IDENTIFICATION

Product Code: MAINDEC-11-DCQKC-D-D

Product Name: 11/40 and 11/45 INSTRUCTION EXERCISER

Date Created: SEPTEMBER 21, 1974

Maintainer: Diagnostic Group

Author: John Adams

Table of Contents

ABSTRACT

This Diagnostic Program is designed to be a comprehensive check of the PDP-11/40 and PDP-11/45 processors. The program executes each instruction in all address modes and includes tests for traps and the Teletype interrupt sequence. The program relocates the test code throughout memory 0-124K. If selected, the program may be relocated by any of the available disks.

CHAPTER 1

REQUIREMENTS

1.1 EQUIPMENT

PDP-11 Family Central Processor with 8K memory.

1.1.2 Optional Equipment Used

1. KW11-P (Programmable Clock)
2. KW11-L (Line Frequency Clock)
3. All parity memory options
4. KT11-C,D (11/40, 11/45 Memory Management)
5. RK11, RF11, RP11, RS03/4, RC11
6. KJ-11 (11/40 Stack Limit)
7. EIS (11/40 Extended Instruction Set)

1.2 STORAGE

The program loads into the first 6K of memory, and runs in all memory (exclusive of loaders).

1.3 PRELIMINARY PROGRAMS

None.

CHAPTER 2

LOADING AND STARTING PROCEDURE

Load the program using the Absolute Loader. If console TTY is a serial device (LA30S, VT05, etc.), filler characters are required. Deposit into location 1002 (FILLS) a 0 (the filler character) and location 1003 11(octal) (the filler count).

Load Address = 200
Press start.
Set operating switches

Contents of OPT.CP is typed on first pass (see Chapt 7)
(initial load)
Pass count is printed after each pass (see Section 5.5)
"DCQKC DONE" is printed when done (see Section 6.1).

If no console TTY is available, set SW15=1 (HALT ON ERROR).

2.1 ACT11 OPERATION

If the program is run in quick verify mode, no subtest iterations are performed.

CHAPTER 3

SWITCH SETTINGS

SW15	HALT ON ERROR	This switch when set will halt the processor when an error is detected. The PC+2 and the current status at the time of the error is stored on the stack (R6). If this switch is set before an error is detected, the program halts as described above. The program may be halted after the error typeout occurs by setting SW15 after the typeout begins.
SW14	LOOP SUBTEST	This switch when set loops the current subtest running regardless of error.
SW13	INHIBIT ERROR PRINTOUT	This switch when set inhibits the error printout.
SW12	INHIBIT RELOCATION	This switch when set causes the program to be executed only in the first 8K of memory. This switch cannot be set when the program is running.
SW11	INHIBIT SUB-TEST ITERATION	This switch when set inhibits subtest reiteration. Normally each subtest is executed 8 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	INHIBIT RELOCATION	This switch when set inhibits relocation of the program above 28K.
SW8 SW7-0	LOAD PDP-11/45 MICRO BREAK-	This switch when set loads the micro break register with the value set into

	REGISTER	SW7-Ø at the beginning of each subtest.
SW7	INHIBIT END OF PASS TYPEOUT	This switch when reset inhibits the end of pass typeout (The Quick Brown Fox,..).
SW6	INHIBIT CLOCK INTERRUPTS	This switch when set will turn the clock(s) off.
SWØ5	ENABLE RELOCATION VIA ALL AVAIL. DISKS	This switch will cause program relocation via all available disks Round Robin style, i.e., first relocation via CP, then RK, RF, RP, etc.
SWØ4	ENABLE RANDOM DISK ADDRESS SELECTION FOR RELOCATION	If not enabled all disk relocation transfers begin at disk address Ø.
SWØ3	ENABLE RELOCATION VIA I/O DEVICE	
SWØ2-SWØØ	DEVICE CODES	These switches when set cause the program to relocate the test code using the device specified below:

Value	Device
Ø	CP
1	RK
2	RF
3	RP
4	RC
5	DO NOT USE
6	RSØ4
7	CP

NOTE

When relocating via an I/O device, set in the value to select the device then set switch 3.

3.1 11/45 DISPLAY REGISTER

The pass count is displayed in bits ØØ-Ø2. The section number is displayed in bits Ø6-Ø3. The most significant byte of the base address (contents of FRSTAD) of the section of code being executed is displayed in bits 15-Ø8. When memory management is enabled the contents of KIPAR2 is displayed. KIPAR2 contains the base page

address of the code being executed.

NOTE

The RF11 Data Buffer Register also displays the above information if the RF is selected.

CHAPTER 4

ERRORS

If an error is detected, the program will trap to the Error Handling Routine (ERROR). If error typeout is enabled, this routine will type the PC and the processor status at the time of the error. Also, (if required), the original PC (where the PC was relocated from).

4.1.1 Error Printout Format (CP Error)

```
PASS # AAAA VPC=BBBBBB PSW=DDDDDD

                    or

PASS # AAAA VPC=BBBBBB PSW=DDDDDD RPC=CCCCCC

                    or

PASS # AAAA VPC=BBBBBB PSW=DDDDDD PPC=EEEEEE
```

where: VPC=Virtual PC
RPC=PC of original code
PPC=Physical PC
AAAA=PASS COUNT
BBBBBB=Virtual PC at the time of the error
CCCCCC=PC of the original code relocated
DDDDDD=PSW at the time of the error
EEEEEE=Physical PC at the time of the error.

The first error format shows an error detected when the program is not relocated, and, in this instance VPC=PPC. The error is probably a CP error.

The second error format show an error detected when the program is relocated below 28K, and, in this instance VPC=PPC. The error is probably due to a memory error.

The third error format shows an error detected when the program is relocated above 28K. The error is probably due to a memory error. Note that VPC is the PC of the original code.

To obtain the 'Physical' PC (11/45 only), set the address selector to the KLI position. Load address and examine the PC address, set the address selector to 'Program Physical'. The address displayed is the Physical PC. On the 11/40 to obtain the 'Physical' PC add the contents of KIPAR2 or KIPAR3 to the virtual PC.

NOTE

Use caution when examining/depositing into addresses when memory management is enabled.

4.1.2 Error Printout Format (Device Error)

PASS # AAAA VPC=BBBBBB XX ERROR

111111 222222 333333 444444 555555 666666

where: VPC=Virtual PC
AAAA=Pass count
BBBBBB=Virtual PC at time of error
XX=Two letter device identifier
111111-666666=Contents of device register

4.1.3 Error Printout Format (Parity Error)

Parity Error

The PC at the time of the error is typed as shown in Section 4.1.1.

Memory Address = XXXXXX, Good Data = XXXXXX, Bad Data = XXXXXX.

NOTE

The address typed is the 18 bit physical address.

4.1.4 Error Printout Format (Relocation Error)

PASS # AAAA VPC=BBBBBB MM ERROR

FROM ADRS=XXXXXX DATA=XXXXXX TO ADRS=XXXXXX DATA=XXXXXX

NOTE

The addresses are 18 bit physical addresses "from" address is in R0 "to" address is in R2.

4.2 PARITY ERROR DETECTION

If a parity error is detected the program will type a message "PARITY ERROR". Print the PC at the time of the error (via HLT) and scan memory for the parity error. When the program finds the parity error it will type a message "MEMORY ADDRESS IS 888888". When the address is found the failing address is scanned with a binary count pattern. When the program finds the failing data the good data and bad data are typed. If the program does not find the parity error on the address/data scan it will type a message "PARITY ERROR NOT DETECTED ON ADDRESS/DATA SCAN". The program is then restarted.

4.3 ERROR LOOPING

The subtest detecting the error may be looped indefinitely by setting SW14. Setting SW13 will inhibit the typeout and allow scoping the faulty signal(s).

4.4 UNPREDICTED ERRORS

The program may on occassion detect a memory error the results of which were not predictable in which case the program may behave unpredictably. When this happens the user must retrace the program steps to resolve where the error occurred. The following items should be considered and may be of use when retracing a failure of this nature.

1. Halt the program (if necessary).
2. Examine RELR1
   Address RELR1 (1006) contains the unrelocated value of the PC of the last test that was successfully executed.
3. Examine FACTOR
   Address FACTOR (1004) contains the relocation factor.
4. Examine all locations starting with the address specified in R1/R11 (if PSW BIT11 = 0/1) comparing their contents with the contents of the corresponding unrelocated code (specified in

1006) as shown in the listing. Examine and compare until either a difference in instruction (i.e., the error) or the next 'scope' is seen.

1A. Examine the stack (R6)
The top word on the stack contains the PC at the time of the trap. If the PC is greater than the last location in the listing then =

2A. Examine location 1004 (FACTOR)
This location contains the program relocation factor which, when subtracted from the PC gives the PC of the original code.

4.5 TRAP TO LOCATION 4

If a trap to location 4 occurs the program will type: "TRAP TO 4". Then the error printout information (as in 4.1.1) will be typed.

NOTE

The PC typed will be the PC-2 at the time that the trap occured. The program will then restart at the beginning (START).

4.6 TRAP TO LOCATION 10

If a trap to location 10 (reserved instruction) occurs the program will type: "RESERVED INSTRUCTION TRAP" and the additionalslk linformation instruction (as in 4.1.1) the PC typed will be the PC-2 at the time of the trap. The program will restart at the beginning (START).

4.7 MEMORY MANAGEMENT (KT11) ABORT

If a KT11 abort (trap at 250) occurs, the program will type a message "KT11 ABORT". Then the error printout information (as in 4.1.1) will be typed.

NOTE

The PC typed will be the contents of SR2 at the time that the trap occurred. The program will then restart a the beginning (START).

4.8 ERROR DISCUSSION

An error detected when the program is not relocated is likely to be a CP malfunction. An error detected when the program is relocated between 40000 and 160000 could be either a CP or memory malfunction. An error detected when the program is relocated above 160000 (28K) is most likely a memory malfunction. The memory exerciser (DZQMB-) should be run if a memory failure is suspected, selecting only those bank(s) deemed bad.

CHAPTER 5

SUBROUTINE ABSTRACTS

5.1 SCOPEA

The SCOPEA Routine is entered by the scope (EMT) instruction and is executed at the start of each subtest. The routine monitors SW14, SW11 and SW8 and takes appropriate action. Also, this routine stores in R1/R11 the first address of the subtest being entered.

5.2 ERROR

The Error Routine is entered by the HLT (trap) instruction, and is executed when a predictable error is detected. This routine monitors SW15, SW13, and SW10.

5.3 PROGRAM RELOCATION

Four routines are used to perform program relocation. The general flow is as follows:

If below 28K	The RELOC Routine is called after a section of code has been executed. If an I/O device is selected, subroutine IODEV is called and the routine WAITIO is executed while the device is transferring code.
If above 28K	The STMM Routine is called after the entire program has been executed. If an I/O device is selected, subroutine IODEV is called and the routine WAITIO is executed while the device is transferring code.

5.3.1 RELOC

The RELOC Routine is entered by a MOV #RELOC,PC instruction. This routine relocates the program code throughout memory and 'jumps' to the relocated code after it has been moved successfully. The code is relocated by 'moving' the code via MOV instructions. If an I/O device is selected via switch register <3-0>, the code is relocated by writing the code onto the I/O device and reading the code back into its relocated position. If the code cannot be relocated (because of insufficient memory) the routine 'jumps' to the next section of unrelocated program code. The code moved is less than 1K ((4000) bytes). At the start and end of each section of code to be moved is a section of code which establishes the first address of the code to be moved, and sets a scope pointer (R1/R11) and also a section which establishes the last address and 'jumps' to the relocation (RELOC) routine. Each section of code is identified as shown below.

```
↑000000000FIRST ADDRESS TO BE RELOCATED000000000

        CODE TO BE MOVED AND EXECUTED

↓000000000LAST ADDRESS OF CODE TO BE RELOCATED000000000
```

THE RELOC Routine does not relocate program code into the last 1000(octal) bytes of memory, thus preserving the loaders. This routine monitors SW12, SW05, and SW03.

5.3.2 Relocation above 28K (STMM)

The STMM Subroutine relocates the program code above 28K if memory and the KT option are available. The routine moves the code at 0-8K upwards to addresses above 28K. Each succeeding relocation is to memory 1K greater than the last. The program is executed in all cases from virtual memory addresses 0-37776, however, the physical address changes by 1K (4000) on each relocation.

NOTE

The 'Virtual' light (11/40) will be on when the program is executing above 28K.

This routine monitors SW12, SW09, SW05, and SW03.

5.3.3 IODEV

The IODEV Subroutine is called from either the RELOC or STMM routines whenever an I/O device is selected to perform program relocation. This routine obtains the physical BUS address for read and write and

the byte count from the calling routine. The device to be used is obtained from location DEV. The code to be relocated is written from its present position and then read into the relocated position. If a device error occurs the error is reported and the operation is retried up to three times.

5.3.4 WAITIO

The purpose of the WAITIO Routine is to reference via the CP the same memory locations as the device during the NPR transfers.

5.3.5 DSKADR

The DSKADR Subroutine is called from the IODEV routine. It generates a random disk address for the selected disk if SW04 is set. Otherwise it generates a '0' disk address. The generated random addresses are limited (so disk overflow will not occur) by the table ADRTAB.

5.4 CLOCK INTERRUPT

The Clock Interrupt for the line and programmable clocks increment locations LTICKS and PTICKS on each interrupt. This routine monitors SW06.

5.5 END

This routine is entered at the completion of each pass. It sets up (loads new processor status) for the next pass and prints an end of pass message:

THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK
0123456789 PASS # AAAA

CHAPTER 6

MISCELLANEOUS

6.1 EXECUTION TIME

The execution time is highly variable (dependent on processor, type of memory, and amount of memory), however, when the program is running successfully there is a noticeable 'Flicker' displayed on the console light pattern. The 'Flicker' will dim when 'T' bit trap passes (every odd pass) are running, the program should be run for a minimum of:

4 PASSES (PASS # 0003) 11/40
8 PASSES (PASS # 0007) 11/45

some typical times follow:

PDP-11/45 WITH 104K MEMORY (96K CORE, 8K MOS)-24 MINS
PDP-11/45 WITH 48K MEMORY-10 MINS

6.2 PASS MODIFICATION

The PSW of the pass may be modified by patching into location PSWTAB+2 the desired PSW. For example patching 040000 into PSWTAB+2 causes the program to run in supervisor mode on the second pass.

6.3 I/O DEVICE ADDRESS MODIFICATION

To modify the program address of the I/O devices on the unibus patch the appropriate device table (see listing table of contents - Device Tables) and also the appropriate table entry at 'REGADR' in the error service routine.

6.4 MEMORY MODIFICATION

The program may be modified to provide extended memory exercising. Essentially the modification increases the test iteration count which causes test code to be executed in memory for a longer period of time. Note that this modification will increase the run time substantially. The modification is:

PATCH	LOCATION	FROM	TO
	5454	020040	100200

6.5 USER DEFINED RELOCATION LIMITS

The program will request a lower and upper limit for relocation. The limits must be between the last location in the listing and 157776. The program will execute in the lower 4K (0-17776) and the limits specified. The starting address is 204. To retain previously specified limits, start at 210.

CHAPTER 7

PROGRAM DESCRIPTION

The program is divided into four sections of position independent relocatable test code. Each section is approximately 1K words long, (except section 3 and A).

Section 0 — This section causes a 256 word 3X or 9 worst case noise test pattern to be relocated throughout memory 0 - 28K.

NOTE

This should not be constructed to be a 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 bytes 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) instructions.

Section A — Following Section 3 is a routine to ascertain which OP the program is running on. The results are used by the following sections to check the additional instructions/features of the 11/40 and 11/45. This routine leaves the results in location 'OPT,CP'. The contents of this location are typed out as follows:

where:

BIT15 = 1/0 = Memory management option available/not available

BIT14 = 1/0 = EIS available/not available

NOTE

EIS is always available on PDP-11/45.

BIT13 = 1/0 = 11/45 FPP available/not available

BIT12 = 1/0 = 11/40 FIS available/not available

BIT11 = 1/0 = Stack limit (11/40 KT option) availale/not available

BIT10 = 1/0 = KW11-P available/not available

BIT09 = 1/0 = KW11-L available/not available

BIT08 = 1/0 = Console TTY available/not available

BITS 07-00 = 06 = 11/45, 04 = 11/40.

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

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

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

Section 7 This section checks the stack limit register (KJ-11 option on 11/40), and memory management abort logic (if system has more than 32K of memory).

Following Section 7 are two routines to check the Teletype printer logic and a routine to start either the KW11-P or the KW11-L clock. If either the KW11-P or the KW11-L is available the priority arbitration logic is tested.

The program then relocates to 160000 (if available) and restarts. The program continues relocating by increments of 4000 bytes (1K) until the end of memory is reached. Relocation of the program throughout

all memory constitutes a pass. When the program is executing above 28K, you will hear several 'kerchunks' on the teletype. The 'kerchunks' are caused by the Teletype test following Section 7 mentioned above.

Upon completion of a pass the program restarts using a new processor status depending on the type of processor and the pass count.

7.1 STACK POINTER

The stack Pointer is set at 500.

NOTE

If the program is running in either user or supervisor mode, the user/supervisor stack pointer is set to 500 and the kernel stack pointer is set to 600. The kernel stack pointer is used only for the SCOPE HIT, TTY, and Clock Trap/Interrupt routines.

7.2 POWER FAILURE

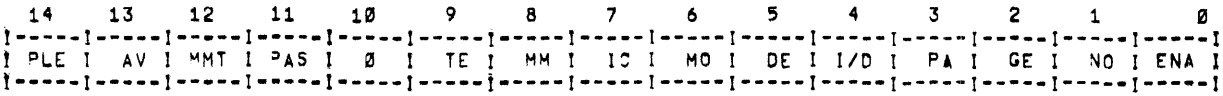
A Power Fail service routine is incorporated in the test. When using this program the power should be turned off when running to check the power fail logic. When the power fails the program will type:

POWER FAILED

and restart the program at the beginning (START).

APPENDIX A
KT11 C/D REGISTERS

SR0=777572 = S T A T U S R E G I S T E R #0

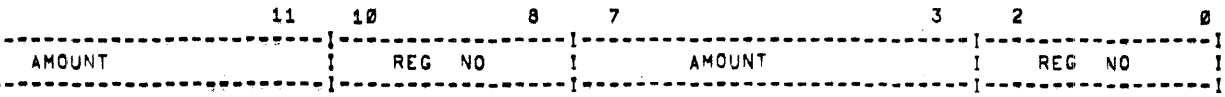


NRA - NON-RESIDENT ABORT
PLE - PAGE LENGTH ERROR
AV - ABORT - READ ONLY ACCESS VIOLATION
* MMT - MEMORY MANAGEMENT TRAP
* PAS - PROGRAMS AID SYSTEM FLAG
* TE - ENABLE MEMORY MANAGEMENT TRAPS
MM - MAINTENANCE MODE

* IC - INSTRUCTION COMPLETED
MODE- CPU MODE
ØØ - KERNAL 1Ø - NOT USED
Ø1 - SUPERVISOR 11 - USER
* I/D - SEGMENT ADDRESS SPACE
Ø - I SPACE 1 -D SPACE
PAGE- PAGE #
ENA - ENABLE MEMORY MANAGEMENT

* SR1=777574 = S T A T U S R E G I S T E R #1

SR1 RECORDS ANY AUTOINCRMENT/DECREMENT OF THE GPR'S, INCLUDING EXPLICIT REFERENCES THROUGH THE PC, SR1 IS CLEARED AT THE BEGINNING OF EACH INSTRUCTION FETCH, WHENEVER A GPR IS EITHER AUTO INCREMENTED/ DECREMENTED THE REGISTER NUMBER AND THE AMOUNT (IN 2'S COMPLEMENT NOTATION) IS RECORDED IN SR1.

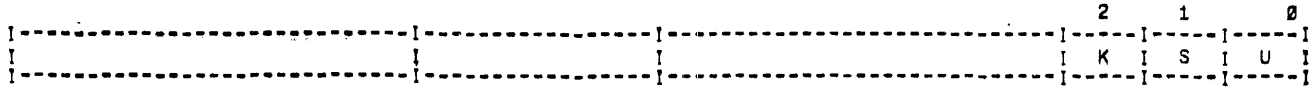


SR2=777576 = S T A T U S R E G I S T E R #2

SR2 IS LOADED WITH THE 16 BIT VIRTUAL ADDRESS AT THE BEGINNING OF EACH INSTRUCTION FETCH, OR WITH THE ADDRESS TRAP VECTOR AT THE BEGINNING OF AN INTERRUPT, 'T' BIT TRAP, PARITY, ODD ADDRESS, AND TIMEOUT TRAPS.

* SR3=772516 = S T A T U S R E G I S T E R #3

SR3 ENABLES OR DISABLES THE USE OF I/D SPACE. WHEN 'D' SPACE IS
DISABLED ALL REFERENCES USE THE 'I' SPACE REGISTERS; WHEN 'D' SPACE
IS ENABLED BOTH 'I' AND 'D' REGISTERS ARE USED. A '1' ENABLES 'D' SPACE.

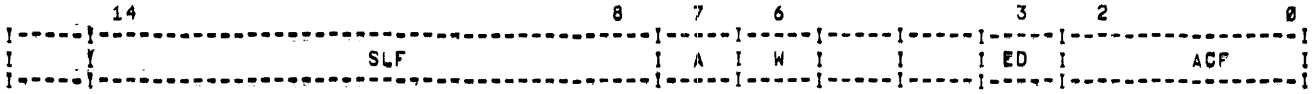


U = USER D SPACE ENABLE
S = SUPERVISOR 'D' SPACE ENABLE
K = KERNEL 'D' SPACE ENABLE

THE PROGRAM DOES NOT ENABLE ANY 'D' SPACES

P A G E D E S C R I P T O R R E G I S T E R S

KIPDR0-KIPDR7	772300-772316	*KDPDR0-KDPDR7	772320-772336
UIPDR0-UIPDR7	777600-777616	*UPDPR0-UPDPR7	777620-777636
* SIPDR0-SIPDR7	772200-772216	*SDPDR0-SDPDR7	772220-777236



ACF - ACCESS CONTROL FIELD
- 000 NON-RESIDENT
- 100 READ/WRITE
- * 101 READ/WRITE
- 110 READ/WRITE
- * 001 READ ONLY
- 010 READ ONLY
- * 011 UNUSED
- * 111 UNUSED

ED - EXPANSION DIRECTION (1/0 = DOWN/UP)

W - SEGMENT HAS BEEN WRITTEN INTO

* A - SEGMENT HAS BEEN ACCESSED

SLF - SEGMENT LENGTH FIELD
BINARY REPRESNETATION OF NUMBER OF 32 WORD
BLOCKS IN SEGMENT.

* 11/45 ONLY

APPENDIX B
RC11 REGISTERS

INTERRUPT VECTOR = 210

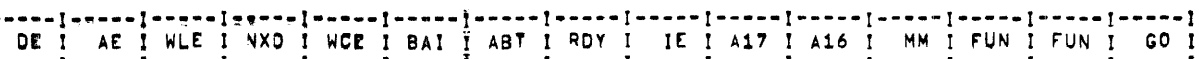
RCLA=777440 - L O O K A H E A D R E G I S T E R

RCDA=777442 - D I S K A D D R E S S R E G I S T E R

RCER=777444 - D I S K E R R O R R E G I S T E R

DLE - DATA LATE ERROR
BCE - BLOCK CHECK ERROR
DSE - DATA SYNC ERROR
NXM - NON-EXIST MEMORY
ATE - A TRACK ERROR
APE - ADDRESS PARITY ERROR
SAE - SYNC ADDRESS ERROR
DOE - DISK OVERFLOW ERROR
MXF - MISSED TRANSFER

RCCS=777446 - D I S K C O N T R O L & S T A T U S R E G I S T E R



SC - SPECIAL CONDITION
DE - DATA ERROR
AE - ADDRESS ERROR
WLE - WRITE LOCK ERROR
NXD - NON-EXISTENT DISK
WCE - WRITE CHECK ERROR
BAI - INHIBIT CA INCREMENT
ABT - ABORT
RDY - READY
IE - INTERRUPT ENABLE
A17 - BUS ADDRESS BIT 17
A16 - BUS ADDRESS BIT 16
MM - MAINTENANCE MODE
FUN - FUNCTION
00 LOOK AHEAD 10 READ
01 WRITE 11 WRITE CHECK
GO - GO

RCWC=777450 - W O R D C O U N T R E G I S T E R

RCCA=777452 - C U R R E N T A D D R E S S R E G I S T E R

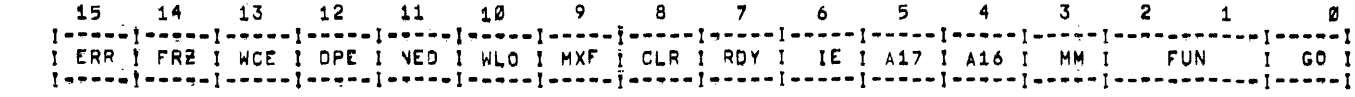
RCMN=777459 - M A I N T E N A N C E R E G I S T E R

RCDB=777456 - D A T A B U F F E R R E G I S T E R

APPENDIX C
RF11 REGISTERS

INTERRUPT VECTOR = 204

RFDCS=777460 = D I S K C O N T R O L S T A T U S R E G I S T E R

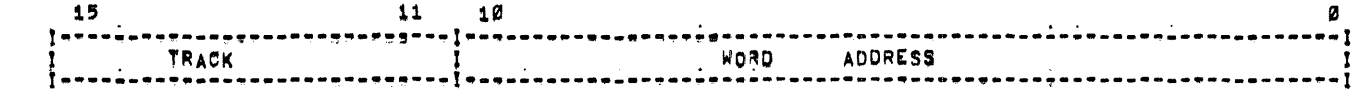


ERR	ERR (LOGICAL OR OF 14-9)	IE	INTERRUPT ENABLE
FRZ	FREEZE (LOGICAL OR OF DAE 15-10)	A17	BUS ADDRESS BIT 17
WCE	WRITE CHECK ERROR	A16	BUS ADDRESS BIT 16
DPE	DATA PARITY ERROR	MM	MAINTENANCE MODE
NED	NON EXISTANT DISK	FUN	FUNCTION
WLO	WRITE LOCK OUT		00 NO OP 10 READ
MXF	MISSED TRANSFER		01 WRITE 11 WRITE CHECK
CLR	DISK CLEAR		
RDY	CONTROL READY		

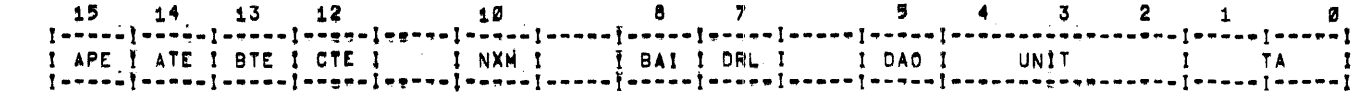
RFWC=777462 = W O R D C O U N T R E G I S T E R

RFMA=777464 = C U R R E N T M E M O R Y A D D R E S S

RFDAR=777466 = D I S K A D D R E S S R E G I S T E R



RFDAE=777470 = D I S K A D D R E S S E X T E N S I O N E R R O R R E G I S T E R



APE	ADDRESS PARITY ERROR	DRL	DATA REQUEST LATE
ATE	A TIMING TRACK ERROR	DAO	DISK ADDRESS OVERFLOW
BTE	B TIMING TRACK ERROR	UNIT	UNIT NUMBER
CTE	C TIMING TRACK ERROR	TA	EXTENSION OF TRACK ADDRESS
NXM	NON-EXISTANT MEMORY		IN RFDAR
BAI	BUS ADDRESS INHIBIT		

RFDBR=777422 = D A T A B U F F E R R E G I S T E R

RFMA=777474 = M A I N T E N A N C E R E G I S T E R

RFADS=777476 = L O O K A H E A D

APPENDIX D
RK11 REGISTERS

INTERRUPT VECTOR = 220

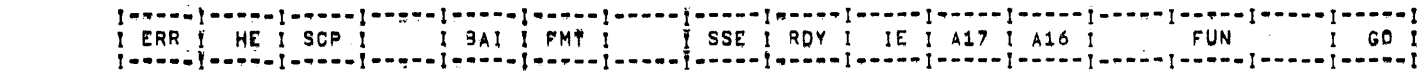
RKDS=777400 - D R I V E S T A T U S R E G I S T E R

ID	IDENT OF DRIVE	DRY	DRIVE READY
DPL	DRIVE POWER LOW	RDY	READ/WRITE/SEEK READY
RKS	SET TO INDICATE RK05	WPS	WRITE PROTECT STATUS
DRV	DRIVE UNSAFE	SC=SA	SECTOR COUNTER=SECTOR ADDRESS
SIN	SEEK INCOMPLETE	SC	SECTOR
SOK	SECTOR COUNTER OK		

RKER=777402 - E R R O R R E G I S T E R

DRE	DRIVE ERROR	TE	TIMING ERROR
OVR	DISK OVERRUN (OVERFLOW)	NXD	NON-EXISTANT DISK
WLO	WRITE LOCK	NXC	NON-EXISTENT CYLINDER
SKE	SEEK ERROR	NXS	NON-EXISANT SECTOR
PGE	PROGRAMMING ERROR	CSE	CHECK SUM ERROR
NXM	NON-EXISTANT	WCE	WRITE CHECK ERROR
DLT	DATA LATE		

RKCS=777404 - C O N T R O L S T A T U S R E G I S T E R

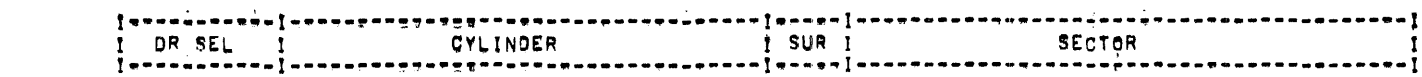


ERR	ERROR(OR OF RKER & RKCS<14>)	A17	BUS ADDRESS BIT 17
HE	HARD ERROR	A16	BUS ADDRESS BIT 16
SCP	SEARCH COMPLETE	FUN	FUNCTION
BAI	BUS ADDRESS INHIBIT INC.		000 RESET 100 SEEK
FMT	FORMAT MODE		001 WRITE 101 RD CHK
SSE	STOP ON SOFT ERROR		010 READ 110 DRV RESET
RDY	CONTROL READY		011 WT CHK 111 WT LOCK
IE	INTERRUPT ENABLE	GO	ENABLE FUNCTION

RKWC=777406 - W O R D C O U N T R E G I S T E R

RKBA=777412 - C U R R E N T B U S A D D R E S S

RKDA=777412 - D I S K A D D R E S S



RKDB=777416 - D A T A B U F F E R R E G I S T E R

APPENDIX E
RP11C REGISTERS

INTERRUPT VECTOR = 254

RPDS=776710 = D R I V E S T A T U S R E G I S T E R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SUR	ONL	RP03	HNF	SI	SU	UNSF	WP	ATN7	ATTENTION						ATN0

SUR	SELECTED UNIT READY	SU	SEEK UNDERWAY
ONL	SELECTED UNIT ON LINE	UNSF	SELECTED UNIT UNSAFE
RP03	SELECTED UNIT IS RP03	WLO	SELECTED UNIT WRITE LOCKED
HNF	HEADER NOT FOUND	ATN7-ATN0	DRIVE ATTENTION
SI	SEEK INCOMPLETE		

RPER=776712 = E R R O R R E G I S T E R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WPV	FUV	NXC	NXT	NXS	PGE	FMT	MOD	LPE	WPE	CSE	TE	WCE	NXM	EOP	DER

WPV	WRITE PROT VIOL	LPE	LONG PARITY ERROR
FUV	FILE UNSAFE VIOL	WPE	WORD PARITY ERROR
NXC	NON-EXISTANT CYLINDER	CSE	CHECK SUM ERROR
NXT	NON-EXISTANT TRACK	TE	TIMING ERROR
NXC	NON-EXISTANT SECTOR	WCE	WRITE CHECK ERROR
PGE	PROGRAM ERROR	NXM	NON-EXISTANT MEMORY
FMT	FORMAT ERROR	EOP	END OF PACK
MOD	MODE ERROR	DER	DISK ERROR

RPCS=776714 = C O N T R O L S T A T U S R E G I S T E R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR	HE	AIE	MOD	HDR	DRIVE SELECT			RDY	IE	A17	A16	FUNCTION			GO

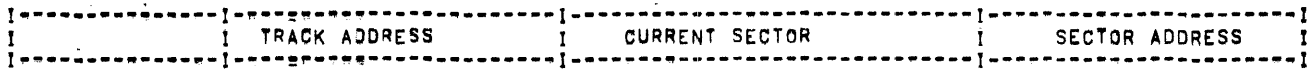
ERR	ERROR (OR OF ALL ERROR BITS)	A17	BUS ADDRESS BIT 17				
HE	HARD ERROR (OR OF ALL BUT DATA ERROR BITS)	A16	BUS ADDRESS BIT 16				
		FUN	FUNCTION				
AIE	ATTENTION INTERRUPT ENABLE		000	INIT	100	SEEK	
MODE	RP11-C IS CONDITIONED TO RD /WRT DISK PACKS IN PDP10 OR PDP 15 FORMAT		001	WRITE	101	WRITE (NO SEEK)	
			010	READ	110	HOME SEEK	
			011	WRT CK	111	RD (NO SEEK)	
HDR	FUNCTION IS A HEADER OPER,	GO	ENABLE OPERATION				
DRIVE SELECT							
RDY	READY						
IE	INTERRUPT ENABLE						

RPWC=776716 = W O R D C O U N T R E G I S T E R

RPBA=776720 = B U S A D D R E S S R E G I S T E R

RPCA=776722 = C Y L I N D E R A D D R E S S R E G I S T E R
<BITS 00-08>

RPDA=776724 - D I S K A D D R E S S R E G I S T E R

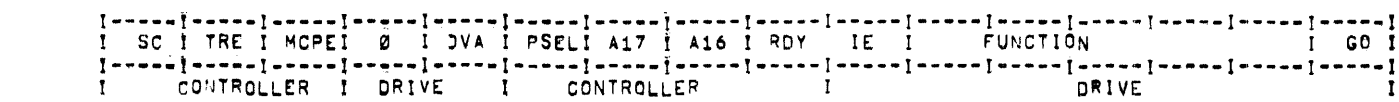


SUCA=776734 - S E L E C T E D U N I T C Y L I N D E R A D R S
<BITS 00-08>

APPENDIX F
RS03/4 REGISTERS

INTERRUPT VECTOR = 204

RSCS1=772040 - C O N T R O L S T A T U S R E G #1



SC	SPECIAL CONDITION	FUN	FUNCTION
TRE	TRANSFER ERROR	00000	NO OPERATION
MCPE	MASS BUS CONTROL BUS PARITY ERROR	00100	DRIVE CLEAR
DVA	DRIVE AVAILABLE	01100	SEARCH
PSEL	PORT SELECT	10100	WRITE CHECK
A17	BUS ADDRESS BIT 17	11000	WRITE
A16	BUS ADDRESS BIT 16	11100	READ
RDY	READY	GO	ENABLE FUNCTION
IE	INTERRUPT ENABLE		

RSWC=771042 - W O R D C O U N T R E G I S T E R

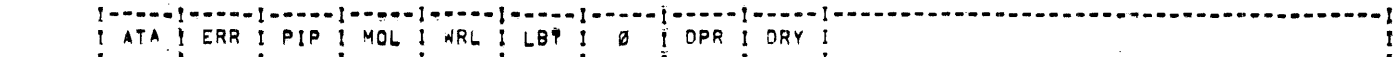
RSBA=772044 - B U S A D D R E S S R E G I S T E R

RSDA=772046 - D E S I R E D A D D R E S S R E G I S T E R

RSCS2=772050 - C O N T R O L S T A T U S R E G I S T E R #2

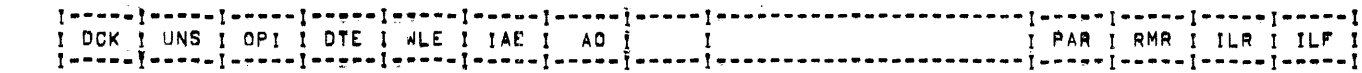
DLT	DATA LATE	MDPE	DATA BUS PARITY ERR
WCE	WRITE CHECK ERROR	OR	OUTPUT READY
UPE	UNIBUS PARITY ERROR	IR	INPUT READY
NED	NON-EXISTENT DISK	CLR	CONTROLLER CLEAR
NXM	NON-EXISTENT MEMORY	PAT	PARITY TEST PAT=1/Ø=EVEN/ODD
PGE	PROGRAM ERROR	BAI	BUS ADDRESS INHIBIT INC.
MXF	MISSED TRANSFER	UNIT	UNIT SELECT

RSDS-772052 - D R I V E S T A T U S R E G I S T E R



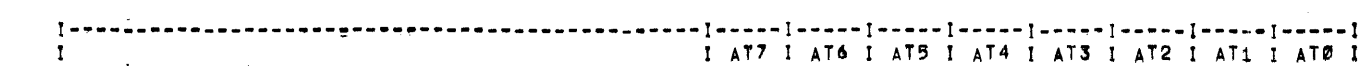
ATA	ATTENTION ACTIVE	WRL	WRITE LOCKED
ERR	ERROR SUMMARY	LBT	LAST BLOCK TRANSFERRED
PIP	POSITIONING IN PROGRESS	DPR	DRIVE PRESENT
MOL	MEDIUM ON LINE	DRY	DRIVE READY

RSER=772054 - E R R O R R E G I S T E R

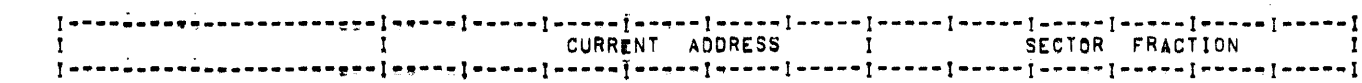


DCK	DATA CHECK	AO	DISK ADDRESS OVERFLOW
UNS	DRIVE UNSAFE	PAR	MASSBUS PARITY ERROR
OPI	OPERATION INCOMPLETE	RMR	REGISTER MODIFY REFUSED
DTE	DRIVE TIMING ERROR	ILR	ILLEGAL REGISTER REF,
WLE	WRITE LOCK ERROR	ILF	ILLEGAL FUNCTION
IAE	INVALID DISK ADDRESS		

RSAS=772056 - A T T E N T I O N S U M M A R Y



RSLA=772060 - L O O K A H E A D



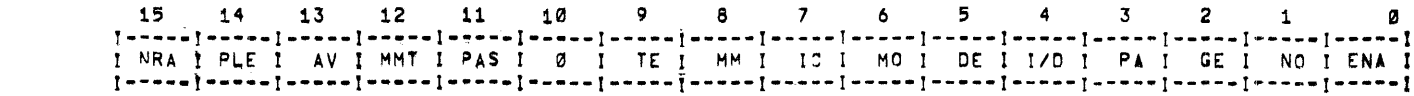
RSDB=772062 - D A T A B U F F E R

RSMR=772064 - M A I N T R E G I S T E R

RSDT=772066 - D R I V E T Y P E R E G I S T E R

APPENDIX A
KT11 C/D REGISTERS

SR0=777572 - S T A T U S R E G I S T E R #0

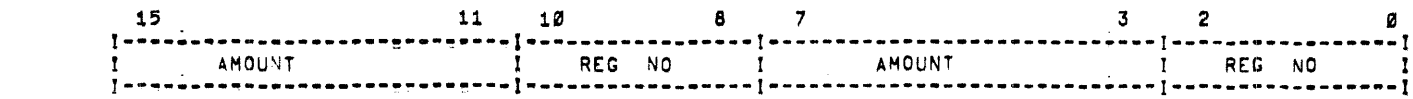


NRA - NON-RESIDENT ABORT
PLE - PAGE LENGTH ERROR
AV - ABORT = READ ONLY ACCESS VIOLATION
* MMT - MEMORY MANAGEMENT TRAP
* PAS - PROGRAMS AID SYSTEM FLAG
* TE - ENABLE MEMORY MANAGEMENT TRAPS
MM - MAINTENANCE MODE

* IC - INSTRUCTION COMPLETED
MODE- CPU MODE
00 = KERNAL 10 - NOT USED
01 = SUPERVISOR 11 - USER
* I/D - SEGMENT ADDRESS SPACE
0 = I SPACE 1 =D SPACE
PAGE- PAGE #
ENA - ENABLE MEMORY MANAGEMENT

* SR1=777574 = S T A T U S R E G I S T E R #1

SR1 RECORDS ANY AUTOINCRMENT/DECREMENT OF THE GPR'S, INCLUDING EXPLICIT REFERENCES THROUGH THE PC. SR1 IS CLEARED AT THE BEGINNING OF EACH INSTRUCTION FETCH, WHENEVER A GPR IS EITHER AUTO INCREMENTED/ DECREMENTED THE REGISTER NUMBER AND THE AMOUNT (IN 2'S COMPLEMENT NOTATION) IS RECORDED IN SR1.

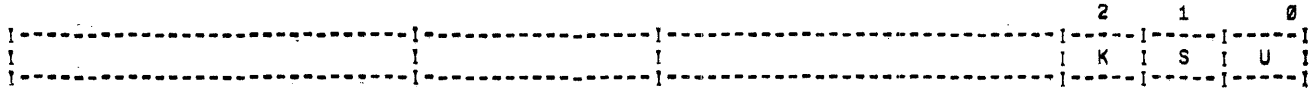


SR2=777576 = S T A T U S R E G I S T E R #2

SR2 IS LOADED WITH THE 16 BIT VIRTUAL ADDRESS AT THE BEGINNING OF EACH INSTRUCTION FETCH, OR WITH THE ADDRESS TRAP VECTOR AT THE BEGINNING OF AN INTERRUPT, 'T' BIT TRAP, PARITY, ODD ADDRESS, AND TIMEOUT TRAPS.

* SR3=772516 = S T A T U S R E G I S T E R #3

SR3 ENABLES OR DISABLES THE USE OF I/D SPACE. WHEN 'D' SPACE IS DISABLED ALL REFERENCES USE THE 'I' SPACE REGISTERS; WHEN 'D' SPACE IS ENABLED BOTH 'I' AND 'D' REGISTERS ARE USED. A '1' ENABLES 'D' SPACE.

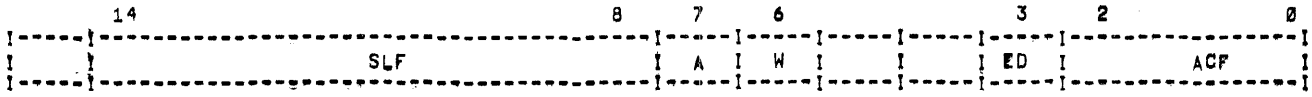


U = USER D SPACE ENABLE
S = SUPERVISOR 'D' SPACE ENABLE
K = KERNEL 'D' SPACE ENABLE

THE PROGRAM DOES NOT ENABLE ANY 'D' SPACES

P A G E D E S C R I P T O R R E G I S T E R S

	KIPDR0-KIPDR7	772300-772316	*KDPDR0-KDPDR7 772320-772336
	UIPDR0-UIPDR7	777600-777616	*UPDPR0-UPDPR7 777620-777636
*	SIPDR0-SIPDR7	772200-772216	*SDPDR0-SDPDR7 772220-777236



ACF = ACCESS CONTROL FIELD
- 000 NON-RESIDENT
- 100 READ/WRITE
- * 101 READ/WRITE
- 110 READ/WRITE
- * 001 READ ONLY
- 010 READ ONLY
- * 011 UNUSED
- * 111 UNUSED

ED = EXPANSION DIRECTION (1/0 = DOWN/UP)

W = SEGMENT HAS BEEN WRITTEN INTO

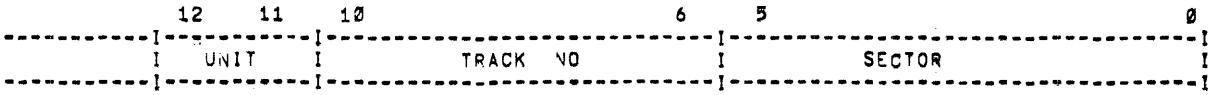
* A = SEGMENT HAS BEEN ACCESSED

SLF = SEGMENT LENGTH FIELD
BINARY REPRESNETATION OF NUMBER OF 32 WORD BLOCKS IN SEGMENT.

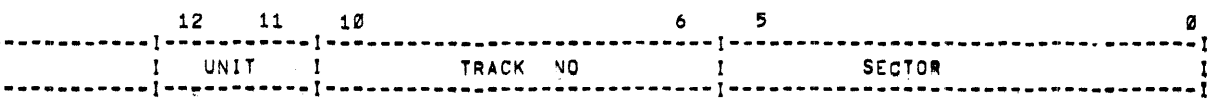
* 11/45 ONLY

APPENDIX B
RC11 REGISTERS

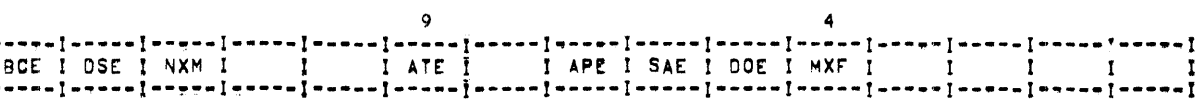
INTERRUPT VECTOR = 210
RCLA=777440 - L O O K A H E A D R E G I S T E R



RCDA=777442 - D I S K A D D R E S S R E G I S T E R

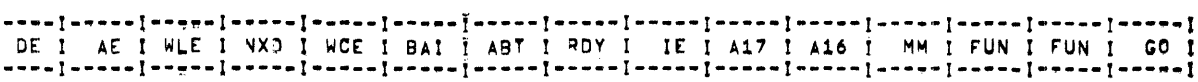


RCER=777444 - D I S K E R R O R R E G I S T E R



DLE - DATA LATE ERROR
BCE - BLOCK CHECK ERROR
DSE - DATA SYNC ERROR
NXM - NON-EXIST MEMORY
ATE - A TRACK ERROR
APE - ADDRESS PARITY ERROR
SAE - SYNC ADDRESS ERROR
DOE - DISK OVERFLOW ERROR
MXF - MISSED TRANSFER

RCCS=777446 - D I S K C O N T R O L & S T A T U S R E G I S T E R



SC - SPECIAL CONDITION
DE - DATA ERROR
AE - ADDRESS ERROR
WLE - WRITE LOCK ERROR
NXD - NON-EXISTENT DISK
WCE - WRITE CHECK ERROR
BAI - INHIBIT CA INCREMENT
ABT - ABORT
RDY - READY
IE - INTERRUPT ENABLE
A17 - BUS ADDRESS BIT 17
A16 - BUS ADDRESS BIT 16
MM - MAINTENANCE MODE
FUN - FUNCTION
00 LOOK AHEAD 10 READ
01 WRITE 11 WRITE CHECK
GO - GO

RCWC=777450 - W O R D C O U N T R E G I S T E R

RCCA=777452 - C U R R E N T A D D R E S S R E G I S T E R

RCMN=777459 - M A I N T E N A N C E R E G I S T E R

RCDB=777456 - D A T A B U F F E R R E G I S T E R

APPENDIX C
RF11 REGISTERS

INTERRUPT VECTOR = 204

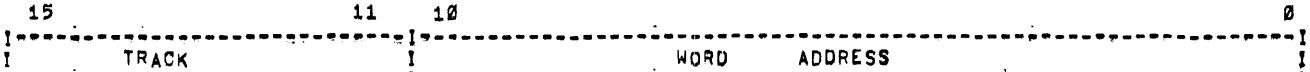
RFDCS=777460 = D I S K C O N T R O L S T A T U S R E G I S T E R

ERR	ERR (LOGICAL OR OF 14-9)	IE	INTERRUPT ENABLE
FRZ	FREEZE (LOGICAL OR OF DAE 15-10)	A17	BUS ADDRESS BIT 17
WCE	WRITE CHECK ERROR	A16	BUS ADDRESS BIT 16
DPE	DATA PARITY ERROR	MM	MAINTENANCE MODE
NED	NON EXISTANT DISK	FUN	FUNCTION
WLO	WRITE LOCK OUT		00 NO OP 10 READ
MXF	MISSED TRANSFER		01 WRITE 11 WRITE CHECK
CLR	DISK CLEAR		
RDY	CONTROL READY		

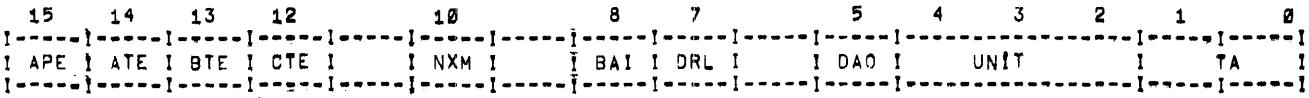
RFWC=777462 = W O R D C O U N T R E G I S T E R

RFCMA=777464 = C U R R E N T M E M O R Y A D D R E S S

RFDAR=777466 = D I S K A D D R E S S R E G I S T E R



RFDAE=777470 = D I S K A D D R E S S E X T E N S I O N E R R O R R E G I S T E R



APE	ADDRESS PARITY ERROR	DRL	DATA REQUEST LATE
ATE	A TIMING TRACK ERROR	DAO	DISK ADDRESS OVERFLOW
BTE	B TIMING TRACK ERROR	UNIT	UNIT NUMBER
CTE	C TIMING TRACK ERROR	TA	EXTENSION OF TRACK ADDRESS
NXM	NON-EXISTANT MEMORY		IN RFDAR
BAI	BUS ADDRESS INHIBIT		

RFDBR=777422 = D A T A B U F F E R R E G I S T E R

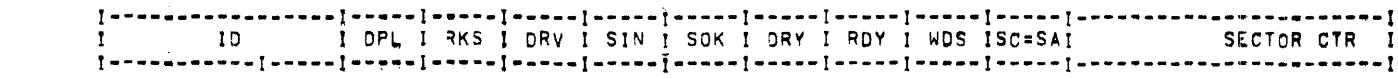
RFMA=777474 = M A I N T E N A N C E R E G I S T E R

RFADS=777476 = L O O K A H E A D

APPENDIX D
RK11 REGISTERS

INTERRUPT VECTOR = 220

RKDS=777400 - D R I V E S T A T U S R E G I S T E R

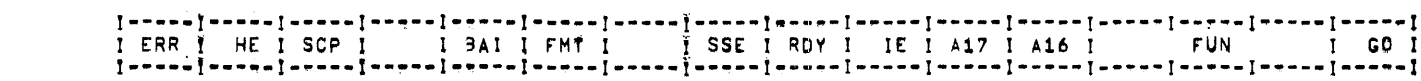


ID	IDENT OF DRIVE	DRY	DRIVE READY
DPL	DRIVE POWER LOW	RDY	READ/WRITE/SEEK READY
RKS	SET TO INDICATE RK05	WPS	WRITE PROTECT STATUS
DRV	DRIVE UNSAFE	SC=SA	SECTOR COUNTER=SECTOR ADDRESS
SIN	SEEK INCOMPLETE	SC	SECTOR
SOK	SECTOR COUNTER OK		

RKER=777402 - E R R O R R E G I S T E R

DRE	DRIVE ERROR	TE	TIMING ERROR
OVR	DISK OVERRUN (OVERFLOW)	NXD	NON-EXISTANT DISK
WLO	WRITE LOCK	NXC	NON-EXISTENT CYLINDER
SKE	SEEK ERROR	NXS	NON-EXISANT SECTOR
PGE	PROGRAMMING ERROR	CSE	CHECK SUM ERROR
NXM	NON-EXISTANT	WCE	WRITE CHECK ERROR
DLT	DATA LATE		

RKCS-777404 - C O N T R O L S T A T U S R E G I S T E R

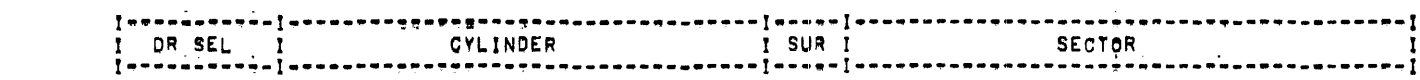


ERR	ERROR(OR OF RKER & RKCS<14>)	A17	BUS ADDRESS BIT 17
HE	HARD ERROR	A16	BUS ADDRESS BIT 16
SCP	SEARCH COMPLETE	FUN	FUNCTION
BAI	BUS ADDRESS INHIBIT INC.		000 RESET 100 SEEK
FMT	FORMAT MODE		001 WRITE 101 RD CHK
SSE	STOP ON SOFT ERROR		010 READ 110 DRV RESET
RDY	CONTROL READY		011 WT CHK 111 WT LOCK
IE	INTERRUPT ENABLE	GO	ENABLE FUNCTION

RKWC-777406 - W O R D C O U N T R E G I S T E R

RKLA-777410 - C U R R E N T B U S A D D R E S S

RKDA-777412 - D I S K A D D R E S S



RKDB-777416 - D A T A B U F F E R R E G I S T E R

APPENDIX E
RP11C REGISTERS

INTERRUPT VECTOR = 254

RPDS=776710 - D R I V E S T A T U S R E G I S T E R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SUR	ONL	RP03	HNF	SI	SU	UNSF	WP	ATN7	ATTENTION						ATN0

SUR	SELECTED UNIT READY	SU	SEEK UNDERWAY
ONL	SELECTED UNIT ON LINE	UNSF	SELECTED UNIT UNSAFE
RP03	SELECTED UNIT IS RP03	WLO	SELECTED UNIT WRITE LOCKED
HNF	HEADER NOT FOUND	ATN7-ATN0	DRIVE ATTENTION
SI	SEEK INCOMPLETE		

RPER=776712 - E R R O R R E G I S T E R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
WPV	FUV	NXC	NXT	NXS	PGE	FMT	MOD	LPE	WPE	CSE	TE	WCE	NXM	EOP	DER

WPV	WRITE PROT VIOL	LPE	LONG PARITY ERROR
FUV	FILE UNSAFE VIOL	WPE	WORD PARITY ERROR
NXC	NON-EXISTANT CYLINDER	CSE	CHECK SUM ERROR
NXT	NON-EXISTANT TRACK	TE	TIMING ERROR
NXC	NON-EXISTANT SECTOR	WCE	WRITE CHECK ERROR
PGE	PROGRAM ERROR	NXM	NON-EXISTANT MEMORY
FMT	FORMAT ERROR	EOP	END OF PACK
MOD	MODE ERROR	DER	DISK ERROR

RPCS=776714 - C O N T R O L S T A T U S R E G I S T E R

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
ERR	HE	AIE	MOD	HDR	DRIVE SELECT			RDY	IE	A17	A16	FUNCTION			GO

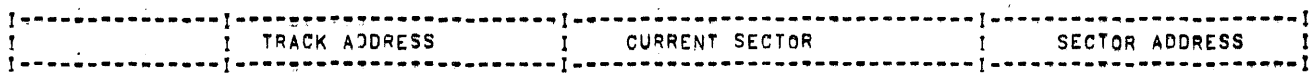
ERR	ERROR (OR OF ALL ERROR BITS)	A17	BUS ADDRESS BIT 17				
HE	HARD ERROR (OR OF ALL BUT DATA ERROR BITS)	A16	BUS ADDRESS BIT 16				
AIE	ATTENTION INTERRUPT ENABLE	FUN	FUNCTION				
MODE	RP11-C IS CONDITIONED TO RD /WRT DISK PACKS IN PDP10 OR PDP 15 FORMAT		000	INIT	100	SEEK	
			001	WRITE	101	WRITE (NO SEEK)	
			010	READ	110	HOME SEEK	
			011	WRT CK	111	RD (NO SEEK)	
HDR	FUNCTION IS A HEADER OPER,	GO	ENABLE OPERATION				
DRIVE SELECT							
RDY	READY						
IE	INTERRUPT ENABLE						

RPWC=776716 - W O R D C O U N T R E G I S T E R

RPBA=776720 - B U S A D D R E S S R E G I S T E R

RPCA=776722 - C Y L I N D E R A D D R E S S R E G I S T E R
<BITS 00-08>

RPDA=776724 = D I S K A D D R E S S R E G I S T E R

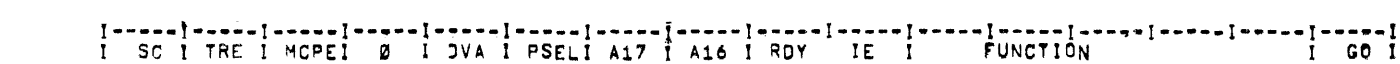


SUCA=776734 = S E L E C T E D U N I T C Y L I N D E R A D R S
<BITS 00=08>

APPENDIX F
RS03/4 REGISTERS

INTERRUPT VECTOR = 204

RSCS1=772040 = C O N T R O L S T A T U S R E G #1

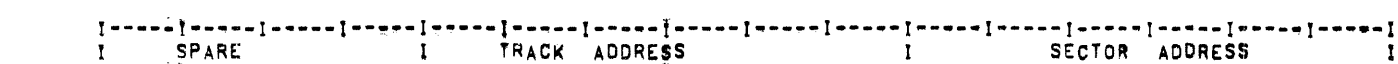


SC	SPECIAL CONDITION	FUN	FUNCTION
TRE	TRANSFER ERROR	00000	NO OPERATION
MCPE	MASS BUS CONTROL BUS PARITY ERROR	00100	DRIVE CLEAR
DVA	DRIVE AVAILABLE	01100	SEARCH
PSEL	PORT SELECT	10100	WRITE CHECK
A17	BUS ADDRESS BIT 17	11000	WRITE
A16	BUS ADDRESS BIT 16	11100	READ
RDY	READY	GO	ENABLE FUNCTION
IE	INTERRUPT ENABLE		

