KD11-K

11/6X SERIES CPU EXER

MD-11-DQKDC-A

EP-DQKDC-A-DL-A

APR 1977

digital

MADE IN USA

KD11-K

11/6X SERIES CPU EXER

MD-11-DQKDC-A

EP-DQKDC-A-DL-A

APR 1977

digital

FICHE 2 OF 2

MADE IN USA

```
                        .REM    @

                                        IDENTIFICATION
                                        --------------

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

                            PRODUCT  NAME:    11/6X SERIES CPU EXERCISER

                            DATE CREATED:     JANUARY 24, 1977

                            MAINTAINER:       DIAGNOSTIC ENGINEERING

                            AUTHOR:           DIAGNOSTIC ENGINEERING
```

CONTENTS

1.0 ABSTRACT

2.0 REQUIREMENTS
- 2.1 EQUIPMENT
- 2.2 STORAGE
- 2.3 PRELIMINARY PROGRAMS

3.0 LOADING PROCEDURE
- 3.1 METHOD

4.0 STARTING PROCEDURE
- 4.1 CONTROL SWITCH SETTINGS
- 4.2 STARTING ADDRESSES
- 4.3 PROGRAM AND OPERATOR ACTION

5.0 OPERATING PROCEDURE
- 5.1 OPERATIONAL SWITCH SETTINGS
- 5.2 DISPLAY REGISTER
- 5.3 OPERATOR ACTION

6.0 ERRORS
- 6.1 ERROR HALTS AND DESCRIPTION
- 6.2 ERROR RECOVERY

7.0 WARNINGS AND EXCEPTIONS
- 7.1 WARNINGS
- 7.2 EXCEPTIONS

8.0 MISCELLANEOUS
- 8.1 EXECUTION TIME
- 8.2 STACK POINTER
- 8.3 PASS COUNT
- 8.4 END OF PASS MESSAGES
- 8.5 ITERATIONS
- 8.6 T BIT TRAPPING
- 8.7 ACT11 COMPATABILITY
- 8.8 PSW TABLE
- 8.9 I/O DEVICE ADDRESS MODIFCATIONS
- 8.10 POWER FAILURE

9.0 PROGRAM DESCRIPTION
- 9.1 UNIBUS EXERCISER FUNCTION
- 9.2 MASS BUS TESTER FUNCTION
- 9.3 LINE CLOCK INITIALIZATION
- 9.4 RELOCATION ALGORITHM

1.0 ABSTRACT

THIS PROGRAM IS DESIGNED TO BE A COMPREHENSIVE CHECK OF THE PDP-11/6X SERIES CPU CLUSTER. THE PROGRAM EXECUTES EACH INSTRUCTION IN ALL ADDRESS MODES AND INCLUDES TESTS FOR TRAPS, INTERRUPTS, FLOATING POINT, MEMORY MANAGEMENT, MEMORY, THE UNIBUS, AND THE MASS BUS. IF NOT DESELECTED, THE PROGRAM RELOCATES THE TEST CODE THROUGHOUT MEMORY (0-124K). ALSO, IF NOT DESELECTED, THE PROGRAM WILL RELOCATE USING AVAILABLE DISKS (RP03,RK05,RP04,RS03/4). SEE SECTION 9.4 FOR A DESCRIPTION OF RELOCATION.

SINCE WORST CASE TESTING OCCURS WITH ALL SWITCHES DOWN, PRECAUTIONS MUST BE TAKEN TO ENSURE THE PROTECTION OF USER DISKS. REFER TO SECTION 7.0 FOR A DESCRIPTION OF WARNINGS AND EXCEPTIONS.

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11/6X SERIES CPU WITH 16K OF MEMORY, A LINE CLOCK, AND AN LA30 (OR EQUIVALENT) CONSOLE TERMINAL.

2.1.1 OPTIONAL EQUIPMENT USED

1. UNIBUS EXERCISER
2. MASS BUS TESTER
3. RP11/RP03, RK11/RK05, RH11/RP04, RH11/RS03/RS04

2.2 STORAGE

THE PROGRAM LOADS INTO THE FIRST 12K OF MEMORY AND RUNS IN ALL MEMORY (EXCLUSIVE OF THE XXDP MONITOR IF RUNNING IN CHAIN MODE).

2.3 PRELIMINARY PROGRAMS

ALTHOUGH THIS PROGRAM IS A TEST OF THE CPU, IT IS ADVISABLE THAT THE CPU (AND FLOATING POINT) DIAGNOSTICS RUN FIRST. THESE CONSIST OF:

```
DQKDA   DQFPA
DQKDB   DQFPB
DQKKA   DQFPC
DQKTA   DQFPD
DQKUA
```

```
                        3.0     LOADING PROCEDURE
                                -----------------
```

3.1 METHOD

THE PROGRAM IS SUPPLIED ON THE DIAGNOSTIC MEDIA. REFER TO THE XXDP OPERATING MANUAL FOR FURTHER INFORMATION. THE PROGRAM CAN ALSO BE DIRECTLY LOADED USING THE ABSOLUTE LOADER AND THE BINARY PAPER TAPE.

IF LOADING A BINARY PAPER TAPE, BE SURE THE SWITCH REGISTER IS CLEARED AFTER THE ABSOLUTE LOADER IS LOADED.

4.0 STARTING PROCEDURE

4.1 CONSOLE SWITCH SETTINGS

CHECK THAT THE SWITCH REGISTER WAS ZERO IF THE PROGRAM WAS LOADED FROM PAPER TAPE.

SEE SECTION 5.1

4.2 STARTING ADDRESSES

THE STARTING ADDRESS FOR THE EXERCISER IS 200.

4.3 PROGRAM AND OPERATOR ACTION
----

1. LOAD PROGRAM INTO MEMORY (SEE SECTION 3)
2. CHECK FOR ANY SYSTEM DISK PACKS OR CONFIGURATION EXCEPTIONS AS DESCRIBED IN SECTION 7.0.
3. LOAD ADDRESS 200
4. SET SWITCHES (SEE SECTION 5.1)
5. PRESS START
6. THE PROGRAM WILL LOOP AND MESSAGES WILL BE TYPED AT THE END OF EACH SUB-PASS AND EACH PASS. (SEE SECTION 8.4 FOR A DESCRIPTION OF THE MESSAGES)

5.0 OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

SW15 (100000) HALT ON ERROR	THIS SWITCH WHEN SET WILL HALT THE PROCESSOR WHEN AN ERROR IS DETECTED. AN ERROR MESSAGE IS TYPED AND THE PROCESSOR WILL HALT. PRESSING CONTINUE WILL RESUME TESTING.
SW14 (040000) LOOP ON TEST	THIS SWITCH WHEN SET WILL CAUSE THE PROGRAM TO LOOP ON THE CURRENT SUBTEST.
SW13 (020000) INHIBIT ERROR TYPEOUT	THIS SWITCH WHEN SET INHIBITS THE ERROR TYPEOUT.
SW12 (010000) INHIBIT UBE	THIS SWITCH WHEN SET INHIBITS THE INITIALIZATION OF THE UNIBUS EXERCISER. SEE SECTION 9.1 FOR A DESCRIPTION OF THE UBE FUNCTION.
SW11 (004000) INHIBIT SUB-TEST ITERATION	THIS SWITCH WHEN SET INHIBITS SUBTEST ITERATION AFTER THE FIRST PASS. EACH SUBTEST IS EXECUTED 10 TIMES BEFORE THE NEXT SUBTEST IS RUN. SETTING SW11 CAUSES EACH TEST TO BE EXECUTED ONCE BEFORE STARTING THE NEXT SUBTEST.
SW10 (002000) RING BELL ON ERROR	THIS SWITCH WHEN SET WILL RING THE BELL WHEN AN ERROR IS DETECTED.
SW9 (001000) LOOP ON ERROR	THIS SWITCH WHEN SET WILL CAUSE THE PROGRAM TO LOOP ON THE FIRST FAILURE EVEN IF THE FAILURE IS INTERMITTANT. SEE SECTION 6.1 FOR A DESCRIPTION OF LOOPING ON RELOCATION ERRORS.
SW8 (000400) RELOCATE WITH CPU ONLY	THIS SWITCH WHEN SET WILL CAUSE RELOCATION TO BE DONE BY THE CPU INSTEAD OF A DISK. SEE SECTION 9.4 FOR A DESCRIPTION OF RELOCATION.

```
                                                                                         PAGE 6

                                 SW7 (000200) INHIBIT SYSTEM      THIS SWITCH WHEN SET  WILL IN-
                                               SIZE TYPEOUT       INHIBIT  THE  TYPEOUT  OF  THE
                                                                  SWITCH   DEFINITIONS  AND  THE
                                                                  DISKS THAT WILL  BE  USED  FOR
                                                                  RELOCATION.    (TYPEOUT   ONLY
                                                                  OCCURS  WHEN  THE  PROGRAM  IS
                                                                  DUMPED)

                                 SW6 (000100) INHIBIT RELOCATION THIS  SWITCH  WHEN  SET   WILL
                                                                  INHIBIT  ALL  RELOCATION.   DO
                                                                  NOT CHANGE THIS  SWITCH  WHILE
                                                                  THE PROGRAM IS RUNNING.

                                 SW5 (000040) INHIBIT ROUND       THIS SWITCH WHEN SET WILL ONLY
                                               ROBIN              RELOCATE  USING   THE   DEVICE
                                                                  SELECTED   BY  SWITCHES  <2:0>
                                                                  RATHER  THAN   ALL   AVAILABLE
                                                                  DEVICES.

                                 SW4 (000020) INHIBIT RANDOM      THIS   SWITCH  WHEN  SET  WILL
                                               DISK ADDRESS       CAUSE  RELOCATION  TO   ALWAYS
                                                                  START  AT  ADDRESS  0  ON  THE
                                                                  DISK(S).

                                 SW3 (000010) INHIBIT MBT         THIS SWITCH WHEN SET  INHIBITS
                                                                  THE INITIALIZATION OF THE MASS
                                                                  BUS TESTER.  SEE  SECTION  9.2
                                                                  FOR  A  DESCRIPTION OF THE MBT
                                                                  FUNCTION.

                                 SW2-SW0 (0 - 7) DEVICE CODES     THESE  SWITCHES  (ALONG   WITH
                                                                  SW5)   CAUSE  THE  PROGRAM  TO
                                                                  RELOCATE THE TEST  CODE  USING
                                                                  THE DEVICE SPECIFIED BELOW:

                                                                  VALUE          DEVICE

                                                                    0           RP11/RP03
                                                                    1           RK11/RK05
                                                                    2           NOT USED
                                                                    3           NOT USED
                                                                    4           RH11/RP04
                                                                    5           RH11/RS03/RS04
                                                                    6           NOT USED
                                                                    7           NOT USED

                                                   NOTE

                                         WHEN RELOCATING VIA A  SPECIFIC  DEVICE,
                                         SET  IN THE VALUE(SW<2:0>) TO SELECT THE
                                         DEVICE THEN SET SWITCH 5.
```

```
                                        UNIT 0 OF THE LOAD DEVICE IS MARKED  NOT
                                        PRESENT  IF  PROGRAM WAS LOADED IN CHAIN
```

MODE (XXDP), AND THEREFORE WILL NOT BE USED TO RELOCATE.

5.2 DISPLAY REGISTER

WHILE THE PROGRAM IS RUNNING, THE LOW BYTE OF THE DISPLAY REGISTER CONTAINS THE SUBTEST NUMBER AND THE HIGH BYTE CONTAINS BITS <11:4> OF KERNEL PAR0. THESE BITS, OF KERNEL PAR0, CORRESPOND TO BITS <17:10> OF THE PHYSICAL ADDRESS OF THE RELOCATED CODE. WHEN AN ERROR IS DETECTED AND LOOP ON ERROR IS SELECTED, THE HIGH BYTE CONTAINS THE ERROR COUNT.

5.3 OPERATOR ACTION

WHEN THE PROGRAM IS LOADED* AND STARTED WITH SWITCH 7 EQUAL TO ZERO THE PROGRAM WILL TYPEOUT THE DISKS AND UNIT NUMBERS THAT WILL BE USED FOR RELOCATION AND THEN WAIT FOR THE OPERATOR TO TYPE A CHARACTER. THIS IS TO ALLOW THE OPERATOR TO WRITE PROTECT ANY DRIVE THAT IS NOT TO BE USED. IF THERE ARE NO DEVICES AVAILABLE FOR RELOCATION, OPERATOR ACTION IS NOT REQUIRED.

IF THE PROGRAM IS LOADED VIA ACT11 IN QV OR AA OR WITH XXDP IN CHAIN MODE NO OPERATOR ACTION IS REQUIRED AND ALL DISKS NOT WRITE PROTECTED (EXCEPT FOR THE XXDP MEDIA) WILL BE USED FOR RELOCATION.

*EXCEPT CHAIN MODE, QV(MANUFACTURING ONLY), OR AUTO ACCEPT (MANUFACTURING ONLY)

6.0 ERRORS

6.1 ERROR HALTS AND DESCRIPTION

IF AN ERROR IS DETECTED, THE PROGRAM WILL TRAP TO THE ERROR HANDLING ROUTINE ($ERROR). IF HALT ON ERROR IS ENABLED, THE PROCESSOR WILL TYPE THE ERROR MESSAGE AND THEN HALT. PRESSING CONTINUE WILL CAUSE TESTING TO RESUME.

THERE ARE MANY DIFFERENT TYPES OF ERRORS. NO MATTER WHICH TYPE OCCURS A MINIMUM SET OF INFORMATION IS TYPED AS FOLLOWS:

```
HHH:MM:SS
ERRORPC PHYSC PC  PSW     TEST NO SUBPAS-PASS CNT
UUUUUU  VVVVVV  WWWWWW    YYYYYY  SSSSSS  PPPPPP
```

WHERE:

UUUUUU = VIRTUAL PC OF THE ERROR CALL.

```
                                          VVVVVV      = PHYSICAL PC OF THE ERROR CALL.
                                          WWWWWW      = PSW AT THE TIME OF THE ERROR CALL.
```

```
YYYYYY      = TEST NUMBER.
SSSSSS      = SUB-PASS COUNT (0 THRU 3)
PPPPPP      = PASS COUNT
```

HHH:MM:SS REPRESENTS THE ELAPSED RUN TIME OF THE PROGRAM SINCE THE MOST RECENT START, WHERE: HHH = HOURS, MM = MINUTES, AND SS = SECONDS.

THE VIRTUAL PC IS THE 16 BIT WORD THAT WAS PUSHED ON THE STACK WHEN THE ERROR CALL WAS MADE. THE PHYSICAL PC IS CALCULATED IN ONE OF TWO WAYS:

1. IF MEMORY MANAGEMENT IS OFF THE CONTENTS OF LOCATION "FACTOR" IS SUBTRACTED FROM THE VIRTUAL PC. THIS GENERATES THE CORRESPONDING PC FOR THE NON-RELOCATED CODE.

2. IF MEMORY MANAGEMENT IS ON THE CONTENTS OF THE APPROPRIATE PAR IS SHIFTED AND ADDED TO THE VIRTUAL PC TO GENERATE A PHYSICAL 18 BIT ADDRESS. IN THIS CASE THE VIRTUAL PC CORRESPONDS TO THE NON-RELOCATED CODE.

DEPENDING ON THE TYPE OF ERROR ADDITIONAL INFORMATION IS TYPED AS DESCRIBED BELOW.

