3           ;GET WORD LENGTH
051134  006303                          ASL     R3
051136  060301                          ADD     R3,R1           ;SELECT NEXT NUMBER ADDRESS
051140  062702  000002                  ADD     #2,R2           ;SELECT NEXT EXTENDED ADDRESS
051144  000753                          BR      3$              ;CONTINUE

                                ;ACCUMULATOR CONVERSION
051146  072027  177776          1$:     ASH     #-2,R0          ;GET ACCUMULATOR NUMBER
```

```
051156  010002                           MOV     R0,R2                  ;GENERATE
051160  072227  177773                   ASH     #-5,R2                 ;ADDRESS OF
051164  062702  001402                   ADD     #$AC0,R2               ;EXTENDED EXPONENT
051170  042767  000300  000004           BIC     #300,5$                ;GENERATE INSTRUCTION
051176  050067  000000                   BIS     R0,5$                  ;TO GET EXPONENT
051202  175003                   5$:     STEXP   AC0,R3                 ;GET EXPONENT
051204  060312                           ADD     R3,(R2)                ;ADD TO EXTENDED EXPONENT
051206  005003                           CLR     R3
051210  042767  000300  000004           BIC     #300,4$                ;GENERATE INSTRUCTION
051216  050067  000000                   BIS     R0,4$                  ;TO LOAD EXPONENT BACK TO ACC
051222  176403                   4$:     LDEXP   R3,AC0                 ;LOAD EXPONENT OF 200
051224  010546                   2$:     MOV     R5,-(SP)               ;RESTORE RETURN PC
051226  000207                           RTS     PC                     ;RETURN
                                 ;;*****************************************************************

                                 .SBTTL  POWER DOWN AND UP ROUTINES

                                 ;POWER DOWN ROUTINE
051230  012737  051356  000024   $PWRDN: MOV     #$ILLUP,@#PWRVEC       ;;SET FOR FAST UP
051236  012737  000340  000026           MOV     #340,@#PWRVEC+2        ;;PRIO:7
051244  010046                           MOV     R0,-(SP)               ;;PUSH R0 ON STACK
051246  010146                           MOV     R1,-(SP)               ;;PUSH R1 ON STACK
051250  010246                           MOV     R2,-(SP)               ;;PUSH R2 ON STACK
051252  010346                           MOV     R3,-(SP)               ;;PUSH R3 ON STACK
051254  010446                           MOV     R4,-(SP)               ;;PUSH R4 ON STACK
051256  010546                           MOV     R5,-(SP)               ;;PUSH R5 ON STACK
051260  010667  000076                   MOV     SP,$SAVR6              ;;SAVE SP
051264  012737  051276  000024           MOV     #$PWRUP,@#PWRVEC       ;;SET UP VECTOR
051272  000000                           HALT
051274  000776                           BR      .-2                    ;;HANG UP

                                 ;POWER UP ROUTINE
051276  016706  000060           $PWRUP: MOV     $SAVR6,SP              ;;GET SP
051302  005067  000054                   CLR     $SAVR6                 ;;WAIT LOOP FOR THE TTY
051306  005267  000050           1$:     INC     $SAVR6                 ;;WAIT FOR THE INC
051312  001375                           BNE     1$                     ;;OF WORD
051314  012605                           MOV     (SP)+,R5               ;;POP STACK INTO R5
051316  012604                           MOV     (SP)+,R4               ;;POP STACK INTO R4
051320  012603                           MOV     (SP)+,R3               ;;POP STACK INTO R3
051322  012602                           MOV     (SP)+,R2               ;;POP STACK INTO R2
051324  012601                           MOV     (SP)+,R1               ;;POP STACK INTO R1
051326  012600                           MOV     (SP)+,R0               ;;POP STACK INTO R0
051330  012737  051230  000024           MOV     #$PWRDN,@#PWRVEC       ;;SET UP THE POWER DOWN VECTOR
051336  012737  000340  000026           MOV     #340,@#PWRVEC+2        ;;PRIO:7
051344  104400                           TYPE                           ;REPORT THE POWER FAILURE
051346  051364                   $PWRMG: .WORD   $POWER                 ;;POWER FAIL MESSAGE POINTER
051350  012716                           MOV     (PC)+,(SP)             ;;RESTART AT START
051352  003212                   $PWRAD: .WORD   START                  ;;RESTART ADDRESS
051354  000002                           RTI
051356  000000                   $ILLUP: HALT                           ;;THE POWER UP SEQUENCE WAS STARTED
051360  000776                           BR      .-2                    ;; BEFORE THE POWER DOWN WAS COMPLETE
051362  000000                   $SAVR6: 0                              ;;PUT THE SP HERE
051364  005015  047520  042527   $POWER: .ASCIZ  <15><12>"POWER"
051372  000122
                                         .EVEN
```

```

                                      .SBTTL  TRAP DECODER

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

051374  010046                        $TRAP:  MOV     R0,-(SP)            ;;SAVE R0
051376  016600  000002                        MOV     2(SP),R0            ;;GET TRAP ADDRESS
051402  005740                                TST     -(R0)               ;;BACKUP BY 2
051404  111000                                MOVB    (R0),R0             ;;GET RIGHT BYTE OF TRAP
051406  016000  051414                        MOV     $TRPAD(R0),R0       ;;INDEX TO TABLE
051412  000200                                RTS     R0                  ;;GO TO ROUTINE


                                      .SBTTL  TRAP TABLE

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

                                      ;       ROUTINE
                                      ;       -------
051414                                $TRPAD:
051414  046436                                $TYPE    ;;CALL=TYPE      TRAP+0(104400)  TTY TYPEOUT ROUTINE
051416  047264                                $TYPOC   ;;CALL=TYPOC     TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
051420  047240                                $TYPOS   ;;CALL=TYPOS     TRAP+4(104404)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
051422  047300                                $TYPON   ;;CALL=TYPON     TRAP+6(104406)  TYPE OCTAL NUMBER (AS PER LAST CALL)
051424  047466                                $TYPDS   ;;CALL=TYPDS     TRAP+10(104410) TYPE DECIMAL NUMBER (WITH SIGN)
051426  050050                                $SAVREG  ;;CALL=SAVREG    TRAP+12(104412) SAVE R0-R5 ROUTINE
051430  050106                                $RESREG  ;;CALL=RESREG    TRAP+14(104414) RESTORE R0-R5 ROUTINE
051432  050144                                $FL20    ;;CALL=FL20      TRAP+16(104416)
051434  050432                                $FLD20   ;;CALL=FLD20     TRAP+20(104420)
                                      ;;*************************************************************
                                      .SBTTL  UNIBUS EXERCISER INITIALIZATION ROUTINE
                                      ;*THIS ROUTINE INITIALIZES THE BASE ADDRESS FOR THE
                                      ;*UNIBUS EXERCISER AND LOADS UP THE EXERCISER REGISTERS.
                                      ;;*************************************************************
051436  012703  001662                UBEINIT:MOV     #UBESAV,R3          ;GET ADDRESS OF NEXT UBE ADRES
051442  005023                                CLR     (R3)+               ;INITIALIZE
051444  005023                                CLR     (R3)+
051446  005023                                CLR     (R3)+
051450  005013                                CLR     (R3)


                                      ;SET UP THE UBE AND START IT
051452  012702  002230                        MOV     #UBETBL,R2          ;GET ADDRESS OF UBE TABLE
051456  005072  000010                        CLR     @10(R2)             ;CLEAR ALL ERRORS
051462  012772  043520  000012                MOV     #UBESRV,@12(R2)     ;SET UP UBE VECTOR
051470  012772  000340  000014                MOV     #PR7,@14(R2)        ;SET UP UBE VECTOR PSW
051476  012732  170000                        MOV     #170000,@(R2)+      ;SET CC FOR 2K WORD TRANSFER
                                                                          ;UBE IS DOING BYTE TRANSFERS
051502  012746  000003                        MOV     #3,-(SP)            ;PUT DEVICE ID IN STACK
051506  012746  001666                        MOV     #UBEADR,-(SP)       ;PUT ADDRESS OF PHYSICAL BA ON STACK
051512  004737  052054                        JSR     PC,@#GETMAP         ;GO GET MAP REGISTER
```

```
051522  013732  001670                  MOV     @#UBEADR+2,@(R2)+;LOAD ADR BITS 16 & 17
051526  052737  000040  172516          BIS     #40,@#SR3       ;ENABLE MAP
051534  000207                          RTS     PC              ;RETURN
                                ;;*************************************************************
                                .SBTTL  CONVERT UNIBUS VIRTUAL ADDRESS TO PHYSICAL ADDRESS
                                ;*      THIS ROUTINE CONVERTS THE CONTENTS OF LOCATIONS
                                ;*      "ERRBA" AND "ERRBA+2" FROM A VIRTUAL 18-BIT ADDRESS
                                ;*      TO A PHYSICAL 22-BIT ADDRESS AS MAPPED BY THE APPROPRIATE
                                ;*      MAP REGISTER. THE 22-BIT ADDRESS IS STORED IN LOCATIONS
                                ;*      "PA2116" AND "PA1500".
                                ;;*************************************************************
051536  104412                  PHYMAP: SAVREG
051540  013703  001672                  MOV     @#ERRBA,R3      ;GET BUS ADDRESS <15:00>
051544  013702  001674                  MOV     @#ERRBA+2,R2    ;GET BUS ADDRESS <17:16>
051550  042702  177774                  BIC     #177774,R2      ;
051554  032737  000040  172516          BIT     #BIT5,@#MMR3    ;MAP ON?
051562  001005                          BNE     1$              ;BRANCH IF YES
051564  010337  001476                  MOV     R3,@#PA1500     ;PHY ADR=BUS ADR
051570  010237  001500                  MOV     R2,@#PA2116
051574  000421                          BR      MAPEND
051576  010305                  1$:     MOV     R3,R5           ;SAVE ADR BITS <15:00>
051600  073227  000005                  ASHC    #5,R2           ;GET MAP REG SELECT BITS
051604  042702  000003                  BIC     #3,R2
051610  062702  170200                  ADD     #MAPLO,R2       ;FORM ADDRESS OF MAP REG
051614  012237  001476                  MOV     (R2)+,@#PA1500  ;GET CONTENTS OF MAP REG LO
051620  011237  001500                  MOV     (R2),@#PA2116   ;GET CONTENTS OF MAP REG HI
051624  042705  160000                  BIC     #160000,R5      ;FORM PHYSICAL ADDRESS
051630  060537  001476                  ADD     R5,@#PA1500     ;THAT TIMED OUT
051634  005537  001500                  ADC     @#PA2116        ;
051640  104414                  MAPEND: RESREG
051642  000207                          RTS     PC

                                ;;*************************************************************
                                .SBTTL  CONVERT A VIRTUAL ADDRESS TO A PHYSICAL ADDRESS
                                ;*      THIS ROUTINE CONVERTS A 16-BIT VIRTUAL ADDRESS TO A
                                ;*      22-BIT PHYSICAL ADDRESS. THE VIRTUAL ADDRESS IS
                                ;*      ASSUMED TO BE IN LOCATION "VADR" AND THE PHYSICAL
                                ;*      ADDRESS IS PLACED IN LOCATIONS "PA2116" AND "PA1500".
                                ;*
                                ;*      IF MEMORY MANAGEMENT IS OFF THE PHYSICAL ADDRESS IS
                                ;*      GENERATED BY SUBTRACTING THE CONTENTS OF LOCATION
                                ;*      "FACTOR" FROM THE VIRTUAL ADDRESS. THIS LOCATION
                                ;*      CONTAINS THE BYTE OFFSET BETWEEN THE RELOCATED CODE
                                ;*      AND THE NON-RELOCATED CODE.
                                ;*
                                ;*      IF MEMORY MANAGEMENT IS ON, THE CONTENTS OF THE
                                ;*      APPROPRIATE PAR REGISTER IS ADDED(AFTER ADJUSTMENT)
                                ;*      TO THE LEAST SIGNIFICANT 13 BITS OF THE VIRTUAL ADDRESS.
                                ;;*************************************************************
051644  104412                  CNVADR: SAVREG
051646  013703  001474                  MOV     @#VADR,R3               ;GET VIRTUAL ADDRESS TO CONVERT
051652  105737  001503                  TSTB    @#MMON                  ;IS MEMORY MGMT ON?
051656  001426                          BEQ     1$                      ;BRANCH IF NO
051660  005002                          CLR     R2
051662  073227  000003                  ASHC    #3,R2                   ;GET PAR SELECT BITS
```

```
051672  042703  160000                          BIC     #160000,R3                      ;MAKE SURE SIGN DIDN'T EXTEND
051676  006102                                  ROL     R2                              ;MAKE R2 EVEN FOR WORD ADDRESSING
051700  062702  172340                          ADD     #KIPAR0,R2                      ;GET ADDRESS OF PAR
051704  011205                                  MOV     (R2),R5                         ;GET PAR DATA
051706  005004                                  CLR     R4                              ;SETUP R4
051710  073427  000006                          ASHC    #6,R4                           ;SHIFT PAR DATA
051714  060305                                  ADD     R3,R5                           ;FORM PHYSICAL ADDRESS
051716  005504                                  ADC     R4
051720  010437  001500                  2$:     MOV     R4,@#PA2116                     ;SAVE PHYSICAL
051724  010537  001476                          MOV     R5,@#PA1500                     ;ADDRESS
051730  104414                                  RESREG
051732  000207                                  RTS     PC                              ;RETURN
051734  163703  001506                  1$:     SUB     @#FACTOR,R3                     ;FORM PHYSICAL ADDRESS
051740  005004                                  CLR     R4
051742  010305                                  MOV     R3,R5
051744  000765                                  BR      2$                              ;RETURN

                                        ;;****************************************************************
                                        .SBTTL  ROUTINE TO CHECK RELOCATED DATA
                                        ;*ROUTINE TO CHECK DATA RELOCATED
                                        ;*CALL: R0=     HIGHEST ADDRESS +2 OF SOURCE DATA
                                        ;*      R2=     HIGHEST ADDRESS +2 OF DEST DATA
                                        ;*      R5=     LOWEST ADDRESS OF THE SOURCE DATA
                                        ;*
                                        ;*
                                        ;*      THIS ROUTINE USES A COMPARE INSTRUCTION TO CHECK
                                        ;*      THE DATA THAT WAS RELOCATED. IF A PARITY ERROR OCCURS
                                        ;*      DURING THIS CHECK A SPECIAL ERROR MESSAGE IS TYPED
                                        ;*      INSTED OF THE UNEXPECTED TRAP MESSAGE.
                                        ;;****************************************************************
051746  012737  052010  000114          CHKDAT: MOV     #2$,@#CACHVEC                   ;SETUP PARITY VECTOR
051754  024042                                  CMP     -(R0),-(R2)                     ;CHECK DATA
051756  001013                                  BNE     99$+2
051760  005112                                  COM     (R2)                            ;COMPLEMENT DEST DATA
051762  005112                                  COM     (R2)                            ;TWICE
051764  021210                                  CMP     (R2),(R0)                       ;CHECK DATA
051766  001006                                  BNE     99$
051770  020005                          1$:     CMP     R0,R5                           ;BRANCH IF ALL DATA NOT CHECKED
051772  001365                                  BNE     CHKDAT
051774  012737  052642  000114                  MOV     #.PARSRV,@#CACHVEC                      ;RESTORE CACHVEC
052002  000207                                  RTS     PC                              ;RETURN
052004  000262                          99$:    SEV
052006  000207                                  RTS     PC
052010  013737  177744  001302          2$:     MOV     @#MEMERR,@#$TMP2                        ;SAVE ERROR REG
052016  013737  177740  001304                  MOV     @#LOADRS,@#$TMP3                        ;SAVE ERROR ADR
052024  013737  177742  001306                  MOV     @#HIADRS,@#$TMP4
052032  010237  001474                          MOV     R2,@#VADR
052036  010037  001276                          MOV     R0,@#$TMP0
052042  104005                                  ERROR   5
052044  012737  177777  177744                  MOV     #-1,@#MEMERR                            ;CLEAR ERROR REG
052052  000754                                  BR      99$                                     ;RETURN


                                        ;;****************************************************************
                                        .SBTTL  ROUTINE TO GET A MAP REGISTER
                                        ;*THIS ROUTINE TAKES AN 18 BIT RANDOM NUMBER, FINDS TWO
                                        ;*CONSECUTIVE MAP REGISTERS THAT ARE NOT IN USE, LOADS THE
```