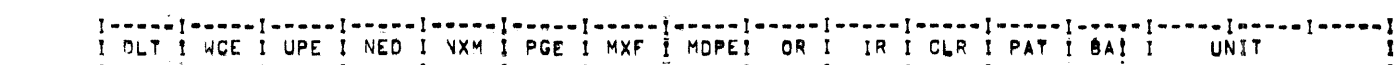
RSWC=771042 = W O R D C O U N T R E G I S T E R

RSBA=772044 = B U S A D D R E S S R E G I S T E R

RSDA=772046 = D E S I R E D A D D R E S S R E G I S T E R



RSCS2=772050 = C O N T R O L S T A T U S R E G I S T E R #2

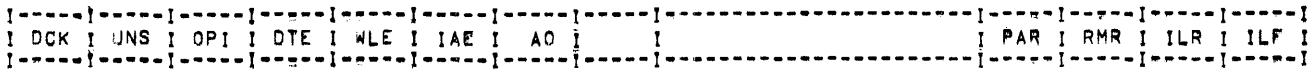


DLT	DATA LATE	MDPE	DATA BUS PARITY ERR
WCE	WRITE CHECK ERROR	OR	OUTPUT READY
UPE	UNIBUS PARITY ERROR	IR	INPUT READY
NED	NON-EXISTENT DISK	CLR	CONTROLLER CLEAR
NXM	NON-EXISTENT MEMORY	PAT	PARITY TEST PAT=1/Ø=EVEN/ODD
PGE	PROGRAM ERROR	BAI	BUS ADDRESS INHIBIT INC.
MXF	MISSED TRANSFER	UNIT	UNIT SELECT

RSDS=772052 = D R I V E S T A T U S R E G I S T E R

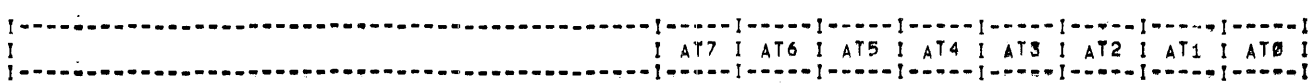
ATA	ATTENTION ACTIVE	WRL	WRITE LOCKED
ERR	ERROR SUMMARY	LBT	LAST BLOCK TRANSFERRED
PIP	POSITIONING IN PROGRESS	DPR	DRIVE PRESENT
MOL	MEDIUM ON LINE	DRY	DRIVE READY

RSER=772054 - E R R O R R E G I S T E R

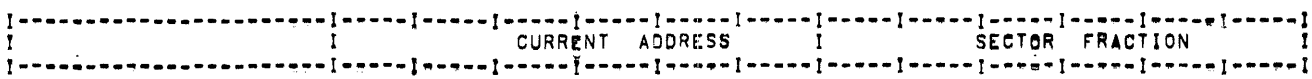


DCK	DATA CHECK	AO	DISK ADDRESS OVERFLOW
UNS	DRIVE UNSAFE	PAR	MASSBUS PARITY ERROR
OPI	OPERATION INCOMPLETE	RMR	REGISTER MODIFY REFUSED
DTE	DRIVE TIMING ERROR	ILR	ILLEGAL REGISTER REF.
WLE	WRITE LOCK ERROR	ILF	ILLEGAL FUNCTION
IAE	INVALID DISK ADDRESS		