6.1.1 UNEXPECTED TRAP TO 4

```
VIRTPC   PHYSPC    PSW      CPUERR
VVVVVV  PPPPPP    YYYYYY    ZZZZZZ
```

```
VVVVVV      = VIRTUAL PC THAT WAS PUSHED ON THE STACK WHEN THE
              TRAP OCCURRED.
PPPPPP      = PHYSICAL PC CALCULATED AS DESCRIBED ABOVE.
YYYYYY      = PSW THAT WAS PUSHED ON THE STACK.
ZZZZZZ      = CONTENTS OF THE CPU ERROR REGISTER(777766).
```

6.1.2 UNEXPECTED TRAP TO 114

```
VIRTPC   PHYSPC       PSW     MEM ERR REG
VVVVVV   PPPPPP      YYYYYY    ZZZZZZ
```

```
V, P, AND Y = ARE THE SAME AS DESCRIBED IN 6.1.1.
ZZZZZZ      = CONTENTS OF THE MEMORY ERROR REGISTER (777744).
```

6.1.3 PARITY ERROR DURING DATA CHECK

THIS ERROR CAN ONLY OCCUR DURING THE DATA CHECK THAT IS MADE ON THE RELOCATED TEST CODE BEFORE IT IS EXECUTED. THIS CHECK IS MADE BY COMPARING THE UNRELOCATED CODE WITH THE RELOCATED CODE. THE SOURCE ADDRESS REFERS TO THE UNRELOCATED CODE AND THE DESTINATION ADDRESS TO THE RELOCATED CODE.

```
SRCADR   DSTADR  MEM ERR REG
SSSSSS   DDDDDD  ZZZZZZ
```

```
                                        SSSSSS      = VIRTUAL ADDRESS OF THE SOURCE DATA.
```

```
                    DDDDDD     = PHYSICAL ADDRESS OF THE DESTINATION DATA.
                    ZZZZZZ     = CONTENTS OF MEMORY ERROR REGISTER (777744).

            6.1.4   ERROR DURING DATA CHECK-RELOC WAS BY CP

                    THIS ERROR IS SIMILAR TO 6.1.3 EXCEPT  INSTEAD  OF  A  PARITY
                    ERROR,  IT IS A DATA COMPARISON ERROR.  REFER TO SECTION 9.4.3
                    FOR A DESCRIPTION OF CP RELOCATION.

                    LOOP ON ERROR (SW<9>) HAS THE FOLLOWING EFFECT:
                         1.  MEMORY MANAGEMENT OFF- IF SWITCH<9>  IS  SET,  LOOPING
                             WILL  BE  PERFORMED  ON  THE  SECTION  RELOCATION (SEE
                             SECTION 9.4.1).  IF SW<9> IS NOT SET,  EXECUTION  WILL
                             CONTINUE AT THE BEGINNING OF THE NEXT SECTION.

                         2.  MEMORY MANAGEMENT ON- IF SW<9> IS SET, LOOPING WILL BE
                             PERFORMED  ON  THE  PROGRAM  RELOCATION  (SEE  SECTION
                             9.4.2) TO THE SAME MEMORY SPACE THAT FAILED.  IF SW<9>
                             IS  NOT SET, PROGRAM RELOCATION WILL BE RETRIED IN THE
                             SAME MEMORY SPACE.

            6.1.5   ERROR DURING DATA CHECK-RELOC WAS BY I/O

                    THIS  ERROR  IS  THE  SAME  AS  6.1.4  EXCEPT  RELOCATION  WAS
                    PERFORMED  VIA  A DISK RATHER THAN THE CP.  THE ERROR PRINTOUT
                    WILL IDENTIFY WHICH DEVICE AND DRIVE  NUMBER  TRANSFERRED  THE
                    PARTICULAR  WORD  THAT  FAILED.   REFER TO SECTION 9.4.4 FOR A
                    DESCRIPTION OF I/O RELOCATION.

                    LOOP ON ERROR (SW<9>) HAS THE FOLLOWING EFFECT:
                         1.  IF SW<9> IS SET, THE DEVICE THAT  RELOCATED  THE  WORD
                             (THAT  CAUSED THE DATA CHECK ERROR) IS INITIATED TO DO
                             THE SAME TRANSFER  WITH  THE  SAME  DISK  ADDRESS  AND
                             MEMORY  ADDRESSES.   THIS TRANSFER WILL CONTINUALLY BE
                             INITIATED AND CHECKED UNTIL SW<9> IS NOT SET.

            6.1.6   DEVICE ERROR

                    THIS ERROR OCCURS IF A DEVICE ERROR OCCURS WHILE THE DEVICE IS
                    DOING  A TRANSFER.  THE DEVICE AND DRIVE NUMBER ARE IDENTIFIED
                    AND THE CONTENTS OF THE DEVICE REGISTERS ARE TYPED.

                    WHEN SW<9> (LOOP ON ERROR) IS SET, THE DEVICE THAT  FAILED  IS
                    CONTINUALLY  RESTARTED  WITH  THE  SAME  DISK  ADDRESS, MEMORY
                    ADDRESS, AND FUNCTION THAT CAUSED THE ERROR.

                    IF SW<9> IS NOT SET, RELOCATION IS RESTARTED.

            6.1.7   UNIBUS EXERCISER FAILED

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

```
  483                                             XXXXXX      = CYCLE COUNT.
  484                                             VVVVVV      = VIRTUAL BUS ADDRESS THAT THE UBE FAILED AT
```

```
WWWWWW      = CONTROL REGISTER NUMBER 2
YYYYYY      = CONTROL REGISTER NUMBER 1
ZZZZZZ      = PHYSICAL MEMORY ADDRESS THAT THE UBE FAILED AT
```

THE PHYSICAL MEMORY ADDRESS IS CALCULATED BY ADDING THE APPROPRIATE MAP REGISTER TO THE VIRTUAL BUS ADDRESS, FORMING A REAL 18 BIT MEMORY ADDRESS.

6.1.8 UBE NON-EXISTANT MEMORY ERROR

THIS ERROR ONLY OCCURS WHEN THE "NO SLAVE SYNC" ERROR OCCURS IN THE UNIBUS EXERCISER. ONLY THE PHYSICAL ADDRESS THAT TIMED OUT IS TYPED. THIS ERROR MIGHT INDICATE THAT THERE IS A HOLE IN MEMORY.

6.1.9 MASS BUS TESTER FAILED

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

  ER       CS3
HHHHHH    JJJJJJ

AAAAAA      = CONTROL AND STATUS REGISTER #1 (760100).
BBBBBB      = WORD COUNT REGISTER (760102).
CCCCCC      = BUS ADDRESS REGISTER (760104).
DDDDDD      = BUS ADDRESS EXTENDED REGISTER (760174).
EEEEEE      = MAINTENANCE REGISTER #2 (760106).
FFFFFF      = CONTROL AND STATUS REGISTER #2 (760110).
GGGGGG      = STATUS REGISTER (760112).
HHHHHH      = ERROR REGISTER (760114).
JJJJJJ      = CONTROL AND STATUS REGISTER #3 (760176).
```

6.1.10 MBT NON-EXISTANT MEMORY ERROR

THIS IS THE SAME AS 6.1.8 EXCEPT THAT IT IS DETECTED BY THE NEXM BIT IN CS2 OF THE MBT.

6.1.11 FLOATING POINT ERROR

THIS ERROR WILL ONLY OCCUR IF THE LEFT AND RIGHT HAND SIDES OF THE FLOATING POINT IDENTITIES DO NOT AGREE WITHIN THE EXPECTED TOLERANCE. THE VALUE OF THE CALCULATIONS ARE TYPED OUT.

THIS ERROR SHOULD ONLY BE A FUNCTION OF THE FLOATING POINT PROCESSOR AND THE FPP DIAGNOSTICS (DQFPA-DQFPD) SHOULD BE USED TO ISOLATE THE PROBLEM.

6.1.12 DEVICE HUNG

THIS ERROR WILL OCCUR IF A DEVICE DOES NOT FINISH ITS RELOCATION FUNCTION WITHIN 2 SECONDS AFTER ITS INITIATION. REFER TO SECTION 9.4.4.4 TO DETERMINE WHICH DEVICE AND DRIVE

541 IS HUNG.

6.2 ERROR RECOVERY

DIFFERENT TYPES OF ERRORS RECOVER IN DIFFERENT WAYS AS DESCRIBED BELOW.

6.2.1 ERRORS WITHIN SUBTESTS

EXECUTION STARTS WITH THE INSTRUCTION FOLLOWING THE ERROR CALL.

6.2.2 RELOCATION WITH MEMORY MGMT. OFF

EXECUTION STARTS AT THE BEGINNING OF THE NEXT SECTION.

6.2.3 DEVICE ERROR OR CP RELOCATION WITH MEMORY MGMT. ON

RELOCATION IS RESTARTED AT THE BEGINNING OF WHATEVER TYPE OF RELOCATION THE ERROR WAS FOUND IN.

6.2.4 UNEXPECTED TRAPS (4,10,114,250)

EXECUTION STARTS AT THE ADDRESS POINTED TO BY LOCATION "$LPERR". THIS LOCATION CONTAINS THE ADDRESS+2 OF THE MOST RECENTLY EXECUTED "SCOPE" INSTRUCTION. HOWEVER, IF A SECOND TRAP OCCURS BEFORE THE FIRST IS COMPLETELY SERVICED, THE PROGRAM WILL HALT.

7.0 WARNINGS AND EXCEPTIONS

7.1 WARNINGS

ANY DRIVE THAT IS NOT "WRITE PROTECTED" WILL BE WRITTEN ON (EXCEPT UNIT 0 OF THE XXDP LOAD DEVICE IN CHAIN MODE).

WHEN THE PROGRAM IS DUMPED (SEE SECTION 5.3) AND SW<7> IS SET, THE DEVICES AND DRIVES THAT ARE NOT WRITE PROTECTED WILL BE IDENTIFIED ON THE TERMINAL. BEFORE TYPING A CHARACTER TO CONTINUE, A DRIVE CAN BE WRITE PROTECTED WITHOUT CAUSING AN ERROR BECAUSE, THE SYSTEM IS SIZED AGAIN.

7.2 EXCEPTIONS

IF ANY OF THE DEVICES IS LOCATED AT A NON-STANDARD ADDRESS (SEE BELOW), THE DEVICE REGISTER ADDRESS TABLES (IN "COMMON TAGS") SHOULD BE CHANGED TO THE CORRECT ADDRESSES. FOLLOWING IS THE DEFAULT ADDRESS OF THE CONTROL AND STATUS REGISTER OF EACH DEVICE:

RP03----176714

```
RK05----177404
RP04----176700
```

RS03/4--172040

IF THE SYSTEM HAS BOTH AN RP03 AND AN RP04, THE BRANCH INSTRUCTION AT 100$, IN THE "SIZE ROUTINE" MUST BE REPLACED BY A NOP (240) FOR BOTH DEVICES TO BE USED. THIS BRANCH IS APPROXIMATELY AT ADDRESS 4706. ALSO, THE OPERATER MUST CHANGE THE RP04 ADDRESSES IN THE TABLE IN THE "COMMON TAGS" AREA.

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

THE EXECUTION TIME IS DEPENDENT ON THE AMOUNT OF MEMORY ON THE SYSTEM. FOLLOWING ARE TWO TYPICAL RUN TIMES:

1. MANUFACTURING BASIC LINE-16K MEMORY,UBE, MBT, AND NO DISKS---3 MINUTES.
2. SYSTEM-128K MEMORY, 2 RK05'S, RP04, AND 2 RS04'S --15 MINUTES.

8.2 STACK POINTER

THE STACK POINTER IS SET TO 700.

NOTE

WHEN THE PROGRAM IS RUNNING IN USER MODE, THE USER STACK POINTER IS SET TO 700 AND THE KERNEL STACK POINTER IS SET TO 1200. THE KERNEL STACK POINTER IS USED ONLY FOR THE ERROR AND INTERRUPT SERVICE ROUTINES.

8.3 PASS COUNT

THERE ARE TWO WORDS USED FOR EFFECTIVE PASS COUNT. LOCATION "SUBPASS" AND "$PASS". SUBPASS CONTAINS THE ASCII REPRESENTATION OF THE SUBPASS COUNT. THIS IS USED TO INDEX THE PSW TABLE (SEE SECTION 8.8).

FOUR SUBPASSES ARE EXECUTED FOR EACH PASS. THIS ALLOWS ALL PSW COMBINATIONS TO BE TESTED BEFORE REPORTING END OF PASS.

8.4 END OF PASS MESSAGES

AT THE END OF EACH SUBPASS THE SUBPASS NUMBER (THAT IS BEING STARTED) IS TYPED FOLLOWED BY "THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789". IF RUNNING ON ACT11 QV OR AA,

```
ONLY  THE  SUB-PASS  NUMBER IS TYPED.  AT THE END OF EACH PASS
THE ELAPSED RUN TIME AND THE MESSAGE "END PASS X TOTAL  ERRORS
SINCE  LAST  REPORT  Y" IS TYPED.  A SAMPLE OF WHAT A COMPLETE
```

PASS LOOKS LIKE IS SHOWN BELOW:

```
0THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789
1THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789
2THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789
3THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789
END PASS 1 TOTAL ERRORS SINCE LAST REPORT 0
```

8.5 ITERATIONS

SUB-TEST ITERATIONS ARE NOT PERFORMED UNTIL THE PASS COUNT ($PASS) IS NON-ZERO. THIS MAKES A QV PASS AS SHORT AS POSSIBLE.

AFTER THE FIRST PASS, FULL 10 OCTAL ITERATIONS ARE PERFORMED ON EACH SUBTEST.

8.6 T-BIT TRAPPING

T BIT TRAPPING IS CONTROLLED BY THE PSW TABLE. THE DEFAULT CONDITION IS TO RUN WITH THE T-BIT ON DURING SUBPASSES 2, 4, AND 6.

8.7 ACT-11 COMPATABILITY

THE PROGRAM IS FULLY ACT-11 COMPATABLE. THE PROGRAM OPERATES IN THE ACT-11 MODE UNDER APT.

8.8 PSW TABLE

AT THE END OF THE PROGRAM, JUST BEFORE THE MESSAGES, IS THE PSW TABLE. THIS TABLE CONTROLS WHAT MODE WILL BE EXECUTED ON A SUBPASS. REFER TO SECTION 9.4.2 FOR A DESCRIPTION OF HOW THIS TABLE IS USED BY THE PROGRAM. THIS TABLE MAY BE MODIFIED IF DESIRED.

8.9 I/O DEVICE ADDRESS MODIFICATION

TO MODIFY THE PROGRAM ADDRESS OF THE I/O DEVICES PATCH THE APPROPRIATE DEVICE TABLE (IN THE COMMON TAGS AREA) TO THE DESIRED ADDRESSES.

IF YOU ARE PATCHING THE RP03 OR RP04 SEE SECTION 7.2.

8.10 POWER FAIL

IF A POWER FAIL OCCURS (FOLLOWED BY A POWER UP), THE WORD

715 "POWER" IS TYPED ON THE TERMINAL AND THE PROGRAM RESTARTS.

9.0 PROGRAM DESCRIPTION

THE PROGRAM IS DIVIDED INTO 9 SECTIONS OF POSITION INDEPENDENT RELOCATABLE TEST CODE. EACH SECTION IS APPROXIMATELY 1K WORDS LONG.

WHEN THE PROGRAM IS INITIALLY LOADED AND STARTED IT WILL IDENTIFY ITSELF AND TYPE THE FUNCTION OF THE SWITCH REGISTER AND THE DEVICES AND DRIVES THAT WILL BE USED FOR RELOCATION, IF SW7=0. IT WILL ALSO TYPE THE CP OPTIONS AVAILABLE INDICATOR WORD (OPT.CP). THE CONTENTS OF OPT.CP CONTAIN THE FOLLOWING INDICATORS:

```
BIT15        =    NOT USED
BIT14        =    NOT USED
BIT13        =    NOT USED
BIT12        =    NOT USED
BIT11        =    NOT USED
BIT10=1/0    =    MBT AVAILABLE/NOT AVAILABLE
BIT09=1/0    =    KW11-L AVAILABLE/NOT AVAILABLE
BIT08=1/0    =    CONSOLE TTY AVAILABLE/NOT AVAILABLE
BIT07=1/0    =    UBE AVAILABLE/NOT AVAILABLE
BITS06-00    =    NOT USED
```

FOLLOWING IS A BRIEF DESCRIPTION OF EACH SECTION:

SECTION 0 THIS SECTION CAUSES A 256 WORD 3 XOR 9 TEST PATTERN TO BE RELOCATED THROUGHOUT MEMORY 0 - 28K.

NOTE: THIS SHOULD NOT BE CONSTRUED TO BE A COMPLETE MEMORY TEST.

SECTION 1 THIS SECTION TESTS THE UNARY INSTRUCTION SET EXECUTING EACH UNARY INSTRUCTION IN EACH ADDRESS MODE (EXCLUDING UNARY INSTRUCTIONS USING ADDRESS MODE 7).

SECTION 2 THIS SECTION TESTS THE UNARY INSTRUCTIONS USING ADDRESS MODE 7 AND BINARIES IN ALL ADDRESS MODES (EXCLUDING BINARY BYTE OPS USING ADDRESS MODE 7).

SECTION 3 THIS SECTION TESTS BINARY BYTE OPS USING ADDRESS MODE 7, JMP, JSR AND PROGRAM TRAP (IOT, TRAP, AND EMT) INSTRUCTION.

SECTION 4 THIS SECTION CHECKS THAT EACH BIT IN THE PROCESSOR STATUS WORD (PSW) CAN BE SET CLEARED, RESERVED INSTRUCTIONS, AND ODD ADDRESS TRAPS. NOTE THAT BITS 12 AND 14 IN THE PSW ARE COPIES OF BITS 13 AND 15 RESPECTIVELY. THE WCS IS TURNED OFF AT THE BEGINNING OF THE PROGRAM BY INITIALIZING THE WHAMI REGISTER SO THAT THE RESERVED INSTRUCTIONS TRAP.

```
772
773                          SECTION 5   THIS SECTION CHECKS THE SXT, XOR, SOB,  MARK,  RTT
```

AND RTT INSTRUCTIONS.

SECTION 6 THIS SECTION CHECKS THE ASH, ASHC, MUL, DIV INSTRUCTIONS. ALSO CHECKED IN THIS SECTION ARE THE MED INSTRUCTION AND ERROR LOGGING CAPABILITIES.

SECTION 7 THIS SECTION CHECKS THE STACK LIMIT REGISTER MEMORY MANAGEMENT ABORT LOGIC AND THE MEMORY MANAGEMENT REGISTERS.

SECTION 8 THIS SECTION CHECKS THE FLOATING POINT PROCESSOR.

FOLLOWING SECTION 8 ARE TWO ROUTINES TO CHECK THE TELETYPE PRINTER LOGIC AND A ROUTINE TO START THE KW11-L CLOCK. IF THE KW11-L IS AVAILABLE THE PRIORITY ARBITRATION LOGIC IS TESTED.

9.1 UNIBUS EXERCISER(UBE)

ANY ONE OF 4 UBE'S WILL BE USED. THE PROGRAM LOOKS FOR A UBE AT ADDRESSES 770000, 770020, 770040, AND 770060.

TEST 106 WILL INITIATE THE UNIBUS EXERCISER IF IT IS PRESENT. THIS IS ONLY DONE ON PASS 1 - SUBPASS 1, SINCE FROM THAT POINT ON, THE SERVICE ROUTINE TAKES CARE OF RESTARTING IT.

THE UBE IS INITIALLY SET UP WITH A BUS ADDRESS OF 0. THE FUNCTION THAT IS LOADED IS "DATA IN PAUSE-DATA OUT BYTE". THE WORD COUNT IS SET FOR 4K BYTES. IT IS ALSO SET TO INTERRUPT ON LEVEL 5.

WHEN AN INTERRUPT OCCURS A CHECK IS MADE TO SEE IF IT WAS CAUSED BY AN ERROR. IF THERE WAS NO ERROR, 776 IS LOADED AS THE BUS ADDRESS AND THE UBE IS STARTED AGAIN. ON THE NEXT INTERRUPT 1774 (776+776) IS LOADED AS THE BUS ADDRESS. THIS SEQUENCE CONTINUES UNTIL A MEMORY TIMEOUT ERROR OCCURS.