```
                                         ;*AND THE NUMBER + 4K, AND RETURNS A NEW BUS ADDRESS, BASED
                                         ;*ON THE RANDOM NUMBER.
                                         ;*
                                         ;*        MAP REGISTERS 0 AND 1 ARE NOT USED IF THE PROGRAM IS
                                         ;*        RUNNING ON ACT11. THIS ALLOWS "MOTHER" TO ACCESS THE
                                         ;*        END OF PASS HOOKS.
                                         ;*
                                         ;*        THE MAP TABLE (MAPTBL) CONTAINS 4 BYTES, ONE FOR EACH
                                         ;*        UNIBUS DEVICE. IF THE UBE IS PRESENT IT USES THE
                                         ;*        4TH BYTE. WHEN A REGISTER IS ASSIGNED TO A DEVICE,
                                         ;*        THE LOWER 4 ADDRESS BITS OF THAT REGISTER ARE PLACED
                                         ;*        IN THE TABLE. WHEN A DEVICE REQUESTS A REGISTER PAIR
                                         ;*        THIS TABLE IS THEN SEARCHED TO SEE IF THE REGISTER
                                         ;*        PAIR IS IN USE.
                                         ;*        ENTER WITH:
                                         ;*                4(SP)=DEVICE ID
                                         ;*                2(SP)=ADDRESS OF THE PHYSICAL ADDRESS
                                         ;*******************************************************************
052054  016600  000004                   GETMAP: MOV     4(SP),R0                ;GET DIVICE ID
052060  016601  000002                           MOV     2(SP),R1                ;GET ADR OF PHY ADR
052064  013746  177776                           MOV     @#PSW,-(SP)             ;SAVE CURRENT PRIORITY
052070  005116                                   COM     (SP)                    ;MAKE IT READY FOR RESTORE
052072  042716  177437                           BIC     #↑CPR7,(SP)
052076  000237                                   SPL     7                       ;IF THIS IS RK CALL, LOCK OUT UBE
052100  104412                                   SAVREG
052102  012137  001226                           MOV     (R1)+,@#$GDDAT          ;SAVE PHYSICAL
052106  012137  001230                           MOV     (R1)+,@#$BDDAT          ;ADDRESS
052112  004737  050604                   2$:     JSR     PC,@#$RAND              ;GET RANDOM NUMBER
052116  013702  001524                           MOV     @#$HINUM,R2             ;GET HIGH RANDOM NUMBER
052122  013703  001522                           MOV     @#$LONUM,R3             ;GET LOW RANDOM NUMBER
052126  073227  177764                           ASHC    #-14,R2                 ;CONVERT TO 20 BIT NUMBER
052132  042702  177760                           BIC     #177760,R2              ;GET RID OF 11 BITS OF SIGN EXT
052136  022702  000016                           CMP     #16,R2                  ;LEGAL MAP REG SELECT?
052142  100001                                   BPL     3$                      ;BRANCH IF YES
052144  000762                                   BR      2$                      ;TRY AGAIN
052146  005737  001504                   3$:     TST     @#QV                    ;ACT11 (QV OR AUTO)?
052152  001403                                   BEQ     4$                      ;BRANCH IF NO
052154  122702  000000                           CMPB    #0,R2                   ;MAP SELECT 0?
052160  001754                                   BEQ     2$                      ;BRANCH IF YES.(ACT MUST
                                                                                 ;USE THIS MAP REG)
052162  010204                           4$:     MOV     R2,R4                   ;SAVE MAP SELECT BITS
052164  042703  100000                           BIC     #BIT15,R3               ;CLEAR SELECT BIT 0
052170  073227  177776                           ASHC    #-2,R2                  ;FORM 18 BIT ADDRESS
052174  042703  000001                           BIC     #BIT0,R3                ;MAKE SURE ITS EVEN
052200  010241                                   MOV     R2,-(R1)                ;RETURN NEW BUS ADDRESS
052202  010341                                   MOV     R3,-(R1)                ;TO THE APPROPRIATE HANDLER
052204  012705  000004                           MOV     #4,R5                   ;SET SOB COUNT
052210  120465  001655                   1$:     CMPB    R4,MAPTBL-1(R5)         ;IS THIS MAP IN USE?
052214  001435                                   BEQ     5$                      ;BRANCH IF YES
052216  077504                                   SOB     R5,1$                   ;CONTINUE
052220  110460  001656                           MOVB    R4,MAPTBL(R0)           ;PUT MAP SELECT BITS IN TABLE
052224  072427  000003                           ASH     #3,R4                   ;FORM INDEX TO GET MAP REG ADDR
052230  062704  170200                           ADD     #MAPL0,R4               ;GENERATE MAP ADDRESS
052234  042703  160000                           BIC     #160000,R3              ;GET LS 13 BITS OF RAND NO.
052240  013701  001226                           MOV     @#$GDDAT,R1             ;GET PHYSICAL
```

```
052250  160301                            SUB     R3,R1                   ;GENERATE MAP
052252  005602                            SBC     R2                      ;REGISTER DATA
052254  010124                            MOV     R1,(R4)+                ;LOAD THE
052256  010224                            MOV     R2,(R4)+                ;FIRST MAP REGISTER
052260  062701  020000                    ADD     #20000,R1               ;ADD 4K
052264  005502                            ADC     R2                      ;TO MAP DATA
052266  010124                            MOV     R1,(R4)+                ;LOAD THE
052270  010224                            MOV     R2,(R4)+                ;SECOND MAP REGISTER
052272  104414                            RESREG
052274  042637  177776                    BIC     (SP)+,@#PSW             ;RETURN PRIORITY TO ORIGINAL VALUE
052300  011666  000004                    MOV     (SP),4(SP)              ;SETUP RETURN PC
052304  022626                            CMP     (SP)+,(SP)+             ;CLEAN UP THE STACK
052306  000207                            RTS     PC                      ;RETURN
                                  ;REGISTER PAIR IS IN USE, TRY ANOTHER RANDOM NUMBER
052310  062701  000004            5$:     ADD     #4,R1                   ;RESTORE R1
052314  000137  052112                    JMP     @#2$                    ;GET ANOTHER RANDOM NUMBER

                                  ;;*********************************************************************
                                  .SBTTL  GIVE MAP SUBROUTINE
                                  ;*      THIS ROUTINE TAKES THE MAP ADDRESS OUT OF THE MAP TABLE
                                  ;*      FOR THE REQUESTING DEVICE AND REPLACES IT WITH 377.
                                  ;;*********************************************************************
052320  010046                    GIVEMAP:MOV     R0,-(SP)                ;SAVE R0
052322  016600  000004                    MOV     4(SP),R0                ;GET DEVICE ID
052326  112760  000377  001656            MOVB    #377,MAPTBL(R0)         ;TAKE IT OUT OF THE TABLE
052334  012600                            MOV     (SP)+,R0                ;RESTORE R0
052336  000207                            RTS     PC                      ;RETURN


                                  ;;*********************************************************************
                                  .SBTTL  ROUTINE TO CLEAR 'T' BIT
                                  ;;*********************************************************************
052340  013746  177776            CLRTBIT:MOV     @#PSW,-(SP)     ;PUSH PSW ONTO STACK
052344  011627                            MOV     (SP),(PC)+      ;SAVE IN RETPSW BELOW
052346  000000                    RETPSW: .WORD   0
052350  042716  000020                    BIC     #20,(SP)        ;CLEAR T BIT IN PSW ON STACK
                                  ;;*********************************************************************
                                  .SBTTL  ROUTINE TO RESTORE THE T BIT
                                  ;;*********************************************************************
052354  012746  052362            RESPSW: MOV     #1$,-(SP)       ;SET RETURN PC FOR RTI
052360  000002                            RTI                     ;CLEAR 'T' BIT IN PSW
052362  000207                    1$:     RTS     PC              ;RETURN

052364  042737  177400  177776    RESTPS: BIC     #177400,@#PSW   ;SET KERNEL MODE
052372  016746  177750                    MOV     RETPSW,-(SP)    ;PUSH ORIG PSW ONTO STACK
052376  000766                            BR      RESPSW

                                  ;;*********************************************************************
                                  .SBTTL  KEYBOARD INT SERV ROUTINE
                                  ;*THIS ROUTINE HANDLES INTERRUPTS FROM THE KEYBOARD
                                  ;*
                                  ;*TYPING A CONTROL 'C' WILL CAUSE THE PROCESSOR TO  HALT
                                  ;*
                                  ;*TYPING A CARRAGE RETURN WILL CAUSE A CARRIAGE RETURN-LINE FEED
                                  ;*TO BE TYPED.
                                  ;*
                                  ;*TYPING A CONTROL 'O' WILL INHIBIT ANY FURTHER TYPEOUT. THE SECOND CONTROL 'O'
```

```
                                        ;*
                                        ;*ANY OTHER CHARACTER WILL JUST BE ECHOED.
                                        ;;*************************************************************

                000003                              CNTRLC=3
                000017                              CNTRLO=17

        052400  017746  126636          TKISR:      MOV     @$TKB,-(SP)                 ;GET CHARACTER
        052404  042716  177600                      BIC     #177600,(SP)                ;STRIP UNUSED BITS
        052410  022716  000003                      CMP     #CNTRLC,(SP)                ;BRANCH IF NOT CONTROL C (↑C)
        052414  001010                              BNE     1$
        052416  012737  001326  001270              MOV     #$CRLF-1,@#$REG5            ;ECHO CR LF
        052424  106277  126614                      ASRB    @$TPS
        052430  005726                              TST     (SP)+                       ;POP CHARACTER OFF THE STACK
        052432  000000                              HALT
        052434  000002                              RTI                                 ;RETURN

        052436  122716  000015          1$:         CMPB    #15,(SP)                    ;BRANCH IF NOT <CR>
        052442  001007                              BNE     2$
        052444  012737  001326  001270              MOV     #$CRLF-1,@#$REG5            ;ECHO CR LF
        052452  106277  126566                      ASRB    @$TPS
        052456  005726                              TST     (SP)+                       ;POP CHARACTEROFF STACK
        052460  000002                              RTI                                 ;RETURN

        052462  122716  000017          2$:         CMPB    #CNTRLO,(SP)                ;BRANCH IF NOT CONTROL O (↑O)
        052466  001012                              BNE     3$
        052470  005726                              TST     (SP)+
        052472  005167  126770                      COM     NOTYPE
        052476  100405                              BMI     7$
        052500  012737  001326  001270              MOV     #$CRLF-1,@#$REG5            ;ECHO CR LF
        052506  106277  126532                      ASRB    @$TPS
        052512  000002                  7$:         RTI

        052514  104412                  3$:         SAVREG
        052516  011605                              MOV     (SP),R5             ;RETRIEVE CHARACTER
        052520  004737  053130                      JSR     PC,@#LDKT                   ;GO TO LOW CORE
        052524  013700  001442                      MOV     @#TKBFRP,R0                 ;GET BUFFER PTR
        052530  110520                  4$:         MOVB    R5,(R0)+                    ;LOAD CHAR INTO BFR
        052532  105010                              CLRB    (R0)                        ;CLEAR NEXT LOC
        052534  022700  001464          5$:         CMP     #TKBFR+20,R0                ;BRANCH IF NOT END OF BFR
        052540  001002                              BNE     6$
        052542  012700  001444                      MOV     #TKBFR,R0                   ;RESET BUFFER PTR
        052546  010037  001442          6$:         MOV     R0,@#TKBFRP                 ;RESTORE BFR PTR
        052552  004737  053216                      JSR     PC,@#RESKT                  ;GO BACK TO ORIGINAL MEMORY
        052556  104414                              RESREG
        052560  005737  001466          ECHO:       TST     @#NOTYPE                    ;TYPEOUT DISABLED?
        052564  100004                              BPL     1$                          ;BRANCH IF NO
        052566  005726                              TST     (SP)+                       ;FIX UP STACK
        052570  105077  126450                      CLRB    @$TPS                       ;CLEAR IE BIT
        052574  000002                              RTI                                 ;RETURN
        052576  105777  126442          1$:         TSTB    @$TPS               ;PRINTER READY?
        052602  100375                              BPL     .-4                         ;BRANCH IF NO
        052604  112677  126436                      MOVB    (SP)+,@$TPB                 ;MOVE CHAR TO PRINTER
        052610  000002                              RTI                                 ;RETURN
```

```
                                        ;;*************************************************************
                                        .SBTTL  TELETYPE INTERRUPT SERVICE ROUTINE
                                        ;*THIS ROUTINE TYPES A MESSAGE POINTED TO BY THE ADR STORED
                                        ;*IN LOCATION $REGS. THIS ROUTINE IS INTERRUPT DRIVEN.
                                        ;;*************************************************************
052612  005237  001270                  TPISR:  INC     @#$REGS         ;STEP MESSAGE ADDRESS PTR
052616  117746  126446                          MOVB    @$REGS,-(SP)            ;GET CHAR TO BE TYPED
052622  001356                                  BNE     ECHO                    ;GO TYPE CHAR IF NOT '0'
052624  005726                                  TST     (SP)+                   ;POP STACK
052626  005077  126412                          CLR     @$TPS                   ;CLEAR IE BIT
052632  012737  001526  001270                  MOV     #NULLS,@#$REGS
052640  000002                                  RTI                             ;RETURN

                                        ;;*************************************************************
                                        .SBTTL  PARITY ERROR SERVICE
                                        ;*      THIS ROUTINE FIELDS UNEXPECTED TRAPS TO 114. IT IS ASSUMED
                                        ;*      THAT THE ERROR WAS IN CACHE AND WAS CAUSED BY THE "OTHER
                                        ;*      WORD" RATHER THAN THE "WANTED WORD" WHICH MEANS THAT THE
                                        ;*      BAD DATA IS STILL IN THE CACHE. SO, TO CLEAR THE BAD DATA
                                        ;*      THE ERROR ADDRESS IS REFERENCED CAUSING THE CACHE TO GO
                                        ;*      TO MAIN MEMORY TO GET THE DATA. THIS PREVENTS AN
                                        ;*      ARBITRARY REFERENCE TO THE BAD WORD FROM TRAPPING.
                                        ;*
                                        ;*      AFTER THE ERROR IS REPORTED, BITS 2 AND 3 OF THE MEMORY
                                        ;*      ERROR REGISTER ARE TESTED TO SEE IF THE BAD DATA IS IN
                                        ;*      MAIN MEMORY. IF IT IS, THE PROGRAM RESTARTS SINCE THE
                                        ;*      GOOD DATA IS NOW LOST FOREVER. OTHERWISE THE PROGRAM
                                        ;*      RETURNS TO THE ADDRESS POINTED TO BY "$LPERR".
                                        ;;*************************************************************
052642  012737  053122  000114          .PARSRV:MOV     #RT1,@#CACHVEC  ;PUT NEW ADDRESS IN PARITY VECTOR
052650  016637  000002  001276                  MOV     2(SP),@#$TMP0   ;SAVER ERROR PSW
052656  011637  001474                          MOV     (SP),@#VADR     ;SAVE PC
052662  162737  000002  001474                  SUB     #2,@#VADR       ;ADJUST ERROR PC
052670  013702  177744                          MOV     @#MEMERR,R2     ;GET ERROR REGISTER
052674  013703  177740                          MOV     @#LOADRS,R3     ;GET LO ADDRESS ERROR REG
052700  010337  001304                          MOV     R3,@#$TMP3      ;PUT LOW ADR IN MEMORY
052704  013737  177742  001306                  MOV     @#HIADRS,@#$TMP4        ;GET HI ADDRESS ERROR REG
052712  042703  176000                          BIC     #176000,R3      ;MASK OFF LOWER TEN BITS
052716  013704  172354                          MOV     @#KIPAR6,R4     ;SAVE PAR6
052722  105737  001503                          TSTB    @#MMON          ;IS MEMORY MGMT ON?
052726  001407                                  BEQ     1$              ;BRANCH IF NO
052730  005037  172354                          CLR     @#KIPAR6        ;CLEAR PAR6
052734  012737  077406  172314                  MOV     #77406,@#KIPDR6 ;ENSURE PDR 6 RESIDENT
052742  052703  140000                          BIS     #140000,R3      ;SETUP R3 TO REFERENCE THRU PAR6
052746  105713                          1$:     TSTB    (R3)            ;REFERENCE ADDRESS THAT TRAPPED
                                                                        ;SHOULD CAUSE ABORT
052750  005102                          2$:     COM     R2              ;GET ORIGINAL MEMORY
052752  010237  177744                          MOV     R2,@#MEMERR     ;ERROR REG DATA
052756  013737  177744  001302          PERET:  MOV     @#MEMERR,@#$TMP2        ;SAVE ERROR REG FOR TYPEOUT
052764  013737  001212  001266                  MOV     @#$LPERR,@#$REG4        ;SAVE LOOP ADDRESS
052772  012737  053002  001212                  MOV     #2$,@#$LPERR            ;SET RETURN ADDRESS IF LOOPING
053000  104004                                  ERROR   4
053002  013737  001266  001212          2$:     MOV     @#$REG4,@#$LPERR        ;RESTORE LOOP ADDRESS
053010  010437  172354                          MOV     R4,@#KIPAR6     ;RESTORE PAR6
053014  013704  177744                          MOV     @#MEMERR,R4     ;GET MEM ERR REG
```

```
053026  012737  052642  000114           MOV     #.PARSRV,@#CACHVEC        ;RESTORE PARITY VECTOR
053034  042704  177763                   BIC     #177763,R4               ;CLEAR ALL BUT BITS 2 & 3
053040  001426                           BEQ     1$                       ;BRANCH IF NOT MAIN MEMORY ERROR
053042  104400  053050                   TYPE    ,65$                     ;;TYPE ASCIZ STRING
053046  000420                           BR      64$                      ;;GET OVER THE ASCIZ
                                 ;;65$:  .ASCIZ  /FATAL PARITY ERROR-RESTARTING/<CRLF>
053110                           64$:
053110  000005                           RESET                            ;CLEAR THE WORLD
053112  000137  003212                   JMP     @#START
053116  012716  053124           1$:     MOV     #X,(SP)                  ;PUT ADDRESS ON STACK TO GET ORIGINAL
                                                                          ;PSW BACK
053122  000002                   RT1:    RTI                              ;GET OLD PSW
053124  000177  126062           X:      JMP     @SLPERR                  ;JUMP TO START OF TEST THAT HAD THE PE
                                 ;;*****************************************************************
                                 .SBTTL  CONTEXT SWITCH DOWN SUBROUTINE
                                 ;*      SUBROUTINE TO SAVE & LOAD KIPAR'S 0,1,AND 2 (IF MEM MGMT ENABLED)
                                 ;*      THIS ROUTINE IS CALLED BY THE KEYBOARD INTERRUPT, LINE CLOCK
                                 ;*      INTERRUPT, UBE SERVICE ROUTINE, MBT SERVICE ROUTINE, AND TYPE TIME ROUTINE.
                                 ;;*****************************************************************
053130  105737  001503           LDKT:   TSTB    @#MMON                  ;BRANCH IF MEM MGMT DISABLED
053134  001427                           BEQ     1$
053136  012604                           MOV     (SP)+,R4                 ;SAVE RETURN PC
053140  013737  177776  053272           MOV     @#PSW,@#$SAVPSW          ;SAVE THE CURRENT PSW
053146  042737  140000  177776           BIC     #140000,@#PSW            ;GO TO KERNEL MODE
053154  012700  172340                   MOV     #KIPAR0,R0               ;GET ADDRESS OF PAR0
053160  012001                           MOV     (R0)+,R1                 ;GET PAR0
053162  012002                           MOV     (R0)+,R2                 ;GET PAR1
053164  012003                           MOV     (R0)+,R3                 ;GET PAR2
053166  012740  000400                   MOV     #400,-(R0)               ;RELOC BACK TO LOW CORE
053172  012740  000200                   MOV     #200,-(R0)
053176  005040                           CLR     -(R0)
053200  012700  053264                   MOV     #$SAVPAR,R0              ;GET ADDRESS OF SAVE BUFFER
053204  010120                           MOV     R1,(R0)+                 ;PUT PAR DATA IN MEMORY
053206  010220                           MOV     R2,(R0)+
053210  010320                           MOV     R3,(R0)+
053212  010446                           MOV     R4,-(SP)                 ;PUT RETURN PC ON STACK
053214  000207                   1$:     RTS     PC