RSAS=772056 - A T T E N T I O N S U M M A R Y



RSLA=772060 - L O O K A H E A D



RSDB=772062 - D A T A B U F F E R

RSMR=772064 - M A I N T R E G I S T E R

RSDT=772066 - D R I V E T Y P E R E G I S T E R

DCQKCD TABLE OF CONTENTS

```
                        .NLIST  MD,MC
                        .LIST   ME
                        .ABS
                        .TITLE  DCQKCD  11/40-11/45 CPU EXERCISER
                        .SBTTL  SWITCH SETTING
                ;       SW15---HALT ON ERROR
                ;       SW14---LOOP TEST
                ;       SW13---INHIBIT ERROR TYPEOUT
                ;       SW12---SEE NOTE BELOW
                ;       SW11---INHIBIT TEST ITERATIONS
                ;       SW10---RING BELL ON ERROR
                ;       SW09---SEE NOTE BELOW
                ;       SW08---LOAD MICRO BRAK REGISTER WITH SW07-SW00
                ;       SW07---TYPE END OF PASS MESSAGE
                ;       SW06---DISABLE CLOCKS
                ;       SW05---RELOCATE USING ALL DEVICES ROUND ROBIN STYLE
                ;       SW04---USE RANDOM DISK ADDRESS FOR RELOCATION
                ;       SW03---RELOCATE USING DEVICE SELECTED IN SW02-SW00
                ;       SW02-SW00---    0=CP
                ;                       1=RK
                ;                       2=RF
                ;                       3=RP
                ;                       4=RC
                ;                       5=DO NOT USE
                ;                       6=RS03/RS04
                ;                       7=RESERVED FOR FUTURE USE (IS CP)
                ;NOTE BELOW! SW12 AND SW09 CONTROL PROGRAM RELOCATION DESRIBED BELOW!
                ;       SW12    SW09    RELOCATION
                ;       1       0       NONE
                ;       0       1       NO RELOCATION ABOVE 28K
                ;       1       1       NOT USED (DO NOT USE)
                ;       0       0       ALL MEMORY


                        .SBTTL  DEFINITIONS & ASSIGNMENTS
                ;GENERAL REGISTER ASSIGNMENTS
000000                  R0=%0
000001                  R1=%1
000002                  R2=%2
000003                  R3=%3
000004                  R4=%4
000005                  R5=%5
000006                  SP=%6
000007                  PC=%7
000000                  R10=%0
000001                  R11=%1
000002                  R12=%2
000003                  R13=%3
000004                  R14=%4
000005                  R15=%5

                ;FLOATING POINT REGISTERS
000000                  AC0=%0
```



```
000001                  AC1=%1
000002                  AC2=%2
000003                  AC3=%3
000004                  AC4=%4
000005                  AC5=%5

                ;STACK POINTER REGISTERS
000006                  KSP=%6                  ;KERNEL STACK POINTER
000006                  SSP=%6                  ;SUPERVISOR STACK POINTER
000006                  USP=%6                  ;USER STACK POINTER

                ;STATUS REGISTER (PSW) BIT ASSIGNMENTS
000001                  C=1                     ;C BIT
000002                  V=2                     ;V BIT
000004                  Z=4                     ;Z BIT
000010                  N=10                    ;N BIT
000020                  T=20                    ;'T' BIT
000340                  PRTY7=340               ;PRIORITY LEVEL 7
000300                  PRTY6=300               ;PRIORITY LEVEL 6
000240                  PRTY5=240               ;PRIORITY LEVEL 5
000200                  PRTY4=200               ;PRIORITY LEVEL 4
000140                  PRTY3=140
000100                  PRTY2=100               ;PRIORITY LEVEL 2
000000                  KM=000000               ;KERNEL MODE
040000                  SM=040000               ;SUPERVISORY MODE
140000                  UM=140000               ;USER MODE
000000                  PKM=000000              ;PREVIOUS KERNEL MODE
010000                  PSM=010000              ;PREVIOUS SUPERVISORY MODE
030000                  PUM=030000              ;PREVIOUS USER MODE
004000                  REG=004000              ;SELECT R10-R15

                ;VECTOR ADDRESSES
000004                  ERRVEC=  4              ;ADDRESS OF ERROR VECTOR
000010                  RESVEC= 10              ;ADDRESS OF RESERVED INST, TRAP VECTOR
000014                  TBITVEC=14              ;ADDRESS OF 'T' BIT TRAP VECTOR
000014                  TRTVEC=  14             ;ADDRESS OF 'TRACE' TRAP VECTOR
000014                  BPTVEC=  14             ;ADDRESS OF 'BREAKPOINT' TRAP VECTOR
000020                  IOTVEC=  20             ;ADDRESS OF IOT TRAP VECTOR
000024                  PFVEC=   24             ;ADDRESS OF POWER FAIL TRAP VECTOR
000030                  EMTVEC=  30             ;ADDRESS OF EMT VECTOR
000034                  TRAPVEC=34              ;ADDRESS OF TRAP VECTOR
000060                  TKVEC=   60             ;ADDRESS OF KEYBOARD INTERRUPT VECTOR
000064                  TPVEC=   64             ;ADDRESS OF TTY PRINTER INTERRUPT VECTOR
000070                  PRVEC=   70             ;HIGH SPEED READER INTERRUPT VECTOR
000074                  PPVEC=   74             ;HIGH SPEED PUNCH INTERRUPT VECTOR
000100                  LKVEC=   100            ;ADDRESS KW11-L LINE CLOCK INT, VECTOR
000104                  PLKVEC=  104            ;ADDRESS OF KW11-P CLOCK INT VECTOR
000204                  RFVEC=   204            ;RF OR RS04 VECTOR
000204                  RSVEC=   204
000210                  RCVEC=   210            ;RC VECTOR
000220                  RKVEC=   220            ;RK DISK VECTOR
000240                  PIRVEC=  240            ;ADDRESS OF PIRQ VECTOR
000244                  FPEVEC=  244            ;ADDRESS OF FLOATING POINT INT, VECTOR
000250                  MMVEC=   250            ;ADDRESS OF MEM MGMT ERROR TRAP VECTOR
```

```
000254                  RPVEC=  254             ;RP VECTOR
000254                  RP4VEC= 254             ;RP04 VECTOR

                        ;REGISTER ADDRESSES
177776                  PSW=    177776          ;ADDRESS OF STATUS REGISTER
177774                  SLR=    177774          ;ADDRESS OF STACK LIMIT REGISTER
177772                  PIRQ=   177772          ;ADDRESS OF PROGRAM INTERRUPT REQUEST
177770                  UBREAK= 177770          ;ADDRESS OF MICRO BREAK REGISTER
177766                  CPUERR= 177766
177744                  ERRREG= 177744
177546                  LKS=    177546          ;ADDRESS OF KW11-L STATUS REG,
177550                  PRS=    177550          ;ADDRESS OF HIGH SPEED READER CSR
177552                  PRB=    177552          ;ADDRESS OF HIGH SPEED READER DATA BUF
177554                  PPS=    177554          ;ADDRESS OF HIGH SPEED PUNCH CSR
177556                  PPB=    177556          ;ADDRESS OF HIGH SPEED PUNCH BUFFER
177560                  TKS=    177560          ;ADDRESS OF KEYBOARD CSR
177562                  TKB=    177562          ;ADDRESS OF KEYBOARD BUFFER
177564                  TPS=    177564          ;ADDRESS OF TELEPRINTER CSR
177566                  TPB=    177566          ;ADDRESS OF TELEPRINTER BUFFER
177572                  SR0=    177572          ;ADDRESS OF MEM MGMT REGISTER SR0
177574                  SR1=    177574          ;ADDRESS OF MEM MGMT REG SR1
177576                  SR2=    177576          ;ADDRESS OF MEM MGMT REGISTER SR2
172516                  SR3=    172516          ;ADDRESS OF MEM MGMT REGISTER SR3
177570                  SWR=    177570          ;ADDRESS OF CONSOL SWITCH REGISTER
177570                  DISPLAY=177570          ;ADDRESS OF CONSOL DISPLAY REGISTER
177514                  LPS=    177514          ;ADDRESS OF LINE PRINTER STATUS REG
177516                  LPB=    177516          ;ADDRESS OF LINE PRINTER DATA BUFFER

                        ;RK REGISTERS
177400                  RKDS=   177400          ;ADDRESS OF RK-11 DISK DRIVE STATUS REGISTER
177402                  RKER=   177402          ;ADDRESS OF RK-11 DISK ERROR REGISTER
177404                  RKCS=   177404          ;ADDRESS OF RK-11 DISK CONT, AND STATUS REG,
177406                  RKWC=   177406          ;ADDRESS OF RK-11 DISK WORD COUNT REG,
177410                  RKBA=   177410          ;ADDRESS OF RK-11 DISK BUS ADDRESS REG,
177412                  RKDA=   177412          ;ADDRESS OF RK-11 DISK ADDRESS REG,

                        ;RF REGISTERS
177460                  RFDCS=  177460          ;ADDRESS OF RF-11 DISK CONT, AND STATUS REG,
177462                  RFWC=   177462          ;ADDRESS OF RF-11 DISK WORD COUNT REG,
177464                  RFCMA=  177464          ;ADDRESS OF RF-11 DISK MEMORY ADR,REG,
177466                  RFDAR=  177466          ;ADDRESS OF RF-11 DISK ADDRESS REG,
177470                  RFDAE=  177470          ;ADDRESS OF RF DAE REGISTER

                        ;RC REGISTERS
177440                  RCLA=   177440          ;ADDRESS OF RC-11 LOOK AHEAD REGISTER
177442                  RCDA=   177442          ;ADDRESS OF RC-11 DISK ADDRESS REG,
177446                  RCCS=   177446          ;ADDRESS OF RC-11 DISK CONT, AND STATUS REG,
177450                  RCWC=   177450          ;ADDRESS OF RC-11 DISK WORD COUNT REG,
177452                  RCCA=   177452          ;ADDRESS OF RC-11 CURRENT DISK ADR REG,

                        ;RP04 REGISTERS
176700                  RP4CS1= 176700          ;RP04 CS1 REGISTER
176702                  RP4WC=  176702          ;WORD COUNT REGISTER
176704                  RP4BA=  176704          ;BUS ADDRESS REGISTER
```



```
176706                  RP4DST= 176706          ;DESIRED SECTOR/TRACK REGISTER
176734                  RP4CA=  176734          ;DISK ADDRESS REGISTER
176712                  RP4DS1= 176712          ;DRIVE STARTUS REGISTER #1
176714                  RP4ER1= 176714          ;ERROR REGISTER #1
176716                  RP4AS=  176716          ;ATTENTION SUMMARY
176720                  RP4LA=  176720          ;LOOK AHEAD REGISTER
176732                  RP4OF=  176732          ;OFFSET REGISTER

                        ;RH11 MASS BUS CONTROLLER REGISTERS
000000                  RHCS2=  0                       ;NOT DEFINED

                        ;RP11C REGISTERS
176710                  RPDS=   176710          ;ADDRESS OF RP DRIVE STATUS REGISTER
176712                  RPER=   176712          ;ADDRESS OF RP ERROR REGISTER
176714                  RPCS=   176714          ;ADDRESS OF RP CONTROL STATUS REGISTER
176716                  RPWC=   176716          ;ADDRESS OF RP WORD COUNT REGISTER
176720                  RPBA=   176720          ;ADDRESS OF RP BUS ADDRESS REGISTER
176722                  RPCA=   176722          ;ADDRESS OF RP CYLINDER ADDRESS REGISTER
176724                  RPDA=   176724          ;ADDRESS OF RP DISK ADDRESS REGISTER

                        ;KW11-P REGISTERS
172540                  PLKCSR= 172540          ;ADDRESS OF KW11-P CLOCK CSR
172542                  PLKCSB= 172542          ;ADDRESS OF KW11-P COUNT SET BUFFER
172544                  PLKCTR= 172544          ;ADDRESS OF KW11-P COUNTER

                        ;RS04 REGISTERS
172040                  RS0S1=  172040          ;CONTROL STATUS REGISTER
172042                  RSWC=   172042          ;WORD COUNT REGISTER
172044                  RSBA=   172044          ;BUS ADDRESS REGISTER
172046                  RSDA=   172046          ;DISK ADDRESS REGISTER
172050                  RSCS2=  172050          ;CONTROL STATUS #2
172052                  RSDS=   172052          ;DRIVE STATUS REGISTER
172054                  RSER=   172054          ;ERROR REGISTER #1
172056                  RSAS=   172056          ;ATTENTION SUMMARY REGISTER
172060                  RSLA=   172060          ;LOOK AHEAD REGISTER

                        ;MEMORY MANAGEMENT REGISTER ADDRESSES
172300                  KIPDR0= 172300
172302                  KIPDR1= 172302
172304                  KIPDR2= 172304
172306                  KIPDR3= 172306
172310                  KIPDR4= 172310
172316                  KIPDR7= 172316
172340                  KIPAR0= 172340
172342                  KIPAR1= 172342
172344                  KIPAR2= 172344
172346                  KIPAR3= 172346
172350                  KIPAR4= 172350
172356                  KIPAR7= 172356

177600                  UIPDR0=177600
177602                  UIPDR1=177602
177610                  UIPDR4=177610
177614                  UIPDR6=177614
```

```
                        UIPDR7=177616
                        UIPAR0=177640
                        UIPAR1=177642
                        UIPAR4=177650
                        UIPAR6=177654
                        UIPAR7=177656

                        SIPDR0=172200
                        SIPDR1=172202
                        SIPDR4=172210
                        SIPDR6=172214
                        SIPDR7=172216
                        SIPAR0=172240
                        SIPAR1=172242
                        SIPAR4=172250
                        SIPAR6=172254
                        SIPAR7=172256
                        KDPDR0=172320
                        UDPDR0=177620
                        SDPDR0=172220
                        KDPAR0=172360
                        UDPAR0=177660
                        SDPAR0=172260

                ;INITIAL STACK POINTER SETTING
                        STKPTR= 500             ;PROGRAM STACK PTR
                        KPTR=   600             ;KERNEL STACK PTR (USED BY KERNEL WHEN
                                                ;PROG IS RUNNING IN OTHER THAN KERNEL MODE

                ;MISCELLANEOUS BIT ASSIGNMENTS (USED IN OPT,CP)
                        KTOPT=  100000                  ;BELOW BIT ASSIGNMENTS ARE USED
                        EISOPT= 040000                  ;IN THE CPCHK ROUTINE
                        FPOPT=  020000                  ;A BIT FOR EACH OPTION PRESENT
                        FISOPT= 010000                  ;IS SET IN OPT,CP (ODD BYTE)
                        KJOPT=  004000
                        PLKOPT= 002000
                        LKOPT=  001000
                        TTOPT=  000400

                ;BIT ASSIGNMENTS USED IN OPTIONS
                        PROPT=  000001
                        PPOPT=  000002

                        BIT15=  100000
                        BIT14=  040000
                        BIT13=  020000
                        BIT8=   400
                        BIT6=   100
                        PIR4=   10000           ;LEVEL 4 PROGRAM INT. RQST, (FOR PIRQ)

                ;INSTRUCTION EQUATES
                        HLT=    TRAP            ;HLT IS A TRAP INST TO THE ERROR ROUTINE
                        SCOPE=  EMT             ;SCOPE IS AN EMT TRAP
                        TYPE=   IOT
```

```
                000020                          .=IOTVEC
000020  002564                                  .WORD   ,TYPE                   ;SET IOT VECTOR TO TYPE ROUTINE
000022  000200                                  .WORD   PRTY4
000024  000610                                  .WORD   PDWN                    ; AT POWER FAIL VECTOR
000026  000340                                  .WORD   PRTY7
000030  001014                                  .WORD   SCOPEA
000032  000200                                  .WORD   PRTY4
000034  003212                                  .WORD   ,HLT
000036  000340                                  .WORD   PRTY7
                000060                          .=TKVEC
000060  003130                                  .WORD   TKISR                   ;SET KEYBOARD VECTOR TO TKISR
000062  000200                                  .WORD   PRTY4                   ;PRIORITY LEVEL 4

                                                .SBTTL  ENABLE PARITY & POWER FAIL ROUTINES
                000120                          .=120
                                        ;ROUTINE TO SET PARITY ACTION ON PARITY MEMORIES
                172100                  PARCSR= 172100          ;ADDRESS OF FIRST POSSIBLE PARITY REG
                000114                  PARVEC= 000114          ;ADDRESS OF PARITY INTERRUPT VECTOR

000120  012737  004356  000114  .MAMF:  MOV     #.PARSRV,@#PARVEC       ;LOAD VECTOR
000126  012737  000340  000116          MOV     #340,@#PARVEC+2         ;AND PRIORITY LEVEL
000134  012737  000006  000004          MOV     #ERRVEC+2,@#ERRVEC      ;DO RTI ON TIME OUT TRAP
000142  012700  172100                  MOV     #PARCSR,R0              ;GET FIRST POSSIBLE ADDRESS
000146  012702  000001                  MOV     #1,R2                   ;SET REGISTER COUNTER

000152  012720  000001          1$:     MOV     #1,(R0)+                ;SET ACTION ENABLE (IF AVAIL)
                                        ;ABOVE INSTRUCTION WILL SET ACTION ENABLE IF MA/MF PARITY OR SET
                                        ;ODD PARITY AND HALT ON PARITY ERROR IF MOS PARITY
000156  006302                          2$:     ASL     R2                      ;CHECK IF 16. REGISTERS HAVE
000160  103374                          BCC     1$                      ;BEEN ENABLED
000162  000207                          RTS     PC                      ;RETURN

                000200                          .=200
000200  012707  005422                  MOV     #START,PC       ;GO TO START OF TEST
000204  012707  005532                  MOV     #START1,PC      ;GO GET LOWER/UPPER RELOCATION BOUNDARY
000210  012707  005600                  MOV     #START3,PC      ;START WITH LAST TYPED BOUNDARY LIMITS

                000244                          .=244
000244  000246                          .WORD   246             ;SET FIS TRAP TO RETURN DIRECT
000246  000002                          .WORD   RTI
                000252                          .=252                   ;MEM MGMT PRIORITY ADDRESS
000252  000340                          .WORD   340             ;SET AT LEVEL 7
                000610                          .=610

                                        ;POWER FAIL SUBROUTINE
000610  005737  000764          PDWN:   TST     @#CNT,CP
000614  100002                          BPL     1$
000616  005037  177572                  CLR     @#SR0
000622  012737  000632  000024  1$:     MOV     #PUP,@#PFVEC
000630  000000                          HALT

                                        ;POWER UP SUBROUTINE
000632  012737  000610  000024  PUP:    MOV     #PDWN,@#PFVEC   ;RESET POWER FAIL TRAP VECTOR TO POWER
                                                                ;DOWN ROUTINE ABOVE
000640  012706  000600                  MOV     #KPVR,SP        ;SET STACK PTR
```



```
000644  005027                          CLR     (PC)+
000646  000000                  1$:     .WORD   0               ;KILL TIME
000650  005267  177772          2$:     INC     1$
000654  001375                          BNE     2$
000656  000004                          TYPE,PFAIL
000662  000137  005422                  JMP     @#START         ;RESTART TEST

000666  005015  047520  042527  PFAIL:  .ASCIZ  <15><12>'POWER FAILED'<15><12>
000674  020122  040506  046111
000702  042105  005015      000
000707     015  050012  051101  PARERR:  .ASCIZ  <15><12>'PARITY ERROR'<15><12>
000714  052111  020131  051105
000722  047522  006522  000012
```

```
                                        .SBTTL  PROG INDICATORS & SCOPE ROUTINE
                                ;THE BELOW TABLE CONTAINS ERROR INFORMATION NEEDED TO REPORT
                                ;MEMORY ERRORS DETECTED DURING PROGRAM RELOCATION, THE ERROR INFOR-
                                ;MATION IS PLACED IN THE TABLE BY THE 'SAVVAL' SUBROUTINE,AND
                                ;IS PROCESSED BY THE 'PNTPEOO' SUBROUTINE.
000730  000000          MEMTBL: .WORD   0                       ;CONTAINS 'GOOD' ADDRESS
000732  000000                  .WORD   0                       ;CONTAINS 'GOOD' DATA
000734  000000                  .WORD   0                       ;CONTAINS 'BAD' DATA
000736  000000                  .WORD   0                       ;CONTAINS 'BAD' DATA
000740  000000          ECHO:   .WORD   0                       ;LOCATION FOR ECHOED CHARACTER
000742  020040  042440  051122 DEVERR: .ASCII  '   ERROR'
000750  051117
000752  005015    000   CRLF:   .ASCIZ  <15><12>
000755     134    000   SLASH:  .ASCIZ  '\'
000757     000          DEV:    .BYTE   0                       ;CONTAINS DEVICE ID FOR ALL
000760     000          IORETRY: .BYTE  0                       ;CONTAINS DEVICE RETRY COUNT
000761     000          PEFLG:  .BYTE   0                       ;PARITY ERROR FLAG
                                .EVEN
000762  000000          EABITS: .WORD   0                       ;CONTAINS EA BITS FOR DISK XFERS
000764  000400          OPT.CP: .WORD   400                     ;CONTAINS OPTION AND CP INDICATORS
                                                                ;EVEN BYTE:4=11/40,6=11/45
                                                                ;ODD BYTE: 200=KT,100=EIS
                                                                ;40=11/45 FPP,20=11/40 FIS
                                                                ;10=STACK LIMIT (KJ)
                                                                ;4=KW11-P,2=KW11-L,1=CONSOLE TTY
000766  000000          OPTIONS: .WORD  0
000770     001          PRDAT:  .BYTE   1                       ;CONTAINS NEXT DATA TO BE READ
000771     000          PRSYNC: .BYTE   0                       ;CONTAINS SYNC COUNT
                        000770          .=770
000770     000          MMON:   .BYTE   0               ;MEM MGMT ON/OFF IND 1/0=ON/OFF
000771     000          QV:     .BYTE   0               ;QUICK VERIFY MODE IND
000772  000000          DEVID:  .WORD   0               ;CONTAINS DEVICE INDENTIFIER
000774  000000          LTICKS: .WORD   0               ;CONTAINS L CLOCK TICK COUNT
000776  000000          PTICKS: .WORD   0               ;CONTAINS P CLOCK TICK COUNT
001000  000000          ICNT:   .WORD   0               ;CONTAINS PASS COUNT
001002  001000          SFILLS: .WORD   1000            ;CONTAINS FILLS COUNT (2) IN ODD BYTE
                                                        ;AND FILLER CHARACTER (0) IN EVEN BYTE
                                ;FILLER COUNTS: VT05 @2400 BD=4,VT05 @1200 BD=2,VT05 @600 BD=1
                                ;               LA30S @110 BD=2,LA30S @150 BD=4,LA30S @300 BD=12
                                ;               ALL VALUES ARE OCTAL
001004  000000          FACTOR: .WORD   0               ;CONTAINS RELOCATION FACTOR, SUBTRACT # IN
                                                        ;FACTOR FROM PC TO GET PC OF ORIG CODE
001006  000000          RELR1:  .WORD   0               ;CONTAINS RELOCATED R1 (THE R1 OF THE
                                                        ;ORIGINAL CODE MOVED)
001010  000000          FRSTAD: .WORD   0               ;CONTAINS FIRST ADRS OF CODE TO BE MOVED
001012  000000          FRSTMEM: .WORD  0               ;CONTAINS LOWER RELOCATION BOUNDARY ADDRESS

                                ;SCOPE (EMT) SERVICE ROUTINE
                                ;THIS ROUTINE ALLOWS THE SUBTEST TO BE CONTINUOUSLY LOOPED, ITERATED
                                ;(OR NOT ITERATED) BEFORE BEGINNING NEXT SUBTEST
001014  122737  000010  000764 SCOPEA: CMPB    #10,@#OPT,CP
001022  001005                  BNE     10$
```



```
001024  005037  177766                  CLR     @#CRUERR
001030  012737  177777  177744          MOV     #-1,@#ERRREG
001036  032766  004000  000002 10$:     BIT     #4000,2(SP)     ;WAS REGISTER SET BIT SET
001044  001403                          BEQ     1$              ;BRANCH IF NOT
001046  052737  004000  177776          BIS     #4000,@#PSW     ;RETAIN REGISTER SET
001054  032737  040000  177570 1$:      BIT     #40000,@#SWR    ;CHECK BIT 14 (CONTINUOUS LOOP)
001062  001416                          BEQ     4$
001064  010116                  2$:     MOV     R1,(SP)         ;LOAD RETURN ADDRESS
001066  010137  001006                  MOV     R1,@#RELR1
001072  163737  001004  001006          SUB     @#FACTOR,@#RELR1;RELR1 CONTAINS UNRELOCATED R1
001100  032737  000400  177570          BIT     #400,@#SWR      ;LOAD PDP11/45 MICRO BREAK REG?
001106  001403                          BEQ     3$
001110  113737  177570  177770          MOVB    @#SWR,@#UBREAK  ;LOAD MICRO BREAK REG WITH SR0-7
001116  000002                  3$:     RTI                     ;RETURN TO SUBTEST
001120  032737  004000  177570 4$:      BIT     #4000,@#SWR     ;SUBTEST ITERATION DESIRED?
001126  001006                          BNE     7$              ;BRANCH IF NO ITERATION DESIRED?
001130  005327                          DEC     (PC)+           ;DECREMENT SUBTEST ITERATION COUNT
001132  000040                  5$:     40                      ;CONTAINS SUBTEST ITERATION COUNT
001134  001353                          BNE     2$
001136  113767  001150  177766          MOVB    @#ITCNT,5$      ;RESET ITERATION COUNT
001144  011601                  7$:     MOV     (SP),R1         ;GET ADDRESS OF NEXT TEST
001146  000746                          BR      2$
001150  000040          ITCNT:  .WORD   40
```

```
                                        ;SBTTL  RELOC ROUTINE
                                ;ROUTINE TO RELOCATE PROGRAM CODE
001152  032737  010000  177570  RELOC:  BIT     #10000,@#SWR    ;BRANCH IF SW12=0
001160  001404                          BEQ     20$             ;SW12=1 & SW09=0 = NO RELOCATION
001162  032737  001000  177570          BIT     #1000,@#SWR     ;BRANCH IF SW09=0
001170  001465                          BEQ     4$              ;NO RELOCATION IF S12=1 & SW09=0
001172  105737  000770          20$:    TSTB    @#MMON          ;BRANCH IF MEM MGMT IS ENABLED
001176  001062                          BNE     4$              ;NO RELOCATION IF MEM MGMT IS ON
001200  013700  001010                  MOV     @#FRSTAD,R0     ;GET FIRST ADDRESS OF CODE TO BE MOVED
001204  010005                          MOV     R0,R5           ;SAVE
001206  010204                          MOV     R2,R4           ;GET LAST ADDRESS OF CODE TO BE MOVED
001210  160504                          SUB     R5,R4           ;R4 CONTAINS # OF BYTES TO RELOCATE
001212  010203                          MOV     R2,R3           ;SAVE LAST ADDRESS OF CODE TO BE MOVED
001214  005737  001004                  TST     @#FACTOR        ;FIRST RELOCATION IS TO ENDTAG+2
001220  001004                          BNE     10$
001222  010237  001360                  MOV     R2,@#RETPC      ;SAVE RETURN PC TO NEXT SECTION OF CODE
001226  013702  001012                  MOV     @#FRSTMEM,R2    ;SET FIRST ADDDRESS
001232  060204          10$:    ADD     R2,R4           ;R4 CONTAINS LAST MEMORY ADDRESS
001234  020437  025504                  CMP     R4,@#LSTMEM     ;EXIT IF INSUFFICIENT MEMORY
001240  101046                          BHI     5$              ;AVAILABLE FOR RELOCATION
001242  160204                          SUB     R2,R4           ;R4 NOW CONTAINS BYTE COUNT
001244  005037  001004                  CLR     @#FACTOR        ;CLEAR RELOCATION FACTOR
001250  032737  000040  177570  11$:    BIT     #40,@#SWR       ;CHECK IF ALL DEVICES DESIRED FOR
001256  001007                          BNE     12$             ;RELOCATION ROUND ROBIN STYLE
001260  032737  000010  177570          BIT     #10,@#SWR       ;CHECK IF A DEVICE IS SPECIFIED
001266  001410                          BEQ     1$
001270  113737  177570  000757          MOVB    @#SWR,@#DEV     ;GET SELECTED DEVICE
001276  005037  000762          12$:    CLR     @#EABITS        ;CLEAR EABITS FOR DEVICE
001302  004767  000112                  JSR     PC,IODEV        ;GO RELOCATE VIA SELECTED DEVICE
001306  102003                          BVC     2$              ;V= 0/1 INDICATES NO ERROR/ERROR
001310  012022          1$:     MOV     (R0)+,(R2)+     ;RELOCATE PROGRAM CODE
001312  020003                          CMP     R0,R3           ;CHECK IF DONE
001314  001375                          BNE     1$
001316  024042          2$:     CMP     -(R0),-(R2)     ;CHECK THAT CODE WAS RELOCATED
001320  001403                          BEQ     3$              ;PROPERLY
001322  004767  001154                  JSR     PC,SAVVAL       ;GO SAVE PERTINENT DATA FOR TYPEOUT
001326  104400                          HLT                     ;ERROR! CODE NOT RELOCATED PROPERLY
001330  020005          3$:     CMP     R0,R5           ;CHECK IF FINISHED CHECKING
001332  001371                          BNE     2$
001334  162737  000010  000772          SUB     #10,@#DEVID     ;BRANCH IF ERROR DETECTED ON RELOCATION
001342  001742                          BEQ     11$
001344  105237  000757          4$:     INCB    @#DEV           ;STEP TO NEXT DEVICE
001350  005037  000772                  CLR     @#DEVID         ;SET DEVICE IND TO CP
001354  010207                          MOV     R2,PC           ;GO EXECUTE RELOCATED CODE
001356  011707          5$:     MOV     (PC)+,PC        ;RETURN TO NEXT SECTION OF CODE
001360  000000  RETPC:  0               ;CONTAINS PC OF NEXT SECTION OF CODE

                                ;WAIT LOOP FOR COMPLETION OF DEVICE TRANSFERS
001362  013704          WAITIO: MOV     @(PC)+,R4               ;GET CONTENTS OF DEVICE'S BUS
001364  000000          BUSADR: .WORD   0                       ;ADDRESS REGISTER
001366  105737  000770          TSTB    @#MMON                  ;BRANCH IF MEM MGMT IS NOT ON
001372  001404                  BEQ     1$
001374  042704  160000          BIC     #160000,R4              ;CONVERT ADDRESS TO VIRTUAL ADRS
001400  052704  040000          BIS     #040000,R4
```



```
001404  024414          1$:     CMP     -(R4),(R4)
001406  001401                  BEQ     2$
001410  104400                  HLT
001412  062714  000002          2$:     ADD     #2,(R4)
001416  000761                  BR      WAITIO
```

```
                                        .SBTTL  IODEV ROUTINE
                                ;ROUTINE TO RELOCATE PROGRAM CODE VIA DEVICE SELECTED IN SWITCHES<2-0>
                                ;(IF SW03=1) OR VIA ALL DEVICES IF SW05=1,
                                ;THIS ROUTINE WRITES THE DATA TO BE RELOCATED ONTO THE SELECTED
                                ;DEVICE AND AFTER COMPLETION READS THE DATA BACK INTO MEMORY WHERE
                                ;THE RELOCTED DATA IS TO GO, AFTER THE READ THE ROUTINE RETURNS TO
                                ;THE CALLER, THE CALLER COMPARES THE DATA READ BACK.
                                ;DEVICES ARE:
                                ;       0-CP                            ;TAKES ERROR EXIT
                                ;       1-RK
                                ;       2-RF
                                ;       3-RP
                                ;       4-RC
                                ;       5-DO NOT USE
                                ;       6-RS04
                                ;       7-CP                    ;RESERVED FOR FUTURE USE
                                ;INPUT PARAMETERS:
                                ;       R0                              ;BUS ADDRESS FOR WRITE
                                ;       R1                              ;DON'T CARE
                                ;       R2                              ;BUS ADDRESS FOR READ
                                ;       R3                              ;DON'T CARE
                                ;       R4                              ;BYTE COUNT
                                ;       R5                              ;DON'T CARE
                                ;       EABITS                          ;LOADED
                                ;       DEV                             ;DEVICE IDENTIFIER
                                ;OUTPUT
                                ;       R0                              ;UPDATED BY BYTE COUNT (IF NO ERROR)
                                ;       R1                              ;UNCHANGED
                                ;       R2                              ;UPDATED BY BYTE COUNT (IF NO ERROR)
                                ;       R3                              ;UNCHANGED
                                ;       R4                              ;CLOBBERED
                                ;       R5                              ;UNCHANGED
                                ;       EABITS                          ;UNCHANGED
                                ;       'V' BIT                         ;CLEAR/SET=NO ERROR/ERROR

001420  004767  001106          IODEV:  JSR     PC,CLRTBIT              ;CLEAR 'T' BIT & SAVE PSW
001424  010546                          MOV     R5,-(SP)                ;SAVE R5 ON THE STACK
001426  052737  000200  177776          BIS     #PRTY4,@#PSW            ;SET PRIORITY LEVEL 4
001434  142737  000370  000757  1$:     BICB    #370,@#DEV              ;LIMIT DEVICE SELECT CODE
001442  113705  000757                  MOVB    @#DEV,R5                ;GET SELECTED DEVICE
001446  006305                          ASL     R5                      ;FORM INDEX POINTER
001450  016505  002266                  MOV     DEVTBL(R5),R5           ;GET SELECTED DEVICE TABLE
001454  001005                          BNE     2$                      ;BRANCH IF I/O DEVICE SELECTED

001456                          99$:                                    ;ERROR EXIT
001456  004737  002556                  JSR     PC,@#RESTPS             ;RESORE ORIGINAL PSW
001462  012605                          MOV     (SP)+,R5                ;RESTORE R5
001464  000262                          SEV                             ;SET 'V' BIT TO INDICATE FAILURE
001466  000207                  10$:    RTS     PC                      ;RETURN

                                ;CHECK IF USER SELECTED DEVICE IS AVAILABLE
001470  012737  000006  000004  2$:     MOV     #ERRVEC+2,@#ERRVEC      ;SET TIME OUT TRAP VECTOR
001476  000261                          SEC                             ;SET 'C' IN PSW
```



```
001500  005775  000012                  TST     @12(R5)                 ;REFERENCE A DEVICE REG
001504  012737  005274  000004          MOV     #ERRRT,@#ERRVEC         ;RESTORE ERROR TRAP VECTOR
                                                                        ;NOTE: MOV DOES NOT AFFECT 'C'
001512  103020                          BCC     20$                     ;EVER ONWARD IF IT'S THERE
001514  032737  000040  177570          BIT     #40,@#SWR               ;DO NOT REPORT ERROR IF ALL
001522  001403                          BEQ     14$
001524  105237  000757                  INCB    @#DEV                   ;STEP TO NEXT DEVICE
001530  000741                          BR      1$
001532  113705  000757          14$:    MOVB    @#DEV,R5                ;GET DEVICE ID
001536  006305                          ASL     R5                      ;FORM INDEX VALUE
001540  016567  004156  030602          MOV     DEVICE(R5),DEVNAM       ;GET HIS NAME
001546  000004  032346                  TYPE,NODEV
001552  000741                          BR      99$                     ;TAKE ERROR EXIT

001554  112737  000003  000760  20$:    MOVB    #3,@#IORETRY            ;SET ERROR RETRY COUNT
001562  010427                          MOV     R4,(PC)+                ;SAVE BYTE COUNT
001564  000000                  10$:    .WORD   0
001566  010446                          MOV     R4,-(SP)                ;FORM TWO'S COMPLEMENT
001570  006216                          ASR     (SP)                    ;WORD COUNT
001572  005416                          NEG     (SP)
001574  012667  000322                  MOV     (SP)+,9$                ;AND SAVE IN 9$ BELOW
001600  113737  000757  000772          MOVB    @#DEV,@#DEVID           ;SET DEVICE IDENT
001606  013727  000762                  MOV     @#EABITS,(PC)+          ;SAVE EABITS IN 11$ BELOW
001612  000000                  11$:    .WORD   0
001614  123727  000757  000004          CMPB    @#DEV,#4                ;BRANCH IF DEVICE IS NOT
001622  003415                          BLE     12$                     ;A MASS BUS DEVICE
001624  006367  177762                  ASL     11$                     ;SHIFT EA BITS TO
001630  006367  177756                  ASL     11$                     ;POSITION 8-9
001634  006367  177752                  ASL     11$                     ;FROM 4-5
001640  006367  177746                  ASL     11$
001644  012735  000021                  MOV     #21,@(R5)+              ;DO A READ IN (TO SET VOL VALID)
001650  012735  010000                  MOV     #10000,@(R5)+           ;SET PDP11 FORMAT (IN RPOF REG)
001654  012535                          MOV     (R5)+,@(R5)+            ;LOAD DEVICES UNIT #
001656  012546                  12$:    MOV     (R5)+,-(SP)             ;GET DEVICE'S VECTOR ADDRESS
001660  012776  001732  000000          MOV     #4$,@(SP)               ;LOAD VECTOR
001666  062716  000002                  ADD     #2,(SP)
001672  012736  000240                  MOV     #PRTY5,@(SP)+           ;AND PSW ON INTERRUPT
001676  004767  000222                  JSR     PC,DSKADR               ;GO GET RANDOM DISK ADDRESS
001702  016735  000306          3$:     MOV     CYLADR,@(R5)+           ;SET 'CYLINDER' ADDRESS
001706  016735  000310                  MOV     TRKSEC,@(R5)+           ;SET 'TRACK/SECTOR' ADDRESS
001712  011537  001364                  MOV     (R5),@#BUSADR           ;SAVE ADDRESS OF BUS ADDRESS REG
001716  010035                          MOV     R0,@(R5)+               ;SET BUS ADDRESS
001720  016735  000176                  MOV     9$,@(R5)+               ;SET WORD COUNT
001724  016535  000002                  MOV     2(R5),@(R5)+            ;AND SET COMMAND
001730  000614                  30$:    BR      WAITIO                  ;GO WAIT FOR WRITE TO FINISH

001732  012716  001740          4$:     MOV     #41$,(SP)               ;ADJUST RETURN PC TO 41$ BELOW
001736  000002                          RTI
001740  015504                  41$:    MOV     @-(R5),R4               ;GET AND CHECK ERROR BIT
001742  100011                          BPL     5$                      ;BRANCH IF NO ERROR
001744  104400                          HLT                             ;REPORT ERROR
001746  016535  000006                  MOV     6(R5),@(R5)+            ;RESET DEVICE'S CONTROLLER
001752  162705  000012                  SUB     #12,R5                  ;RESET TABLE POINTER
001756  105337  000760                  DECB    @#IORETRY               ;RETRY WRITE COMMAND
```

```
001762  001347                  BNE     3$
001764  000634          4$:     BR      99$                     ;TAKE ERROR EXIT
                                                                ;AFTER THREE RETRYS

001766  112737  000003  000760  5$:     MOVB    #3,@#IORETRY    ;RESET ERROR RETRY COUNT
001774  162705  000012          6$:     SUB     #12,R5          ;RESET TABLE POINTER
002000  012735  002046                  MOV     #7$,@(R5)+      ;RESET DEVICE'S INT VECTOR
002004  016735  000224                  MOV     CYLADR,@(R5)+   ;GET 'CYLINDER' ADDRESS
002010  016735  000206                  MOV     TRKSEC,@(R5)+   ;GET 'TRACK/SECTOR' ADDRESS
002014  011537  001364                  MOV     (R5),@#BUSADR   ;SAVE ADDRESS OF BUS ADDRESS REG
002020  010235                          MOV     R2,@(R5)+       ;SET BUS ADDRESS
002022  016735  000074                  MOV     9$,@(R5)+       ;SET WORD COUNT
002026  016746  177560                  MOV     11$,-(SP)       ;GET EA BITS
002032  050516  000004                  BIS     4(R5),(SP)      ;SET IN READ COMMAND
002036  012675  000000                  MOV     (SP)+,@(R5)     ;LOAD COMMAND
002042  000240                          NOP
002044  000731                          BR      30$             ;GO TO WAITIO VIA 30$

002046  012716  002054          7$:     MOV     #71$,(SP)       ;ADJUST RETURN PC TO 71$ BELOW
002052  000002                          RTI
002054  013504          71$:    MOV     @(R5)+,R4               ;GET & CHECK ERROR BIT IN COMMAND REG
002056  100007                  BPL     8$                      ;BRANCH IF NO ERROR
002060  104400                  HLT                             ;REPORT ERROR
002062  016555  000004                  MOV     4(R5),@-(R5)    ;RESET DEVICE'S CONTROLLER
002066  105337  000760                  DECB    @#IORETRY       ;RETRY READ COMMAND
002072  001340                  BNE     6$                      ;3 TIMES AND IF STILL FAILS
002074  000733                  BR      4$                      ;TAKE ERROR EXIT
002076  012605          8$:     MOV     (SP)+,R5                ;RESTORE R5
002100  066700  177460                  ADD     10$,R0          ;ADD BYTE COUNT TO WRITE AND
002104  066702  177454                  ADD     10$,R2          ;READ ADDRESSES (FOR CHECKING)
002110  004767  000442                  JSR     PC,RESTPS       ;GO RESTORE 'T' IN PSW
002114  000242                          CLV                     ;CLEAR ERROR INDICATOR
002116  000167  177344                  JMP     100$            ;EXIT

002122  000000          9$:     .WORD   0                       ;CONTAINS TWO'S COMP WORD COUNT

                                ;SUBROUTINE TO GENERATE RANDOM DSK SURFACE ADDRESSES
002124  032737  000020  177570  DSKADR: BIT     #20,@#SWR       ;BRANCH IF USER DOES NOT WANT
002132  001426                  BEQ     2$                      ;RANDOM DISK ADDRESSES
002134  010046                  MOV     R0,-(SP)                ;SAVE R0 ON THE STACK
002136  013700  000772                  MOV     @#DEVID,R0      ;GET I/O DEVICE ID
002142  006300                  ASL     R0                      ;FORM INDEX INTO
002144  006300                  ASL     R0                      ;ADRTAB BELOW
002146  060146                  ADD     R1,-(SP)                ;FORM RANDOM #
002150  005516                  ADC     (SP)
002152  011667  000036                  MOV     (SP)+,CYLADR    ;MOVE TO 'CYLINDER' ADDRESS
002156  046067  002226  000030          BIC     ADRTAB(R0),CYLADR       ;LIMIT 'CYLINDER' ADDRESS
002164  060116                  ADD     R1,(SP)
002166  005516                  ADC     (SP)
002170  012667  000026                  MOV     (SP)+,TRKSEC    ;MOVE TO 'TRACK/SECTOR' ADDRESS
002174  005720                  TST     (R0)+
002176  046067  002226  000016          BIC     ADRTAB(R0),TRKSEC       ;LIMIT 'TRACK/SEC' ADRS
002204  012600                  MOV     (SP)+,R0                ;RESTORE R0
002206  000207                  RTS     PC                      ;RETURN
```



```
002210  012727  000000          2$:     MOV     #0,(PC)+        ;SET CYLINDER ADDRESS = 0
002214  000000          CYLADR: .WORD   0
002216  012727  020000                  MOV     #0,(PC)+        ;SET TRACK & SECTOR = 0
002222  000000          TRKSEC: .WORD   0
002224  000207                  RTS     PC

                                ;TABLE OF DEVICE ;CYLINDER; AND ;TRACK/SECTOR; ADDRESS LIMITERS
002226  000000          ADRTAB: .WORD   0                       ;NOT USED
002230  000000                  .WORD   0                       ;NOT USED
002232  163350                  .WORD   163350                  ;RKDA LIMITER
002234  163350                  .WORD   163350                  ;RKDA LIMITER
002236  177774                  .WORD   177774                  ;RFDAE LIMITER
002240  020000                  .WORD   020000                  ;RFDAR LIMITER
002242  177152                  .WORD   177152                  ;RPCA LIMITER
002244  170370                  .WORD   170370                  ;RPDA LIMITER
002246  176400                  .WORD   176400                  ;RCDA LIMITER
002250  176400                  .WORD   176400                  ;RCDA LIMITER
002252  177145                  .WORD   177345                  ;RP4CA LIMITER
002254  170370                  .WORD   170370                  ;RP4DST LIMITER
002256  170400                  .WORD   170400                  ;RSDA LIMITER
002260  170400                  .WORD   170400                  ;RSDA LIMITER
002262  177777                  .WORD   -1                      ;NOT USED
002264  177777                  .WORD   -1                      ;NOT USED

                                .SBTTL  DEVICE TABLES
002266  000000          DEVTBL: .WORD   0
002270  002306                  .WORD   RKTBL
002272  002330                  .WORD   RFTBL
002274  002352                  .WORD   RPTBL
002276  002374                  .WORD   RCTBL
002300  000000                  .WORD   0                       ;RESERVED FOR RP04
002302  002450                  .WORD   RSTBL
002304  000000                  .WORD   0

002306  000220          RKTBL:  .WORD   RKVEC
002310  177412                  .WORD   RKDA
002312  177412                  .WORD   RKDA
002314  177410                  .WORD   RKBA
002316  177406                  .WORD   RKWC
002320  177404                  .WORD   RKCS
002322  000503                  .WORD   503                     ;WRITE COMMAND
002324  000505                  .WORD   505                     ;READ COMMAND
002326  000001                  .WORD   1                       ;CONTROL RESET

002330  000204          RFTBL:  .WORD   RFVEC
002332  177470                  .WORD   RFDAE
002334  177466                  .WORD   RFDAR
002336  177464                  .WORD   RFCMA
002340  177462                  .WORD   RFWC
002342  177460                  .WORD   RFDCS
002344  000103                  .WORD   103                     ;WRITE COMMAND
002346  000105                  .WORD   105                     ;READ COMMAND
002350  000001                  .WORD   1                       ;CONTROL RESET
```

```
                                RPTBL:  .WORD   RPVEC
002352  000254                          .WORD   RPCA
002354  176722                          .WORD   RPDA
002356  176724                          .WORD   RPBA
002360  176720                          .WORD   RPWC
002362  176716                          .WORD   RPCS
002364  176714                          .WORD   RPCS
002366  000103                          .WORD   103             ;WRITE COMMAND
002370  000105                          .WORD   105             ;READ COMMAND
002372  000001                          .WORD   1               ;CONTROL RESET

002374  000210                  RCTBL:  .WORD   RCVEC
002376  177442                          .WORD   RCDA
002400  177442                          .WORD   RCDA
002402  177452                          .WORD   RCCA
002404  177450                          .WORD   RCWC
002406  177446                          .WORD   RCCS
002410  000103                          .WORD   103             ;WRITE COMMAND
002412  000105                          .WORD   105             ;READ COMMAND
002414  000001                          .WORD   1               ;CONTROL RESET

                                ;RP04 TABLE
002416  176700                  RP4TBL: .WORD   RP4CS1
002420  176732                          .WORD   RP4OF
002422  000000                          .WORD   0               ;RP04 UNIT #
002424  000000                          .WORD   RHCS2           ;RHCS2 REGISTER ADDRESS
002426  000254                          .WORD   RP4VEC
002430  176734                          .WORD   RP4CA
002432  176706                          .WORD   RP4DST
002434  176704                          .WORD   RP4BA
002436  176702                          .WORD   RP4WC
002440  176700                          .WORD   RP4CS1
002442  000161                          .WORD   161             ;WRITE COMMAND
002444  000171                          .WORD   171             ;READ COMMAND
002446  040011                          .WORD   40011           ;DRIVE CLEAR

                                ;RS04 TABLE
002450  172040                  RSTBL:  .WORD   RSCS1
002452  172040                          .WORD   RSCS1
002454  000000                          .WORD   0               ;RS04 UNIT #
002456  000000                          .WORD   RHCS2           ;RHCS2 REGISTER ADDRESS
002460  000204                          .WORD   RSVEC
002462  172046                          .WORD   RSDA
002464  172046                          .WORD   RSDA
002466  172044                          .WORD   RSBA
002470  172042                          .WORD   RSWC
002472  172040                          .WORD   RSCS1
002474  000161                          .WORD   161
002476  000171                          .WORD   171
002500  040011                          .WORD   40011

                                ;ROUTINE TO SAVE MEMORY VALUES ON RELOCATION ERROR
002502  012737  000010  000772  SAVVAL: MOV     #10,@#DEVID             ;SET DEVICE IND=MEMORY
002510  010446                  MOV     R4,-(SP)                ;SAVE R4 ON THE STACK
002512  012704  000730          MOV     #MEMTBL,R4              ;GET STARTING ADDRESS OF TABLE
```



```
002516  010024                  MOV     R0,(R4)+                ;LOAD 'GOOD' ADDRESS
002520  011024                  MOV     (R0),(R4)+              ;LOAD 'GOOD' DATA
002522  010224                  MOV     R2,(R4)+                ;LOAD 'BAD' ADDRESS
002524  011274                  MOV     (R2),(R4)+              ;LOAD 'BAD' DATA
002526  012604                  MOV     (SP)+,R4                ;RESTORE R4
002530  000207                  RTS     PC                      ;EXIT

                                ;ROUTINE TO CLEAR 'T' BIT
002532  013746  177776  CLRTBIT: MOV     @#PSW,-(SP)     ;PUSH PSW ONTO STACK
002536  011627                  MOV     (SP)+,(PC)+     ;SAVE IN RETPSW BELOW
002540  000000          RETPSW: .WORD   0
002542  042716  000020          BIC     #20,(SP)        ;CLEAR 'T' BIT IN PSW ON STACK
002546  012746  002554  RESPSW: MOV     #1$,-(SP)       ;SET RETURN PC FOR RTI
002552  000002                  RTI                     ;CLEAR 'T' BIT IN PSW
002554  000207          1$:     RTS     PC              ;RETURN
002556  016746  177756  RESTPS: MOV     RETPSW,@(SP)    ;PUSH ORIG PSW ONTO STACK
002562  000771                  BR      RESPSW
```

```
                        ;SBTTL  TYPE SUBROUTINE
                        ;ROUTINE TO TYPE ASCII MESSAGE, MESSAGE MUST TERMINATE WITH A 0 BYTE,
002564  010046          ,TYPE:  MOV     R0,-(SP)        ;SAVE R0 ON THE STACK
002566  017600  000002          MOV     @2(SP),R0       ;GET MESSAGE ADDRESS
002572  062766  000002  000002  ADD     #2,2(SP)        ;ADJUST RETURN PC
002600  032737  000400  000764  BIT     #TTOPT,@#OPT,CP ;BRANCH IF NO CONSOLE TTY AVAIL
002606  001403                  BEQ     6$
                        ;
002610  112046          1$:     MOVB    (R0)+,-(SP)     ;PUSH CHAR ON THE STACK
002612  001003                  BNE     2$              ;BRANCH IF NOT TERMINATOR
002614  005726                  TST     (SP)+           ;POP TERMINATOR OFF THE STACK
002616  012600          6$:     MOV     (SP)+,R0        ;RESTORE R0
002620  000002                  RTI                     ;RETURN
                        ;
002622  004767  000026  2$:     JSR     PC,5$           ;TYPE CHARACTER
002626  122726  000012  3$:     CMPB    #12,(SP)+       ;CHECK IF CHAR WAS A LINE FEED
002632  001366                  BNE     1$              ;BRANCH IF NOT LINE FEED
                        ;
002634  016746  076142          MOV     $FILLS,-(SP)    ;GET # OF FILLERS REQUIRED AFTER
                                                        ;LINE FEED AND FILLER CHARACTER
002640  105366  000001  4$:     DECB    1(SP)           ;DECREMENT FILLERS COUNT
002644  002770                  BLT     3$              ;BRANCH IF NO MORE FILLERS NEEDED
002646  004767  000002          JSR     PC,5$           ;TYPE FILLER CHARACTER
002652  000772                  BR      4$
                        ;
002654  105737  177564  5$:     TSTB    @#TPS           ;WAIT FOR OUTPUT DEVICE
002660  100375                  BPL     .-4             ;TO BECOME READY
002662  116637  000002  177566  MOVB    2(SP),@#TPB     ;TYPE CHARACTER
002670  000207                  RTS     PC
                        ;
000000                  NULL=0
                        ;
                        ;SUBROUTINE TO CONVERT 16 BIT DATA TO ASCII STRING, THE ASCII STRING
                        ;STARTS AT DIGITS AND IS 8 BYTES LONG, 6 ASCII DIGITS + 'SPACE' + '0',
                        ;
002672                  CNVDAT:
002672  004767  002234          JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
002676  012704  003102          MOV     #DIGBUF+8.,R4           ;SET ADDRESS OF DIGIT BUFFER
002702  010201                  MOV     R2,R1                   ;GET DATA
002704  005003                  CLR     R3
002706  012700  000006          MOV     #6.,R0                  ;SET DIGIT COUNT
002712  000167  000100          JMP     CNVDIG                  ;GO TO DIGIT CONVERSION ROUTINE
                        ;
                        ;SUBROUTINE TO CONVERT A VIRTUAL ADDRESS TO AN ASIZ STRING PHYSICAL
                        ;ADDRESS, THE CONVERTED ASCII STRING IS AT 'DIGBUF' AND IS 10 BYTES LONG
                        ;(8 DIGITS + 1 SPACE + 0 BYTE)
                        ;CALL:  MOV     ADDRESS,R1              ;GET ADDRESS
                        ;       JSR     PC,CNVADR
                        ;NOTE:  SUBROUTINE SUBRACTS 2 FROM ADDRESS BEFORE CONVERSION
                        ;FOR EXAMPLE TO TYPE ERROR PC
                        ;       MOV     PC,R1           ;IT IS THE PC OF THE MOV
                        ;       JSR     PC,CNVADR       ;THAT GETS TYPED
                        ;       TYPE
                        ;       DIGBUF
```



```
                        ;
002716                  CNVADR:
002716  004767  002202          JSR     PC,$SAVR        ;GO SAVE REGISTERS ON THE STACK
002722  012704  003102          MOV     #DIGBUF+8.,R4           ;GET ADDRESS OF DIGIT BUFFER
002726  162701  000002          SUB     #2,R1                   ;SUBTRACT 2 FROM ADDRESS
002732  010105                  MOV     R1,R5                   ;SAVE ADDRESS TO BE CONVERTED
002734  005003                  CLR     R3
002736  105737  000770          TSTB    @#MMON                  ;BRANCH IF MEM MGMT IS DISABLED
002742  001423                  BEQ     3$
002744  042701  017777          BIC     #17777,R1               ;CLEAR ALL BUT PAR SELECTOR BITS
002750  006301                  ASL     R1                      ;SHIFT BITS 15-13 OF ADDRESS
002752  006101                  ROL     R1                      ;LEFT TO
002754  006101                  ROL     R1                      ;3-1
002756  006101                  ROL     R1
002760  006301                  ASL     R1
002762  062701  172340          ADD     #KIPAR0,R1              ;FORM ADDRESS OF PAR REG
002766  011101                  MOV     (R1),R1                 ;GET CONTENTS OF PAR
002770  012700  000006          MOV     #6,R0                   ;SET SHIFT COUNTER
002774  006301          2$:     ASL     R1                      ;SHIFT PAR BITS IN R1
002776  006103                  ROL     R3                      ;6 PLACES LEFT TO R3-R1
003000  077003                  SOB     R0,2$
003002  042705  160000          BIC     #160000,R5              ;CLEAR PAR SELECTOR BITS IN ADDRESS
003006  060501                  ADD     R5,R1                   ;FORM PHYSICAL ADDRESS
003010  005503                  ADC     R3                      ;IN R1 & R3
003012  012700  000010  3$:     MOV     #8.,R0                  ;SET DIGIT COUNT
                        ;
003016  012705  000003  CNVDIG: MOV     #3,R5                   ;AND BITS PER DIGIT COUNT
003022  005002                  CLR     R2                      ;R2 WILL CONTAIN DIGIT
003024  006203          5$:     ASR     R3                      ;R3<00> TO 'C'
003026  006001                  ROR     R1                      ;'C' TO R1<15> & R1<00> TO 'C'
003030  106002                  RORB    R2                      ;'C' TO R2<07>
003032  005305                  DEC     R5                      ;DECREMENT SHIFT COUNT
003034  001373                  BNE     5$
003036  012705  000005          MOV     #5,R5                   ;SET SHIFT COUNT
003042  000241          6$:     CLC
003044  106002                  RORB    R2                      ;SHIFT DIGIT FROM <07-05>
003046  005305                  DEC     R5                      ;TO <02-00>
003050  001374                  BNE     6$
                        ;
003052  062702  003260          ADD     #260,R2                 ;CONVERT DIGIT TO ASCII
003056  110244                  MOVB    R2,-(R4)                ;MOVE DIGIT INTO DIGIT BUFFER
003060  005300                  DEC     R0                      ;DECREMENT DEGIT COUNT
003062  001355                  BNE     CNVDIG                  ;CONVERT NEXT DIGIT
003064  004767  002054          JSR     PC,$RESTR               ;RESTORE REGISTERS FROM STACK
003070  000207                  RTS     PC
                        ;
                        ;DIGIT BUFFER
003072    000     000   DIGBUF: .BYTE   0,0
003074  000006  DIGITS: .BLKB   6.
003102    040           .BYTE   40                      ;'SPACE'
003103    000           .BYTE   0                       ;'0' TERMINATOR
                        ;
                        ;SUBROUTINE TO CONVERT 16 BIT OCTAL DATA TO AN ASCII STRING AND TYPE IT.
                        ;CALL:  MOV     #DATA,R2                ;LOAD R2 WITH THE DATA
```

```
                                ;       JSR     PC,TYPDAT

                                TYPDAT: JSR     PC,CNVDAT               ;CONVERT DATA TO ASCII STRING
003110  000004  003074                  TYPE,DIGITS
003114  000207                          RTS     PC

                                ;SUBROUTINE TO CONVERT A VIRTUAL ADDRESS TO A PHYSICAL ADDRESS AND TYPE IT.
                                ;CALL:  MOV     #ADDRESS,R1             ;LOAD R1 WITH THE ADDRESS
                                ;       JSR     PC,TYPADR

003116  004767  177574          TYPADR: JSR     PC,CNVADR               ;CONVERT ADDRESS TO ASCII STRING
003122  000004  003072                  TYPE,DIGBUF                     ;TYPE ADDRESS
003126  000207                          RTS     PC

                                ;KEYBOARD INTERRUPT SERVICE ROUTINE

        000003                  CNTRLC=3

003130  000240                  TKISR:  NOP
003132  013746  177562                  MOV     @#TKB,-(SP)             ;GET CHARACTER
003136  042716  177600                  BIC     #177600,(SP)            ;STRIP UNUSED BITS
003142  022716  000003                  CMP     #CNTRLC,(SP)            ;BRANCH IF NOT CONTROL C (^C)
003146  001005                          BNE     1$
003150  000004  000752                  TYPE,CRLF                       ;ECHO <CR><LF>
003154  005726                          TST     (SP)+                   ;POP CHARACTER OFF THE STACK
003156  000000                          HALT
003160  000002                          RTI                             ;RETURN

003162  122716  000015          1$:     CMPB    #15,(SP)                ;BRANCH IF NOT <CR>
003166  001004                          BNE     2$
003170  000004  000752                  TYPE,CRLF                       ;ECHO <CR><LF>
003174  005726                          TST     (SP)+                   ;POP CHARACTEROFF STACK
003176  000002                          RTI                             ;RETURN

003200  112667  175534          2$:     MOVB    (SP)+,ECHO
003204  000004  000740                  TYPE,ECHO                       ;ECHO CHARACTER
003210  000002                          RTI                             ;RETURN
```



```
                                        .SBTTL  ERROR SERVICE ROUTINE
                                ;ERROR SERVICE CALLED BY TRAP (HLT) INSTRUCTION
003212  005737  177570          .HLT:   TST     @#SWR                   ;HALT ON ERROR?
003216  100001                          BPL     .+4
003220  000000                          HALT                            ;ERROR PC IS TOP WORD ON STACK
003222  032737  020000  177570          BIT     #20000,@#SWR            ;TYPE OUT DESIRED?
003230  001117                          BNE     1$                      ;BRANCH IF NO TYPEOUT
003232  004767  001666                  JSR     PC,$SAVR                ;GO SAVE REGISTERS ON THE STACK
003236  013702  001000                  MOV     @#IGNT,R2               ;GET PASS COUNT
003242  004767  177424                  JSR     PC,CNVDAT
003246  016767  177624  000320          MOV     DIGITS+2,PASSES         ;LOAD ASCII VALUES
003254  016767  177620  000314          MOV     DIGITS+4,PASSES+2
003262  000004  003564                  TYPE,PASCNT
003266  016602  000016                  MOV     16(SP),R2               ;GET PC OF ERROR CALL
003272  124242                          CMPB    -(R2),-(R2)             ;DECREMENT PC TO HLT
003274  000004  003601                  TYPE,VIRPC
003300  004767  177600                  JSR     PC,TYPDAT               ;TYPE DATA
003304  016702  175462                  MOV     DEVID,R2                ;GET DEVICE IDENTIFICATION
003310  001411                          BEQ     13$                     ;AND BRANCH IF DEVICE WAS CP
003312  006302                          ASL     R2
003314  016267  004156  175420          MOV     DEVICE(R2),DEVERR
003322  000004  000742                  TYPE,DEVERR
003326  004767  000424                  JSR     PC,RNTREGS
003332  000454                          BR      19$
003334  000004  003607          13$:    TYPE,STATUS
003340  016602  000020                  MOV     20(SP),R2               ;GET STATUS AT TIME OF ERROR
003344  004767  177534                  JSR     PC,TYPDAT               ;TYPE STATUS
003350  122737  000010  000764          CMPB    #10,@#OPT,CP
003356  001014                          BNE     12$
003360  000004  003614                  TYPE,CPERR
003364  013702  177766                  MOV     @#CPUERR,R2
003370  004767  177510                  JSR     PC,TYPDAT
003374  000004  003621                  TYPE,ERREG
003400  013702  177744                  MOV     @#ERRREG,R2
003404  004767  177474                  JSR     PC,TYPDAT
003410  016602  000016          12$:    MOV     16(SP),R2               ;GET PC OF ERROR
003414  124242                          CMPB    -(R2),-(R2)
003416  105737  000770                  TSTB    @#MMON                  ;CHECK IF MEM MGMT IS ENABLED
003422  001012                          BNE     10$                     ;BRANCH IF ENABLED
003424  005737  001004                  TST     @#FACTOR
003430  001415                          BEQ     19$


003432  000004  003626                  TYPE,RELPC
003436  163702  001004                  SUB     @#FACTOR,R2             ;FORM PC OF ORIGINAL CODE
003442  004767  177436                  JSR     PC,TYPDAT               ;TYPE DATA
003446  000406                          BR      19$                     ;GO TO 19$
003450  000004  003633          10$:    TYPE,PHYSPC
003454  016601  000016                  MOV     16(SP),R1               ;GET ERROR PC
003460  004767  177432                  JSR     PC,TYPADR               ;TYPE ADDRESS
003464                          19$:
003464  004767  001454                  JSR     PC,$RESTR               ;RESTORE REGISTERS FROM STACK
003470  032737  002000  177570  1$:     BIT     #2000,@#SWR             ;RING BELL ON ERROR
003476  001402                          BEQ     2$
```

```
                                        TYPE,BELL
                                2$:     TST     @#SWR           ;HALT AFTER TYPEOUT
                                        BPL     .+4
                                        HALT
                                        TST     @#ERFLAG
                                        BEQ     4$
                                        CLR     @#ERFLAG
                                        CLR     R0              ;ALLOW TIME FOR TTY TO TYPE CHAR
                                3$:     DEC     R0
                                        BNE     3$
                                        JMP     @#START
                                4$:     TSTB    @#PEFLG         ;BRANCH IF NO PARITY ERROR
                                        BEQ     5$
                                        JMP     @#PERET         ;RETURN TO PARITY ERROR SERVICE
                                5$:     RTI

                                ;DIGIT TABLE
                                DIGTAB: "01
                                        "23
                                        "45
                                        "67
                                PASCNT: .ASCII  <15><12>'PASS #'

                                ;NOTE:   PASSES MUST BE AT AN EVEN ADDRESS!

                                PASSES: .ASCIZ  '0000'
                                VIRPC:  .ASCIZ  ' VPC='
                                STATUS: .ASCIZ  'PSW='
                                CPERR:  .ASCIZ  'CPU='
                                ERREG:  .ASCIZ  'ERR='
                                RELPC:  .ASCIZ  'RPC='
                                PHYSPC: .ASCIZ  'PPC='
                                BELL:   .ASCIZ  <7>
                                        .EVEN