WHEN AN ERROR OCCURS A CHECK IS MADE TO SEE IF IT WAS CAUSED BY A MEMORY TIMEOUT. IF IT WAS, THE ADDRESS IN THE UBE BUS ADDRESS REGISTER IS COMPARED WITH THE LAST ADDRESS IN THE SYSTEM (MXMMHI AND MXMMLO). IF THEY ARE THE SAME (NO HOLES IN MEMORY) THE UBE IS RESTARTED AT ADDRESS 0 AND THE ABOVE SEQUENCE IS REPEATED. IF THE ADDRESSES ARE NOT THE SAME A MEMORY-HOLE ERROR IS REPORTED.

IF THE ERROR WAS NOT DUE TO A TIMEOUT A UBE ERROR IS REPORTED.

9.2 MASS BUS TESTER(MBT)

ANY ONE OF 4 MBT'S WILL BE USED. THE PROGRAM LOOKS FOR AN MBT AT ADDRESSES 770100, 770200, 770300, AND 770400. IF AN MBT IS

```
830                                     FOUND, THE DRIVE TYPE REGISTER (770X26)  IS  CHECKED  TO  MAKE
831                                     SURE THAT IT REALLY IS AN MBT.
```

TEST 106 ALSO INITIATES THE MASS BUS TESTER. AGAIN, THIS IS ONLY DONE ON PASS 1 - SUBPASS 1 SINCE THE SERVICE ROUTINE KEEPS IT RUNNING.

THE BUS ADDRESS REGISTER IS INITIALLY SET TO 0, THE WORD COUNT TO 2K WORDS, AND A READ FUNCTION IS INITIATED.

WHEN AN INTERRUPT OCCURS AN ERROR CHECK IS MADE. THIS ERROR CHECK IS THE SAME AS THAT DESCRIBED FOR THE UBE. IF THERE WAS NO ERROR, THE WORD COUNT IS RELOADED AND THE FUNCTION IS ISSUED. THE BUS ADDRESS REGISTER IS NOT CHANGED SO IT WILL CONTINUE FROM WHERE IT LEFT OFF.

9.3 LINE CLOCK INITIALIZATION

TEST 106 TURNS ON THE LINE CLOCK. TWO LOCATIONS IN "COMMON TAGS" KEEP TRACK OF THE ELAPSED RUN TIME OF THE PROGRAM. WHEN THE CLOCK INTERRUPTS, THE LOW BYTE OF LOCATION "LTICKS"IS INCREMENTED. WHEN THIS BYTE GETS TO 60(DECIMAL) IT IS CLEARED AND THE HIGH BYTE IS INCREMENTED(SECONDS). WHEN THE SECOND COUNT GETS TO 60(DECIMAL) LOCATION "MTICKS" IS INCREMENTED AND LTICKS IS CLEARED. THIS GIVES THE TIMER A 64K DECIMAL MINUTE RANGE.

NOTE

> FOR THE UBE, MBT, AND LINE CLOCK, WHEN AN INTERRUPT OCCURS, PROGRAM ECUTION RETURNS TO KERNEL MODE AND THE KERNEL PAR'S ARE MAPPED DOWN TO THE 0-12K BANK OF MEMORY. UPON RETURNING FROM THE INTERRUPT THE PAR'S ARE MAPPED BACK TO WHERE THEY WERE AND THE PREVIOUS PROCESSOR MODE IS RESTORED.

9.4 RELOCATION ALGORITHM

9.4.1 SECTION RELOCATION

AS EACH SECTION IS ENTERED THE VIRTUAL START ADDRESS IS SAVED IN LOCATION "FRSTAD" AND THE RELOCATION FACTOR (BYTE OFFSET FROM NON-RELOCATED CODE) IS CALCULATED AND SAVED IN LOCATION "FACTOR". THE TEST CODE IS THEN EXECUTED.

AT THE END OF EACH SECTION, CONTROL IS TRANSFERED TO THE "RELOCATION ROUTINE". IF SW(8) IS SET, THIS ROUTINE WILL RELOCATE THE SECTION VIA THE CP (SEE 9.4.3). IF SW(8) IS NOT SET, THE LENGTH OF THE SECTION IS CALCULATED, SAVED AS A WORD COUNT, AND CONTROL IS TRANSFERED TO THE "I/O MONITOR" (SEE SECTION 9.4.4) WHICH RELOCATES THE SECTION BY USING A DISK.

```
888                                     EACH SECTION IS INITIALLY RELOCATED TO THE END ADDRESS OF  THE
889                                     PROGRAM.   SUBSEQUENT  RELOCATIONS  START  AT  THE  END OF THE
```

PREVIOUS RELOCATION. FOR EXAMPLE: IF SECTION 0 IS 1000 BYTES LONG AND THE END ADDRESS OF THE PROGRAM IS 60000, THE FIRST RELOCATION STARTS AT ADDRESS 60000, THE SECOND AT 61000, THE THIRD AT 62000, ETC. THIS CONTINUES UNTIL 28K OR THE END OF MEMORY HAS BEEN REACHED AT WHICH TIME EXECUTION GOES TO THE START OF THE NEXT SECTION AND THE PROCESS REPEATS WITH THE NEW SECTION.

EACH SECTION IS WRITTEN IN POSITION INDEPENDENT CODE SO THAT IT CAN BE RELOCATED AND EXECUTED WITHOUT THE USE OF MEMORY MANAGEMENT.

9.4.2 PROGRAM RELOCATION

WHEN ALL NINE SECTIONS HAVE BEEN RELOCATED AND EXECUTED THRU 28K (SEE SECTION 9.4.1), MEMORY MANAGEMENT IS SETUP ACCORDING TO THE VALUE IN LOCATION "NEXPAR" (MEMORY MANAGEMENT WILL NOT BE TURNED ON IF THERE IS LESS THAN 28K OF MEMORY). THIS VALUE IS INITIALIZED TO 600 (OR 1600 IF RUNNING UNDER THE XXDP MONITOR), MAKING RELOCATION START AT ADDRESS 60000 (OR 160000). THE "I/O MONITOR" IS THEN ENTERED (SEE SECTION 9.4.4) TO RELOCATE THE PROGRAM. WHEN THE I/O MONITOR COMPLETES THE RELOCATION, EXECUTION IS TRANSFERED TO THE START OF THE PROGRAM AT THE RELOCATED POSITION.

EACH SECTION IS EXECUTED ONLY ONCE WITH MEMORY MANAGEMENT ON. AT THE END OF SECTION 8, 77 IS ADDED TO "NEXPAR" AND RELOCATION IS PERFORMED AGAIN. THIS CAUSES THE NEXT RELOCATION TO MOVE UP BY 7700 BYTES. FOR EXAMPLE: IF NEXPAR=1600 THE FIRST RELOCATION STARTS AT ADDRESS 160000, THE SECOND AT ADDRESS 167700, THE THIRD AT 177600, ETC.

THIS CONTINUES UNTIL THE END OF MEMORY IS REACHED AND CONSTITUTES A SUB-PASS. THE PSW IS THEN SETUP FOR THE NEXT SUB-PASS AND THE PROGRAM RESTARTS.

THE VALUE FOR THE PSW IS TAKEN FROM THE TABLE (SEE SECTION 8.8). THE PARTICULAR ENTRY THAT IS USED IS OBTAINED BY INDEXING THE TABLE BY THE SUB-PASS NUMBER (SEE SECTION 8.3). FOR EXAMPLE, SUB-PASS 3 USES WORD 3 (THE FIRST WORD IS COUNTED AS ZERO) OF EACH TABLE. THEREFORE, TO CHANGE THE VALUE IN THE PSW ONLY REQUIRES CHANGING THE VALUE IN THE TABLE.

THE COMPLETION OF 4 SUB-PASSES CONSTITUTES A PASS AND AN END OF PASS MESSAGE IS TYPED. THE PROGRAM THEN RESTARTS IN PASS 2, SUB-PASS 0.

9.4.3 RELOCATION VIA CP

IF SW(8) IS SET, BOTH SECTION AND PROGRAM RELOCATION (SEE SECTIONS 9.4.1 AND 9.4.2), ARE PERFORMED BY AN INSTRUCTION MOVE LOOP RATHER THAN A DISK. FOR EXAMPLE:

```
946                     1$:  MOV (R0)+,(R2)+
947                               CMP R0,R3
```

BNE 1$

WHERE R0 IS THE ADDRESS OF THE CODE BEING MOVED, R2 IS THE ADDRESS THAT IT IS BEING MOVED TO, AND R3 IS THE LAST ADDRESS THAT IS TO BE MOVED.

WHEN THIS IS FINISHED, THE RELOCATED DATA IS CHECKED BY AN INSTRUCTION COMPARE LOOP TO ENSURE THAT THE RELOCATION WAS PERFORMED CORRECTLY.

9.4.4 RELOCATION VIA I/O

IF SW<8> IS NOT SET, BOTH SECTION AND PROGRAM RELOCATION (SEE SECTION 9.4.1 AND 9.4.2), ARE PERFORMED BY WRITING THE DATA TO A DISK AND READING IT BACK TO THE RELOCATED POSITION. THIS RELOCATION IS CONTROLLED BY THE "I/O MONITOR".

9.4.4.1 SECTION RELOCATION

WHEN THE I/O MONITOR IS ENTERED FROM THE "RELOCATION ROUTINE" (SEE SECTION 9.4.1) A DEVICE IS SELECTED (SEE 9.4.4.3), THE MEMORY ADDRESSES (FROM AND TO) AND WORD COUNT ARE PASSED TO THE DEVICE HANDLER (SEE SECTION 9.4.4.4), AND THE HANDLER IS CALLED. WHEN THE HANDLER FINISHES, THE I/O MONITOR CHECKS THE RELOCATED DATA WITH AN INSTRUCTION COMPARE LOOP TO ENSURE THE RELOCATED DATA IS CORRECT, AND RETURNS TO THE "RELOCATION ROUTINE" (SEE 9.4.1).

9.4.4.2 PROGRAM RELOCATION

WHEN THE I/O MONITOR IS ENTERED FOR PROGRAM RELOCATION (SEE SECTION 9.4.2) THE BASE ADDRESS FOR THE RELOCATION IS CALCULATED FROM THE CONTENTS OF KERNEL PAR3 WHICH WAS SET UP WITH MEMORY MANAGEMENT (SEE 9.4.2). IF SW<8> IS SET, RELOCATION IS PERFORMED VIA THE CP (SEE SECTION 9.4.3).

IF SW<8> IS NOT SET, A DEVICE IS SELECTED (SEE 9.4.4.3), THE WORD COUNT IS SET TO 2K, AND THE MEMORY ADDRESSES (FROM AND TO) AND WORD COUNT ARE PASSED TO THE DEVICE HANDLER (SEE 9.4.4.4), AND THE HANDLER IS CALLED. THE I/O MONITOR THEN ADDS 2K TO THE MEMORY ADDRESSES, SELECTS ANOTHER DEVICE, PASSES THE ADDRESSES TO THE DEVICE HANDLER, AND CALLS THE HANDLER. THIS CONTINUES UNTIL ALL 12K HAS BEEN RELOCATED. THE RELOCATED DATA IS THEN CHECKED WITH AN INSTRUCTION COMPARE LOOP. THE RELOCATED PROGRAM IS THEN EXECUTED AS DESCRIBED IN 9.4.2.

9.4.4.3 DEVICE SELECTION

IF SW<5> IS NOT SET, AN INDEX IS PICKED UP FROM LOCATION "DEVINDX". THIS INDEX IS USED TO INDEX THE SYSTEM SIZE TABLE. THE SYSTEM SIZE TABLE CONSISTS OF 8 WORDS (ONE FOR EACH DEVICE TYPE). BITS <7:0> OF EACH WORD ARE USED TO INDICATE THE DRIVE

```
                                        NUMBERS THAT ARE AVAILABLE ON THE DEVICE, AND ARE  INITIALIZED
                                        IN  THE  SIZE  ROUTINE.   BITS <15:8> OF EACH WORD ARE USED TO
```

INDICATE WHETHER THE DRIVE HAS BEEN USED FOR A DATA TRANSFER (UNIT USED BIT).

THE SYSTEM SIZE TABLE IS THEN SEARCHED, USING THE INDEX DESCRIBED ABOVE, FOR A DRIVE THAT HAS NOT BEEN USED. WHEN A DRIVE IS FOUND, THE "UNIT USED BIT" IS SET, THE CURRENT INDEX IS PUT BACK IN LOCATION DEVINDX, AND EXECUTION CONTINUES AS DESCRIBED IN 9.4.4.1 OR 9.4.4.2.

IF AN UNUSED UNIT IS NOT FOUND, ALL THE "UNIT USED" BITS ARE CLEARED AND THE SEARCH IS RESTARTED. IF THE SEARCH FINDS THE SYSTEM SIZE TABLE EMPTY (NO DEVICES ON THE SYSTEM), THE MESSAGE "NO I/O DEVICES" IS TYPED AND RELOCATION IS PERFORMED VIA THE CP AS DESCRIBED IN 9.4.3.

IF SW<5> IS SET, SW'S<2:0> ARE USED TO INDEX THE SYSTEM SIZE TABLE. IN THIS CASE ONLY ONE WORD OF THE TABLE IS USED CORRESPONDING TO THE DEVICE BEING SELECTED BY SW'S<2:0> (SEE SECTION 5.1). IN THIS MODE, A ROUND ROBIN SELECTION IS PERFORMED ON THE DRIVES OF THE SELECTED DEVICE.

9.4.4.4 DEVICE HANDLERS

EACH DEVICE THAT IS USED FOR RELOCATION HAS A HANDLER. THESE HANDLERS ARE FUNCTIONALLY THE SAME.

THE HANDLER IS CALLED BY THE I/O MONITOR (SEE SECTION 9.4.4). IT FIRST CLEARS THE DONE BIT (BIT 7) IN THE HANDLER STATUS WORD. THIS PREVENTS THE MONITOR FROM CALLING THIS HANDLER AGAIN BEFORE IT IS FINISHED.

IF A "DEVICE HUNG" ERROR (SEE SECTION 6.1.12) IS DETECTED, THE HANDLER STATUS WORDS CAN BE EXAMINED TO DETERMINE WHICH DEVICE DID NOT FINISH (SET BIT 7). THE DRIVE CAN THEN BE DETERMINED BY LOOKING IN THE "DEVICE HANDLER UNIT NUMBER" TABLE. THE HANDLER STATUS WORDS AND DEVICE HANDLER UNIT NUMBER TABLES, ARE LOCATED IN THE "COMMON TAGS" AREA OF THE LISTING.

THEN THE HANDLER CALCULATES A DISK ADDRESS. THIS ADDRESS IS EITHER GENERATED FROM A RANDOM NUMBER (SW4=0) OR IS SET TO ZERO (SW4=1). THE DEVICE ID, UNIT NUMBER, AND CYLINDER ADDRESS ARE COMBINED AND PLACED IN THE "RUN TABLE" (RUNTBL). THE POSITION IN THE RUN TABLE CORRESPONDS TO WHICH 2K BLOCK OF THE PROGRAM IS BEING TRANSFERED (I.E. THE FIRST 2K BLOCK IS IDENTIFIED BY WORD 1, THE SECOND 2K BY WORD 2, ETC.). THE BIT CONFIGURATION OF EACH WORD IN THE RUN TABLE IS AS FOLLOWS:

```
<15:13> = DEVICE ID
<12:10> = UNIT NUMBER
    <9> = NOT USED
  <8:0> = CYLINDER ADDRESS
```

THE TRACK-SECTOR ADDRESS OF THE TRANSFER IS SAVED IN THE "RUN

```
                                        TRACK  TABLE"  (RUNTRAK).   THE  POSITION  IN THIS TABLE IS AS
                                        DESCRIBED ABOVE.  THE BIT CONFIGURATION OF EACH  WORD  IS  THE
```

SAME AS THAT FOR THE DISK ADDRESS REGISTER FOR THE PARTICULAR DEVICE. BIT 15 IS USED TO INDICATE A DEVICE ERROR. IT IS SET BY THE DEVICE SERVICE ROUTINE.(SEE SECTION 9.4.4.5)

THE HANDLER THEN INITIALIZES THE DEVICE REGISTERS WITH ALL THE APPROPRIATE INFORMATION AND STARTS A WRITE FUNCTION. EXECUTION THEN RETURNS TO THE I/O MONITOR AT THE POINT WHERE THE HANDLER WAS CALLED.

9.4.4.5 DEVICE SERVICE ROUTINES

EACH DEVICE THAT IS USED FOR RELOCATION HAS A SERVICE ROUTINE. THESE ROUTINES ARE ALL FUNCTIONALLY THE SAME.

THE ROUTINE IS ENTERED BY A DEVICE INTERRUPT. THE DEVICE IS CHECKED FOR ANY ERRORS. IF NO ERROR OCCURRED THE DEVICE REGISTERS ARE LOADED AND THE NEXT FUNCTION TO PERFORM IS INITIATED. THREE FUNCTIONS ARE EXECUTED: WRITE, WRITE CHECK, AND READ. ALL THE NECESSARY BUS ADDRESS INFORMATION IS CALCULATED BY THE I/O MONITOR, SO THE SERVICE ROUTINE JUST TAKES CARE OF THE DEVICE.

WHEN THE READ FUNCTION HAS BEEN COMPLETED SUCCESSFULLY, THE DONE BIT (BIT 7) IN THE HANDLER STATUS WORD IS SET.

UPON INITIATION OF A FUNCTION, OR COMPLETION OF ALL THREE FUNCTIONS, THE SERVICE ROUTINE RETURNS EXECUTION TO WHERE IT WAS WHEN IT WAS INTERRUPTED.