                                 ;;*****************************************************************
                                 .SBTTL  CONTEXT SWITCH UP SUBROUTINE
                                 ;SUBROUTINE TO RESTORE KIPAR0, 1, AND 2 (IF MGMT ENABLED)
                                 ;;*****************************************************************
053216  105737  001503           RESKT:  TSTB    @#MMON                   ;BRANCH IF MEM MGMT DISABLED
053222  001417                           BEQ     1$
053224  012604                           MOV     (SP)+,R4                 ;GET RETURN PC
053226  012700  053264                   MOV     #$SAVPAR,R0              ;GET ADDRESS OF SAVE BUFF
053232  012001                           MOV     (R0)+,R1                 ;GET OLD PAR DATA
053234  012002                           MOV     (R0)+,R2
053236  012003                           MOV     (R0)+,R3
053240  012700  172340                   MOV     #KIPAR0,R0               ;GET ADDRESS OF PAR0
053244  010120                           MOV     R1,(R0)+                 ;RELOCATE BACK
053246  010220                           MOV     R2,(R0)+
053250  010310                           MOV     R3,(R0)
053252  013737  053272  177776           MOV     @#$SAVPSW,@#PSW
053260  010446                           MOV     R4,-(SP)
```

```
053264  000003                          $SAVPAR:.BLKW   3
053272  000000                          $SAVPSW:.WORD
                                        ;;*********************************************************
                                        .SBTTL  KT ABORT SUBROUTINE
                                        ;;*********************************************************
053274  016637  000002  001276          KTABRT: MOV     2(SP),@#$TMP0           ;SAVE ERROR PSW
053302  011637  001474                          MOV     (SP),@#VADR             ;SAVE ERROR PC
053306  162737  000002  001474                  SUB     #2,@#VADR
053314  013737  177572  001302                  MOV     @#MMR0,@#$TMP2          ;SAVE MMR0
053322  013737  177576  001304                  MOV     @#MMR2,@#$TMP3          ;SAVE MMR2
053330  013737  001212  001266                  MOV     @#$LPERR,@#$REG4        ;SAVE LOOP ADDRESS
053336  012737  053346  001212                  MOV     #1$,@#$LPERR            ;SET RETURN ADR IF LOOPING
053344  104003                                  ERROR   3
053346  013737  001266  001212          1$:     MOV     @#$REG4,@#$LPERR        ;RESTORE LOOP ADR
053354  042737  170000  177572                  BIC     #170000,@#MMR0          ;CLEAR ERRORS
053362  013716  001212                          MOV     @#$LPERR,(SP)           ;GET LOOP ADDRESS
053366  000002                                  RTI                             ;RETURN

                                        ;;*********************************************************
                                        .SBTTL  RESERVED INSTRUCTION ROUTINE
                                        ;;*********************************************************
053370  016637  000002  001276          RESERR: MOV     2(SP),@#$TMP0           ;SAVE PSW
053376  011637  001474                          MOV     (SP),@#VADR             ;SAVE ERROR PC
053402  162737  000002  001474                  SUB     #2,@#VADR
053410  013737  001212  001266                  MOV     @#$LPERR,@#$REG4        ;SAVE LOOP ADR
053416  012737  053426  001212                  MOV     #1$,@#$LPERR            ;SET RETURN ADR IF LOOPING
053424  104002                                  ERROR   2
053426  013737  001266  001212          1$:     MOV     @#$REG4,@#$LPERR        ;RESTORE LOOP ADR
053434  013716  001212                          MOV     @#$LPERR,(SP)           ;GET LOOP ADDRESS
053440  000002                                  RTI                             ;RETURN

                                        ;;*********************************************************
                                        .SBTTL  TRAP TO 4 SERVICE ROUTINE
                                        ;;*********************************************************
053442  016637  000002  001276          ERPRT:  MOV     2(SP),@#$TMP0           ;SAVE ERROR PSW
053450  011637  001474                          MOV     (SP),@#VADR             ;SAVE ERROR PC
053454  162737  000002  001474                  SUB     #2,@#VADR
053462  012706  000700                          MOV     #SUPSTK,SP              ;RESTORE SP
053466  013737  177766  001302                  MOV     @#CPUERR,@#$TMP2        ;GET ERROR REG
053474  013737  001212  001266                  MOV     @#$LPERR,@#$REG4        ;SAVE LOOP ADR
053502  012737  053512  001212                  MOV     #1$,@#$LPERR            ;SET RETURN ADR IF LOOPING
053510  104001                                  ERROR   1
053512  013737  001266  001212          1$:     MOV     @#$REG4,@#$LPERR        ;SET LOOP ADR
053520  005037  177766                          CLR     @#CPUERR
053524  013746  001276                          MOV     @#$TMP0,-(SP)   ;SETUP STACK TO RETURN
053530  013746  001212                          MOV     @#$LPERR,-(SP)
053534  000002                                  RTI                             ;RETURN
```

```
                                        ;THE BELOW TABLE REPRESENTS THE 'NEW' PSW SET BY THE PROGRAM ON
                                        ;SUCCESSIVE SUB-PASSES.
                                        ;NOTE THE BELOW TABLE MAY BE MODIFIED TO CAUSE THE PROGRAM TO RUN
                                        ;UNDER USER DEFINED PARAMETERS BY PATCHING IN THE DESIRED PASS PARAMETER
                                        ;FOR EXAMPLE TO CAUSE THE PROGRAM TO RUN WITHOUT SETTING THE 'T' BIT
                                        ;IN ALL PASSES PATCH OUT THE 'T' BIT IN THE TABLE.
053536  000000                          PSWTAB: 000000
053540  000020                                  000020                  ;T-BIT TRAPPING
053542  140000                                  140000                  ;USER MODE
053544  144020                                  144020                  ;USER MODE, REG SET #1, T-BIT TRAPPING
053546  040000                                  040000                  ;SUPERVISOR MODE
053550  044020                                  044020                  ;SUPERVISOR MODE, REG SET #1, T-BIT TRAPPING

                                        ;THE BELOW TABLE IS USED TO SET MEMORY MARGINS
053552  000000                          MRGTAB: .WORD   0                       ;NO MARGINS
053554  000004                                  .WORD   4                       ;EARLY STROBE
053556  000006                                  .WORD   6                       ;LATE STROBE
053560  000010                                  .WORD   10                      ;LOW DRIVE CURRENT
053562  000000                                  .WORD   0                       ;NO MARGINS
053564  000012                                  .WORD   12                      ;HIGH DRIVE CURRENT
                                        ;MESSAGES
                                                .EVEN
053566  002100                          REGINX: RP3DS
053570  002122                                  RKDS
053572  000000                                  .WORD
053574  000000                                  .WORD
053576  002142                                  RP4CS1
053600  002202                                  RSCS1
053602  000000                                  .WORD
053604  000000                                  .WORD
053606  002246                                  MBTTBL
053610  002230                                  UBETBL
053612  054066                          MSGINX: .WORD   MSG5
053614  054074                                  .WORD   MSG6
053516  054703                                  .WORD   MSG21
053620  054703                                  .WORD   MSG21
053622  054102                                  .WORD   MSG10
053624  054110                                  .WORD   MSG11
053626  054703                                  .WORD   MSG21
053630  054703                                  .WORD   MSG21
053632  054377                                  .WORD   MSG15
053634  054742                                  .WORD   MSG24
053636  046200  053517  046040          MSG1:   .ASCIZ  <CRLF>'LOW LIM?'
053644  046511  000077
053650  044510  044107  046040          MSG2:   .ASCIZ  'HIGH LIM?'
053656  046511  000077
053662  051105  047522  050122          MSG3:   .ASCIZ  /ERRORPC PHYSC PC    PSW   MAINT   TEST NO SUB-PASS CNT/
053670  020103  044120  051531
053676  020103  041520  020040
053704  020040  051520  020127
053712  020040  040515  047111
053720  020124  020040  042524
053726  052123  047040  020117
053734  052523  026502  040520
053742  051523  041440  052116
```

```
053751     124  042510  043040  MSG4:   .ASCII  /THE FOLLOWING DEVICES AND DRIVES WILL BE USED FOR RELOCATION:/<CRLF>
053756  046117  047514  044527
053764  043516  042040  053105
053772  041511  051505  040440
054000  042116  042040  044522
054006  042526  020123  044527
054014  046114  041040  020105
054022  051525  042105  043040
054030  051117  051040  046105
054036  041517  052101  047511
054044  035116     200
054047     104  053105  041511          .ASCIZ  /DEVICE DRIVES/<CRLF>
054054  004505  051104  053111
054062  051505  000200
054066  050122  031460  000011  MSG5:   .ASCIZ  ?RP03   ?
054074  045522  032460  000011  MSG6:   .ASCIZ  ?RK05   ?
054102  050122  032060  000011  MSG10:  .ASCIZ  ?RP04   ?
054110  051522  032060  000011  MSG11:  .ASCIZ  ?RS04   ?
054116  051104  051526  040524  MSG12:  .ASCIZ  /DRVSTA  ERRREG  CSREG   WRDCNT  BUSADR  DSKADR  CYLADR(RP03)  PHYS BUSA
054124  020040  051105  051122
054132  043505  020040  051503
054140  042522  020107  020040
054146  051127  041504  052116
054154  020040  052502  040523
054162  051104  020040  051504
054170  040513  051104  020040
054176  054503  040514  051104
054204  051050  030120  024463
054212  020040  044120  051531
054220  041040  051525  042101
054226  100122     000
054231     040  051503  020061  MSG13:  .ASCIZ  / CS1    WRDCNT  BUSADR  BADREX  DSKADR   CS2     CS3    DRVSTA  ERRREG/
054236  020040  053440  042122
054244  047103  020124  041040
054252  051525  042101  020122
054260  041040  042101  042522
054266  020130  042040  045523
054274  042101  020122  020040
054302  051503  020062  020040
054310  020040  051503  020063
054316  020040  042040  053122
054324  052123  020101  042440
054332  051122  042522  100107
054340     000
054341     104  051505  054503  MSG14:  .ASCIZ  /DESCYL   ER2     ER3    RPCC/<CRLF>
054346  020114  020040  051105
054354  020062  020040  020040
054362  051105  020063  020040
054370  051040  041520  100103
054376     000
054377     115  051501  020123  MSG15:  .ASCIZ  /MASS BUS TESTER /
054404  052502  020123  042524
054412  052123  051105  000040
054420  041440  030523  020040  MSG16:  .ASCIZ  / CS1    WRDCNT  BUSADR  BADREX    MR2     CS2      ST      ER       CS3/<
054426  020040  051127  041504
```

```
054442  040523  051104  020040
054450  040502  051104  054105
054456  020040  020040  051115
054464  020062  020040  020040
054472  051503  020062  020040
054500  020040  051440  020124
054506  020040  020040  051105
054514  020040  020040  020040
054522  051503  100063     000
054527     040  041440  020103  MSG17:  .ASCIZ  /  CC    BUSADR    CR2      CR1  PHYS BUSADR/<CRLF>
054534  020040  041040  051525
054542  042101  020122  020040
054550  041440  031122  020040
054556  020040  041440  030522
054564  020040  044120  051531
054572  041040  051525  042101
054600  100122     000
054603     124  042510  050440  MSG20:  .ASCIZ  /THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789/<15><12>
054610  044525  045503  041040
054616  047522  047127  043040
054624  054117  045040  046525
054632  042520  020104  053117
054640  051105  052040  042510
054646  046040  055101  020131
054654  047504  051507  041040
054662  041501  020113  030460
054670  031462  032464  033466
054676  034470  005015     000
054703     111  046114  043505  MSG21:  .ASCIZ  /ILLEGAL DEVICE/<CRLF>
054710  046101  042040  053105
054716  041511  100105     000
054723     040  020040  020040  MSG22:  .ASCII  /       /
054730  020040     040
054733     040  020040  020040  MSG23:  .ASCIZ  /      /
054740  000040
054742  047125  041111  051525  MSG24:  .ASCIZ  /UNIBUS EXERCISER /
054750  042440  042530  041522
054756  051511  051105  000040
054764  050117  027124  050103  MSG25:  .ASCIZ  /OPT.CP=/
054772  000075
054774  047125  054105  042520  EM1:    .ASCIZ  /UNEXPECTED TRAP TO 4/
055002  052103  042105  052040
055010  040522  020120  047524
055016  032040     000
055021     120  047503  052106  DH1:    .ASCIZ  /PCOFTP  PHYSPC    PSW   CPUERR/
055026  020120  050040  054510
055034  050123  020103  020040
055042  050040  053523  020040
055050  041440  052520  051105
055056  000122
055060     000     001     000  DF1:    .BYTE   0,1,0,0,0
055063     000     000
        055066                          .EVEN
055066  001474  001474  001276  DT1:    .WORD   VADR,VADR,$TMP0,$TMP2,0
055074  001302  000000
```

```
055106  052103  042105  052040
055114  040522  020120  047524
055122  030440  000060
055126  041520  043117  050124  DH2:     .ASCIZ  /PCOFTP  PHYSPC     PSW/
055134  020040  044120  051531
055142  041520  020040  020040
055150  051520  000127
                                         .EVEN
055154  001474  001474  001276  DT2:     .WORD   VADR,VADR,$TMP0,0
055162  000000
055164  047125  054105  042520  EM3:     .ASCIZ  /UNEXPECTED TRAP TO 250(MGMT)/
055172  052103  042105  052040
055200  040522  020120  047524
055206  031040  030065  046450
055214  046507  024524     000
055221     120  047503  052106  DH3:     .ASCIZ  /PCOFTP  PHYSPC     PSW    MMR0    MMR2/
055226  020120  050040  054510
055234  050123  020103  020040
055242  050040  053523  020040
055250  020040  046515  030122
055256  020040  020040  046515
055264  031122     000
        055270                           .EVEN
055270  001474  001474  001276  DT3:     .WORD   VADR,VADR,$TMP0,$TMP2,$TMP3,0
055276  001302  001304  000000
055304  047125  054105  042520  EM4:     .ASCIZ  /UNEXPECTED TRAP TO 114/
055312  052103  042105  052040
055320  040522  020120  047524
055326  030440  032061     000
055333     120  047503  052106  DH4:     .ASCIZ  /PCOFTP  PHYSC PC     PSW   ERRREG  ERR ADR REG/
055340  020120  050040  054510
055346  041523  050040  020103
055354  020040  050040  053523
055362  020040  042440  051122
055370  042522  020107  042440
055376  051122  040440  051104
055404  051040  043505     000
055411     000     001     000  DF4:     .BYTE   0,1,0,0,2
055414     000     002
055416  040520  044522  054524  EM5:     .ASCIZ  /PARITY ERROR DURING DATA CHECK/
055424  042440  051122  051117
055432  042040  051125  047111
055440  020107  040504  040524
055446  041440  042510  045503
055454     000
055455     123  041522  042101  DH5:     .ASCIZ  /SRCADR  DSTADR  EADRREG  MEM ERR REG/
055462  020122  042040  052123
055470  042101  020122  042440
055476  042101  051122  043505
055504  020040  042515  020115
055512  051105  020122  042522
055520  000107
055522     000     001     002  DF5:     .BYTE   0,1,2,0
055525     000
                                         .EVEN
```

```
055534  001302  000000
055540  051105  047522  020122  EM6:     .ASCIZ  /ERROR DURING DATA CHECK-RELOC WAS BY CP/
055546  052504  044522  043516
055554  042040  052101  020101
055562  044103  041505  026513
055570  042522  047514  020103
055576  040527  020123  054502
055604  041440  000120
055610  051123  040503  051104  DH6:     .ASCIZ  /SRCADR  DSTADR/
055616  020040  051504  040524
055624  051104     000
        055630                           .EVEN
055630  001276  001474  000000  DT6:     .WORD   $TMPO,VADR,0
055636  051105  047522  020122  EM10:    .ASCIZ  ?ERROR DURING DATA CHECK-RELOC WAS BY I/O?
055644  052504  044522  043516
055652  042040  052101  020101
055660  044103  041505  026513
055666  042522  047514  020103
055674  040527  020123  054502
055702  044440  047457     000
055707     123  041522  042101  DH10:    .ASCIZ  /SRCADR   DSTADR   DEVICE THAT DID XFER/
055714  020122  020040  051504
055722  040524  051104  020040
055730  042040  053105  041511
055736  020105  044124  052101
055744  042040  042111  054040
055752  042506  000122
055756     000     001     003  DF10:    .BYTE   0,1,3,0
055761     000
                                         .EVEN
055762  001276  001474  001302  DT10:    .WORD   $TMPO,VADR,$TMP2,$TMP3,0
055770  001304  000000
055774  044502  024124  024523  EM11:    .ASCIZ  /BIT(S) STUCK IN MICRO-BREAK REGISTER/
056002  051440  052524  045503
056010  044440  020116  044515
056016  051103  026517  051102
056024  040505  020113  042522
056032  044507  052123  051105
056040     000
056041     107  047517  042104  DH11:    .ASCIZ  /GOODDAT BAD DATA/
056046  052101  041040  042101
056054  042040  052101  000101
056062     000     000          DF11:    .BYTE   0,0
                                         .EVEN
056064  001276  001300  000000  DT11:    .WORD   $TMPO,$TMP1,0
056072  041125  020105  047516  EM12:    .ASCIZ  /UBE NON-EXISTANT MEMORY ERROR/
056100  026516  054105  051511
056106  040524  052116  046440
056114  046505  051117  020131
056122  051105  047522  000122
056130  044120  051531  041040  DH12:    .ASCIZ  /PHYS BUSADR/
056136  051525  042101  000122
056144     002                  DF12:    .BYTE   2
        056146                           .EVEN
056146  001226  000000          DT12:    .WORD   $GDDAT,0
```