                                SUCCESS: .ASCII  <15><12>
                                        .ASCII  / THE QUICK BROWN FOX JUMPS OVER THE LAZY DOGS BACK #123456789 PASS# /
                                PASSNO: .ASCIZ  '0000'
                                        .EVEN
                                ;ROUTINE TO TYPE CONTENTS OF DEVICE REGISTER ON AN ERROR
                                ;INPUT:
                                ;       R2                              ;INDEX VALUE TO APPROPRIATE DEV
```



```
                                PNTREGS: MOV    REGS(2),R0              ;GET # OF REGS TO TYPE
                                        MOV     REGADR(2),R3            ;GET FIRST ADDRESS OF DATA TABLE
                                        CMP     #HEMTBL,R3              ;BRANCH IF MEMORY ERROR
                                        BEQ     2$
                                1$:     MOV     (R3)+,R2
                                        JSR     PC,TYPDAT               ;TYPE DATA
                                        DEC     R0
                                        BNE     1$
                                        RTS     PC

                                2$:     TYPE,GDADR
                                        MOV     (R3)+,R1                ;GET 'FROM' ADDRESS
                                        TST     (R1)+                   ;ADD 2
                                        JSR     PC,TYPADR               ;TYPE ADDRESS
                                        TYPE,A,DATA
                                        MOV     (R3)+,R2                ;GET 'FROM' DATA
                                        JSR     PC,TYPDAT               ;TYPE DATA
                                        TYPE,BDADR
                                        MOV     (R3)+,R1                ;GET 'TO' ADDRESS
                                        TST     (R1)+                   ;ADD 2
                                        JSR     PC,TYPADR               ;TYPE ADDRESS
                                        TYPE,A,DATA
                                        MOV     (R3)+,R2                ;GET 'TO' DATA
                                        JSR     PC,TYPDAT               ;TYPE DATA
                                        RTS     PC

                                GDADR:  .ASCII  'ERROR ON PROGRAM RELOCATION'<15><12>

                                A,DATA: .ASCIZ  'DATA='
                                BDADR:  .ASCIZ  'BAD ADRS='

                                        .EVEN

                                DEVICE: .ASCII  '00'
                                        .ASCII  'RK'
                                        .ASCII  'RF'
                                        .ASCII  'RP'
                                        .ASCII  'RC'
                                        .ASCII  'RP'
                                        .ASCII  'RS'
                                        .ASCII  'XX'                    ;RESERVED FOR FUTURE USE
                                        .ASCII  'MM'                    ;MEMORY

                                ;THE BELOW TABLE CONTAINS THE # OF DEVICE REGISTERS TO TYPE ON A
                                ;DEVICE ERROR
                                REGS:   .WORD   1                       ;NOT USED (FOR CP)
                                        .WORD   6                       ;TYPE 6 RK REGISTERS
                                        .WORD   6                       ;TYPE 6 RF REGISTERS
                                        .WORD   8.                      ;TYPE 8, RP REGISTERS
                                        .WORD   6                       ;TYPE 6 RC REGISTERS
```

```
1069  004212  000011                  .WORD   9,                      ;TYPE 9 RP04 REGISTERS
1070  004214  000011                  .WORD   9,
1071  004216  000001                  .WORD   1
1072  004220  000004                  .WORD   4
1073
1074  004222  000000          REGADR: .WORD   0
1075  004224  177400                  .WORD   RKDS
1076  004226  177460                  .WORD   RFDCS
1077  004230  176710                  .WORD   RPDS
1078  004232  177440                  .WORD   RCLA
1079  004234  176700                  .WORD   RP4CS1
1080  004236  172040                  .WORD   RSCS1
1081  004240  000000                  .WORD   0
1082  004242  000730                  .WORD   MEMTBL
1083
1084
1085                                  ;ROUTINE TO GET TYPED OCTAL ADDRESS AND CONVERT TO OCTAL, CALL:
1086                                  ;       JSR     R5,RECO
1087                                  ;       .WORD   0               ;CONVERTED DATA IS PLACED HERE
1088  004244  010046          RECO:   MOV     R0,-(SP)        ;SAVE R0 ON THE STACK
1089  004246  005015                  CLR     (R5)            ;CLEAR OLD DATA
1090  004250  105737  177560  1$:     TSTB    @#TKS           ;WAIT FOR USER TO INPUT CHARACTER
1091  004254  100375                  BPL     1$
1092  004256  113700  177562          MOVB    @#TKB,R0        ;GET CHARACTER
1093  004262  042700  000200          BIC     #200,R0         ;STRIP MSB
1094  004266  122700  000177          CMPB    #177,R0         ;CHECK IF RUBOUT
1095  004272  001027                  BNE     2$              ;BRANCH IF NOT RUBOUT
1096  004274  000004  000755          TYPE,SLASH              ;ECHO SLASH
1097  004300  000241                  CLC                     ;CLEAR CARRY
1098  004302  006015                  ROR     (R5)            ;SHIFT LAST TYPED CHARACTER
1099  004304  006215                  ASR     (R5)            ;OUT OF DATA WORD
1100  004306  006215                  ASR     (R5)
1101  004310  000757                  BR      1$              ;GO WAIT FOR NEXT CHARACTER
1102
1103  004312  122700  000015  2$:     CMPB    #15,R0          ;CHECK IF CARRIAGE RETURN
1104  004316  001004                  BNE     3$              ;BRANCH IF NOT CARRIAGE RETURN
1105  004320  000004  000752          TYPE,CRLF
1106  004324  005725                  TST     (R5)+           ;STEP RETURN ADDRESS
1107  004326  000205                  RTS     R5              ;RETURN
1108
1109  004330  110067  174404  3$:     MOVB    R0,ECHO
1110  004334  000004  000740          TYPE,ECHO
1111  004340  042700  177770          BIC     #177770,R0      ;STRIP NON-ESSENTIAL BITS
1112  004344  006315                  ASL     (R5)            ;SHIFT LAST CHARACTER 3 PLACES
1113  004346  006315                  ASL     (R5)            ;LEFT
1114  004350  006315                  ASL     (R5)
1115  004352  050015                  BIS     R0,(R5)         ;AND INSERT NEW CHARACTER
1116  004354  000735                  BR      1$              ;WAIT FOR NEXT CHARACTER
1117
```



```
1118                                  .SBTTL  PARITY ERROR SERVICE
1119                                  ;PARITY ERROR SERVICE ROUTINE
1120  004356  005737  177570  ,PARSRV:TST     @#SWR           ;CHECK IF HALT ON ERROR
1121  004362  100001                  BPL     1$              ;BRANCH IF NOT HALT ON ERROR
1122  004364  000000                  HALT
1123  004366  000004  000707  1$:     TYPE,PARERR
1124  004372  110037  000761          MOVB    SP,@#PEFLG      ;SET PARITY ERROR INDICATOR
1125  004376  000137  003212          JMP     @#.HLT          ;GO TO ERROR SERVICE
1126  004402  105037  000761  PERET:  CLRB    @#PEFLG         ;CLEAR PARITY ERROR FLAG
1127  004406  005001                  CLR     R1
1128  004410  005737  000764          TST     @#0PT,CP        ;CHECK IF MEM MGMT IS AVAIL
1129  004414  100032                  BPL     1$              ;BRANCH IF NOT AVAILABLE
1130  004416  012702  077406          MOV     #77406,R2       ;SET UP MEM MGMT
1131  004422  005037  172340          CLR     @#KIPAR0
1132  004426  010237  172300          MOV     R2,@#KIPDR0
1133  004432  012737  000200  172342  MOV     #200,@#KIPAR1
1134  004440  010237  172302          MOV     R2,@#KIPDR1
1135  004444  012737  000400  172344  MOV     #400,@#KIPAR2
1136  004452  010237  172304          MOV     R2,@#KIPDR2
1137  004456  005037  172306          CLR     @#KIPDR3
1138  004462  012737  007600  172356  MOV     #7600,@#KIPAR7
1139  004470  010237  172316          MOV     R2,@#KIPDR7
1140  004474  012737  000001  177572  MOV     #1,@#SR0        ;ENABLE MEM MGMT
1141  004502  012737  004530  000114  1$:     MOV     #2$,@#PARVEC    ;SET MEM PARITY ERROR TRAP VECTOR
1142  004510  012737  004710  000004          MOV     #7$,@#ERRVEC    ;SET TIME OUT TRAP
1143  004516  012737  004722  000250          MOV     #8$,@#MMVEC     ;SET MEM MGMT ABORT VECTOR
1144
1145  004524  005721                  TST     (R1)+           ;SCAN MEMORY FOR PARITY ERROR
1146  004526  000776                  BR      .-2
1147
1148  004530  000004  004744  2$:     TYPE,ADRS1S
1149  004534  004767  176356          JSR     PC,TYPADR               ;TYPE ADDRESS
1150  004540  000005                  RESET                   ;DISABLE PARITY ERROR DETECTION & MEM MGMT
1151  004542  005737  000764          TST     @#0PT,CP        ;BRANCH IF MEM MGMT NOT AVAILABLE
1152  004546  100002                  BPL     3$
1153  004550  005237  177572          INC     @#SR0           ;RE-ENABLE MEM MGMT
1154  004554  005002          3$:     CLR     R2              ;INITIALIZE DATA FOR DATA SCAN
1155  004556  014103                  MOV     -(R3),R3        ;GET DATA IN FAILING ADDRESS
1156  004560  010211          4$:     MOV     R2,@R1)         ;LOAD BINARY COUNT INTO ADDRESS
1157  004562  021102                  CMP     (R1)+,R2        ;BRANCH IF DATA DOES NOT COMPARE
1158  004564  001016                  BNE     5$
1159  004566  005102                  COM     R2              ;COMPLEMENT DATA
1160  004570  010211                  MOV     R2,(R1)         ;LOAD COMPLEMENT DATA INTO FAILING ADDRESS
1161  004572  021102                  CMP     (R1)+,R2        ;BRANCH IF DATA DOES NOT COMPARE
1162  004574  001012                  BNE     5$
1163  004576  005402                  NEG     R2              ;STEP DATA
1164  004600  001367                  BNE     4$
1165  004602  000004  004771          TYPE,NOTFND             ;TYPE PARITY ERROR NOT FOUND ON
1166  004606  000004  005046          TYPE,DSCAN              ;DATA SCAN ORIG DATA =
1167  004612  010302                  MOV     R3,R2           ;GET ORIGINAL DATA
1168  004614  004767  176264          JSR     PC,TYPDAT       ;TYPE ORIGINAL DATA
1169  004620  000411                  BR      6$              ;EXIT VIA 6$
1170
1171  004622  000004  005075  5$:     TYPE,GODAT              ;TYPE GOOD DATA =
```

```
1172  004626  004767  176252          JSR     PC,TYPDAT       ;AND THE GOOD DATA
1173  004632  000004  005110          TYPE,BDDAT              ;TYPE BAD DATA =
1174  004636  011102                  MOV     (R1),R2         ;GET BAD DATA
1175  004640  004767  176240          JSR     PC,TYPDAT       ;TYPE BAD DATA
1176
1177  004644  000004  000752  6$:     TYPE,CRLF
1178  004650  005737  177570          TST     @#SWR                   ;CHECK FOR HALT ON ERROR
1179  004654  100001                  BPL     .+4
1180  004656  000000                  HALT
1181  004660  000005                  RESET                           ;DISABLE MEM MGMT & PARITY
1182  004662  012737  004356  000114  MOV     #,PARSRV,@#PARVEC       ;RESET PARITY ERROR TRAP
1183  004670  012737  005274  000004  MOV     #ERRRT,@#ERRVEC         ;AND ERROR VECTOR
1184  004676  012737  000252  000250  MOV     #MMVEC+2,@#MMVEC        ;RESET MEM MGMT ABORT TRAP
1185  004704  000137  005600          JMP     @#START                 ;RESTART TEST
1186
1187  004710  000004  004771  7$:     TYPE,NOTFND
1188  004714  000004  005031          TYPE,ASCAN
1189  004720  000751                  BR      6$
1190
1191                                  ;MEMORY MANAGEMENT ABORT ROUTINE
1192  004722  062737  000200  172344  8$:     ADD     #200,@#KIPAR2           ;ADJUST PHYSICAL ADDRESS
1193  004730  012701  020000          MOV     #20000,R1               ;RESET VIRTUAL ADDRESS
1194  004734  012737  000001  177572  MOV     #1,@#SR0                ;RESET ERROR AND ENABLE
1195  004742  000002                  RTI                             ;RETURN
1196
1197  004744  005015  042515  047515  ADRSIS:  .ASCIZ  <15><12>'MEMORY ADDRESS IS '
1198  004752  054522  040440  042104
1199  004760  042522  051523  044440
1200  004766  020123     000
1201  004771     015  050012  051101  NOTFND:  .ASCIZ  <15><12>'PARITY ERROR NOT DETECTED ON '
1202  004776  052111  020131  051105
1203  005004  047522  020122  047516
1204  005012  020124  042504  042524
1205  005020  052103  042105  047440
1206  005026  020116     000
1207  005031     101  042104  042522  ASCAN:   .ASCIZ  'ADDRESS SCAN'
1208  005036  051523  051440  040503
1209  005044  000116
1210  005046  040504  040524  051440  DSCAN:   .ASCIZ  'DATA SCAN ORIG DATA = '
1211  005054  040503  020116  051117
1212  005062  043511  042040  052101
1213  005070  020101  020075     000
1214  005075     040  042107  042040  GDDAT:   .ASCIZ  ' GD DATA= '
1215  005102  052101  036501  000040
1216  005110  041040  020104  040504  BDDAT:   .ASCIZ  ' BD DATA= '
1217  005116  040524  020075     000
1218          005124                  .EVEN
```



```
1219                                  .SBTTL  MISC SUBROUTINES
1220                                  ;ROUTINE TO SAVE REGISTERS ON THE STACK
1221                                  ;CALLED BY SAVE MACRO OR JSR    PC,$SAVR
1222  005124  010546          $SAVR:  MOV     %5,-(SP)
1223  005126  010446                  MOV     %4,-(SP)
1224  005130  010346                  MOV     %3,-(SP)
1225  005132  010246                  MOV     %2,-(SP)
1226  005134  010146                  MOV     %1,-(SP)
1227  005136  010046                  MOV     %0,-(SP)
1228  005140  016687  000014          MOV     14(SP),PC               ;RETURN
1229
1230                                  ;ROUTINE TO RESTORE REGISTERS SAVED ON THE STACK
1231                                  ;CALLED BY RESTORE MACRO OR JSR  PC,$RESTR
1232  005144  012666  000014  $RESTR:  MOV     (SP)+,14(SP)    ;SAVE RETURN PC
1233  005150  012600                  MOV     (SP)+,%0
1234  005152  012601                  MOV     (SP)+,%1
1235  005154  012602                  MOV     (SP)+,%2
1236  005156  012603                  MOV     (SP)+,%3
1237  005160  012604                  MOV     (SP)+,%4
1238  005162  012605                  MOV     (SP)+,%5
1239  005164  000207                  RTS     PC
1240
1241                                  ;SUBROUTINE TO LOAD DISPLAY REGISTER
1242  005166  013727  001000  LDDISP:  MOV     @#IGNT,(PC)+            ;LOAD PASSCOUNT
1243  005172  000000          DISPLY:  .WORD   0
1244  005174  012746                  MOV     (PC)+,-(SP)             ;GET SECTION #
1245  005176  000000          SECT:    .WORD   0
1246  005200  006316                  ASL     (SP)
1247  005202  006316                  ASL     (SP)
1248  005204  006316                  ASL     (SP)
1249  005206  052667  177760          BIS     (SP)+,DISPLY            ;LOAD SECTION #
1250  005212  113767  001011  177753  MOVB    @#FRSTAD+1,DISPLY+1     ;LOAD BASE ADDRESS
1251  005220  105737  000770          TSTB    @#MMON                  ;CHECK IF MEM MGMT IS ON
1252  005224  001403                  BEQ     1$                      ;BRANCH IF OFF
1253  005226  013737  172344  005172  MOV     @#KIPAR2,@#DISPLY       ;LOAD CONTENTS OF KIPAR2
1254  005234  013737  005172  177570  1$:     MOV     @#DISPLY,@#DISPLAY      ;DISPLAY IN DISPLAY REGISTER
1255  005242  000207                  RTS     PC                      ;RETURN
1256
```

```
                                        ;SBTTL  KT ABORT, RESERVED & ERROR TRAP SERVICE
                                ;MEMORY MANAGEMENT ABORT SERVICE ROUTINE
005244  012737  005357  005326  KTABRT: MOV     #KTAMSG,@#ERTAG ;SET UP KT11 ABORT MSG
005252  013716  177576                  MOV     @#SR2,(SP)      ;! SR2 ONTO STACK
005256  062716  000002                  ADD     #2,(SP)
005262  000416                         BR      ERRPRT

                                ;RESERVED INSTRUCTION TRAP SERVICE ROUTINE
005264  012737  005374  005326  RESERR: MOV     #RESMSG,@#ERTAG ;LOAD RESERVED TRAP MESSAGE
005272  000412                         BR      ERRPRT

                                ;TRAP TO 4 ERROR SERVICE ROUTINE
005274  012737  000340  177776  ERRPRT: MOV     #PRIY7,@#PSW    ;SET PRIORITY LEVEL 7
005302  005737  005322                  TST     @#ERFLAG        ;CHECK IF LAST ERROR TRAP HAS BEEN
005306  001401                         BEQ     .+4             ;REPORTED
005310  000000                         HALT                    ;ERROR; TRAPPING TO LOCATION 4
                                ;STACK CONTENTS:
                                ;       (SP)                    ;THIS TRAP PC
                                ;       2(SP)                   ;THIS TRAP PSW
                                ;       4(SP)                   ;FIRST TRAP PC
                                ;       6(SP)                   ;FIRST TRAP PSW

005312  012737  005340  005326          MOV     #ERMSG,@#ERTAG  ;SET UP TIME OUT TRAP MSG
005320  005227                  ERRPRT: INC     (PC)+
005322  000000                  ERFLAG: .WORD   0
005324  000004                          TYPE
005326  000000                  ERTAG:  .WORD   0               ;CONTAINS ADR OF ERROR MSG
005330  005037  000772                  CLR     @#DEVID                 ;SET DEVICE ID = CP
005334  000137  003212                  JMP     @#.ULT

005340  005015  051124  050101  ERMSG:  .ASCIZ  <15><12> !TRAPPED TO 4'
005346  042520  020104  047324
005354  032040     000
005357     015  045412  030524  KTAMSG: .ASCIZ  <15><12> !KT11 ABORT'
005364  020061  041101  051117
005372  000124
005374  005015  042522  042523  RESMSG: .ASCIZ  <15><12>'RESERVED INST TRAP'
005402  053122  042105  044440
005410  051516  020124  051124
005416  050101     000
                005422          .EVEN
```



```
                                        ;SBTTL  PROGRAM INITIALIZATION
005422  000005  START:  RESET
005424  012706  000600                  MOV     #KPTR,SP        ;SET KERNEL STACK PTR

                                ;DETERMINE IF PROGRAM LOADED VIA ACT11 IN QUICK VERIFY MODE
005430  105037  000771                  CLRB    @#QV                    ;SET IND NOT QV MODE
005434  005737  000042                  TST     @#42                    ;BRANCH IF NOT VIA ACT11
005440  001405                         BEQ     1$
005442  005737  032174                  TST     @#LOGICAL+2             ;BANCH IF NOT QV
005446  100002                         BPL     1$
005450  110637  000771                  MOVB    SP,@#QV                 ;SET ACT11 QV MODE
                                ;ROUTINE TO DETERMINE LAST MEMORY ADDRESS
005454  012737  005476  000004  1$:    MOV     #2$,@#ERRVEC            ;SET TIME OUT TRAP TO RETURN
005462  012737  000002  000006          MOV     #RTI,@#ERRVEC+2
005470  005000                         CLR     R0
005472  005720                         TST     (R0)+                   ;WILL TIME OUTWHEN END OF MEMORY
005474  000776                         BR      .-2
005476  162700  000002          2$:    SUB     #2,R0
005502  010027                         MOV     R0,(PC)+                ;SET VALUE INTO LSTMEM
005504  000000                  LSTMEM: .WORD   0                       ;CONTAINS VALUE OF LAST MEMORY ADDRESS
005506  105737  000771                  TSTB    @#QV                    ;NO NEED TO PRESERVE LOADERS
005512  001003                         BNE     1$                      ;IF QV
005514  162737  004000  005504          SUB     #4000,@#LSTMEM          ;SET PROTECTION FOR LOADERS
005522  012737  032372  001012  1$:    MOV     #ENDTAG+2,@#FRSTMEM     ;SET LOWER BOUNDARY
005530  000423                         BR      START3                  ;GO TO START3

                                ;PROGRAM STARTS HERE WHEN ADDRESS 204 IS USED AS STARTING ADDRESS,
005532  012737  002564  000020  START1: MOV     #.TYPE,@#IOTVEC         ;SET IOT VECTOR TO TYPE ROUTINE
005540  000004  032277                  TYPE,MSG1
005544  004567  176474                  JSR     R5,RECO                 ;GET LOWER LIMIT
005550  000000                  1$:     .WORD   0                       ;CONTAINS TYPED LOWER LIMIT
005552  016737  177772  001012          MOV     1$,@#FRSTMEM            ;SET IN LOWER LIMIT
005560  000004  032314                  TYPE,MSG2
005564  004567  176454                  JSR     R5,RECO                 ;GET UPPER LIMIT
005570  000000          2$:     .WORD   0                               ;CONTAINS UPPER LIMIT
005572  016737  177772  005504          MOV     2$,@#LSTMEM

005600  005037  001000  START3: CLR     @#ICNT                  ;CLEAR PASS COUNT
005604  105037  000770          CLRB    @#MMON                  ;SET MEM MGMT ON IND=NOT ON
005610  004737  000120          JSR     PC,@#.MAMF              ;GO ENABLE PARITY IF AVAILABLE
005614  012737  001600  032006          MOV     #1600,@#NEXPAR
005622  012737  020040  001150          MOV     #20040,@#ITCNT          ;SET TEST ITERATION SOUNT
005630  105737  000771                  TSTB    @#QV                    ;BRANCH IF NOT IN QV MODE
005634  001403                         BEQ     START2
005636  012737  000401  001150          MOV     #401,@#ITCNT            ;SET 1 ITERATION FOR TESTS

                                ;PROGRAM RESTARTS HERE AFTER RELOCATION ABOVE 28K IS COMPLETE.
005644  012706  000500  START2: MOV     #STKPTR,SP              ;SET STACK PTR
005650  012737  005274  000004          MOV     #ERRPRT,@#ERRVEC        ;SET ERROR TRAP
005656  012737  005264  000010          MOV     #RESERR,@#RESVEC        ;SET RESERVED INST TRAP VECTOR
005664  012737  000002  000012          MOV     #RTI,@#RESVEC+2
005672  012737  000610  000024          MOV     #POWN,@#PFVEC           ;SET POWER FAIL TRAP VECTOR
005700  012737  000340  000026          MOV     #340,@#PFVEC+2          ;AND PRIORITY LEVEL 7
005706  012737  005244  000250          MOV     #KTABRT,@#MMVEC         ;SET KT11 ABORT VECTOR
```

```
 1353  005714  012737  002564  000020          MOV     #,TYPE,@#IOTVEC         ;SET IOT VECTOR TO TYPE ROUTINE
 1354  005722  012737  000200  000022          MOV     #PRTY4,@#IOTVEC+2       ;SET LEVEL 4 ON TRAP
 1355  005730  012737  001014  000030          MOV     #SCOPEA,@#EMTVEC        ;SET EMT(SCOPE) TRAP VECTOR
 1356  005736  012737  003212  000034          MOV     #,HLT,@#TRAPVEC         ;SET TRAP (HLT) VECTOR
 1357  005744  012737  000200  000036          MOV     #200,@#TRAPVEC+2        ;PRIORITY LEVEL 4 ON TRAP
 1358  005752  005037  005322                  CLR     @#ERFLAG                ;CLEAR ABORT & TRAP TO 4 FLAG
 1359  005756  005037  000772                  CLR     @#DEVID
 1360  005762  004737  005166                  JSR     PC,@#LODISP             ;LOAD DISPLAY REGISTER
 1361  005766  105037  000761                  CLRB    @#PEFLG                 ;CLEAR PARITY ERROR FLAG
 1362  005772  052737  000100  177560          BIS     #100,@#TKS              ;SET IE BIT IN KEYBOARD STATUS REG
 1363
 1364                                          ;THE BELOW ROUTINE ASCERTAINS WHICH CP & CP OPTIONS THE PROGRAM IS RUN-
 1365                                          ;NING ON AND SETS AN INDICATOR IN OPT,CP ACCORDINGLY.
 1366  006000  105727                  CPCHK:  TSTB    (PC)+                   ;BRANCH IF OPT,CP HAS BEEN TYPED
 1367  006002  000000                  10$:    .WORD   0
 1368  006004  001126                          BNE     9$
 1369  006006  012737  000006  000004          MOV     #ERRVEC+2,@#ERRVEC      ;SET UP ERROR TRAP TO RETURN
 1370  006014  012737  000012  000010          MOV     #RESVEC+2,@#RESVEC      ;AND ALSO RESERVED INST TRAP
 1371  006022  012700  000004                  MOV     #4,R0
 1372  006026  000261                          SEC
 1373  006030  005037  177766                  CLR     @#CPUERR                ;CLEAR CPU ERROR REG
 1374  006034  005600                          SBC     R0
 1375  006036  000261                          SEC
 1376  006040  005737  177772                  TST     @#PIRQ                  ;R0=3 IF 11/45
 1377  006044  005600                          SBC     R0                      ;R0=2 IF 11/40
 1378  006046  000261                          SEC
 1379  006050  105737  177777                  TSTB    @#PSW+1                 ;TIMES OUT IF 11/20
 1380  006054  005600                          SBC     R0                      ;R0=1 IF 11/20
 1381  006056  005037  177700                  CLR     @#177700                ;CLEARS R0 IF 11/05
 1382  006062  006300                          ASL     R0                      ;SHIFT CP INDICATOR
 1383  006064  010002                          MOV     R0,R2                   ;MOVE CP TYPE TO R2
 1384  006066  000261                          SEC
 1385  006070  005737  177572                  TST     @#SR0                   ;CHECK IF MEM MGMT IS AVAILABLE
 1386  006074  103402                          BCS     1$
 1387  006076  052702  100000                  BIS     #KTOPT,R2               ;SET MEM MGMT AVAIL INDICATOR
 1388  006102  005004                  1$:     CLR     R4
 1389  006104  000261                          SEC
 1390  006106  072404                          ASH     R4,R4                   ;WILL TRAP IF 11/40 WITHOUT EIS
 1391  006110  103402                          BCS     2$                      ;BRANCH IF NO EIS AVAILABLE
 1392  006112  052702  040000                  BIS     #EISOPT,R2              ;SET EIS AVAIL INDICATOR
 1393  006116  000261                  2$:     SEC                             ;SET CARRY
 1394  006120  170500                          TSTF    R0                      ;WILL CLEAR CARRY IF 11/45 FLOATING POINT
 1395  006122  170000                          CFCC                            ;IS AVAIL, COPY FLOATING CC'S INTO PSW
 1396  006124  103402                          BCS     3$                      ;BRANCH IF NO FLOATING POINT
 1397  006126  052702  020000                  BIS     #FPOPT,R2               ;SET FP OPTION AVAIL INDICATOR
 1398  006132  000261                  3$:     SEC                             ;SET 'C' BIT
 1399  006134  075000                          FADD    R0
 1400  006136  103402                          BCS     4$                      ;BRANCH IF NO FIS OPTION
 1401  006140  052702  010000                  BIS     #FISOPT,R2              ;SET FIS OPTION AVAIL INDICATOR
 1402  006144  000261                  4$:     SEC                             ;SET 'C' BIT
 1403  006146  005037  177774                  CLR     @#SLR                   ;CLEAR STACK LIMIT REGISTER
 1404  006152  103402                          BCS     5$                      ;BRANCH IF NOT AVAILABLE
 1405  006154  052702  004000                  BIS     #KJOPT,R2               ;SET KJ OPTION AVAIL INDICATOR
 1406  006160  000261                  5$:     SEC
```



```
 1407  006162  005737  172540                  TST     @#PLKCSR                ;BRANCH IF NO KW11-P
 1408  006166  103402                          BCS     6$
 1409  006170  052702  002000                  BIS     #PLKOPT,R2              ;SET OPTION INDICATOR
 1410  006174  000261                  6$:     SEC
 1411  006176  005737  177546                  TST     @#LKS                   ;BRANCH IF NO KW11-L
 1412  006202  103402                          BCS     7$
 1413  006204  052702  001000                  BIS     #LKOPT,R2               ;SET OPTION INDICATOR
 1414  006210  005737  177564          7$:     TST     @#TPS                   ;BRANCH IF NO CONSOLE TTY
 1415  006214  103402                          BCS     8$
 1416  006216  052702  000400                  BIS     #TTOPT,R2
 1417  006222  012737  005274  000004  8$:     MOV     #ERPRT,@#ERRVEC         ;RESTORE ERROR TRAP
 1418  006230  012737  005264  000010          MOV     #RESERR,@#RESVEC        ;AND ALSO RESERVED INST TRAP
 1419  006236  010237  000764                  MOV     R2,@#OPT,CP             ;LOAD INDICATOR
 1420  006242  000004  032264                  TYPE,AOPT,CP
 1421  006246  004767  174632                  JSR     PC,TYPDAT
 1422  006252  000004  000752                  TYPE,CRLF
 1423  006256  105267  177520                  INCB    10$                     ;SET OPT,CP HAS BEEN TYPED IND,
 1424  006262                           9$:
```

```
                                        ;SBTTL  START OF SECTION 0
                                    ;000000000000 FIRST ADDRESS TO BE RELOCATED 000000000
000262  010700                  REL0:   MOV     PC,R0           ;R0 = PC
000264  005740                          TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL0
000266  010037  001010                  MOV     R0,@#FRSTAD     ;SAVE
000272  012737  000000  005176          MOV     #0,@#SECT       ;SET SECTION #
000300  004737  005166                  JSR     PC,@#LODISP     ;LOAD DISPLAY SEG
000304  013767  005172  000746          MOV     @#DISPLY,REL00
000312  010700                          MOV     PC,R0           ;GET CURRENT PC
000314  162700  000314                  SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
000320  010037  001004                  MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
000324  010701                          MOV     PC,R1           ;SET NEW SCOPE PTR
                                    ;MEMORY AND DISK (IF SELECTED) VERIFICATION TEST.
000326  000167  000714                  JMP     1$
000332  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000340  177777  000000  000000
000346  000000  000000
000352  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000360  177777  000000  000000
000366  000000  000000
000372  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000400  177777  000000  000000
000406  000000  000000
000412  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000420  177777  000000  000000
000426  000000  000000
000432  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000440  177777  000000  000000
000446  000000  000000
000452  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000460  177777  000000  000000
000466  000000  000000
000472  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000500  177777  000000  000000
000506  000000  000000
000512  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000520  177777  000000  000000
000526  000000  000000
000532  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000540  177777  000000  000000
000546  000000  000000
000552  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000560  177777  000000  000000
000566  000000  000000
000572  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000600  177777  000000  000000
000606  000000  000000
000612  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000620  177777  000000  000000
000626  000000  000000
000632  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000640  177777  000000  000000
000646  000000  000000
```



```
000652  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000660  177777  000000  000000
000666  000000  000000
000672  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000700  177777  000000  000000
000706  000000  000000
000712  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000720  177777  000000  000000
000726  000000  000000
000732  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000740  177777  000000  000000
000746  000000  000000
000752  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
000760  177777  000000  000000
000766  000000  000000
000772  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
001000  177777  000000  000000
001006  000000  000000
001012  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
001020  177777  000000  000000
001026  000000  000000
001032  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
001040  177777  000000  000000
001046  000000  000000
001052  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
001060  177777  000000  000000
001066  000000  000000
001072  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
001100  177777  000000  000000
001106  000000  000000
001112  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
001120  177777  000000  000000
001126  000000  000000
001132  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
001140  177777  000000  000000
001146  000000  000000
001152  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
001160  177777  000000  000000
001166  000000  000000
001172  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
001200  177777  000000  000000
001206  000000  000000
001212  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
001220  177777  000000  000000
001226  000000  000000
001232  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0
001240  177777  000000  000000
001246                          1$:
001246  010702                          MOV     PC,R2
001250  062702  000012                  ADD     #12,R2
001254  012707  001152                  MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
001260  000000                  REL001: .WORD   0
                                    ;000000000000 LAST ADDRESS OF CODE TO BE RELOCATED 00000000000
```

```
                                        ;SBTTL  START OF SECTION 1
                                        ;!!!!!!!!!!!! FIRST ADDRESS TO BE RELOCATED !!!!!!!!
007262  010700                  REL1:   MOV     PC,R0           ;GET PC
007264  005740                          TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL1
007266  010037  001010                  MOV     R0,@#FRSTAD     ;SAVE
007272  012737  000001  005176          MOV     #1,@#SECT       ;SET SECTION #
007300  004737  005166                  JSR     PC,@#LODISP     ;LOAD DISPLAY REG
007304  013767  005172  003776          MOV     @#DISPLY,REL11
007312  010700                          MOV     PC,R0           ;GET CURRENT PC
007314  162700  007314                  SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
007320  010037  001004                  MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
007324  010701                          MOV     PC,R1           ;SET NEW SCOPE PTR
                                        ;CHECK BRANCH INSTRUCTIONS
007326  000257                          CCC                     ;CC'S=0000
007330  103407                          BCS     CC0             ;SAME AS BLO
007332  102406                          BVS     CC0
007334  001405                          BEQ     CC0
007336  100404                          BMI     CC0
007340  002403                          BLT     CC0
007342  003402                          BLE     CC0
007344  101401                          BLOS    CC0
007346  101001                          BHI     .+4
007350  104400                  CC0:    HLT                     ;ONE OF THE ABOVE BRANCHES FAILED

                                        ;CONTINUE
007352  000270                          SEN                     ;CC'S=1000
007354  100003                          BPL     CC1
007356  002002                          BGE     CC1
007360  003001                          BGT     CC1
007362  002401                          BLT     .+4
007364  104400                  CC1:    HLT                     ;ONE OF THE ABOVE BRANCHES FAILED

                                        ;CONTINUE
007366  000262                          SEV                     ;CC'S=1010
007370  102003                          BVC     CC2
007372  002402                          BLT     CC2
007374  003401                          BLE     CC2
007376  002001                          BGE     .+4
007400  104400                  CC2:    HLT                     ;ERROR; ONE OF THE ABOVE BRANCHES FAILED

                                        ;CONTINUE
007402  000261                          SEC                     ;CC'S=1011
007404  103002                          BCC     CC3
007406  101001                          BHI     CC3
007410  003001                          BGT     .+4
007412  104400                  CC3:    HLT                     ;ERROR; ONE OF THE ABOVE BRANCHES FAILED

                                        ;CONTINUE
007414  000264                          SEZ                     ;CC'S=1111
007416  001003                          BNE     CC4
007420  003002                          BGT     CC4
007422  101001                          BHI     CC4
007424  003401                          BLE     .+4
```



```
007426  104400                  CC4:    HLT                     ;ERROR; ONE OF THE ABOVE BRANCHES FAILED
007430  104000                          SCOPE

                                        ;TEST UNARY CONDITION CODES
                                        ;CLR    R0
007432  000277                          SCC
007434  000244                          CLZ
007436  005000                          CLR     R0              ;R0=0,CC'S=0100
007440  103404                          BCS     CLR0
007442  102403                          BVS     CLR0
007444  001002                          BNE     CLR0
007446  100401                          BMI     CLR0
007450  003401                          BLE     .+4
007452  104400                  CLR0:   HLT                     ;ERROR; INCORRECT CC'S AFTER CLR

007454  000277                          SCC
007456  000244                          CLZ
007460  005700                          TST     R0              ;R0=0,CC'S=0100
007462  103404                          BCS     TST0
007464  102403                          BVS     TST0
007466  001002                          BNE     TST0
007470  100401                          BMI     TST0
007472  101401                          BLOS    .+4
007474  104400                  TST0:   HLT                     ;ERROR; INCORRECT CC'S AFTER TST

007476  000257                          CCC
007500  000266                          .SEZ!SEV
007502  005100                          COM     R0              ;R0=-1,CC'S=1001
007504  103004                          BCC     COM0
007506  102403                          BVS     COM0
007510  001402                          BEQ     COM0
007512  100001                          BPL     COM0
007514  002401                          BLT     .+4
007516  104400                  COM0:   HLT                     ;ERROR; INCORRECT CC'S AFTER COM

007520  000261                          SEC
007522  005500                          ADC     R0              ;R0=000000,CC'S=0101
007524  103003                          BCC     ADC0
007526  102402                          BVS     ADC0
007530  001001                          BNE     ADC0
007532  002001                          BGE     .+4
007534  104400                  ADC0:   HLT                     ;ERROR; INCORRECT CC'S AFTER ADC

007536  000261                          SEC
007540  006000                          ROR     R0              ;R0=100000,CC'S=1010
007542  103404                          BCS     ROR0
007544  102003                          BVC     ROR0
007546  001402                          BEQ     ROR0
007550  100001                          BPL     ROR0
007552  003001                          BGT     .+4
007554  104400                  ROR0:   HLT                     ;ERROR; INCORRECT CC'S AFTER ROR
007556  000277                          SCC
007560  000242                          CLV
007562  005300                          DEC     R0              ;R0=077777,CC'S=0011
```

```
1641  007564  103004                  BCC     DEC0
1642  007566  102003                  BVC     DEC0
1643  007570  001402                  BEQ     DEC0
1644  007572  100401                  BMI     DEC0
1645  007574  003401                  BLE     .+4
1646  007576  104400          DEC0:   HLT                     ;ERROR: INCORRECT CC'S AFTER DEC
1647
1648  007600  000257                  CCC
1649  007602  005200                  INC     R0              ;R0=100000,CC'S=1010
1650  007604  103404                  BCS     INC0
1651  007606  102003                  BVC     INC0
1652  007610  001402                  BEQ     INC0
1653  007612  100001                  BPL     INC0
1654  007614  003001                  BGT     .+4
1655  007616  104400          INC0:   HLT                     ;ERROR: INCORRECT CC'S AFTER INC
1656
1657  007620  000277                  SCC
1658  007622  000242                  CLV
1659  007624  005400                  NEG     R0              ;R0=100000,CC'S=1011
1660  007626  103003                  BCC     NEG0
1661  007630  102002                  BVC     NEG0
1662  007632  001401                  BEQ     NEG0
1663  007634  002001                  BGE     .+4
1664  007636  104400          NEG0:   HLT                     ;ERROR: INCORRECT CC'S AFTER NEG
1665
1666  007640  000261                  SEC
1667  007642  006300                  ASL     R0              ;R0=000000,CC'S=0111
1668  007644  103004                  BCC     ASL0
1669  007646  102003                  BVC     ASL0
1670  007650  001002                  BNE     ASL0
1671  007652  100401                  BMI     ASL0
1672  007654  101401                  BLOS    .+4
1673  007656  104400          ASL0:   HLT                     ;ERROR: INCORRECT CC'S AFTER ASL
1674
1675  007660  006100                  ROL     R0              ;R0=000001,CC'S=0000
1676  007662  103402                  BCS     ROL0
1677  007664  003401                  BLE     ROL0
1678  007666  002001                  BGE     .+4
1679  007670  104400          ROL0:   HLT                     ;ERROR: INCORRECT CC'S AFTER ROL
1680
1681  007672  006200                  ASR     R0              ;R0=000000,CC'S=0111
1682  007674  103003                  BCC     ASR0
1683  007676  102002                  BVC     ASR0
1684  007700  001001                  BNE     ASR0
1685  007702  002401                  BLT     .+4
1686  007704  104400          ASR0:   HLT                     ;ERROR: INCORRECT CC'S AFTER ASR
1687
1688  007706  000277                  SCC
1689  007710  005600                  SBC     R0              ;R0=-1,CC'S=1001
1690  007712  103002                  BCC     SBC0
1691  007714  102401                  BVS     SBC0
1692  007716  003401                  BLE     .+4
1693  007720  104400          SBC0:   HLT                     ;ERROR: INCORRECT CC'S AFTER SBC
1694
```



```
1695  007722  005400                  NEG     R0              ;R0=000001,CC'S=0001
1696  007724  000300                  SWAB    R0              ;R0=000400 ,CC'S=0100
1697  007726  103403                  BCS     SWAB0
1698  007730  102402                  BVS     SWAB0
1699  007732  001001                  BNE     SWAB0
1700  007734  002001                  BGE     .+4
1701  007736  104400          SWAB0:  HLT                     ;ERROR: INCORRECT CC'S AFTER SWAB
1702  007740  104000                  SCOPE
1703
1704                          ;CHECK REGISTER SELECTION
1705  007742  005000                  CLR     R0
1706  007744  000277                  SCC
1707  007746  006100                  ROL     R0              ;R0=1
1708  007750  010002                  MOV     R0,R2
1709  007752  006302                  ASL     R2              ;R2=2
1710  007754  010203                  MOV     R2,R3
1711  007756  006303                  ASL     R3              ;R3=4
1712  007760  010304                  MOV     R3,R4
1713  007762  006304                  ASL     R4              ;R4=10
1714  007764  010405                  MOV     R4,R5
1715  007766  006305                  ASL     R5              ;R5=20
1716  007770  010546                  MOV     R5,-(SP)        ;SET BITS SET IN REGISTERS
1717  007772  050416                  BIS     R4,(SP)         ;INTO STACK ADDRESS
1718  007774  050316                  BIS     R3,(SP)
1719  007776  050216                  BIS     R2,(SP)
1720  010000  050016                  BIS     R0,(SP)
1721  010002  022726  000037          CMP     #37,(SP)+
1722  010006  001401                  BEQ     .+4             ;WERE SET
1723  010010  104400                  HLT                     ;MISSING BIT(S) REPRESENT
1724                                                          ;INCORRECT REGISTER SELECTION
1725
1726                          ;CHECK THAT ALL BITS CAN BE SET & CLEARED IN ALL REGISTERS
1727  010012  000257                  CCC
1728  010014  112700  000377          MOVB    #377,R0         ;SET ALL BITS (MOVB EXTENDS SIGN)
1729  010020  006100          1$:     ROL     R0              ;ROTATE A 0 THROUGH ALL BIT
1730  010022  103776                  BCS     1$              ;POSITIONS
1731  010024  005200                  INC     R0              ;FINAL RESULT IS -1
1732  010026  001401                  BEQ     .+4
1733  010030  104400                  HLT                     ;ERROR!
1734
1735  010032  012700  000020          MOV     #16.,R0         ;SET SHIFT COUNT
1736  010036  005002                  CLR     R2
1737  010040  000261          2$:     SEC
1738  010042  006002                  ROR     R2              ;ROTATE 1 THROUGH ALL BIT POSITS
1739  010044  005300                  DEC     R0              ;DECREMENT SHIFT COUNT
1740  010046  001374                  BNE     2$
1741  010050  005102                  COM     R2              ;R2 SHOULD CONTAIN -1
1742  010052  001401                  BEQ     .+4
1743  010054  104400                  HLT                     ;ERROR! CHECK R2 SHOULD = 0
1744
1745  010056  012703  100000          MOV     #100000,R3
1746  010062  006203          3$:     ASR     R3              ;EXTEND 1 BIT THROUGH ALL POSITIONS
1747  010064  103376                  BCC     3$
1748  010066  005203                  INC     R3
```

```
 1749  010070  001401                        BEQ     .+4
 1750  010072  104400                        HLT                     ;ERROR!
 1751
 1752  010074  112704  177401                MOVB    #177401,R4      ;R4=:
 1753  010100  060404                4$:     ADD     R4,R4           ;HAS THE AFFECT OF SHIFTING A BIT
 1754  010102  103376                        BCC     4$              ;THROUGH ALL POSITIONS
 1755  010104  005704                        TST     R4              ;RESULT SHOULD BE 0
 1756  010106  001401                        BEQ     .+4
 1757  010110  104400                        HLT
 1758
 1759  010112  012705  000021                MOV     #1,R5
 1760  010116  006305                5$:     ASL     R5
 1761  010120  102376                        BVC     5$
 1762  010122  006305                        ASL     R5
 1763  010124  103002                        BCC     6$
 1764  010126  005705                        TST     R5
 1765  010130  001401                        BEQ     .+4
 1766  010132  104400                6$:     HLT
 1767
 1768                                        ;CHECK REGISTER VOLATILITY
 1769  010134  005002                        CLR     R2
 1770  010136  005102                        COM     R2              ;R2=-1
 1771  010140  010203                        MOV     R2,R3
 1772  010142  000257                        CCC
 1773  010144  006002                        ROR     R2              ;R2=LOOP COUNT
 1774  010146  006202                        ASR     R2
 1775  010150  010304                7$:     MOV     R3,R4
 1776  010152  005302                        DEC     R2              ;DECREMENT LOOP COUNT
 1777  010154  001375                        BNE     7$
 1778  010156  005203                        INC     R3              ;CHECK R3
 1779  010160  001002                        BNE     8$
 1780  010162  005204                        INC     R4              ;CHECK R4
 1781  010164  001401                        BEQ     .+4
 1782  010166  104400                8$:     HLT
 1783
 1784                                        ;CHECK TRANSFER OF REGISTER DATA BETWEEN THE GS AND GD REGISTERS (11/45)
 1785  010170  032737  000020  177776  GSTST:  BIT     #20,@#PSW       ;CHECK IF 'T' BIT IS SET
 1786  010176  001052                        BNE     7$              ;SKIP TEST IF 'T' BIT SET
 1787  010200  010146                        MOV     R1,-(SP)        ;SAVE SCOPE PTR
 1788  010202  010627                        MOV     SP,(PC)+        ;SAVE STACK PTR
 1789  010204  000000                1$:     .WORD   0               ;CONTAINS SAVED STACK PTR
 1790  010206  010727                        MOV     PC,(PC)+        ;LOAD DATA, THE CURRENT PC IS USED AS
 1791  010210  000000                2$:     .WORD   0               ;DATA, IF THIS TEST FAILS 2$ CON-
 1792                                                                ;TAINS THE DATA BEING USED,
 1793  010212  005267  177772                INC     2$              ;MAKE ODD TO CHECK BIT 0
 1794  010216  016700  177766        3$:     MOV     2$,R0           ;LOAD GD REGISTER 0
 1795  010222  010001                        MOV     R0,R1           ;TRANSFER GS REG 0 TO GD REG 1
 1796  010224  010102                        MOV     R1,R2           ;AND GS REG 1 TO GD REG 2
 1797  010226  010203                        MOV     R2,R3           ;ETC...
 1798  010230  010304                        MOV     R3,R4
 1799  010232  010405                        MOV     R4,R5
 1800  010234  152737  000340  177776        BISB    #340,@#PSW      ;SET PRIORITY LEVEL 7
 1801  010242  010506                        MOV     R5,SP           ;TRANSFER GS REG 5 TO GD STK PTR
 1802  010244  010627                        MOV     SP,(PC)+        ;TRANSFER GS STK PTR TO MEMORY
```



```
 1803  010246  000000                4$:     .WORD   0               ;CONTAINS GS STACK PTR
 1804  010250  016706  177738                MOV     1$,SP           ;RESTORE STK PTR NEEDED FOR HLT/SCOPE
 1805  010254  142737  000340  177776        BICB    #340,@#PSW      ;SET PRIORITY LEVEL 0
 1806  010262  026700  177760                CMP     4$,R0           ;COMPARE GS/GD STKPTR WITH GS REG 0
 1807  010266  001004                        BNE     5$              ;BRANCH IF THEY WERE NOT =
 1808  010270  006367  177714                ASL     2$              ;SHIFT TEST DATA UNTIL = 000000
 1809  010274  001350                        BNE     3$
 1810  010276  000411                        BR      6$
 1811  010300  010046                5$:     MOV     R0,-(SP)        ;GET GS REG 0
 1812  010302  010146                        MOV     R1,-(SP)        ;ETC...
 1813  010304  010246                        MOV     R2,-(SP)
 1814  010306  010346                        MOV     R3,-(SP)
 1815  010310  010446                        MOV     R4,-(SP)
 1816  010312  010546                        MOV     R5,-(SP)
 1817  010314  104400                        HLT                     ;ERROR! DATA IN GS STK PTR NOT = GS REG 0
 1818                                                                ;GS REG 0-GS REG 5 ARE ON THE STACK
 1819  010316  016706  177662                MOV     1$,SP           ;RESTORE STACK PTR
 1820  010322  012601                6$:     MOV     (SP)+,R1        ;RESTORE SCOPE PTR
 1821  010324  104000                7$:     SCOPE
 1822
 1823                                        ;TEST UNARY WORD INSTRUCTIONS USING ADDRESS MODE 1
 1824  010326  000401                        BR      .+4
 1825  010330  000000                        .WORD   0               ;RESERVE ADDRESS FOR TESTS
 1826  010332  010702                        MOV     PC,R2
 1827  010334  162702  000004                SUB     #4,R2           ;R2 POINTS TO RESERVED WORD
 1828  010340  005012                        CLR     (R2)            ;PRESET (R2)
 1829
 1830  010342  000261                        SEC
 1831  010344  006012                        ROR     (R2)            ;(R2)=100000,CC=1010
 1832  010346  101402                        BLOS    ROR1
 1833  010350  100001                        BPL     ROR1
 1834  010352  002001                        BGE     .+4
 1835  010354  104400                ROR1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
 1836
 1837  010356  000257                        CCC
 1838  010360  000261                        SEC
 1839  010362  005312                        DEC     (R2)            ;(R2)=077777,CC=0011
 1840  010364  103001                        BCC     DEC1
 1841  010366  003401                        BLE     .+4
 1842  010370  104400                DEC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
 1843
 1844  010372  000257                        CCC
 1845  010374  000261                        SEC
 1846  010376  005512                        ADC     (R2)            ;(R2)=100000,CC=1010
 1847  010400  103403                        BCS     ADC1
 1848  010402  102002                        BVC     ADC1
 1849  010404  100001                        BPL     ADC1
 1850  010406  001001                        BNE     .+4
 1851  010410  104400                ADC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
 1852
 1853  010412  006112                        ROL     (R2)            ;(R2)=000000,CC=0111
 1854  010414  103003                        BCC     ROL1
 1855  010416  102002                        BVC     ROL1
 1856  010420  001001                        BNE     ROL1
```

```
010422  100001                  BPL     .+4
010424  104400          ROL1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010426  006112                  ROL     (R2)            ;(R2)=000001,CC=0000
010430  101402                  BLOS    ROL1A           ;BRANCH IF C OR Z IS SET
010432  102401                  BVS     ROL1A
010434  100001                  BPL     .+4
010436  104400          ROL1A:  HLT

010440  006212                  ASR     (R2)            ;(R2)=000000,CC=0111
010442  103003                  BCC     ASR1
010444  102002                  BVC     ASR1
010446  001001                  BNE     ASR1
010450  100001                  BPL     .+4
010452  104400          ASR1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010454  006012                  ROR     (R2)            ;(R2)=100000,CC=1010
010456  103403                  BCS     ROR1A
010460  102002                  BVC     ROR1A
010462  001401                  BEQ     ROR1A
010464  100401                  BMI     .+4
010466  104400          ROR1A:  HLT

010470  000261                  SEC
010472  005212                  INC     (R2)            ;(R2)=100001,CC=1001
010474  103003                  BCC     INC1
010476  102402                  BVS     INC1
010500  001401                  BEQ     INC1
010502  100401                  BMI     .+4
010504  104400          INC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010506  005612                  SBC     (R2)            ;(R2)=100000,CC=1000
010510  103403                  BCS     SBC1
010512  102402                  BVS     SBC1
010514  001401                  BEQ     SBC1
010516  100401                  BMI     .+4
010520  104400          SBC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010522  000261                  SEC
010524  005612                  SBC     (R2)            ;(R2)=077777,CC=0010
010526  103403                  BCS     SBC1A
010530  102002                  BVC     SBC1A
010532  001401                  BEQ     SBC1A
010534  100001                  BPL     .+4
010536  104400          SBC1A:  HLT                     ;ERROR! INCORRECT CC'S AS SHOEN ABOVE

010540  000261                  SEC
010542  005512                  ADC     (R2)            ;(R2)=100000,CC=1010
010544  100401                  BMI     .+4
010546  104400                  HLT

010550  000261                  SEC
010552  006312                  ASL     (R2)            ;(R2)=000000,CC=0111
010554  103003                  BCC     ASL1
```



```
010556  102002                  BVC     ASL1
010560  001001                  BNE     ASL1
010562  100001                  BPL     .+4
010564  104400          ASL1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010566  005112                  COM     (R2)            ;(R2)=177777,CC=1001
010570  103002                  BCC     COM1
010572  102401                  BVS     COM1
010574  100401                  BMI     .+4
010576  104400          COM1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010600  000250                  CLN
010602  005712                  TST     (R2)            ;(R2)=177777,CC=1000
010604  103403                  BCS     TST1
010606  102402                  BVS     TST1
010610  100001                  BPL     TST1
010612  001001                  BNE     .+4
010614  104400          TST1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010616  000262                  SEV
010620  005412                  NEG     (R2)            ;(R2)=000001,CC=0000
010622  103002                  BCC     NEG1
010624  102401                  BVS     NEG1
010626  001001                  BNE     .+4
010630  104400          NEG1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010632  005312                  DEC     (R2)            ;(R2)=000000,CC=0101
010634  103001                  BCC     DEC1A
010636  001401                  BEQ     .+4
010640  104400          DEC1A:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
010642  104000                  SCOPE

                        ;CHECK UNARY BYTE INSTRUCTIONS USING ADDRESS MODE 1
010644  000401                  BR      .+4             ;RESERVE A WORD
010646  000000                  .WORD   0               ;ADDRESS RESERVED FOR TESTS
010650  010703                  MOV     PC,R3
010652  162703  000004          SUB     #4,R3           ;R3 POINTS TO EVEN BYTE OF WORD
010656  010304                  MOV     R3,R4           ;R4 POINTS TO ODD BYTE OF WORD
010660  005204                  INC     R4
010662  005013                  CLR     (R3)            ;PRESET DATA

010664  000261          1$:     SEC
010666  105513                  ADCB    (R3)            ;ADD CARRY TO EVEN BYTE
010670  100402                  BMI     2$              ;UNTIL EVEN BYTE BECOMES NEGATIVE
010672  105214                  INCB    (R4)            ;INCREMENT ODD BYTE
010674  000773                  BR      1$
010676  102401          2$:     BVS     .+4             ;(R3)=077600=[0774][200],CC=1010
010700  104400                  HLT
010702  000242                  CLV
010704  105214                  INCB    (R4)            ;(R3)=100200=[1000][200],CC=1010
010706  103402                  BCS     INCB1
010710  102001                  BVC     INCB1
010712  100401                  BMI     .+4
010714  104400          INCB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
```

```
1965
1966  010716  106114                  ROLB    (R4)            ;(R3)=000200=[0000][200],CC=0111
1967  010720  103002                  BCC     ROLB1
1968  010722  102001                  BVC     ROLB1
1969  010724  001401                  BEQ     .+4
1970  010726  104400          ROLB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
1971
1972  010730  105614                  SCCB    (R4)            ;(R3)=177600=[1774][200], CC=1001
1973  010732  103002                  BCC     SBCB1
1974  010734  102401                  BVS     SBCB1
1975  010736  100401                  BMI     .+4
1976  010740  104400          SBCB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
1977
1978  010742  106313                  ASLB    (R3)            ;(R3)=177400,CC=0111
1979  010744  103002                  BCC     ASLB1
1980  010746  102001                  BVC     ASLB1
1981  010750  001401                  BEQ     .+4
1982  010752  104400          ASLB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
1983
1984  010754  105413                  NEGB    (R3)            ;(R3)=177400,CC=0100
1985  010756  103402                  BCS     NEGB1
1986  010760  102401                  BVS     NEGB1
1987  010762  001401                  BEQ     .+4
1988  010764  104400          NEGB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
1989
1990  010766  000277                  SCC
1991  010770  105313                  DECB    (R3)            ;(R3)=177777,CC=1001
1992  010772  103002                  BCC     DECB1
1993  010774  102401                  BVS     DECB1
1994  010776  001001                  BNE     .+4
1995  011000  104400          DECB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
1996
1997  011002  000241                  CLC
1998  011004  106013                  RORB    (R3)            ;(R3)=177577,CC=0011
1999  011006  103002                  BCC     RORB1
2000  011010  102001                  BVC     RORB1
2001  011012  100001                  BPL     .+4
2002  011014  104400          RORB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2003
2004  011016  000241                  CLC
2005  011020  105114                  COMB    (R4)            ;(R3)=000177,CC=0101
2006  011022  103002                  BCC     COMB1
2007  011024  102401                  BVS     COMB1
2008  011026  001401                  BEQ     .+4
2009  011030  104400          COMB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2010
2011  011032  106213          1$:     ASRB    (R3)            ;SHIFT EVEN BYTE UNTIL V CLEARS
2012  011034  102002                  BVC     2$
2013  011036  105514                  ADCB    (R4)            ;AND ADD CARRY TO ODD BYTE
2014  011040  000774                  BR      1$
2015  011042  103401          2$:     BCS     ASRB1
2016  011044  001401                  BEQ     .+4
2017  011046  104400          ASRB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2018
```



```
2019  011050  106214                  ASRB    (R4)
2020  011052  106214                  ASRB    (R4)            ;(R3)=000400,CC=0011
2021  011054  103002                  BCC     ASRB1A
2022  011056  102001                  BVC     ASRB1A
2023  011060  001001                  BNE     .+4
2024  011062  104400          ASRB1A: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2025
2026  011064  105314                  DECB    (R4)            ;(R3)=000000,CC=0100
2027  011066  001401                  BEQ     .+4
2028  011070  104400                  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2029
2030  011072  000261                  SEC
2031  011074  106014                  RORB    (R4)            ;(R3)=100000,CC=1010
2032  011076  103402                  BCS     RORB1A
2033  011100  102001                  BVC     RORB1A
2034  011102  100401                  BMI     .+4
2035  011104  104400          RORB1A: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2036
2037  011106  000242                  CLV
2038  011110  105314                  DECB    (R4)            ;(R3)=077400,CC=0100
2039  011112  102401                  BVS     .+4
2040  011114  104400                  HLT
2041
2042  011116  000261                  SEC
2043  011120  105313                  DECB    (R3)            ;(R3)=077777,CC=1001
2044  011122  103002                  BCC     DECB1A
2045  011124  102401                  BVS     DECB1A
2046  011126  100401                  BMI     .+4
2047  011130  104400          DECB1A: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2048
2049  011132  000277                  SCC
2050  011134  000313                  SWAB    (R3)            ;(R3)=177577=[1774][177],CC=0000
2051  011136  103402                  BCS     SWAB1
2052  011140  102401                  BVS     SWAB1
2053  011142  100001                  BPL     .+4
2054  011144  104400          SWAB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2055
2056  011146  105714                  TSTB    (R4)            ;(R3)=177577=[1774][177],CC=1000
2057  011150  103402                  BCS     TSTB1
2058  011152  102401                  BVS     TSTB1
2059  011154  100401                  BMI     .+4
2060  011156  104400          TSTB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2061
2062  011160  105014                  CLRB    (R4)            ;(R3)=000177=[0000][177],CC=0100
2063  011162  001401                  BEQ     .+4
2064  011164  104400                  HLT
2065  011166  106313                  ASLB    (R3)            ;(R3)=000376 ,CC=1010
2066  011170  103402                  BCS     ASLB1A
2067  011172  102001                  BVC     ASLB1A
2068  011174  100401                  BMI     .+4
2069  011176  104400          ASLB1A: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2070
2071  011200  105113                  COMB    (R3)            ;(R3)=000001,CC=0001
2072  011202  103002                  BCC     COMB1A
```

```
2073  011204  102401                        BVS     COMB1A
2074  011206  100001                        BPL     .+4
2075  011210  104400                COMB1A: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2076
2077  011212  000313                        SWAB    (R3)            ;(R3)=000400, CC=0100
2078  011214  001401                        BEQ     .+4
2079  011216  104400                        HLT
2080
2081  011220  105213                        INCB    (R3)
2082  011222  000261                        SEC
2083  011224  105613                        SBCB    (R3)            ;(R3)=000400,CC=0100
2084  011226  001401                        BEQ     .+4
2085  011230  104400                        HLT