IF AN ERROR IS DETECTED, THE FUNCTION THAT FAILED IS RETRIED TWO MORE TIMES. IF THE ERROR IS STILL PRESENT THE DONE BIT AND THE ERROR BIT (BIT 15) IS SET IN THE HANDLER STATUS WORD ALONG WITH BIT 15, IN THE APPROPRIATE ENTRY, IN THE RUN TRACK TABLE, AND THE ROUTINE EXITS AS DESCRIBED ABOVE.

```
.TITLE  MAINDEC-11-DQKDC-A PDP 11/6X SERIES CPU EXERCISER
;*COPYRIGHT (C) JANUARY,1977
;*DIGITAL EQUIPMENT CORP.
;*MAYNARD, MASS. 01754
;*
;*
;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
;*
```

```










.SBTTL  OPERATIONAL SWITCH SETTINGS
;*
;*      SWITCH                  USE
;*      ------          --------------------
;*        15            HALT ON ERROR
;*        14            LOOP ON TEST
;*        13            INHIBIT ERROR TYPEOUTS
;*        12            INHIBIT UBE
;*        11            INHIBIT ITERATIONS
;*        10            BELL ON ERROR
;*         9            LOOP ON ERROR
;*         8            INHIBIT RELOCATION VIA I/O DEVICE
;*         7            INHIBIT SYSTEM SIZE TYPEOUT
;*         6            INHIBIT RELOCATION
;*         5            INHIBIT ROUND ROBIN
;*         4            INHIBIT RANDOM DISK ADDRESS
;*         3            INHIBIT MBT
;*         2            THESE THREE SWITCHES
;*         1            ARE ENCODED TO SELECT RELOCATION
;*         0            ON THE FOLLOWING DEVICES:
;*              0...RP11/RP03
;*              1...RK11/RK05
;*              2...NOT USED
;*              3...NOT USED
;*              4...RH11/RP04
;*              5...RH11/RS04
;*              6...NOT USED
;*              7...NOT USED

;*      SWITCH          OCTAL VALUE
;*      ------          -----------
;*        15            100000
;*        14             40000
;*        13             20000
;*        12             10000
;*        11              4000
;*        10              2000
;*         9              1000
;*         8               400
;*         7               200
;*         6               100
;*         5                40
;*         4                20
;*         3                10
;*         2                 4
;*         1                 2
```

```
1169                                    ;*         0                1
```

```
                                .SBTTL  BASIC DEFINITIONS

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

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

                                ;*GENERAL PURPOSE REGISTER DEFINITIONS
          000000                R0=     %0              ;;GENERAL REGISTER
          000001                R1=     %1              ;;GENERAL REGISTER
          000002                R2=     %2              ;;GENERAL REGISTER
          000003                R3=     %3              ;;GENERAL REGISTER
          000004                R4=     %4              ;;GENERAL REGISTER
          000005                R5=     %5              ;;GENERAL REGISTER
          000006                R6=     %6              ;;GENERAL REGISTER
          000007                R7=     %7              ;;GENERAL REGISTER
          000006                SP=     %6              ;;STACK POINTER
          000007                PC=     %7              ;;PROGRAM COUNTER

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

                                ;*"SWITCH REGISTER" SWITCH DEFINITIONS
          100000                SW15=   100000
          040000                SW14=   40000
          020000                SW13=   20000
          010000                SW12=   10000
          004000                SW11=   4000
          002000                SW10=   2000
          001000                SW09=   1000
          000400                SW08=   400
          000200                SW07=   200
          000100                SW06=   100
          000040                SW05=   40
          000020                SW04=   20
          000010                SW03=   10
          000004                SW02=   4
```

```
        000002          SW01=   2
        000001          SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

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

                        ;*BASIC "CPU" TRAP VECTOR ADDRESSES
        000004          ERRVEC= 4               ;;TIME OUT AND OTHER ERRORS
        000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
        000014          TBITVEC=14              ;;"T" BIT
        000014          TRTVEC= 14              ;;TRACE TRAP
        000014          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
        000020          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
        000024          PWRVEC= 24              ;;POWER FAIL
        000030          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
        000034          TRAPVEC=34              ;;"TRAP" TRAP
        000060          TKVEC=  60              ;;TTY KEYBOARD VECTOR
        000064          TPVEC=  64              ;;TTY PRINTER VECTOR
        000240          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                        .SBTTL  MEMORY MANAGEMENT DEFINITIONS
```

```
                               ;*KT11 VECTOR ADDRESS

        000250                 MMVEC=  250

                               ;*KT11 STATUS REGISTER ADDRESSES

        177572                 SR0=    177572
        177574                 SR1=    177574
        177576                 SR2=    177576
        172516                 SR3=    172516

                               ;*USER "I" PAGE DESCRIPTOR REGISTERS

        177600                 UIPDR0= 177600
        177602                 UIPDR1= 177602
        177604                 UIPDR2= 177604
        177606                 UIPDR3= 177606
        177610                 UIPDR4= 177610
        177612                 UIPDR5= 177612
        177614                 UIPDR6= 177614
        177616                 UIPDR7= 177616

                               ;*USER "I" PAGE ADDRESS REGISTERS

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

                               ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

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

                               ;*KERNEL "I" PAGE ADDRESS REGISTERS

        172340                 KIPAR0= 172340
        172342                 KIPAR1= 172342
        172344                 KIPAR2= 172344
        172346                 KIPAR3= 172346
        172350                 KIPAR4= 172350
        172352                 KIPAR5= 172352
        172354                 KIPAR6= 172354
        172356                 KIPAR7= 172356

        000700                         USESTK  =STACK-300
```

```
        000200                  CRLF    =200
        076600                  MED     =76600          ;OPCODE FOR "MED" INSTRUCTION
        000114                  CACHVEC =114            ;CACHE ERROR INTERRUPT VECTOR
        177744                  MEMERR  =177744         ;CACHE ERROR REGISTER
        177746                  CCR     =177746         ;MEMORY CONTROL REGISTER
        177752                  HITMIS  =177752         ;HIT-MISS REGISTER, A "1"
                                                        ;IMPLIES A HIT IN CACHE
        177766                  CPUERR  =177766         ;CPU ERROR REGISTER HOLDS CONDITION
                                                        ;THAT CAUSED TRAP TO ERRVEC (004)
        177770                  UBREAK  =177770         ;ADDRESS OF MICROBREAK REG.

                                .EQUIV  SP,KSP
                                .EQUIV  SP,USP
                                .EQUIV  STACK,KERSTK
                                .EQUIV  SR0,MMR0
                                .EQUIV  SR2,MMR2
        000000                  AC0=    %0
        000001                  AC1=    %1
        000002                  AC2=    %2
        000003                  AC3=    %3
        000004                  AC4=    %4
        000005                  AC5=    %5
                        ;LINE CLOCK AND PROGRAMMABLE LINE CLOCK REGISTERS
        172540                  PLKCSR=172540
        172542                  PLKCSB=172542
        000104                  PLKVEC=104
        177546                  LKS=177546
        000100                  LKVEC=100

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

                        ;MASS BUS TESTER REGISTERS
        160100                  MBTCS1= 160100          ;CONTROL & STATUS REG. 1
        160102                  MBTWC=  160102          ;WORD COUNT
        160104                  MBTBA=  160104          ;BUS ADDRESS REG.
        160106                  MBTMR2= 160106          ;MAINTENANCE REG. 2
        160110                  MBTCS2= 160110          ;CONTROL & STATUS REG. 2
        160112                  MBTST=  160112          ;STATUS REG. (TAPE)
        160114                  MBTER=  160114          ;ERROR REGISTER
        160116                  MBTAS=  160116          ;ATTENTION SUMMARY REG.
        160120                  MBTDB=  160120          ;DATA BUFFER
        160124                  MBTMR1= 160124          ;MAINTENANCE REG. 1
        160126                  MBTDT=  160126          ;DRIVE TYPE REG.
        160174                  MBTBAE= 160174          ;BUS ADDRESS EXTENSION REG.
        160176                  MBTCS3= 160176          ;CONTROL & STATUS REG. 3
        000774                  MBTVEC= 774             ;INTERRUPT VECTOR
        000776                  MBTPSW= 776
```

```
                                ;MISCELLANEOUS BIT ASSIGNMENTS (USED IN OPT.CP)
        100000                          KTOPT=  100000          ;BIT15 - BELOW BIT ASSIGNMENTS ARE USED
        040000                          EISOPT= 040000          ;BIT14 - IN THE CPCHK ROUTINE
        020000                          FPOPT=  020000          ;BIT13 - A BIT FOR EACH OPTION PRESENT
        002000                          MBTOPT= 002000          ;BIT10 - BITS 15,14,13,9 NON OPTION BITS
        001000                          LKOPT=  001000          ;BIT09 - BITS 12,11,06-00 NOT USED
        000400                          TTOPT=  000400          ;BIT08
        000200                          UBEOPT= 000200          ;BIT07

                                ;MED OPERATION CODE DEFINITIONS
        000022                          RDWHAMI=022
        000144                          RDFLAG=144
        000222                          WRWHAMI=222
        000226                          WCNSSW=226
        000344                          WRFLAG=344
        000100                          RDLJAM=100
        000300                          WRLJAM=300
        000101                          RDLSERVICE=101
        000301                          WRLSERVICE=301
        000102                          RDLPBA=102
        000302                          WRLPBA=302
        000103                          RDLCUA=103
        000303                          WRLCUA=303
        000104                          RDLFGINT=104
        000304                          WRLFGINT=304
        000105                          RDLWHAMI=105
        000305                          WRLWHAMI=305
        000106                          RDLDATA=106
        000306                          WRLDATA=306
        000107                          RDLTAG=107
        000307                          WRLTAG=307
        000352                          WRINIT=352
        000071                          SWB01=71                ;MICRO ADDR. IN SWAB INST.
        000710                          SWB1A=710               ;SAME MICRO ADDR. SHIFTED RIGHT

                                ;ADDITIONAL DEFINITIONS
                                        .EQUIV  ERROR,HLT
        000010                          CALLHANDLER=10
        000000                          KM=0
        140000                          UM=140000
        000000                          PKM=0
        030000                          PUM=30000
        000100                          WWP=BIT6
        000001                          DPTRP=BIT0
        000200                          PABORT=BIT7
        000100                          LO=BIT6
        000200                          HI=BIT7
        000040                          TAG=BIT5

                                .SBTTL  TRAP CATCHER

        000000                          .=0
                                ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
                                ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
                                ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
        000174                          .=174
```

```
 1450  000174  000000                    DISPREG: .WORD  0                 ;;SOFTWARE DISPLAY REGISTER
 1451  000176  000000                    SWREG:   .WORD  0                 ;;SOFTWARE SWITCH REGISTER
 1452                                    .SBTTL  STARTING ADDRESS(ES)
 1453  000200  000137  003276                    JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM
 1454                                    .SBTTL  ACT11 HOOKS
 1455
 1456                                    ;;*************************************************************
 1457                                    ;HOOKS REQUIRED BY ACT11
 1458          000204                            $SVPC=.                   ;SAVE PC
 1459          000046                            .=46
 1460  000046  041230                            $ENDAD                    ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
 1461          000052                            .=52
 1462  000052  040000                            .WORD   40000             ;;2)SET LOC.52 TO 40000
 1463          000204                            .=$SVPC                   ;; RESTORE PC
```

```
                        .SBTTL  COMMON TAGS

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

        001200                  .=1200
001200                  $CMTAG:                         ;;START OF COMMON TAGS
001200  000000          $PASS:  .WORD   0               ;;CONTAINS PASS COUNT
001202  000000          $TSTNM: .WORD   0               ;;CONTAINS THE TEST NUMBER
001204     000          $ERFLG: .BYTE   0               ;;CONTAINS ERROR FLAG
        001206                  .EVEN
001206  000000          $ICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001210  000000          $LPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001212  000000          $LPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001214  000000          $ERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001216     000          $ITEMB: .BYTE   0               ;;CONTAINS ITEM CONTROL BYTE
001217     001          $ERMAX: .BYTE   1               ;;CONTAINS MAX. ERRORS PER TEST
001220  000000          $ERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001222  000000          $GDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001224  000000          $BDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001226  000000          $GDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001230  000000          $BDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001232  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001234  000000                  .WORD   0
001236     000          $AUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001237     000          $INTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001240  000000                  .WORD   0
001242  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001244  177570          DISPLAY: .WORD  DDISP           ;;ADDRESS OF DISPLAY REGISTER
001246  177560          $TKS:   177560                  ;;TTY KBD STATUS
001250  177562          $TKB:   177562                  ;;TTY KBD BUFFER
001252  177564          $TPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001254  177566          $TPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001256     000          $NULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001257     002          $FILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001260     012          $FILLC: .BYTE   12              ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001261     000          $TPFLG: .BYTE   0               ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001262  000000          $REGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH ($REG0) WAS OBTAINED
001264  000000          $REG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001266  000000          $REG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
001270  000000          $REG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
001272  000000          $REG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
001274  000000          $REG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
001276  000000          $REG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
001300  000000          $REG6:  .WORD   0               ;;CONTAINS (($REGAD)+14)
001302  000000          $REG7:  .WORD   0               ;;CONTAINS (($REGAD)+16)
001304  000000          $TMP0:  .WORD   0               ;;USER DEFINED
001306  000000          $TMP1:  .WORD   0               ;;USER DEFINED
001310  000000          $TMP2:  .WORD   0               ;;USER DEFINED
001312  000000          $TMP3:  .WORD   0               ;;USER DEFINED
001314  000000          $TMP4:  .WORD   0               ;;USER DEFINED
001316  000000          $TMP5:  .WORD   0               ;;USER DEFINED
001320  000000          $TMP6:  .WORD   0               ;;USER DEFINED
001322  000000          $TMP7:  .WORD   0               ;;USER DEFINED
```

```
 1520  001324  000000                   $TIMES: 0                                ;;MAX. NUMBER OF ITERATIONS
 1521  001326  000000                   $ESCAPE:0                                ;;ESCAPE ON ERROR ADDRESS
 1522  001330  177607  000377           $BELL:   .ASCIZ   <207><377><377>        ;;CODE FOR BELL
 1523  001334     077                   $QUES:   .ASCII   /?/                    ;;QUESTION MARK
 1524  001335     015                   $CRLF:   .ASCII   <15>                   ;;CARRIAGE RETURN
 1525  001336  000012                   $LF:     .ASCIZ   <12>                   ;;LINE FEED
 1526                                   ;;*********************************************************************
 1527  001340  000000                   ERRRTN: .WORD
 1528  001342  000044                   $FLBUFF:.BLKB     44      ;BUFFER FOR FLOATING POINT CONVERSION
 1529  001406  000000                   $.JFF:   .WORD            ;
 1530  001410  000000                   $AC0:    .WORD            ;EXTENDED EXPONENT VALUES
 1531  001412  000000                   $AC1:    .WORD            ;FOR THE SIX FLOATING POINT
 1532  001414  000000                   $AC2:    .WORD            ;ACCUMULATORS
 1533  001416  000000                   $AC3:    .WORD
 1534  001420  000000                   $AC4:    .WORD
 1535  001422  000000                   $AC5:    .WORD
 1536  001424  000000                   FTMP0:   .WORD   0        ;LOCATIONS TO HOLD FLT. PT. OPERANDS
 1537  001426  000000                   FTMP1:   .WORD   0
 1538  001430  000000                   FTMP2:   .WORD   0
 1539  001432  000000                   FTMP3:   .WORD   0
 1540  001434  000000                   FTMP4:   .WORD   0
 1541  001436  000000                   FTMP5:   .WORD   0
 1542  001440  000000                   FTMP6:   .WORD   0
 1543  001442  000000                   FTMP7:   .WORD   0
 1544  001444  000000                   FREG0:   .WORD   0
 1545  001446  000000                   FREG1:   .WORD   0
 1546  001450  000000                   FREG2:   .WORD   0
 1547  001452  000000                   FREG3:   .WORD   0
 1548  001454  000000                   FREG4:   .WORD   0
 1549  001456  000000                   FREG5:   .WORD   0
 1550  001460  000000                   FREG6:   .WORD   0
 1551  001462  000000                   FREG7:   .WORD   0
 1552  001464  000004                   FLTMP0:  .BLKW   4        ;FLOATING POINT DBL PREC BUFFER
 1553  001474  000004                   FLTMP1:  .BLKW   4
 1554  001504  001506                   TKBFRP:  .WORD   TKBFR    ;POINTER FOR KEYBOARD BUFFER
 1555  001506  000011                   TKBFR:   .BLKW   11       ;KEYBOARD BUFFER
 1556  001530  000000                   NOTYPE:  .WORD            ;NO TYPEOUT FLAG (INHIBIT WHEN SET)
 1557  001532  000000                   OPT.CP:  .WORD            ;CPU OPTION FLAGS
 1558  001534  000006                   $RTRN:   RTT              ;RETURN FOR T-BIT TRAP
 1559  001536  000000                   VADR:    .WORD            ;BUFFER FOR VIRTUAL ADDRESS
 1560  001540  000000                   PA1500:  .WORD            ;BUFFER FOR PHYSICAL ADDRESS BITS<15:00>
 1561  001542  000000                   PA1716:  .WORD            ;PHYSICAL ADDRESS BITS<17:16>
 1562  001544     000                   NEXEC:   .BYTE            ;NO EXECUTE FLAG(NO TEST EXECUTION WHEN SET)
 1563  001545     000                   MMON:    .BYTE            ;MEMORY MGMT FLAG(MGMT IS ON WHEN NON-ZERO)
 1564  001546     000                   QV:      .BYTE            ;QV FLAG(QV PASS WHEN SET)
 1565  001547     000                   AA:      .BYTE            ;AUTO ACCEPT FLAG (AA PASS WHEN SET)
 1566  001550  000000                   FACTOR:  .WORD            ;RELOCATION FACTOR(NUMBER OF
 1567  001552  000000                   $FACTOR: .WORD            ;BYTES ABOVE BASE CODE)
 1568  001554  000000                   FRSTAD:  .WORD            ;FIRST ADDRESS OF SECTION BEING EXECUTED
 1569  001556  000000                   FRSTMEM: .WORD            ;ADDRESS OF FIRST FREE MEMORY
 1570  001560  000000                   LSTMEM:  .WORD            ;ADDRESS OF LAST FREEE MEMORY(IN 28K)
 1571  001562  000000                   NEXPAR:  .WORD            ;NEXT VALUE TO PUT IN PAR0
 1572  001564  123456                   $LONUM:  .WORD   123456   ;LOW 16 BITS OF RANDOM NUMBER
 1573  001566  065432                   $HINUM:  .WORD   65432    ;HIGH 16 BITS OF RANDOM NUMBER
 1574  001570     001     001     001   NULLS:   .BYTE   1,1,1,0  ;BUFFER FOR PRINTER TEST
 1575  001573     000
```

```
001574  000060          SUBPASS:.WORD   60      ;SUB-PASS COUNT IN ASCII
001576  000000          SERPSW: .WORD           ;ERROR PSW FOR TYPEOUT
001600  000000          EXITFL: .WORD
001602  000000          OLDBASE:.WORD           ;SOURCE BASE ADDRESS FOR DEVICE RELOCATION
001604  000C00          NWBASL: .WORD           ;DEST ADDRESS FOR DEVICE RELOC BITS<15:00>
001606  000000          NWBASH: .WORD           ;DEST ADDRESS FOR DEVICE RELOC BITS<21:16>
001610  000000          IOWC:   .WORD           ;TWO'S COMPLIMENT WORD COUNT FOR DEVICE RELOC
001612  000000          DEVICE: .WORD
001614  000000          DEVINDX:.WORD           ;DEVICE INDEX (0 TO 7)
001616  000000          UNITNO: .WORD           ;DEVICE UNIT NUMPER
001620  000000          RNTBINX:.WORD           ;INDEX TO RUN TABLE
001622  000000          MXMMHI: .WORD           ;BITS<21:16> OF LAST MEM ADDRESS ON SYSTEM
001624  000000          MXMMLO: .WORD           ;BITS<15:00> OF LAST MEM ADDRESS ON SYSTEM
001626  000000          RP310:  .WORD           ;DATA TO LOAD INTO RP03 CS REGISTER
001630  000000          RP311:  .WORD           ;RP03 FLAG FOR FIRST 2K OF PROGRAM
001632  000000          RK10:   .WORD           ;DATA TO LOAD INTO RK05 CS REGISTER
001634  000000          RK11:   .WORD           ;RK05 FLAG FOR FIRST 2K OF PROGRAM
001636  000000          RP411:  .WORD           ;RP04 FLAG FOR FIRST 2K OF PROGRAM
001640  000000          RS11:   .WORD           ;RS04 FLAG FOR FIRST 2K OF PROGRAM
001642  000000          MTICKS: .WORD           ;ELAPSED RUN TIME IN MINUTES
001644  000000          LTICKS: .WORD                   ;LOW BYTE=NUMBER OF CLOCK INTERRUPTS (0 TO 59)
                                                ;HIGH BYTE=ELAPSED RUN TIME IN SECONDS(0 TO 59)
001646  000010          SYSSIZE:.BLKW   10      ;SYSTEM SIZE TABLE(ONE ENTRY FOR EACH DEVICE)
001666  000006          RUNTBL: .BLKW   6       ;RUN TIME TABLE(ONE ENTRY FOR EACH 2K BLOCK)
001702  000006          RUNTRAK:.BLKW   6       ;RUN TRACK TABLE(ONE ENTRY FOR EACH 2K BLOCK)
001716  000002          UBESAV: .BLKW   2       ;BASE ADDRESS OF UBE TRANSFER IN PROGRESS
001722  000002          UBEADR: .BLKW   2       ;ADDRESS THAT GETS LOADED INTO UBE BA REG
001726  000002          ERRBA:  .BLKW   2       ;18 BIT UNIBUS ADDRESS WHEN DEVICE DETECTED AN ERROR
001732  000000          MEMFLG: .WORD   0       ;HOLDS FLAG FOR UBE & MBT
001734  000000          ERRADR: .WORD   0       ;HOLDS ADDR. FOR UBE & MBT
001736  000000          REPPSW: .WORD   0       ;HOLDS PSW FOR ERROR REPORT
001740  000000          REPREG: .WORD   0       ;HOLDS REGISTER FOR ERROR REPORT
001742  000000          REPADR: .WORD   0       ;HOLDS LOOP ADDR. FOR ERROR REPROT
001744  000000          REPMR0: .WORD   0       ;HOLDS MMR0 FOR ERROR REPORT
001746  000000          REPMR2: .WORD   0       ;HOLDS MMR2 FOR ERROR REPORT
001750  000000          SRCADR: .WORD   0       ;HOLDS ADDR. DURING DATA CHECK
                        .SBTTL  DEVICE HANDLER STATUS WORDS
                        ;*      EACH WORD HAS THE FOLLOWING BIT ASSIGNMENTS:
                        ;*              7       HANDLER READY
                        ;*              8       REPEAT LAST FUNCTION
                        ;*              15      ERROR
001752  000200          RP3HSTAT:.WORD  200     ;RP03
001754  000200          RKHSTAT:.WORD   200     ;RK05
001756  000200          SPARE0: .WORD   200
001760  000200          SPARE1: .WORD   200
001762  000200          RP4HSTAT:.WORD  200     ;RP04
001764  000200          RSHSTAT:.WORD   200     ;RS04
001766  000200                  .WORD   200     ;SPARE
001770  000200                  .WORD   200     ;SPARE

                        .SBTTL  DEVICE HANDLER WORD COUNTS
                        ;*      THIS TABLE GETS LOADED BY THE I/O
                        ;*      RELOCATION ROUTINE WITH THE TWO'S COMPLIMENT WORD
                        ;*      COUNT FOR THE TRANSFER FOR THE PARTICULAR DEVICE.
001772  000000          RP3HWC: .WORD           ;RP03
001774  000000          RKHWC:  .WORD           ;RK05
```

```
 001776  000000                      .WORD           ;SPARE
 002000  000000                      .WORD           ;SPARE
 002002  000000             RP4HWC:  .WORD           ;RP04
 002004  000000             RSHWC:   .WORD           ;RS04

                            .SBTTL  DEVICE HANDLER OLD BASE ADDRESS
                            ;*      THIS TABLE GETS LOADED BY THE I/O RELOCATION ROUTINE
                            ;*      WITH THE BASE ADDRESS OF THE SOURCE DATA FOR THE
                            ;*      DEVICE THAT IS GOING TO TRANSFER THE DATA.
 002006  000000             RP3OLD:  .WORD           ;RP03
 002010  000000                      .WORD
 002012  000000             RKOLD:   .WORD           ;RK05
 002014  000000                      .WORD