```
056160  026516  054105  051511
056166  040524  052116  046440
056174  046505  051117  020131
056202  051105  047522  000122
056210  044120  051531  040440  DH13:    .ASCIZ  /PHYS ADDRESS/
056216  042104  042522  051523
056224     000
056225     106  047514  052101  EM14:    .ASCIZ  /FLOATING POINT ERROR/
056232  047111  020107  047520
056240  047111  020124  051105
056246  047522  000122
056252  042011  040524  030524  DH14:    .ASCIZ  /        DTAT1                               DATA2/
056260  004411  004411  040504
056266  040524  000062
                                         .EVEN
056272  001306  001262  001312  DT14:    .WORD   $TMP4,$REG2,$TMP6,$REG3,0
056300  001264  000000
056304     004     000     004  DF14:    .BYTE   4,0,4,0
056307     000
056310  042504  044526  042503  EM15:    .ASCIZ  /DEVICE HUNG/
056316  044040  047125  000107
056324  004411  040504  040524  DH16:    .ASCIZ  /               DATA1                            DATA2/
056332  004461  004411  020011
056340  020040  042040  052101
056346  031101     000
056351     005     000     005  DF16:    .BYTE   5,0,5,0
056354     000
        056356                           .EVEN
056356  001422  001262  001432  DT16:    .WORD   FLTMP0,$REG2,FLTMP1,$REG3,0
056364  001264  000000
056370  000000                  ENDTAG:  .WORD   0
                                ;;*********************************************************************
                                ;THE FOLLOWING ASCII GETS OVERLAYED WHEN THE PROGRAM RUNS.
056372  050117  051105  052101  SWITCH:  .ASCII  /OPERATIONAL SWITCH SETTINGS/<CRLF>
056400  047511  040516  020114
056406  053523  052111  044103
056414  051440  052105  044524
056422  043516  100123
056426  053523  052111  044103           .ASCII  /SWITCH                  USE/<CRLF>
056434  004411  052411  042523
056442     200
056443     040  030440  004465           .ASCII  /  15          HALT ON ERROR/<CRLF>
056450  044011  046101  020124
056456  047117  042440  051122
056464  051117     200
056467     040  030440  004464           .ASCII  /  14          LOOP ON TEST/<CRLF>
056474  046011  047517  020120
056502  047117  052040  051505
056510  100124
056512  020040  031461  004411           .ASCII  /  13          INHIBIT ERROR TYPEOUTS/<CRLF>
056520  047111  044510  044502
056526  020124  051105  047522
056534  020122  054524  042520
056542  052517  051524     200
056547     040  030440  004462           .ASCII  /  12          INHIBIT UBE/<CRLF>
```

```
056562  052111  052440  042502
056570     200
056571     040  030440  004461          .ASCII  /  11          INHIBIT ITTERATIONS/<CRLF>
056576  044411  044116  041111
056604  052111  044440  052124
056612  051105  052101  047511
056620  051516     200
056623     040  030440  004460          .ASCII  /  10          BELL ON ERROR/<CRLF>
056630  041011  046105  020114
056636  047117  042440  051122
056644  051117     200
056647     040  020040  004471          .ASCII  /   9          LOOP ON ERROR/<CRLF>
056654  046011  047517  020120
056662  047117  042440  051122
056670  051117     200
056673     040  020040  004470          .ASCII  ?   8          INHIBIT RELOCATION VIA I/O DEVICE?<CRLF>
056700  044411  044116  041111
056706  052111  051040  046105
056714  041517  052101  047511
056722  020116  044526  020101
056730  027511  020117  042504
056736  044526  042503     200
056743     040  020040  004467          .ASCII  /   7          INHIBIT TYPEOUT OF THIS TEXT AND SYS SIZE/<CRLF>
056750  044411  044116  041111
056756  052111  052040  050131
056764  047505  052125  047440
056772  020106  044124  051511
057000  052040  054105  020124
057006  047101  020104  054523
057014  020123  044523  042532
057022     200
057023     040  020040  004466          .ASCII  /   6          INHIBIT RELOCATION/<CRLF>
057030  044411  044116  041111
057036  052111  051040  046105
057044  041517  052101  047511
057052  100116
057054  020040  032440  004411          .ASCII  /   5          INHIBIT ROUND ROBIN RELOCATION/<CRLF>
057062  047111  044510  044502
057070  020124  047522  047125
057076  020104  047522  044502
057104  020116  042522  047514
057112  040503  044524  047117
057120     200
057121     040  020040  004464          .ASCII  /   4          INHIBIT RANDOM DISK ADDRESS/<CRLF>
057126  044411  044116  041111
057134  052111  051040  047101
057142  047504  020115  044504
057150  045523  040440  042104
057156  042522  051523     200
057163     040  020040  004463          .ASCII  /   3          INHIBIT MBT/<CRLF>
057170  044411  044116  041111
057176  052111  046440  052102
057204     200
057205     040  020040  004462          .ASCII  /   2          THESE THREE SWITCHES/<CRLF>
057212  052011  042510  042523
```

```
9769 057226  051440  044527  041524
9770 057234  042510  100123
9771 057240  020040  030440  004411          .ASCII  /   1          ARE ENCODED TO SELECT RELOCATION/<CRLF>
9772 057246  051101  020105  047105
9773 057254  047503  042504  020104
9774 057262  047524  051440  046105
9775 057270  041505  020124  042522
9776 057276  047514  040503  044524
9777 057304  047117     200
9778 057307     040  020040  004460          .ASCII  /   0          ON THE FOLLOWING DEVICES:/<CRLF>
9779 057314  047411  020116  044124
9780 057322  020105  047506  046114
9781 057330  053517  047111  020107
9782 057336  042504  044526  042503
9783 057344  035123     200
9784 057347     011  027060  027056          .ASCII  ?       0...RP11/RP03?<CRLF>
9785 057354  050122  030461  051057
9786 057362  030120  100063
9787 057366  030411  027056  051056          .ASCII  ?       1...RK11/RK05?<CRLF>
9788 057374  030513  027461  045522
9789 057402  032460     200
9790 057405     011  027062  027056          .ASCII  ?       2...NOT USED?<CRLF>
9791 057412  047516  020124  051525
9792 057420  042105     200
9793 057423     011  027063  027056          .ASCII  ?       3...NOT USED?<CRLF>
9794 057430  047516  020124  051525
9795 057436  042105     200
9796 057441     011  027064  027056          .ASCII  ?       4...RH70/RP04?<CRLF>
9797 057446  044122  030067  051057
9798 057454  030120  100064
9799 057460  032411  027056  051056          .ASCII  ?       5...RH70/RS04 OR RS03?<CRLF>
9800 057466  033510  027460  051522
9801 057474  032060  047440  020122
9802 057502  051522  031460     200
9803 057507     011  027066  027056          .ASCII  ?       6...NOT USED?<CRLF>
9804 057514  047516  020124  051525
9805 057522  042105     200
9806 057525     011  027067  027056          .ASCIZ  ?       7...NOT USED?<CRLF>
9807 057532  047516  020124  051525
9808 057540  042105  000200
```

J16

```
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 186
CEQKCC.P11     03-MAR-78 13:13             CROSS REFERENCE TABLE -- USER SYMBOLS                                       SEQ 0204

AA      001505              644#     1103*
ADCB2   011170             2265      2267#
ADCB5   012014             2495      2497#
ADCB6   012516             2651      2652      2654#
ADCB7   013440             2874      2875      2876      2878#
ADC0    007062             1664      1665      1666      1668#
ADC1    007762             1893      1894      1895      1897#
ADC2    010772             2195      2197#
ADC5    011614             2423      2424      2426#
ADC6    012320             2594      2595      2597#
ADC7    013306             2832      2833      2835#
ADD0    014134             3041      3042      3043      3045#
ADD1    014240             3078      3079      3081#
ADD1A   014464             3161      3162      3163      3165#
ADD1B   014502             3170      3171      3173#
ADD2    015124             3315      3316      3318#
ADD3    015702             3507      3509#
ADD6    016260             3608      3609      3611#
ADD7    016736             3724      3725      3726      3728#
ARBEX   033376             6314      6357#
ARBFIN  033364             6299      6321      6342      6344      6355#
ASHCL0  025230             5144#
ASHCR0  025306             5165#
ASHL0   025020             5094#
ASHL1   025616             5276#
ASHR0   025134             5121#
ASHR1   025704             5299#
ASLB1   010332             2027      2028      2030#
ASLB1A  010556             2114      2115      2117#
ASLB3   012004             2489      2490      2492#
ASLB4   011274             2302      2303      2304      2306#
ASLB6   012500             2643      2644      2645      2647#
ASLB7   013536             2906      2907      2909#
ASL0    007204             1708      1709      1710      1711      1713#
ASL1    010136             1956      1957      1958      1960#
ASL3    011530             2392      2393      2395#
ASL4    011064             2227      2228      2229      2231#
ASL6    012270             2582      2583      2585#
ASL7    013134             2779      2780      2782#
ASRB1   010426             2063      2065#
ASRB1A  010442             2069      2070      2072#
ASRB2   011240             2287      2288      2290#
ASRB2A  011256             2295      2296      2298#
ASRB5   011744             2470      2471      2473#
ASRB6   012616             2683      2684      2686#
ASRB7   013554             2913      2914      2916#
ASR0    007232             1722      1723      1724      1726#
ASR1    010024             1913      1914      1915      1917#
ASR2    011006             2201      2202      2204#
ASR3    011514             2386      2388#
ASR6    012152             2544      2545      2547#
ASR7    013170             2793      2794      2796#
BICB1   014664             3233      3234      3236#
BICB1A  014706             3244      3247#     3305      3306
BIC0    014046             3012      3013      3014      3016#
```

K16

```
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 187
CEQKCC.P11     03-MAR-78 13:13             CROSS REFERENCE TABLE -- USER SYMBOLS                                   SEQ 0205

BIC2      015214           3344    3345    3346    3348#
BIC3      015714           3512    3514#
BIC7      017602           3903#   3905
BIN8      015440           3408    3410    3413    3415    3418    3420    3423    3426#
BIN87     017350           3838    3846#
BIN1      015044           3276    3279    3282    3285    3288    3291    3294    3299#
BISB1     014652           3228    3230#
BIS0      014024           3003    3004    3006#
BIS0A     014102           3030    3032#
BIS1      014350           3118    3119    3121#
BIS2      015152           3328    3330#
BIS2A     015254           3362    3363    3365#
BIS7      017542           3893#   3895
BITB1     014642           3222    3223    3225#
BITB2     015524           3448    3449    3451#
BITB3     016072           3566    3569#
BITB6     016472           3657#
BITT1     014276           3095    3096    3098#
BITT2     015240           3355    3356    3358#
BIT0    = 000001            179#   1386    1394    1396    7221    7259    7300    9092
BIT00   = 000001            169#    179
BIT01   = 000002            168#    178
BIT02   = 000004            167#    177
BIT03   = 000010            166#    176
BIT04   = 000020            165#    175
BIT05   = 000040            164#    174
BIT06   = 000100            163#    173
BIT07   = 000200            162#    172
BIT08   = 000400            161#    171
BIT09   = 001000            160#    170    7824    7892
BIT1    = 000002            178#
BIT10   = 002000            159#   7875
BIT11   = 004000            158#    513    7724    7831
BIT12   = 010000            157#    512    1350    1370    7134
BIT13   = 020000            156#   7114    7882
BIT14   = 040000            155#    511    1050    7387    7459    7482    7497    7508    7527    7558    7575    7590    7601
                           7620    7717    8814
BIT15   = 100000            154#   6754    7231    7238    7350    7357    7477    7484    7570    7577    8765    9090
BIT2    = 000004            177#
BIT3    = 000010            176#   7266
BIT4    = 000020            175#   1427    4443    5006    5635    6825    7038    7042
BIT5    = 000040            174#   1358    1378    5802    6377    6762    6764    6972    7053    7119    7271    7307    7392
                           7428    7463    7502    7532    7562    7595    7625    8952
BIT6    = 000100            173#   7065    7789
BIT7    = 000200            172#   1296
BIT8    = 000400            171#   1302    1339    1356    1376    6768    6931    7008    7081    7148    7215    7250    7293
                           7333    7370    7412    7457    7468    7495    7525    7556    7588    7618
BIT9    = 001000            170#   6778
BPTVEC= 000014              186#   4355    4358*   4359*   4432*   4433*
CACHVE= 000114              193#   1440*   1442*   9022*   9031*   9246*   9273*
CALLHA= 000010              514#   6663    6770
CBIT      023020           4610#
CC0       006670           1586    1587    1588    1589    1590    1591    1592    1594#
CC1       006704           1598    1599    1600    1602#
CC2       006720           1606    1607    1608    1610#
```

```
CC4      006746         1621     1622     1623     1625#
CHKDAT   051746         6529     6709     6793     9022#    9030
CHKSP    022612         4544#
CLRTBI   052340         4349     4500     4538     6427     6844     9135#
CLR0     007000         1635     1636     1637     1638     1640#
CMPB1    014616         3212     3214#
CMPB2    015510         3441     3442     3444#
CMPB3    016104         3572     3575#
CMPN     023054         4624#
CMP0     013726         2966     2967     2968     2970#
CMP0A    014170         3056     3059#
CMP1     014260         3086     3087     3088     3090#
CMP1A    014400         3131     3132     3133     3135#
CMP2     015142         3322     3323     3325#
CMP7     016672         3710     3711     3713#
CNTRLC=  000003         9165#    9170
CNTRLO=  000017         9166#    9185
CNVADR   051644         6533     7944     8024     8986#
COMB1    010410         2054     2055     2057#
COMB1A   010570         2120     2121     2123#
COMB2    011152         2256     2257     2259#
COMB5    011722         2462     2464#
COMB6    012550         2664     2665     2667#
COMB7    013454         2881     2882     2884#
COM0     007044         1655     1656     1657     1658     1660#
COM1     010150         1963     1964     1966#
COM3     011572         2413     2415#
COM4     010674         2158     2160#
COM6     012136         2539     2540     2542#
COM7     013260         2821     2822     2824#
CONTRL=  177746          203#
CPCHK    003766         1188#
CPUERR=  177766          216#    4444*    5524*    5636*    9367     9372*
CR     = 000015           91#    8246     8256
CRLF   = 000200           92#    1029     1033     1075     1279     1415     6608     6617     8079     8217     8256     9279     9419
                        9433     9444     9451     9464     9477     9486     9498     9517     9690     9695     9698     9702     9706
                        9711     9715     9720     9724     9728     9735     9744     9749     9756     9762     9766     9771     9778
                        9784     9787     9790     9793     9796     9799     9803     9806
DBINB7   017344         3843#    3875*    3876*    3880*    3881*    3882     3886*    3893     3894*    3901*    3903     3904*
DBIN7    016572         3685#    3700*    3704*    3709     3715*    3723*    3730*    3734*    3738
DDATA    016122         3583#    3587*    3589*    3593     3597     3598     3601*    3602     3607*    3614*    3621*    3622     3625*
                        3628     3632
DDATAB   016556         3653*    3654*    3659*    3662     3666*    3667     3671     3676#
DECB1    010360         2040     2041     2043#
DECB1A   010510         2092     2093     2095#
DECB2    011306         2309     2310     2312#
DECB5    012062         2518     2520#
DECB6A   012650         2695     2696     2697     2699#
DECB7    013522         2900     2901     2903#
DEC0     007124         1681     1682     1683     1684     1686#
DEC1     007742         1886     1888#
DEC1A    010212         1984     1986#
DEC2     011042         2218     2219     2221#
DEC5     011546         2399     2400     2401     2403#
DEC6     012304         2588     2589     2591#
```

```
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 189
CEQKCC.P11    03-MAR-78 13:13            CROSS REFERENCE TABLE -- USER SYMBOLS                                      SEQ 0207

DEVICE   001550              662#
DEVIND   001552              663#   6565*   6566*   6567*   6571    6573*   6574*   6575
DF1      055060              916     921     926    9539#
DF10     055756              951    9628#
DF11     056062              956    9643#
DF12     056144              961     966    9653#
DF14     056304              971    9674#
DF16     056351              981    9682#
DF4      055411              931     941    9582#
DF5      055522              936    9597#
DH1      055021              914    9533#
DH10     055707              949    9621#
DH11     056041              954    9640#
DH12     056130              959    9651#
DH13     056210              964    9661#
DH14     056252              969    9668#
DH16     056324              979    9678#
DH2      055126              919    9548#
DH3      055221              924    9560#
DH4      055333              929    9574#
DH5      055455              934    9590#
DH6      055610              939    9609#
DISPLA=  177570               86#   7847*   7871*
DIV0     025532             5247#
DIV1     026000             5330#   5343
DT1      055066              915    9542#
DT10     055762              950    9631#
DT11     056064              955    9645#
DT12     056146              960     965    9655#
DT14     056272              970    9672#
DT16     056356              980    9685#
DT2      055154              920    9553#
DT3      055270              925     930    9568#
DT5      055526              935    9600#
DT6      055630              940    9613#
DUMMY    033204             6307#
ECHO     052560             9206#   9224
EISOPT=  040000              504#
EMTVEC=  000030              189#   1003*   1165*   1166*   4233*   4235*   4272*   4273*
EMT1     021150             4232    4240#
EMT1B    021230             4240    4244    4245    4248    4249    4253    4254    4257    4258    4259    4264#
EMT1C    021234             4238    4266#
EMT1D    021246             4267    4268    4271#
EM1      054774              913    9529#
EM10     055636              948    9614#
EM11     055774              953    9633#
EM12     056072              958    9646#
EM13     056152              963    9656#
EM14     056225              968     978    9664#
EM15     056310              973    9676#
EM2      055100              918    9544#
EM3      055164              923    9555#
EM4      055304              928    9570#
EM5      055416              933    9584#
EM6      055540              938    9602#
```