2086  011232  022713  000400               CMP     #400,(R3)       ;CHECK REMAINING RESULT
2087  011236  001401                        BEQ     .+4
2088  011240  104400                        HLT
2089  011242  104000                        SCOPE
2090
2091                                  ;CHECK UNARY WORD OPS USING ADDRESS MODES 2 AND 4 (AUTO INC/DEC)
2092  011244  000401                        BR      .+4
2093  011246  000000                .WORD   0                       ;ADDRESS RESERVED FOR TESTS
2094  011250  010704                        MOV     PC,R4
2095  011252  162704  000004               SUB     #4,R4           ;R4 AND R5 POINT TO
2096  011256  010405                        MOV     R4,R5           ;RESERVED WORD
2097  011260  005015                        CLR     (R5)            ;PRESET DATA=0
2098
2099  011262  000277                        SCC
2100  011264  000244                        CLZ
2101  011266  005725                        TST     (R5)+           ;(R5)=000000,CC=0100
2102  011270  103402                        BCS     TST2
2103  011272  102401                        BVS     TST2
2104  011274  001401                        BEQ     .+4
2105  011276  104400                TST2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2106
2107  011300  005145                        COM     -(R5)           ;(R5)=177777,CC=1001
2108  011302  103001                        BCC     COM4
2109  011304  100401                        BMI     .+4
2110  011306  104400                COM4:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2111
2112  011310  000241                        CLC
2113  011312  006024                        ROR     (R4)+           ;(R4)=077777,CC=0011
2114  011314  103002                        BCC     ROR2
2115  011316  102001                        BVC     ROR2
2116  011320  100001                        BPL     .+4
2117  011322  104400                ROR2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2118
2119  011324  000257                        CCC
2120  011326  005244                        INC     -(R4)           ;(R4)=100000,CC=1010
2121  011330  102002                        BVC     INC4
2122  011332  001401                        BEQ     INC4
2123  011334  100401                        BMI     .+4
2124  011336  104400                INC4:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2125
2126  011340  000261                        SEC
```



```
2127  011342  000324                        SWAB    (R4)+           ;(R4)=000200,CC=1000
2128  011344  103401                        BCS     SWAB2
2129  011346  100401                        BMI     .+4
2130  011350  104400                SWAB2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2131
2132  011352  005425                        NEG     (R5)+           ;(R5)=177600,CC=1001
2133  011354  103001                        BCC     NEG2
2134  011356  100401                        BMI     .+4
2135  011360  104400                NEG2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2136
2137  011362  005044                        CLR     -(R4)           ;(R4)=000000,CC=0100
2138  011364  001401                        BEQ     .+4
2139  011366  104400                        HLT
2140
2141  011370  000261                        SEC
2142  011372  006045                        ROR     -(R5)           ;(R5)=100000,CC=1010
2143  011374  000261                        SEC
2144  011376  005525                        ADC     (R5)+           ;(R5)=100001,CC=1000
2145  011400  102401                        BVS     ADC2
2146  011402  100401                        BMI     .+4
2147  011404  104400                ADC2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2148
2149  011406  000262                        SEV
2150  011410  006224                        ASR     (R4)+           ;(R4)=140000,CC=1001
2151  011412  103002                        BCC     ASR2
2152  011414  102401                        BVS     ASR2
2153  011416  100401                        BMI     .+4
2154  011420  104400                ASR2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2155
2156  011422  000262                        SEV
2157  011424  006144                        ROL     -(R4)           ;(R4)=100001, CC=1001
2158  011426  103002                        BCC     ROL4
2159  011430  102401                        BVS     ROL4
2160  011432  100401                        BMI     .+4
2161  011434  104400                ROL4:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2162
2163  011436  005645                        SBC     -(R5)           ;(R5)=100000,CC=1000
2164  011440  103001                        BCC     .+4
2165  011442  104400                        HLT                     ;ERROR! 'C' BIT FAILED TO CLEAR
2166
2167  011444  005325                        DEC     (R5)+           ;(R5)=077777,CC=0010
2168  011446  103402                        BCS     DEC2
2169  011450  102001                        BVC     DEC2
2170  011452  100001                        BPL     .+4
2171  011454  104400                DEC2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
2172
2173  011456  006324                        ASL     (R4)+           ;(R4)=177776,CC=1010
2174  011460  102401                        BVS     .+4
2175  011462  104400                        HLT
2176  011464  006344                        ASL     -(R4)           ;(R4)=177774,CC=1001
2177  011466  103003                        BCC     ASL4
2178  011470  102402                        BVS     ASL4
2179  011472  001401                        BEQ     ASL4
2180  011474  100401                        BMI     .+4
```

```
011476  104400                  ASL41:  HLT                     ;ERROR: INCORRECT CC'S AS SHOWN ABOVE

011500  022724  177774                  CMP     #177774,(R4)+
011504  001401                          BEQ     .+4
011506  104400                          HLT
011510  020405                          CMP     R4,R5
011512  001401                          BEQ     .+4
011514  104400                          HLT
011516  104000                          SCOPE

                                ;CHECK UNARY BYTE OPS USING ADDRESS MODES 2 AND 4
011520  000401                          BR      .+4             ;RESERVE A WORD
011522  000000                  ;WORD   0                       ;RESERVED WORD
011524  010705                          MOV     PC,R5
011526  162705  000004                  SUB     #4,R5           ;R5 POINTS TO EVEN BYTE OF RESERVED WORD
011532  010500                          MOV     R5,R0
011534  010002                          MOV     R0,R2
011536  005202                          INC     R2              ;R2 POINTS TO ODD BYTE OF RESERVED WORD
011540  005010                          CLR     (R0)            ;PRESET

011542  000277                          SCC
011544  000241                          CLC
011546  105125                  COMB1:  COMB    (R5)+           ;(R0)=000377,CC=1001
011550  103002                          BCC     COMB2
011552  102401                          BVS     COMB2
011554  100401                          BMI     .+4
011556  104400                  COMB2:  HLT                     ;ERROR: INCORRECT CC'S AS SHOWN ABOVE

011560  105542                          ADCB    -(R2)           ;(R0)=000000,CC=0101
011562  001401                          BEQ     .+4
011564  104400                          HLT                     ;ERROR: INCORRECT RESULT AS SHOWN ABOVE
011566  105525                  ADCB1:  ADCB    (R5)+           ;(R0)=000400,CC=0000
011570  103401                          BCS     ADCB2
011572  001001                          BNE     .+4
011574  104400                  ADCB2:  HLT                     ;ERROR: INCORRECT CC'S AS SHOWN ABOVE

011576  000263                          +SEC!SEV
011600  106045                  RORB1:  RORB    -(R5)           ;(R0)=100000,CC=1001
011602  103003                          BCC     RORB4
011604  102402                          BVS     RORB4
011606  001401                          BEQ     RORB4
011610  100401                          BMI     .+4
011612  104400                  RORB4:  HLT                     ;ERROR: INCORRECT CC'S AS SHOWN ABOVE

011614  000277                          SCC
011616  106122                  ROLB1:  ROLB    (R2)+           ;(R0)=100001,CC=0000
011620  103403                          BCS     ROLB2
011622  102402                          BVS     ROLB2
011624  001401                          BEQ     ROLB2
011626  100001                          BPL     .+4
011630  104400                  ROLB2:  HLT                     ;ERROR: INCORRECT CC'S AS SHOWN ABOVE

011632  000257                          CCC
011634  106225                          ASRB    (R5)+           ;(R0)=140001, CC=1010
```



```
011636  103402                          BCS     ASRB2
011640  102001                          BVC     ASRB2
011642  100401                          BMI     .+4
011644  104400                  ASRB2:  HLT                     ;ERROR: INCORRECT CC'S AS SHOWN ABOVE

011646  105242                          INCB    -(R2)           ;(R0)=140002,CC=0000
011650  000277                          SCC
011652  106222                  ASRB1:  ASRB    (R2)+           ;(R0)=140001,CC=0000
011654  103402                          BCS     ASRB2A
011656  102401                          BVS     ASRB2A
011660  100001                          BPL     .+4
011662  104400                  ASRB2A: HLT                     ;ERROR: INCORRECT CC'S AS SHOWN ABOVE

011664  000266                          +SEZ!SEV                ;SET Z,V
011666  106345                  ASLB1:  ASLB    -(R5)           ;(R0)=100001,CC=1001
011670  103003                          BCC     ASLB4
011672  102402                          BVS     ASLB4
011674  001401                          BEQ     ASLB4
011676  100401                          BMI     .+4
011700  104400                  ASLB4:  HLT                     ;ERROR: INCORRECT CC'S AS SHOWN ABOVE

011702  105322                  DECB1:  DECB    (R2)+           ;(R0)=077401=[0774][001] ,CC=0010
011704  103002                          BCC     DECB2
011706  102001                          BVC     DECB2
011710  100001                          BPL     .+4
011712  104400                  DECB2:  HLT                     ;ERROR: INCORRECT CC'S AS SHOWNABOVE

011714  105645                  SBCB1:  SBCB    -(R5)           ;(R0)=077400, CC=0100
011716  103402                          BCS     SBCB4
011720  102401                          BVS     SBCB4
011722  001401                          BEQ     .+4
011724  104400                  SBCB4:  HLT                     ;ERROR: INCORRECT CC'S AS SHOWN ABOVE

011726  105442                  NEGB1:  NEGB    -(R2)           ;(R0)=100400,CC=1001
011730  103002                          BCC     NEGB4
011732  102401                          BVS     NEGB4
011734  100401                          BMI     .+4
011736  104400                  NEGB4:  HLT                     ;ERROR: INCORRECT CC'S AS SHOWN ABOVE

011740  105725                  TSTB1:  TSTB    (R5)+           ;(R0)=100400,CC=0100
011742  103401                          BCS     TSTB2
011744  001401                          BEQ     .+4
011746  104400                  TSTB2:  HLT

011750  105722                  TSTB2A: TSTB    (R2)+           ;(R0)=100400,CC=1000
011752  001401                          BEQ     TSTB2A
011754  100401                          BMI     .+4
011756  104400                  TSTB2A: HLT

011760  000261                          SEC
011762  000342                  SWAB1:  SWAB    -(R2)           ;(R0)=000201,CC=1000
011764  103401                          BCS     SWAB4
011766  100401                          BMI     .+4
011770  104400                  SWAB4:  HLT
```

```
2289
2290  011772  000277                        SCC
2291  011774  105225                        INCB    (R5)+           ;(R0)=000601#[0004][2011],CC=0000
2292  011776  103003                        BCC     INCB2
2293  012000  102402                        BVS     INCB2
2294  012002  001401                        BEQ     INCB2
2295  012004  100001                        BPL     .+4
2296  012006  104400                INCB2:  HLT
2297
2298  012010  022227  000601              CMP     (R2)+,#000601   ;CHECK END RESULT
2299  012014  001401                        BEQ     .+4
2300  012016  104400                        HLT
2301  012020  020205                        CMP     R2,R5           ;CHECK REGISTERS
2302  012022  001401                        BEQ     .+4
2303  012024  104400                        HLT
2304  012026  104000                        SCOPE
2305
2306                                ;CHECK UNARY WORD OPS USING ADDRESS MODES 3 AND 5
2307  012030  000402                        BR      .+6             ;RESERVE 2 WORDS
2308  012032  000000                        .WORD   0               ;1 FOR THE ADDRESS
2309  012034  000000                        .WORD   0               ;AND 1 FOR DATA
2310  012036  010723                        MOV     PC,R3
2311  012040  162703  000004              SUB     #4,R3
2312  012044  005013                        CLR     (R3)            ;PRESET DATA
2313  012046  010300                        MOV     R3,R0           ;R0 POINTS TO DATA WORD
2314  012050  005743                        TST     -(R3)
2315  012052  010013                        MOV     R0,(R3)
2316  012054  010304                        MOV     R3,R4
2317
2318  012056  000257                        CCC
2319  012060  005733                        TST     @(R3)+          ;(R0)=000000,CC=0100
2320  012062  001401                        BEQ     .+4
2321  012064  104400                        HLT
2322
2323  012066  000261                        SEC
2324  012070  006053                        ROR     @-(R3)          ;(R0)=100000,CC=1010
2325  012072  103402                        BCS     ROR5
2326  012074  102001                        BVC     ROR5
2327  012076  100401                        BMI     .+4
2328  012100  104400                ROR5:   HLT
2329
2330  012102  000257                        CCC
2331  012104  006234                        ASR     @(R4)+          ;(R0)=140000,CC=1010
2332  012106  102001                        BVC     ASR3
2333  012110  100401                        BMI     .+4
2334  012112  104400                ASR3:   HLT
2335
2336  012114  000250                        CLN
2337  012116  006333                        ASL     @(R3)+          ;(R0)=100000,CC=1001
2338  012120  103002                        BCC     ASL3
2339  012122  102401                        BVS     ASL3
2340  012124  100401                        BMI     .+4
2341  012126  104400                ASL3:   HLT
2342
```



```
2343  012130  000277                        SCC
2344  012132  005354                        DEC     @-(R4)          ;(R0)=077777, CC=0010
2345  012134  103003                        BCC     DEC5
2346  012136  102002                        BVC     DEC5
2347  012140  001401                        BEQ     DEC5
2348  012142  100001                        BPL     .+4
2349  012144  104400                DEC5:   HLT
2350
2351  012146  005453                        NEG     @-(R3)          ;(R0)=100001, CC=1001
2352  012150  103002                        BCC     NEG5
2353  012152  102401                        BVS     NEG5
2354  012154  100401                        BMI     .+4
2355  012156  104400                NEG5:   HLT
2356
2357  012160  000262                        SEV
2358  012162  005134                        COM     @(R4)+          ;(R0)=077776, CC=0001
2359  012164  103001                        BCC     COM3
2360  012166  102001                        BVC     .+4
2361  012170  104400                COM3:   HLT
2362
2363  012172  005233                        INC     @(R3)+          ;(R0)=077777, CC=0001
2364  012174  103001                        BCC     INC3
2365  012176  100001                        BPL     .+4
2366  012200  104400                INC3:   HLT
2367
2368  012202  005554                        ADC     @-(R4)          ;(R0)=100000, CC=1010
2369  012204  103402                        BCS     ADC5
2370  012206  102001                        BVC     ADC5
2371  012210  100401                        BMI     .+4
2372  012212  104400                ADC5:   HLT
2373
2374  012214  000257                        CCC
2375  012216  006134                        ROL     @(R4)+          ;(R0)=000000,CC=0111
2376  012220  103002                        BCC     ROL3
2377  012222  102001                        BVC     ROL3
2378  012224  001401                        BEQ     .+4
2379  012226  104400                ROL3:   HLT
2380
2381  012230  005253                        INC     @-(R3)          ;(R0)=000001, CC=0001
2382  012232  005654                        SBC     @-(R4)          ;(R0)=000000, CC=0100
2383  012234  103401                        BCS     SBC5
2384  012236  001401                        BEQ     .+4
2385  012240  104400                SBC5:   HLT
2386  012242  104000                        SCOPE
2387
2388                                ;CHECK UNARY BYTE OPS USING ADDRESS MODES 3 AND 5
2389  012244  000403                        BR      .+10            ;RESERVE 3 WORDS
2390  012246  000000                        .WORD   0               ;1 FOR EVEN BYTE ADDRESS
2391  012250  000000                        .WORD   0               ;1 FOR ODD BYTE ADDRESS
2392  012252  000000                        .WORD   0               ;AND 1 FOR DATA
2393  012254  010702                        MOV     PC,R2
2394  012256  005742                        TST     -(R2)           ;BACK R2 UP TO
2395  012260  005742                        TST     -(R2)           ;DATA WORD
2396  012262  010200                        MOV     R2,R0           ;R0 POINTS TO THE DATA WORD
```

```
 2397  012264  005010                        CLR     (R0)            ;PRESET DATA
 2398  012266  005742                        TST     -(R2)           ;BACK R2 UP TO
 2399  012270  005742                        TST     -(R2)           ;EVEN BYTE ADDRESS WORD
 2400  012272  010022                        MOV     R0,(R2)+        ;LOAD EVEN BYTE ADDRESS
 2401  012274  005200                        INC     R0              ;ODD BYTE ADDRESS
 2402  012276  010022                        MOV     R0,(R2)+        ;LOAD ODD BYTE ADDRESS
 2403  012300  010200                        MOV     R2,R0           ;RESET R0
 2404  012302  010205                        MOV     R2,R5
 2405
 2406  012304  105152                        COMB    @-(R2)          ;(R0)=177400,CC=1001
 2407  012306  103001                        BCC     COMB5
 2408  012310  100401                        BMI     .+4
 2409  012312  104400                COMB5:  HLT
 2410
 2411  012314  105752                        TSTB    @-(R2)          ;(R0)=177400, CC=0100
 2412  012316  001401                        BEQ     .+4
 2413  012320  104400                        HLT
 2414
 2415  012322  000262                        SEV
 2416  012324  106255                        ASRB    @-(R5)          ;(R0)=177400, CC=1001
 2417  012326  103002                        BCC     ASRB5
 2418  012330  102401                        BVS     ASRB5
 2419  012332  100401                        BMI     .+4
 2420  012334  104400                ASRB5:  HLT
 2421
 2422  012336  105232                        INCB    @(R2)+          ;(R0)=177401, CC=000
 2423  012340  103001                        BCC     INCB3
 2424  012342  100001                        BPL     .+4
 2425  012344  104400                INCB3:  HLT
 2426
 2427  012346  000241                        CLC
 2428  012350  106055                        RORB    @-(R5)          ;(R0)=177400, CC=0111
 2429  012352  103003                        BCC     RORB5
 2430  012354  102002                        BVC     RORB5
 2431  012356  001001                        BNE     RORB5
 2432  012360  100001                        BPL     .+4
 2433  012362  104400                RORB5:  HLT
 2434
 2435  012364  106332                        ASLB    @(R2)+          ;(R0)=177000, CC=1001
 2436  012366  103002                        BCC     ASLB3
 2437  012370  102401                        BVS     ASLB3
 2438  012372  100401                        BMI     .+4
 2439  012374  104400                ASLB3:  HLT
 2440
 2441  012376  105552                        ADCB    @-(R2)          ;(R0)=177400, CC=1000
 2442  012400  103401                        BCS     ADCB5
 2443  012402  100401                        BMI     .+4
 2444  012404  104400                ADCB5:  HLT
 2445
 2446  012406  000277                        SCC
 2447  012410  106135                        ROLB    @(R5)+          ;(R0)=177401, CC=0000
 2448  012412  101402                        BLOS    ROLB3           ;BRANCH IF C OR Z IS SET
 2449  012414  102401                        BVS     ROLB3
 2450  012416  100001                        BPL     .+4
```



```
 2451  012420  104400                ROLB3:  HLT
 2452
 2453  012422  000352                        SWAB    @-(R2)          ;(R0)=000777, CC=1000
 2454  012424  100401                        BMI     .+4
 2455  012426  104400                        HLT
 2456
 2457  012430  000261                        SEC
 2458  012432  105635                        SBCB    @(R5)+          ;(R0)=000377, CC=0100
 2459  012434  103401                        BCS     SBCB3
 2460  012436  001401                        BEQ     .+4
 2461  012440  104400                SBCB3:  HLT
 2462
 2463  012442  105432                        NEGB    @(R2)+          ;(R0)=000001
 2464  012444  105352                        DECB    @-(R2)          ;(R0)=000000, CC=0101
 2465  012446  103001                        BCC     DECB5
 2466  012450  001401                        BEQ     .+4
 2467  012452  104400                DECB5:  HLT
 2468  012454  104000                        SCOPE
 2469
 2470                                  ;CHECK UNARY WORD OPS USING ADDRESS MODE 6 (PC)
 2471  012456  005027                        CLR     (PC)+           ;PRESET DATA = 0
 2472  012460  000000                UWM6:   .WORD   0               ;RESERVED FOR DATA
 2473  012462  010700                        MOV     PC,R0
 2474  012464  024040                        CMP     -(R0),-(R0)     ;R0 POINTS TO DATA WORD
```

```
2475  012466  000277                        SCC
2476  012470  006167  177764                ROL     UWM6            ;(R0)=000001,CC=0000
2477  012474  103403                        BCS     ROL6
2478  012476  102402                        BVS     ROL6
2479  012500  001401                        BEQ     ROL6
2480  012502  100001                        BPL     .+4
2481  012504  104400              ROL6:     HLT
2482
2483  012506  005167  177746                COM     UWM6            ;(R0)=177776, CC=1001
2484  012512  103002                        BCC     COM6
2485  012514  102401                        BVS     COM6
2486  012516  100401                        BMI     .+4
2487  012520  104400              COM6:     HLT
2488  012522  006267  177732                ASR     UWM6            ;(R0)=177777, CC=1010
2489  012526  103402                        BCS     ASR6
2490  012530  102001                        BVC     ASR6
2491  012532  100401                        BMI     .+4
2492  012534  104400              ASR6:     HLT
2493
2494  012536  000277                        SCC
2495  012540  005467  177714                NEG     UWM6            ;(R0)=000001, CC=0001
2496  012544  103003                        BCC     NEG6
2497  012546  102402                        BVS     NEG6
2498  012550  001401                        BEQ     NEG6
2499  012552  100001                        BPL     .+4
2500  012554  104400              NEG6:     HLT
2501
2502  012556  000277                        SCC
2503  012560  006067  177674                ROR     UWM6            ;(R0)=100000, CC=1001
2504  012564  103003                        BCC     ROR6
2505  012566  102402                        BVS     ROR6
2506  012570  001401                        BEQ     ROR6
2507  012572  100401                        BMI     .+4
2508  012574  104400              ROR6:     HLT
2509
2510  012576  005667  177656                SBC     UWM6            ;(R0)=077777, CC=0010
2511  012602  103402                        BCS     SBC6
2512  012604  102001                        BVC     SBC6
2513  012606  100001                        BPL     .+4
2514  012610  104400              SBC6:     HLT
2515
2516  012612  000242                        CLV
2517  012614  005267  177640                INC     UWM6            ;(R0)=100000, CC=1011
2518  012620  103403                        BCS     INC6
2519  012622  102002                        BVC     INC6
2520  012624  001401                        BEQ     INC6
2521  012626  100401                        BMI     .+4
2522  012630  104400              INC6:     HLT
2523
2524  012632  006267  177622                ASR     UWM6            ;(R0)=140000, CC=1010
2525  012636  000261                        SEC
2526  012640  006367  177614                ASL     UWM6            ;(R0)=100000, CC=1001
2527  012644  103002                        BCC     ASL6
2528  012646  102401                        BVS     ASL6
```



```
2529  012650  100401                        BMI     .+4
2530  012652  104400              ASL6:     HLT
2531
2532  012654  005367  177600                DEC     UWM6            ;(R0)=077777, CC=0011
2533  012660  103002                        BCC     DEC6
2534  012662  102001                        BVC     DEC6
2535  012664  100001                        BPL     .+4
2536  012666  104400              DEC6:     HLT
2537
2538  012670  005567  177564                ADC     UWM6            ;(R0)=100000, CC=1010
2539  012674  103402                        BCS     ADC6
2540  012676  102001                        BVC     ADC6
2541  012700  100401                        BMI     .+4
2542  012702  104400              ADC6:     HLT
2543  012704  000242                        CLV
2544  012706  000367  177546                SWAB    UWM6
2545  012712  100401                        BMI     .+4
2546  012714  104400                        HLT
2547  012716  022710  000200                CMP     #200,(R0)
2548  012722  001401                        BEQ     .+4
2549  012724  104400                        HLT
2550  012726  104000                        SCOPE
2551
2552                                  ;CHECK UNARY BYTE OPS (EVEN/ODD) USING ADDRESS MODE 6 (PC)
2553  012730  012700  013272                MOV     #UBM6,R0
2554  012734  063700  001004                ADD     @#FACTOR,R0     ;R0 POINTS TO ADDRESS OF DATA
2555  012740  005067  000326                CLR     UBM6            ;CLEAR DATA
2556  012744  000277                        SCC
2557  012746  000244                        CLE
2558  012750  105767  000316                TSTB    UBM6
2559  012754  103403                        BCS     TSTB6
2560  012756  102402                        BVS     TSTB6
2561  012760  001001                        BNE     TSTB6
2562  012762  100001                        BPL     .+4
2563  012764  104400              TSTB6:    HLT
2564
2565  012766  000257                        CCC
2566  012770  105767  000277                TSTB    UBM6+1          ;TEST ODD BYTE
2567  012774  001401                        BEQ     .+4
2568  012776  104400                        HLT
2569
2570  013000  105667  000266                SBCB    UBM6            ;(R0)=000000, CC=0100
2571  013004  103402                        BCS     SBCB6
2572  013006  102401                        BVS     SBCB6
2573  013010  001401                        BEQ     .+4
2574  013012  104400              SBCB6:    HLT
2575
2576  013014  000261              1$:       SEC
2577  013016  105267  000250                INCB    UBM6            ;LOOP UNTIL (R0)=077600, CC=1011
2578  013022  100403                        BMI     2$
2579  013024  105567  000243                ADCB    UBM6+1          ;INCB INST INCREMENTS EVEN BYTE
2580  013030  000771                        BR      1$              ;ADCB INCREMENTS ODD BYTE
2581  013032  103001              2$:       BCC     INCB6
2582  013034  102401                        BVS     .+4
```

```
2583  013036  104400                  INCB6$: HLT
2584
2585  013040  106367  000226                  ASLB    UBM6            ;(R0)=077400, CC=0111
2586  013044  103003                  ASLB6:  BCC     ASLB6
2587  013046  102002                          BVC     ASLB6
2588  013050  001001                          BNE     ASLB6
2589  013052  100001                          BPL     .+4
2590  013054  104400                  ASLB6$: HLT
2591
2592  013056  000242                          CLV
2593  013060  105567  000207                  ADCB    UBM6+1          ;(R0)=100000, CC=1010
2594  013064  103402                  ADCB6:  BCS     ADCB6
2595  013066  102001                          BVC     ADCB6
2596  013070  100401                          BMI     .+4
2597  013072  104400                  ADCB6$: HLT
2598
2599  013074  000261                          SEC
2600  013076  106067  000171                  RORB    UBM6+1          ;(R0)=140000, CC=1010
2601  013102  103402                  RORB6:  BCS     RORB6
2602  013104  102001                          BVC     RORB6
2603  013106  100401                          BMI     .+4
2604  013110  104400                  RORB6$: HLT
2605
2606  013112  105167  000154                  COMB    UBM6            ;(R0)=140377 CC=1001
2607  013116  103002                  COMB6:  BCC     COMB6
2608  013120  102401                          BVS     COMB6
2609  013122  100401                          BMI     .+4
2610  013124  104400                  COMB6$: HLT
2611
2612  013126  000262                          SEV
2613  013130  105467  000137                  NEGB    UBM6+1          ;(R0)=040377, CC=0001
2614  013134  103002                  NEGB6:  BCC     NEGB6
2615  013136  102401                          BVS     NEGB6
2616  013140  100001                          BPL     .+4
2617  013142  104400                  NEGB6$: HLT
2618
2619  013144  106167  000123                  ROLB    UBM6+1          ;(R0)=100777, CC=1010
2620  013150  103402                  ROLB6:  BCS     ROLB6
2621  013152  102001                          BVC     ROLB6
2622  013154  100401                          BMI     .+4
2623  013156  104400                  ROLB6$: HLT
2624
2625  013160  106267  000106                  ASRB    UBM6            ;(R0)=100777, CC=1001
2626  013164  103002                  ASRB6:  BCC     ASRB6
2627  013166  102401                          BVS     ASRB6
2628  013170  100401                          BMI     .+4
2629  013172  104400                  ASRB6$: HLT
2630
2631  013174  105267  000072                  INCB    UBM6            ;(R0)=100400, CC=0101
2632  013200  103002                  INCB6:  BCC     INCB6A
2633  013202  102401                          BVS     INCB6A
2634  013204  001401                          BEQ     .+4
2635  013206  104400                  INCB6A: HLT
2636
```



```
2637  013210  105367  000057                  DECB    UBM6+1          ;(R0)=100000, CC=1001
2638  013214  103003                  DECB6:  BCC     DECB6A
2639  013216  102402                          BVS     DECB6A
2640  013220  001401                          BEQ     DECB6A
2641  013222  100401                          BMI     .+4
2642  013224  104400                  DECB6A: HLT
2643
2644  013226  000367  000040                  SWAB    UBM6            ;(R0)=000200, CC=1000
2645  013232  103401                  SWAB6:  BCS     SWAB6
2646  013234  100401                          BMI     .+4
2647  013236  104400                  SWAB6$: HLT
2648
2649  013240  106167  000026                  ROLB    UBM6            ;(R0)=000000, CC=0111
2650  013244  103002                  ROLB6:  BCC     ROLB6A
2651  013246  102001                          BVC     ROLB6A
2652  013250  001401                          BEQ     .+4
2653  013252  104400                  ROLB6A: HLT
2654
2655  013254  005767  000012                  TST     UBM6            ;(R0)=000000, CC=0100
2656  013260  103402                  TST6:   BCS     TST6
2657  013262  102401                          BVS     TST6
2658  013264  001401                          BEQ     .+4
2659  013266  104400                  TST6$:  HLT
2660
2661  013270  000401                          BR      .+4             ;RESERVE A WORD
2662  013272  000000                  UBM6:   .WORD   0               ;WORD RESERVED FOR DATA
2663  013274  104000                          SCOPE
2664  013276  010702                          MOV     PC,R2
2665  013300  062702  000012                  ADD     #12,R2
2666  013304  012707  001152                  MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
2667  013310  000000                  REL11:  .WORD   0
2668                                  ;1111111111111 LAST ADDRESS OF CODE TO BE RELOCATED 11111111111
2669
2670
2671                                          .SBTTL  START OF SECTION 2
2672                                  ;2222222222222 FIRST ADDRESS TO BE RELOCATED 222222222
2673  013312  010700                  REL21:  MOV     PC,R0           ;GET PC
2674  013314  005740                          TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL2
2675  013316  010037  001010                  MOV     R0,@#FRSTAD     ;SAVE
2676  013322  012737  000002  005176          MOV     #2,@#SECT       ;SET SECTION #
2677  013330  004737  005166                  JSR     PC,@#LDDISP     ;LOAD DISPLAY REG
2678  013334  013767  005172  003744          MOV     @#DISPLY,REL22
2679  013342  010700                          MOV     PC,R0           ;GET CURRENT PC
2680  013344  162700  013344                  SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
2681  013350  010037  001004                  MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
2682  013354  010701                          MOV     PC,R1           ;SET NEW SCOPE PTR
2683
2684                                  ;CHECK UNARY WORD OPS USING ADDRESS MODE 7
2685  013356  000403                          BR      UW7             ;RESERVE 3 WORDS FOR ADDRESSES & DATA
2686  013360  000000                          .WORD   0               ;CONTAINS ADDRESS OF UWM7
2687  013362  000000                  UWM7:   .WORD   0               ;CONTAINS DATA
2688  013364  000000                          .WORD   0               ;CONTAINS ADDRESS OF UWM7
2689
2690  013366  010700                  UW7:    MOV     PC,R0
```

```
 2691  013370  005740                        TST     -(R0)
 2692  013372  005740                        TST     -(R0)
 2693  013374  005040                        CLR     -(R0)           ;CLEAR TEST DATA
 2694  013376  010002                        MOV     R0,R2
 2695  013400  010240                        MOV     R2,-(R0)        ;SET UP ADDRESS
 2696  013402  005720                        TST     (R0)+           ;MOVE R0 TO NEXT ADDRESS
 2697  013404  005720                        TST     (R0)+
 2698  013406  010210                        MOV     R2,(R0)         ;SET NEXT ADDRESS
 2699  013410  010200                        MOV     R2,R0           ;SET R0 POINTING TO DATA
 2700  013412  000277                        SCC
 2701  013414  000244                        CLZ
 2702  013416  005772  000002                TST     @2(2)           ;(R0)=000000, CC=0100
 2703  013422  001401                        BEQ     .+4
 2704  013424  104400                        HLT
 2705
 2706  013426  000277                        SCC
 2707  013430  005672  177776                SBC     @-2(2)          ;(R0)=177777, CC=1001
 2708  013434  103002                        BCC     SBC7
 2709  013436  102401                        BVS     SBC7
 2710  013440  100401                        BMI     .+4
 2711  013442  104400              SBC7:     HLT
 2712
 2713  013444  000277                        SCC
 2714  013446  000241                        CLC
 2715  013450  006372  000002                ASL     @2(2)           ;(R0)=177776, CC=1001
 2716  013454  103002                        BCC     ASL7
 2717  013456  102401                        BVS     ASL7
 2718  013460  100401                        BMI     .+4
 2719  013462  104400              ASL7:     HLT
 2720
 2721  013464  000257                        CCC
 2722  013466  005372  000002                DEC     @2(2)           ;(R0)=177775, CC=1000
 2723  013472  103002                        BCS     DEC7
 2724  013474  102401                        BVS     DEC7
 2725  013476  100401                        BMI     .+4
 2726  013500  104400              DEC7:     HLT
 2727
 2728  013502  000262                        SEV
 2729  013504  006272  177776                ASR     @-2(2)          ;(R0)=177776, CC=1001
 2730  013510  103002                        BCC     ASR7
 2731  013512  102401                        BVS     ASR7
 2732  013514  100401                        BMI     .+4
 2733  013516  104400              ASR7:     HLT
 2734
 2735  013520  000241                        CLC
 2736  013522  000262                        SEV
 2737  013524  006072  177776                ROR     @-2(2)          ;(R0)=077777, CC=0000
 2738  013530  101402                        BLOS    ROR7            ;BRANCH IF C OR Z IS SET
 2739  013532  102401                        BVS     ROR7
 2740  013534  100001                        BPL     .+4
 2741  013536  104400              ROR7:     HLT
 2742
 2743  013540  000262                        SEV
 2744  013542  005472  000002                NEG     @2(2)           ;(R0)=100001, CC=1001
```



```
 2745  013546  103002                        BCC     NEG7
 2746  013550  102401                        BVS     NEG7
 2747  013552  100401                        BMI     .+4
 2748  013554  104400              NEG7:     HLT
 2749
 2750  013556  000250                        CLN
 2751  013560  000372  177776                SWAB    @-2(2)          ;(R0)=000600, CC=1000
 2752  013564  103401                        BCS     SWAB7
 2753  013566  100401                        BMI     .+4
 2754  013570  104400              SWAB7:    HLT
 2755
 2756  013572  000262                        SEV
 2757  013574  005172  000002                COM     @2(2)           ;(R0)=177177, CC=1001
 2758  013600  103002                        BCC     COM7
 2759  013602  102401                        BVS     COM7
 2760  013604  100401                        BMI     .+4
 2761  013606  104400              COM7:     HLT
 2762
 2763  013610  000372  000002                SWAB    @2(2)           ;(R0)=077776, CC=1000
 2764  013614  100401                        BMI     .+4
 2765  013616  104400                        HLT
 2766
 2767  013620  000277                        SCC
 2768  013622  005572  177776                ADC     @-2(2)          ;(R0)=077777, CC=0000
 2769  013626  103402                        BCS     ADC7
 2770  013630  102401                        BVS     ADC7
 2771  013632  100001                        BPL     .+4
 2772  013634  104400              ADC7:     HLT
 2773
 2774  013636  005272  000002                INC     @2(2)           ;(R0)=100000, CC=1010
 2775  013642  102001                        BVC     INC7
 2776  013644  100401                        BMI     .+4
 2777  013646  104400              INC7:     HLT
 2778
 2779  013650  000257                        CCC
 2780  013652  006172  177776                ROL     @-2(2)          ;(R0)=000000, CC=0111
 2781  013656  103002                        BCC     ROL7
 2782  013660  102001                        BVC     ROL7
 2783  013662  001401                        BEQ     .+4
 2784  013664  104400              ROL7:     HLT
 2785  013666  104000                        SCOPE
 2786
 2787                                  ;CHECK UNARY BYTE OPS USING ADDRESS MODE 7
 2788  013670  005720                        TST     (R0)+
 2789  013672  005210                        INC     (R0)            ;WORD FOLLOWING UUO7 CONTAINS ADDRESS
 2790  013674  005740                        TST     -(R0)           ;OF ODD BYTE, R0 POINTS TO DATA WORD
 2791  013676  005010                        CLR     (R0)            ;PRESET DATA
 2792  013700  010701                        MOV     PC,R1           ;SET SCOPE PTR
 2793                                  ;NOTE;  @2(2) REFERENCES THE ODD BYTE, AND @-2(2) REFERENCES THE EVEN BYTE.
 2794
 2795  013702  000263                        .SECISEV                ;SET C AND V
 2796  013704  105672  000002                SBCB    @2(2)           ;(R0)=177400, CC=1001
 2797  013710  103003                        BCC     SBCB7
 2798  013712  102402                        BVS     SBCB7
```

```
2799  013714  001401                    BEQ     SBCB7
2800  013716  100401                    BMI     .+4
2801  013720  104400                SBCB7:  HLT
2802
2803  013722  000277                    SCC                     ;SET CONDITION CODES
2804  013724  105572  177776            ADCB    @-2(2)          ;(R0)=177401, CC=0000
2805  013730  103405                    BCS     ADCB7
2806  013732  102402                    BVS     ADCB7
2807  013734  001401                    BEQ     ADCB7
2808  013736  100001                    BPL     .+4
2809  013740  104400                ADCB7:  HLT
2810
2811  013742  105172  177776            COMB    @-2(2)          ;(R0)=177776, CC=1001
2812  013746  103002                    BCC     COMB7
2813  013750  102401                    BVS     COMB7
2814  013752  100401                    BMI     .+4
2815  013754  104400                COMB7:  HLT
2816
2817  013756  000241                    CLC                     ;CLEAR CARRY
2818  013760  106072  000002            RORB    #2(2)           ;(R0)=077776, CC=0011
2819  013764  103002                    BCC     RORB7
2820  013766  102001                    BVC     RORB7
2821  013770  100001                    BPL     .+4
2822  013772  104400                RORB7:  HLT
2823
2824  013774  105272  000002            INCB    #2(2)           ;(R0)=100376, CC=1011
2825  014000  103002                    BCC     INCB7
2826  014002  102001                    BVC     INCB7
2827  014004  100401                    BMI     .+4
2828  014006  104400                INCB7:  HLT
2829
2830  014010  105372  177776            DECB    @-2(2)          ;(R0)=100375, CC=1001
2831  014014  103002                    BCC     DECB7
2832  014016  102401                    BVS     DECB7
2833  014020  100401                    BMI     .+4
2834  014022  104400                DECB7:  HLT
2835
2836  014024  106372  000002            ASLB    #2(2)           ;(R0)=000375, CC=0111
2837  014030  103002                    BCC     ASLB7
2838  014032  102001                    BVC     ASLB7
2839  014034  001401                    BEQ     .+4
2840  014036  104400                ASLB7:  HLT
2841
2842  014040  000241                    CLC                     ;CLEAR CARRY
2843  014042  106272  177776            ASRB    @-2(2)          ;(R0)=000376, CC=1001
2844  014046  103002                    BCC     ASRB7
2845  014050  102401                    BVS     ASRB7
2846  014052  100401                    BMI     .+4
2847  014054  104400                ASRB7:  HLT
2848
2849  014056  105472  000002            NEGB    #2(2)           ;(R0)=000376, CC=0100
2850  014062  103402                    BCS     NEGB7
2851  014064  102401                    BVS     NEGB7
2852  014066  001401                    BEQ     .+4
```



```
2853  014070  104400                NEGB7:  HLT
2854
2855  014072  000262                    SEV
2856  014074  106172  177776            ROLB    @-2(2)          ;(R0)=000374, CC=1001
2857  014100  103002                    BCC     ROLB7
2858  014102  102401                    BVS     ROLB7
2859  014104  100401                    BMI     .+4
2860  014106  104400                ROLB7:  HLT
2861
2862  014110  105272  177776            INCB    @-2(2)          ;(R0)=000375, CC=1001
2863  014114  105272  177776            INCB    @-2(2)          ;(R0)=000376, CC=1001
2864  014120  105572  177776            ADCB    @-2(2)          ;(R0)=000377, CC=1000
2865  014124  105172  177776            COMB    @-2(2)          ;(R0)=000000, CC=0100
2866  014130  001401                    BEQ     .+4
2867  014132  104400                    HLT
2868  014134  104000                    SCOPE
2869
2870                                ;CHECK BINARY OPS  USING ADDRESS MODE 0
2871  014136  000277                    SCC                     ;SET CONDITION CODES
2872  014140  010700                    MOV     PC,R0           ;R0=PC, CC=X001
2873  014142  103002                    BCC     MOV0
2874  014144  102401                    BVS     MOV0
2875  014146  001001                    BNE     .+4
2876  014150  104400                MOV0:   HLT
2877
2878  014152  010002                    MOV     R0,R2           ;R2=R0
2879  014154  000262                    SEV                     ;SET V
2880  014156  160002                    SUB     R0,R2           ;R2=000000, CC=0100
2881  014160  103402                    BCS     SUB0
2882  014162  102401                    BVS     SUB0
2883  014164  001401                    BEQ     .+4
2884  014166  104400                SUB0:   HLT
2885
2886  014170  000244                    CLZ
2887  014172  010203                    MOV     R2,R3           ;R2=R3=000000, CC=0100
2888  014174  103401                    BCS     MOV0A
2889  014176  001401                    BEQ     .+4
2890  014200  104400                MOV0A:  HLT
2891
2892  014202  000257                    CCC
2893  014204  000272                    *SEV;SEN                ;SET V & N
2894  014206  020203                    CMP     R2,R3           ;R2=R3=000000, CC=0100
2895  014210  103403                    BCS     CMP0
2896  014212  102402                    BVS     CMP0
2897  014214  001001                    BNE     CMP0
2898  014216  100001                    BPL     .+4
2899  014220  104400                CMP0:   HLT
2900
2901  014222  010002                    MOV     R0,R2           ;R0=R2
2902  014224  010203                    MOV     R2,R3           ;R0=R2=R3
2903  014226  060203                    ADD     R2,R3           ;R3=2*R0
2904  014230  006302                    ASL     R2              ;R2=2*R0
2905  014232  020203                    CMP     R2,R3           ;R2=R3=2*R0
2906  014234  001401                    BEQ     .+4
```

```
 2907  014236  104400                        HLT                     ;ERROR! CHECK ADD INSTRUCTION
 2908
 2909                                        ;THE FOLLOWING SUBTEST SHIFTS A BIT THROUGH R2 AND R5 AND DOES A
 2910                                        ;BIT TEST (BIT) USING R2 AND R5.
 2911  014240  005002                        CLR     R2
 2912  014242  005202                        INC     R2
 2913  014244  000402                        BR      2$
 2914  014246  006302              1$:       ASL     R2
 2915  014250  100407                        BMI     4$
 2916  014252  010205              2$:       MOV     R2,R5
 2917  014254  000277                        SCC
 2918  014256  030205                        BIT     R2,R5           ;R2=R5
 2919  014260  103002                        BCC     3$
 2920  014262  102401                        BVS     3$
 2921  014264  001370                        BNE     1$
 2922  014266  104400              3$:       HLT
 2923  014270  010205              4$:       MOV     R2,R5
 2924  014272  000257                        CCC
 2925  014274  030205                        BIT     R2,R5
 2926  014276  100401                        BMI     .+4
 2927  014300  104400                        HLT
 2928
 2929  014302  005002                        CLR     R2
 2930  014304  000277                        SCC
 2931  014306  050002                        BIS     R0,R2
 2932  014310  103002                        BCC     BIS0
 2933  014312  102401                        BVS     BIS0
 2934  014314  001001                        BNE     .+4
 2935  014316  104400              BIS0:     HLT
 2936
 2937  014320  010003                        MOV     R0,R3
 2938  014322  000277                        SCC
 2939  014324  000244                        CLZ
 2940  014326  040003                        BIC     R0,R3
 2941  014330  103003                        BCC     BIC0
 2942  014332  102402                        BVS     BIC0
 2943  014334  001001                        BNE     BIC0
 2944  014336  100001                        BPL     .+4
 2945  014340  104400              BIC0:     HLT
 2946
 2947  014342  010004                        MOV     R0,R4
 2948  014344  005104                        COM     R4
 2949  014346  040004                        BIC     R0,R4
 2950  014350  005104                        COM     R4
 2951  014352  020004                        CMP     R0,R4
 2952  014354  001401                        BEQ     .+4
 2953  014356  104400                        HLT
 2954
 2955  014360  010004                        MOV     R0,R4
 2956  014362  005104                        COM     R4
 2957  014364  010403                        MOV     R4,R3
 2958  014366  050003                        BIS     R0,R3
 2959  014370  103001                        BCC     BIS0A
 2960  014372  100401                        BMI     .+4
```



```
 2961  014374  104400              BIS0A:    HLT
 2962  014376  005203                        INC     R3
 2963  014400  001401                        BEQ     .+4
 2964  014402  104400                        HLT
 2965  014404  010304                        MOV     R3,R4           ;R3=R4=0
 2966  014406  005103                        COM     R3              ;R3=177777
 2967  014410  000261                        SEC                     ;SET C
 2968  014412  006004                        ROR     R4              ;R4=100000
 2969  014414  060304                        ADD     R3,R4           ;R3=177777,R4=077777, CC=0011
 2970  014416  103003                        BCC     ADD0
 2971  014420  102002                        BVC     ADD0
 2972  014422  001401                        BEQ     ADD0
 2973  014424  100001                        BPL     .+4
 2974  014426  104400              ADD0:     HLT
 2975  014430  010700                        MOV     PC,R0
 2976  014432  022020                        CMP     (R0)+,(R0)+
 2977  014434  020007                        CMP     R0,PC
 2978  014436  001401                        BEQ     .+4
 2979  014440  104400                        HLT
 2980
 2981  014442  010700                        MOV     PC,R0
 2982  014444  062700  000010                ADD     #10,R0
 2983  014450  010002                        MOV     R0,R2
 2984  014452  020700                        CMP     PC,R0
 2985  014454  001002                        BNE     CMP0A
 2986  014456  020200                        CMP     R2,R0
 2987  014460  001401                        BEQ     .+4
 2988  014462  104400              CMP0A:    HLT
 2989  014464  104000                        SCOPE
 2990
 2991
 2992                                        ;CHECK BINARY OPS USING ADDRESS MODE 1
 2993  014466  000402                        BR      .+6             ;RESERVE TWO WORDS
 2994  014470  000000                        .WORD   0               ;RESERVED FOR SOURCE DATA
 2995  014472  000000                        .WORD   0               ;RESERVED FOR DESTINATION DATA
 2996  014474  010704                        MOV     PC,R4
 2997  014476  005744                        TST     -(R4)
 2998  014500  005044                        CLR     -(R4)           ;R4 POINTS TO DESTINATION DATA
 2999  014502  010403                        MOV     R4,R3
 3000  014504  005043                        CLR     -(R3)           ;R3 POINTS TO SOUCE DATA
 3001
 3002  014506  005113                        COM     (R3)            ;(R3)=177777
 3003  014510  005214                        INC     (R4)            ;(R4)=000001
 3004  014512  000262                        SEV                     ;SET V
 3005  014514  061314                        ADD     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=0101
 3006  014516  103002                        BCC     ADD1
 3007  014520  102401                        BVS     ADD1
 3008  014522  001401                        BEQ     .+4
 3009  014524  104400              ADD1:     HLT
 3010
 3011  014526  000277                        SCC
 3012  014530  000250                        CLN
 3013  014532  021314                        CMP     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=1000
 3014  014534  103403                        BCS     CMP1
```

```
3015  014536  102402                  BVS     CMP1$
3016  014540  001401                  BEQ     CMP1$
3017  014542  100401                  BMI     .+4
3018  014544  104400          CMP1:   HLT
3019
3020  014546  000277                  SCC
3021  014550  000244                  CLZ
3022  014552  031314                  BIT     (R3)+,(R4)      ;(R3)=177777,(R4)=000000, CC=0101
3023  014554  103002                  BCC     BIT1$
3024  014556  102401                  BVS     BIT1$
3025  014560  001401                  BEQ     .+4
3026  014562  104400          BIT1:   HLT
3027
3028  014564  000277                  SCC
3029  014566  000245                  +CLC!CLZ
3030  014570  005114                  COM     (R4)            ;(R4)=177777
3031  014572  161314                  SUB     (R3)+,(R4)      ;(R3)=177777,(R4)=000000, CC=0100
3032  014574  103402                  BCS     SUB1$
3033  014576  102401                  BVS     SUB1$
3034  014600  001401                  BEQ     .+4
3035  014602  104400          SUB1:   HLT
3036
3037  014604  105013                  CLRB    (R3)            ;(R3)=177400
3038  014606  000313                  SWAB    (R3)            ;(R3)=000377
3039  014610  000270                  SEN
3040  014612  011314                  MOV     (R3)+,(R4)      ;(R3)=(R4)=000377
3041  014614  100001                  BPL     .+4
3042  014616  104400                  HLT
3043  014620  000314                  SWAB    (R4)            ;(R3)=000377,(R4)=177400
3044  014622  000263                  +SEC!SEV                ;SET C & V
3045  014624  051314                  BIS     (R3)+,(R4)      ;(R3)=000377,(R4)=177777, CC=1001
3046  014626  103002                  BCC     BIS1$
3047  014630  102401                  BVS     BIS1$
3048  014632  100401                  BMI     .+4
3049  014634  104400          BIS1:   HLT
3050
3051  014636  041314                  BIC     (R3)+,(R4)      ;(R3)=000377,(R4)=177400, CC=1001
3052  014640  103002                  BCC     BIC1$
3053  014642  102401                  BVS     BIC1$
3054  014644  100401                  BMI     .+4
3055  014646  104400          BIC1:   HLT
3056
3057  014650  000262                  SEV                     ;SET V
3058  014652  021314                  CMP     (R3),(R4)       ;(R3)=000377,(R4)=177400, CC=0001
3059  014654  103003                  BCC     CMP1A
3060  014656  102402                  BVS     CMP1A
3061  014660  001401                  BEQ     CMP1A
3062  014662  100001                  BPL     .+4
3063  014664  104400          CMP1A:  HLT
3064
3065  014666  005013                  CLR     (R3)            ;(R3)=000000
3066  014670  000261                  SEC
3067  014672  006013                  ROR     (R3)            ;(R3)=100000
3068  014674  011314                  MOV     (R3)+,(R4)      ;(R3)=(R4)=100000
```



```
3069  014676  005114                  COM     (R4)            ;(R4)=077777
3070  014700  161314                  SUB     (R3)+,(R4)      ;(R3)=100000,(R4)=177777, CC=1011
3071  014702  103002                  BCC     SUB1A
3072  014704  102001                  BVC     SUB1A
3073  014706  100401                  BMI     .+4
3074  014710  104400          SUB1A:  HLT
3075
3076  014712  000277                  SCC
3077  014714  161314                  SUB     (R3)+,(R4)      ;(R3)=100000,(R4)=077777, CC=0000
3078  014716  101402                  BLOS    SUB1B           ;BRANCH IF C OR Z IS SET
3079  014720  102401                  BVS     SUB1B
3080  014722  100001                  BPL     .+4
3081  014724  104400          SUB1B:  HLT
3082
3083  014726  011314                  MOV     (R3)+,(R4)      ;(R3)=100000,(R4)=100000, CC=1000
3084  014730  001401                  BEQ     MOV1
3085  014732  100401                  BMI     .+4
3086  014734  104400          MOV1:   HLT
3087
3088  014736  061314                  ADD     (R3)+,(R4)      ;(R3)=100000,(R4)=000000, CC=0111
3089  014740  103003                  BCC     ADD1A
3090  014742  102002                  BVC     ADD1A
3091  014744  001001                  BNE     ADD1A
3092  014746  100001                  BPL     .+4
3093  014750  104400          ADD1A:  HLT
3094
3095  014752  005113                  COM     (R3)            ;(R3)=077777
3096  014754  011314                  MOV     (R3)+,(R4)      ;(R4)=077777
3097  014756  061314                  ADD     (R3)+,(R4)      ;(R3)=077777,(R4)=177776, CC=1010
3098  014760  103402                  BCS     ADD1B
3099  014762  102001                  BVC     ADD1B
3100  014764  100401                  BMI     .+4
3101  014766  104400          ADD1B:  HLT
3102
3103  014770  062714  000002          ADD     #2,(R4)
3104  014774  005714                  TST     (R4)            ;CHECK FINAL RESULT
3105  014776  001401                  BEQ     .+4
3106  015000  104400                  HLT
3107  015002  104000                  SCOPE
3108
3109                          ;CHECK BINARY BYTE OPS USING ADDRESS MODE 1
3110  015004  000402                  BR      .+6
3111  015006  000000                  .WORD   0
3112  015010  000000                  .WORD   0
3113  015012  010705                  MOV     PC,R5
3114  015014  005745                  TST     -(R5)
3115  015016  005045                  CLR     -(R5)           ;(R5)=000000
3116  015020  010502                  MOV     R5,R2
3117  015022  005042                  CLR     -(R2)           ;(R2)=000000
3118  015024  005202                  INC     R2              ;R2 POINTS TO ODD BYTE
3119  015026  105112                  COMB    (R2)            ;(R2)=177400
3120
3121  015030  000277                  SCC
3122  015032  111215                  MOVB    (R2),(R5)       ;(R2)=177400,(R5)=000377,CC=1001
```

```
3123  015034  103005                  BCC     MOVB1
3124  015036  102404                  BVS     MOVB1
3125  015040  001403                  BEQ     MOVB1
3126  015042  100002                  BPL     MOVB1
3127  015044  105215                  INCB    (R5)            ;CHECK RESULT
3128  015046  001401                  BEQ     .+4
3129  015050  104400          MOVB1:  HLT
3130
3131  015052  106312                  ASLB    (R2)+           ;SHIFT (R2) UNTIL
3132  015054  102376                  BVC     .-2             ;(R2)=000000
3133  015056  106012                  RORB    (R2)            ;(R2)=100000
3134  015060  105315                  DECB    (R5)+           ;(R5)=00377
3135  015062  106015                  RORB    (R5)            ;(R5)=000177
3136  015064  000257                  CCC
3137  015066  121512                  CMPB    (R5)+,(R2)      ;(R5)=000177,(R2)=100000, CC=1010
3138  015070  102001                  BVC     CMPB1
3139  015072  100401                  BMI     .+4
3140  015074  104400          CMPB1:  HLT
3141
3142  015076  005003                  CLR     R3
3143  015100  000261                  SEC
3144  015102  006003                  ROR     R3              ;R3=100000
3145  015104  050315                  BIS     R3,(R5)         ;(R5)=100177
3146  015106  000273                  .SEC!.SEVISEN           ;SET C,V, & N
3147  015110  131215                  BITB    (R2)+,(R5)      ;(R2)=100000,(R5)=100177, CC=0101
3148  015112  103002                  BCC     BITB1
3149  015114  102401                  BVS     BITB1
3150  015116  001401                  BEQ     .+4
3151  015120  104400          BITB1:  HLT
3152
3153  015122  151215                  BISB    (R2)+,(R5)      ;(R2)=100000,(R5)=100377, CC=1001
3154  015124  103001                  BCC     BISB1
3155  015126  100401                  BMI     .+4
3156  015130  104400          BISB1:  HLT
3157
3158  015132  141215                  BICB    (R2)+,(R5)      ;(R2)=100000,(R5)=100177, CC=0001
3159  015134  103002                  BCC     BICB1
3160  015136  001401                  BEQ     BICB1
3161  015140  100001                  BPL     .+4
3162  015142  104400          BICB1:  HLT
3163
3164  015144  105112                  COMB    (R2)            ;(R2)=077400,(R5)=100177
3165  015146  121215                  CMPB    (R2)+,(R5)
3166  015150  001401                  BEQ     .+4
3167  015152  104400                  HLT
3168
3169  015154  141512                  BICB    (R5)+,(R2)      ;(R5)=100177,(R2)=000000, CC=0100
3170  015156  001002                  BNE     BICB1A
3171  015160  105712                  TSTB    (R2)
3172  015162  001401                  BEQ     .+4
3173  015164  104400          BICB1A: HLT
3174
3175  015166  000402                  BR      .+6             ;RESERVE TWO WORDS FOR DATA
3176  015170  000000                  .WORD   0               ;SOURCE DATA
```



```
3177  015172  000000                  .WORD   0               ;DEST DATA
3178  015174  012705                  MOV     PC,R5
3179  015176  005745                  TST     -(R5)
3180  015200  105045                  CLRB    -(R5)           ;R5 POINTS TO DEST ODD BYTE
3181  015202  010504                  MOV     R5,R4
3182  015204  105044                  CLRB    -(R4)           ;R4 POINTS TO DEST EVEN BYTE
3183  015206  010403                  MOV     R4,R3
3184  015210  105043                  CLRB    -(R3)           ;R3 POINTS TO SOURCE ODD BYTE
3185  015212  010302                  MOV     R3,R2
3186  015214  105042                  CLRB    -(R2)           ;R2 POINTS TO SOURCE EVEN BYTE
3187
3188
3189
3190  015216  000261                  SEC                     ;SET CARRY
3191                                                          ;(R2),(R3),(R4),(R5)
3192  015220  106112                  ROLB    (R2)            ;0001,0000,0000,0000
3193  015222  111214                  MOVB    (R2),(R4)       ;0001,0000,0001,0000
3194  015224  106112                  ROLB    (R2)            ;0010,0000,0001,0000
3195  015226  111213                  MOVB    (R2),(R3)       ;0010,0010,0001,0000
3196  015230  106112                  ROLB    (R2)            ;0100,0010,0001,0000
3197  015232  111315                  MOVB    (R3)+,(R5)      ;0100,0010,0001,0010
3198  015234  106112                  ROLB    (R2)            ;1000,0010,0001,0010
3199  015236  106113                  ROLB    (R3)            ;1000,0100,0001,0010
3200  015240  151215                  BISB    (R2)+,(R5)      ;1000,0100,0001,1010
3201  015242  131512                  BITB    (R5)+,(R2)      ;1000,0100,0001,1010
3202  015244  001426                  BEQ     BIN1
3203  015246  151314                  BISB    (R3)+,(R4)      ;1000,0100,0101,1010
3204  015250  131413                  BITB    (R4)+,(R3)      ;1000,0100,0101,1010
3205  015252  001423                  BEQ     BIN1
3206  015254  105213                  INCB    (R3)            ;1000,0101,0101,1010
3207  015256  121314                  CMPB    (R3)+,(R4)      ;1000,0101,0101,1010
3208  015260  001020                  BNE     BIN1
3209  015262  106113                  ROLB    (R3)            ;1000,1010,0101,1010
3210  015264  121315                  CMPB    (R3)+,(R5)      ;1000,1010,0101,1010
3211  015266  001015                  BNE     BIN1
3212  015270  106212                  ASRB    (R2)            ;0100,1010,0101,1010
3213  015272  131214                  BITB    (R2)+,(R4)      ;0100,1010,0101,1010
3214  015274  001412                  BEQ     BIN1
3215  015276  106015                  RORB    (R5)            ;0100,1010,0101,0101
3216  015300  121415                  CMPB    (R4)+,(R5)      ;0100,1010,0101,0101
3217  015302  001007                  BNE     BIN1
3218  015304  105314                  DECB    (R4)            ;0100,1010,0100,0101
3219  015306  141214                  BICB    (R2)+,(R4)      ;0100,1010,0000,0101
3220  015310  001004                  BNE     BIN1
3221  015312  111314                  MOVB    (R3)+,(R4)      ;0100,1010,1010,0101
3222  015314  106213                  ASRB    (R3)            ;0100,0101,1010,0101
3223  015316  141315                  BICB    (R3)+,(R5)      ;0100,0101,1010,0101
3224  015320  001401                  BEQ     .+4
3225  015322  104400          BIN1:   HLT
3226  015324  104000                  SCOPE
3227
3228
3229  015326  010405                  MOV     R4,R5           ;SET DESTINATION REGISTER
3230  015330  012715  000001          MOV     #1,(R5)
```