 002016  000000                      .WORD           ;SPARE
 002020  000000                      .WORD           ;SPARE
 002022  000000                      .WORD
 002024  000000                      .WORD           ;SPARE
 002026  000000             RP4OLD:  .WORD           ;RP04
 002030  000000                      .WORD
 002032  000000             RSOLD:   .WORD           ;RS04
 002034  000000                      .WORD

                            .SBTTL  DEVICE HANDLER NEW BASE ADDRESSES
                            ;*      THIS TABLE GETS LOADED BY THE I/O RELOCATION ROUTINE
                            ;*      WITH THE BASE ADDRESS OF THE DESTINATION FOR THE
                            ;*      PARTICULAR DEVICE THAT IS GOING TO DO THE TRANSFER.
 002036  000000             RP3NWL:  .WORD           ;RP03
 002040  000000             RP3NWH:  .WORD
 002042  000000             RKNEWL:  .WORD           ;RK05
 002044  000000             RKNEWH:  .WORD
 002046  000000                      .WORD           ;SPARE
 002050  000000                      .WORD
 002052  000000                      .WORD           ;SPARE
 002054  000000                      .WORD
 002056  000000             RP4NWL:  .WORD           ;RP04
 002060  000000             RP4NWH:  .WORD
 002062  000000             RSNEWL:  .WORD           ;RS04
 002064  000000             RSNEWH:  .WORD

                            .SBTTL  DEVICE HANDLER UNIT NUMBER
                            ;*      THIS TABLE GETS LOADED BY THE I/O RELOCATION ROUTINE.
                            ;*      IT TELLS THE DEVICE HANDLER WHICH UNIT NUMBER IS
                            ;*      TO DO THE TRANSFER.
 002066  000000             RP3UNIT:.WORD            ;RP03
 002070  000000             RKUNIT:  .WORD           ;RK05
 002072  000000                      .WORD           ;SPARE
 002074  000000                      .WORD           ;SPARE
 002076  000000             RP4UNIT:.WORD            ;RP04
 002100  000000             RSUNIT:  .WORD           ;RS04

                            .SBTTL  ADDRESS OF THE DEVICE HANDLERS
                            ;*      THIS TABLE CONTAINS THE ADDRESS OF THE DEVICE HANDLER
                            ;*      ROUTINES. IT IS USED BY THE I/O RELOCATION ROUTINE
                            ;*      TO TRANSFER CONTROL TO THE DEVICE HANDLER.
 002102  041250             RP3HANA:.WORD   RP3DRV   ;RP03
 002104  041644             RKHANA:  .WORD   RKDRV    ;RK05
```

```
002106  000000                           .WORD             ;SPARE
002110  000000                           .WORD             ;SPARE
002112  042220                 RP4HANA:  .WORD   RP4DRV    ;RP04
002114  042614                 RSHANA:   .WORD   RSDRV     ;RS04

                               .SBTTL  DEVICE HANDLER DISK ADDRESS TABLE
                               ;*      THIS TABLE GETS LOADED BY THE DEVICE HANDLER WITH THE
                               ;*      DISK ADDRESS(SECTOR AND CYLINDER) OF THE CURRENT
                               ;*      TRANSFER.
002116  000000                 RP3HDA:   .WORD             ;RP03 DISK ADDRESS
002120  000000                 RP3HDC:   .WORD             ;RP03 DESIRED CYLINDER
002122  000000                 RKHDA:    .WORD             ;RK05 DISK ADDRESS
002124  000000                           .WORD             ;SPARE
002126  000000                 RP4HDA:   .WORD
002130  000000                 RP4HDC:   .WORD             ;RP04 DESIRED CYLINDER
002132  000000                 RSHDA:    .WORD             ;RS04 DISK ADDRESS

                               .SBTTL  DEVICE HANDLER FUNCTION TABLE
                               ;*      THIS TABLE GETS LOADED BY THE DEVICE HANDLERS
                               ;*      AND THE DEVICE SERVICE ROUTINES. IT TELLS THE ROUTINES
                               ;*      WHICH FUNCTION TO DO NEXT.
002134  000000                 RP3FUN:   .WORD             ;RP03
002136  000000                 RKFUN:    .WORD             ;RK05
002140  000000                           .WORD             ;SPARE
002142  000000                 RP4FUN:   .WORD             ;RP04
002144  000000                 RSFUN:    .WORD             ;RS04

                               .SBTTL  DEVICE HANDLER RETRY COUNT
                               ;*      THIS TABLE GETS LOADED BY THE DEVICE HANDLERS AND IS USED
                               ;*      BY THE DEVICE SERVICE ROUTINES. IF AN ERROR OCCURS
                               ;*      THE DEVICE SERVICE ROUTINE WILL RETRY THE FUNCTION UNTIL
                               ;*      THE BYTE IN THIS TABLE GOES TO ZERO. IT IS INITIALIZED
                               ;*      TO A -3.
002146     000                 RP3TRY:   .BYTE             ;RP03
002147     000                 RKTRY:    .BYTE             ;RK05
002150     000                           .BYTE             ;SPARE
002151     000                 RP4TRY:   .BYTE             ;RP04
002152     000                 RSTRY:    .BYTE             ;RS04
        002154                           .EVEN

                               .SBTTL  DEVICE REGISTER TABLES
                               ;*      THE FOLLOWING TABLES CONTAIN THE STANDARD ADDRESS FOR
                               ;*      THE DEVICES USED BY THIS PROGRAM. IF A DEVICE IS PLACED
                               ;*      AT A NON-STANDARD ADDRESS THE APPROPRIATE TABLE CAN BE
                               ;*      CHANGED AND THE PROGRAM WILL OPERATE THAT DEVICE.
                               ;*
                               ;*      EXCEPTION--SEE DOCUMENTATION FOR RP03 AND RP04 PROBLEMS.
                               .SBTTL  RP11/RP03 REGISTERS
002154  176710                 RP3DS:    .WORD   176710    ;DRIVE STATUS
002156  176712                 RP3ER:    .WORD   176712    ;ERROR REGISTER
002160  176714                 RP3CS:    .WORD   176714    ;CONTROL AND STATUS
002162  176716                 RP3WC:    .WORD   176716    ;WORD COUNT
002164  176720                 RP3BA:    .WORD   176720    ;BUS ADDRESS
002166  176724                 RP3DA:    .WORD   176724    ;DISK ADDRESS
002170  176722                 RP3DC:    .WORD   176722    ;DESIRED CYLINDER
002172  000254                 RP3VEC:   .WORD   254       ;INTERRUPT VECTOR
```

```
002174  000256          RP3PSW: .WORD   256     ;INTERRUPT VECTOR+2

                        .SBTTL  RK11/RK05 REGISTERS
002176  177400          RKDS:   .WORD   177400  ;DRIVE STATUS
002200  177402          RKER:   .WORD   177402  ;ERROR REGISTER
002202  177404          RKCS:   .WORD   177404  ;CONTROL AND STATUS
002204  177406          RKWC:   .WORD   177406  ;WORD COUNT
002206  177410          RKBA:   .WORD   177410  ;BUS ADDRESS
002210  177412          RKDA:   .WORD   177412  ;DISK ADDRESS
002212  000220          RKVEC:  .WORD   220     ;INTERRUPT VECTOR
0[illegible]2214  000222          RKPSW:  .WORD   222     ;INTERRUPT VECTOR+2

                        .SBTTL  RH11/RP04 REGISTERS
002216  176700          RP4CS1: .WORD   176700  ;CONTROL AND STATUS #1
0[illegible]20  176702          RP4WC:  .WORD   176702  ;WORD COUNT
0[illegible]22  176704          RP4BA:  .WORD   176704  ;BUS ADDRESS
0[illegible]24  176700          RP4BAE: .WORD   176700  ;CONTROL AND STATUS #1
0[illegible]26  176706          RP4DA:  .WORD   176706  ;DISK ADDRESS
0[illegible]30  176710          RP4CS2: .WORD   176710  ;CONTROL AND STATUS #2
0[illegible]32  176752          RP4CS3: .WORD   176752  ;CONTROL AND STATUS #3
0[illegible]34  176712          RP4DS:  .WORD   176712  ;DRIVE STATUS
0[illegible]36  176714          RP4ER1: .WORD   176714  ;ERROR REG #1
0[illegible]240  176734          RP4DC:  .WORD   176734  ;DESIRED CYLINDER
0[illegible]242  176740          RP4ER2: .WORD   176740  ;ERROR REG #2
0[illegible]244  176742          RP4ER3: .WORD   176742  ;ERROR REG #3
0[illegible]246  176736          RPCC:   .WORD   176736  ;CURRENT CYLINDER
002250  176732          RP4OF:  .WORD   176732  ;OFFSET REGISTER
002252  000254          RP4VEC: .WORD   254     ;INTERRUPT VECTOR
002254  000256          RP4PSW: .WORD   256     ;INTERRUPT VECTOR+2

                        .SBTTL  RH11/RS04 REGISTERS
002256  172040          RSCS1:  .WORD   172040  ;CONTROL AND STATUS #1
002260  172042          RSWC:   .WORD   172042  ;WORD COUNT
002262  172044          RSBA:   .WORD   172044  ;BUS ADDRESS
002264  172040          RSBAE:  .WORD   172040  ;CONTROL AND STATUS #1
002266  172046          RSDA:   .WORD   172046  ;DISK ADDRESS
002270  172050          RSCS2:  .WORD   172050  ;CONTROL AND STATUS #2
002272  172072          RSCS3:  .WORD   172072  ;CONTROL AND STATUS #3
002274  172052          RSDS:   .WORD   172052  ;DRIVE STATUS
002276  172054          RSER:   .WORD   172054  ;ERROR REG
002300  000204          RSVEC:  .WORD   204     ;INTERRUPT VECTOR
002302  000206          RSPSW:  .WORD   206     ;INTERRUPT VECTOR+2

                        .SBTTL  UNIBUS EXERCISER REGISTER ADDRESS TABLE
                        ;*      THIS TABLE IS ASSEMBLED FOR UBE #0. IF THE UBE
                        ;*      ADDRESSES ARE CUT FOR OTHER THAN UNIT #0, THE PROGRAM
                        ;*      WILL CHANGE THIS TABLE. THE PROGRAM LOOKS FOR A
                        ;*      UBE AT ADDRESSES 770000, 770020, 770040, AND 770060.
002304  170002          UBETBL: .WORD   UBECC   ;CYCLE COUNT
002306  170004                  .WORD   UBEBA   ;BUS ADDRESS REG
002310  170016                  .WORD   UBECR2  ;CONTROL REGISTER #2
002312  170006                  .WORD   UBECR1  ;CONTROL REGISTER #1
002314  170010                  .WORD   UBECLR  ;UBE CLEAR ADDRESS
002316  000510                  .WORD   UBEVEC  ;INTERRUPT VECTOR
002320  000512                  .WORD   UBEVEC+2        ;INTERRUPT VECTOR +2
```

```
                                .SBTTL  MASS BUS TESTER REGISTER ADDRESSES
                                ;*      THE PROGRAM IS ASSEMBLED WITH ADDRESSES FOR A MBT
                                ;*      AT 760100. IF THE MBT IS AT ANOTHER ADDRESS THE PROGRAM
                                ;*      WILL CHANGE THIS TABLE. THE PROGRAM LOOKS FOR A MBT
                                ;*      AT ADDRESSES 760100, 760200, 760300, AND 760400.
002322  160100                  MBTTBL: .WORD   MBTCS1  ;CONTROL AND STATUS #1
002324  160102                          .WORD   MBTWC   ;WORD COUNT
002326  160104                          .WORD   MBTBA   ;BUS ADDRESS
002330  160100                          .WORD   MBTCS1  ;CONTROL AND STATUS #1
002332  160106                          .WORD   MBTMR2  ;MAINTENANCE REGISTER #2
002334  160110                          .WORD   MBTCS2  ;CONTROL REGISTER #2
002336  160112                          .WORD   MBTST   ;STATUS REGISTER
002340  160114                          .WORD   MBTER   ;ERROR REGISTER
002342  160176                          .WORD   MBTCS3  ;CONTROL REGISTER #3
002344  000774                          .WORD   MBTVEC  ;INTERRUPT VECTOR
002346  000776                          .WORD   MBTPSW  ;INTERRUPT VECTOR+2
002350  160126                          .WORD   MBTDT   ;DRIVE TYPE REGISTER
002352  160200                  MBTN2:  .WORD   160200  ;MASS BUS TESTER #2
002354  160300                  MBTN3:  .WORD   160300  ;MASS BUS TESTER #3
002356  160400                  MBTN4:  .WORD   160400  ;MASS BUS TESTER #4

                                .SBTTL  MED TEST TABLES
002360  000000                  TLOC1:  .WORD   0
002362  000000                  PSWHOL: .WORD   0
002364  000000                  TABREG: .WORD   0
002366  000000                  TABEND: .WORD   0
002370  000040                  STGBLK: .BLKW   40
002470  000000                  TLOC2:  .WORD   0
002472  000000                  MEDTP0: .WORD   0
002474  000000                  MEDTP1: .WORD   0
002476  000000                  MEDTP2: .WORD   0
002500  000000                  MEDTP3: .WORD   0


                                ;*
                                ;*      TABLE II
                                ;*
                                ;*      FOLLOWING IS A TABLE OF INTERNAL REGISTER OPERATION CODES
                                ;*      USED FOR TESTING THE MED INSTRUCTION.  LABELS CORRESPOND
                                ;*      TO REGISTER NAMES, THE HIGH BYTE IS THE READ OPERATION
                                ;*      CODE, THE LOW BYTE THE WRITE CODE.
                                ;*              NOTE:  WHEN ADDING OR DELETING
                                ;*              ENTRIES IN THIS TABLE, CHECK DUAL
                                ;*              ADDRESING TEST TO SEE THAT THE "SCRATCH
                                ;*              PAD LIMITS" ARE MAINTAINED.
                                ;*

002502                          TBL2:


002502                          ASP1:                           ;A SCRATCH PAD - LO
002502     201     001          R1A:    .BYTE   201,001         ;LOBYTE, HIBYTE=WRITE CODE, READ CODE
002504     202     002          R2A:    .BYTE   202,002
002506     203     003          R3A:    .BYTE   203,003
002510     204     004          R4A:    .BYTE   204,004
002512     205     005          R5A:    .BYTE   205,005
```

```
002514   206   006      R6A:     .BYTE   206,006
002516   210   010      FAC3.0:  .BYTE   210,010
002520   211   011      FAC3.1:  .BYTE   211,011
002522   212   012      FAC3.2:  .BYTE   212,012
002524   213   013      FAC3.3:  .BYTE   213,013
002526   214   014      FAC3.4:  .BYTE   214,014
002530   215   015      FAC3.5:  .BYTE   215,015
002532   216   016      UR6A:    .BYTE   216,016
002534   217   017      FDST3:   .BYTE   217,017
002536   220   020      WCSA.0:  .BYTE   220,020         ;A SCRATCH PAD-HI
002540   221   021      WCSA.1:  .BYTE   221,021
002542   222   022      GNWHAM:  .BYTE   222,022
002544   223   023      CNSTSW:  .BYTE   223,023
002546   224   024      RT1A:    .BYTE   224,024
002550   225   025      RT2A:    .BYTE   225,025
002552   226   026      CNSSW:   .BYTE   226,026
002554   227   027      CNSCDR:  .BYTE   227,027
002556   230   030      FAC1.0:  .BYTE   230,030
002560   231   031      FAC1.1:  .BYTE   231,031
002562   232   032      FAC1.2:  .BYTE   232,032
002564   233   033      FAC1.3:  .BYTE   233,033
002566   234   034      FAC1.4:  .BYTE   234,034
002570   235   035      FAC1.5:  .BYTE   235,035
002572   236   036      FPSHI:   .BYTE   236,036
002574   237   037      ASP2:    FDST1:  .BYTE   237,037


002576                  BSP1:
002576   241   041      R1B:     .BYTE   241,041         ;B SCRATCH PAD - LO
002600   242   042      R2B:     .BYTE   242,042
002602   243   043      R3B:     .BYTE   243,043
002604   244   044      R4B:     .BYTE   244,044
002606   245   045      R5B:     .BYTE   245,045
002610   246   046      R6B:     .BYTE   246,046
002612   250   050      FAC2.0:  .BYTE   250,050
002614   251   051      FAC2.1:  .BYTE   251,051
002616   252   052      FAC2.2:  .BYTE   252,052
002620   253   053      FAC2.3:  .BYTE   253,053
002622   254   054      FAC2.4:  .BYTE   254,054
002624   255   055      FAC2.5:  .BYTE   255,055
002626   256   056      UR6B:    .BYTE   256,056
002630   257   057      FDST2:   .BYTE   257,057
002632   260   060      WCSB.0:  .BYTE   260,060         ;B SCRATCH PAD - HI
002634   261   061      WCSB.1:  .BYTE   261,061
002636   262   062      WCSADR:  .BYTE   262,062
002640   263   063      RZERO:   .BYTE   263,063
002642   264   064      RT1B:    .BYTE   264,064
002644   265   065      RT2B:    .BYTE   265,065
002646   266   066      RVECT:   .BYTE   266,066
002650   270   070      FAC0.0:  .BYTE   270,070
002652   272   072      FAC0.1:  .BYTE   272,072
002654   273   073      FAC0.2:  .BYTE   273,073
002656   274   074      FAC0.4:  .BYTE   274,074
002660   275   075      FAC0.5:  .BYTE   275,075
002662   276   076      FEA:     .BYTE   276,076
002664   277   077      BSP2:    FDST0:  .BYTE   277,077
```

```


        002666                          CSP1:
        002666    300      100          LJAM:    .BYTE   300,100         ;C SCRATCH PAD
        002670    301      101          LSERV:   .BYTE   301,101
        002672    302      102          LPBA:    .BYTE   302,102
        002674    303      103          LCUA:    .BYTE   303,103
        002676    304      104          LFGIN:   .BYTE   304,104
        002700    305      105          LWHAM:   .BYTE   305,105
        002702    307      107          LTAG:    .BYTE   307,107
        002704    310      110          CNSCO:   .BYTE   310,110
        002706    311      111          CNSC1:   .BYTE   311,111
        002710    312      112          CNSC2:   .BYTE   312,112
        002712    313      113          CST200:  .BYTE   313,113
        002714    316      116          CSP2:    CNSTO:  .BYTE   316,116

        002716  000000                           .WORD   0               ;END OF ADDRESS TABLE = 0


                                        ;*
                                        ;*       TABLE IV
                                        ;*
                                        ;*       THE LIST BELOW CONTAINS THOSE OPERATION CODES
                                        ;*       CORRESPONDING TO THE INTERNAL REGISTERS WHICH MUST
                                        ;*       BE TESTED SEPERATELY BECAUSE THEY ARE READ-ONLY,
                                        ;*       WRITE-ONLY, OR USED IN MACRO CODE EXECUTION, ETC. . .

        002720                          TBL4:
        002720    200      000          R0A:     .BYTE   200,000         ;LOBYTE,HYBYTE - WRITE CODE, READ CODE
        002722    207      007          R7A:     .BYTE   207,007
        002724    240      040          R0B:     .BYTE   240,040
        002726    247      047          R7B:     .BYTE   247,047
        002730    314      114          CNST2:   .BYTE   314,114
        002732    317      117          CNST1:   .BYTE   317,117
                                        ;*       TABLE V
                                        ;*
        002734                          TBL5:

        002734    306                   LCDTA:   .BYTE   306             ;THIS TABLE CONTAINS THE OPERATION
        002735    106                            .BYTE   106             ;CODES OF THOSE INTERNAL REGISTERS
        002736    315                   MD:      .BYTE   315             ;WHICH MUST BE TESTED USING THE
        002737    115                            .BYTE   115             ;MICROBREAK REGISTER.  THEIR
        002740    267                   CNSCTL:  .BYTE   267             ;ASSOCIATED MICRO-ADDRESSES ARE IN
        002741    067                            .BYTE   067             ;THE NEXT TABLE
        002742    140                   JAM:     .BYTE   140
        002743    141                   SERV:    .BYTE   141             ;THESE MICROADDRESSES ARE THE ENTRY
        002744    142                   PBA:     .BYTE   142             ;POINTS INTO THE MICROCODE FOR THE SERVICE
        002745    143                   CUA:     .BYTE   143             ;OF THE INTERNAL REGISTERS
        002746    344                   FLAG:    .BYTE   344
        002747    144                            .BYTE   144
        002750    345                   DREG:    .BYTE   345
        002751    146                   REV:     .BYTE   146
        002752    346                   SREG:    .BYTE   346
        002753    147                   COUNT:   .BYTE   147
        002754    347                   NUA:     .BYTE   347
        002755    351                   RES:     .BYTE   351
```

```
002756      152                  DCS0:   .BYTE   152
002757      352                  INIT:   .BYTE   352
002760      153                  DCS1:   .BYTE   153
002761      000                          .BYTE   0               ;0 BYTE TERMINATES THE PROGRAM

                                         .EVEN
                                 ;*      TABLE VI
                                 ;*
002762                           TBL6:

002762  003330                   ULCDTA: .WORD   3330            ;THIS TABLE CONTAINS THE MICRO-ADDRESSES
002764  003150                           .WORD   3150            ;WHICH ARE LOADED INTO THE MICROBREAK
002766  003375                   UMD:    .WORD   3375            ;REG. TO TEST THE OPERATION CODES
002770  003271                           .WORD   3271            ;CONTAINED IN THE PRECEEDING TABLE.
002772  003240                   UCNSCTL:.WORD   3240
002774  003224                           .WORD   3224
002776  003160                   UJAM:   .WORD   3160            ;THESE MICROADDRESSES SHOULD BE
003000  003161                   USERV:  .WORD   3161            ;CHANGED (IF NECESSARY) IF THE BASE MACHINE
003002  003170                   UPBA:   .WORD   3170            ;MICROCODE SHIFTS (MICROCODE SERVICING
003004  003171                   UCUA:   .WORD   3171            ;THE INTERNAL REGISTERS).
003006  003344                   UFLAG:  .WORD   3344
003010  003320                           .WORD   3320
003012  003345                   UDREG:  .WORD   3345
003014  003340                   UREV:   .WORD   3340
003016  003350                   USREG:  .WORD   3350
003020  003341                   UCOUNT: .WORD   3341
003022  003351                   UNUA:   .WORD   3351
003024  003355                   URES:   .WORD   3355
003026  003720                   UDCS0:  .WORD   3720
003030  003724                   UINIT:  .WORD   3724
003032  003721                   UDCS1:  .WORD   3721

                                 ;*      TABLE VII
                                 ;*
                                 ;*      THIS TABLE HOLDS THE OPERATION CODES AND THE CONSTANT
                                 ;*      VALUE EXPECTED FOR CERTAIN INTERNAL REGISTERS.
003034                           TBL7:

003034  000100  077600           CLJAM:  .WORD   100,77600
003040  000101  000010           CLSERV: .WORD   101,10
003044  000102  020000           CLPBA:  .WORD   102,20000
003050  000103  000004           CLCUA:  .WORD   103,4
003054  000104  050000           CLFGIN: .WORD   104,50000
003060  000105  054000           CLWHAM: .WORD   105,54000
003064  000107  024000           CLTAG:  .WORD   107,24000
003070  000110  177400           CCNSC0: .WORD   110,177400
003074  000111  177600           CCNSC1: .WORD   111,177600
003100  000112  100000           CCNSC2: .WORD   112,100000
003104  000113  000200           CCST200:.WORD   113,200
003110  000114  000002           CCNST2: .WORD   114,2
003114  000116  000000           CCNST0: .WORD   116,0
003120  000117  000001           CCNST1: .WORD   117,1
003124  000000                           .WORD   0
```

```
                                   .SBTTL  ERROR POINTER TABLE

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

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


 003126                            $ERRTB:
                                           ;ITEM 1
 003126  056411                            EM1             ;UNEXPECTED TRAP TO 4
 003130  056436                            DH1             ;VIRTPC  PHYSC      PSW  CPUERR
 003132  056502                            DT1             ;VADR,VADR,REPPSW,REPREG
 003134  056475                            DF1             ;0,1,0,0,0
                                           ;ITEM 2
 003136  056514                            EM2             ;UNEXPECTED TRAP TO 10
 003140  056542                            DH2             ;VIRTPC  PHYSPC  PSW
 003142  056570                            DT2             ;VADR,VADR,REPPSW
 003144  056475                            DF1
                                           ;ITEM 3
 003146  056600                            EM3             ;UNEXPECTED TRAP TO 250(MGMT)
 003150  056635                            DH3             ;VIRTPC  PHYSPC    PSW    MMR0    MMR2
 003152  056704                            DT3             ;VADR,VADR,REPPSW,REPMR0,REPMR2
 003154  056475                            DF1
                                           ;ITEM 4
 003156  056720                            EM4             ;UNEXPECTED TRAP TO 114
 003160  056747                            DH4             ;VIRTPC  PHYSPC   PSW     MEMERRREG
 003162  057014                            DT4             ;VADR,VADR,REPPSW,REPREG
 003164  057026                            DF4             ;0,1,0,0
                                           ;ITEM 5
 003166  057032                            EM5             ;PARITY ERROR DURING DATA CHECK
 003170  057071                            DH5             ;SRCADR DSTADR MEM ERR REG
 003172  057130                            DT5             ;SRCADR,PA1500,REPREG
 003174  057125                            DF5
                                           ;ITEM 6
 003176  057140                            EM6             ;ERROR DURING CHECK OF RELOCATED DATA
 003200  057210                            DH6             ;SRCADR  DSTADR
 003202  057230                            DT6             ;$TMP0,PA1500
 003204  057026                            DF4
                                           ;ITEM 7
 003206  000000                            0               ;DEVICE ERROR
 003210  000000                            0
 003212  000000                            0
 003214  000000                            0
                                           ;ITEM 10
 003216  057236                            EM10            ;ERROR DURING DATA CHECK-RELOC WAS BY I/O
 003220  057307                            DH10            ;SRCADR    DSTADR   DEVICE THAT DID XFER
 003222  057362                            DT10            ;$TMP0,VADR,$TMP2,$TMP3
 003224  057356                            DF10            ;0,1,3,0
                                           ;ITEM 11
```

```
003226  057637          EM14            ;FLOATING POINT ERROR
003230  057374          DH11            ;                DATA1           DATA2
003232  057472          DT11            ;FLTMP0,FREG2,FLTMP1,FREG3
003234  057465          DF11            ;5,0,5,0
                        ;ITEM 12
003236  057504          EM12            ;UNIBUS EXERCISOR NON-EXISTANT MEMORY
003240  057542          DH12            ;PHYSICAL ADDRESS
003242  057560          DT12            ;PA1500
003244  057556          DF12            ;2
                        ;ITEM 13
003246  057564          EM13            ;MASS BUS TESTER NON-EXISTANT MEMORY
003250  057622          DH13            ;PHYSICAL ADDRESS
003252  057560          DT12
003254  057556          DF12
                        ;ITEM 14
003256  057637          EM14            ;FLOATING POINT ERROR
003260  057664          DH14            ;        DATA1           DATA2
003262  057704          DT14            ;FTMP4,FREG2,FTMP6,FREG3
003264  057716          DF14            ;4,0,4,0
                        ;ITEM 15
003266  057722          EM15            ;DEVICE HUNG
003270  000000          0
003272  000000          0
003274  000000          0
```

```
                        .SBTTL  PROGRAM INITIALIZATION

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