```
ENDCP     027036           5467    5538#
ENDM      036316           6417    6824#
ENDMEM    036314           6474    6823#
ENDTAG    0?5370           1118    9687#
END1      036356           6842#
ENTER2    034514           6524    6557#   6609    6751
ERPRT     053442           1269    1425    3959    4440    5521    5632    6480    6842    9363#
ERRBA     001672            685#   7664*   7665*   7666*   8138*   8141*   8142*   8143*   8155*   8156*   8157*   8158*   8159*
                           8949    8950
ERRRTN    001332            620#
ERRVEC= 000004              182#   1048*   1049*   1188*   1189*   1269*   1290*   1315*   1318*   1324*   1344*   1364*   1423
                           3917*   3918*   3919    3959*   4353*   4354*   4397*   4440*   4441*   4442*   4443*   5512*   5521*
                           5592*   5593*   5632*   5633*   5634*   5635*   6473*   6474*   6480*   6842*   7811    7812*   7814*
                           7817*
EXIT      036202           6713    6794    6799#
EXITFL    001536            657#
EXITRE    034430           6497    6499    6530    6536#   6801
EXPEXT    051042           6222    6233    6278    8776    8798#
EXTINS    023234           4674#
FACTOR    001506            645#   1460*   1577*   2611    2741*   2854    3307    3308    3585    3645    3758    3759    3763
                           3764    3794*   3864    3918    4043    4057    4065    4081    4106    4143    4339*   4451    4456
                           4467    4667*   4773    5087*   5406    5425    5560*   5878*   6295*   6507    6515*   9005
FLD20 = 104420             8063    8913#
FLTADD    032742           5894    5934    5937    5967    6063    6103    6106    6136    6244#
FLTDBL    050702           6057    8754#
FLTDIV    032714           6027    6196    6230#
FLTMPY    032672           5897    5907    5916    5926    5929    5979    5987    5995    6028    6066    6076    6085    6095
                           6098    6148    6156    6164    6197    6219#
FLTMPO    001422            631#   6067*   6119    6149*   6180    6198*   9685
FLTMP1    001432            632#   6107*   6170*   6201*   9685
FLTSGL    050710           5888    8755#
FLTSUB    032736           5952    5975    6000    6013    6035    6121    6144    6169    6182    6204    6243#
FL20    = 104416           8053    8679    8912#
FPOPT   = 020000            505#   1197    5885
FRSTAD    001512            647#   1457*   1574*   2738*   3385    3791*   4336*   4664*   5084*   5557*   5875*   6500
FRSTME    001514            648#   1118*   6510
GETMAP    052054           6976    7057    7276    7397    7648    8935    9067#
GIVEMA    052320           7274    7310    7395    7431    7642    7675    9126#
GNS     = ****** U          527    1028    1032    1074    1278    1414    6607    6616    6900    6907    8078    8905    8906
                           8907    8908    8909    8910    8911    8912    8913    9278
GSTST     007532           1828#
HALT1     026720           5510#
HIADRS= 177742              201#   9037    9253
HITMIS= 177752              205#
HT      = 000011             89#   8215    8256
INCB1     010274           2009    2010    2012#
INCB2     011402           2344    2345    2346    2348#
INCB3     011754           2476    2478#
INCB6     012462           2638    2640#
INCB6A    012632           2689    2690    2692#
INCB7     013506           2894    2895    2897#
INC0      007144           1690    1691    1692    1693    1695#
INC1      010056           1928    1929    1930    1932#
INC3      011602           2418    2420#
INC4      010724           2171    2172    2174#
```

```
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 191
CEQKCC.P11     03-MAR-78 13:13          CROSS REFERENCE TABLE -- USER SYMBOLS                                    SEQ 0209

INC7     013320            2838     2840#
INSTBL   003172            1055     1062     1082#
IOMON    034436            6481     6546#    6752
IOTTST   020654            4161#
IOTVEC=  000020             187#    1163*    1164*    4161     4174*    4228*    4229*    4274*    4275*    5484*    5503*
IOWC     001546             661#    6523*    6556*    6647
JMP1     020070            3974     3975     3976     3978#
JMP3     020140            3994     3998#
JMP4     020176            4010     4014#
JMP5     020242            4028     4032#
JMP6     020262            4040#
JMP7     020320            4051#
JSR1     020332            4057#
JSR3     020416            4067     4081#
JSR4     020504            4093     4106#
JSR6     020562            4115     4129#
JSR7     020614            4133     4143#
KDPAR0=  172360             359#    5730
KDPAR1=  172362             360#
KDPAR2=  172364             361#
KDPAR3=  172366             362#
KDPAR4=  172370             363#
KDPAR5=  172372             364#
KDPAR6=  172374             365#
KDPAR7=  172376             366#
KDPDR0=  172320             337#    5722
KDPDR1=  172322             338#
KDPDR2=  172324             339#
KDPDR3=  172326             340#
KDPDR4=  172330             341#
KDPDR5=  172332             342#
KDPDR6=  172334             343#
KDPDR7=  172336             344#
KERSTK=  001200              76#    1085     1130     4431     6813     6845
KIPAR0=  172340             348#    5727     6439*    6803*    6810     8995     9297     9322
KIPAR1=  172342             349#    6440*    6804*
KIPAR2=  172344             350#    6441*    6805*
KIPAR3=  172346             351#    6442*    6443     6445     6548     6803
KIPAR4=  172350             352#    5745*    6443*    6444*    6804
KIPAR5=  172352             353#    6445*    6446*    6805
KIPAR6=  172354             354#    9255     9258*    9270*
KIPAR7=  172356             355#    6447*    6458     6469
KIPDR0=  172300             326#    5719     6431*
KIPDR1=  172302             327#    6432*
KIPDR2=  172304             328#    6433*
KIPDR3=  172306             329#    6434*
KIPDR4=  172310             330#    5746*    6435*
KIPDR5=  172312             331#    6436*
KIPDR6=  172314             332#    9259*
KIPDR7=  172316             333#    6437*
KM     = 000000             515#
KTABRT   053274            1443     5502     9334#
KTABT    027716            5743#
KTEX     030156            5744     5795#
KTOPT  = 100000             503#
```

```
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 192
CEQKCC.P11    03-MAR-78 13:13         CROSS REFERENCE TABLE -- USER SYMBOLS                                SEQ 0210

KTPDR    027426          5646#
LDKT     053130          7637     7714     7777     8264     9196     9292#
LDRP3    037356          6990     6996#    7245
LDRP4    040300          7126     7133#    7464     7490     7503     7518     7533
LDRS     040604          7181     7190#    7583     7611
LF     = 000012            90#    8250     8256
LKOPT  = 001000           507#    1201     6320     6365
LKS    = 177546           472#    1199     6330*    6332     6369*    7789*
LKSRV    044444          6351     6367     7776#
LKVEC  = 000100           473#    6323*    6324*    6351*    6367*    6368*
LOADRS = 177740           200#    9036     9251
LOOP     005372          1398     1404     1406     1408     1410     1421#    6816     6864     6922
LSTMEM   001516           649#    1116*    1117*    6512
LTICKS   001602           675#    6672     6689     7778*    7779     7781*    7782*    7783     7785*    8265
MAINT  = 177750           204#    6854*    6866*
MAPEND   051640          8956     8966#
MAPH0  = 170202           443#    5849*
MAPH00 = 170202           379#     443
MAPH01 = 170206           381#     445
MAPH02 = 170212           383#     447
MAPH03 = 170216           385#     449
MAPH04 = 170222           387#     451
MAPH05 = 170226           389#     453
MAPH06 = 170232           391#     455
MAPH07 = 170236           393#     457
MAPH1  = 170206           445#    5847*
MAPH10 = 170242           395#
MAPH11 = 170246           397#
MAPH12 = 170252           399#
MAPH13 = 170256           401#
MAPH14 = 170262           403#
MAPH15 = 170266           405#
MAPH16 = 170272           407#
MAPH17 = 170276           409#
MAPH2  = 170212           447#
MAPH20 = 170302           411#
MAPH21 = 170306           413#
MAPH22 = 170312           415#
MAPH23 = 170316           417#
MAPH24 = 170320           419#
MAPH25 = 170326           421#
MAPH26 = 170332           423#
MAPH27 = 170336           425#
MAPH3  = 170216           449#
MAPH30 = 170342           427#
MAPH31 = 170346           429#
MAPH32 = 170352           431#
MAPH33 = 170356           433#
MAPH34 = 170362           435#
MAPH35 = 170366           437#
MAPH36 = 170372           439#
MAPH37 = 170376           441#
MAPH4  = 170222           451#
MAPH5  = 170226           453#
MAPH6  = 170232           455#
```

```
                                                                          E01
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 193
CEQKCC.P11    03-MAR-78 13:13           CROSS REFERENCE TABLE -- USER SYMBOLS                                   SEQ 0211

MAPL0 = 170200             442#    5804     5837     5848*    8960     9101
MAPL00= 170200             378#     442
MAPL01= 170204             380#     444
MAPL02= 170210             382#     446
MAPL03= 170214             384#     448
MAPL04= 170220             386#     450
MAPL05= 170224             388#     452
MAPL06= 170230             390#     454
MAPL07= 170234             392#     456
MAPL1 = 170204             444#    5846*
MAPL10= 170240             394#
MAPL11= 170244             396#
MAPL12= 170250             398#
MAPL13= 170254             400#
MAPL14= 170260             402#
MAPL15= 170264             404#
MAPL16= 170270             406#
MAPL17= 170274             408#
MAPL2 = 170210             446#
MAPL20= 170300             410#
MAPL21= 170304             412#
MAPL22= 170310             414#
MAPL23= 170314             416#
MAPL24= 170320             418#
MAPL25= 170324             420#
MAPL26= 170330             422#
MAPL27= 170334             424#
MAPL3 = 170214             448#
MAPL30= 170340             426#
MAPL31= 170344             428#
MAPL32= 170350             430#
MAPL33= 170354             432#
MAPL34= 170360             434#
MAPL35= 170364             436#
MAPL36= 170370             438#
MAPL37= 170374             440#
MAPL4 = 170220             450#
MAPL5 = 170224             452#
MAPL6 = 170230             454#
MAPL7 = 170234             456#
MAPTBL  001656             681#    1143*    1144*    9096     9099*    9128*
MAPTST  030176            5804#
MAPTWO  030324            5803     5839     5844#
MARKEX  024256            4935     4939     4943#
MARK1   024232            4931     4933#
MBRK    026446            5448#
MBTAS = 160116             493#
MBTBA = 160104             488#     881
MBTBAE= 160174             497#     882
MBTCS1= 160100             486#     879
MBTCS2= 160110             490#     884
MBTCS3= 160176             498#     887
MBTDB = 160120             494#
MBTDT = 160126             496#     890
MBTER = 160114             492#     886
```

```
MBTMR1= 160124            495#
MBTMR2= 160106            489#     883
MBTN2   002276            891#    1246
MB.N3   002300            892#    1250
MBTN4   002302            893#    1254
MBTOPT= 002000            506#    1244     1268     6384
MBTPSW= 000776            500#     889
MBTSET  033506           6374     6376     6384#
MBTSRV  044132           6397     7713#
MBTST = 160112            491#     885
MBTTBL  002246            879#    1242     1260     1264*    1265*    1266     6394*    6395*    6396*    6397*    6398*    6399*    7716
                         7721*    7753     7767*    9407
MBTVEC= 000774            499#     888
MBTWC = 160102            487#     880
MBT1    033552           6394#
MEMERR= 177744            202#    9035     9041*    9250     9264*    9265     9271     9272*
MEMHOL  044300           7732     7734     7741#
MEMSIZ  003326           1112     1114     1116#
MHOLE   044002           7671     7673     7683#
MMON    001503            642#    1132*    4496     5474     5651     5689     5743     6392     6472*    6498     6618     6664     6696
                         6722     6749     6765     6799     6834*    8988     9256     9292     9315
MMR0  = 177572            227#     231     9337     9343*
MMR1  = 177574            228#     232
MMR2  = 177576            229#     233     9338
MMR3  = 172516            230#     234     5802     6377     6825*    6972     7053     7271     7307     7392     7428     8952
MMVEC = 000250            195#    1443*    1444*    5470*    5502*    5751     5752     5755*    5756*    5792*    5793*
MOVB1   014572           3197     3198     3199     3200     3203#
MOV0    013656           2944     2945     2947#
MOV0A   013706           2959     2961#
MOV1    014450           3156     3158#
MOV7    016642           3701     3703#
MPI     026514           5466#
MRGTAB  053552           6865     6866     9391#
MRKTST  024174           4920#
MSGINX  053612           6601     8044     8072     8333     9409#
MSG1    053636           9419#
MSG10   054102           9413     9449#
MSG11   054110           9414     9450#
MSG12   054116           8083     9451#
MSG13   054231           8090     9464#
MSG14   054341           8173     9477#
MSG15   054377           9417     9483#
MSG16   054420           8108     9486#
MSG17   054527           8114     9498#
MSG2    053650           9421#
MSG20   054603           6871     9506#
MSG21   054703           9411     9412     9415     9416     9517#
MSG22   054723           8129     9520#
MSG23   054733           8134     9522#
MSG24   054742           9418     9524#
MSG25   054764           1399     9527#
MSG3    053662           7932     9423#
MSG4    053751           8324     9433#
MSG5    054066           9409     9447#
MSG6    054074           9410     9448#
```

```
MULO      025402                5196#
MXMMHI    001560                 666#     1127*     7672      7733
MXMMLO    001562                 667#     1128*     7670      7731
NEGB1     010344                2033      2034      2036#
NEGB4     011332                2321      2322      2324#
NEGB6     012566                2671      2672      2674#
NEGB7     013570                2919      2920      2922#
NEGO      007164                1700      1701      1702      1704#
NEG1      010202                1978      1979      1981#
NEG2      010746                2183      2185#
NEG5      011560                2406      2407      2409#
NEG6      012172                2551      2552      2553      2555#
NEG7      013226                2808      2809      2811#
NEXEC     001502                 641#     1464      1581      2745      3798      4343      4671      5091      5564      5882      6817
NEXPAR    001520                 650#     1133*     1136*     6442      6453      6464      6802*     6830*     6833*
NOMEM     034432                6513      6535      6537#
NOTYPE    001466                 635#     1184*     8241      9188*     9206
NULLS     001526                 653#     6302      9227
NWBASH    001544                 660#     6520*     6552*     6661      6670*
NWBASL    001542                 659#     6519*     6551*     6621      6660      6669*     6702
OAERR     017626                3910#
OLDBAS    001540                 658#     6518*     6547*     6620      6659      6668*     6698
OLDPSW    033652                6425#     6815
OPT.CP    001470                 636#     1271*     1400      4452      5885      6297      6320      6365      6373      6384
OVFLW     021600                4347#
PARTBL    027670                5684      5727#
PA1500    001476                 639#     7668*     7670      7691      7727*     7729*     7731      7749      7947*     7950      8025      8037*     8145
                                8954*     8961*     8964*     9002*
PA2116    001500                 640#     7669*     7672      7692      7728*     7730*     7733      7750      7948*     8038*     8955*     8962*     8965*
                                9001*
PDRTBL    027652                5646      5719#
PERET     052756                9265#
PHYMAP    051536                7667      8144      8948#
PIRQ    = 177772                  84#     5400      5419      5420*
PIRQVE= 000240                   194#     5402*     5403*     5405*     5406*     5410*     5438*     5439*
PIRQO     026226                5396#
PKM     = 000000                 517#
PLKCSB= 172542                   470#
PLKCSR= 172540                   469#
PLKVEC= 000104                   471#
PROT      003244                1096#     1115*     1117      1134      6831
PR0     = 000000                 116#
PR1     = 000040                 117#
PR2     = 000100                 118#     5030      5038
PR3     = 000140                 119#
PR4     = 000200                 120#     1181      1183      4163      4181      5055      5061      6426
PR5     = 000240                 121#     4170      5028      6398      6656
PR6     = 000300                 122#
PR7     = 000340                 123#     1426      1432      1437      1439      1441      1445      4169      4219      4229      4273      4282      4283
                                4313      4315      4350      4442      4978      4980      4986      5006      5044      5049      5056      5356      5403
                                5404      5441      5634      6324      6368      8930      9071
PS      = 177776                  81#       82      1157*
PSM     = 010000                 512#     4196
PSW     = 177776                  82#     1050*     1071*     1416*     1424      1445*     1828      1842*     1847*     4220*     4235      4282*     4288
                                4316*     4347      4348*     4350*     4378*     4383*     4441      4498      4499*     4502      4536      4537*     4600*
```