;CHECK BINARY WORD OPS USING ADDRESS MODE 2 & 4

```
3231  015334  012712  1 '777              MOV     #-1,(R2)
3232  015340  000257                      CCC
3233  015342  000262                      SEV
3234  015344  062225                      ADD     (R2)+,(R5)+     ;(R2)-1 '77,(R5)=000000, CC=0101
3235  015346  103002                      BCC     ADD2
3236  015350  102401                      BVS     ADD2
3237  015352  001401                      BEQ     .+4
3238  015354  104400              ADD2:   HLT
3239
3240  015356  000262                      SEV                     ;SET V
3241  015360  024527  000001              CMP     -(R5),#1        ;(R5)=000000, CC=1001
3242  015364  103002                      BCC     CMP2
3243  015366  102401                      BVS     CMP2
3244  015370  100401                      BMI     .+4
3245  015372  104400              CMP2:   HLT
3246
3247  015374  054225                      BIS     -(R2),(R5)+     ;(R2)=177777,(R5)=177777,CC=1001
3248  015376  103001                      BCC     BIS2
3249  015400  100401                      BMI     .+4
3250  015402  104400              BIS2:   HLT
3251  015404  000277                      SCC
3252  015406  000244                      CLZ
3253  015410  162245                      SUB     (R2)+,-(R5)     ;(R2)=177777,(R5)=000000, CC=0100
3254  015412  103402                      BCS     SUB2
3255  015414  102401                      BVS     SUB2
3256  015416  001401                      BEQ     .+4
3257  015420  104400              SUB2:   HLT
3258
3259  015422  005442                      NEG     -(R2)           ;(R2)=000001
3260  015424  005115                      COM     (R5)            ;(R5)=177777
3261  015426  000277                      SCC
3262  015430  000250                      CLN
3263  015432  042225                      BIC     (R2)+,(R5)+     ;(R2)=000001,(R5)=177776, CC=1001
3264  015434  103003                      BCC     BIC2
3265  015436  102402                      BVS     BIC2
3266  015440  001401                      BEQ     BIC2
3267  015442  100401                      BMI     .+4
3268  015444  104400              BIC2:   HLT
3269
3270  015446  012742  125252              MOV     #125252,-(R2)
3271  015452  012245                      MOV     (R2)+,-(R5)
3272  015454  005125                      COM     (R5)+           ;(R5)=052525
3273  015456  000262                      SEV
3274  015460  034245                      BIT     -(R2),-(R5)     ;(R2)=125252,(R5)=052525, CC=0101
3275  015462  103002                      BCC     BIT2
3276  015464  102401                      BVS     BIT2
3277  015466  001401                      BEQ     .+4
3278  015470  104400              BIT2:   HLT
3279
3280  015472  000262                      SEV
3281  015474  052225                      BIS     (R2)+,(R5)+     ;(R2)=125252,(R5)=177777, CC=1001
3282  015476  103002                      BCC     BIS2A
3283  015500  102401                      BVS     BIS2A
3284  015502  100401                      BMI     .+4
```



```
3285  015504  104400              BIS2A:  HLT
3286
3287  015506  042745  125252              BIC     #125252,-(R5)   ;(R5)=052525
3288  015512  005125                      COM     (R5)+           ;(R5)=125252
3289  015514  024245                      CMP     -(R2),-(R5)
3290  015516  001401                      BEQ     .+4
3291  015520  104400                      HLT
3292
3293  015522  005012                      CLR     (R2)
3294  015524  005122                      COM     (R2)+           ;(R2)=177777
3295  015526  162742  000001              SUB     #1,-(R2)        ;(R2)=177776, CC=1000
3296  015532  103402                      BCS     SUB2A
3297  015534  102401                      BVS     SUB2A
3298  015536  100401                      BMI     .+4
3299  015540  104400              SUB2A:  HLT
3300  015542  104000                      SCOPE
3301
3302  015544  010702                      MOV     PC,R2           ;GET CURRENT PC
3303  015546  010205                      MOV     R2,R5           ;MOVE TO R5
3304  015550  124245              1$:     CMPB    -(R2),-(R5)     ;COMPARE ALL PREVIOUS MEMORY ADDRESSES
3305  015552  001401                      BEQ     .+4
3306  015554  104400                      HLT                     ;ERROR!
3307  015556  020237  001010              CMP     R2,@#FRSTAD     ;CHECK FOR LOW LIMIT
3308  015562  001372                      BNE     1$
3309  015564  104000                      SCOPE
3310
3311                                  ;CHECK BINARY BYTE OPS USING ADDRESS MODES 2 & 4.
3312  015566  000402                      BR      .+6             ;RESERVE TWO WORDS
3313  015570  000000                      .WORD   0               ;SOURCE DATA
3314  015572  000000                      .WORD   0               ;DESTINATION DATA
3315  015574  010703                      MOV     PC,R3
3316  015576  005743                      TST     -(R3)
3317
3318                                  ;FIRST CHECK AUTO INCREMENT/DECREMENT
3319  015600  010300                      MOV     R3,R0
3320  015602  010002                      MOV     R0,R2
3321  015604  005302                      DEC     R2
3322  015606  010604                      MOV     SP,R4
3323  015610  010605                      MOV     SP,R5
3324  015612  005745                      TST     -(R5)
3325
3326  015614  114046                      MOVB    -(R0),-(SP)
3327  015616  020506                      CMP     R5,SP
3328  015620  001021                      BNE     BINB
3329  015622  020200                      CMP     R2,R0
3330  015624  001017                      BNE     BINB
3331  015626  122026                      CMPB    (R0)+,(SP)+
3332  015630  020406                      CMP     R4,SP
3333  015632  001014                      BNE     BINB
3334  015634  020003                      CMP     R0,R3
3335  015636  001012                      BNE     BINB
3336  015640  154640                      BISB    -(SP),-(R0)
3337  015642  020506                      CMP     R5,SP
3338  015644  001007                      BNE     BINB
```

```
3339  015646  022200                  CMP     R2,R0
3340  015650  001005                  BNE     BINB
3341  015652  142620                  BICB    (SP)+,(R0)+
3342  015654  020406                  CMP     R4,SP
3343  015656  001002                  BNE     BINB
3344  015660  020003                  CMP     R0,R3
3345  015662  001401                  BEQ     .+4
3346  015664  104400          BINB:   HLT
3347  015666  104000                  SCOPE
3348
3349  015670  010003                  MOV     R0,R3
3350  015672  112743  000200          MOVB    #200,-(R3)
3351  015676  112743  000377          MOVB    #377,-(R3)      ;(R3)=100377
3352  015702  010304                  MOV     R3,R4
3353  015704  112744  000177          MOVB    #177,-(R4)
3354  015710  112744  000000          MOVB    #0,-(R4)        ;(R4)=077400
3355  015714  001401                  BEQ     .+4
3356  015716  104400                  HLT
3357
3358  015720  152324                  BISB    (R3)+,(R4)+     ;(R3)=100377,(R4)=077777
3359  015722  100401                  BMI     .+4
3360  015724  104400                  HLT
3361
3362  015726  122324                  CMPB    (R3)+,(R4)+
3363  015730  103402                  BCS     CMPB2
3364  015732  102001                  BVC     CMPB2
3365  015734  100001                  BPL     .+4
3366  015736  104400          CMPB2:  HLT
3367
3368  015740  000261                  SEC
3369  015742  134344                  BITB    -(R3),-(R4)
3370  015744  103002                  BCC     BITB2
3371  015746  102401                  BVS     BITB2
3372  015750  001401                  BEQ     .+4
3373  015752  104400          BITB2:  HLT
3374
3375  015754  000244                  CLZ
3376  015756  144344                  BICB    -(R3),-(R4)     ;(R3)=100377,(R4)=077400
3377  015760  001401                  BEQ     .+4
3378  015762  104400                  HLT
3379  015764  104000                  SCOPE
3380
3381                          ;CHECK BINARY WORD OPS USING ADDRESS MODES 3 & 5,
3382  015766  000404                  BR      2$              ;RESERVE SPACE FOR DATA AND ADDRESSES
3383  015770  000000          A:WORD  0               ;CONTAINS ADDRESS OF SOURCE DATA
3384  015772  000000          A:WORD  0               ;CONTAINS ADDRESS OF DEST DATA
3385  015774  000000          .WORD   0               ;CONTAINS SOURCE DATA
3386  015776  000000          .WORD   0               ;CONTAINS DEST DATA
3387  016000  010701          2$:     MOV     PC,R1
3388  016002  010100                  MOV     R1,R0           ;SET SCOPE PTR
3389  016004  024040                  CMP     -(R0),-(R0)     ;ADJUST R0
3390  016006  010005                  MOV     R0,R5           ;R5 POINTS TO DEST DATA
3391  016010  024545                  CMP     -(R5),-(R5)     ;SUB 4 FROM R5
3392  016012  010015                  MOV     R0,(R5)         ;R5 POINTS TO ADDRESS OF DEST DATA
```



```
3393  016014  010502                  MOV     R5,R2
3394  016016  010004                  MOV     R0,R4           ;R4 POINTS TO DEST DATA
3395  016020  005740                  TST     -(R0)
3396  016022  010003                  MOV     R0,R3           ;R3 POINTS TO SOURCE DATA
3397  016024  010042                  MOV     R0,-(R2)        ;R2 POINTS TO ADDRESS OF SOURCE DATA
3398  016026  005013                  CLR     (R3)            ;PRESET SOURCE DATA
3399  016030  005014                  CLR     (R4)            ;PRESET DEST DATA
3400
3401  016032  000277                  SCC
3402  016034  000244                  CLZ
3403  016036  163235                  SUB     @(R2)+,@(R5)+   ;(R3)=000000,(R4)=000000, CC=0100
3404  016040  103402                  BCS     SUB3
3405  016042  102401                  BVS     SUB3
3406  016044  001401                  BEQ     .+4
3407  016046  104400          SUB3:   HLT
3408
3409  016050  052752  100000          BIS     #100000,@-(R2)  ;(R3)=100000
3410  016054  062755  000001          ADD     #1,@-(R5)       ;(R4)=000001
3411  016060  163235                  SUB     @(R2)+,@(R5)+   ;(R3)=100000,(R4)=100001, CC=1011
3412  016062  103002                  BCC     SUB3A
3413  016064  102001                  BVC     SUB3A
3414  016066  100401                  BMI     .+4
3415  016070  104400          SUB3A:  HLT
3416
3417  016072  005414                  NEG     (R4)            ;(R4)=077777
3418  016074  035255                  BIT     @-(R2),@-(R5)   ;(R3)=100000,(R4)=077777
3419  016076  001401                  BEQ     .+4
3420  016100  104400                  HLT
3421  016102  023235                  CMP     @(R2)+,@(R5)+
3422  016104  102401                  BVS     .+4
3423  016106  104400                  HLT
3424  016110  005152                  COM     @-(R2)
3425  016112  000257                  CCC
3426  016114  063255                  ADD     @(R2)+,@-(R5)
3427  016116  102001                  BVC     ADD3
3428  016120  100401                  BMI     .+4
3429  016122  104400          ADD3:   HLT
3430  016124  000261                  SEC
3431  016126  045235                  BIC     @-(R2),@(R5)+   ;(R3)=077777,(R4)=100000
3432  016130  103001                  BCC     BIC3
3433  016132  100401                  BMI     .+4
3434  016134  104400          BIC3:   HLT
3435
3436  016136  005155                  COM     @-(R5)          ;(R4)=077777
3437  016140  023235                  CMP     @(R2)+,@(R5)+   ;(R3)=077777,(R4)=077777
3438  016142  001401                  BEQ     .+4
3439  016144  104400                  HLT
3440  016146  104000                  SCOPE
3441
3442                          ;CHECK BINARY BYTE OPS USING ADDRESS MODES 3 & 5,
3443  016150  000406                  BR      1$              ;RESERVE SPACE FOR ADDRESSES & DATA
3444  016152  000000          A:WORD  0               ;CONTAINS ADDRESS OF SOURCE DATA (EVEN BYTE)
3445  016154  000000          .WORD   0               ;CONTAINS ADDRESS OF SOURCE DATA (ODD BYTE)
3446  016156  000000          .WORD   0               ;CONTAINS ADDRESS OF DEST DATA (EVEN BYTE)
```

```
3447  016160  000000                          .WORD   0               ;CONTAINS ADDRESS OF DEST DATA (ODD BYTE)
3448  016162  000000                          .WORD   0               ;CONTAINS SOURCE DATA
3449  016164  000000                          .WORD   0               ;CONTAINS DEST DATA
3450
3451  016166  010700                  1$:     MOV     PC,R0
3452  016170  024040                          CMP     -(R0),-(R0)     ;R0=ADDRESS OF DEST DATA
3453  016172  010003                          MOV     R0,R3           ;R3     "       "
3454  016174  010305                          MOV     R3,R5           ;R5     "       "
3455  016176  005743                          TST     -(R3)           ;SUB 2 FROM R3
3456  016200  010043                          MOV     R0,-(R3)        ;R3 POINTS TO ADDRESS OF DEST DATA
3457  016202  005213                          INC     (R3)            ;ODD BYTE
3458  016204  010043                          MOV     R0,-(R3)        ;EVEN BYTE
3459  016206  010304                          MOV     R3,R4
3460  016210  005740                          TST     -(R0)           ;R0=ADDRESS OF SOURCE DATA
3461  016212  010044                          MOV     R0,-(R4)        ;R4 POINTS TO ADDRESS OF SOURCE DATA
3462  016214  005214                          INC     (R4)            ;ODD BYTE
3463  016216  010044                          MOV     R0,-(R4)        ;EVEN BYTE
3464
3465  016220  000261                          SEC                     ;SET CARRY
3466  016222  012734  177001                  MOV     #177001,@(R4)+
3467  016226  112734  000200                  MOVB    #200,@(R4)+     ;SOURCE DATA=100001
3468  016232  115433                          MOVB    @-(R4),@(R3)+
3469  016234  115433                          MOVB    @-(R4),@(R3)+   ;DEST DATA=000600
3470  016236  103401                          BCS     .+4
3471  016240  104400                          HLT                     ;ERROR! MOV DOES AFFECT C BIT IN PSW
3472  016242  022715  000600                 CMP     #600,(R5)       ;CHECK DEST DATA
3473  016246  001401                          BEQ     .+4
3474  016250  104400                          HLT                     ;ERROR! INCORRECT RESULT
3475  016252  024343                          CMP     -(R3),-(R3)     ;POINT R4 BACK TO EVEN BYTE
3476  016254  153433                          BISB    @(R4)+,@(R3)+
3477  016256  153433                          BISB    @(R4)+,@(R3)+   ;DEST DATA=100601
3478  016260  022715  100601                 CMP     #100601,(R5)    ;CHECK RESULT
3479  016264  001401                          BEQ     .+4
3480  016266  104400                          HLT                     ;ERROR! INCORRECT DEST DATA AFTER BISB
3481  016270  145453                          BICB    @-(R4),@-(R3)
3482  016272  145453                          BICB    @-(R4),@-(R3)
3483  016274  133433                          BITB    @(R4)+,@(R3)+
3484  016276  001002                          BNE     BITB3
3485  016300  133433                          BITB    @-(R4),@(R3)+
3486  016302  001001                          BNE     .+4
3487  016304  104400                  BITB3:  HLT
3488
3489  016306  123453                          CMPB    @(R4)+,@-(R3)
3490  016310  001002                          BNE     CMPB3
```



```
3491  016312  123453                          CMPB    @(R4)+,@-(R3)
3492  016314  001401                          BEQ     .+4
3493  016316  104400                  CMPB3:  HLT
3494  016320  104000                          SCOPE
3495
3496                                  ;CHECK BINARY OPS USING ADDRESS MODE 6
3497  016322  000402                          BR      .+6             ;RESERVE TWO LOCATIONS
3498  016324  000000                  SDATA:  .WORD   0               ;RESERVED FOR SOURCE DATA
3499  016326  000000                  DDATA:  .WORD   0               ;RESERVED FOR DESTINATION DATA
3500
3501  016330  013702  001004                  MOV     @#FACTOR,R2     ;GET RELOCATION FACTOR AND USE AS AN
3502  016334  010205                          MOV     R2,R5           ;INDEX VALUE TO POINT TO DATA
3503  016336  005065  016326                  CLR     DDATA(5)        ;PRESET DESTINATION DATA
3504  016342  012762  000001  016324          MOV     #1,SDATA(2)     ;THIS ROUTSINE PUT A 1 BIT INTO EVERY
3505  016350  056265  016324  016326  1$:     BIS     SDATA(2),DDATA(5)       ;OTHER BIT POSITION IN THE DEST-
3506  016356  006362  016324                  ASL     SDATA(2)                ;INATION ADDRESS (52525)
3507  016362  006362  016324                  ASL     SDATA(2)
3508  016366  103370                          BCC     1$
3509  016370  022765  052525  016326          CMP     #52525,DDATA(5) ;CHECK RESULT
3510  016376  001401                          BEQ     .+4
3511  016400  104400                          HLT                     ;ERROR! INCORRECT RESULT
3512  016402  012762  177777  016324          MOV     #-1,SDATA(2)
3513  016410  046562  016326  016324          BIC     DDATA(5),SDATA(2)       ;SOURCE DATA=125252
3514  016416  036265  016324  016326          BIT     SDATA(2),DDATA(5)
3515  016424  001401                          BEQ     .+4
3516  016426  104400                          HLT                     ;ERROR! BIT INST FAILED
3517  016430  006365  016326                  ASL     DDATA(5)        ;DDATA=125252
3518  016434  026265  016324  016326          CMP     SDATA(2),DDATA(5)
3519  016442  001401                          BEQ     .+4
3520
3521  016444  104400                          HLT                     ;ERROR! CMP INST FAILED
3522  016446  000257                          CCC
3523  016450  066265  016324  016326          ADD     SDATA(2),DDATA(5)
3524  016456  103002                          BCC     ADD6
3525  016460  102001                          BVC     ADD6
3526  016462  100001                          BPL     .+4
3527  016464  104400                  ADD6:   HLT
3528
3529  016466  006362  016324                  ASL     SDATA(2)        ;SDATA=52524
3530  016472  166265  016324  016326          SUB     SDATA(2),DDATA(5)
3531  016500  103401                          BCS     SUB6
3532  016502  001401                          BEQ     .+4
3533  016504  104400                  SUB6:   HLT
3534
3535  016506  112700  000377                  MOVB    #377,R0         ;R0=177777 (MOVB %R EXTENDS SIGN)
3536  016512  010062  016324                  MOV     R0,SDATA(2)
3537  016516  012765  177777  016326          MOV     #-1,DDATA(5)
3538  016524  166500  016326                  SUB     DDATA(5),R0
3539  016530  001401                          BEQ     .+4
3540  016532  104400                          HLT
3541  016534  066265  016324  016326  1$:     ADD     SDATA(2),DDATA(5)
3542  016542  006362  016324                  ASL     SDATA(2)
3543  016546  005162  016324                  COM     SDATA(2)
3544  016552  036265  016324  016326          BIT     SDATA(2),DDATA(5)
```

```
016560  001401                          BEQ     .+4
016562  104400                          HLT
016564  005162  016324                  COM     SDATA(2)
016570  026265  016324  016326          CMP     SDATA(2),DDATA(5)
016576  001401                          BEQ     .+4
016600  104400                          HLT
016602  026200  016324                  CMP     SDATA(2),R0
016606  001352                          BNE     1$
016610  104000                          SCOPE

                                ;CHECK BINARY BYTE OPS USING ADDRESS MODE 6
                                ;NOTE: SDATAB(2), AND DDATAB(4) REFERENCE EVEN BYTE OF SOURCE & DEST DATA
                                ;AND SDATAB(3), AND DDATAB(5) REFERENCE ODD BYTE OF SOURCE & DEST DATA

016612  013702  001004                  MOV     @#FACTOR,R2     ;GET INDEX VALUE
016616  010204                          MOV     R2,R4           ;R2 FOR SOURCE EVEN BYTE INDEX, R4 FOR
016620  010403                          MOV     R4,R3           ;DEST ODD BYTE, R3 FOR SOURCE EVEN
016622  005203                          INC     R3              ;AND R3 FOR DEST ODD BYTE
016624  010305                          MOV     R3,R5
016626  000261                          SEC                     ;SET CARRY
016630  012762  125252  016754          MOV     #125252,SDATAB(2)
016636  112763  177125  016754          MOVB    #177125,SDATAB(3)       ;SOURCE DATA = 052652
016644  016264  016754  016756          MOV     SDATAB(2),DDATAB(4)
016652  052764  125125  016756          BIS     #125125,DDATAB(4)       ;DEST DATA = 177777
016660  136263  016754  016754          BITB    SDATAB(2),$DATAB(3)
016666  001401                          BEQ     .+4
016670  104400                  BITB6:  HLT

016672  146264  016754  016756          BICB    SDATAB(2),DDATAB(4)
016700  103401                          BCS     .+4
016702  104400                          HLT                     ;ERROR MOV,BIS,BIT;BIC DO NOT AFFECT 'C'
016704  126364  016754  016756          CMPB    SDATAB(3),DDATAB(4)
016712  001401                          BEQ     .+4
016714  104400                          HLT

016716  146365  016754  016756          BICB    SDATAB(3),DDATAB(5)
016724  126265  016754  016756          CMPB    SDATAB(2),DDATAB(5)
016732  001401                          BEQ     .+4
016734  104400                          HLT

016736  136564  016756  016756          BITB    DDATAB(5),DDATAB(4)
016744  001401                          BEQ     .+4
016746  104400                          HLT
016750  104000                          SCOPE

016752  000406                          BR      UB7             ;RESERVE TWO WORDS
016754  000000                  SDATAB: .WORD   0               ;RESERVED FOR SOURCE DATA
016756  000000                  DDATAB: .WORD   0               ;RESERVED FOR DEST DATA

                                ;CHECK BINARY WORD OPS USING ADDRESS MODE 7
                                ;R2=ADDRESS OF SOURCE DATA, AND R3= ADDRESS OF DEST DATA
016760  000000                  SBIN7:  .WORD   0               ;CONTAINS ADDRESS OF SOURCE DATA
016762  000000                  DBIN7:  .WORD   0               ;CONTAINS ADDRESS OF DEST DATA
016764  000000                          .WORD   0               ;CONTAINS SOURCE DATA
```



```
016766  000000                          .WORD   0               ;CONTAINS DEST DATA

016770  010700                  UB7:    MOV     PC,R0
016772  024040                          CMP     -(R0),-(R0)
016774  010002                          MOV     R0,R2
016776  024242                          CMP     -(R2),-(R2)
017000  010012                          MOV     R0,(R2)
017002  010203                          MOV     R2,R3
017004  024043                          CMP     -(R0),-(R3)
017006  010013                          MOV     R0,(R3)

017010  000261                          SEC
017012  012777  100000  177740          MOV     #100000,@SBIN7  ;SOURCE DATA = 100000
017020  017777  177734  177734          MOV     @SBIN7,@DBIN7   ;DEST DATA = 100000
017026  103001                          BCC     MOV7
017030  100401                          BMI     .+4
017032  104400                  MOV7:   HLT
017034  006377  177722                  ASL     @DBIN7          ;DEST DATA = 000000
017040  102001                          BVC     .+4
017042  001401                          BEQ     .+4
017044  104400                          HLT

017046  027777  177706  177706          CMP     @SBIN7,@DBIN7   ;(R2)=100000,(R3)=000000
017054  103402                          BCS     CMP7
017056  102401                          BVS     CMP7
017060  100401                          BMI     .+4
017062  104400                  CMP7:   HLT

017064  167777  177670  177670          SUB     @SBIN7,@DBIN7   ;(R2)=100000,(R3)=100000
017072  103003                          BCC     SUB7
017074  102002                          BVC     SUB7
017076  001401                          BEQ     SUB7
017100  100401                          BMI     .+4
017102  104400                  SUB7:   HLT

017104  006277  177650                  ASR     @SBIN7          ;(R2)=140000
017110  067777  177644  177644          ADD     @SBIN7,@DBIN7   ;(R2)=140000,(R3)=040000
017116  103003                          BCC     ADD7
017120  102002                          BVC     ADD7
017122  001401                          BEQ     ADD7
017124  100001                          BPL     .+4
017126  104400                  ADD7:   HLT

017130  047777  177624  177624          BIC     @SBIN7,@DBIN7   ;(R2)=140000,(R3)=000000
017136  001401                          BEQ     .+4
017140  104400                          HLT

017142  057777  177612  177612          BIS     @SBIN7,@DBIN7   ;(R2)=140000,(R3)=140000
017150  100401                          BMI     .+4
017152  104400                          HLT

017154  027777  177600  177600          CMP     @SBIN7,@DBIN7
017162  001401                          BEQ     .+4
017164  104400                          HLT
```

```
3653  017166  104000                  SCOPE
3654
3655                                  ;SOME MISCELLANEOUS OPERATION INVOLVING THE PC
3656                                  ;NOTE: NONE OF THESE OPERATIONS SHOULD AFFECT THE PC
3657  017170  005000                  CLR     R0
3658  017172  005067  000072          CLR     1$
3659  017176  010707                  MOV     PC,PC
3660  017200  120707                  CMPB    PC,PC
3661  017202  030707                  BIT     PC,PC
3662  017204  060007                  ADD     R0,PC
3663  017206  105707                  TSTB    PC
3664  017210  005507                  ADC     PC
3665  017212  021007                  CMP     (R0),PC
3666  017214  131007                  BITB    (R0),PC
3667  017216  062707  000000          ADD     #0,PC
3668  017222  023707  001004          CMP     @#FACTOR,PC
3669  017226  133707  001004          BITB    @#FACTOR,PC
3670  017232  000240                  NOP
3671                                  ;THE NEXT TWO INSTRUCTION CAUSE THE PROGRAM TO JUMP TO THE UNRELOCATED
3672                                  ;CODE AND TO RETURN ON THE FOLLOWING INST (IF THE CODE IS RELOCATED)
3673  017234  163707  001004          SUB     @#FACTOR,PC     ;JUMPS TO UNRELOCATED CODE
3674  017240  063707  001004          ADD     @#FACTOR,PC     ;RETURNS
3675  017244  000240                  NOP
3676  017246  024607                  CMP     -(SP),PC
3677  017250  132607                  BITB    (SP)+,PC
3678  017252  026707  000012          CMP     1$,PC
3679  017256  166707  000006          SUB     1$,PC
3680  017262  046707  000002          BIC     1$,PC
3681  017266  000401                  BR      .+4             ;BRANCH OVER 1$
3682  017270  000000          1$:     0
3683  017272  104000                  SCOPE
3684
3685  017274  010702                  MOV     PC,R2
3686  017276  062702  000012          ADD     #12,R2
3687  017302  012707  001152          MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
3688  017306  000000          REL2:   .WORD   0
3689                                  ;2222222222222 LAST ADDRESS OF CODE TO BE RELOCATED 22222222222
3690
3691
3692                                  .SBTTL  START OF SECTION 3
3693                                  ;3333333333333 FIRST ADDRESS TO BE RELOCATED 333333333
3694  017310  010700          REL3:   MOV     PC,R0           ;GET PC
3695  017312  005740                  TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL3
3696  017314  010037  001010          MOV     R0,@#FRSTAD     ;SAVE
3697  017320  012737  000003  005176  MOV     #3,@#SECT       ;SET SECTION #
3698  017326  004737  005166          JSR     PC,@#LDDISP     ;LOAD DISPLAY REG
3699  017332  013767  005172  002104  MOV     @#DISPLY,REL33
3700  017340  010700                  MOV     PC,R0           ;GET CURRENT PC
3701  017342  162700  017342          SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
3702  017346  010037  001004          MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
3703  017352  010701                  MOV     PC,R1           ;SET NEW SCOPE PTR
3704
3705                                  ;CHECK BINARY BYTE OPS USING ADDRESS MODE 0
3706  017354  012703  125252          MOV     #125252,R3
```



```
3707  017360  010304                  MOV     R3,R4           ;R3=R4=125252
3708  017362  140304                  BICB    R3,R4           ;R3=125252,R4=125000
3709  017364  022704  125000          CMP     #125000,R4      ;CHECK RESULT
3710  017370  001401                  BEQ     .+4
3711  017372  104400                  HLT
3712
3713  017374  005004                  CLR     R4              ;R3=125252,R4=0
3714  017376  150304                  BISB    R3,R4           ;R3=125252,R4=000252
3715  017400  022704  000252          CMP     #252,R4         ;CHECK RESULT
3716  017404  001401                  BEQ     .+4
3717  017406  104400                  HLT
3718
3719  017410  110404                  MOVB    R4,R4           ;R4=177652
3720  017412  022704  177652          CMP     #177652,R4      ;CHECK RESULT
3721  017416  001401                  BEQ     .+4
3722  017420  104400                  HLT
3723
3724  017422  132704  177525          BITB    #177525,R4
3725  017426  001401                  BEQ     .+4
3726  017430  104400                  HLT
3727
3728  017432  105104                  COMB    R4              ;R4=177525
3729  017434  110404                  MOVB    R4,R4           ;R4=000125
3730  017436  022704  000125          CMP     #125,R4         ;CHECK RESULT
3731  017442  001401                  BEQ     .+4
3732  017444  104400                  HLT
3733
3734  017446  150304                  BISB    R3,R4           ;R3=125252,R4=000377
3735  017450  105204                  INCB    R4
3736  017452  001401                  BEQ     .+4
3737  017454  104400                  HLT
3738  017456  104000                  SCOPE
3739
3740                                  ;CHECK BINARY BYTE OPS USING ADDRESS MODE 7
3741  017460  000406                  BR      BINB7           ;RESERVE SPACE FOR ADDRESSES & DATA
3742  017462  000000          SBINB7: .WORD   0               ;CONTAINS ADDRESS OF SOURCE EVEN BYTE
3743  017464  000000                  .WORD   0               ;CONTAINS ADDRESS OF SOURCE ODD BYTE
3744  017466  000000                  .WORD   0               ;CONTAINS ADDRESS OF DEST EVEN BYTE
3745  017470  000000                  .WORD   0               ;CONTAINS ADDRESS OF DEST ODD BYTE
3746  017472  000000          DBINB7: .WORD   0               ;CONTAINS SOURCE DATA
3747  017474  000000                  .WORD   0               ;CONTAINS DEST DATA
3748
3749  017476  010700          BINB7:  MOV     PC,R0
3750  017500  024040                  CMP     -(R0),-(R0)     ;R0 = ADDRESS OF DEST DATA
3751  017502  010060  177772          MOV     R0,-6(R0)       ;LOAD ADDRES OF DEST EVEN BYTE DATA
3752  017506  010060  177774          MOV     R0,-4(R0)
3753  017512  005260  177774          INC     -4(R0)          ;LOAD ADDRESS OF DEST ODD BYTE DATA
```

```
3754 017516 005740                      TST     -(R0)           ;R0=ADDRESS OF SOURCE DATA
3755 017520 010060 177770               MOV     R0,-10(R0)      ;LOAD ADDRESS OF SOURCE EVEN BYTE DATA
3756 017524 010060 177772               MOV     R0,-6(R0)
3757 017530 005260 177772               INC     -6(R0)          ;LOAD ADDRESS OF SOURCE ODD BYTE DATA
3758
3759 017534 005002                      CLR     R2              ;SET INDEX REGISTERS
3760 017536 012703 000002               MOV     #2,R3           ;@SBINB7(2);@SBINB7(3) REFERENCE EVEN &
3761 017542 012704 177774               MOV     #-4,R4          ;ODD BYTE SOURCE DATA; @DBINB7(4);@DBINB7(5)
3762 017546 012705 177776               MOV     #-2,R5          ;REFERENCE DEST EVEN& ODD BYTE DATA
3763
3764
3765 017552 005020                      CLR     (R0)+           ;PRESET SOURCE DATA
3766 017554 005010                      CLR     (R0)            ;PRESET DEST DATA
3767 017556 013746 001004               MOV     @#FACTOR,-(SP)  ;GET RELOCATION FACTOR
3768 017562 061602                      ADD     (SP),R2         ;AND ADD TO INDEX VALUES
3769 017564 061603                      ADD     (SP),R3
3770 017566 061604                      ADD     (SP),R4
3771 017570 062605                      ADD     (SP)+,R5
3772
3773 017572 112773 177777 017462        MOVB    #-1,@SBINB7(3)  ;SRC DATA = 177400
3774 017600 132772 000377 017462        BITB    #377,@SBINB7(2) ;CHECK THAT EVEN BYTE WAS NOT AFFECTED
3775 017606 001401                      BEQ     .+4             ;BY MOVB INSTRUCTION
3776 017610 104400                      HLT
3777
3778 017612 157374 017462 017472        BISB    @SBINB7(3),@DBINB7(4)
3779 017620 105274 017472               INCB    @DBINB7(4)      ;CHECK THAT BIS SET ALL BITS
3780 017624 001401                      BEQ     .+4
3781 017626 104400                      HLT
3782
3783 017630 105375 017472               DECB    @DBINB7(5)      ;DEST DATA = 177400
3784 017634 005274 017472               INC     @DBINB7(4)      ;DEST DATA = 177401
3785 017640 127375 017462 017472        CMPB    @SBINB7(3),@DBINB7(5)
3786 017646 001401                      BEQ     .+4
3787 017650 104400                      HLT
3788
3789 017652 147375 017462 017472        BICB    @SBINB7(3),@DBINB7(5)
3790 017660 001401                      BEQ     .+4
3791 017662 104400                      HLT
3792
3793 017664 105073 017462               CLRB    @SBINB7(3)      ;SRC DATA = 000000
3794                                ;THIS ROUTINE SETS ALL BITS IN THE SOURCE ODD BYTE BY BISING A BIT FROM
3795                                ;THE DEST EVEN BYTE INTO THE SOURCE ODD BYTE
3796 017670 157473 017472 017462 BIS7: BISB    @DBINB7(4),@SBINB7(3)
3797 017676 106174 017472               ROLB    @DBINB7(4)
3798 017702 103372                      BCC     BIS7
3799 017704 022772 177400 017462        CMP     #177400,@SBINB7(2)      ;CHECK RESULT
3800 017712 001401                      BEQ     .+4
3801 017714 104400                      HLT
3802
3803 017716 000372 017462               SWAB    @SBINB7(2)      ;SRC DATA = 000377
3804 017722 112775 000200 017472        MOVB    #200,@DBINB7(5) ;DEST DATA = 100000
3805
3806 017730 147572 017472 017462 BIC7: BICB    @DBINB7(5),@SBINB7(2)
3807 017736 106075 017472               RORB    @DBINB7(5)
```



```
3808 017742 103372                      BCC     BIC7
3809 017744 005772 017462               TST     @SBINB7(2)
3810 017750 001401                      BEQ     .+4
3811 017752 104400                      HLT
3812 017754 104000                      SCOPE
3813
3814 017756 012702 000001       DAERR:  MOV     #1,R2           ;LOAD R2 WITH ODD #
3815 017762 010703                      MOV     PC,R3
3816 017764 000401                      BR      .+4             ;RESERVE SPACE FOR A WORD
3817 017766 000000              .WORD   0                       ;WILL CONTAIN AN ODD ADDRESS
3818 017770 005723                      TST     (R3)+           ;STEP R3 TO POINT TO WORD ABOVE
3819 017772 010313                      MOV     R3,(R3)
3820 017774 005213                      INC     (R3)            ;AND MAKE ODD
3821 017776 012737 020124 000004        MOV     #1$,@#ERRVEC    ;SET ODD ADDRESS & RESERVED INSTRUCTION
3822 020004 063737 001004 000004        ADD     @#FACTOR,@#ERRVEC
3823 020012 013737 000004 000010        MOV     @#ERRVEC,@#RESVEC       ;TO TRAP TO 1$ BELOW
3824
3825 020020 000277                      SCC                     ;SET ALL CC'S
3826 020022 160212                      SUB     R2,(R2)
3827 020024 104400                      HLT
3828 020026 060222                      ADD     R2,(R2)+
3829 020030 104400                      HLT
3830 020032 006342                      ASL     -(R2)
3831 020034 104400                      HLT
3832 020036 106512                      MFPD    (R2)            ;NOTE: MAY BE RESERVED
3833 020040 104400                      HLT
3834 020042 170412                      CLRF    (R2)
3835 020044 104400                      HLT
3836 020046 042202                      BIC     (R2)+,R2
3837 020050 104400                      HLT
3838 020052 164202                      SUB     -(R2),R2
3839 020054 104400                      HLT
3840 020056 155202                      BISB    @-(R2),R2
3841 020060 104400                      HLT
3842 020062 105532                      ADCB    @(R2)+
3843 020064 104400                      HLT
3844 020066 163302                      SUB     @(R3)+,R2
3845 020070 104400                      HLT
3846 020072 005733                      TST     @(R3)+
3847 020074 104400                      HLT
3848 020076 106533                      MFPD    @(R3)+
3849 020100 104400                      HLT
3850 020102 170453                      CLRD    @-(R3)
3851 020104 104400                      HLT
3852 020106 137702 177775               BITB    @.+1,R2
3853 020112 104400                      HLT
3854 020114 105477 177773               NEGB    @.-1
3855 020120 104400                      HLT
3856 020122 000406                      BR      2$
3857
3858 020124 062716 000002       1$:     ADD     #2,(SP)         ;ADJUST RETURN PC
3859 020130 052766 000017 000002        BIS     #17,2(SP)       ;SET CONDITION CODES ON RETURN
3860 020136 000002                      RTI
3861
```

```
3862  020140  012706  020500          2$:     MOV     #STKPTR,SP      ;RESET STACK PTR
3863  020144  012737  005274  000004          MOV     #ERRRT,@#ERRVEC ;RESET TIME OUT VECTOR
3864  020152  012737  005264  000010          MOV     #RESERR,@#RESVEC
3865  020160  104000                          $SCOPE
3866
3867                                  ;CHECK JMP INSTRUCTIONS
3868
3869  020162  010700                          MOV     PC,R0
3870  020164  062700  000012                  ADD     #12,R0          ;SET ADDRESS FOR JMP INST
3871  020170  000277                          SCC                     ;SET CC'S
3872  020172  000110                          JMP     (R0)
3873  020174  000402                          BR      .+6
3874  020176  000250                          CLN                     ;JMP INST JUMPS HERE
3875  020200  000775                          BR      .-4
3876
3877  020202  103003                          BCC     JMP1
3878  020204  102002                          BVC     JMP1
3879  020206  001001                          BNE     JMP1
3880  020210  100001                          BPL     .+4
3881  020212  104400          JMP1:   HLT                             ;ERROR! INCORRECT CC'S AFTER JMP
3882
3883  020214  005002                          CLR     R2              ;SET INDICATOR
3884  020216  010703                          MOV     PC,R3
3885  020220  000401                          BR      .+4             ;RESERVE WORD FOR JMP ADDRESS
3886  020222  000000                          .WORD   0               ;CONTAINS ADDRESS FOR JMP INST
3887  020224  005723                          TST     (R3)+
3888  020226  010313                          MOV     R3,(R3)
3889  020230  010300                          MOV     R3,R0
3890  020232  062713  000022                  ADD     #22,(R3)        ;(R3) IS JMP ADDRESS
3891  020236  010300                          MOV     R3,R0
3892  020240  000133                          JMP     @(R3)+          ;JUMP TO ADDRESS CONTAINED IN R3
3893  020242  000402                          BR      .+6
3894  020244  005102                          COM     R2              ;COMPLEMENT INDICATOR
3895  020246  000775                          BR      .-4
3896  020250  005202                          INC     R2              ;CHECK INDICATOR
3897  020252  001003                          BNE     JMP3
3898  020254  005720                          TST     (R0)+
3899  020256  020003                          CMP     R0,R3           ;CHECK AUTO-INC R3
3900  020260  001401                          BEQ     .+4
3901  020262  104400          JMP3:   HLT
3902
3903  020264  005002                          CLR     R2              ;SET INDICATOR
3904  020266  010704                          MOV     PC,R4           ;SET UP JMP REGISTER
3905  020270  010400                          MOV     R4,R0           ;SET UP CHECK REGISTER
3906  020272  000402                          BR      1$
3907  020274  005102                          COM     R2              ;COMPLEMENT INDICATOR
3908  020276  000403                          BR      2$
3909  020300  022424          1$:     CMP     (R4)+,(R4)+
3910  020302  005724                          TST     (R4)+           ;R4=JMP ADDRESS
3911  020304  000144                          JMP     -(R4)           ;USE R4 AS ADDRESS
3912  020306  005202          2$:     INC     R2              ;CHECK INDICATOR
3913  020310  001003                          BNE     JMP4
3914  020312  022020                          CMP     (R0)+,(R0)+
3915  020314  020004                          CMP     R0,R4           ;CHECK AUTO-DEC R4
```



```
3916  020316  001401                          BEQ     .+4
3917  020320  104400          JMP4:   HLT
3918
3919  020322  010703                          MOV     PC,R3
3920  020324  000401                          BR      .+4             ;RESERVE WORD FOR JMP ADDRESS
3921  020326  000000          1$:     .WORD   0               ;CONTAINS JUMP ADDRESS
3922  020330  005723                          TST     (R3)+
3923  020332  010313                          MOV     R3,(R3)
3924  020334  062723  000016                  ADD     #16,(R3)+
3925  020340  010300                          MOV     R3,R0           ;LOAD CHECK REGISTER
3926  020342  000402                          BR      3$
3927  020344  005102          2$:     COM     R2
3928  020346  000401                          BR      4$
3929  020350  000153          3$:     JMP     @-(R3)          ;JUMP TO 2$ VIA 1$ ABOVE
3930  020352  005202          4$:     INC     R2              ;CHECK INDICATOR
3931  020354  001003                          BNE     JMP5
3932  020356  005740                          TST     -(R0)
3933  020360  020003                          CMP     R0,R3           ;CHECK AUTO-DEC R3
3934  020362  001401                          BEQ     .+4
3935  020364  104400          JMP5:   HLT
3936
3937  020366  000402                          BR      2$
3938  020370  005102          1$:     COM     R2              ;COMPLEMENT INDICATOR
3939  020372  000402                          BR      3$
3940  020374  000167  177770          2$:     JMP     1$
3941  020400  005202          3$:     INC     R2
3942  020402  001401                          BEQ     .+4
3943  020404  104400          JMP6:   HLT
3944
3945  020406  012767  020424  000028          MOV     #1$,7$          ;SET UP JMP ADDRESS
3946  020414  063767  001004  000012          ADD     @#FACTOR,7$     ;ADD RELOCATION FACTOR
3947  020422  000402                          BR      2$              ;GO TO JMP  @7$ INST
3948  020424  005102          1$:     COM     R2              ;COMPLEMENT INDICATOR
3949  020426  000403                          BR      3$              ;GO TO CHECK ROUTINE
3950  020430  000177  000000          2$:     JMP     @7$             ;JMP TO 1$ ABOVE VIA 7$
3951  020434  000000          7$:     .WORD   0               ;CONTAINS JMP ADDRESS
3952  020436  005202          3$:     INC     R2              ;CHECK INDICATOR
3953  020440  001401                          BEQ     .+4
3954  020442  104400          JMP7:   HLT
3955  020444  104000                          $SCOPE
3956
3957                                  ;CHECK JSR INSTRUCTIONS
3958  020446  013705  001004          JSRTST: MOV     @#FACTOR,R5     ;GET RELOCATION FACTOR
3959  020452  012702  020504                  MOV     #3$,R2          ;FORM DEST ADR$
3960  020456  060502                          ADD     R5,R2           ;ADD RELOCATION FACTOR
3961  020460  000277                          SCC                     ;PRESET CC'S
3962  020462  000242                          CLV
3963  020464  004512                          JSR     R5,(R2)         ;GO TO 3$ VIA R2
3964  020466  005702          1$:     TST     R2              ;CHECK INDICATOR
3965  020470  001017                          BNE     JSR1            ;R2 SHOULD=0
3966  020472  023705  001004                  CMP     @#FACTOR,R5     ;CHECK THAT RTS  R5 RESTORED R5
3967  020476  001014                          BNE     JSR1
3968  020500  000414                          BR      JSR1A           ;EXIT TO SCOPE
3969  020502  000205          3$:     RTS     R5              ;RETURN FROM SUBROUTINE
```

```
3970  020504  113011          3$:     BCC     JSR1            ;CHECK THAT JSR DID NOT
3971  020506  112410                  BVS     JSR1
3972  020510  001007                  BNE     JSR1            ;AFFECT CC'S
3973  020512  100006                  BPL     JSR1
3974  020514  005002                  CLR     R2              ;CLEAR INDICATOR
3975  020516  012704  020466          MOV     #1$,R4          ;GET UNRELOCATED RETURN ADDRESS
3976  020522  061604                  ADD     (SP),R4         ;ADD RELOCATION FACTOR (OLD R5)
3977  020524  020405                  CMP     R4,R5           ;CHECK THAT OLD R5 WAS PLACED ON THE
3978  020526  001765                  BEQ     2$              ;STACK,& THAT NEW R5 CONTAINS RETURN PC
3979  020530  104400          JSR1:   HLT                     ;ERROR! ABOVE
3980
3981  020532  013704  001004  JSR1A:  MOV     @#FACTOR,R4     ;GET RELOCATION FACTOR
3982  020536  005000                  CLR     R0              ;SET INDICATOR
3983  020540  012705  020560          MOV     #1$,R5
3984  020544  060405                  ADD     R4,R5           ;SET UP JSR DEFERRED ADRS
3985  020546  010502                  MOV     R5,R2
3986  020550  012715  020576          MOV     #5$,(R5)
3987  020554  060415                  ADD     R4,(R5)         ;(R5)=DEST ADRS
3988  020556  000401                  BR      2$              ;RESERVE WORD FOR ADDRESS
3989  020560  000000          1$:     .WORD   0               ;CONTAINS DEST ADRS FOR JSR
3990  020562  004435          2$:     JSR     R4,@(R5)+       ;JSR TO 5$ VIA 1$ ABOVE
3991  020564  005200          3$:     INC     R0              ;CHECK INDICATOR
3992  020566  001013                  BNE     JSR3
3993  020570  000413                  BR      JSR3A
3994  020572  005100          4$:     COM     R0              ;COMPLEMENT INDICATOR
3995  020574  000204                  RTS     4               ;RETURN FROM SUBROUTINE
3996  020576  012703  020564  5$:     MOV     #3$,R3          ;GET UNRELOCATED RETURN ADDRESS
3997  020602  061603                  ADD     (SP),R3         ;ADD RELOCATION FACTOR (OLD R4)
3998  020604  020403                  CMP     R4,R3
3999  020606  001003                  BNE     JSR3
4000  020610  005722                  TST     (R2)+
4001  020612  020205                  CMP     R2,R5           ;CHECK AUTO-INC R5
4002  020614  001766                  BEQ     4$              ;GO TO RTS
4003  020616  104400          JSR3:   HLT                     ;ERROR ABOVE
4004
4005  020620  013704  001004  JSR3A:  MOV     @#FACTOR,R4
4006  020624  012405                  MOV     R4,R5
4007  020626  010703                  MOV     PC,R3
4008  020630  000401                  BR      2$
4009  020632  000405          1$:     BR      4$
4010  020634  022323          2$:     CMP     (R3)+,(R3)+
4011  020636  000277                  SCC
4012  020640  004443                  JSR     R4,-(R3)        ;GO TO 2$
4013  020642  104400          3$:     HLT
4014  020644  000414                  BR      JSR0A
4015  020646  103012          4$:     BCC     JSR0
4016  020650  102011                  BVC     JSR0
4017  020652  001010                  BNE     JSR0
4018  020654  100007                  BPL     JSR0
4019  020656  012702  020642          MOV     #3$,R2          ;GET UNRELOCATED RETURN ADDRESS
4020  020662  061602                  ADD     (SP),R2         ;ADD RELOCATION FACTOR (OLD R4)
4021  020664  020204                  CMP     R2,R4           ;CHECK THAT CALCULATED RETURN
4022  020666  001002                  BNE     JSR0            ;PC = NEW R4
4023  020670  005724                  TST     (R4)+
```



```
4024  020672  000204                  RTS     R4
4025  020674  104400          JSR0:   HLT
4026
4027
4028  020676  000401          JSR4A:  BR      2$
4029  020700  000405          1$:     BR      3$
4030  020702  010700          2$:     MOV     PC,R0
4031  020704  004767  177770          JSR     PC,1$
4032  020710  100407                  BMI     JSR0A
4033  020712  104400                  HLT
4034  020714  022020          3$:     CMP     (R0)+,(R0)+
4035  020716  020016                  CMP     R0,(SP)         ;CHECK THAT RETURN ADDRESS IS ON THE
4036  020720  001401                  BEQ     .+4             ;STACK
4037  020722  104400                  HLT
4038  020724  000270                  SEN                     ;SET N
4039  020726  000207                  RTS     PC
4040  020730  104000          JSR0A:  $SCOPE
4041
4042                                  ;CHECK IOT TRAP (AND ROLB/ASLB)
4043                                  ;THIS TEST CHECKS THAT THE PSW IS CORRECT AFTER THE IOT AND THAT THE
4044                                  ;'NEW'PSW (FROM IOTVEC+2) IS CORRECT,
4045  020732  012705  000020  IOTTST: MOV     #IOTVEC,R5              ;SET R5=ADDRESS OF IOTVECTOR
4046  020736  010746                  MOV     PC,-(SP)
4047  020740  062716  000040          ADD     #1$-.,(SP)
4048  020744  012625                  MOV     (SP)+,(R5)+             ;LOAD IOT TRAP VECTOR
4049  020746  005000                  CLR     R0
4050  020750  052740  000200          BIS     #PRTY4,-(R0)            ;SET PRIORITY LEVEL 4 IN PSW
4051                                                                  ;PSW=X XXX X00 001 1X1 000
4052  020754  011015                  MOV     (R0),(R5)               ;SET IOTVEC+2=PSW ABOVE
4053  020756  011504                  MOV     (R5),R4                 ;SAVE IN R4
4054  020760  042710  000357          BIC     #PRTY7+17,(R0)
4055  020764  052710  000144          BIS     #PRTY3+Z,(R0)           ;PSW=X XXX X00 001 1X1 000
4056  020770  012003                  MOV     (R0)+,R3                ;R3 = PSW ABOVE
4057  020772  010340                  MOV     R3,-(R0)
4058  020774  000004                  IOT
4059  020776  104400          10$:    HLT                             ;ERROR! IOT FAILED TO TRAP
4060
4061  021000  012002          1$:     MOV     (R0)+,R2                ;GET PSW AFTER IOT TRAP
4062                                                                  ;NOTE! R0=0
4063  021002  012715  000200          MOV     #PRTY4,(R5)             ;RESTORE IOTVEC+2
4064  021006  012745  002564          MOV     #.TYPE,-(R5)            ;AND IOTVEC
4065  021012  010746                  MOV     PC,-(SP)                ;FORM PC OF 10$ ABOVE
4066  021014  062716  177762          ADD     #10$-.,(SP)
4067  021020  022626                  CMP     (SP)+,(SP)+             ;CHECK RETURN PC ON STACK
4068  021022  001036                  BNE     99$
4069  021024  022603                  CMP     (SP)+,R3                ;CHECK SAVED PSW
4070  021026  001034                  BNE     99$
4071  021030  032703  140000          BIT     #UM,R3                  ;BRANCH TO 3$ IF IN USER MODE
4072  021034  100413                  BMI     3$
4073  021036  001003                  BNE     2$                      ;BRANCH TO 2$ IF IN SUPER MODE
4074  021040  020204                  CMP     R2,R4                   ;CHECK PSW AFTER IOT
4075  021042  001026                  BNE     99$
4076  021044  000413                  BR      4$
4077
```

```
4078  021046  042704  000000          2$:     BIC     #PUM,R4                 ;CLEAR PREV MODE BITS
4079  021052  052704  010000                  BIS     #PSM,R4                 ;SET PREV SUPER MODE
4080  021056  020204                          CMP     R2,R4                   ;CHECK PSW AFTER IOT
4081  021060  001017                          BNE     99$
4082  021062  000404                          BR      4$
4083
4084  021064  052704  030000          3$:     BIS     #PUM,R4                 ;SET PREV USER MODE
4085  021070  020204                          CMP     R2,R4                   ;CHECK PSW AFTER IOT
4086  021072  001012                          BNE     99$
4087
4088  021074  005002                  4$:     CLR     R2
4089  021076  000261                          SEC
4090  021100  106100                          ROLB    R0                      ;ROTATE R0
4091  021102  102376                          BVC     .-2                     ;UNTIL V SETS (R0=200)
4092
4093  021104  106300                          ASLB    R0                      ;SHIFT SHOULD SET CARRY
4094  021106  103004                          BCC     99$
4095  021110  102003                          BVC     99$
4096  021112  001002                          BNE     99$
4097  021114  005700                          TST     R0
4098  021116  001401                          BEQ     .+4
4099  021120  104400                  99$:    HLT                             ;ERROR! ROL/ASL FAILED TO SET
4100                                                                          ;CC'S PROPERLY (IF R2=0) OR IN-
4101                                                                          ;CORRECT PSW AFTER IOT (IF R2 NOT 0)
4102  021122  042704  000340                  BIC     #PRTY7,R4
4103  021126  010437  177776                  MOV     R4,@#PSW                ;RESTORE PSW
4104  021132  012706  000500                  MOV     #STKPTR,SP              ;RESTORE STACK PTR
4105  021136  104000                          SCOPE
4106
4107                                  ;CHECK EMT TRAP SEQUENCE
4108  021140  005000                          CLR     R0
4109  021142  010746                          MOV     PC,-(SP)
4110  021144  062716  000030                  ADD     #EMT1-.,(SP)
4111  021150  012637  000030                  MOV     (SP)+,@#EMTVEC
4112  021154  000262                          SEV                             ;SET V
4113  021156  013737  177776  000032          MOV     @#PSW,@#EMTVEC+2        ;RETAIN CURRENT PSW ON TRAP
4114  021164  000265                          +SE2!SEC
4115  021166  104000                          EMT                     ;TRAP TO EMT1
4116  021170  001433                          BEQ     EMT1C           ;GO TO EMT1C
4117  021172  104400                          HLT                     ;ERROR! INCORRECT CC'S WERE SET ON RETURN
4118  021174  102027                  EMT1:   BVC     EMT1B           ;'V' SHOULD'VE SET ON EMT TRAP
4119  021176  105100                          COMB    R0              ;R0=000377,CC'S=1001
4120  021200  105500                          ADCB    R0              ;R0=000000,CC'S=0101
4121  021202  106000                          RORB    R0              ;R0=000200,CC'S=1010
4122  021204  102023                          BVC     EMT1B
4123  021206  100022                          BPL     EMT1B
4124  021210  000257                          CCC
4125  021212  105400                          NEGB    R0              ;R0=000200,CC'S=1010
4126  021214  102017                          BVC     EMT1B
4127  021216  100016                          BPL     EMT1B
4128  021220  000242                          CLV                     ;CLEAR 'V'
4129  021222  000261                          SEC                     ;AND SET 'C'
4130  021224  105300                          DECB    R0              ;R0=000177,CC'S=0011
4131  021226  102012                          BVC     EMT1B
```



```
4132  021230  100411                          BMI     EMT1B
4133  021232  000242                          CLV                     ;CLEAR 'V'
4134  021234  105200                          INCB    R0              ;R0=000200,CC'S=1011
4135  021236  103006                          BCC     EMT1B
4136  021240  102005                          BVC     EMT1B
4137  021242  100004                          BPL     EMT1B
4138  021244  000242                          CLV                     ;CLEAR 'V'
4139  021246  106200                          ASRB    R0              ;SHIFT R0 UNTIL 'V' CLEARS
4140  021250  102776                          BVS     .-2
4141  021252  000401                          BR      .+4
4142  021254  104400                  EMT1B:  HLT                     ;ERROR!
4143  021256  000002                          RTI                     ;EXIT WITH R0=000377
4144  021260  105500                  EMT1C:  ADCB    R0              ;R0=000000
4145  021262  103003                          BCC     EMT1D
4146  021264  001002                          BNE     EMT1D
4147  021266  005700                          TST     R0
4148  021270  001401                          BEQ     .+4
4149  021272  104400                  EMT1D:  HLT
4150  021274  012737  001014  000030          MOV     #SCOPEA,@#EMTVEC        ;RESTORE EMT TO SCOPE
4151  021302  005037  000032                  CLR     @#EMTVEC+2
4152  021306  104000                          SCOPE
4153
4154                                  ;CHECK TRAP INSTRUCTION TRAP SEQUENCE
4155          104000                          HLT=EMT                         ;REDEFINE HLT
4156  021310  013737  000034  000030          MOV     @#TRAPVEC,@#EMTVEC      ;SET EMT (HLT) TRAP VECTOR
4157  021316  010746                          MOV     PC,-(SP)
4158  021320  062716  000042                  ADD     #TRAP1-.,(SP)
4159  021324  012637  000034                  MOV     (SP)+,@#TRAPVEC
4160  021330  000270                          SEN                             ;SET N
4161  021332  013737  177776  000036          MOV     @#PSW,@#TRAPVEC+2       ;RETAIN CURRENT PSW ON TRAP
4162  021340  000261                          SEC                             ;SET CARRY
4163  021342  010700                          MOV     PC,R0
4164  021344  000264                          SEZ                     ;SET Z BIT
4165  021346  104400                          TRAP                    ;TRAP TO TRAP1
4166  021350  103401                          BCS     .+4
4167  021352  104000                          HLT
4168  021354  001401                          BEQ     .+4
4169  021356  104000                          HLT
4170  021360  000412                          BR      TRAP1C
4171  021362  100401                  TRAP1:  BMI     .+4             ;N BIT GOT SET ON TRAP
4172  021364  104000                          HLT
4173  021366  062700  000004                  ADD     #4,R0
4174  021372  020016                          CMP     R0,(SP)         ;CHECK LOW BYTE OF RETURN PC ON
4175  021374  001401                          BEQ     .+4             ;STACK
4176  021376  104000                          HLT
4177  021400  124646                          CMPB    -(SP),-(SP)
4178  021402  032626                          BIT     (SP)+,(SP)+
4179  021404  000002                          RTI                             ;RETURN TO INST FOLLOWING TRAP (1$)
4180
4181  021406  012702  000036          TRAP1C: MOV     #TRAPVEC+2,R2           ;RESTORE VECTORS
4182  021412  012712  000340                  MOV     #PRTY7,(R2)
4183  021416  012742  003212                  MOV     #,HLT,-(R2)
4184  021422  005042                          CLR     -(R2)
4185  021424  012742  001214                  MOV     #SCOPEA,-(R2)
```

```
                                        SCOPE
                                        HLT=TRAP                ;RESTORE HLT TO A TRAP INST

                        010702          MOV     PC,R2
                        062702  000012  ADD     #12,R2
                        012707  001152  MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
                        000000  REL33:  .WORD   0
                                ;3333333333333 LAST ADDRESS OF CODE TO BE RELOCATED 33333333333

                        010701          MOV     PC,R1           ;SET SCOPE POINTER
                        122737  000004  000764  CMPB    #4,@#OPT,CP             ;BRANCH IF 11/40 OR 11/45
                        101405          BLOS    REL4
                        012737  000002  031204  MOV     #RTI,@#RTI1             ;SET 'T' TRAP RETURN TO RTI
                        000137  027640  JMP     @#TTYCHK                ;JUMP IF 11/05 OR 11/20

                                        .SBTTL  START OF SECTION 4
                                ;444444444444 FIRST ADDRESS TO BE RELOCATED 444444444
                        010700  REL4:   MOV     PC,R0           ;GET PC
                        005740          TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL4
                        010037  001010  MOV     R0,@#FRSTAD     ;SAVE
                        012737  000004  005176  MOV     #4,@#SECT       ;SET SECTION #
                        004737  005166  JSR     PC,@#LDDISP     ;LOAD DISPLAY GEG
                        013767  005172  001370  MOV     @#DISPLY,REL44
                        010700          MOV     PC,R0           ;GET CURRENT PC
                        162700  021524  SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
                        010037  001004  MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
                        010701          MOV     PC,R1           ;SET NEW SCOPE PTR

                                ;CHECK STACK OVERFLOW
                        013767  177776  000332  OVFLW: MOV     @#PSW,7$        ;SAVE STATUS IN 7$ BELOW
                        005037  177776          CLR     @#PSW           ;SET KERNEL MODE
                        004737  002532          JSR     PC,@#CLRTBIT    ;GO CLEAR 'T' BIT IF SET
                        052737  000340  177776  BIS     #PRTY7,@#PSW    ;SET PRIORITY LEVEL 7 TO BLOCK CLOCK
                        010746          MOV     PC,-(SP)        ;PUSH CURRENT PC ONTO STACK
                        062716  000146  ADD     #2$-.,(SP)      ;FORM ADDRESS OF 2$ BELOW
                        011637  000004  MOV     (SP),@#ERRVEC   ;SET ERROR VECTOR
                        012737  000340  000006  MOV     #340,@#ERRVEC+2 ;SET PRIORITY LEVEL 7 ON TRAP
                        013727  000016  MOV     @#BPTVEC+2,(PC)+        ;SAVE CONTENTS OF BPT VECTOR +2
                        000000  42$:    .WORD   0
                        062716  000100  ADD     #41$-2$,(SP)    ;FORM ADDRESS OF 41$ BELOW
                        011637  000014  MOV     (SP)+,@#BPTVEC  ;SET BPT TRAP VECTOR TO 41$
                        012737  000340  000016  MOV     #340,@#BPTVEC+2

                        012703  000376          MOV     #376,R3
                        010313          MOV     R3,(R3)         ;LOAD 376 INTO ADDRESS 376
                        010306          MOV     R3,SP           ;SET STACK PTR AT BOUNDARY
                        032767  140000  000232  BIT     #UM,7$          ;CHECK IF ENTERED TEST IN KERNEL
                        001015          BNE     1$              ;MODE, BRANCH IF NOT IN KERNEL

                                ;THE BELOW INSTRUCTIONS SHOULD NOT CAUSE AN OVERFLOW TRAP
                        005716          TST     (SP)            ;BECAUSE TST IS A NON MODIFYING INST
                        021666  177776  CMP     (SP),-2(SP)     ;SO IS COMPARE
                        012656          MOV     (SP)+,@-(SP)    ;BECAUSE OF ADDRESS MODE 5
                        057636  000000  BIS     @(SP),@(SP)+    ;BECAUSE OF ADDRESS MODE 3
```



```
                        054676  000000          BIS     -(SP),@(SP)     ;BECAUSE OF ADDRESS MODE 7
                        005006          CLR     SP
                        013766  020000  020000  MOV     @#20000,20000(SP)
                        000423          BR      3$              ;BRANCH OVER NON KERNEL MODE TESTS

                                ;NOTE:  NO OVERFLOW TRAP WILL OCCUR IF NOT IN KERNEL MODE!!!
                        156737  000173  177777  1$:     BISB    7$+1,@#PSW+1    ;RESTORE MODE BITS IN PSW
                        012706  000376          MOV     #376,SP         ;SET STACK PTR
                        016646  177776          MOV     -2(SP),-(SP)    ;SHOULD NOT TRAP
                        051616          BIS     (SP),(SP)
                        061666  177776          ADD     (SP),-2(SP)
                        105037  177777          CLRB    @#PSW+1         ;SET KERNEL MODE
                        000451          BR      6$              ;EXIT TEST

                                ;ERROR SERVICE ROUTINE
                        012600          2$:     MOV     (SP)+,R0        ;SAVE PC OF INSTRUCTION THAT TRAPPED
                        012602          MOV     (SP)+,R2        ;SAVE PSW
                        012706  000500  MOV     #STKPTR,SP      ;SET STACK PTR
                                        HLT                     ;ERROR! AN INSTRUCTION THAT WAS NOT
                                                                ;SUPPOESED TO TRAP TRAPPED
                                                                ;R0 CONTAINS PC, R2 CONTAINS PSW
                        000443          BR      6$              ;EXIT TEST
                                ;THE BELOW INSTRUCTIONS WILL CAUSE A STACK OVERFLOW
                                ;STACK PTR IS AT 376
                        062737  000066  000004  3$:     ADD     #4$-2$,@#ERRVEC ;SET ERROR VECTOR TO 4$
                        010306          MOV     R3,SP           ;SET STACK PTR AT 376
                        112702  000001  MOVB    #1,R2
                        005000          CLR     R0
                        005016          CLR     (SP)            ;SETS BIT 0 IN R0
                        006302          ASL     R2              ;SHIFT INDICATOR BIT
                        105226          INCB    (SP)+           ;SETS BIT 1 IN R0
                        006302          ASL     R2
                        060746          ADD     PC,-(SP)        ;SETS BIT 2 IN R0
                        006302          ASL     R2
                        000003          BPT                     ;SETS BIT 3 IN R0
                        006302          ASL     R2
                        004767  000014  JSR     PC,40$          ;SETS BIT 4 IN R0
                        006302          ASL     R2
                        050666  177776  BIS     SP,-2(SP)       ;SETS BIT 5 IN R0
                        000410          BR      5$

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

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

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

                                ;CHECK THAT ABOVE INSRUCTIONS DID TRAP
                        012706  000500  5$:     MOV     #STKPTR,SP      ;SET STACK PTR
                        022700  001477          CMP     #1477,R0        ;EACH INSTRUCTION SET A BIT IN R0
```

```
4294  022050  001401                        BEQ     .+4             ;R0= 1477
4295  022052  104400                        HLT
4296
4297                                        ;EXIT ROUTINE
4298  022054  012706  000600          6$:   MOV     #KPTR,SP        ;SET KERNEL STACK PTR
4299  022060  012737  000016  000014        MOV     #BPTVEC+2,@#BPTVEC
4300  022066  016737  177514  000016        MOV     42$,@#BPTVEC+2
4301  022074  012746                        MOV     (PC)+,-(SP)     ;PUSH OLD PSW ONTO STACK
4302  022076  000000                  7$:   .WORD   0               ;CONTAINS SAVED PSW
4303  022100  010746                        MOV     PC,-(SP)        ;PUSH CURRENT PC ONTO STACK
4304  022102  062716  000006                ADD     #6,(SP)         ;ADD OFFSET
4305  022106  000002                        RTI
4306  022110  012706  000500                MOV     #STKPTR,SP      ;SET STACK PTR
4307  022114  012737  005274  000004        MOV     #ERRPRT,@#ERRVEC ;RESET TIME OUT VECTOR
4308  022122  012737  000002  000006        MOV     #RTI,@#ERRVEC+2
4309  022130  104000                        SCOPE
4310
4311                                        ;CHECK THAT ALL RESERVED INSTRUCTIONS TRAP (TO LOCATION 10)
4312  022132  012702  022236          RESTRP: MOV     #5$,R2          ;GET ADDRESS OF RESERVED INSTRUCTION TABLE
4313  022136  063702  001004                ADD     @#FACTOR,R2
4314  022142  132737  000040  000765        BITB    #40,@#OPT,CP+1  ;CHECK IF 11/45 FLOATING POINT IS AVAIL.
4315  022150  001402                        BEQ     .+6             ;BRANCH IF NOT AVAILABLE
4316  022152  005067  000110                CLR     5@$             ;SET TABLE TERMINATOR AT GROUP 7
4317  022156  012737  022214  000010        MOV     #4$,@#RESVEC    ;SET RESERVED INSTRUCTION TRAP
4318  022164  063737  001004  000010        ADD     @#FACTOR,@#RESVEC
4319  022172  012203                  1$:   MOV     (R2)+,R3        ;GET FIRST RESERVED INSTRUCTION
4320  022174  001437                        BEQ     7$              ;0 TERMINATES THE TABLE
4321  022176  012204                        MOV     (R2)+,R4        ;GET LAST RESERVED INSTRUCTION IN GROUP
4322  022200  010317                  2$:   MOV     R3,(PC)         ;EXECUTE RESERVED INSTRUCTION
4323  022202  000000                  3$:   .WORD   0               ;CONTAINS RESERVED INSTRUCTION
4324  022204  104400                        HLT                     ;ERROR! INSTRUCTION IN R3
4325  022206  104400                        HLT                     ;(2$) ABOVE FAILED TO CAUSE A
4326  022210  104400                        HLT                     ;RESERVED INSTRUCTION TRAP
4327  022212  000405                        BR      41$
4328  022214  012716  022226          4$:   MOV     #41$,(SP)       ;ADJUST RETURN PC
4329  022220  063716  001004                ADD     @#FACTOR,(SP)   ;TO RETURN TO 41$
4330  022224  000002                        RTI                     ;RETURN TO 41$
4331  022226  020304                  41$:  CMP     R3,R4           ;HAS GROUP OF RESERVED INSTRUCTIONS
4332  022230  001760                        BEQ     1$              ;BEEN EXECUTED
4333  022232  005203                        INC     R3              ;INCREMENT THIS RESERVED INSTRUCTION
4334  022234  000761                        BR      2$              ;TO NEXT ONE AND EXECUTE
4335                                        ;TABLE OF 11/40,11/45 RESERVED INSTRUCTIONS (0 TERMINATES THE TABLE)
4336  022236  000007                  5$:   7                       ;GROUP 1
4337  022240  000077                        77                      ;  "
4338  022242  000210                        210                     ;GROUP 2