```
003276  012706  001200          START:  MOV     #KERSTK,SP      ;SET KERNEL STACK PTR
003302  012737  017654  001566          MOV     #17654,@#$HINUM         ;INITIALIZE RANDOM NUM GEN
003310  012737  123456  001564          MOV     #123456,@#$LONUM

                                ;DETERMINE HOW PROGRAM WAS LOADED AND WHAT MODE (IF ACT11)
                                ;AND SET MEMORY PROTECTION.
003316  005037  001546                  CLR     @#QV                    ;SET NOT QV NOR AA MODE
003322  005027                          CLR     (PC)+                   ;SET NOT XXDP
003324     000                  XXDP:   .BYTE   0                       ;XXDP INDICATOR
003325     000                  XXDPC:  .BYTE   0                       ;XXDP CHAIN MODE INDICATOR
003326  005027                          CLR     (PC)+                   ;CLEAR MEMORY PROTECTION LIMIT
003330  000000                  PROT:   .WORD   0                       ;WILL CONTAIN MEM PROT LIMIT
003332  005737  041234                  TST     @#$ENDAD+4              ;BRANCH IF NOT QV
003336  100003                          BPL     1$
003340  110637  001546                  MOVB    SP,@#QV                 ;SET ACT11 QV MODE
003344  000414                          BR      3$

003346  001003                  1$:     BNE     2$
003350  110637  001547                  MOVB    SP,@#AA                 ;SET ACT11 AA MODE
003354  000410                          BR      3$