```
                         5109*   5111    5127    5133    5283    5287    5307    5311    5402    5407*   5440*   5466    5501*
                         5523*   5531    5534    5537*   5593    5600    5633    5756    5766    6313    6317*   6336*   6424
                         6426*   6814*   6843*   7225    7263    7304    7344    7384    7425    7466    7505    7535    7564
                         7597    7627    9069    9114*   9135    9146*   9295    9296*   9326*
PSWCHK   022404          4496#
PSWTAB   053536          6863    9383#
PUM    = 030000           518#   4195    4201    5028    5044    5049    5055    5061
PWRVEC=  000024           188#   1169*   1170*   8843*   8844*   8852*   8867*   8868*
QV       001504           643#   1091*   1099*   1111    1403    5844    6610    6867    9084
REG    = 004000           513#   5017    5020    5028    5049    5061
REGINX   053566          8095    8110    8163    9399#
RELE0    006536          1466    1556#
RELE1    012720          1583    2720#
RELE2    017110          2747    3773#
RELE3    021470          3800    4318#
RELE4    023124          4345    4646#
RELE5    024710          4673    5066#
RELE6    027036          5093    5539#
RELE7    030354          5566    5845    5850#
RELE8    033100          5884    6209    6211    6282#
RELNIO   034374          6517    6526#   6528    6625
RELOC    034242          1559    2723    3776    4321    4649    5069    5542    5853    6285    6496#
RELOCP   036144          6555    6629    6789#   6791
REL0     005542          1455#
REL00    006552          1560#
REL1     006572          1572#
REL11    012734          2724#
REL2     012754          2736#
REL22    017124          3777#
REL3     017144          3789#
REL33    021504          4322#
REL4     021524          4334#
REL44    023140          4650#
REL5     023160          4662#
REL55    024724          5070#
REL6     024744          5082#
REL66    027052          5543#
REL7     027072          5555#
REL77    030370          5854#
REL8     030410          5873#
REL88    033114          5887    6286#
RESERR   053370          1270    1429    3960    4490    5522    9350#
RESET1   027010          5530#
RESKT    053216          7652    7678    7705    7719    7765    7787    8308    9204    9315#
RESPSW   052354          9142#   9148
RESREG=  104414          6933    6986    7010    7063    7083    7125    7150    7180    7653    7679    7706    7720    7766
                         7788    7998    8309    8525    8651    8704    8911#   8966    9003    9113    9205
RESTPS   052364          9146#
RESTRP   022224          4450#
RESVEC=  000010           183#   1035*   1036*   1190*   1191*   1270*   3919*   3960*   4455*   4456*   4490*   5513*   5522*
                         6656*   6657*   6769*
RETPC    034434          6509*   6538#
RETPSW   052346          9137#   9147
RETRY    034202          6473#   6798
RKBA     002132           825#   7068*   7365*   7407*
```

I01

```
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 197
CEQKCC.P11     03-MAR-78 13:13            CROSS REFERENCE TABLE -- USER SYMBOLS                                SEQ 0215

                               7387    7408#   7414    7420#   7422    8130
RKDA     002134                 826#   1294#   7064#   7363#   7405#
RKDRV    037410                 761    7006#
RKDS     002122                 821#   1298    7065    9400
RKER     002124                 822#   1296
RKERR    041620                7338    7347#   7377    7418
RKFUN    002062                 784#   7071#   7319#   7320    7325#   7327    7378#   7419#   7501#
RKHANA   002030                 761#
RKHDA    002046                 773#   7051#   7064    7363    7405
RKHSTA   001700                 692#   7007#   7008    7013#   7333    7347#   7354#   7357#   7370    7412    7433#
RKHWC    001720                 705#   7067    7364    7406
RKLOOP   041646                7334    7354#   7371    7413
RKNEWH   001770                 735#   7399    7401#   7404
RKNEWL   001766                 734#   7396    7407
RKOLD    001736                 717#   7056    7059    7062#   7068    7362    7365
RKPSW    002140                 828#   7070#
RKREAD   042176                7330    7412#
RKRPT    041462                7011    7318#
RKSRV    041512                7069    7325#
RKTRY    002073                 796#   7052#   7337    7343#   7360#   7376    7383#   7391#   7416    7424#
RKUNIT   002014                 750#   7048
RKVEC    002136                 827#   7069#
RKWC     002130                 824#   7067#   7364#   7406#
RKWRCK   041746                7328    7370#
RKWTRY   037724                7064#   7323    7345
RK1      041672                7321    7336    7360#
RK10     001570                 670#   7058#   7061#   7072    7361#   7362#   7366    7403#   7404#   7408
RK11     001572                 671#   7012#   7030    7349    7374
RK2      041714                7363#   7385
RK3      042144                7324    7405#   7426
RNTBIN   001556                 665#   6558#   6652#   6653    6666    6935    7012    7085    7152
ROLB1    010306                2015    2016    2018#
ROLB2    011224                2279    2280    2281    2283#
ROLB3    012030                2501    2502    2504#
ROLB6    012602                2677    2678    2680#
ROLB6A   012676                2707    2708    2710#
ROLB7    013606                2926    2927    2929#
ROL0     007216                1716    1717    1719#
ROL1     007776                1900    1901    1902    1904#
ROL1A    010010                1907    1908    1910#
ROL3     011630                2430    2431    2433#
ROL4     011022                2208    2209    2211#
ROL6     012122                2532    2533    2534    2536#
ROL7     013336                2844    2845    2847#
RORB1    010374                2047    2048    2050#
RORB1A   010464                2080    2081    2083#
RORB4    011206                2271    2272    2273    2275#
RORB5    011772                2482    2483    2484    2486#
RORB6    012534                2658    2659    2661#
RORB7    013472                2888    2889    2891#
ROR0     007102                1672    1673    1674    1675    1677#
ROR1     007726                1878    1879    1881#
ROR1A    010040                1920    1921    1922    1924#
ROR2     010710                2164    2165    2167#
ROR5     011502                2379    2380    2382#
```

```
ROR7      013210                2801    2802    2804#
RPCC      002172                 843#
RP3BA     002110                 814#   6999*   7288*
RP3CS     002104                 812#   1328*   1329    6989*   6994*   7217    7221*   7222    7236    7246*   7252    7259*   7260
                                7289*   7295    7300*   7301    8133
RP3DA     002112                 815#   6970*   6996*   7285*
RP3DC     002114                 816#   6997*   7286*
RP3DRV    036772                 760    6929#
RP3DS     002100                 810#   1335    6987    9399
RP3ER     002102                 811#   7266
RP3ERR    040774                7220    7228#   7257    7298
RP3FUN    002060                 783#   6993*   7202*   7203    7208*   7209    7258*   7299*
RP3HAN    002026                 760#   6657    6769
RP3HDA    002042                 771#   6996    7285
RP3HDC    002044                 772#   6969*   6997    7286
RP3HST    001676                 691#   1146    6630    6635    6681    6754*   6768*   6771    6773    6930*   6931    7215    7228*
                                7235*   7238*   7250    7293    7312*
RP3HWC    001716                 704#   6647*   6998    7287
RP3LOO    041022                7216    7235#   7251    7294
RP3NWH    001764                 733#   6661*   7278    7280*   7283
RP3NWL    001762                 732#   6660*   7275    7288
RP3OLD    001732                 715#   6659*   6662*   6975    6978    6981*   6999    7243
RP3PSW    002120                 818#   6992*
RP3REA    041350                7212    7293#
RP3RPT    040644                6934    7201#
RP3SRV    040674                6991    7208#
RP3TRY    002072                 795#   6971*   7219    7224*   7241*   7256    7262*   7270*   7297    7303*
RP3UNI    002012                 749#   6655*   6982    6985*   6989    7244    7284
RP3VEC    002116                 817#   6991*
RP3WC     002106                 813#   6998*   7287*
RP3WCK    041112                7210    7250#
RP3WTR    037310                6987#   6988    7206    7226
RP31      041046                7204    7218    7241#
RP310     001564                 668#   6977*   6980*   6984*   6994    7242*   7243*   7244*   7246    7282*   7283*   7284*   7289
RP311     001566                 669#   6935*   6950    7230    7254
RP32      041076                7245#   7264
RP33      041310                7207    7285#   7305
RP4BA     002146                 833#   7139*   7520*
RP4BAE    002150                 834#   7138*   7519*
RP4CS1    002142                 831#   1319    1349*   7130*   7470*   7491*   7521*   9403
RP4CS2    002154                 836#   1348*   1350    1358*   6764*   7133*   7463*   7502*   7508    7532*
RP4CS3    002156                 837#
RP4DA     002152                 835#   7136*
RP4DC     002164                 840#   7135*
RP4DRV    040004                 764    7079#
RP4DS     002160                 838#   1352    7459    7471    7482    7488    7497    7516    7527
RP4ERR    042504                7462    7474#   7500    7530
RP4ER1    002162                 839#
RP4ER2    002166                 841#
RP4ER3    002170                 842#
RP4FUN    002066                 786#   7129*   7446*   7447    7451*   7452    7531*
RP4HAN    002036                 764#
RP4HDA    002052                 775#   7123*   7136
RP4HDC    002054                 776#   7122*   7135
RP4HST    001706                 695#   7080*   7081    7086*   7457    7468    7474*   7481*   7484*   7495    7525    7538*
```

```
RP4LOO  042532            7458     7481#    7496     7526
RP4NWH  002004             741#    7519
RP4NWL  002002             740#    7520
RP4OF   002174             844#    7134*
RP4OLD  001752             723#    7138     7139
RP4PSW  002200             846#    7128*
RP4REA  042756            7454     7525#
RP4RPT  042340            7084     7445#
RP4SRV  042364            7127     7451#
RP4TRY  002075             798#    7124*    7461     7465*    7487*    7499     7504*    7515*    7529     7534*
RP4UNI  002022             753#    7133
RP4VEC  002176             845#    7127*
RP4WC   002144             832#    7137*
RP4WCK  042610            7453     7495#
RP4WTR  040246            7126#    7450     7467
RP41    042560            7448     7460     7487#
RP411   001574             672#    7085*    7102     7476     7510
RP42    042566            7488#    7489     7506
RP43    042720            7449     7516#    7517     7536
RSBA    002206             851#    7194*    7613*
RSBAE   002210             852#    7193*    7612*
RSCS1   002202             849#    1369*    7187*    7584*    7614*    9404
RSCS2   002214             854#    1368*    1370     1378*    6762*    7190*    7562*    7595*    7601     7625*
RSCS3   002216             855#
RSDA    002212             853#    7191*
RSDRV   040354             765     7146#
RSDS    002220             856#    1372     7185     7558     7575     7581     7590     7609     7620
RSER    002222             857#
RSERR   043164            7561     7567#    7593     7623
RSFUN   002070             787#    7184*    7545*    7546     7550*    7551     7594*    7624*
RSHANA  002040             765#
RSHDA   002056             777#    7178*    7191
RSHSTA  001710             696#    7147*    7148     7556     7567*    7574*    7577*    7588     7618     7630*
RSHWC   001730             709#    7192
RSLOOP  043212            7557     7574#    7589     7619
RSNEWH  002010             743#    7612
RSNEWL  002006             742#    7613
RSOLD   001756             725#    7193     7194
RSPSW   002226             859#    7183*
RSREAD  043432            7553     7618#
RSRPT   043050            7151     7544#
RSSRV   043074            7182     7550#
RSTRY   002076             799#    7179*    7560     7563*    7580*    7592     7596*    7608*    7622     7626*
RSUNIT  002024             754#    7190
RSVEC   002224             858#    7182*
RSWC    002204             850#    7192*
RSWCK   043270            7552     7588#
RSWTRY  040544            7181#    7549     7565
RS11    001576             673#    7152*    7167     7569     7603
RS41    043240            7547     7559     7580#
RS42    043246            7581#    7582     7598
RS43    043374            7548     7609#    7610     7628
RTI1    036556            6868     6873#
RTT1    024270            4970#
RTT1EX  024466            5007#
```

```
                                                 L01
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 200
CEQKCC.P11    03-MAR-78 13:13          CROSS REFERENCE TABLE -- USER SYMBOLS                               SEQ 0218

RTT2EX   024702          4972    5024    5039    5041    5050    5052    5062    5065#
RT1      053122          9246    9284#
RUNTBL   001626           679#   6654*   6729    6951    6954*   7031    7034*   7103    7106*   7169*
RUNTRA   001642           680#   6784    6968*   7045*   7121*   7175*   7231*   7350*   7477*   7570*
SAVPSW   024500          4970*   4971    4976    5001    5014#   5015    5065
SAVREG=  104412          6929    7006    7079    7146    7636    7713    7776    7929    8263    8507    8601    8676    8910#
                         8948    8986    9073    9194
SBCB1    010320          2021    2022    2024#
SBCB3    012050          2512    2514#
SBCB4    011320          2315    2316    2318#
SBCB6    012436          2628    2629    2631#
SBCB7    013420          2866    2867    2868    2870#
SBC0     007246          1730    1731    1733#
SBC1     010072          1935    1936    1937    1939#
SBC1A    010110          1943    1944    1945    1947#
SBC5     011642          2437    2439#
SBC6     012226          2566    2567    2569#
SBC7     013114          2771    2772    2774#
SBINB7   017334          3839#   3870*   3871    3875    3882    3886    3890*   3893*   3896    3900*   3903*   3906
SBIN7    016570          3684#   3699*   3700    3709    3715    3722*   3723    3730    3734    3738
SDATA    016120          3582#   3588*   3589    3590*   3591*   3596*   3597*   3598    3602    3607    3613*   3614    3620*
                         3625    3626*   3627*   3628    3631*   3632    3635
SDATAB   016554          3651*   3652*   3653    3655    3659    3662    3666    3667    3675#
SDPAR0=  172260           315#   5732
SDPAR1=  172262           316#
SDPAR2=  172264           317#
SDPAR3=  172266           318#
SDPAR4=  172270           319#
SDPAR5=  172272           320#
SDPAR6=  172274           321#
SDPAR7=  172276           322#
SDPDR0=  172220           293#   5724
SDPDR1=  172222           294#
SDPDR2=  172224           295#
SDPDR3=  172226           296#
SDPDR4=  172230           297#
SDPDR5=  172232           298#
SDPDR6=  172234           299#
SDPDR7=  172236           300#
SECT2    031074          5958    5961#
SECT2D   032200          6127    6130#
SECT3    031440          6019    6023#
SECT3D   032544          6188    6192#
SIPAR0=  172240           304#   5478    5729    6464*   6465    6467
SIPAR1=  172242           305#   6465*   6466*
SIPAR2=  172244           306#   6467*   6468*
SIPAR3=  172246           307#
SIPAR4=  172250           308#   5749*
SIPAR5=  172252           309#
SIPAR6=  172254           310#   5478*
SIPAR7=  172256           311#   6469*
SIPDR0=  172200           282#   5721    6460*
SIPDR1=  172202           283#   6461*
SIPDR2=  172204           284#   6462*
SIPDR3=  172206           285#
```

```
                                                                        M01
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 201
CEQKCC.P11     03-MAR-78 13:13            CROSS REFERENCE TABLE -- USER SYMBOLS                                              SEQ 0219

SIPDRS= 172212            287#
SIPDR6= 172214            288#   5479*
SIPDR7= 172216            289#   6463*
SIZE    004470           1290#   1418
SIZEHI= 177762            213#
SIZELO= 177760            211#   1122
SM    = 040000            511#   5020    5061
SOBDBL  051040           8754*   8755    8761    8773    8777*   8779#
SOB0    023766           4833    4844#
SOB1    023774           4846#   4850
SOB10   023750           4835#   4898
SOB2    024022           4846    4847    4848    4849    4851    4852    4853    4854    4857#
SOB3    024034           4861#   4869
SOB4    024066           4861    4862    4863    4864    4867    4870    4871    4874#
SOB5    024070           4876#
SOB5A   024110           4885#
SOB6    024134           4896#
SOB7    024142           4899#
SOB8    024144           4842    4901#
SOB9    023760           4840#   4911
SPARE0  001702            693#
SPARE1  001704            694#
SPCHK   022642           4558#
SPLTST  026056           5353#
SR0   = 177572            231#   5771    5794*   6470*   6824*
SR1   = 177574            232#   5777    5780
SR2   = 177576            233#   5774
SR3   = 172516            234#   6471*   8938*
STACK = 001200             75#     76      77      78    1162
START   003212            531    1077    1085#   8872    9281
START1  002464            533     982#    983
START2  002474            534    1001#   1022
STKLMT= 177774             83#   5568
STMM    033614           6378    6385    6387    6389    6391    6393    6412#   6819
SUBFLG  033076           6243*   6272    6279*   6281#
SUBPAS  001532            655#   1149*   6388    6848*   6849    6853*   6861    7968
SUB0    013674           2952    2953    2955#
SUB1    014316           3104    3105    3107#
SUB1A   014424           3143    3144    3146#
SUB1B   014440           3150    3151    3153#
SUB2    015170           3334    3335    3337#
SUB2A   015310           3376    3377    3379#
SUB3    015626           3484    3485    3487#
SUB3A   015650           3492    3493    3495#
SUB6    016300           3615    3617#
SUB7    016712           3716    3717    3718    3720#
SUPSTK= 000700             77#   1421    3958    4221    4384    4390    4425    4439    5197    5504    5606    5626    5630
                         5805    9366

SWAB0   007264           1737    1738    1739    1741#
SWAB1   010524           2099    2100    2102#
SWAB2   010736           2178    2180#
SWAB4   011364           2338    2340#
SWAB6   012662           2702    2704#
SWAB7   013242           2815    2817#
SWITCH  056372           8323    9690#
```

```
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 202
CEQKCC.P11    03-MAR-78 13:13           CROSS REFERENCE TABLE -- USER SYMBOLS                           SEQ 0220

                         6561     6565     6714     6778     6936     7014     7087     7153     7806     7824     7831     7872     7875
                         7882     7886     7892
SW0     = 000001          151#
SW00    = 000001          141#     151
SW01    = 000002          140#     150
SW02    = 000004          139#     149
SW03    = 000010          138#     148
SW04    = 000020          137#     147
SW05    = 000040          136#     146
SW06    = 000100          135#     145
SW07    = 000200          134#     144
SW08    = 000400          133#     143
SW09    = 001000          132#     142
SW1     = 000002          150#
SW10    = 002000          131#
SW11    = 004000          130#
SW12    = 010000          129#    6375
SW13    = 020000          128#
SW14    = 040000          127#    7806
SW15    = 100000          126#
SW2     = 000004          149#
SW3     = 000010          148#    6386
SW4     = 000020          147#    6936     7014     7087     7153
SW5     = 000040          146#    6561
SW6     = 000100          145#    6415     6496
SW7     = 000200          144#    1409
SW8     = 000400          143#    6516     6553
SW9     = 001000          142#    6714
SXRA      023610         4783#    4792*
SXRB      023612         4784#    4810
SXT0      023256         4677     4678     4679     4680     4683#
SXT1      023342         4710#
SXT2      023626         4790#
SXT3      023642         4796#
SXT4      023410         4728     4730#
SXT5      023652         4801#
SXT6      023552         4765     4766     4767     4768     4770#
SXT6A     023602         4776     4779#
SXT7      023716         4818#
SYSSIZ    001606          678#    1307*    1341*    1361*    1381*    1386*    1394*    1396*    6578     6581     6591     6598*    6640*
                         8327     8338
SYSTID= 177764            215#
TBITVE= 000014            184#    4283*    4315*    4501     4512*    4515*    4530*    4975*    4977*    5004*    5005*    5006*    5404*
                         5441*    6860*
TEST1     010166         1970     1971     1972     1974#
TEST2     010664         2152     2153     2155#
TEST6     012712         2713     2714     2716#
TIMEBU    047106         1150     1155*    1156*    8269*    8274*    8279*    8284*    8291*    8298*    8305*    8306     8311#
TIMER     042322         7340     7380     7421     7435#
TKBFR     001444          633      634#    9200     9202
TKBFRP    001442          633#    9197     9203*
TKISR     052400         1180     9168#
TKVEC   = 000060          191#    1180*    1181*
TPISR     052612         1182     6355     9222#
TPVEC   = 000064          192#    1182*    1183*    6322*    6355*
```