4339  022244  000227                        227                     ;  "
4340  022246  007000                        7000                    ;GROUP 3
4341  022250  007777                        7777                    ;  "
4342  022252  075040                        75040                   ;GROUP 4
4343  022254  076777                        76777                   ;  "
4344  022256  106400                        106400                  ;GROUP 5
4345  022260  106477                        106477                  ;  "
4346  022262  106700                        106700                  ;GROUP 6
4347  022264  107777                        107777                  ;  "
```



```
4348  022266  170000                  6@$:  170000                  ;GROUP 7        FLOATING POINT
4349  022270  177777                        177777                  ;  "            INSTRUCTIONS
4350  022272  000000                        0                       ;0 TERMINATES THE TABLE
4351
4352  022274  012737  005264  000010  7$:   MOV     #RESERR,@#RESVEC        ;RESTORE RESERVED TRAP
4353  022302  104000                        SCOPE
4354
4355                                        ;CHECK THAT ALL BITS IN THE PROCESSER STATUS WORD (PSW) CAN BE SET AND
4356                                        ;CLEARED.
4357  022304  105737  000770          PSWCHK: TSTB    @#MMON          ;IF MEM MGMT IS ON SKIP THIS TEST
4358  022310  001072                        BNE     4$
4359  022312  013767  177776  000144        MOV     @#PSW,3$        ;SAVE STATUS
4360  022320  005037  177776                CLR     @#PSW           ;CLEAR MODE BITS IN PSW
4361  022324  004737  002532                JSR     PC,@#CLRTBIT    ;GO CLEAR 'T' BIT IF SET
4362  022330  013746  000016                MOV     @#TBITVEC+2,-(SP)
4363  022334  012704  177776                MOV     #PSW,R4         ;LOAD ADDRESS OF PSW INTO R4
4364  022340  000250                        CLN
4365  022342  005714                        TST     (R4)            ;CHECK THAT PSW WAS CLEARED
4366  022344  001401                        BEQ     .+4
4367  022346  104400                        HLT                     ;ERROR! PSW FAILED TO CLEAR
4368  022350  113700  000764                MOVB    @#OPT,CP,R0     ;GET CP TYPE
4369  022354  016000  032240                MOV     PSWBIT(0),R0    ;GET BIT MASK FOR TEST R0=THOSE BITS IN
4370                                                                ;THE PSW WHICH CAN BE SET/CLEARED.
4371  022360  005737  000764                TST     @#OPT,CP        ;CHECK IF MEM MGMT IS AVAILABLE
4372  022364  100002                        BPL     10$             ;BRANCH IF NOT AVAILABLE
4373  022366  052700  170000                BIS     #170000,R0      ;SET BITS 15-12 IF MEM MGMT
4374  022372  012702  000001          10$:  MOV     #1,R2           ;R2 = TEST BIT
4375  022376  030200                  1$:   BIT     R2,R0           ;CHECK IF BIT CAN BE SET/CLEARED
4376  022400  001423                        BEQ     2$
4377  022402  005037  000016                CLR     @#TBITVEC+2
4378  022406  030227  000020                BIT     R2,#20          ;CHECK IF TEST WILL SET 'T' BIT
4379  022412  001403                        BEQ     20$
4380  022414  012737  000002  000016        MOV     #RTI,@#TBITVEC+2;SET RTI INTO RETURN
4381  022422  005014                  20$:  CLR     (R4)            ;CLEAR PSW
4382  022424  050214                        BIS     R2,(R4)         ;SET R2 INTO PSW
4383  022426  011403                        MOV     (R4),R3         ;GET BIT
4384  022430  020203                        CMP     R2,R3           ;CHECK THAT BIT WAS SET IN PSW
4385  022432  001401                        BEQ     .+4
4386  022434  104400                        HLT                     ;ERROR! BIT IN R2 FAILED TO SET IN PSW
4387  022436  000244                        CLZ                     ;CLEAR Z BIT
4388  022440  040214                        BIC     R2,(R4)         ;CLEAR BIT IN PSW
4389  022442  011403                        MOV     (R4),R3         ;GET PSW RESULT
4390  022444  001401                        BEQ     2$              ;BRANCH IF BIC ABOVE CLEARED BIT IN PSW
4391  022446  104400                        HLT                     ;ERROR! BIT IN R2 FAILED TO CLEAR IN PSW
4392  022450  006302                  2$:   ASL     R2              ;SHIFT TEST BIT
4393  022452  103351                        BCC     1$              ;BRANCH IF ALL BITS NOT TESTED
4394  022454  005014                        CLR     (R4)            ;CLEAR STATUS
4395  022456  012637  000010                MOV     (SP)+,@#TBITVEC+2       ;RESTORE T BIT RETURN
4396  022462  012746                        MOV     (PC)+,-(SP)     ;PUSH ORIGINAL STATUS ON STACK
4397  022464  000000                  3$:   .WORD   0               ;CONTAINS ORIGINAL PSW
4398  022466  010746                        MOV     PC,-(SP)        ;SET RETURN PC
4399  022470  062716  000006                ADD     #6,(SP)
4400  022474  000002                        RTI                     ;RETURN
4401  022476  104000                  4$:   SCOPE
```

```
4402
4403  022500  013704  177776          MOV     @#PSW,R4        ;SAVE PSW IN R4
4404  022504  112737  000300  177776  MOVB    #300,@#PSW      ;SET PRIORITY LEVEL 6
4405  022512  004737  002532          JSR     PC,@#CLRTBIT    ;CLEAR 'T' BIT IF SET
4406
4407                                  ;CHECK THAT ALL BITS IN THE CURRENT STACK PTR CAN BE SET/CLEARED
4408  022516  010603          CHKSP:  MOV     SP,R3           ;SAVE STACK PTR
4409  022520  000257                  CCC
4410  022522  112706  000377          MOVB    #377,SP         ;SET STACK PTR = -1
4411  022526  006006          1$:     ROR     SP              ;ROTATE 0 BIT THROUGH ALL BIT
4412  022530  103776                  BCS     1$              ;BIT POSITIONS
4413  022532  005206                  INC     SP              ;SHOULD INCREMENT SP TO 0
4414  022534  001403                  BEQ     2$
4415  022536  010602                  MOV     SP,R2           ;SAVE ERROR STACK PTR
4416  022540  010306                  MOV     R3,SP           ;SET STACK PTR FOR TRAP
4417  022542  104400                  HLT                     ;ERROR!
4418
4419  022544  010306          2$:     MOV     R3,SP           ;RESTORE ORIGINAL STACK PTR
4420
4421                                  ;CHECK BYTE OPERATIONS USING THE STACK
4422  022546  010600          SPCHK:  MOV     SP,R0           ;SAVE STACK PTR
4423  022550  010003                  MOV     R0,R3
4424
4425  022552  005043                  CLR     -(R3)
4426  022554  112746  177777          MOVB    #-1,-(SP)       ;(SP) = 377
4427  022560  022713  000377          CMP     #377,(R3)       ;CHECK THAT ONLY EVEN BYTE WAS AFFECTED
4428  022564  001002                  BNE     1$
4429  022566  020306                  CMP     R3,SP           ;CHECK AUTO-DEC
4430  022570  001401                  BEQ     .+4
4431  022572  104400          1$:     HLT
4432
4433  022574  105226                  INCB    (SP)+
4434  022576  005723                  TST     (R3)+           ;CHECK RESULT
4435  022600  001002                  BNE     2$
4436  022602  020006                  CMP     R0,SP           ;CHECK AUTO-INC
4437  022604  001401                  BEQ     .+4
4438  022606  104400          2$:     HLT
4439
4440  022610  005143                  COM     -(R3)           ;(R3)=177777
4441  022612  144613                  BICB    -(SP),(R3)
4442  022614  022713  177400          CMP     #177400,(R3)    ;CHECK RESULT
4443  022620  001002                  BNE     3$
4444  022622  020603                  CMP     SP,R3
4445  022624  001401                  BEQ     .+4
4446  022626  104400          3$:     HLT
4447
4448  022630  132627  000377          BITB    (SP)+,#377
4449  022634  001002                  BNE     4$
4450  022636  020600                  CMP     SP,R0
4451  022640  001401                  BEQ     .+4
4452  022642  104400          4$:     HLT
4453
4454  022644  012746  000001          MOV     #1,-(SP)
4455  022650  062706  000002          ADD     #2,SP
```



```
4456  022654  012702  177401          MOV     #177401,R2
4457  022660  120246                  CMPB    R2,-(SP)
4458  022662  001004                  BNE     5$
4459  022664  122602                  CMPB    (SP)+,R2
4460  022666  001002                  BNE     5$
4461  022670  020006                  CMP     R0,SP
4462  022672  001401                  BEQ     .+4
4463  022674  104400          5$:     HLT
4464  022676  105037  177776          CLRB    @#PSW
4465  022702  010446                  MOV     R4,-(SP)        ;RESTORE ORIGINAL PSW TO STACK
4466  022704  010746                  MOV     PC,-(SP)
4467  022706  062716  000006          ADD     #6,(SP)
4468  022712  000002                  RTI
4469  022714  104000                  $COPE
4470
4471                                  ;CHECK THAT 'C' BIT SETS/CLEARS PROPERLY
4472  022716  012727  177776  CBIT:   MOV     #177776,(PC)+   ;LOAD CONSTANT
4473  022722  000000          1$:     .WORD   0
4474  022724  010700                  MOV     PC,R0           ;GET CURRENT PC
4475  022726  162700  000004          SUB     #4,R0           ;POINT R0 TO 1$ ABOVE
4476  022732  005520          2$:     ADC     (R0)+           ;ADD 'C' BIT TO 1$ ABOVE
4477  022734  006340                  ASL     -(R0)           ;SHIFT 1$
4478  022736  102375                  BVC     2$              ;UNTIL 'V' BIT SETS
4479  022740  022767  077776  177754  CMP     #077776,1$      ;CHECK RESULT
4480  022746  001401                  BEQ     .+4
4481  022750  104400                  HLT                     ;ERROR! INCORRECT RESULT IN 1$ ABOVE
4482                                                          ;R0=ADDRESS OF DATA
4483
4484
4485                                  ;CHECK THAT CONDITION CODES ARE SET PROPERLY WHEN A NUMBER (CURRENT PC)
4486                                  ;AND THAT NUMBER +1 ARE COMPARED, AND VICE VERSA.
4487  022752  010700          CMPN:   MOV     PC,R0           ;GET CURRENT PC
4488  022754  010002                  MOV     R0,R2           ;SAVE IN R2
4489  022756  005202                  INC     R2              ;MAKE R2 = R0+1
4490  022760  000277                  SCC
4491  022762  000251                  +CLCICLN                ;CLEAR C & N BITS
4492  022764  020002                  CMP     R0,R2           ;COMPARE # WITH #+1
4493  022766  103003                  BCC     1$              ;CARRY BIT SHOULD SET
4494  022770  102402                  BVS     1$              ;V BIT SHOULD CLEAR
4495  022772  001401                  BEQ     1$              ;Z BIT SHOULD CLEAR
4496  022774  100401                  BMI     .+4             ;N BIT SHOULD SET
4497  022776  104400          1$:     HLT                     ;ERROR! COMPARE # WITH #+1 FAILED TO
4498                                                          ;SET CONDITION CODES IN PSW CORRECTLY
4499  023000  000277                  SCC                     ;SET CONDITION CODES IN PSW
4500  023002  120200                  CMPB    R2,R0           ;COMPARE #+1 WITH #
4501  023004  103403                  BCS     2$              ;C BIT SHOULD CLEAR
4502  023006  102402                  BVS     2$              ;V BIT SHOULD CLEAR
4503  023010  001401                  BEQ     2$              ;Z BIT SHOULD CLEAR
4504  023012  100001                  BPL     .+4             ;N BIT SHOLD CLEAR
4505  023014  104400          2$:     HLT                     ;ERROR! COMPARE #+1 WITH # FAILED TO SET
4506                                                          ;CONDITION CODES IN PSW CORRECTLY
4507
4508                                  ;24 NOP (240) INSTRUCTIONS FOLLOW, THESE NOPS MAY
4509                                  ;BE CHANGED TO TEST CODE IF THE NEED ARISES, THE TEST CODE SHOULD
```

```
                                ;BE POSITION INDEPENDENT AND SHOULD RUN WHEN RELOCATED BY THE PROGRAM.
023016  000240                          NOP
023020  000240                          NOP
023022  000240                          NOP
023024  000240                          NOP
023026  000240                          NOP
023030  000240                          NOP
023032  000240                          NOP
023034  000240                          NOP
023036  000240                          NOP
023040  000240                          NOP
023042  000240                          NOP
023044  000240                          NOP
023046  000240                          NOP
023050  000240                          NOP
023052  000240                          NOP
023054  000240                          NOP
023056  000240                          NOP
023060  000240                          NOP
023062  000240                          NOP
023064  000240                          NOP
023066  000240                          NOP
023070  000240                          NOP
023072  000240                          NOP
023074  000240                          NOP
023076  104000                          SCOPE
023100  010702                          MOV     PC,R2
023102  062702  000012                    ADD     #12,R2
023106  012707  001152                    MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
023112  000000                  REL44:  .WORD   0
                                ;44444444444444 LAST ADDRESS OF CODE TO BE RELOCATED 44444444444



                                        .SBTTL  START OF SECTION 5
                                ;55555555555555 FIRST ADDRESS TO BE RELOCATED 555555555
023114  010700                  REL5:   MOV     PC,R0           ;GET PC
023116  005740                          TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL5
023120  010037  001010                  MOV     R0,@#FIRSTAD    ;SAVE
023124  012737  000005  005176          MOV     #5,@#SECT       ;SET SECTION #
023132  004737  005166                  JSR     PC,@#LDDISP     ;LOAD DISPLAY REG
023136  013767  005172  001462          MOV     @#DISPLY,REL55
023144  010700                          MOV     PC,R0           ;GET CURRENT PC
023146  162700  023146                  SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
023152  010037  001004                  MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
023156  010701                          MOV     PC,R1           ;SET NEW SCOPE PTR

                                ;CHECK EXTENDED INSTRUCTION SET ($XT, XOR, SOB, MARK, RTI/RTT)
023160  005000                  EXTINST:CLR     R0
023162  000277                          SCC                     ;PRESET CC'S
023164  006700                          SXT     R0              ;EXTEND SIGN (1) INTO R0
023166  103005                          BCC     SXT0            ;CHECK RESULT CC'S
023170  102404                          BVS     SXT0
```



```
023172  001403                          BEQ     SXT0
023174  100002                          BPL     SXT0
023176  005200                          INC     R0              ;CHECK RESULT
023200  001401                          BEQ     .+4
023202  104400                  SXT0:   HLT

023204  010700                          MOV     PC,R0
023206  010002                          MOV     R0,R2
023210  012703  177777                  MOV     #-1,R3
023214  005102                          COM     R2
023216  000243                          +CLV!CLC                ;CLEAR C AND V BITS
023220  074003                          XOR     R0,R3           ;R3 SHOULD CONTAIN COMPLEMENT OF R0
023222  103404                          BCS     XORR0           ;CHECK THAT C WAS NOT AFFECTED
023224  102403                          BVS     XORR0           ;AND THAT V WAS CLEARED
023226  001402                          BEQ     XORR0
023230  020203                          CMP     R2,R3           ;CHECK RESULT
023232  001401                          BEQ     .+4
023234  104400                  XORR0:  HLT                     ;ERROR!   XOR FAILED

023236  010700                          MOV     PC,R0
023240  022020                          CMP     (R0)+,(R0)+     ;SET ADDRESS REGISTER
023242  000401                          BR      1$              ;RESERVE WORD FOR TEST DATA
023244  000000                          .WORD   0               ;CONTAINS TEST DATA
023246  005700                  1$:     TST     R0              ;EXTEND SIGN OF ADDRESS INTO
023250  006710                          SXT     (R0)+           ;ADDRESS (R0)=-1 IF MSB R0=1
023252  005002                          CLR     R2              ;OTHERWISE, (R0)=0
023254  005700                          TST     R0              ;CHECK SIGN OF ADDRESS
023256  100001                          BPL     .+4
023260  005102                          COM     R2              ;COMPLEMENT CHECK REG IF NEG
023262  021002                          CMP     (R0)+,R2        ;CHECK RESULT OF SXT
023264  001401                          BEQ     .+4
023266  104400                  SXT1:   HLT                     ;ERROR!   SXT FAILED TO EXTEND SIGN PROPERLY

023270  012710  100000                  MOV     #100000,(R0)    ;PRESET DATA
023274  011002                          MOV     (R0)+,R2
023276  000277                          SCC                     ;PRESET CC'S
023300  074210                          XOR     R2,(R0)         ;XOR 100000 WITH 100000 RESULT = 0
023302  103007                          BCC     XOR1            ;CHECK CC'S AFTER XOR
023304  102406                          BVS     XOR1
023306  001005                          BNE     XOR1
023310  100404                          BMI     XOR1
023312  005710                          TST     (R0)+           ;CHECK RESULT (0)
023314  001002                          BNE     XOR1
023316  005402                          NEG     R2              ;CHECK THAT REG WAS NOT AFFECTED
023320  102401                          BVS     .+4
023322  104400                  XOR1:   HLT

023324  010702                          MOV     PC,R2
023326  022222                          CMP     (R2)+,(R2)+
023330  000401                          BR      SXT4            ;PRESERVE WORD FOR DATA
023332  000000                          .WORD   0               ;RESERVED FOR DATA
023334  012722  125252          SXT4:   MOV     #125252,(R2)+   ;PRESET DATA
023340  006742                          SXT     -(R2)           ;EXTEND SIGN
023342  074722                          XOR     PC,(R2)+
```

```
 4618  023344  010700                          MOV     PC,R0           ;GET PC
 4619  023346  005740                          TST     -(R0)           ;SUBTRACT 2 FROM PC
 4620  023350  005100                          COM     R0              ;R0=RESULT OF XOR PC-1 ABOVE
 4621  023352  074042                          XOR     R0,-(R2)        ;CHECK RESULT OF SXT AND XOR ABOVE
 4622  023354  001401                          BEQ     .+4
 4623  023356  104400                XOR24:    HLT                     ;ERROR:  SXT & XOR ABOVE INCORRECT
 4624
 4625  023360  012704  000001                  MOV     #1,R4           ;SET R4
 4626  023364  006767  000060                  SXT     XOR6A           ;PRESET DATA=0
 4627  023370  074467  000054        2$:       XOR     R4,XOR6A
 4628  023374  100423                          BMI     XOR6
 4629  023376  006304                          ASL     R4              ;SHIFT R4
 4630  023400  102373                          BVC     2$              ;UNTIL V SETS (R4=100000)
 4631  023402  100020                          BPL     XOR6            ;BRANCH IF 'N' IS CLEAR
 4632  023404  074467  000040                  XOR     R4,XOR6A        ;XOR6A=177777
 4633  023410  100015                          BPL     XOR6
N4634  023412  074767  000032                  XOR     PC,XOR6A        ;XOR PC WITH XOR6A (177777)
N4635  023416  010767  000030                  MOV     PC,XOR6B        ;FORM PC AS USED IN XOR ABOVE
 4636  023422  162767  000004  000022          SUB     #4,XOR6B
 4637  023430  005167  000016                  COM     XOR6B
 4638  023434  026767  000012  000006          CMP     XOR6B,XOR6A     ;XOR6A SHOULD = COMPLEMENT OF PC
 4639  023442  001401                          BEQ     .+4
 4640  023444  104400                XOR6:     HLT                     ;ERROR:  XOR TESTS ABOVE FAILED
 4641
 4642  023446  000402                          BR      .+6
 4643
 4644  023450  000000                XOR6A:    .WORD   0               ;CONTAINS DATA USED BY TEST ABOVE
 4645  023452  000000                XOR6B:    .WORD   0
 4646
 4647
 4648  023454  012700  077777                  MOV     #077777,R0      ;SET SOURCE OPERAND FOR ADD
 4649  023460  006767  177764                  SXT     XOR6A           ;CLEAR XOR6A
 4650  023464  001004                          BNE     SXT6            ;CHECK CC'S AFTER EXTENDING ZERO'S
 4651  023466  100403                          BMI     SXT6
 4652  023470  103402                          BCS     SXT6
 4653  023472  102401                          BVS     SXT6
 4654  023474  000401                          BR      .+4
 4655  023476  104400                SXT6:     HLT                     ;ERROR:  SXT FAILED
 4656
 4657  023500  012702  000001                  MOV     #1,R2           ;SET DEST OPERAND FOR ADD
 4658  023504  013703  001004                  MOV     @#FACTOR,R3     ;LOAD INDEX REGISTER
 4659  023510  060002                          ADD     R0,R2           ;RESULT OF ADD=100000
 4660  023512  006763  023450                  SXT     XOR6A(3)        ;EXTEND SIGN OF ADD ABOVE
 4661  023516  001403                          BEQ     SXT6A
 4662  023520  005267  177724                  INC     XOR6A           ;CHECK RESULT OF SXT
 4663  023524  001401                          BEQ     .+4
 4664  023526  104400                SXT6A:    HLT                     ;ERROR:  SXT ABOVE FAILED TO EXTEND
 4665                                                                  ;SIGN
 4666  023530  010703                          MOV     PC,R3
 4667  023532  000402                          BR      .+6             ;PRESERVE 2 WORDS FOR DATA
 4668  023534  000000                SXRA:     .WORD   0               ;RESERVED WORD FOR DATA
 4669  023536  000000                SXRB:     .WORD   0               ;RESERVED WORD FOR DATA
 4670  023540  005723                          TST     (R3)+
 4671  023542  010304                          MOV     R3,R4           ;R3 = ADDRESS OF SXRA
```



```
 4672  023544  000250                          CLN                     ;CLEAR N BIT
 4673  023546  006724                          SXT     (R4)+           ;EXTEND ZEROS INTO SXRA
 4674  023550  001401                          BEQ     .+4
 4675  023552  104400                SXT2:     HLT                     ;ERROR:  SXT FAILED
 4676
 4677  023554  010467  177754                  MOV     R4,SXRA         ;SXRA = ADDRESS OF SXRB
 4678  023560  000257                          CCC                     ;CLEAR CONDITION CODES
 4679  023562  006733                          SXT     @(R3)+          ;EXTEND ZEROS INTO SXRB
 4680  023564  001401                          BEQ     .+4
 4681  023566  104400                SXT3:     HLT                     ;ERROR:
 4682
 4683  023570  000270                          SEN                     ;SET N BIT
 4684  023572  006753                          SXT     @-(R3)          ;EXTEND ONES INTO SXRB
 4685  023574  100401                          BMI     .+4
 4686  023576  104400                SXT5:     HLT                     ;ERROR:
 4687
 4688  023600  012704  025252                  MOV     #025252,R4      ;R4 = 025252
 4689  023604  074433                          XOR     R4,@(R3)+       ;SXRB = 152525 (COMPLEMENT OF R4)
 4690  023606  005002                          CLR     R2
 4691  023610  074253                          XOR     R2,@-(R3)       ;SXRB REMAINS UNCHANGED
 4692  023612  001405                          BEQ     XOR35           ;CHECK CONDITION CODES
 4693  023614  100004                          BPL     XOR35
 4694  023616  005104                          COM     R4              ;R4 = 152525
 4695  023620  020467  177712                  CMP     R4,SXRB         ;CHECK XOR
 4696  023624  001401                          BEQ     .+4
 4697  023626  104400                XOR35:    HLT                     ;ERROR:  XOR FAILED
 4698
 4699  023630  005743                          TST     -(R3)           ;R3 = ADDRESS OF SXRA-2
 4700  023632  000250                          CLN                     ;CLEAR N BIT
 4701  023634  006773  000002                  SXT     @2(R3)          ;SXRB = 0
 4702  023640  001401                          BEQ     .+4
 4703  023642  104400                SXT7:     HLT                     ;ERROR:  SXT FAILED
 4704
 4705  023644  074473  000002                  XOR     R4,@2(R3)       ;SXRB = R4
 4706  023650  020473  000002                  CMP     R4,@2(R3)       ;CHECK XOR
 4707  023654  001421                          BEQ     .+4
 4708  023656  104400                XOR7:     HLT                     ;ERROR:  XOR FAILED
 4709  023660  104000                          SCOPE
 4710
 4711                                ;NOTE:    DO NOT INSERT ANY CODE IN FOLLOWING SOB TESTS
 4712                                ;         SINCE IT TESTS THE MAXIMUM BRANCH WIDTH OF THE INSTRUCTION,
 4713
 4714  023662  005005                          CLR     R5              ;CLEAR ERROR INDICATOR
 4715  023664  000407                          BR      SOB0            ;BRANCH TO SOB TEST
 4716
 4717  023666  005004                SOB10:    CLR     R4              ;R4 = 0
 4718  023670  005705                          TST     R5              ;CHECK ERROR INDICATOR
 4719  023672  001401                          BEQ     .+4             ;SOB BRANCHED CORRECTLY
 4720  023674  104400                          HLT                     ;ERROR!
 4721
 4722  023676  005005                SOB9:     CLR     R5              ;CLEAR INDICATOR (R5)
 4723  023700  006004                          ROR     R4              ;ROTATE RIGHT R4
 4724  023702  000467                          BR      SOB8
 4725
```

```
4726  023704  012700  000010          SOB0:   MOV     #10,R0          ;R0=10
4727  023710  000277                          SCC                     ;SET CONDITION CODES
4728  023712  001012                  SOB1:   BNE     SOB2            ;CHECK CONDITION CODES AFTER SOB
4729  023714  100011                          BPL     SOB2            ;SOB SHOULD NOT EFFECT THE
4730  023716  102010                          BVC     SOB2            ;CONDITION CODES,
4731  023720  103007                          BCC     SOB2
4732  023722  077005                          SOB     R0,SOB1
4733  023724  001005                          BNE     SOB2            ;CHECK CONDITION CODES AFTER
4734  023726  100004                          BPL     SOB2            ;SOB FALLS THROUGH,
4735  023730  102003                          BVC     SOB2            ;SOB SHOULD NOT EFFECT
4736  023732  103002                          BCC     SOB2            ;CONDITION CODES,
4737  023734  005700                          TST     R0              ;CHECK IF R0=0
4738  023736  001401                          BEQ     .+4
4739  023740  104400                  SOB2:   HLT                     ;ERROR!
4740
4741  023742  012702  000100                  MOV     #100,R2         ;R2=100
4742  023746  012700  000101                  MOV     #101,R0         ;SET CHECK REGISTER, R0=101
4743  023752  001414                  SOB3:   BEQ     SOB4            ;CHECK CONDITION CODES AFTER
4744  023754  100413                          BMI     SOB4            ;SOB BRANCH,
4745  023756  102412                          BVS     SOB4            ;SOB SHOULD NOT EFFECT
4746  023760  103411                          BCS     SOB4            ;CONDITION CODES,
4747  023762  005300                          DEC     R0              ;DECREMENT CHECK REGISTER
4748  023764  020002                          CMP     R0,R2           ;CHECK THAT SOB DECREMENTS
4749  023766  001006                          BNE     SOB4
4750  023770  000257                          CCC                     ;SET CONDITION CODES BEFORE SOB
4751  023772  077211                          SOB     R2,SOB3         ;BRANCH TO SOB3 UNTIL R2=0
4752  023774  001403                          BEQ     SOB4            ;CHECK CONDITION CODES AFTER
4753  023776  100402                          BMI     SOB4            ;SOB FALLS THROUGH
4754  024000  005702                          TST     R2              ;CHECK IF R2=0
4755  024002  001401                          BEQ     .+4
4756  024004  104400                  SOB4:   HLT                     ;ERROR!
4757
4758  024006  012700  000001          SOB5:   MOV     #1,R0           ;R0=1
4759  024012  000401                          BR      .+4
4760  024014  104400                          HLT                     ;ERROR!
4761  024016  077002                          SOB     R0,.-2          ;SOB SHOULD NOT BRANCH
4762
4763  024020  005700                          TST     R0              ;CHECK IF R0=0 AFTER SOB
4764  024022  001401                          BEQ     .+4
4765  024024  104400                          HLT                     ;ERROR!
4766
4767  024026  012704  100000          SOB5A:  MOV     #100000,R4      ;R4=100000
4768  024032  000403                          BR      1$
4769  024034  005204                  3$:     INC     R4              ;R4=100000
4770  024036  100403                          BMI     2$              ;N BIT SHOULD BE SET
4771  024040  104400                          HLT                     ;ERROR!  SOB DID NOT
4772                                                                  ;INCREMENT PROPERLY
4773
4774  024042  077404                  1$:     SOB     R4,3$           ;SOB SHOULD BRANCH
4775  024044  104400                          HLT                     ;ERROR!  SOB DID NOT BRANCH
4776
4777  024046  012703  000100          2$:     MOV     #100,R3         ;R3=100
4778  024052  077301                  SOB6:   SOB     R3,SOB6         ;USE SOB TO BRANCH TO ITSELF
4779  024054  005703                          TST     R3              ;CHECK IF R3=0
```



```
4780  024056  001703                          BEQ     SOB10
4781  024060  104400                  SOB7:   HLT                     ;ERROR!
4782
4783  024062  005705                  SOB8:   TST     R5              ;CHECK INDICATOR (R5)
4784                                                                  ;IF SOB BRANCHES INCORRECTLY
4785                                                                  ;WHEN CHECKING MAX, BRANCH,
4786                                                                  ;R5 WILL NOT BE CLEARED AT
4787                                                                  ;THIS POINT INDICATING AN ERROR,
4788
4789  024064  001401                          BEQ     .+4             ;BRANCH IF SOB BRANCHES CORRECTLY
4790  024066  104400                          HLT                     ;ERROR!
4791
4792  024070  005205                          INC     R5              ;SET INDICATOR (R5)
4793  024072  077477                          SOB     R4,SOB9         ;TEST MAX, BRANCH OF SOB
4794  024074  005704                          TST     R4              ;CHECK IF R4=0
4795  024076  001401                          BEQ     .+4
4796  024100  104400                          HLT                     ;ERROR!
4797  024102  104000                          SCOPE
4798
4799
4800                                  ;CHECK THAT MARK INSTRUCTION POPS OVER THE CORRECT # OF ARGUMENTS,RESTORES R5 FR
4801                                  ;THE STACK POINTER
4802
4803  024104  010602                  MRKTST: MOV     SP,R2
4804  024106  010705                          MOV     PC,R5           ;THE STACK LOOKS LIKE THIS AFTER
4805  024110  010500                          MOV     R5,R0           ;THE JSR INSTRUCTION
4806  024112  010546                          MOV     R5,-(SP)        ; -2(SP)= R0    THIS IS A
4807  024114  010746                          MOV     PC,-(SP)        ; -4(SP)= PC    STRING
4808  024116  010746                          MOV     PC,-(SP)        ; -6(SP)= PC+2  OF
4809  024120  010746                          MOV     PC,-(SP)        ;-10(SP)= PC+4  FIVE
4810  024122  010746                          MOV     PC,-(SP)        ;-12(SP)= PC+6  DUMMY
4811  024124  010746                          MOV     PC,-(SP)        ;-14(SP)= PC+10 ARGUMENTS
4812  024126  012746  006405                  MOV     #MARK+5,-(SP)   ;-16(SP)= MARK 5
4813  024132  010605                          MOV     SP,R5           ;-20(SP)= PC PUSHED BY JSR
4814  024134  004767  000002                  JSR     PC,MARK1
4815  024140  000403                          BR      .+10
4816  024142  000205                  MARK1:  RTS     R5
4817  024144  104400                          HLT                     ;ERROR! SHOULD BE DOING MARK 5 INST,
4818  024146  000407                          BR      MARKEX
4819  024150  020602                          CMP     SP,R2
4820  024152  001402                          BEQ     .+6
4821  024154  104400                          HLT                     ;ERROR! SP NOT RETURNED TO PROPER
4822  024156  000403                          BR      MARKEX          ;VALUE BY MARK INSTRUCTION
4823  024160  020005                          CMP     R0,R5
4824  024162  001401                          BEQ     .+4
4825  024164  104400                          HLT                     ;ERROR! DID NOT RESTORE R5 FROM STACK
4826  024166  010206                  MARKEX: MOV     R2,SP           ;RESTORE SP
4827  024170  104000                          SCOPE
4828
4829                                  ;RTT/RTI TEST INSURES THAT CP DOES THE INSTRUCTION FOLLOWING
4830                                  ;AN RTT IF THE "T"BIT IS SET IN THE PSW,BUT DOES HONOR
4831                                  ;THE TRAP IMMEDIATELY IF IT EXECUTES AN RTI
4832                                  ;INSTRUCTION SEQUENCE-RTT
4833                                  ;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
024172  013767  177776  000166  RTT1:   MOV     @#PSW,SAVPSW            ;SAVE PSW
024200  032767  000020  000160          BIT     #T,SAVPSW               ;CHECK IF "T"BIT SET
024206  001176                          BNE     RTT2EX                  ;BRANCH TO EXIT
024210  010746                  1$:     MOV     PC,-(SP)                ;GET CURRENT PC
024212  062716  000116                  ADD     #5$-.,(SP)              ;FORM RELOCATED PC
024216  012637  000014                  MOV     (SP)+,@#TBITVEC         ;LOAD INTO TRAP VECTOR
024222  016746  000140                  MOV     SAVPSW,-(SP)            ;GET CURRENT PSW
```



```
024226  011637  000016                  MOV     (SP)+,@#TBITVEC+2
024232  052737  000340  177776          BIS     #PRTY7,@#PSW            ;SET PRIORITY LEVEL 7
```

```
4859  024240  005000                  CLR     R0
4860  024242  052716  000360          BIS     #PRTY7*T,(SP)           ;SET "T"BIT IN PSW ON STACK
4861  024246  010746                  MOV     PC,-(SP)                ;PUT THE PC ON THE STACK
4862  024250  062716  000006          ADD     #6,(SP)                 ;ADJUST PC FOR NEXT INSTRUCTION
4863  024254  000006                  RTT
4864  024256  005200          2$:     INC     R0                      ;DONE TO SEE IF INSTR, FOLLOWING
4865                                                                   ;RTT IS EXECUTED IF T-BIT SET
4866  024260  042737  000340  177776  BIC     #PRTY7,@#PSW            ;SET PRIORITY LEVEL 0
4867  024266  022767  000006  177760  CMP     #RTT,2$
4868  024274  001005                  BNE     3$                      ;CHECK IF INC WAS EXECUTED
4869  024276  022700  177776          CMP     #177776,R0              ;CHECK IF COM-R0 EXECUTED
4870  024302  001406                  BEQ     4$
4871  024304  104400                  HLT                             ;ERROR!R0 NOT COMPLIMENTED
4872  024306  000415                  BR      6$                      ;EXIT TEST
4873  024310  005700          3$:     TST     R0                      ;TEST IF TRAPED BEFORE INC INST,
4874                                                                   ;WAS EXECUTED
4875  024312  001413                  BEQ     6$
4876  024314  104400                  HLT                             ;ERROR!
4877  024316  000411                  BR      6$                      ;EXIT TEST
4878  024320  012767  000002  177726  4$:     MOV     #RTI,2$
4879  024326  000730                  BR      1$
4880  024330  005100          5$:     COM     R0                      ;RTT CHECK
4881  024332  016766  000030  000002  MOV     SAVPSW,2(SP)
4882  024340  000002                  RTI
4883  024342  012767  000006  177704  6$:     MOV     #RTT,2$
4884  024350  012737  000016  000014          MOV     #TBITVEC+2,@#TBITVEC    ;RESTORE TRAP VECTORES
4885  024356  005037  000016          CLR     @#TBITVEC+2
4886  024362  104000          RTT1EX: SCOPE

4887
4888  024364  000401                  BR      RTT2
4889  024366  000000          SAVPSW: .WORD   0
4890                                  ;CHECK IF AN 11/45 AND DETERMINE WHICH MODE AND REG, SET ARE SELECTED BY THE PSW
4891  024370  122737  000004  000764  RTT2:   CMPB    #4,@#OPT,CP             ;TEST IF AN 11/40
4892  024376  001002                  BNE     RTT2A                   ;BRANCH IF NOT AN 11/40
4893  024400  000167  000220          JMP     RTT2EX                  ;GO TO RTT2EX IF 11/40
4894  024404  016700  177756          RTT2A:  MOV     SAVPSW,R0               ;GET SAVED PSW
4895  024410  105000                  CLRB    R0                      ;CLEAR PRIOITY LEVEL,T, AND COND CODES
4896  024412  012702  144000          MOV     #UM+REG,R2
4897  024416  074002                  XOR     R0,R2
4898  024420  001435                  BEQ     2$                      ;USER MODE REG, SET #1 ON
4899  024422  012702  044000          MOV     #SM+REG,R2
4900  024426  074002                  XOR     R0,R2
4901  024430  001447                  BEQ     3$                      ;SUPER MODE REG, SET #1 ON
4902  024432  032700  140000          BIT     #UM,R0
4903  024436  001062                  BNE     RTT2EX
4904
4905                                  ;TEST THAT RTT CLEARS BITS 11,12,13 & PRIORITY LEVEL BITS IN KERNEL MODE
4906  024440  012702  177777          MOV     #-1,R2                  ;KERNEL MODE REG, SET 0 ON
4907  024444  012737  034240  177776  MOV     #PUM+REG+PRTY5,@#PSW    ;SELECT REG, SET #1
4908  024452  005002                  CLR     R12                     ;SHOULD CLEAR REG #12
4909  024454  012746  000100          MOV     #PRTY2,-(SP)
4910  024460  010746                  MOV     PC,-(SP)
4911  024462  062716  000006          ADD     #16.,(SP)               ;FORM NEW PC
4912  024466  000006                  RTT
```



```
4913  024470  013700  177776          1$:     MOV     @#PSW,R0                ;NOW USING REG SET 0
4914  024474  005702                  TST     R2                      ;SHOULD TEST R2 NOT R12
4915  024476  001001                  BNE     4$
4916  024500  104400                  HLT                             ;ERROR!DID NOT CLEAR BIT #11 OF PSW
4917  024502  022700  000100          4$:     CMP     #PRTY2,R0               ;TESTS THE PSW AFTER THE RTT
4918  024506  001436                  BEQ     RTT2EX
4919  024510  104400                  HLT                             ;ERROR! INCORRECT PSW AFTER THE RTT
4920  024512  000434                  BR      RTT2EX
4921
4922                                  ;TEST TO INSURE THAT RTI DOES NOT CLEAR BITS 11-15 IN USER MODE
4923  024514  052737  030340  177776  2$:     BIS     #PUM+PRTY7,@#PSW        ;PSW<15-5>=144X
4924  024522  005046                  CLR     -(SP)
4925  024524  010746                  MOV     PC,-(SP)
4926  024526  062716  000006          ADD     #5$-.,(SP)
4927  024532  000002                  RTI                             ;ATTEMPS TO INSERT A PSW OF 0
4928  024534  022737  174340  177776  5$:     CMP     #UM+PUM+REG+PRTY7,@#PSW ;SHOULD CHECK AGAINST REG #0
4929  024542  001420                  BEQ     RTT2EX
4930  024544  104400                  HLT                             ;ERROR! RTI CLEARED BITS IN PSW
4931  024546  000416                  BR      RTT2EX
4932
4933                                  ;TEST THAT BITS 11-15 AND PRIORTY BITS ARE NOT ALTERED IN SUPER MODE
4934  024550  052737  030200  177776  3$:     BIS     #PUM+PRTY4,@#PSW        ;PSW<15-5>=044X
4935  024556  012746  000340          MOV     #PRTY7,-(SP)
4936  024562  010746                  MOV     PC,-(SP)
4937  024564  062716  000006          ADD     #6$-.,(SP)
4938  024570  000006                  RTT                             ;ATTEMPTS TO CLEAR 11-15 AND ALTER PRTY
4939
4940  024572  022737  074200  177776  6$:     CMP     #SM+PUM+REG+PRTY4,@#PSW
4941  024600  001401                  BEQ     RTT2EX
4942  024602  104400                  HLT                             ;ERROR! RTT ALTERED PRTY IN
4943                                                                   ;SUPER MODE OR BITS 11-15,
4944  024604  016737  177556  177776  RTT2EX: MOV     SAVPSW,@#PSW
4945  024612  104000                  SCOPE
4946
4947  024614  010702                  MOV     PC,R2
4948  024616  062702  000012          ADD     #12,R2
4949  024622  012707  001152          MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
4950  024626  000000          REL55:  .WORD   0
4951                                  ;55555555555555 LAST ADDRESS OF CODE TO BE RELOCATED 55555555555
4952
4953
4954
4955                                  .SBTTL  START OF SECTION 6
4956                                  ;6666666666666 FIRST ADDRESS TO BE RELOCATED 666666666
4957  024630  010700          REL6:   MOV     PC,R0           ;GET PC
4958  024632  005740                  TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL6
4959  024634  010037  001010          MOV     R0,@#FRSTAD     ;SAVE
4960  024640  012737  000006  005176  MOV     #6,@#SECT       ;SET SECTION #
4961  024646  004737  005166          JSR     PC,@#LDDISP     ;LOAD DISPLAY GEG
4962  024652  013767  005172  001710  MOV     @#DISPLY,REL66
4963  024660  010700                  MOV     PC,R0           ;GET CURRENT PC
4964  024662  162700  024662          SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
4965  024666  010037  001004          MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
4966  024672  010701                  MOV     PC,R1           ;SET NEW SCOPE PTR
```

```
024674  032737  040000  000764           BIT     #EISOPT,@#OPT.CP      ;CHECK IF 11/40 WITH EIS OPTION
024702  001002                           BNE     ASHL0                 ;BRANCH IF EIS OPT AVAIL.
024704  000167  001336                   JMP     MPI

                                 ;CHECK ASH,ASHC,MUL, AND DIV INSTRUCTIONS
024710  012700  000001           ASHL0:  MOV     #1,R0                 ;R0 WILL BE THE SHIFT COUNT
024714  012703  000021                   MOV     #17.,R3               ;MAX SHIFT COUNT
024720  005067  000014           1$:     CLR     2$                    ;PRESET SAVED CC'S LOCATION=0
024724  010002                           MOV     R0,R2                 ;GET SHIFT COUNT FOR PASS
024726  010705                           MOV     PC,R5                 ;R5 & R4 WILL BE DATA SHIFTED BY
024730  010504                           MOV     R5,R4                 ;ASH & ASL INSTRUCTIONS
024732  072502                           ASH     R2,R5                 ;SHIFT R5
024734  113727  177776                   MOVB    @#PSW,(PC)+           ;SAVE CC'S
024740  000000                   2$:     .WORD   0                     ;CONTAINS ASH CC'S IN EVEN BYTE
                                                                       ;ASL CC'S IN ODD BYTE
024742  006304                   3$:     ASL     R4                    ;SHIFT R4
024744  113746  177776                   MOVB    @#PSW,-(SP)           ;SAVE PSW ON STACK
024750  132716  000002                   BITB    #V,(SP)               ;CHECK IF ASL SET V BIT
024754  001403                           BEQ     30$
024756  152767  000002  177755           BISB    #V,2$+1               ;IF ASL SET V THEN SET V IN 2$+1
024764  112637  177776           30$:    MOVB    (SP)+,@#PSW           ;RESTORE ORIGINAL PSW
024770  077214                           SOB     R2,3$                 ;SHIFT R4 R2 TIMES
024772  153767  177776  177741           BISB    @#PSW,2$+1            ;SAVE CC'S AFTER ASL
025000  020504                           CMP     R5,R4                 ;CHECK ASH & ASL RESULTS
025002  001004                           BNE     4$
025004  126767  177730  177727           CMPB    2$,2$+1               ;CHECK ASH & ASL CC'S
025012  001401                           BEQ     .+4
025014  104400                   4$:     HLT                           ;ERROR! INCORRECT RESULT OR CC'S
025016  005200                           INC     R0                    ;INCREMENT PASS SHIFT COUNT
025020  020003                           CMP     R0,R3
025022  001336                           BNE     1$

025024  012700  177777           ASHR0:  MOV     #-1,R0                ;R0 = RIGHT SHIFT COUNT FOR PASS
025030  012703  177757                   MOV     #-17.,R3              ;MAX SHIFT COUNT
025034  010002                   1$:     MOV     R0,R2                 ;GET SHIFT COUNT FOR PASS
025036  010705                           MOV     PC,R5                 ;R5 & R4 = DATA TO BE SHIFTED
025040  010504                           MOV     R5,R4                 ;BY ASH & ASR INSTRUCTIONS
025042  072502                           ASH     R2,R5                 ;SHIFT R5 R2 TIMES
025044  113727  177776                   MOVB    @#PSW,(PC)+           ;SAVE CC'S IN EVEN BYTE
025050  000000                   2$:     .WORD   0                     ;CONTAINS ASH CC'S IN EVEN BYTE
                                                                       ;ASR CC'S IN ODD BYTE
025052  005402                           NEG     R2
025054  006204                   3$:     ASR     R4                    ;SHIFT R4
025056  077202                           SOB     R2,3$                 ;SHIFT R4 R2 TIMES
025060  113767  177776  177763           MOVB    @#PSW,2$+1            ;SAVE CC'S AFTER ASR
025066  142767  000002  177755           BICB    #V,2$+1               ;ASH RIGHT WILL NOT SET V ASR MAY SET V
025074  020504                           CMP     R5,R4                 ;CHECK ASH & ASR RESULTS
025076  001004                           BNE     4$
025100  126767  177744  177743           CMPB    2$,2$+1               ;CHECK ASH & ASR CC'S
025106  001401                           BEQ     .+4
025110  104400                   4$:     HLT
025112  005300                           DEC     R0                    ;DECREMENT PASS SHIFT COUNT
025114  020003                           CMP     R0,R3
```



```
025116  001346                           BNE     1$

025120  012746  000037           ASHCL0: MOV     #31.,-(SP)            ;PUT MAX SHIFT COUNT ON STACK
025124  012746  000001                   MOV     #1,-(SP)              ;PUT LEFT SHIFT COUNT ON STACK
025130  011600                   1$:     MOV     (SP),R0               ;GET PASS SHIFT COUNT
025132  010705                           MOV     PC,R5                 ;CURRENT PC IS DATA TO BE SHIFTED
025134  010503                           MOV     R5,R3                 ;ASHC SHIFTS R4,R5;ASL,ROL SHIFTS R2,R3
025136  005004                           CLR     R4
025140  005002                           CLR     R2
025142  073400                           ASHC    R0,R4                 ;SHIFT R4 LEFT AS SPECIFIED BY R0
025144  006303                   2$:     ASL     R3                    ;SHIFT R2,R3 LEFT
025146  006102                           ROL     R2                    ;AS SPECIFIED BY R0
025150  077003                           SOB     R0,2$
025152  020402                           CMP     R4,R2                 ;CHECK RESULTS
025154  001002                           BNE     3$
025156  020503                           CMP     R5,R3
025160  001401                           BEQ     .+4
025162  104400                   3$:     HLT
025164  005216                           INC     (SP)                  ;INCREMENT NEXT PASS SHIFT COUNT
025166  021666  000002                   CMP     (SP),2(SP)            ;REACHED MAX COUNT (31.)
025172  001356                           BNE     1$
025174  022626                           CMP     (SP)+,(SP)+           ;RESTORE STACK PTR

025176  012746  177740           ASHCR0: MOV     #-32.,-(SP)           ;PUT MAX RIGHT SHIFT COUNT ON STACK
025202  012746  177777                   MOV     #-1,-(SP)             ;PUT PASS SHIFT COUNT ON STACK
025206  011600                   1$:     MOV     (SP),R0               ;GET PASS SHIFT COUNT
025210  010702                           MOV     PC,R2                 ;R2,R3 & R4,R5 ARE THE DATA REGISTERS
025212  010204                           MOV     R2,R4                 ;TO BE SHIFTED BY TEST
025214  005003                           CLR     R3
025216  005005                           CLR     R5
025220  000262                           SEV                           ;SET V BIT IN PSW
025222  073200                           ASHC    R0,R2                 ;SHIFT R2,R3 RIGHT R0 TIMES
025224  102410                           BVS     3$                    ;SHIFT RIGHT CLEARS V
025226  005400                           NEG     R0                    ;NEGATE SHIFT COUNT FOR SOB
025230  006204                   2$:     ASR     R4                    ;SHIFT R4,R5 RIGHT R0 TIMES
025232  006005                           ROR     R5
025234  077003                           SOB     R0,2$
025236  020204                           CMP     R2,R4                 ;CHECK RESULT
025240  001002                           BNE     3$
025242  020305                           CMP     R3,R5
025244  001401                           BEQ     .+4
025246  104400                   3$:     HLT
025250  005316                           DEC     (SP)                  ;SET SHIFT COUNT FOR NEXT PASS
025252  021666  000002                   CMP     (SP),2(SP)            ;CHECK IF MAX SHIFT COUNT
025256  001353                           BNE     1$
025260  022626                           CMP     (SP)+,(SP)+           ;RESTORE STACK PTR
025262  104000                           SCOPE

                                 ;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.
025264  012700  000001           MUL0:   MOV     #1,R0                 ;R0 CONTAINS MULTIPLIER FOR MUL
025270  005016                           CLR     (SP)                  ;(SP) CONTAINS SHIFT VALUE FOR ASHC
025272  010702                   1$:     MOV     PC,R2                 ;R3,R2 & R5,R4 ARE DATA REGISTERS
```

```
 5075  025274  010227                        MOV     R2,(PC)+        ;SAVE MULTIPICAND
 5076  025276  000000                        .WORD   0               ;CONTAINS ORIGINAL MULTIPICAND
 5077  025300  005003                        CLR     R3
 5078  025302  005004                        CLR     R4
 5079  025304  010205                        MOV     R2,R5           ;FOR MUL AND ASHC
 5080  025306  100001                        BPL     .+4             ;IF MULTPICAND IS NEG THEN SET R4 = -1
 5081  025310  005104                        COM     R4              ;FOR ASHC
 5082  025312  000277                        SCC                     ;PRESET CC'S
 5083  025314  070200                        MUL     R0,R2           ;MULTIPLY R2 BY R0 LEAVE PRODUCT
 5084                                                                ;IN R2,R3 MSH IN R2,LSH IN R3
 5085  025316  102406                        BVS     2$
 5086  025320  001405                        BEQ     2$              ;PRODUCT WILL NEVER BE = 0
 5087  025322  073416                        ASHC    (SP)+,R4        ;'MULTIPLY' R4,R5 BY (SP) LEAVE PRODUCT
 5088                                                                ;IN R4,R5 MSH IN R4,LSH IN R5
 5089  025324  020204                        CMP     R2,R4           ;CHECK MSH RESULT
 5090  025326  001002                        BNE     2$
 5091  025330  020305                        CMP     R3,R5           ;CHECK LSH RESULT
 5092  025332  001401                        BEQ     .+4
 5093  025334  104400                2$:     HLT
 5094  025336  005216                        INC     (SP)            ;INCREMENT ASHC SHIFT COUNT
 5095  025340  006300                        ASL     R0              ;SHIFT MUL MULTIPLIER
 5096  025342  102353                        BVC     1$
 5097                                ;CHECK MUL INST WITH MULTIPIER (R0) = 100000
 5098  025344  010702                        MOV     PC,R2           ;R2 = MULTIPICAND
 5099  025346  005202                        INC     R2
 5100  025350  010227                        MOV     R2,(PC)+        ;SAVE MULTIPICAND
 5101  025352  000000                        .WORD   0               ;CONTAINS ORIGINAL MULTIPICAND
 5102  025354  005103                        COM     R3
 5103  025356  010204                        MOV     R2,R4           ;R4 WILL BE MSH 'PRODUCT'
 5104  025360  006204                        ASR     R4              ;FORM 'PRODUCT'
 5105  025362  005104                        COM     R4              ;COMPLEMENT MSH 'PRODUCT'
 5106  025364  070200                        MUL     R0,R2           ;MULTIPLY R2 BY 100000 LEAVING
 5107                                                                ;R2 = MSH, R3 = LSH PRODUCT
 5108  025366  020204                        CMP     R2,R4           ;COMPARE MSH PRODUCTS
 5109  025370  001002                        BNE     3$
 5110  025372  020003                        CMP     R0,R3           ;CHECK LSH PRODUCT
 5111  025374  001401                        BEQ     .+4
 5112  025376  104400                3$:     HLT
 5113  025400  104000                        SCOPE
 5114
 5115                                ;THE BELOW TEST OF THE DIV INSTRUCTION DIVIDES THE CURRENT PC BY
 5116                                ;1,2,4,8,ETC LEAVING THE QUOTIENT/REMAINDER IN R2/R3, NEXT THE QUOTIENT
 5117                                ;IS MULTPLIED BY 1,2,4,8,ETC AND THE REMAINDER ADDED, THE RESULT IS
 5118                                ;THEN COMPARED WITH THE ORIGINAL CURRENT PC,
 5119  025402  012700  000001        DIV0:   MOV     #1,R0           ;R0=DIVISOR
 5120  025406  010737  025500                MOV     PC,@#10$        ;SAVE DATA IN 10$
 5121  025412  013703  025500        1$:     MOV     @#10$,R3        ;GET DATA
 5122  025416  005002                        CLR     R2              ;CLEAR MSH DIVIDEND
 5123  025420  000277                        SCC
 5124  025422  071200                        DIV     R0,R2           ;DIVIDE R2 BY R0 LEAVING QUOTIENT IN R2
 5125                                                                ;AND REMAINDER IN R3
 5126  025424  103421                        BCS     2$
 5127  025426  100420                        BMI     2$
 5128  025430  102010                        BVC     20$             ;BRANCH IF DIVIDE WORKED
```



```
 5129  025432  022700  000001                CMP     #1,R0           ;V BIT SHOULD ONLY SET IF DIVIDING BY 1
 5130  025436  001014                        BNE     2$              ;AND THE LSH OF DIVIDEND
 5131  025440  032737  100000  025500        BIT     #100000,@#10$   ;IS NEGATIVE
 5132  025446  001410                        BEQ     2$
 5133  025450  000410                        BR      3$
 5134  025452  010204                2$:     MOV     R2,R4           ;GET QUOTIENT
 5135  025454  070400                        MUL     R0,R4           ;MULTIPLY QUOTIENT BY DIVISOR
 5136  025456  060305                        ADD     R3,R5           ;ADD REMAINDER TO LSH PRODUCT
 5137  025460  103403                        BCS     2$              ;SHOULD BE NO CARRY
 5138  025462  023705  025500                CMP     @#10$,R5        ;CHECK RESULT
 5139  025466  001401                        BEQ     .+4
 5140  025470  104400                2$:     HLT                     ;ERROR! DIVIDE FAILED
 5141                                                                ;QUOTIENT IS IN R2,REMAINDER IN R3
 5142                                                                ;ORIGINAL PC IS IN 10$ AND FINAL
 5143                                                                ;PRODUCT IN R4,R5 [MSH][LSH]
 5144  025472  006300                3$:     ASL     R0              ;GET NEXT DIVISOR
 5145  025474  102346                        BVC     1$
 5146  025476  000401                        BR      ASHL1
 5147  025500  000000                10$:    .WORD   0               ;CONTAINS ORIGINAL PC USED AS DATA
 5148
 5149                                ;CHECK ASH,ASHC,MUL, AND DIV INSTRUCTIONS USING ADDRESS MODE 1
 5150  025502  005016                ASHL1:  CLR     (SP)            ;(SP) = SHIFT COUNT
 5151  025504  005000                        CLR     R0              ;R0 = SHIFT COUNT FOR CHECK ASH
 5152  025506  012702  000020                MOV     #16.,R2         ;R2 = MAX LEFT SHIFT COUNT
 5153  025512  005067  000012        1$:     CLR     2$              ;CLEAR CC'S HOLDING ADDRESS
 5154  025516  010703                        MOV     PC,R3           ;R3,R4 = DATA TO BE SHIFTED
 5155  025520  010304                        MOV     R3,R4
 5156  025522  072316                        ASH     (SP)+,R3        ;SHIFT R3 LEFT (SP) TIMES
 5157  025524  013727  177776                MOV     @#PSW,(PC)+     ;SAVE CC'S
 5158  025530  000000                2$:     .WORD   0               ;CONTAINS ASH (SP),R3 CC'S IN EVEN BYTE
 5159                                                                ;AND ASH R0,R4 CC'S IN ODD BYTE
 5160  025532  072400                        ASH     R0,R4           ;SHIFT R4 LEFT R0 TIMES
 5161  025534  113767  177776  177767        MOVB    @#PSW,2$+1      ;SAVE CC'S IN ODD BYTE OF 2$
 5162  025542  020304                        CMP     R3,R4           ;COMPARE RESULTS
 5163  025544  001004                        BNE     3$              ;BRANCH IF THEY DO NOT COMPARE
 5164  025546  126767  177756  177755        CMPB    2$,2$+1         ;CHECK CC'S AFTER ASH INSTRUCTIONS
 5165  025554  001401                        BEQ     .+4
 5166  025556  104400                3$:     HLT                     ;ERROR! EITHER RESULTS OF SHIFT OR
 5167                                                                ;RESULT CC'S ARE INCORRECT
 5168  025560  005200                        INC     R0              ;INCREMENT SHIFT COUNT FOR ASH R0,R4
 5169  025562  005216                        INC     (SP)            ;INCREMENT SHIFT COUNT FOR ASH (SP),R3
 5170  025564  020200                        CMP     R2,R0           ;CHECK FOR MAX SHIFT COUNT
 5171  025566  001351                        BNE     1$
 5172
 5173  025570  005016                ASHR1:  CLR     (SP)            ;(SP) = SHIFT COUNT FOR ASH (SP),R4
 5174  025572  005000                        CLR     R0              ;R0 = SHIFT COUNT FOR ASH R0,R5
 5175  025574  005402                        NEG     R2              ;R2 = MAX RIGHT SHIFT COUNT (SET BY
 5176                                                                ;ABOVE TEST TO 16, NOW = -16,
 5177  025576  005067  000012        1$:     CLR     2$              ;CLEAR CC'S HOLDING ADDRESS
 5178  025602  010704                        MOV     PC,R4           ;R4,R5 = DATA TO BE SHIFTED RIGHT
 5179  025604  010405                        MOV     R4,R5
 5180  025606  072416                        ASH     (SP)+,R4        ;SHIFT R4 RIGHT (SP) TIMES
 5181  025612  013727  177776                MOV     @#PSW,(PC)+     ;SAVE CC'S
 5182  025614  000000                2$:     .WORD   0               ;CONTAINS ASH (SP),R4 CC'S IN EVEN BYTE
```

```
                                                        ;AND ASH R0,R5 CC'S IN ODD BYTE
025616  072500                  ASH     R0,R5           ;SHIFT R5 RIGHT R0 TIMES
025620  113767  177776  177767  MOVB    @#PSW,2$+1      ;SAVE CC'S IN ODD BYTE 2$
025626  020405                  CMP     R4,R5           ;CHECK RESULTS
025630  001004                  BNE     3$
025632  126767  177756  177755  CMPB    2$,2$+1         ;CHECK RESULT CC'S
025640  001401                  BEQ     .+4
025642  104400                  3$:     HLT             ;ERROR! EITHER RESULTS OR RESULT CC'S
                                                        ;DID NOT COMPARE
025644  005300                  DEC     R0              ;DECREMENT SHIFT COUNT
025646  005316                  DEC     (SP)            ;DECREMENT SHIFT COUNT FOR ASH (SP),R4
025650  020002                  CMP     R0,R2           ;CHECK FOR MAX RIGHT SHIFT
025652  001351                  BNE     1$
025654  104000                  SCOPE

025656  122737  000004  000764  CMPB    #4,@#OPT.CP     ;CHECK IF AN 11/40
025664  001002                  BNE     SPL0            ;BRANCH IF NOT AN 11/40
025666  002167  000354          JMP     MPI             ;GO TO MPI IF 11/40
                                ;CHECK SPL INSTRUCTION
025672  012702                  SPL0:   MOV     (PC)+,R2        ;R2 CONTAINS OP CODE FOR SPL 7
025674  000237                          SPL     7
025676  012704  177776                  MOV     #PSW,R4         ;R4 CONTAINS ADDRESS OF PSW
025702  011403                  MOV     (R4),R3         ;GET CURRENT PSW
025704  042703  177757          BIC     #177757,R3      ;R3 CONTAINS CORRECT PSW AFTER SPL

025710  012767  000230  000010  MOV     #SPL+0,2$       ;INITIALIZE SPL INSTRUCTIONS
025716  012767  000237  000050  MOV     #SPL+7,5$
025724  000257          1$:     CCC                     ;CLEAR CONDITION CODES
025726  000230          2$:     SPL     0               ;SET PRIORITY LEVEL
025730  121403                  CMPB    (R4),R3         ;CHECK RESULT OF SPL ABOVE
025732  001401                  BEQ     .+4
025734  104400                  HLT                     ;ERROR! SPL ABOVE FAILED
025736  032714  140000          BIT     #UM,(R4)        ;IF NOT IN KERNEL MODE THEN SPL
025742  001002                  BNE     3$              ;ACTS AS A NOP
025744  062703  000040          ADD     #40,R3          ;SET NEXT CORRECT PSW RESULT
025750  005267  177752          3$:     INC     2$      ;SET NEXT SPL INSTRUCTION
025754  026702  177746          CMP     2$,R2           ;CHECK IF DONE
025760  002761                  BLT     1$              ;LOOP UNTIL DONE CHANGING SPL EACH PASS
025762  012702                  MOV     (PC)+,R2        ;R2 CONTAINS SPL INSTRUCTION BELOW
025764  000230                  SPL     0
025766  052703  000017          BIS     #17,R3          ;SET CONDITION CODE RESULT INTO R3
025772  000277          4$:     SCC                     ;SET CONDITION CODES
025774  000237          5$:     SPL     7               ;SET PRIORITY LEVEL
025776  121403                  CMPB    (R4),R3         ;CHECK RESULT OF SPL ABOVE
026000  001401                  BEQ     .+4
026002  104400                  HLT                     ;ERROR! SPL ABOVE FAILED
026004  032714  140000          BIT     #UM,(R4)        ;CHECK IF IN KERNEL MODE
026010  001002                  BNE     6$
026012  162703  000040          SUB     #40,R3          ;SET NEXT CORRECT PSW RESULT
026016  005367  177752          6$:     DEC     5$      ;SET NEXT SPL
026022  026702  177746          CMP     5$,R2           ;CHECK IF DONE ALL SPL'S
026026  002361                  BGE     4$
026030  104000                  SCOPE
```



```
                                ;CHECK PROGRAM INTERRUPT REQUEST 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-
                                ;CESSER LEVEL THAT AN INTERRUPT OCCURS
026032  012700  026172          PIRQ0:  MOV     #4$,R0          ;R0 POINTS TO A TABLE OF CORRECT PIRQ
                                                                ;CONTENTS AFTER AN INTERRUPT
026036  012702  000400                  MOV     #400,R2         ;R2 CONTAINS INTERRUPT REQUEST LEVEL
026042  005003                  CLR     R3              ;R3 CONTAINS PROCESSER PRICRITY LEVEL
026044  012704  177772                  MOV     #PIRQ,R4        ;R4 CONTAINS ADDRESS OF PIRQ REGISTER
026050  005014                  CLR     (R4)            ;INITIALIZE REQUEST LEVEL TO 0
026052  013737  177776  000242          MOV     @#PSW,@#PIRVEC+2        ;RETAIN MODE & REG SET ON TRAP
026060  112737  000340  000242  MOVB    #PRTY7,@#PIRVEC+2       ;ASSUME LEVEL 7 ON INTERRUPT
026066  012737  026170  000014          MOV     #30$,@#TBITVEC          ;SET NEW TBIT TRAP VECTTOR
026074  012737  000340  000016          MOV     #PRTY7,@#TBITVEC+2      ;PRIORITY LEVEL 7 ON TRAP
026102  012737  026142  000240  1$:     MOV     #2$,@#PIRVEC            ;SET PIRQ ERROR INTERRUPT VECTOR
026110  063737  001004  000240          ADD     @#FACTOR,@#PIRVEC       ;ADD RELOCATION FACTOR
026116  110337  177776                  MOVB    R3,@#PSW                ;SET CP PRIORITY LEVEL
026122  050214                  BIS     R2,(R4)         ;MAKE REQUEST AT LEVEL = TO CP LEVEL
026124  100431                  BMI     5$              ;BRANCH WHEN DONE
026126  062737  000002  000240          ADD     #3$-2$,@#PIRVEC         ;SET PIRQ INTERRUPT VECTOR TO 3$
026134  006302                  ASL     R2
026136  050214                  BIS     R2,(R4)         ;MAKE REQUEST AT LEVEL 1 HIGHER
026140  000240                  NOP
026142  104400          2$:     HLT                     ;ERROR! EITHER AN INTERRUPT OCCURED
                                                        ;WHEN RQST LEVEL = CP LEVEL (PIRVEC)=2$
                                                        ;OR INTERRUPT FAILED (PIRVEC)=3$
026144  022014          3$:     CMP     (R0)+,(R4)      ;CHECK CONTENTS OF PIRQ REGISTER
026146  001401                  BEQ     .+4
026150  104400                  HLT                     ;ERROR! INCORRECT PIRQ CONTENTS
026152  062703  000040          ADD     #40,R3          ;SET NEXT CP PRIORITY LEVEL
026156  040214                  BIC     R2,(R4)         ;LOWER LEVEL BY 1
026160  012716  026102          MOV     #1$,(SP)        ;ADJUST RETURN ADDRESS
026164  063716  001004          ADD     @#FACTOR,(SP)   ;TO RETURN TO 1$
026170  000006          30$:    RTT

                                ;TABLE OF CORRECT PIRQ REGISTER CONTENTS ON INTERRUPT
026172  001042          4$:     1042                    ;PIR1+PIA1
026174  003104                  3104                    ;PIR2+PIR1+PIA2
026176  007146                  7146                    ;PIR3+PIR2+PIR1+PIA3
026200  017210                  17210                   ;PIR4+PIR3+PIR2+PIR1+PIA4
026202  037252                  37252                   ;PIR5+PIR4+PIR3+PIR2+PIR1+PIA5
026204  077314                  77314                   ;PIR6+PIR5+PIR4+PIR3+PIR2+PIR1+PIA6
026206  177356                  177356                  ;PIR7+PIR6+PIR5+PIR4+PIR3+PIR2+PIR1+PIA7

026210  005014          5$:     CLR     (R4)            ;CLEAR PIRQ REGISTER
026212  012737  000242  000240  MOV     #PIRVEC+2,@#PIRVEC      ;RESET PIRVEC TO HALT AT PIRVEC+2
026220  005037  000242          CLR     @#PIRVEC+2
026224  105037  177776          CLRB    @#PSW
026230  012737  000006  000016  MOV     #6,@#TBITVEC+2          ;RESTORE 'T' BIT TRAP TO RETURN
026236  012737  000016  000014  MOV     #TBITVEC+2,@#TBITVEC    ;VIA RTT IN TBITVEC+2
026244  104000                  SCOPE

                                ;CHECK MFPI/MTPI INSTRUCTIONS
```

```
5291  026246  032737  14 020  177776  MPI:    BIT     #UM,@#PSW       ;KERNEL MODE?
5292  026254  001537                          BEQ     ENDCP           ;YES EXIT TEST
5293  026256  010746                          MOV     PC,-(SP)
5294  026260  062716  000144                  ADD     #55$-.,(SP)
5295  026264  012637  000250                  MOV     (SP)+,@#MMVEC   ;SET MEM MGMT ABORT VECTOR
5296  026270  005046                          CLR     -(SP)           ;CLEAR CHECK WORD
5297  026272  010603                          MOV     SP,R3
5298  026274  010346                          MOV     R3,-(SP)        ;PUT ADDRESS OF CHECK WORD ON THE STACK
5299  026276  105737  000770                  TSTB    @#MMON          ;CHECK IF MEM MGMT IS ENABLED
5300  026302  001423                          BEQ     1$              ;BRANCH IF OFF
5301  026304  013737  177640  177654          MOV     @#UIPAR0,@#UIPAR6       ;SET UP USER PAGE ADDR, REG,
5302  026312  012737  006006  177614          MOV     #6006,@#UIPDR6  ;SET USER PAGE DESC REG R/W UP 6 PAGES
5303  026320  122737  000004  000764          CMPB    #4,@#OPT,CP     ;BRANCH IF 11/40
5304  026326  001406                          BEQ     10$
5305  026330  013737  172240  172254          MOV     @#SIPAR0,@#SIPAR6
5306  026336  012737  006006  172214          MOV     #6006,@#SIPDR6  ;SET SUPER PAGE DESC, REG,
5307  026344  062706  140000          10$:    ADD     #140000,SP      ;SET CURRENT MODE'S STACK POINTER
5308  026350  000240                          NOP
5309  026352  010746                  1$:     MOV     PC,-(SP)
5310  026354  062716  000024                  ADD     #3$-.,(SP)
5311  026360  012637  000030                  MOV     (SP)+,@#EMTVEC  ;SET EMT TRAP VECTOR
5312  026364  104000                          EMT                     ;TRAP TO 3$ BELOW
5313  026366  005266  000002                  INC     2(SP)           ;INCREMENT CHECK WORD
5314  026372  001417                          BEQ     6$
5315  026374  104400                  4$:     HLT                     ;ERROR! MFPI,MTPI FAILURE-FOR BETTER
5316  026376  000415                          BR      6$              ;ISOLATION SUGGEST RUNNING MFPI DIAG, DCKTD/E
5317  026400  000240                  3$:     NOP                     ;PSW=KERNEL MODE,PREV USER OR SUPER MODE
5318  026402  006506                          MFPI    SP              ;GET PREV. MODES STACK POINTER
5319  026404  006536                          MFPI    @(SP)+          ;GET DATA (AN ADDRESS) ON PREV MODE'S STACK
5320  026406  006576  000000                  MFPI    @(SP)           ;GET DATA (=0) FROM PREV MODES ADDRESS
5321  026412  000240                          NOP                     ;SPACE AND PUSH ONTO KERNEL STACK
5322  026414  001367                          BNE     4$              ;ERROR IF BRANCH TAKEN! SHOULD HAVE A ZERO ON THE STACK
5323  026416  005116                          COM     (SP)            ;COMPLEMENT OPERAND
5324  026420  006636                          MTPI    @(SP)+          ;POP OPERAND OFF KERNEL STACK AND MOVE