003356  113737  000041  003324  2$:     MOVB    @#41,@#XXDP             ;GET LOAD MEDIA
003364  005737  000042                  TST     @#42                    ;BRANCH IF NOT IN CHAIN MODE
003370  001402                          BEQ     3$
003372  110637  003325                  MOVB    SP,@#XXDPC              ;SET CHAIN MODE INDICATOR

                                ;SET MEMORY PROTECTION LIMITS
003376  005737  001546          3$:     TST     @#QV                    ;BRANCH IF QV OR AA
003402  001006                          BNE     MEMSIZ
003404  005737  003324                  TST     @#XXDP                  ;BRANCH IF NOT VIA XXDP
003410  001403                          BEQ     MEMSIZ
003412  012737  005700  003330          MOV     #5700,@#PROT            ;PROTECT XXDP MONITOR
003420  005000                  MEMSIZ: CLR     R0                      ;INITIALIZE R0 WITH FIRST ADDR.
003422  012737  003434  000004          MOV     #11$,@#ERRVEC           ;SET TIMEOUT VECTOR TO 11$
003430  005720                  10$:    TST     (R0)+                   ;DOES ADDR. IN R0 EXSIST?
003432  000776                          BR      10$                     ;CONTINUE UNTIL AN ADDR. TIMES OUT

003434  022626                  11$:    CMP     (SP)+,(SP)+             ;CLEAN UP STACK
003436  012737  055064  000004          MOV     #ERPRT,@#ERRVEC         ;RESTORE TIMEOUT SERVICE ROUTINE
003444  162700  000004                  SUB     #4,R0                   ;GET ADDR. OF LAST MEM. LOC.
003450  010037  001560                  MOV     R0,@#LSTMEM             ;SAVE IT IN "LSTMEM"
003454  163737  003330  001560          SUB     @#PROT,@#LSTMEM         ;SET PROTECTION
003462  012737  057740  001556          MOV     #ENDTAG+2,@#FRSTMEM     ;SET FIRST RELOCATION ADDRESS


003470  012706  001200                  MOV     #KERSTK,SP              ;SET STACK PTR
003474  005037  001200                  CLR     @#$PASS                 ;CLEAR PASS COUNT
003500  105037  001545                  CLRB    @#MMON                  ;SET MEM MGMT ON IND=NOT ON
003504  012737  000600  001562          MOV     #600,@#NEXPAR           ;SET FIRST 'PAR' VALUE
003512  005737  003330                  TST     @#PROT
003516  001403                          BEQ     1$
003520  012737  001600  001562          MOV     #1600,@#NEXPAR
003526                          1$:
003526  012700  000027                  MOV     #27,R0                  ;SET SOB COUNT
003532  005001                          CLR     R1                      ;SETUP INDEX
003534  005061  001642          2$:     CLR     MTICKS(R1)              ;CLEAR TABLES
```

```
003540  062701  000002                    ADD     #2,R1
003544  077005                            SOB     R0,2$                         ;CONTINUE
003546  012700  000010                    MOV     #10,R0                        ;SET SOB COUNT
003552  012701  001752                    MOV     #RP3HSTAT,R1                  ;GET ADDRESS OF HANDLER STAT
003556  012721  000200            3$:     MOV     #200,(R1)+                    ;INITIALIZE STATUS TABLE
003562  077003                            SOB     R0,3$                         ;CONTINUE
003564  012737  000060  001574            MOV     #60,@#SUBPASS                 ;INIT SUBPASS TO ASCII 0
003572  012700  051236                    MOV     #TIMEBUF,R0                   ;GET ADR OF TIME BUFFER
003576  012701  000012                    MOV     #12,R1                        ;SET SOB COUNT
003602  112720  000060            4$:     MOVB    #60,(R0)+                     ;INIT TIME BUFFER
003606  077103                            SOB     R1,4$
003610  105040                            CLRB    -(R0)                         ;INSERT TERMINATOR
003612  112737  000072  051241            MOVB    #72,@#TIMEBUF+3               ;INSERT COLON
003620  112737  000072  051244            MOVB    #72,@#TIMEBUF+6
003626  005037  001624                    CLR     @#MXMMLO                      ;CLEAR MEM. SIZE INDICATOR
                                  .SBTTL  INITIALIZE THE COMMON TAGS
                                  ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
003632  012706  001200                    MOV     #$CMTAG,R6            ;;FIRST LOCATION TO BE CLEARED
003636  005026                            CLR     (R6)+                 ;;CLEAR MEMORY LOCATION
003640  022706  001242                    CMP     #SWR,R6 ;;DONE?
003644  001374                            BNE     .-6                   ;;LOOP BACK IF NO
003646  012706  001200                    MOV     #STACK,SP             ;;SETUP THE STACK POINTER
                                  ;;INITIALIZE A FEW VECTORS
003652  012737  046720  000020            MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
003660  012737  000340  000022            MOV     #340,@#IOTVEC+2 ;;LEVEL 7
003666  012737  047160  000030            MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
003674  012737  000340  000032            MOV     #340,@#EMTVEC+2 ;;LEVEL 7
003702  012737  053566  000034            MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
003710  012737  000340  000036            MOV     #340,@#TRAPVEC+2;LEVEL 7
003716  012737  053376  000024            MOV     #$PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
003724  012737  000340  000026            MOV     #340,@#PWRVEC+2 ;;LEVEL 7
003732  016767  035136  035126            MOV     $ENDCT,$EOPCT         ;;SETUP END-OF-PROGRAM COUNTER
003740  005067  175360                    CLR     $TIMES                ;;INITIALIZE NUMBER OF ITERATIONS
003744  005067  175356                    CLR     $ESCAPE               ;;CLEAR THE ESCAPE ON ERROR ADDRESS
003750  112767  000001  175241            MOVB    #1,$ERMAX             ;;ALLOW ONE ERROR PER TEST
003756  012767  003756  175224            MOV     #.,$LPADR             ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
003764  012767  003764  175220            MOV     #.,$LPERR             ;;SETUP THE ERROR LOOP ADDRESS
                                  ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                  ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
003772  013746  000004                    MOV     @#ERRVEC,-(SP)        ;;SAVE ERROR VECTOR
003776  012737  004032  000004            MOV     #64$,@#ERRVEC         ;;SET UP ERROR VECTOR
004004  012767  177570  175230            MOV     #DSWR,SWR             ;;SETUP FOR A HARDWARE SWICH REGISTER
004012  012767  177570  175224            MOV     #DDISP,DISPLAY        ;;AND A HARDWARE DISPLAY REGISTER
004020  022777  177777  175214            CMP     #-1,@SWR              ;;TRY TO REFERENCE HARDWARE SWR
004026  001012                            BNE     66$                   ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
004030  000403                            BR      65$                   ;;BRANCH IF NO TIMEOUT
004032  012716  004040            64$:    MOV     #65$,(SP)             ;;SET UP FOR TRAP RETURN
004036  000002                            RTI
004040  012767  000176  175174    65$:    MOV     #SWREG,SWR            ;;POINT TO SOFTWARE SWR
004046  012767  000174  175170            MOV     #DISPREG,DISPLAY
004054  012637  000004            66$:    MOV     (SP)+,@#ERRVEC        ;;RESTORE ERROR VECTOR

004060  012700  002002                    MOV     #2002,R0                      ;TURN OFF WCS BY INITIALIZING
004064  076600                            MED                                   ;WHAMI REG. WITH AN INIT.
004066  000352                            WRINIT                                ;SO RESERVED INSTRUCTIONS CAN BE
```

```
                                                                        ;TESTED.

                                        ;CLEAR PROGRAM INDICATORS
004070  052777  000100  175150                  BIS     #100,@$TKS      ;SET IE BIT IN KEYBOARD STATUS REG
004076  012737  054152  000060                  MOV     #TKISR,@#TKVEC  ;SETUP KEYBOARD VECTOR
004104  012737  000200  000062                  MOV     #PR4,@#TKVEC+2
004112  012737  054364  000064                  MOV     #TPISR,@#TPVEC
004120  012737  000200  000066                  MOV     #PR4,@#TPVEC+2
004126  005037  001530                          CLR     @#NOTYPE        ;CLEAR 'NO TYPING' INDICATOR

                                        ;THE BELOW ROUTINE ASCERTAINS WHICH CP & CP OPTIONS THE PROGRAM IS RUN-
                                        ;NING ON AND SETS AN INDICATOR IN OPT.CP ACCORDINGLY.
004132  012737  000006  000004  CPCHK:  MOV     #ERRVEC+2,@#ERRVEC      ;SET UP ERROR TRAP TO RETURN
004140  012737  000002  000006          MOV     #2,@#ERRVEC+2           ;RTI OPCODE = 2
004146  012737  000012  000010          MOV     #RESVEC+2,@#RESVEC      ;AND ALSO RESERVED INST TRAP
004154  012737  000002  000012          MOV     #2,@#RESVEC+2
004162  012702  160000                  MOV     #160000,R2              ;SET PROCESSER'S NON-OPTION BITS
                                                                        ;EXCEPT FOR LKOPT
004166  000261                          SEC
004170  005737  177546                  TST     @#LKS                   ;BRANCH IF NO KW11-L
004174  103402                          BCS     7$                      ;OR IF IT'S NOT WORKING
004176  052702  001000                  BIS     #LKOPT,R2               ;SET OPTION INDICATOR
004202  000261                  7$:     SEC
004204  005777  175042                  TST     @$TPS                   ;BRANCH IF NO CONSOLE TTY
004210  103402                          BCS     9$
004212  052702  000400                  BIS     #TTOPT,R2
004216  005003                  9$:     CLR     R3
004220  000261                          SEC
004222  005737  170000                  TST     @#UBEDB         ;IS UBE1 THERE?
004226  103410                          BCS     12$             ;BRANCH IF NO
004230  105037  170006                  CLRB    @#UBECR1        ;IS THIS A TESTER OR EXERCISOR?
004234  105737  170006                  TSTB    @#UBECR1
004240  100045                          BPL     15$             ;BRANCH IF TESTER
004242  052702  000200          16$:    BIS     #UBEOPT,R2      ;SET INDICATOR
004246  000425                          BR      17$
004250  000261                  12$:    SEC
004252  005737  170020                  TST     @#UBEDB+20              ;IS UBE2 THERE?
004256  103403                          BCS     13$             ;BRANCH IF NO
004260  012703  000020                  MOV     #20,R3          ;SET OFFSET IN R3
004264  000766                          BR      16$
004266  000261                  13$:    SEC
004270  005737  170040                  TST     @#UBEDB+40      ;IS UBE3 THERE?
004274  103403                          BCS     14$             ;BRANCH IF NO
004276  012703  000040                  MOV     #40,R3          ;PUT OFFSET IN R3
004302  000757                          BR      16$
004304  000261                  14$:    SEC
004306  005737  170060                  TST     @#UBEDB+60      ;IS UBE4 THERE?
004312  103420                          BCS     15$             ;BRANCH IF NO
004314  012703  000060                  MOV     #60,R3          ;PUT OFFSET IN R3
004320  000750                          BR      16$
004322  005227  177777          17$:    INC     #-1
004326  001012                          BNE     15$
004330  012704  002304                  MOV     #UBETBL,R4      ;GET ADDRESS OF UBE TABLE
004334  012705  000005                  MOV     #5,R5           ;SET SOB COUNT
004340  060324                  18$:    ADD     R3,(R4)+        ;ADJUST UBE TABLE ENTRIES
004342  077502                          SOB     R5,18$          ;CONTINUE
```

```
004344  006003                           ROR     R3
004346  006003                           ROR     R3              ;ADJUST OFFSET FOR UBE VECTOR
004350  060324                           ADD     R3,(R4)+        ;ADJUST UBEVEC ENTRY
004352  060314                           ADD     R3,(R4)         ;ADJUST UBEVEC PSW ENTRY
004354  005003                   15$:    CLR     R3                      ;INIT R3
004356  000261                           SEC
004360  005777  175736                   TST     @MBTTBL         ;IS MASS BUS TESTER THERE?
004364  103403                           BCS     20$             ;BRANCH IF NO
004366  052702  002000           21$:    BIS     #MBTOPT,R2      ;SET OPTION AVAILABLE
004372  000422                           BR      24$
004374  005777  175752           20$:    TST     @MBTN2          ;IS MBT2 THERE?
004400  103403                           BCS     22$             ;BRANCH IF NO
004402  012703  000100                   MOV     #100,R3         ;SETUP R3
004406  000767                           BR      21$
004410  005777  175740           22$:    TST     @MBTN3          ;IS MBT3 THERE?
004414  103403                           BCS     23$             ;BRANCH IF NO
004416  012703  000200                   MOV     #200,R3
004422  000761                           BR      21$
004424  005777  175726           23$:    TST     @MBTN4          ;IS MBT4 THERE?
004430  103427                           BCS     30$             ;BRANCH IF NO
004432  012703  000300                   MOV     #300,R3
004436  000753                           BR      21$
004440  005227  177777           24$:    INC     #-1
004444  001021                           BNE     30$
004446  012704  002322                   MOV     #MBTTBL,R4      ;GET ADDRESS OF MBT TABLE
004452  012705  000011                   MOV     #11,R5          ;SET SOB COUN
004456  060324                   25$:    ADD     R3,(R4)+        ;ADJUST MBT TABLE
004460  077502                           SOB     R5,25$          ;CONTINUE
004462  060337  002350                   ADD     R3,@#MBTTBL+26  ;ADJUST DRIVE TYPE ADDRESS
004466  112777  000007  175640           MOVB    #7,@MBTTBL+12
004474  122777  000040  175646           CMPB    #40,@MBTTBL+26  ;IS THIS REALLY A MBT?
004502  001402                           BEQ     30$             ;BRANCH IF YES
004504  042702  002000                   BIC     #MBTOPT,R2      ;CLEAR OPTION AVAILABLE BIT
004510  012737  055064  000004   30$:    MOV     #ERRPRT,@#ERRVEC ;RESTORE ERROR TRAP
004516  012737  054774  000010           MOV     #RESERR,@#RESVEC;AND ALSO RESERVED INST TRAP
004524  010237  001532                   MOV     R2,@#OPT.CP     ;LOAD INDICATOR
                                 .SBTTL  TYPE PROGRAM NAME
                                 ;;TYPE THE NAME OF THE PROGRAM IF FIRST PASS
004530  005227  177777                   INC     #-1             ;;FIRST TIME?
004534  001037                           BNE     64$             ;;BRANCH IF NO
004536  022737  041230  000042           CMP     #SENDAD,@#42    ;;ACT-11?
004544  001433                           BEQ     64$             ;;BRANCH IF YES
004546  104401  004554                   TYPE    ,65$            ;;TYPE ASCIZ STRING
004552  000430                           BR      64$             ;;GET OVER THE ASCIZ
                                 ,65$:   .ASCIZ  <CRLF>"MAINDEC-11-DQKDC-A...PDP 11/6X CPU EXERCISER"<CRLF>
004634                           64$:
                                 ;;************************************************************
                                 .SBTTL  SYSTEM SIZER
                                 ;       THIS ROUTINE DETERMINES WHAT DRIVES ARE AVAILABLE ON
                                 ;       THE FOLLOWING DEVICES: RK05, RP03, RP04, AND RS04. THE
                                 ;       INFORMATION IS STORED IN THE TABLE "SYSSIZE" IN THE FOLLOWING FORMAT:
                                 ;               A. EACH DEVICE IS ASSIGNED A WORD
                                 ;               B. THE LOW BYTE OF THIS WORD INDICATES WHICH DRIVES ARE AVAILABLE
                                 ;               C. THE HIGH BYTE INDICATES WHICH DRIVES HAVE BEEN USED
                                 ;                  BY THE RELOCATION ROUTINE.
                                 ;;************************************************************
```

```
004634  012737  004746  000004  SIZE:   MOV     #21$,@#ERRVEC           ;SETUP TIMEOUT VECTOR
004642  005037  001304                  CLR     @#$TMPO                 ;ENSURE $TMPO CLEAR
004646  005000                          CLR     R0                      ;USED TO SET THE UNIT AVAIL BITS
004650  012701  000010                  MOV     #10,R1                  ;SOB COUNT
004654  013777  001304  175326  9$:     MOV     @#$TMPO,@RKDA           ;SET UNIT NUMBER
004662  012777  000015  175312          MOV     #15,@RKCS               ;SEND DRIVE RESET
004670  032777  000200  175302          BIT     #BIT7,@RKER             ;NON EXISTANT DISK?
004676  001011                          BNE     7$                      ;BRANCH IF YES
004700  017702  175272                  MOV     @RKDS,R2                ;GET DRIVE STATUS
004704  042702  177537                  BIC     #177537,R2              ;GET BITS 5 & 7 ONLY
004710  022702  000200                  CMP     #200,R2                 ;IS DRIVE READY?
004714  001002                          BNE     7$                      ;BRANCH IF NO
004716  052700  000400                  BIS     #BIT8,R0                ;SET UNIT AVAILABLE
004722  006000                  7$:     ROR     R0
004724  012777  000001  175250          MOV     #1,@RKCS                ;CLEAR THE ERRORS
004732  062737  020000  001304          ADD     #20000,@#$TMPO          ;SELECT NEXT UNIT
004740  077133                          SOB     R1,9$                   ;CONTINUE
004742  110037  001650                  MOVB    R0,@#SYSSIZE+2          ;STORE IN TABLE

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

004746  012737  005206  000004  21$:    MOV     #11$,@#ERRVEC           ;SET THE ERROR VECTOR
004754  005737  176710                  TST     @#176710                ;IS THERE AN RP ON THE SYSTEM?
                                                                        ;STAY HERE IF YES
004760  012737  004774  000004          MOV     #1$,@#ERRVEC
004766  005777  175224                  TST     @RP4CS1         ;IS THERE AN RP04 ON SYSTEM?
004772  000441                  100$:   BR      10$                     ;BRANCH IF YES
                                ;;*********************************************************************