```
TRAP1    021402          4285     4300#
TRAP1C   021442          4299     4312#
TRTVEC=  000014           185#
TSTB1    010536          2105     2106     2108#
TSTB2    011342          2327     2329#
TSTB2A   011352          2332     2334#
TSTB6    012410          2616     2617     2618     2620#
TST0     007022          1645     1646     1647     1648     1650#
TST1     005524          1449#
TST10    011104          2242#
TST11    011422          2359#
TST12    011644          2443#
TST13    012064          2524#
TST14    012344          2608#
TST15    012736          2730#
TST16    013340          2851#
TST17    013634          2940#
TST2     006554          1566#
TST20    014172          3063#
TST21    014516          3182#
TST22    015046          3303#
TST23    015332          3390#
TST24    015536          3460#
TST25    015726          3523#
TST26    016106          3579#
TST27    016404          3640#
TST3     006750          1629#
TST30    016560          3682#
TST31    016776          3745#
TST32    017126          3783#
TST33    017322          3836#
TST34    020030          3964#
TST35    020322          4055#
TST36    020644          4159#
TST37    021070          4176     4225#
TST4     007266          1745#
TST40    021276          4280#
TST41    021506          4328#
TST42    022214          4448#
TST43    022374          4494#
TST44    022602          4542#
TST45    023010          4608#
TST46    023142          4656#
TST47    023734          4829#
TST5     007662          1867#
TST50    024164          4918#
TST51    024260          4968#
TST52    024466          5011#
TST53    024726          5076#
TST54    025372          5194#
TST55    025522          5245#
TST56    025770          5328#
TST57    026046          5351#
TST6     010214          1990#
TST60    026216          5394#
```

```
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 204
CEQKCC.P11    03-MAR-78 13:13              CROSS REFERENCE TABLE -- USER SYMBOLS                                        SEQ 0222

TST62     026504               5464#
TST63     026710               5508#
TST64     027000               5528#
TST65     027054               5549#
TST66     027416               5643#
TST67     027534               5682#
TST7      010622               2140#
TST70     027706               5717     5741#
TST71     030156               5800#
TST72     030372               5867#
TST73     031566               6041     6054#
TST74     033116               6292#
TST75     033376               6362#
TTOPT  =  000400                508#    1205     6297
TTYCHK    033134               6295#
TYPDS  =  104410               6904     6911     8909#
TYPE   =  104400               1026     1030     1072     1276     1399     1402     1412     6603     6605     6614     6898     6905     6912
                               7877     7885     7930     7932     7933     7937     7954     7956     7959     7963     7967     7971     7975
                               7988     7990     7993     7995     7999     8019     8029     8046     8056     8066     8074     8076     8083
                               8090     8108     8114     8119     8129     8134     8149     8151     8168     8170     8171     8173     8176
                               8220     8306     8307     8323     8324     8335     8342     8345     8410     8486     8869     8905#    9276
TYPOC  =  104402               1401     7936     7958     7962     7966     7970     7974     8016     8118     8167     8175     8906#
TYPON  =  104406               8908#
TYPOS  =  104404               8907#
TYPSIZ    047120               1411     8323#
TYPTIM    046660               6887     7931     8263#
UBEADR    001666                684#    7645*    7646*    7647     7649     7650     7700     7702     8934     8936     8937
UBEBA  =  170004                478#     867
UBECC  =  170002                477#     866
UBECLR=   170010                480#     870
UBECR1=   170006                479#     869     1210*    1211
UBECR2=   170016                482#     868
UBEDB  =  170000                476#    1208     1216     1221     1226
UBEERR    044006               7663     7684#
UBEGO  =  170014                481#
UBEINI    051436               6379     7677     8919#
UBEOPT=   000200                509#    1213
UBESAV    001662                683#    7643*    7644*    7645     7646     8919
UBESET    033446               6366     6373#
UBESRV    043520               7636#    8929
UBETBL    002230                866#    1232     7638     7654*    7680*    7698     8154     8927     9408
UBEVEC=   000510                483#     871      872
UBE2      043640               7640     7658#
UBE3      044056               7685     7690     7696#
UBM6      012716               2610     2612*    2615     2623     2627*    2634*    2636*    2642*    2650*    2657*    2663*    2670*    2676*
                               2682*    2688*    2694*    2701*    2706*    2712     2719#
UBREAK=   177770                519#    1005*    1006     1009*    1010     1013*    1014     1017*    1018     1021     1025*    1040*    1042*
                               1044*    1046*    1059*    1065*    1070*    5448
UBRERR    002602               1007     1011     1015     1020#
UBRK2     002624               1019     1025#
UBRTBL    003160               1039     1061     1078#
UB7       016600               3674     3689#
UDPAR0=   177660                271#    5731
UDPAR1=   177662                272#
UDPAR2=   177664                273#
```

```
UDPAR4= 177670                    275#
UDPAR5= 177672                    276#
UDPAR6= 177674                    277#
UDPAR7= 177676                    278#
UDPDR0= 177620                    249#  5723
UDPDR1= 177622                    250#
UDPDR2= 177624                    251#
UDPDR3= 177626                    252#
UDPDR4= 177630                    253#
UDPDR5= 177632                    254#
JDPDR6= 177634                    255#
UDPDR7= 177636                    256#
UIPAR0= 177640                    260#  5476    5728    6453*   6454    6456
UIPAR1= 177642                    261#  6454*   6455*
UIPAR2= 177644                    262#  6456*   6457*
UIPAR3= 177646                    263#
UIPAR4= 177650                    264#  5747*
UIPAR5= 177652                    265#
UIPAR6= 177654                    266#  5476*
UIPAR7= 177656                    267#  6458*
UIPDR0= 177600                    238#  5720    6449*
UIPDR1= 177602                    239#  6450*
UIPDR2= 177604                    240#  6451*
UIPDR3= 177606                    241#
UIPDR4= 177610                    242#  5748*
UIPDR5= 177612                    243#
UIPDR6= 177614                    244#  5477*
UIPDR7= 177616                    245#  6452*
UM    = 140000                    516#  4188    4364    5017    5023    5049    5367    5381    5466    5600    5609
UNITNO  001554                    664#  6646*   6650    6655
USESTK= 000600                     78#
UWM6    012076                   2527#  2531*   2538*   2543*   2550*   2558*   2565*   2572*   2579*   2581*   2587*   2593*   2599*
UWM7    013034                   2750#  2853    2856*   2861*
UW7     013040                   2748   2753#
VADR    001474                    638#  6532*   6719*   6796*   7938    7939*   7945*   7947    7949*   8987    9038*   9248*   9249*
                                 9335*  9336*   9351*   9352*   9364*   9365*   9542    9553    9568    9600    9613    9631
X       053124                   9282   9285#
XOR0    023310                   4691   4692    4693    4696#
XOR1    023376                   4716   4717    4718    4719    4721    4724#
XOR24   023432                   4738#
XOR35   023702                   4807   4808    4812#
XOR6    023520                   4743   4746    4748    4755#
XOR6A   023524                   4741*  4742*   4747*   4749*   4753    4759#   4764*   4775*   4777*
XOR6B   023526                   4750*  4751*   4752*   4753    4760#
XOR7    023732                   4823#
XXDP    003240                   1093#  1113
XXDPC   003241                   1094#  1108*   1405
$AC0    001402                    623#  5893*   5896*   5903*   5905    5914*   5915    5924*   5928*   5932*   5936*   5951*   5966*
                                 5974*  5978*   5984*   5986    5992*   5994    5999*   6012*   6026*   6033*   6062*   6065*   6072*
                                 6074   6083*   6084    6093*   6097*   6101*   6105*   6120*   6135*   6143*   6147*   6153*   6155
                                 6161*  6163    6168*   6181*   6195*   6202*   6219    6230    6244    6249    6260    6262    8827
$AC1    001404                    624#  5892*   5896    5899    5905*   5909    5915*   5918    5925*   5939    5947    5953*   5956*
                                 5957   5965*   5969    5972*   5981    5986*   5989    5994*   5996    6002    6008    6014*   6017*
                                 6018   6025*   6030    6036*   6038*   6039    6061*   6065    6068    6074*   6078    6084*   6087
                                 6094*  6108    6116    6122*   6125*   6126    6134*   6138    6141*   6150    6155*   6158    6163*
```

E02

```
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 206
CEQKCC.P11     03-MAR-78 13:13          CROSS REFERENCE TABLE -- USER SYMBOLS                              SEQ 0224

                          6244     6248     6261     6262*
$AC2      001406           625#    5909*    5936     5944     5946*    5947*    5948*    5949*    5957     5969*    5978     5989*    5999
                          6005     6007*    6008*    6009*    6010*    6018     6078*    6105     6113     6115*    6116*    6117*    6118*
                          6126     6138*    6147     6158*    6168     6174     6176*    6177*    6178*    6179*    6187
$AC3      001410           626#    5918*    5932     5944     5946     5996*    6005     6007     6087*    6101     6113     6115     6165*
                          6174     6176
$AC4      001412           627#
$AC5      001414           628#
$BDADR    001224           584#
$BDDAT    001230           586#    7692*    7750*    9075*    9104
$BELL     001322           616#    7877     7900
$BUFF     001400           622#    8682*    8707
$CHARC    046654          8230*    8237     8248*    8253#
$CMTAG    001200           571#    1157     1158     1165     1171     1172     1173
$CM1    = 000010           598#     599#     600#     601#     602#     603#     604#     605#     606#
$CM2    = 000020           598#     599#     600#     601#     602#     603#     604#     605#     606#
$CM3    = 000010           596#     598
$CM4    = 000010           606#     607#     608#     609#     610#     611#     612#     613#     614#
$CRLF     001327           618#    1402     6912     7885     7900     7930     7933     7975     7990     7995     7999     8151     8170
                          8171     8220     8256     8307     8345     9172     9180     9190
$DBLK     047702          8452     8486     8494#
$DB20     047712          7951     8026     8146     8507#
$DOAGN    036762          6894     6915     6921#
$DTBL     047672          8455     8490#
$ENDAD    036752           557     1097     1274     6917#    7889
$ENDCT    036616          1171     6896#
$ENULL    036766          6923#
$EOP      036560          6857     6886#
$EOPCT    036610          1171*    6893#    6897
$ERFLG    001204           574#    6711     7798     7820     7822     7828*    7844     7846     7851     7868     7869*    7900
$ERMAX    001217           581#    1174*    7822     7843*    7851
$ERPSW    001534           656#
$ERROR    044774          1003     1165     4228     4272     7867#
$ERRPC    001220           582#    7879*    7880*    7881     7900     7934     7939
$ERRTB    002304           911#    7985
$ERRTY    045160          7884     7929#
$ERTTL    001214           579#    6909     6913*    7878*    7900
$ESCAP    001320           615#    1173*    7842*    7895     7897     7900
$FACTO    001510           646#    6506*    6622     6700     6703
$FILLC    001252           594#    8223     8256
$FILLS    001251           593#    8256
$FLBUF    001334           621#    8606     8654
$FLD20    050432          8676#    8913
$FL20     050144          8601#    8912
$GDADR    001222           583#
$GDDAT    001226           585#    7691*    7749*    9074*    9103     9655
$GET42    036742          6914#
$HD     = 000000            32
$HINUM    001524           652#    1086*    6942     7020     7093     7159     8727     8735     8740*    8763     8767     9077
$ICNT     001206           576#    7835*    7836     7838*    7850
$ILLUP    051356          8843     8874#
$ITEMB    001216           580#    7881*    7900     7940     7942     7977
$LF       001330           619#    7900     8256
$LONUM    001522           651#    1087*    6943     7094     8726     8733     8739*    8768     9078
$LPADR    001210           577#    1175*    1463*    1580*    2744*    3797*    4342*    4670*    5090*    5563*    5881*    7826*    7840*
```

```
                                                      F02
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 207
CEQKCC.P11     03-MAR-78 13:13             CROSS REFERENCE TABLE -- USER SYMBOLS                                   SEQ 0225

$LPERR   001212            578#   1022*   1176*   1461*   1462*   1463    1578*   1579*   1580    2742*   2743*   2744    3795*
                          3796*   3797    4340*   4341*   4342    4668*   4669*   4670    5088*   5089*   5090    5561*   5562*
                          5563    5879*   5880*   5881    6546*   6692    6745*   7684    7685*   7696*   7742    7743*   7752*
                          7826    7841*   7850    7894    9266    9267*   9269*   9285    9339    9340*   9342*   9344    9353
                          9354*   9356*   9357    9368    9369*   9371*   9374
$MAINT   001604            677#   6855*   6865*   7960
$MXCNT   044772           7839    7850#
$NULL    001250            592#   8225    8256
$NWTST=  000001           1446#   1563#   1626#   1742#   1864#   1987#   2137#   2239#   2356#   2440#   2521#   2605#   2727#
                          2848#   2937#   3060#   3179#   3300#   3387#   3457#   3520#   3576#   3637#   3678#   3680    3741#
                          3743    3780#   3833#   3961#   4052#   4154#   4156    4222#   4277#   4325#   4445#   4491#   4539#
                          4605#   4653#   4824#   4826    4915#   4944#   4946    5008#   5073#   5188#   5190    5238#   5240
                          5322#   5324    5348#   5387#   5389    5442#   5444    5461#   5505#   5525#   5546#   5638#   5640
                          5678#   5680    5735#   5737    5796#   5798    5857#   5859    6044#   6046    6289#   6358#   6360
$OCNT    047462           8382*   8411*   8424#
$OCTVL   050032           8509    8537#
$OMODE   047464           8377*   8381*   8386    8389*   8400*   8426#
$OVER    044742           7807    7827    7837    7844#
$PASS    001200            572#   1131*   1407    6390    6890*   6891*   6902    6923    7833    7851    7972
$POWER   051364           8870    8877#
$PWRAD   051352           8872#
$PWRDN   051230           1169    8843#   8867
$PWRMG   051346           8870#
$PWRUP   051276           8852    8857#
$QUES    001326            617#   7900
$RAND    050604           6941    7019    7092    7158    8722#   8758    9076
$RDCHR=  ****** U         8910
$RDDEC=  ****** U         8910
$RDLIN=  ****** U         8910
$RDOCT=  ****** U         8910
$REGAD   001254            596#
$REG0    001256            598#   5892    5903    5925    5965    5972    5992    6025    6033    6034    6036    6061    6072
                          6094    6134    6141    6161    6194    6202    6203    6205    8807
$REG1    001260            599#   5893    5914    5924    5966    5974    5984    6026    6062    6083    6093    6135    6143
                          6153    6195
$REG2    001262            600#   5899*   5951    5981*   6012    6030*   6068*   6120    6150*   6181    6199*   9672    9685
$REG3    001264            601#   5939*   5953    6002*   6014    6034*   6108*   6122    6171*   6183    6203*   9672    9685
$REG4    001266            602#   9266*   9269    9339*   9342    9353*   9356    9368*   9371
$REG5    001270            603#   6302*   6861*   6871*   9172*   9180*   9190*   9222*   9223    9227*
$REG6    001272            604#
$REG7    001274            605#
$RESRE   050106           8572#   8911
$RTRN    001472            637#   1431    5004    6860
$SAVPA   053264           9304    9318    9329#
$SAVPS   053272           9295*   9326    9330#
$SAVRE   050050           8556#   8910
$SAVR6   051362           8851*   8857    8858*   8859*   8876#
$SCOPE   044534           1163    4174    4180    4274    5503    7805#
$SETUP=  000037           1138#   1163    1165    1167    1169    1171    1172    1173    1175    1274    6888    7886
$STUP =  177777           1138#
$SVLAD   044720           7815    7840#
$SVPC =  000220            555#    560
$SWR  =  167377              1#     32      47      48      49      50      51      52      53      54     614     615     616
                          1172    1173    1175    1176    1452    1569    1631    1747    1869    1992    2142    2244    2361
                          2445    2526    2610    2733    2853    2942    3065    3184    3305    3392    3462    3525    3581
```