5325                                                                  ;IT TO PREV MODE'S SPACE
5326  026422  000002                          RTI                     ;RETURN TO INST FOLLOWING EMT ABOVE
5327  026424  104400                  5$:     HLT                     ;ERROR! MEMORY HANG, ABORT
5328  026426  105037  177776                  CLRB    @#PSW           ;SET PRIORITY LEVEL BACK TO 0
5329  026432  012737  005244  000250  6$:     MOV     #KTABRT,@#MMVEC ;RESTORE VECTOR
5330  026440  012737  001014  000030          MOV     #SCOPE1A,@#EMTVEC
5331  026446  012706  000500                  MOV     #STKPTR,SP      ;RESTORE STACK POINTER
5332  026452  104000                          SCOPE
5333
5334                                  ;CHECK THAT HALT INSTRUCTION TRAPS TO 4 (11/45),10 (11/40) IN USER/SUPER MODE
5335  026454  010746                  HALT1:  MOV     PC,-(SP)        ;GET CURRENT PC
5336  026456  062716  000022                  ADD     #2$-.,(SP)
5337  026462  011637  000004                  MOV     (SP),@#ERRVEC   ;SET ERROR TRAP VECTOR TO 2$ BELOW
5338  026466  012637  000010                  MOV     (SP)+,@#RESVEC  ;LOAD RESERVED INST TRAP VECTOR (11/40)
5339  026472  000000                          HALT                    ;SHOULD TRAP TO 4 IN USER/SUPER MODE
5340  026474  104400                  1$:     HLT                     ;ERROR! HALT ABOVE FAILED IN USER/SUPER MODE
5341  026476  000404                          BR      3$
5342  026500  010716                  2$:     MOV     PC,(SP)         ;REPLACE RETURN PC WITH
5343  026502  062716  000006                  ADD     #3$-.,(SP)      ;ADDRESS OF 3$ BELOW
5344  026506  000002                          RTI                     ;RETURN (TO 3$)
```



```
5345
5346  026510  012737  005274  000004  3$:     MOV     #ERRRT,@#ERRVEC ;RESTORE ERROR TRAP VECTOR
5347  026516  012737  005264  000010          MOV     #RESERR,@#RESVEC
5348  026524  104000                          SCOPE
5349
5350                                  ;TEST THAT RESET IS A 'NOP' IN USER/SUPER MODE
5351  026526  000277                  RESET1: SCC
5352  026530  013700  177776                  MOV     @#PSW,R0        ;GET CURRENT PSW
5353  026534  000277                          SCC
5354  026536  000005                          RESET
5355  026540  023700  177776                  CMP     @#PSW,R0        ;CHECK THAT PSW UNCHANGED BY RESET ABOVE
5356  026544  001401                          BEQ     .+4
5357  026546  104400                          HLT                     ;ERROR! RESET CLEARED MODE BITS IN PSW
5358  026550  010037  177776                  MOV     R0,@#PSW        ;RESTORE PSW (FOR ERROR)
5359  026554  104000                  ENDCP:  SCOPE
5360
5361  026556  010702                          MOV     PC,R2
5362  026560  062702  000012                  ADD     #12,R2
5363  026564  012707  001152                  MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
5364  026570  000000                  REL66:  .WORD   0
5365                                  ;66666666666666 LAST ADDRESS OF CODE TO BE RELOCATED 66666666666
5366
5367
5368                                          .SBTTL  START OF SECTION 7
5369                                  ;77777777777777 FIRST ADDRESS TO BE RELOCATED 777777777
5370  026572  010700                  REL7:   MOV     PC,R0           ;GET PC
5371  026574  005740                          TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL7
5372  026576  010037  001010                  MOV     R0,@#FRSTAD     ;SAVE
5373  026602  012737  000007  005176          MOV     #7,@#SECT       ;SET SECTION #
5374  026610  004737  005166                  JSR     PC,@#LODISP     ;LOAD DISPLAY REG
5375  026614  013767  005172  001014          MOV     @#DISPLY,REL77
5376  026622  010700                          MOV     PC,R0           ;GET CURRENT PC
5377  026624  162700  026624                  SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
5378  026630  010037  001004                  MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
5379  026634  010701                          MOV     PC,R1           ;SET NEW SCOPE PTR
5380                                  ;TEST STACK LIMIT REGISTER
5381                                  ;THIS TEST SHIFTS A '1' BIT THROUGH ALL BIT POSITIONS
5382  026636  032737  004000  000764  STKLIM: BIT     #KJOPT,@#OPT,CP         ;CHECK IF OPTION IS AVAILABLE
5383  026644  001512                          BEQ     10$             ;EXIT IF NOT AVAILABLE
5384  026646  012702  177774                  MOV     #SLR,R2         ;GET ADDRESS OF STACK LIM REG
5385  026652  005022                          CLR     (R2)+           ;CLEAR STACK LIMIT REG
5386  026654  032712  000020                  BIT     #T,(R2)         ;EXIT TEST IF 'T' BIT IS SET
5387  026660  001104                          BNE     10$
5388  026662  052712  000340                  BIS     #340,(R2)       ;SET PRIORITY LEVEL 7 TO PREVENT
5389                                                                  ;ANY INTERRUPTS FROM OCCURRING
5390  026666  012700  000400                  MOV     #400,R0         ;SET CHECK DATA
5391  026672  010042                  1$:     MOV     R0,-(R2)        ;MOVE TO STACK LIMIT REG
5392  026674  022220                          CMP     (R2)+,R0        ;AND CHECK RESULT
5393  026676  001401                          BEQ     2$
5394  026700  104400                          HLT                     ;ERROR! STACK LIMIT DID NOT
5395                                                                  ;LOAD CORRECTLY, CORRECT RESULT
5396                                                                  ;IS IN R0
5397  026702  006300                  2$:     ASL     R0              ;SHIFT '1' BIT LEFT
5398  026704  103372                          BCC     1$              ;LOOP UNTIL 1 BIT SHIFTS OUT
```

```
5399  026726  2 5042                          CLR     @(R2)                   ;CLEAR STACK LIMIT REG
5400
5401                                          ;THIS TEST CHECKS THAT A PROPER 'RED' ZONE VIOLATION OCCURS, NOTE THAT
5402                                          ;NO 'RED ZONE' VIOLATION WILL OCCU' IF IN USER/SUPER MODES,
5403                                          ;A RED ZONE VIOLATION PUSHES THE CURRENT PSW,PC ON A STACK AT 2 AND 0
5404                                          ;AND TAKES THE NEXT INSTRUCTION FROM THE PC IN LOCATION4, THE INST-
5405                                          ;RUCTION CAUSING THE RED ZONE VIOLATION IS 'ABORTED',
5406  026710  010746                          MOV     PC,-(SP)                ;GET CURRENT PC
5407  026712  062716  000054                  ADD     #4$-.,(SP)              ;FORM ADDRESS OF 4$ BELOW
5408  026716  012637  000004                  MOV     (SP)+,@#ERRVEC          ;SET ERROR TRAP VECTOR TO 4$ BELOW
5409  026722  013737  177776  000006          MOV     @#PSW,@#ERRVEC+2        ;RETAIN CURRENT STATUS ON TRAP
5410  026730  010712                          MOV     PC,(R2)                 ;SET STACK LIMIT TO CURRENT PC
5411                                                                          ;+400
5412  026732  011206                          MOV     (R2),SP                 ;AND STACK PTR = STACK LIMIT REG
5413  026734  010603                          MOV     SP,R3                   ;SAVE STACK PTR
5414  026736  016304  000336                  MOV     336(R3),R4              ;SAVE MEMORY LOC CONTENTS
5415                                                                          ;AT 'RED ZONE' BOUNDARY
5416  026742  032737  140000  177776          BIT     #UM,@#PSW               ;BRANCH IF IN KERNEL MODE
5417  026750  001403                          BEQ     20$
5418  026752  010466  000336                  MOV     R4,336(SP)              ;SHOULD NOT CAUSE TRAP
5419  026756  000430                          BR      100$
5420
5421  026760  005066  000336          20$:    CLR     336(SP)                 ;SHOULD CAUSE 'RED ZONE' TRAP
5422  026764  104400                  3$:     HLT                             ;ERROR! FAILED TO TRAP
5423
5424  026766  032737  140000  000002  4$:     BIT     #UM,@#2                 ;CHECK IF TRAPPED WHEN IN USER
5425                                                                          ;/SUPER MODES (2 CONTAINS OLD PSW)
5426  026774  001013                          BNE     99$                     ;GO TO ERROR CALL
5427  026776  010600                          MOV     SP,R0                   ;STACK PTR SHOULD = 0
5428  027000  001011                          BNE     99$                     ;GO TO ERROR CALL IF NOT 0
5429  027002  026304  000336                  CMP     336(R3),R4              ;CHECK THAT INST WAS ABORTED
5430  027006  001006                          BNE     99$                     ;GO REPORT ERRPR
5431  027010  005012                  5$:     CLR     (R2)                    ;CLEAR STACK LIMIT REG
5432  027012  010705                          MOV     PC,R5                   ;GET CURRENT PC
5433  027014  062705  177750                  ADD     #3$-.,R5                ;FORM ADDRESS OF  3$ ABOVE
5434  027020  020516                          CMP     R5,(SP)                 ;CHECK THAT RETURN PC IS ON
5435                                                                          ;THE STACK (AT 0)
5436  027022  001406                          BEQ     100$                    ;EXIT TEST
5437
5438                                          ;ERROR
5439  027024  005012                  99$:    CLR     (R2)                    ;CLEAR STACK LIMIT REG
5440  027026  010463  000336                  MOV     R4,336(R3)              ;RESTORE MEM LOCATION
5441  027032  012706  000500                  MOV     #STKPTR,SP              ;SET STACK PTR
5442  027036  104400                          HLT                             ;ERROR!
5443  027040  010463  000336          100$:   MOV     R4,336(R3)              ;RESTORE MEM LOCATION
5444  027044  005022                          CLR     (R2)+                   ;CLEAR STACK LIM REG
5445  027046  012706  000500                  MOV     #STKPTR,SP              ;SET STACK PTR
5446  027052  042712  000340                  BIC     #340,(R2)               ;SET PRIORITY LEVEL BACK TO 0
5447  027056  012737  005274  000004          MOV     #ERRRT,@#ERRVEC         ;RESTORE ERROR TRAP VECTOR
5448  027064  012737  000002  000006          MOV     #RTI,@#ERRVEC+2
5449  027072  104000                  101$:   SCOPE
5450
5451                                          ;MEMORY MANAGEMENT REGISTER TESTS
5452                                          ;PDR TEST - THIS TEST WRITES/READS A COMPLIMENTING BINARY
```



```
5453                                          ;COUNT PATTERN INTO ALL PDR REGISTERS
5454
5455  027074  005737  000764          KTPDR:  TST     @#ORT,CP                ;EXIT TEST IF NO KT OPTION
5456  027100  100124                          RPL     KT1
5457  027102  105737  000770                  TSTB    @#MMON                  ;EXIT TEST IF KT IS ENABLED
5458  027106  001121                          BNE     KT1
5459  027110  012702  027316                  MOV     #PDRTBL,R2              ;SET TABLE ADDRESS OF PDR'S
5460  027114  012705  100360                  MOV     #100360,R5              ;SET BIT MASK (11/45)
5461  027120  122737  000004  000764          CMPB    #4,@#OPT,CP             ;BRANCH IF 11/45
5462  027126  001005                          BNE     1$
5463  027130  005062  000004                  CLR     4(R2)                   ;TERMINATE TABLE AT SIPDR0
5464  027134  005062  000022                  CLR     22(R2)
5465  027140  005205                          INC     R5                      ;SET BIT MASK (11/40)
5466  027142  012200                  1$:     MOV     (R2)+,R0                ;GET PDR ADDRESS
5467  027144  001423                          BEQ     100$                    ;EXIT ON '0' TERMINATOR
5468  027146  012703  000010                  MOV     #8.,R3                  ;SET LOOP COUNT (FOR 8 REGS)
5469  027152  005004                  3$:     CLR     R4                      ;INITIALIZE DATA TO BE WRITTEN
5470  027154  040504                  4$:     BIC     R5,R4                   ;CLEAR NON-SETTABLE BITS
5471  027156  010410                          MOV     R4,(R0)                 ;WRITE INTO PDR
5472  027160  021004                          CMP     (R0),R4                 ;AND CHECK DATA READ BACK
5473  027162  001012                          BNE     99$                     ;GO TO ERROR CALL
5474  027164  005104                          COM     R4                      ;COMPLEMENT DATA
5475  027166  040504                          BIC     R5,R4                   ;CLEAR NON-SETTABLE BITS
5476  027170  010410                          MOV     R4,(R0)                 ;WRITE COMPLEMENT DATA INTO PDR
5477  027172  021004                          CMP     (R0),R4                 ;AND CHECK
5478  027174  001005                          BNE     99$                     ;GO TO ERROR CALL
5479  027176  005404                          NEG     R4                      ;STEP DATA
5480  027200  100365                          BPL     4$                      ;AND LOOP UNTIL BINARY COUNT
5481                                                                          ;FINISHED
5482  027202  005020                  5$:     CLR     (R0)+                   ;STEP TO NEXT REGISTER
5483  027204  077316                          SOB     R3,3$                   ;UNTIL 8 REGISTERS ARE DONE
5484  027206  000755                          BR      1$                      ;GET NEXT SET OF 8 REGISTERS
5485  027210  104400                  99$:    HLT                             ;ERROR! INCORRECT DATA READ
5486                                                                          ;BACK FROM PDR, ADDRESS OF
5487                                                                          ;PDR IS IN R0, DATA IS IN R4
5488  027212  000773                          BR      5$                      ;STEP TO NEXT REGISTER
5489  027214  104000                  100$:   SCOPE
5490
5491                                          ;PAR TEST - THIS TEST WRITES/READS A COMPLEMENTING BINARY COUNT
5492                                          ;PATTERN INTO ALL PAR REGISTERS,
5493  027216  012702  027334          KTPAR:  MOV     #PARTBL,R2              ;GET TABLE ADDRESS OF PAR'S
5494  027222  012705  170000                  MOV     #170000,R5              ;SET BIT MASK
5495  027226  122737  000010  000764          CMPB    #10,@#OPT,CP
5496  027234  001001                          BNE     1$
5497  027236  005005                          CLR     R5
5498  027240  012200                  1$:     MOV     (R2)+,R0                ;GET PAR ADDRESS
5499  027242  001423                          BEQ     100$                    ;EXIT ON '0' TERMINATOR
5500  027244  012703  000010                  MOV     #8.,R3                  ;SET LOOP COUNT (FOR 8 REGS.)
5501  027250  005004                  3$:     CLR     R4                      ;INITIALIZE DATA
5502  027252  040504                  4$:     BIC     R5,R4                   ;CLEAR NON-SETTABLE BITS
5503  027254  010410                          MOV     R4,(R0)                 ;WRITE INTO PAR
5504  027256  021004                          CMP     (R0),R4                 ;AND CHECK
5505  027260  001012                          BNE     99$                     ;TAKE ERROR EXIT
5506  027262  005104                          COM     R4                      ;COMPLEMENT DATA
```

```
5507  027264  040504                  BIC     R5,R4           ;CLEAR NON-SETTABLE BITS
5508  027266  010410                  MOV     R4,(R0)         ;WRITE COMPLEMENT DATA
5509  027270  021004                  CMP     (R0),R4         ;AND CHECK
5510  027272  001005                  BNE     99$             ;TAKE ERROR EXIT
5511  027274  005404                  NEG     R4              ;STEP DATA
5512  027276  100365                  BPL     4$              ;LOOP UNTI FINISHED
5513
5514  027300  005020          5$:     CLR     (R0)+
5515  027302  077316                  SOB     R3,5$
5516  027304  000755                  BR      1$
5517
5518  027306  104400          99$:    HLT                     ;ERROR; INCORRECT DATA READ BACK
5519                                                          ;FROM PAR, ADDRESS OF PAR IS IN
5520                                                          ;R0, DATA IS IN R4
5521  027310  000773                  BR      5$              ;DO NEXT REGISTER
5522  027312  104000          100$:   SCOPE
5523  027314  000416                  BR      KT1
5524                          ;TABLES FOR PDR & PAR TESTS ABOVE
5525  027316  172300          PDRTBL: .WORD   KIPDR0
5526  027320  177600                  .WORD   UIPDR0
5527  027322  172200                  .WORD   SIPDR0          ;CHANGED TO '0' IF 11/40
5528  027324  172320                  .WORD   KDPDR0
5529  027326  177620                  .WORD   UDPDR0
5530  027330  172220                  .WORD   SDPDR0
5531  027332  000000                  .WORD   0               ;TERMINATOR
5532
5533  027334  172340          PARTBL: .WORD   KIPAR0
5534  027336  177640                  .WORD   UIPAR0
5535  027340  172240                  .WORD   SIPAR0          ;CHANGED TO '0' IF 11/40
5536  027342  172360                  .WORD   KDPAR0
5537  027344  177660                  .WORD   UDPAR0
5538  027346  172260                  .WORD   SDPAR0
5539  027350  000000                  .WORD   0               ;TERMINATOR
5540
5541  027352  105737  000770          KT1:    TSTB    @#MMON          ;BRANCH IF MEM MGMT NOT
5542  027356  001522                  BEQ     KTEX            ;ENABLED
5543  027360  005037  172354          CLR     @#KIPAR4        ;SET UP MEM MGMT REGISTERS
5544  027364  005037  172310          CLR     @#KIPDR4        ;TO ABORT IF A MEMORY
5545  027370  005037  177654          CLR     @#UIPAR4        ;REFERENCE IS MADE TO
5546  027374  005037  177610          CLR     @#UIPDR4        ;ADDRESSES (VIRTUAL) BETWEEN
5547  027400  122737  000004  000764  CMPB    #4,@#OPT,CP     ;100000-117776 IN ALL MODES
5548  027406  001404                  BEQ     1$
5549  027410  005037  172250          CLR     @#SIPAR4
5550  027414  005037  172210          CLR     @#SIPDR4
5551  027420  013746  000250          1$:     MOV     @#MMVEC,-(SP)   ;SAVE MEM MGMT VECTOR
5552  027424  013746  000252          MOV     @#MMVEC+2,-(SP) ;AND PRIORITY
5553  027430  010746                  MOV     PC,-(SP)        ;SET MEM MGMT
5554  027432  062716  000040          ADD     #4$-.,(SP)      ;VECTOR TO 4$ BELOW
5555  027436  012637  000250          MOV     (SP)+,@#MMVEC
5556  027442  013737  177776  000252  MOV     @#PSW,@#MMVEC+2
5557  027450  005000                  CLR     R0              ;CLEAR ABORT INDICATOR
5558  027452  010702                  MOV     PC,R2           ;SET R2 AND R3 NOTE:
5559  027454  012703  100000          MOV     #100000,R3      ;THE REF VIA R3 CAUSES THE
5560  027460  014223          2$:     MOV     -(R2),(R3)+     ;ABORT
```



```
5561  027462  005700          3$:     TST     R0              ;BRANCH IF THE ABORT OCCURRED
5562  027464  001001                  BNE     .+4
5563  027466  104400                  HLT                     ;REPORT ERROR
5564  027470  000451                  BR      100$
5565                          ;ABORT HERE
5566  027472  013700  177776  4$:     MOV     @#PSW,R0        ;$R0 SHOULD CONTAIN
5567  027476  000300                  SWAB    R0              ;CAUSE FOR ABORT AND
5568  027500  006200                  ASR     R0              ;ALSO WHICH SEGMENT
5569  027502  042700  177637          BIC     #177637,R0      ;WAS IN USE WHEN ABORT
5570  027506  062700  100011          ADD     #100011,R0      ;OCCURRED,
5571  027512  020037  177572          CMP     R0,@#SR0
5572  027516  001031                  BNE     99$
5573  027520  012700  027460          MOV     #2$,R0          ;GET ADDRESS OF INST THAT ABORTED
5574  027524  020037  177576          CMP     R0,@#SR2        ;THAT ABORTED
5575  027530  001024                  BNE     99$
5576  027532  122737  000004  000764  CMPB    #4,@#OPT,CP
5577  027540  001414                  BEQ     5$
5578  027542  012700  000362          MOV     #362,R0
5579  027546  120037  177574          CMPB    R0,@#SR1        ;$R1 (11/45) CONTAINS REGISTER
5580  027552  001013                  BNE     99$             ;MODIFICATIONS MADE
5581  027554  012700  000023          MOV     #23,R0
5582  027560  120037  177575          CMPB    R0,@#SR1+1
5583  027564  001006                  BNE     99$
5584  027566  012700  027460          MOV     #2$,R0
5585  027572  005720          5$:     TST     (R0)+           ;R0=ADDRESS OF INST FOLLOWING ABORT
5586  027574  020016                  CMP     R0,(SP)         ;(3$)
5587  027576  001001                  BNE     99$
5588  027600  000002                  RTI                     ;RETURN
5589                          ;ENTER HERE ON ERROR
5590  027602  104400          99$:    HLT                     ;REPORT ERROR
5591  027604  010716                  MOV     PC,(SP)
5592  027606  062716  177654          ADD     #3$-.,(SP)
5593  027612  000002                  RTI                     ;RETURN
5594  027614  012637  000252          100$:   MOV     (SP)+,@#MMVEC+2 ;RESTORE ABORT VECTOR
5595  027620  012637  000250          MOV     (SP)+,@#MMVEC   ;& PRIORITY,
5596  027624                  KTEX:
5597
5598  027624  010702                  MOV     PC,R2
5599  027626  062702  000012          ADD     #12,R2
5600  027632  012707  001152          MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
5601  027636  000000          REL77:  .WORD   0
5602                          ;7777777777777 LAST ADDRESS OF CODE TO BE RELOCATED 77777777777
5603
5604                                  .SBTTL  TELETYPE & CLOCK TESTS
5605                          ;CHECK TTY INTERRUPT,
5606  027640  005037  001004  TTYCHK: CLR     @#FACTOR
5607  027644  010701                  MOV     PC,R1
5608  027646  032737  000400  000764  BIT     #TTOPT,@#OPT,CP ;BRANCH IF CONSOLE TTY AVAIL
5609  027654  001002                  BNE     1$
5610  027656  000167  000556          JMP     ARRFIN          ;JUMP IF NOT AVAILABLE
5611  027662  032737  000100  177564  1$:     BIT     #100,@#TPS      ;CHECK IF TTY IS READY
5612  027670  001374                  BNE     .-6
5613  027672  012737  027746  000064  MOV     #3$,@#TPVEC     ;SET TTY INTERRUPT VECTOR
5614  027700  012737  000200  000066  MOV     #200,@#TPVEC+2  ;PRIORITY LEVEL 4 ON INTERRUPT
```

```
5615  027726  0 2767  030034  000114          MOV     #NULLS,MSG      ;ADDRESS OF MESSAGE TO BE TYPPED
5616  027714  1 7737  000110  177566          MOVB    @MSG,@#TPB      ;TYPE FIRST CHARACTER OF MESSAGE
5617  027722  1 5737  177564                  TSTB    @#TPS
5618  027726  1 0375                          BPL     .-4
5619  027730  006237  177564                  ASR     @#TPS           ;SET IE BIT IN TTY CSR REG
5620  027734  000001                          WAIT                    ;WAIT FOR FIRST INTERRUPT
5621  027736  000440                          BR      KW11
5622  027740  006337  177564          2$:     ASL     @#TPS           ;CLEAR IE BIT
5623  027744  000002                          RTI
5624
5625  027746  122777  000012  000054  3$:     CMPB    #12,@MSG        ;CHECK IF CHARACTER IS <LF>
5626  027754  001020                          BNE     4$
5627  027756  006337  177564                  ASL     @#TPS           ;CLEAR IE BIT
5628  027762  052737  000340  177776          BIS     #PRTY7,@#PSW    ;SET PRIORITY LEVEL 7
5629  027770  013746  177776                  MOV     @#PSW,-(SP)     ;PUSH PSW ONTO STACK
5630  027774  004767  152564                  JSR     PC,,TYPE
5631  030000  000752                          CRLF
5632  030002  052737  000100  177564          BIS     #100,@#TPS      ;SET IE BIT
5633  030010  005267  000014                  INC     MSG             ;STEP POINTER
5634  030014  000002                          RTI
5635  030016  117737  000006  177566  4$:     MOVB    @MSG,@#TPB      ;TYPE CHARACTER
5636  030024  001745                          BEQ     2$              ;BRANCH IF TERMINATOR
5637  030026  005227                  5$:     INC     (PC)+           ;SET MSG TO NEXT CHAR ADDRESS
5638  030030  000000                  MSG:    .WORD   0               ;CONTAINS ADDRESS OF CHAR TO BE TYPED
5639  030032  000002                          RTI                     ;RETURN
5640  030034  020015  000015          NULLS:  .ASCIZ  <15><40><15>    ;CAR RET,SPACE,CAR RET,
5641                                          .EVEN
5642
5643                                  ;ROUTINE TO TURN ON KW11-P OR KW11-L LINE CLOCK IF AVAILABLE
5644  030040  010701                  KW11:   MOV     PC,R1
5645  030042  012737  030350  000100          MOV     #LKSRV,@#LKVEC          ;LOAD INTERRUPT VECTOR
5646  030050  012737  030400  000104          MOV     #PLKSRV,@#PLKVEC        ;FOR KW11-L & KW11-P CLOCKS
5647  030056  012737  000300  000102          MOV     #300,@#LKVEC+2          ;SET PRIORITY LEVEL 6 ON INT,
5648  030064  012737  000300  000106          MOV     #300,@#PLKVEC+2
5649  030072  032737  002000  000754          BIT     #PLKOPT,@#OPT,CP        ;BRANCH IF 'P' CLOCK NOT AVAIL
5650  030100  001407                          BEQ     10$
5651  030102  012737  000002  172542          MOV     #2,@#PLKCSB             ;LOAD COUNT SET BUFFER
5652  030110  012737  000101  172540          MOV     #101,@#PLKCSR           ;SET IE,100KHZ AND GO BITS
5653  030116  000415                          BR      1$
5654  030120  032737  001000  000764  10$:    BIT     #LKOPT,@#OPT,CP         ;BRANCH IF 'L' CLOCK NOT AVAIL
5655  030126  001560                          BEQ     ARBEX                   ;SKIP PRIORITY ARBITRATION TEST
5656                                                                          ;BELOW IF NO KW11-L OR KW11-P
5657  030130  012737  000100  177546          MOV     #100,@#LKS              ;SET IE BIT
5658  030136  012767  177546  000104          MOV     #LKS,6$
5659  030144  012767  000240  000174          MOV     #NOP,9$
5660
5661                                  ;ROUTINE TO CHECK PRIORITY ARBITRATION LOGIC
5662                                  ;THE BELOW TEST WILL INHIBIT INTERRUPTS ON LEVEL 6 AND ABOVE (LOCKING
5663                                  ;OUT THE LINE CLOCK) AND THEN SET UP THE TTY TO INTERRUPT, NEXT THE
5664                                  ;PRIORITY LEVEL WILL BE SET TO 0 ALLOWING INTERRUPTS IN WHICH CASE
5665                                  ;THE LINE CLOCK (AT LEVEL 6) SHOULD INTERRUPT BEFORE THE TTY (AT LEVEL 4).
5666  030152  132737  000020  177776  1$:     BITB    #20,@#PSW
5667  030160  001143                          BNE     ARBEX                   ;EXIT TEST IF 'T' BIT SET
5668  030162  032737  000100  177570          BIT     #100,@#SWR              ;BRANCH IF USER HAS DESLECTED
```



```
5669  030170  001137                          BNE     ARBEX                   ;CLOCKS
5670  030172  032737  000100  177564  2$:     BIT     #100,@#TPS              ;WAIT FOR TTY TO BE NOT
5671  030200  001374                          BNE     2$                      ;BUSY
5672  030202  112737  000300  177776          MOVB    #300,@#PSW              ;SET PRIORITY LEVEL 6
5673  030210  152737  000100  177564  3$:     BISB    #100,@#TPS              ;SET IE BIT
5674  030216  100374                          BPL     3$                      ;AND WAIT FOR EADY
5675  030220  013767  000064  000210          MOV     @#TPVEC,,TPVEC          ;SAVE TTY VECTOR
5676  030226  012737  030312  000064          MOV     #7$,@#TPVEC             ;SET TTY VECTOR
5677  030234  005027                          CLR     (PC)+                   ;CLEAR CHECK WORD
5678  030236  000000                  4$:     .WORD   0
5679  030240  000240                          NOP
5680  030242  000240                          NOP
5681  030244  000240                          NOP
5682  030246  113700                  5$:     MOVB    @(PC)+,R0               ;GET CLOCK STATUS & BRANCH IF READY
5683  030252  172540                  6$:     .WORD   PLKCSR                  ;CONTAINS ADDRESS OF L OR P CLOCK STATUS REG,
5684  030252  100375                          BPL     5$
5685  030254  000240                          NOP                             ;AT THIS TIME BOTH THE CLOCK
5686                                                                          ;AND THE TTY ARE READY TO INT
5687  030256  012737  030326  000100          MOV     #8$,@#LKVEC             ;SET CLOCK VECTOR
5688  030264  013737  000100  000104          MOV     @#LKVEC,@#PLKVEC
5689  030272  105037  177776                  CLRB    @#PSW                   ;SET PRIORITY LEVEL 0
5690
5691  030276  022767  000002  177732          CMP     #2,4$                   ;CHECK THAT THE CLOCK
5692  030304  001455                          BEQ     ARBFIN                  ;& TTY INTERRUPTED IN
5693  030306  104400                          HLT                             ;THE PROPER SEQUENCE
5694  030310  000453                          BR      ARBFIN
5695
5696  030312  042737  000100  177564  7$:     BIC     #100,@#TPS              ;CLEAR IE BIT
5697  030320  006367  177712                  ASL     4$                      ;SHIFT INDICATOR
5698  030324  000002                          RTI                             ;RETURN
5699
5700  030326  005267  177704          8$:     INC     4$
5701  030332  012737  030350  000100          MOV     #LKSRV,@#LKVEC          ;SET CLOCK VECTORS
5702  030340  012737  030400  000104          MOV     #PLKSRV,@#PLKVEC
5703  030346  000414                  9$:     BR      PLKSRV                  ;FINISH SERVICE (NOTE: CONTAINS NOP IF NO P CLOC
5704
5705
5706  030350  005267  150420          LKSRV:  INC     LTICKS                  ;INCREMENT CLOCK TICK COUNT
5707  030354  012737  000100  177546          MOV     #100,@#LKS              ;CLEAR READY
5708  030362  032737  000100  177570          BIT     #100,@#SWR              ;BRANCH IF USER DESIRES TO
5709  030370  001402                          BEQ     1$                      ;KEEP CLOCK ENABLED
5710  030372  005037  177546                  CLR     @#LKS                   ;DISABLE CLOCK
5711  030376  000002                  1$:     RTI                             ;RETURN
5712
5713                                  ;KW11-P INTERRUPT SERVICE
5714  030400  005267  150372          PLKSRV: INC     PTICKS
5715  030404  012737  000100  172542          MOV     #100,@#PLKCSB
5716  030412  012737  000101  172540          MOV     #101,@#PLKCSR           ;RE-ENABLE KW11-P
5717  030420  032737  000120  177570          BIT     #100,@#SWR
5718  030426  001402                          BEQ     1$
5719  030430  005037  172540                  CLR     @#PLKCSR
5720  030434  000002                  1$:     RTI
5721
5722  030436  000000                  .TPVEC: .WORD   0
```

```
 5723  030440  013737  03 436  000064  ARBFIN: MOV     @#,TPVEC,P#TPVEC        ;RESTORE TTY VECTOR
 5724  030446  042737  000100  177564          BIC     #100,@#TPS
 5725  030454  012737  030350  000100          MOV     #LKSRV,@#LKVEC          ;SET CLOCK VECTORS
 5726  030462  012737  030400  000104          MOV     #PLKSRV,@#PLKVEC
 5727  030470  104000                  ARBEX:  SCOPE
 5728
 5729                                          ;TURN ON KW11-L CLOCK IF BOTH ARE AVAILABLE
 5730  030472  032737  001000  000764          BIT     #LKOPT,@#OPT,CP         ;BRANCH IF NOT AVAIL
 5731  030500  001411                          BEQ     1$
 5732  030502  012737  030350  000100          MOV     #LKSRV,@#LKVEC          ;SET VECTOR
 5733  030510  012737  000300  000102          MOV     #300,@#LKVEC+2          ;AND PRIOITY LEVEL 6 ON INT,
 5734  030516  052737  000100  177546          BIS     #100,@#LKS              ;SET IE BIT
 5735  030524                          1$:
```



```
 5736                                          ;HIGH SPEED READER TESTS
 5737                                          ;TO RUN TEST LOAD PAPER TAPE TEST LOOP (A BINARY COUNT PATTERN) INTO
 5738                                          ;READER AND ENABLE READER,
 5739
 5740                                          ;ROUTINE TO SYNCHRONIZE READER, THE ROUTINE READS DATA UNTIL A 0 CHAR-
 5741                                          ;ACTER IS FOUND AND THEN SWITCHES THE INTERRUPT TO THE READER SERVICE
 5742                                          ;ROUTINE BELOW (PRTST)
 5743  030524  032737  002001  000766          BIT     #PROPT,@#OPTIONS        ;CHECK IF READER IS AVAILABLE
 5744  030532  001506                          BEQ     PREXIT
 5745  030534  012737  030572  000070          MOV     #PRSTRT,@#PRVEC         ;SET INTERRUPT VECTOR
 5746  030542  012737  000400  000072          MOV     #400,@#PRVEC+2          ;SET PRIORITY LEVEL 4
 5747  030550  112737  000001  000770          MOVB    #1,@#PRDAT              ;SET FIRST DATA CHAR
 5748  030556  105037  000771                  CLRB    @#PRSYNC                ;INITIALIZE SYNC COUNT
 5749  030562  012737  000101  177550          MOV     #101,@#PRS              ;SET READER ENABLE & IE
 5750  030570  000467                          BR      PREXIT                  ;CONTINUE TEST
 5751
 5752  030572  032737  100200  177550  PRSTRT: BIT     #100200,@#PRS           ;BRANCH IF READY & NO ERROR
 5753  030600  003002                          BGT     1$
 5754  030602  104400                          HLT
 5755  030604  000461                          BR      PREXIT
 5756
 5757  030606  105237  000771          1$:     INCB    @#PRSYNC                ;COUNT A MAXIMUM OF 128, NON
 5758  030612  001002                          BNE     2$                      ;'0' CHARACTERS
 5759  030614  104400                          HLT                             ;ERROR: MORE THAN 128 NON 0
 5760  030616  000410                          BR      3$                      ;CHARACTERS DETECTED
 5761
 5762  030620  105737  177552          2$:     TSTB    @#PRB                   ;BRANCH IF A NON 0 CHARACTER
 5763  030624  001005                          BNE     3$
 5764  030626  105037  000771                  CLRB    @#PRSYNC                ;RESET SYNC
 5765  030632  012737  030646  000070          MOV     #PRTST,@#PRVEC          ;SET VECTOR TO TEST
 5766  030640  005237  177550          3$:     INC     @#PRS                   ;SET READER ENABLE
 5767  030644  000002                          RTI                             ;EXIT INTERRUPT
 5768
 5769                                          ;HIGH SPEED READER TEST,
 5770                                          ;ROUTINE READS EACH CHARACTER FROM TAPE EXPECTING THE CHARACTER TO BE
 5771                                          ;1 GREATER THAN LAST ONE READ UNTIL A ZERO CHARACTER IS READ, WHEN
 5772                                          ;A ZERO CHARACTER IS READ THE TEST ALLOWS UP TO 128, 0 CHARACTERS TO
 5773                                          ;BE CONSECUTIVELY READ BEFORE A '1' CHARACTER IS FOUND,
 5774  030646  032737  100200  177550  PRTST:  BIT     #100200,@#PRS           ;BRANCH IF READY & NO ERROR
 5775  030654  003003                          BGT     1$
 5776  030656  104400                          HLT                             ;ERROR: ERROR BIT OR NOT READY
 5777  030660  005746                          TST     -(SP)                   ;PUSH FAKE CHAR ON STACK
 5778  030662  000426                          BR      100$                    ;EXIT VIA 100$
 5779
 5780  030664  113746  177552          1$:     MOVB    @#PRB,-(SP)             ;GET CHARACTER READ
 5781  030670  001413                          BEQ     2$                      ;BRANCH IF A '0'
 5782  030672  121637  000770                  CMPB    (SP)+,@#PRDAT           ;BRANCH IF INCORRECT DATA READ
 5783  030676  001017                          BNE     99$
 5784  030700  105237  000770                  INCB    @#PRDAT                 ;SET NEXT CHARACTER
 5785  030704  001015                          BNE     100$                    ;BRANCH IF NOT '0'
 5786  030706  105237  000770                  INCB    @#PRDAT                 ;MAKE NEXT CHAR TO BE A '1'
 5787  030712  105037  000771                  CLRB    @#PRSYNC                ;INITIALIZE SYNC COUNT
 5788  030716  000410                          BR      100$                    ;EXIT VIA 100$
 5789
```

```
5790  030720  12?  37  000001  000770  2$:     CMPB    #1,@#PRDAT              ;CAN ONLY READ A '0' BEFORE A '1'
5791  030726  001  33                          BNE     99$
5792  030730  105237  000771                  INCB    @#PRSYNC                ;COUNT MAXIMUM OF 128 '0' CHARS
5793  030734  001001                          BNE     100$
5794  030736  104400                  99$:    HLT                             ;READER DATA ERROR
5795  030740  005726                  100$:   TST     (SP)+                   ;POP DATA READ
5796  030742  005237  177550                  INC     @#PRS                   ;SET READER ENABLE
5797  030746  000002                          RTI                             ;EXIT TEST
5798
5799  030750                          PREXIT:
```



```
5800  030750  004767  000232          END:    JSR     PC,STMM                 ;GO START RELOCATING ABOVE 28K
5801  030754  012737  005274  000004  END1:   MOV     #ERRRT,@#ERRVEC
5802  030762  005037  177776                  CLR     @#PSW                   ;CLEAR MODE BITS IN PSW
5803  030766  004767  151540                  JSR     PC,CLRTBIT              ;GO CLEAR 'T' BIT IF SET
5804  030772  012706  000600                  MOV     #KPTR,SP                ;SET KERNEL STACK PTR
5805  030776  032737  000400  000764          BIT     #TTOPT,@#OPT,CP         ;BRANCH IF NO CONSOLE TTY
5806  031004  001433                          BEQ     1$
5807  031006  032737  000100  177564          BIT     #100,@#TPS              ;CHECK IF OUTPUT DEVICE IS BUSY
5808  031014  001374                          BNE     .-6                     ;IS AVAILABLE
5809  031016  105737  177570                  TSTB    @#SWR                   ;DELETE END OF PASS TYPE OUT IF SW7=0
5810  031022  100024                          BPL     1$                      ;BRANCH IF SW7 IS DOWN
5811  031024  016702  147750                  MOV     ICNT,R2                 ;GET PASS COUNT
5812  031030  004767  151636                  JSR     PC,CNVDAT                       ;CONVERT TO ASCIZ STRING
5813  031034  012702  003076                  MOV     #DIGITS+2,R2            ;GET ASCII VALUES
5814  031040  012703  003574                  MOV     #PASSES,R3              ;AND MOVE THEM INTO MESSAGE
5815  031044  012704  003750                  MOV     #PASSNO,R4
5816  031050  011223                          MOV     (R2)+,(R3)+
5817  031052  012224                          MOV     (R2)+,(R4)+
5818  031054  011223                          MOV     (R2),(R3)+
5819  031056  012224                          MOV     (R2)+,(R4)+
5820  031060  012737  003642  030030          MOV     #SUCCESS,@#MSG          ;PASS MESSAGE ADRS TO TELETYPE SERVICE
5821  031066  052737  000100  177564          BIS     #100,@#TPS              ;SET IE BIT
5822  031074  005267  147700          1$:     INC     ICNT
5823  031100  116700  147660                  MOVB    OPT,CP,R0               ;GET CP TYPE
5824  031104  026067  032252  147666          CMP     PASTAB(R0),ICNT         ;CHECK IF END OF TEST
5825  031112  001002                          BNE     2$                      ;BRANCH IF NOT AT END
5826  031114  000167  000776                  JMP     DONE
5827  031120  016702  147654          2$:     MOV     ICNT,R2                 ;GET PASS COUNT
5828  031124  006302                          ASL     R2
5829  031126  046002  032226                  BIC     CPPASS(0),R2            ;LIMIT PASS COUNT TO 0-6
5830  031132  012737  000016  000014          MOV     #16,@#14                ;RESTORE VECTOR  FOR RTT TEST
5831  031140  005037  000016                  CLR     @#16                    ;CLEAR T BIT TRAP ADDRESS
5832  031144  113737  001151  001150          MOVB    @#ITCNT+1,@#ITCNT       ;RESET ITERATION COUNT
5833  031152  016216  032206                  MOV     PSWTAB(2),(SP)          ;PUSH NEXT PASS PSW ON STACK
5834  031156  032716  000020                  BIT     #20,(SP)                ;WILL 'T' BIT BE SET ON NEXT PASS?
5835  031162  001406                          BEQ     3$                      ;BRANCH IF NOT
5836  031164  112737  000001  001150          MOVB    #1,@#ITCNT              ;SET ITERATION COUNT = 1 FOR 'T' BIT
5837  031172  016737  000006  000016          MOV     RTI3,@#16               ;SET 'T' BIT TRAP TO RETURN VIA 16
5838  031200  012746  005644          3$:     MOV     #START2,-(SP)           ;RESART PROGRAM AT START2
5839  031204  000006                  RTI1:   RTT                             ;RESTART PROGRAM AT START2 WITH NEW PSW
5840                                                                          ;(FROM TABLE BELOW)
5841
```

```
                                        ;SBTTL  STMM ROUTINE
                                ;ROUTINE TO SET UP MEMORY MANAGEMENT TO RELOCATE PROGRAM CODE ABOVE 28K
031206  005737  000764          STMM:   TST     @#OPT,CP        ;CHECK FOR MEM MGMT OPTION
031212  100401                          BMI     2$              ;BRANCH IF AVAILABLE
031214  000207                  1$:     RTS     PC              ;RETURN
031216  032737  001000  177570  2$:     BIT     #1000,@#SWR     ;BRANCH IF SW09 IS = 0
031224  001406                          BEQ     3$
031226  032737  010000  177570          BIT     #10000,@#SWR    ;JMP IF NO RELOCATION ABOVE 28K
031234  001010                          BNE     4$              ;IE SW12=0 & SW9=1
031236  000167  000614                  JMP     ENDM            ;EXIT
031242  032737  010000  177570  3$:     BIT     #10000,@#SWR    ;BRANCH IF SW12=0
031250  001402                          BEQ     4$
031252  000167  000600                  JMP     ENDM
031256  013727  177776          4$:     MOV     @#PSW,(PC)+     ;SAVE OLD PSW
031262  000000                  OLDPSW: ;WORD   0
031264  012737  000240  177776          MOV     #PRTY4,@#PSW    ;SET LEVEL 4 & KERNEL MODE
031272  004767  151234                  JSR     PC,CLRTBIT      ;CO CLEAR 'T' BIT IF SET
031276  012700  077406                  MOV     #77406,R0
031302  010037  172300                  MOV     R0,@#KIPDR0     ;SET KIPDR0,1,& 7 R/W UP 4K WORDS
031306  010037  172302                  MOV     R0,@#KIPDR1
031312  010037  172304                  MOV     R0,@#KIPDR2
031316  010037  172306                  MOV     R0,@#KIPDR3
031322  010037  172316                  MOV     R0,@#KIPDR7
031326  005037  172340                  CLR     @#KIPAR0
031332  012737  000200  172342          MOV     #200,@#KIPAR1
031340  016737  000442  172344          MOV     NEXPAR,@#KIPAR2
031346  013737  172344  172346          MOV     @#KIPAR2,@#KIPAR3
031354  062737  000200  172346          ADD     #200,@#KIPAR3
031362  012737  177600  172356          MOV     #177600,@#KIPAR7
031370  005046                          CLR     -(SP)
031372  032737  011000  177570          BIT     #11000,@#SWR    ;
031400  001006                          BNE     1$
031402  122737  000010  000764          CMPB    #10,@#OPT,CP
031410  001002                          BNE     1$
031412  012716  000020                  MOV     #20,(SP)
031416  010037  177600          1$:     MOV     R0,@#UIPDR0     ;SET UP USER MEM MGMT REGS
031422  010037  177602                  MOV     R0,@#UIPDR1
031426  010037  177616                  MOV     R0,@#UIPDR7
031432  016737  000350  177640          MOV     NEXPAR,@#UIPAR0
031440  013737  177640  177642          MOV     @#UIPAR0,@#UIPAR1
031446  062737  000200  177642          ADD     #200,@#UIPAR1
031454  013737  172356  177656          MOV     @#KIPAR7,@#UIPAR7
031462  122737  000004  000764          CMPB    #4,@#OPT,CP     ;BRANCH IF AN 11/40
031470  001424                          BEQ     3$
031472  010037  172200                  MOV     R0,@#SIPDR0     ;SET UP SUPERVISOR MEM MGMT REGS
031476  010037  172202                  MOV     R0,@#SIPDR1
031502  010037  172216                  MOV     R0,@#SIPDR7
031506  016737  000274  172240          MOV     NEXPAR,@#SIPAR0
031514  013737  172240  172242          MOV     @#SIPAR0,@#SIPAR1
031522  062737  000200  172242          ADD     #200,@#SIPAR1
031530  013737  172356  172256          MOV     @#KIPAR7,@#SIPAR7
```



```
031536  011637  172516                  MOV     (SP),@#SR3              ;SETUP SR3

031542  005726                  3$:     TST     (SP)+                   ;POP STACK
031544  000240                          NOP
031546  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEM MGMT (PROG IS IN KER MODE)
031554  105237  000770                  INCB    @#MMON          ;SET MEM MGMT ON IND = ON
031560  004767  153402                  JSR     PC,LODISP       ;LOAD DISPLAY REGISTER
031564  013767  005172  000576          MOV     @#DISPLY,ENDTAG ;AND ALSO AS LAST WORD XFERED
031572  012737  032054  000004          MOV     #ENDMEM,@#ERRVEC;SET TIME OUT TRAP VECTOR
031600  012702  040000          RETRY:  MOV     #40000,R2
031604  005000                          CLR     R0              ;DATA WILL BE RELOCATED FROM
                                                                ;ADDRESS IN R0 TO ADDRESS IN R2
031606  012704  032372                  MOV     #ENDTAG+2,R4    ;GET # OF BYTES TO RELOCATE
031612  010203                          MOV     R2,R3           ;GET 'TO' ADDRESS
031614  060403                          ADD     R4,R3           ;FORM LAST 'TO' ADDRESS FOR RELOCATION
031616  010013                          MOV     R0,(R3)         ;CHECK IF SUFFICIENT MEMORY AVAILABLE
031620  012737  005274  000004          MOV     #ERRRT,@#ERRVEC ;RESTORE ERROR TRAP VECTOR
031626  032737  000040  177570          BIT     #40,@#SWR       ;CHECK IF ALL DEVICES DESIRED FOR
031634  001007                          BNE     11$             ;RELOCATION ROUND ROBIN STYLE
031636  032737  000010  177570          BIT     #10,@#SWR       ;CHECK IF A DEVICE IS SPECIFIED
031644  001431                          BEQ     1$
031646  113737  177570  000757          MOVB    @#SWR,@#DEV     ;GET SELECTED DEVICE
031654  005046                  11$:    CLR     -(SP)           ;CLEAR WORKING LOCATION
031656  013702  172344                  MOV     @#KIPAR2,R2     ;FORM ADDRESS FOR READ DATA
031662  006302                          ASL     R2              ;SHIFT KIAPR BITS TO FORM
031664  006302                          ASL     R2              ;18 BIT PHYSICAL ADDRESS
031666  006302                          ASL     R2              ;IN R2 AND TOP OF STACK
031670  006302                          ASL     R2
031672  006302                          ASL     R2
031674  006116                          ROL     (SP)
031676  006302                          ASL     R2
031700  006116                          ROL     (SP)
031702  006316                          ASL     (SP)            ;POSITION EA BITS AT
031704  006316                          ASL     (SP)            ;BIT POSITION
031706  006316                          ASL     (SP)            ;4 AND 5
031710  006316                          ASL     (SP)
031712  112637  000762                  MOVB    (SP)+,@#EABITS  ;AND SAVE IN EABITS
031716  004737  001420                  JSR     PC,@#IODEV      ;GO RELOCATE DATA VIA I/O DEVICE
031722  102005                          BVC     10$             ;BRANCH IF NO ERRORS
031724  012702  040000                  MOV     #40000,R2       ;RESTORE 'TO' ADDRESS
031730  012022                  1$:     MOV     (R0)+,(R2)+     ;RELOCATE PROGRAM CODE TO ADDRESS SPEC-
                                                                ;IN R2/KIPAR2
031732  020302                          CMP     R3,R2           ;CHECK IF AT LAST ADDRESS
031734  001375                          BNE     1$
031736  010302                  10$:    MOV     R3,R2
031740  012703  001000                  MOV     #1000,R3        ;DO NOT CHECK FIRST 1000 (8) LOCATIONS
031744  024042                  2$:     CMP     -(R0),-(R2)     ;CHECK THAT DATA WAS RELOCATED PROPERLY
031746  001403                          BEQ     3$
031750  004737  002502                  JSR     PC,@#SAVVAL     ;GO SAVE APPROPRIATE VALUES
031754  104400                          HLT                     ;ERROR! DATA NOT RELOCATED PROPERLY
                                                                ;R0= SOURCE/R2=DEST ADDRESS
031756  020003                  3$:     CMP     R0,R3           ;BRANCH IF NOT AT LAST ADDRESS
031760  001371                          BNE     2$
031762  162737  000010  000772          SUB     #10,@#DEVID     ;BRANCH IF ERROR ON RELOCATION
```

```
                        BEQ     RETRY
                        INCB    @#DEV           ;STEP TO NEXT DEVICE
                        CLR     @#DEVID         ;SET DEVICE IND = CP
                        ADD     #40,(PC)+       ;STEP NEXT VALUE FOR KIPAR1
NEXPAR1: .WORD  0               ;CONTAINS NEXT VALUE FOR KIPAR1
                        MOV     @#K,PAR2,@#KIPAR0
                        MOV     @#KIPAR3,@#KIPAR1
;*****************************************************************************
;*****************************************************************************
;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)
                        MOV     #KPTR,SP        ;SET KERENL STACK PTR
                        CLR     @#PSW
                        MOV     OLDPSW,-(SP)    ;RESTORE OLD PSW
                        MOV     #1$,-(SP)
                        RTI
;*****************************************************************************
;*****************************************************************************
;PROGRAM NOW EXECUTING IN MODE AS DETERMEINED BY PASS COUNT (SEE PASTAB
;BELOW), ADDRESS IS DETERMINED BY MODE'S PAR0 REGISTER,
1$:     NOP                     ;DON'T REPLACE WITH HALT IF USER/SUPER MODE
                        JMP     @#START2        ;RESTART PROGRAM AT START2
```



```
;WHEN RELOCATION ABOVE 28K IS COMPLETE PROGRAM TRAPS TO ENDMEM,
ENDMEM: CMP     (SP)+,(SP)+     ;POP STACK TWICE
ENDM:   CLR     @#SR0           ;DISABLE MEM MGMT
                        CMPB    #4,@#OPT,CP     ;BRANCH IF 11/40
                        BEQ     1$
                        CLR     @#SR3
1$:     NOP
;*****************************************************************************
;*****************************************************************************
;PROGRAM NOW EXECUTING IN KERNEL MODE AT PC AS SHOWN (NO RELOCATION)
                        MOV     #1600,NEXPAR    ;RESET NEXT VALUE FOR PAR REGISTERS
                        CLRB    @#MMON          ;SET MEM MGMT ON IND = OFF
                        JMP     @#END1          ;RETURN TO INST FOLLOWING JSR PC,LDMM

DONE:   BIT     #100,@#TPS      ;WAIT FOR TTY OUTPUT TO FINISH
                        BNE     DONE
                        CLRB    @#TPB           ;TYPE NULL CHARACTER
                        TSTB    @#TPS           ;WAIT UNTIL DONE
                        BPL     .-4
                        CLR     R0
1$:     SUB     #1,R0
                        BNE     1$
                        RESET
                        TSTB    @#SWR           ;BRANCH IF NOT TYPEOUT DESIRED
                        BPL     2$
                        TYPE,ENDMSG
2$:     MOV     @#42,R2         ;CHECK DDP/ACT11 MONITOR HOOK
                        BEQ     DONE1
LOGICAL:JSR     PC,(R2)         ;GO TO DDP/ACT11 MONITOR VIA 42
                        NOP
                        NOP
                        NOP
DONE1:  JMP     @#START3        ;RESTART PROGRAM

;THE BELOW TABLE REPRESENTS THE 'NEW' PSW SET BY THE PROGRAM ON
;SUCCESSIVE 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,
PSWTAB: 000000                  ;ALL 11 FAMILY CP'S
                        000020
                        140000                  ;11/45, 11/40 ONLY
                        140020
                        144000                  ;11/45 ONLY
                        144020
                        044000
                        044020

;THE BELOW TABLE IS THE 'BIT MASK' USED TO DETERMINE THE INDEX VALUE
;NEEDED TO SET THE 'NEW' PSW,
CPPASS: 177774
```

```
 6028  032230  177774                          177774
 6029  032232  177770                          177770                  ;11/40
 6030  032234  177760                          177760                  ;11/45
 6031  032236  177760                          177760
 6032
```



```
 6033                                         ;THE BELOW TABLE REPRESENTS THOSE BITS IN THE CP WHICH CAN BE SET/CLEARED
 6034                                         ;EXCLUDING THE REGISTER SET BIT IN THE 11/45,
 6035  032240  000377                         PSWBIT: 000377
 6036  032242  000377                                 000377
 6037  032244  000357                                 000357                  ;11/40
 6038  032246  170357                                 170357                  ;11/45 (REGSET BIT IS CHECKED ELSEWHERE)
 6039  032250  170357                                 170357
 6040
 6041                                         ;THE BELOW TABLE CONTAINS THE # OF PASSES REQUIRED TO COMPLETE TEST
 6042  032252  000002                         PASTAB: .WORD   2
 6043  032254  000002                                 .WORD   2
 6044  032256  000004                                 .WORD   4               ;11/40
 6045  032260  000010                                 .WORD   10              ;11/45
 6046  032262  000010                                 .WORD   10
 6047
 6048                                         ;MESSAGES
 6049  032264  050015  050117  027124         AOPT.CP: .ASCIZ  <15><12>'OPT,CP= '
 6050  032272  050103  020075     000
 6051  032277     015  046012  053517         MSG1:   .ASCIZ  <15><12>'LOW LIMIT?'
 6052  032304  046040  046511  052111
 6053  032312  000077
 6054  032314  044510  044107  046040         MSG2:   .ASCIZ  'HIGH LIMIT?'
 6055  032322  046511  052111  000077
 6056  032330  005015  042040  050503         ENDMSG: .ASCIZ  <15><12>' DCQKC DONE'
 6057  032336  041513  042040  047117
 6058  032344  000105
 6059                                                 .EVEN
 6060  032346  005015                         NODEV:  .ASCII  <15><12>
 6061  032350  054130  047040  052117         DEVNAM: .ASCIZ  'XX NOT ON BUS'<15><12>
 6062  032356  047440  020116  052502
 6063  032364  006523  000012
 6064
 6065                                                 .EVEN
 6066  032370  000000                         ENDTAG: .WORD   0
 6067                                         ;NOTE: THE FOLLOWING CODE GETS OVERLAID WHEN THE PROGRAM IS STARTED,
 6068                                         ;THESE ROUTINES ARE USED TO CHECK THE TYPE,HLT,AND SCOPE ROUTINES,
 6069                                         ;CHECK TYPE ROUTINE:
 6070  032372  012706  000500                 CHKTYP: MOV     #500,SP         ;SET STACK PTR
 6071  032376  000004  003642                         TYPE,SUCCESS
 6072  032402  00C773                                 BR      CHKTYP
 6073
 6074  032404  012706  000500                 CHKHLT: MOV     #500,SP         ;SET STACK PTR
 6075  032410  104400                                 HLT
 6076  032412  000774                                 BR      CHKHLT
 6077
 6078  032414  012706  000500                 CHKSCP: MOV     #500,SP                 ;SET STACK PTR
 6079  032420  005000                                 CLR     R0
 6080  032422  010701                                 MOV     PC,R1                   ;SET SCOPE PTR
 6081  032424  010037  177570                         MOV     R0,@#DISPLAY
 6082  032430  005200                                 INC     R0
 6083  032432  104000                                 SCOPE
 6084  032434  000767                                 BR      CHKSCP
 6085
 6086
```

```
 6087                 000001                 .END
```



```
AC0     =%000000        AC1     =%000001        AC2     =%000002        AC3     =%000003
AC4     =%000004        AC5     =%000005        ADCB2   011574          ADC05   012404
ADCB6   013072          ADCB7   013740          ADC0    007534          ADC1    016410
ADC2    011404          ADC5    012212          ADC6    012302          ADC7    013634
ADD0    014426          ADD1    014524          ADD1A   014750          ADD1B   014766
ADD2    015354          ADD3    016122          ADD6    016464          ADD7    017126
ADRSIS  004744          ADRTAB  002226          AOPT,C  032264          ARBEX   030470
ARBFIN  030440          ASCAN   005031          ASHCL0  025320          ASHCR0  025176
ASHL0   024710          ASHL1   025502          ASHR0   025024          ASHR1   025570
ASLB1   010752          ASLB1A  011176          ASLB3   012374          ASLB4   011700
ASLB6   013054          ASLB7   014036          ASL0    007656          ASL1    010564
ASL3    012126          ASL4    011476          ASL6    012652          ASL7    013462
ASRB1   011046          ASRB1A  011062          ASRB2   011644          ASRB2A  011662
ASRB5   012334          ASRB6   013172          ASRB7   014054          ASR0    007704
ASR1    010452          ASR2    011420          ASR3    012112          ASR6    012534
ASR7    013516          A,DATA  004136          BDADR   004144          BDDAT   005110
BELL    003640          BICB1   015142          BICB1A  015164          BIC0    014340
BIC1    014646          BIC2    015444          BIC3    016134          BIC7    017730
BINB    015664          BINB7   017476          BIN1    015322          BISB1   015130
BIS0    014316          BIS0A   014374          BIS1    014634          BIS2    015402
BIS2A   015504          BIS7    017670          BITB1   015120          BITB2   015752
BITB3   016304          BITB6   016670          BIT1    014562          BIT13   = 020000
BIT14   = 040000        BIT15   = 100000        BIT2    015470          BIT6    = 000100
BIT8    = 000400        BPTVEC= 000014          BUSADR  001364          C       = 000001
CBIT    022716          CC0     007350          CC1     007364          CC2     007400
CC3     007412          CC4     007426          CHKHLT  032404          CHKSCP  032414
CHKSP   022516          CHKTYP  032372          CLRTBI  002532          CLR0    007452
CMPB1   015074          CMPB2   015736          CMPB3   016316          CMPN    022752
CMP0    014220          CMP0A   014462          CMP1    014544          CMP1A   014664
CMP2    015372          CMP7    017062          CNTRLC= 000003          CNVADR  002716
CNVDAT  002672          CNVDIG  003016          COMB1   011030          COMB1A  011210
COMB2   011556          COMB5   012312          COMB6   013124          COMB7   013754
COM0    007516          COM1    010576          COM3    012170          COM4    011306
COM6    012520          COM7    013606          CPCHK   006000          CPERR   003614
CPPASS  032226          CPUERR= 177766          CRLF    000752          CYLADR  002214
DBINB7  017472          DBIN7   016762          DDATA   016326          DDATAB  016756
DECB1   011000          DECB1A  011130          DECB2   011712          DECB5   012452
DECB6A  013224          DECB7   014022          DEC0    007576          DEC1    010370
DEC1A   010640          DEC2    011454          DEC5    012144          DEC6    012666
DEC7    013500          DEV     000757          DEVERR  000742          DEVICE  004156
DEVID   000772          DEVNAM  032350          DEVTBL  002266          DIGBUF  003072
DIGITS  003074          DIGTAB  003554          DISPLA= 177570          DISPLY  005172
DIV0    025402          DONE    032116          DONE1   032202          DSCAN   005046
DSKADR  002124          EABITS  000762          ECHO    000740          EISOPT= 040000
EMTVEC= 000030          EMT1    021174          EMT1B   021254          EMT1C   021260
EMT1D   021272          END     030750          ENDCP   026554          ENDM    032056
ENDMEM  032054          ENDMSG  032330          ENDTAG  032370          END1    030754
ERFLAG  005322          ERMSG   005340          ERPRT   005274          ERREG   003621
ERRPRT  005320          ERRREG= 177744          ERRVEC= 000004          ERTAG   005326
EXTINS  023160          FACTOR  001004          FISOPT= 010000          FPEVEC= 000244
FPOPT   = 020000        FRSTAD  001010          FRSTME  001012          GDADR   004066
GDDAT   005075          GSTST   010170          HALT1   026454          HLT     = 104400
ICNT    001000          INCB1   010714          INCB2   012006          INCB3   012344
INCB6   013236          INCB6A  013206          INCB7   014006          INC0    007616
```

```
INC1    010504          INC3    012200          INC4    011336          INC8    012630
INC7    013646          IODEV   001420          IORETR  000960          IOTTST  020732
IOTVEC= 000020          ITCNT   001150          JMP1    020212          JMP3    020262
JMP4    020320          JMP5    020364          JMP6    020404          JMP7    020442
JSRTST  020446          JSR1    020530          JSR1A   020532          JSR5    020616
JSR3A   020620          JSR4    020674          JSR4A   020676          JSR6A   020730
KDPAR0= 172360          KDPDR0= 172320          KIPAR0= 172340          KIPAR1= 172342
KIPAR2= 172344          KIPAR3= 172346          KIPAR4= 172350          KIPAR7= 172356
KIPDR0= 172300          KIPDR1= 172302          KIPDR2= 172304          KIPDR3= 172306
KIPDR4= 172310          KIPDR7= 172316          KJOPT = 004000          KM    = 000000
KPTR  = 000600          KSP   =X000006          KTABRT  005244          KTAMSG  005357
KTEX    027624          KTOPT = 100000          KTPAR   027216          KTPDR   027074
KT1     027352          KW11    030040          LDDISP  005166          LKOPT = 001000
LKS   = 177546          LKSRV   030350          LKVEC = 000100          LOGICA  032172
LPB   = 177516          LPS   = 177514          LSTMEM  005504          LTICKS  000774
MARKEX  024166          MARK1   024142          MEMTBL  000730          MMON    000770
MMVEC = 000250          MOVB1   015090          MOV0    014150          MOV0A   014200
MOV1    014734          MOV7    017032          MPI     026246          MRKTST  024104
MSG     030030          MSG1    032277          MSG2    032314          MUL0    025264
N     = 000010          NEGB1   010764          NEGB4   011736          NEGB6   013142
NEGB7   014070          NEG0    007636          NEG1    010630          NEG2    011360
NEG5    012156          NEG6    012554          NEG7    013554          NEXPAR  032006
NODEV   032346          NOTFND  004771          NULL  = 000000          NULLS   030034
OAERR   017756          OLDPSW  031262          OPTION  000766          OPT.CP  000764
OVFLW   031536          PARCSR= 172100          PARERR  000707          PARTBL  027334
PARVEC= 000114          PASCNT  003564          PASSES  003574          PASSNO  003750
PASTAB  032252          PC    =X000007          PDRTBL  027316          PDWN    000610
PEFLG   000761          PERET   004402          PFAIL   000466          PFVEC = 000024
PHYSPC  003633          PIRQ  = 177772          PIRQ0   026032          PIRVEC= 000240
PIR4  = 010000          PKM   = 000000          PLKCSB= 172542          PLKCSR= 172540
PLKCTR= 172544          PLKOPT= 002000          PLKSRV  030400          PLKVEC= 000104
PNTREG  003756          PPB   = 177556          PPOPT = 000002          PPS   = 177554
PPVEC = 000074          PRB   = 177552          PRDAT   000770          PREXIT  030750
PROPT = 000001          PRS   = 177550          PRSTRT  030572          PRSYNC  000771
PRTST   030646          PRTY2 = 000100          PRTY3 = 000140          PRTY4 = 000200
PRTY5 = 000240          PRTY6 = 000300          PRTY7 = 000340          PRVEC = 000070
PSM   = 010000          PSW   = 177776          PSWBIT  032240          PSWCHK  022304
PSWTAB  032206          PTICKS  000776          PUN   = 030000          PUP     000632
QV      000771          RCCA  = 177452          RCCS  = 177446          RCDA  = 177442
RCLA  = 177440          RCTBL   002374          RCVEC = 000210          RCWC  = 177450
RECO    004244          REG   = 004000          REGADR  004422          REGS    004200
RELOC   001152          RELPC   003626          RELR1   001006          REL0    006262
REL00   007260          REL1    007262          REL11   013310          REL2    013312
REL22   017306          REL3    017310          REL33   021444          REL4    021472
REL44   023112          REL5    023114          REL55   024626          REL6    024630
REL66   026570          REL7    026572          REL77   027636          RESERR  005264
RESET1  026526          RESMSG  005374          RESPSW  002546          RESTPS  002556
RESTRP  022132          RESVEC= 000010          RETPC   001360          RETPSW  002540
RETRY   031600          RFCMA = 177464          RFDAE = 177470          RFDAR = 177466
RFDCS = 177460          RFTBL   002330          RFVEC = 000204          RFWC  = 177462
RHCS2 = 000000          RKBA  = 177410          RKCS  = 177404          RKDA  = 177412
RKDS  = 177400          RKER  = 177402          RKTBL   002306          RKVEC = 000220
RKWC  = 177406          ROLB1   010726          ROLB2   011630          ROLB3   012420
ROLB6   013156          ROLB6A  013252          ROLB7   014106          ROL0    007670
```



```
ROL1    010424          ROL1A   010436          ROL3    012226          ROL4    011434
ROL6    012504          ROL7    013664          RORB1   011014          RORB1A  011104
RORB4   011612          RORB5   012362          RORB6   013010          RORB7   013772
ROR0    007554          ROR1    010354          ROR1A   010466          ROR2    011322
ROR5    012100          ROR6    012574          ROR7    013336          RPBA  = 176720
RPCA  = 176722          RPCS  = 176714          RPDA  = 176724          RPDS  = 176710
RPER  = 176712          RPTBL   002352          RPVEC = 000254          RPWC  = 176716
RP4AS = 176716          RP4BA = 176704          RP4CA = 176734          RP4CS1= 176700
RP4DST= 176706          RP4DS1= 176712          RP4ER1= 176714          RP4LA = 176720
RP4OF = 176732          RP4TBL  002416          RP4VEC= 000254          RP4WC = 176702
RSAS  = 172056          RSBA  = 172044          RSCS1 = 172040          RSCS2 = 172050
RSDA  = 172046          RSDS  = 172052          RSER  = 172054          RSLA  = 172060
RSTBL   002450          RSVEC = 000204          RSWC  = 172042          RTI1    031204
RTT1    024172          RTT1EX  024362          RTT2    024470          RTT2A   024404
RTT2EX  024604          R0    =X000000          R1    =X000001          R10   =X000000
R11   =X000001          R12   =X000002          R13   =X000003          R14   =X000004
R15   =X000005          R2    =X000002          R3    =X000003          R4    =X000004
R5    =X000005          SAVPSW  024366          SAVVAL  002502          SBCB1   010740
SBCB3   012440          SBCB4   011724          SBCB6   013012          SBCB7   013720
SBC0    007720          SBC1    010520          SBC1A   010536          SBC5    012240
SBC6    012610          SBC7    013442          SBIN87  017462          SBIN7   016760
SCOPE = 104000          SCOPEA  001014          SDATA   016324          SDATAB  016754
SDPAR0= 172260          SDPDR0= 172220          SECT    005176          SIPAR0= 172240
SIPAR1= 172242          SIPAR4= 172250          SIPAR6= 172254          SIPAR7= 172256
SIPDR0= 172200          SIPDR1= 172202          SIPDR4= 172210          SIPDR6= 172214
SIPDR7= 172216          SLASH   000755          SLR   = 177774          SM    = 040000
SOB0    023704          SOB1    023712          SOB10   023666          SOB2    023740
SOB3    023752          SOB4    024004          SOB5    024006          SOB5A   024026
SOB6    024052          SOB7    024060          SOB8    024062          SOB9    023676
SP    =X000006          SPCHK   022546          SPL0    025472          SR0   = 177572
SR1   = 177574          SR2   = 177576          SR3   = 172516          SSP   =X000006
START   005422          START1  005532          START2  005644          START3  005600
STATUS  003607          STKLIM  026636          STKPTR= 000500          STMM    031206
SUB0    014166          SUB1    014602          SUB1A   014710          SUB1B   014724
SUB2    015420          SUB2A   015540          SUB3    016046          SUB3A   016070
SUB6    016504          SUB7    017102          SUCCES  003642          SWAB0   007736
SWAB1   011144          SWAB2   011350          SWAB4   011770          SWAB6   013236
SWAB7   013570          SWR   = 177570          SXRA    023534          SXRB    023536
SXT0    023202          SXT1    023266          SXT2    023552          SXT3    023566
SXT4    023334          SXT5    023576          SXT6    023476          SXT6A   023526
SXT7    023642          T     = 000020          TBITVE= 000014          TKB   = 177562
TKISR   003130          TKS   = 177560          TKVEC = 000060          TPB   = 177566
TPS   = 177564          TPVEC = 000064          TRAPVE= 000034          TRAP1   021362
TRAP1C  021406          TRKSEC  002222          TRTVEC= 000014          TSTB1   011156
TSTB2   011746          TSTB2A  011756          TSTB6   012764          TST0    007474
TST1    010614          TST2    011276          TST6    013266          TTOPT = 000400
TTYCHK  027640          TYPADR  003116          TYPDAT  003104          TYPE  = 000004
UBM6    013272          UBREAK= 177770          UB7     016970          UDPAR0= 177660
UDPDR0= 177620          UIPAR0= 177640          UIPAR1= 177642          UIPAR4= 177650
UIPAR6= 177654          UIPAR7= 177656          UIPDR0= 177600          UIPDR1= 177602
UIPDR4= 177610          UIPDR6= 177614          UIPDR7= 177616          UM    = 140000
USP   =X000006          UWM6    012460          UWM7    013362          UW7     013366
V     = 000002          VIRPC   003601          WAITID  001362          XOR0    023234
XOR1    023322          XOR24   023356          XOR35   023426          XOR6    023444
```

DCQKCD SYMBOL TABL

```
XOR6A   023450          XOR6B   023452          XOR7    023636          Z       = 000004
$FILLS  001002          $RESTR  005144          $SAVR   005124          .HLT    003212
.MAMF   000120          .PARSR  004356          .TPVEC  030436          .TYPE   002564
.       = 032436
```

ERRORS DETECTED: 0/5

*C,C/SOL=DCQKCD
RUN-TIME: 23 41 0 SECONDS
CORE USED: 11K