```
                      4544     4610     4659     4831     4920     4970     5013     5079     5196     5247     5330     5353     5396
                      5448     5466     5510     5530     5552     5645     5684     5743     5802     5869     6056     6294     6364
                      6883     6889     6916     6922     6923     7798     7799     7800     7801     7802     7806     7818     7820
                      7821     7822     7829     7830     7831     7841     7844     7850     7858     7859     7860     7861     7862
                      7863     7872     7875     7882     7886     7892     7900
$SWRMK= 000000        7802
$TIMES  001316         614#    1172*    1450*    1567*    1747*    1869*    2731*    3784*    4329*    4657*    5077*    5550*    5869*
                      6056*    6889*    7829*    7836     7839*    7850
$TKB    001242         589#    9168
$TKS    001240         588#    1160     1179*
$TMP0   001276         606#    1020*    1291*    1294     1305*    1325*    1328     1332*    1345*    1348     1359*    1365*    1368
                      1379*    5419*    5890     5904     5963     5971     5991     6023     6031     6059     6073     6132     6140
                      6160     6192     6200     6531*    6559*    6563     6568*    6585     6632     6637     6718*    6795*    7658*
                      7683*    7687     7715*    7741*    7745     8756     8771     8803     9039*    9247*    9334*    9350*    9363*
                      9373     9542     9553     9568     9600     9613     9631     9645
$TMP1   001300         607#    1021*    6576*    6578     6581     6583*    6640     6642*    6740*    6743*    9645
$TMP2   001302         608#    5891     5912     5921     5964     5973     5983     6024     6557*    6739*    7684*    7696     7742*
                      7752     9035*    9265*    9337*    9367*    9542     9568     9600     9631
$TMP3   001304         609#    6737*    9036*    9252*    9338*    9568     9600     9631
$TMP4   001306         610#    5898*    5950     5980*    6011     6029*    6060     6081     6090     6133     6142     6152     6193
                      9037*    9253*    9672
$TMP5   001310         611#
$TMP6   001312         612#    5938*    6001*    6032*    9672
$TMP7   001314         613#
$TN   = 000076           1#      32     1446     1452#    1563     1569#    1626     1631#    1742     1747#    1864     1869#    1987
                      1992#    2137     2142#    2239     2244#    2356     2361#    2440     2445#    2521     2526#    2605     2610#
                      2727     2733#    2848     2853#    2937     2942#    3060     3065#    3179     3184#    3300     3305#    3387
                      3392#    3457     3462#    3520     3525#    3576     3581#    3637     3642#    3678     3684#    3741     3747#
                      3780     3786#    3833     3838#    3961     3966#    4052     4057#    4154     4161#    4176     4222     4227#
                      4277     4282#    4325     4331#    4445     4450#    4491     4496#    4539     4544#    4605     4610#    4653
                      4659#    4824     4831#    4915     4920#    4944     4970#    5008     5013#    5073     5079#    5188     5196#
                      5238     5247#    5322     5330#    5348     5353#    5387     5396#    5442     5448#    5461     5466#    5505
                      5510#    5525     5530#    5546     5552#    5638     5645#    5678     5684#    5717     5735     5743#    5796
                      5802#    5857     5869#    6041     6044     6056#    6289     6294#    6358     6364#
$TPB    001246         591#    8245*    8256     9213*
$TPFLG  001253         595#    8203     8256
$TPS    001244         590#    1203     6300     6303*    6315     6318*    6346*    6356*    6846     6862*    6869     6872*    8243
                      8256     9173*    9181*    9191*    9209*    9211     9226*
$TRAP   051374        1002     1167     4294     4297     4301     4306     4314     8889#
$TRP  = 000022        8896#    8906#    8907#    8908#    8909#    8910#    8911#    8912#    8913#    8914#
$TRPAD  051414        8893     8904#
$TSTNM  001202         573#    1449*    1566*    1629*    1745*    1867*    1990*    2140*    2242*    2359*    2443*    2524*    2608*
                      2730*    2851*    2940*    3063*    3182*    3303*    3390*    3460*    3523*    3579*    3640*    3682*    3745*
                      3783*    3836*    3964*    4055*    4159*    4225*    4280*    4328*    4448*    4494*    4542*    4608*    4656*
                      4829*    4918*    4968*    5011*    5076*    5194*    5245*    5328*    5351*    5394*    5446*    5464*    5508*
                      5528*    5549*    5643*    5682*    5741*    5800*    5867*    6054*    6292*    6294     6362*    6364     6412*
                      6414     6812*    6888*    7798     7846*    7847     7851     7868*    7871     7900     7964
$TYPBN= ****** U      8910
$TYPDS  047466        8440#    8909
$TYPE   046436        8203#    8896     8905
$TYPEC  046602        8222     8229     8236     8241#    8244
$TYPEX  046656        8242     8249     8251     8254#
$TYPOC  047264        8380#    8906
$TYPON  047300        8379     8382#    8908
$TYPOS  047240        8375#    8907
```

H02

```
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 209
CEQKCC.P11     03-MAR-78 13:13              CROSS REFERENCE TABLE -- USER SYMBOLS                                   SEQ 0227

$$GET4= 000000         6916#
$$TMP4  001416          629#
$$TMP6  001420          630#
$$STRP = 000002        8895#   8906    8907    8908    8909    8910    8911    8912    8913    8914
$OFILL  047463         8376#   8380*   8390    8425#
.     = 057544          523#    527     529#    532#    555     556#    558#    560#    569#    575#    620     621#    631#
                        632#    634#    678#    679#    680#    683#    684#    685#    800#   1081#   1161    1175    1176
                       1279#   1459    1465    1576    1582    1593    1601    1609    1616    1624    1639    1649    1659
                       1667    1676    1685    1694    1703    1712    1718    1725    1732    1740    1765    1775    1785
                       1792    1799    1808    1824    1870    1880    1887    1896    1903    1909    1916    1923    1931
                       1938    1946    1951    1959    1965    1973    1980    1985    1992    2005    2011    2017    2023
                       2029    2035    2042    2049    2056    2064    2071    2075    2082    2087    2094    2101    2107
                       2111    2116    2122    2126    2132    2135    2142    2154    2159    2166    2173    2179    2184
                       2188    2196    2203    2210    2214    2220    2224    2230    2234    2237    2244    2258    2262
                       2266    2274    2282    2289    2297    2305    2311    2317    2323    2328    2333    2339    2347
                       2351    2354    2361    2374    2381    2387    2394    2402    2408    2414    2419    2425    2432
                       2438    2445    2463    2466    2472    2477    2485    2491    2496    2503    2507    2513    2519
                       2535    2541    2546    2554    2562    2568    2576    2584    2590    2596    2600    2603    2619
                       2624    2630    2639    2646    2653    2660    2666    2673    2679    2685    2691    2698    2703
                       2709    2715    2718    2740    2746    2766    2773    2781    2788    2795    2803    2810    2816
                       2823    2827    2834    2839    2846    2869    2877    2883    2890    2896    2902    2908    2915
                       2921    2928    2935    2946    2954    2960    2969    2977    2997    3005    3015    3023    3031
                       3034    3044    3049    3058    3065    3080    3089    3097    3106    3113    3120    3126    3134
                       3145    3152    3157    3164    3172    3177    3184    3202    3206    3213    3224    3229    3235
                       3240    3246    3249    3298    3317    3324    3329    3336    3347    3357    3364    3370    3378
                       3383    3392    3425    3433    3437    3443    3450    3455    3486    3494    3499    3502    3508
                       3513    3518    3552    3555    3561    3568    3574    3581    3594    3599    3603    3610    3616
                       3623    3629    3633    3656    3660    3663    3668    3672    3702    3705    3706    3712    3719
                       3727    3731    3735    3739    3771    3793    3799    3805    3811    3816    3820    3826    3831
                       3872    3877    3883    3887    3897    3907    3912    3948    3950*   3970    3972    3977    3982
                       3990    3992    3997    4013    4017    4031    4039    4050    4137    4167    4183    4208    4215
                       4232    4262    4263    4270    4285    4293    4296    4300    4305    4338    4344    4352    4427
                       4453    4505    4520    4566    4573    4581    4587    4598    4618    4633    4642    4666    4672
                       4682    4695    4706    4709    4723    4737    4754    4757    4769    4778    4782    4789    4795
                       4800    4811    4817    4822    4837    4856    4873    4877    4879    4882    4907    4913    4932
                       4937    4941    4974    5032    5047    5058    5086    5092    5115    5138    5158    5182    5205
                       5217    5236    5267    5291    5315    5346    5365    5379    5469    5483    5511    5518    5535
                       5559    5565    5591    5618    5754    5762    5790    5877    5883    6608#   6800    6847    6901#
                       6923    6924#   7066    7211    7223    7261    7302    7329    7342    7382    7423    7850    7851
                       7900    8180#   8256    8494#   8537#   8854    8875    9212    9279#   9329#   9541#   9567#   9612#
                       9654#   9684#
```

```
                                                 I02
CEQKCC PDP 11/70 CPU EXERCISOR   MACY11 30A(1052)  03-MAR-78  13:15  PAGE 211
CEQKCC.P11     03-MAR-78 13:13            CROSS REFERENCE TABLE -- MACRO NAMES                                    SEQ 0228

COMMEN        1#      461#
ENDCOM        1#      461#
ERROR        79#      510     1023     4276     5959     6020     6042     6128     6189     6210     6534     6691     6748     6753     6797
           7689      7693     7747     7751     9040     9268     9341     9355     9370
ESCAPE        1#      461#
FLTSTI     5857#     5859     6046
HLT         510#     1594     1602     1610     1617     1625     1640     1650     1660     1668     1677     1686     1695     1704     1713
           1719      1726     1733     1741     1766     1776     1786     1793     1800     1809     1825     1859     1881     1888     1897
           1904      1910     1917     1924     1932     1939     1947     1952     1960     1966     1974     1981     1986     2006     2012
           2018      2024     2030     2036     2043     2050     2057     2065     2072     2076     2083     2088     2095     2102     2108
           2112      2117     2123     2127     2133     2136     2155     2160     2167     2174     2180     2185     2189     2197     2204
           2211      2215     2221     2225     2231     2235     2238     2259     2263     2267     2275     2283     2290     2298     2306
           2312      2318     2324     2329     2334     2340     2348     2352     2355     2375     2382     2388     2395     2403     2409
           2415      2420     2426     2433     2439     2464     2467     2473     2478     2486     2492     2497     2504     2508     2514
           2520      2536     2542     2547     2555     2563     2569     2577     2585     2591     2597     2601     2604     2620     2625
           2631      2640     2647     2654     2661     2667     2674     2680     2686     2692     2699     2704     2710     2716     2767
           2774      2782     2789     2796     2804     2811     2817     2824     2828     2835     2840     2847     2870     2878     2884
           2891      2897     2903     2909     2916     2922     2929     2936     2947     2955     2961     2970     2978     2993     2998
           3006      3016     3024     3032     3035     3045     3050     3059     3081     3090     3098     3107     3114     3121     3127
           3135      3146     3153     3158     3165     3173     3178     3203     3214     3225     3230     3236     3241     3247     3299
           3318      3325     3330     3337     3348     3358     3365     3371     3379     3384     3426     3434     3438     3444     3451
           3456      3487     3495     3500     3503     3509     3514     3519     3553     3556     3562     3569     3575     3595     3600
           3605      3611     3617     3624     3630     3634     3657     3661     3664     3669     3673     3703     3707     3713     3720
           3728      3732     3736     3740     3806     3812     3817     3821     3827     3832     3873     3878     3884     3888     3898
           3908      3923     3925     3927     3929     3931     3933     3935     3937     3939     3941     3943     3945     3947     3949
           3951      3978     3998     4014     4032     4040     4051     4078     4103     4114     4126     4134     4138     4152     4175
           4216      4227#    4239     4264     4271     4276#    4295     4298     4302     4307     4391     4428     4462     4463     4464
           4506      4521     4526     4553     4567     4574     4582     4588     4599     4619     4634     4643     4683     4696     4710
           4724      4738     4755     4770     4779     4790     4796     4801     4812     4818     4823     4838     4857     4874     4878
           4883      4889     4893     4899     4908     4914     4934     4938     4942     4991     4996     5037     5040     5051     5063
           5116      5139     5159     5183     5218     5237     5268     5292     5316     5347     5366     5380     5414     5421     5458
           5488      5500     5515     5536     5578     5607     5627     5673     5712     5763     5788     5841     6343
MSGA       3678#     3680
MSGB       3741#     3743
MSGC       4153#     4156
MSGD       4824#     4826
MSGE       4944#     4946
MSGF       5188#     5190
MSGG       5238#     5240
MSGH       5322#     5324
MSGI       5387#     5389
MSGJ       5442#     5444
MSGK       5637#     5640
MSGL       5677#     5680
MSGM       5735#     5737
MSGN       5795#     5798
MSGO       6357#     6360
MULT          1#      461#
MYTAGS      562#      620
NEWTST        1#        2#     461#    1446     1563     1626     1742     1864     1987     2137     2239     2356     2440     2521     2605
           2727      2848     2937     3060     3179     3300     3387     3457     3520     3576     3637     3678     3741     3780     3833
           3961      4052     4153     4222     4277     4325     4445     4491     4539     4605     4653     4824     4915     4944     5007
           5073      5188     5238     5322     5348     5387     5442     5461     5505     5525     5546     5637     5677     5735     5795
           5857      6044     6289     6357
POP           1#      461#    8481     8577     8741     8861
```

```
                                                         J02
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 212
CEQKCC.P11    03-MAR-78 13:13            CROSS REFERENCE TABLE -- MACRO NAMES                                SEQ 0229

RELOCA     982#     1452      1569      2733      3786      4331      4659      5079      5552      5870
RELOCB     982#     1556      2720      3773      4318      4646      5066      5538      5850      6282
SCOPE       80#     1451      1556      1568      1630      1746      1868      1991      2141      2243      2360      2444      2525      2609      2720
          2732      2852      2941      3064      3183      3304      3391      3461      3524      3580      3641      3683      3746      3773      3785
          3837      3965      4056      4160      4226      4281      4318      4330      4449      4495      4543      4609      4646      4658      4830
          4919      4969      5012      5066      5078      5195      5246      5329      5352      5395      5447      5465      5509      5529      5539
          5551      5644      5683      5742      5801      5850      5868      6055      6282
SETTRA    8896#     8906      8907      8908      8909      8910      8911      8912      8913
SETUP        1#      461#     1157
SKIP         1#      461#     1398      1406      1408      1410      4176      5716      6041
SLASH        1#      461#
SPACE      461#
STARS        1#      461#      535       562       895       986      1000      1280      1289      1309      1321      1323      1343      1363      1446
          1448      1563      1565      1626      1628      1742      1744      1864      1866      1987      1989      2137      2139      2239      2241
          2356      2358      2440      2442      2521      2523      2605      2607      2727      2729      2848      2850      2937      2939      3060
          3062      3179      3181      3300      3302      3387      3389      3457      3459      3520      3522      3576      3578      3637      3639
          3678      3681      3741      3744      3780      3782      3833      3835      3961      3963      4052      4054      4154      4158      4222
          4224      4277      4279      4325      4327      4445      4447      4491      4493      4539      4541      4605      4607      4653      4655
          4824      4828      4915      4917      4944      4967      5008      5010      5073      5075      5188      5193      5238      5244      5322
          5327      5348      5350      5387      5393      5442      5445      5461      5463      5505      5507      5525      5527      5546      5548
          5638      5642      5678      5681      5735      5740      5796      5799      5857      5866      5960      6022      6044      6053      6129
          6191      6213      6218      6225      6229      6236      6242      6289      6291      6358      6361      6371      6382      6401      6411
          6482      6495      6539      6545      6835      6840      6876      6925      6928      7002      7005      7075      7078      7142      7145
          7197      7200      7314      7317      7441      7444      7540      7543      7632      7635      7709      7712      7769      7775      7792
          7851      7900      7927      8012      8021      8033      8040      8049      8059      8069      8085      8093      8106      8121      8161
          8181      8256      8262      8316      8322      8349      8427      8495      8538      8584      8600      8657      8675      8710      8745
          8753      8780      8797      8838      8880      8914      8918      8940      8947      8969      8985      9010      9021      9044      9066
          9121      9125      9132      9134      9139      9141      9150      9163      9217      9221      9230      9245      9286      9291      9311
          9314      9331      9333      9347      9349      9360      9362      9688
TRMTRP    8896#
TYPBIN       1#      461#
TYPDEC       1#      461#     6902      6909
TYPNAM       1#      461#     1272
TYPNUM       1#      461#
TYPOCS       1#      461#
TYPOCT       1#      461#     7934      7960      7972
TYPTXT       1#      461#     1026      1030      1072      1412      6605      6614      6898      6905      8076      9276
$$CMRE     562#      598       599       600       601       602       603       604       605
$$CMTM     562#      606       607       608       609       610       611       612       613
$$ESCA       1#      461#
$$NEWT       1#        4#      461#     1446      1563      1626      1742      1864      1987      2137      2239      2356      2440      2521      2605
          2727      2848      2937      3060      3179      3300      3387      3457      3520      3576      3637      3678      3741      3780      3833
          3961      4052      4154      4222      4277      4325      4445      4491      4539      4605      4653      4824      4915      4944      5008
          5073      5188      5238      5322      5348      5387      5442      5461      5505      5525      5546      5638      5678      5735      5796
          5857      6044      6289      6358
$$SET     8896#     8906      8907      8908      8909      8910      8911      8912      8913
$$SKIP       1#      461#     4176      5717      6041
.EQUAT       1#
.HEADE       1#       22
.KT11        1#
.SETUP       1#     1137
.SWRHI       1#       42
.SWRLO       1#       55#       58        61
.$ACT1       1#      535
.$CATC       1#      520
```

```
CEQKCC PDP 11/70 CPU EXERCISOR  MACY11 30A(1052)  03-MAR-78  13:15  PAGE 213                                      SEQ 0230
CEQKCC.P11     03-MAR-78 13:13          CROSS REFERENCE TABLE -- MACRO NAMES

.$DB2D        1#
.$DB2O        1#        6#    8495
.$DIV         1#
.$EOP         1#    6876
.$ERRO        1#    7851
.$ERRT        1#
.$MULT        1#
.$POWE        1#    8838
.$RAND        1#        7#    8710
.$RDDE        1#
.$RDOC        1#
.$READ        1#
.$SAVE        1#    8538
.$SB2D        1#
.$SB2O        1#
.$SCOP        1#        2#    7792
.$SIZE        1#
.$SUPR        1#
.$TRAP        1#    8880
.$TYPB        1#
.$TYPD        1#    8427
.$TYPE        1#      562#    8181
.$TYPO        1#    8349
.1170         1#      71

. ABS.  057544      000

 ERRORS DETECTED:  0

 CEQKCC.BIN,CEQKCC.LST/CRF/SOL/NL:TOC=DSKZ:CEQKCC.SML,CEQKCC.P11
 RUN-TIME: 32 38 3 SECONDS
 RUN-TIME RATIO: 1039/75=13.7
 CORE USED:  32K  (63 PAGES)
```

L02

M02