                                ;;*********************************************************************
004774  012737  005076  000004  1$:     MOV     #10$,@#ERRVEC           ;SETUP TIMEOUT VEC FOR RP03 TEST
005002  012737  000001  001304          MOV     #1,@#$TMPO              ;SETUP TEMPO
005010  005000                          CLR     R0                      ;USED TO SET UNIT AVAILABLE BITS
005012  012701  000010                  MOV     #10,R1                  ;SOB COUNT
005016  013777  001304  175134  3$:     MOV     @#$TMPO,@RP3CS          ;SET FUNCTION IDLE WITH UNIT NO
005024  005777  175130                  TST     @RP3CS                  ;WAS THERE AN ERROR?
005030  100006                          BPL     6$                      ;BRANCH IF NO
005032  006000                  4$:     ROR     R0                      ;UNIT NOT AVAILABLE
005034  062737  000400  001304          ADD     #400,@#$TMPO            ;SELECT NEXT UNIT
005042  077113                          SOB     R1,3$                   ;CONTINUE
005044  000412                          BR      5$
005046  017702  175102          6$:     MOV     @RP3DS,R2               ;GET STATUS REGISTER
005052  042702  136377                  BIC     #136377,R2              ;GET BITS 14, 9 & 8 ONLY
005056  022702  041400                  CMP     #41400,R2               ;IS DRIVE READY?
005062  001363                          BNE     4$                      ;BRANCH IF NO
005064  052700  000400                  BIS     #BIT8,R0                ;SET DRIVE AVAILABLE BIT
005070  000760                          BR      4$                      ;CONTINUE
005072  110037  001646          5$:     MOVB    R0,@#SYSSIZE            ;STORE IN TABLE

                                ;;*********************************************************************
005076  012737  005206  000004  10$:    MOV     #11$,@#ERRVEC           ;SETUP ERROR VEC FOR RP04 TEST
005104  005037  001304                  CLR     @#$TMPO
```

```
005110  005000                          CLR     R0                      ;UNIT AVAILABLE WORD
005112  012701  000010                  MOV     #10,R1                  ;SOB COUNT
005116  113777  001304  175104  14$:    MOVB    @#$TMP0,@RP4CS2         ;SET UNIT NUMBER
005124  012777  000021  175064          MOV     #21,@RP4CS1             ;TRY READ-IN-PRESET
005132  032777  010000  175070          BIT     #BIT12,@RP4CS2          ;NON EXISTANT DRIVE?
005140  001011                          BNE     12$                     ;BRANCH IF YES
005142  017702  175066                  MOV     @RP4DS,R2               ;GET DRIVE STATUS
005146  042702  163277                  BIC     #163277,R2              ;GET BITS 12, 11, 8, & 6 ONLY
005152  022702  010500                  CMP     #10500,R2               ;IS DRIVE READY?
005156  001002                          BNE     12$                     ;BRANCH IF NO
005160  052700  000400                  BIS     #BIT8,R0                ;SET UNIT AVAILABLE
005164  006000                  12$:    ROR     R0
005166  052777  000040  175034          BIS     #BIT5,@RP4CS2           ;CLEAR ERROR BITS
005174  005237  001304                  INC     @#$TMP0                 ;SELECT NEXT DRIVE
005200  077132                          SOB     R1,14$                  ;CONTINUE
005202  110037  001656                  MOVB    R0,@#SYSSIZE+10         ;STORE IN TABLE

                                ;;*************************************************************
005206  012737  005316  000004  11$:    MOV     #15$,@#ERRVEC           ;SETUP ERROR VEC FOR RS04 TEST
005214  005037  001304                  CLR     @#$TMP0
005220  005000                          CLR     R0
005222  012701  000010                  MOV     #10,R1                  ;SOB COUNT
005226  113777  001304  175034  18$:    MOVB    @#$TMP0,@RSCS2          ;SET UNIT NUMBER
005234  012777  000001  175014          MOV     #1,@RSCS1               ;TRY NOP OPERATION
005242  032777  010000  175020          BIT     #BIT12,@RSCS2           ;NON EXISTANT DRIVE?
005250  001011                          BNE     16$                     ;BRANCH IF YES
005252  017702  175016                  MOV     @RSDS,R2                ;GET DRIVE STATUS
005256  042702  163577                  BIC     #163577,R2              ;GET BITS 12, 11, & 7 ONLY
005262  022702  010200                  CMP     #10200,R2               ;IS DRIVE READY?
005266  001002                          BNE     16$                     ;BRANCH IF NO
005270  052700  000400                  BIS     #BIT8,R0                ;SET DRIVE AVAILABLE BIT
005274  006000                  16$:    ROR     R0
005276  052777  000040  174764          BIS     #BIT5,@RSCS2            ;CLEAR ANY ERROR BITS
005304  005237  001304                  INC     @#$TMP0                 ;SELECT NEXT UNIT
005310  077132                          SOB     R1,18$                  ;CONTINUE
005312  110037  001660                  MOVB    R0,@#SYSSIZE+12         ;STORE IN TABLE
                                ;
                                ;NEXT, DELETE XXDP UNIT 0 FROM TABLE
005316  122737  000002  000041  15$:    CMPB    #2,@#41                 ;RK?
005324  001004                          BNE     19$                     ;BRANCH IF NO
005326  042737  000001  001650          BIC     #BIT0,@#SYSSIZE+2       ;MAKE UNIT ZERO NOT AVAILABLE
005334  000420                          BR      20$
005336  113700  000041          19$:    MOVB    @#41,R0                 ;GET LOCATION 41
005342  042700  177770                  BIC     #177770,R0              ;GET LEAST SIG 3 BITS
005346  000241                          CLC                             ;ENSURE C CLEAR
005350  006100                          ROL     R0                      ;ADJUST
005352  122700  000002                  CMPB    #2,R0                   ;
005356  002404                          BLT     40$                     ;BRANCH IF NO
005360  042737  000001  001646          BIC     #BIT0,@#SYSSIZE
005366  000403                          BR      20$
005370  042760  000001  001652  40$:    BIC     #BIT0,SYSSIZE+4(R0)
005376  005737  001546          20$:    TST     @#QV                    ;ACT11?
005402  001052                          BNE     LOOP            ;;BRANCH IF YES
005404  105737  003325                  TSTB    @#XXDPC                 ;XXDP CHAIN MODE?
005410  001047                          BNE     LOOP            ;;BRANCH IF YES
005412  105737  001200                  TSTB    @#$PASS                 ;FIRST PASS?
```

```
005416  001044                           BNE     LOOP                ;;BRANCH IF NO
005420  005227  177777                   INC     #-1
005424  001041                           BNE     LOOP                ;;BRANCH IF NOT FIRST TIME
005426  032777  000200  173606           BIT     #SW7,@SWR                   ;INHIBIT SIZE TYPEOUT?
005434  001035                           BNE     LOOP                ;;BRANCH IF YES
005436  032737  000400  001532           BIT     #BIT8,@#OPT.CP              ;TTY AVAILABLE?
005444  001431                           BEQ     LOOP                ;;BRANCH IF NO TTY
005446  004767  043576                   JSR     PC,TYPSIZ                   ;GO TYPE SYSTEM SIZE
005452  104401  005460                   TYPE    ,65$                ;;TYPE ASCIZ STRING
005456  000417                           BR      64$                 ;;GET OVER THE ASCIZ
                                 ;;65$:  .ASCIZ  /TYPE A CHARACTER TO CONTINUE/<CRLF>
005516                           64$:
005516  005037  177776                   CLR     @#PSW
005522  000001                           WAIT
005524  000137  004634                   JMP     @#SIZE                      ;GO CHECK SYSTEM AGAIN
```

```
                                        ;PROGRAM RESTARTS HERE AFTER RELOCATION ABOVE 28K IS COMPLETE.
                                        ;INITIALIZE TRAP VECTORS
005530  012706  000700                  LOOP:   MOV     #USESTK,SP              ;SET THE STACK...WILL BE DIFFERENT
                                                                                ;THAN KERN STACK WHEN IN OUTER MODE
005534  012700  000004                          MOV     #ERRVEC,R0
005540  013701  177776                          MOV     @#PSW,R1                ;GET CURRENT PSW
005544  012720  055064                          MOV     #ERPRT,(R0)+            ;SET ERROR VEC
005550  052701  000340                          BIS     #PR7,R1         ;SET PRIORITY 7 IN CURRENT PSW
005554  042701  000020                          BIC     #BIT4,R1
005560  010120                                  MOV     R1,(R0)+
005562  012720  054774                          MOV     #RESERR,(R0)+           ;SET RESERVED INST TRAP VECTOR
005566  010120                                  MOV     R1,(R0)+
005570  012720  001534                          MOV     #SRTRN,(R0)+            ;SET T BIT VEC
005574  005020                                  CLR     (R0)+                   ;SET TBIT VEC+2
005576  005720                                  TST     (R0)+                   ;BUMP R0 TO SCOPE VEC+2
005600  005020                                  CLR     (R0)+                   ;SET SCOPE VEC+2
005602  062700  000006                          ADD     #6,R0                   ;SET R0 TO ERROR TRAP VEC
005606  012720  000340                          MOV     #PR7,(R0)+              ;SET ERROR VEC
005612  005720                                  TST     (R0)+
005614  012720  000340                          MOV     #PR7,(R0)+              ;SET TRAP VEC+2
005620  012737  054414  000114                  MOV     #.PARSRV,@#CACHVEC      ;SET PARITY ERROR VECTOR
005626  010137  000116                          MOV     R1,@#CACHVEC+2
005632  012737  054662  000250                  MOV     #KTABRT,@#MMVEC         ;SET MEM. MGMT. ABORT VECTOR
005640  010137  000252                          MOV     R1,@#MMVEC+2
005644  005077  173374                          CLR     @DISPLAY
005650  042737  000340  177776                  BIC     #PR7,@#PSW
                                        ;;*****************************************************************
                                        ;*TEST 1        MEMORY VERIFICATION TEST
                                        ;;*****************************************************************
005656                                  TST1:
005656  012767  000001  173440                  MOV     #1,$TIMES       ;;DO 1 ITERATION
005664  000004                                  SCOPE
005666  112737  000001  001202                  MOVB    #1,@#$TSTNM

                                                .SBTTL  START OF SECTION 0
                                        ;000000000000 FIRST ADDRESS TO BE RELOCATED 000000000
005674  112737  000001  001202          RELO:   MOVB    #$TN-1,@#$TSTNM
005702  010700                                  MOV     PC,R0           ;GET PC
005704  005740                                  TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF RELO
005706  010037  001554                          MOV     R0,@#FRSTAD     ;SAVE
005712  010700                                  MOV     PC,R0           ;GET CURRENT PC
005714  162700  005714                          SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
005720  010037  001550                          MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
005724  010737  001212                          MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
005730  062737  000026  001212                  ADD     #26,@#$LPERR    ;ADJUST
005736  013737  001212  001210                  MOV     @#$LPERR,@#$LPADR
005744  105737  001544                          TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
005750  001402                                  BEQ     .+6
005752  000167  000720                          JMP     RELEO
                                        ;MEMORY AND DISK (IF SELECTED) VERIFICATION TEST.
005756  000167  000714                          JMP     1$
005762  177777  177777  177777                  .WORD   -1,-1,-1,-1,0,0,0,0
005770  177777  000000  000000
005776  000000  000000
006002  177777  177777  177777                  .WORD   -1,-1,-1,-1,0,0,0,0
006010  177777  000000  000000
```

```
006016  000000  000000
006022  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006030  177777  000000  000000
006036  000000  000000
006042  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006050  177777  000000  000000
006056  000000  000000
006062  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006070  177777  000000  000000
006076  000000  000000
006102  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006110  177777  000000  000000
006116  000000  000000
006122  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006130  177777  000000  000000
006136  000000  000000
006142  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006150  177777  000000  000000
006156  000000  000000
006162  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006170  177777  000000  000000
006176  000000  000000
006202  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006210  177777  000000  000000
006216  000000  000000
006222  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006230  177777  000000  000000
006236  000000  000000
006242  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006250  177777  000000  000000
006256  000000  000000
006262  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006270  177777  000000  000000
006276  000000  000000
006302  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006310  177777  000000  000000
006316  000000  000000
006322  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006330  177777  000000  000000
006336  000000  000000
006342  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006350  177777  000000  000000
006356  000000  000000
006362  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006370  177777  000000  000000
006376  000000  000000
006402  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006410  177777  000000  000000
006416  000000  000000
006422  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006430  177777  000000  000000
006436  000000  000000
006442  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006450  177777  000000  000000
006456  000000  000000
006462  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
```

```
006470  177777  000000  000000
006476  000000  000000
[illegible]  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006510  177777  000000  000000
006516  000000  000000
006522  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006530  177777  000000  000000
[illegible]  000000  000000
[illegible]  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
[illegible]  177777  000000  000000
[illegible]  000000  000000
[illegible]  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
[illegible]  177777  000000  000000
[illegible]  000000  000000
006602  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006610  177777  000000  000000
006616  000000  000000
006622  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006630  177777  000000  000000
006636  000000  000000
006642  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
006650  177777  000000  000000
006656  000000  000000
006662  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0
006670  177777  000000  000000
006676                          1$:
006676  000004                  RELEO:  SCOPE
006700  010702                          MOV     PC,R2
006702  062702  000012                  ADD     #12,R2
006706  012707  036574                  MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
006712  000000                  RELOO:  .WORD   0
                                ;0000000000000 LAST ADDRESS OF CODE TO BE RELOCATED 00000000000

                                ;;*****************************************************************
                                ;*TEST 2        CHECK BRANCH INSTRUCTIONS
                                ;;*****************************************************************
006714                          TST2:
006714  012767  000001  172402          MOV     #1,$TIMES       ;;DO 1 ITERATION
006722  000004                          SCOPE
006724  112737  000002  001202          MOVB    #2,@#$TSTNM

                                        .SBTTL  START OF SECTION 1
                                ;1111111111111 FIRST ADDRESS TO BE RELOCATED 111111111
006732  112737  000002  001202  REL1:   MOVB    #$TN-1,@#$TSTNM
006740  010700                          MOV     PC,R0           ;GET PC
006742  005740                          TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL1
006744  010037  001554                  MOV     R0,@#FRSTAD     ;SAVE
006750  010700                          MOV     PC,R0           ;GET CURRENT PC
006752  162700  006752                  SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
006756  010037  001550                  MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
006762  010737  001212                  MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
006766  062737  000026  001212          ADD     #26,@#$LPERR    ;ADJUST
006774  013737  001212  001210          MOV     @#$LPERR,@#$LPADR
007002  105737  001544                  TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
007006  001402                          BEQ     .+6
007010  000167  004052                  JMP     RELE1
```

```
                                 ;
007014  000257                           CCC                      ;CC'S=0000
007016  103407                           BCS     CC0              ;SAME AS BLO
007020  102406                           BVS     CC0
007022  001405                           BEQ     CC0
007024  100404                           BMI     CC0
007026  002403                           BLT     CC0
007030  003402                           BLE     CC0
007032  101401                           BLOS    CC0
007034  101001                           BHI     .+4
007036  104000                   CC0:    HLT                      ;ONE OF THE ABOVE BRANCHES FAILED
                                 ;CONTINUE
007040  000270                           SEN                      ;CC'S=1000
007042  100003                           BPL     CC1
007044  002002                           BGE     CC1
007046  003001                           BGT     CC1
007050  002401                           BLT     .+4
007052  104000                   CC1:    HLT                      ;ONE OF THE ABOVE BRANCHES FAILED
                                 ;CONTINUE
007054  000262                           SEV                      ;CC'S=1010
007056  102003                           BVC     CC2
007060  002402                           BLT     CC2
007062  003401                           BLE     CC2
007064  002001                           BGE     .+4
007066  104000                   CC2:    HLT                      ;ERROR! ONE OF THE ABOVE BRANCHES FAILED
                                 ;CONTINUE
007070  000261                           SEC                      ;CC'S=1011
007072  103002                           BCC     CC3
007074  101001                           BHI     CC3
007076  003001                           BGT     .+4
007100  104000                   CC3:    HLT                      ;ERROR! ONE OF THE ABOVE BRANCHES FAILED
                                 ;CONTINUE
007102  000264                           SEZ                      ;CC'S=1111
007104  001003                           BNE     CC4
007106  003002                           BGT     CC4
007110  101001                           BHI     CC4
007112  003401                           BLE     .+4
007114  104000                   CC4:    HLT                      ;ERROR! ONE OF THE ABOVE BRANCHES FAILED
                                 ;;*************************************************************
                                 ;*TEST 3        TEST UNIARY CONDITION CODES
                                 ;;*************************************************************
007116                           TST3:
007116  000004                           SCOPE
007120  112737  000003  001202           MOVB    #3,@#$TSTNM
                                 ;CLR    R0
007126  000277                           SCC
007130  000244                           CLZ
007132  005000                           CLR     R0               ;R0=0,CC'S=0100
007134  103404                           BCS     CLR0
007136  102403                           BVS     CLR0
007140  001002                           BNE     CLR0
007142  100401                           BMI     CLR0
```

```
007144  003401                          BLE     .+4
007146  104000                  CLRO:   HLT                     ;ERROR! INCORRECT CC'S AFTER CLR

007150  000277                          SCC
007152  000244                          CLZ
007154  005700                          TST     R0              ;R0=0,CC'S=0100
007156  103404                          BCS     TSTO
007160  102403                          BVS     TSTO
007162  001002                          BNE     TSTO
007164  100401                          BMI     TSTO
007166  101401                          BLOS    .+4
007170  104000                  TSTO:   HLT                     ;ERROR! INCORRECT CC'S AFTER TST

007172  000257                          CCC
007174  000266                          +SEZ!SEV
007176  005100                          COM     R0              ;R0=-1,CC'S=1001
007200  103004                          BCC     COMO
007202  102403                          BVS     COMO
007204  001402                          BEQ     COMO
007206  100001                          BPL     COMO
007210  002401                          BLT     .+4
007212  104000                  COMO:   HLT                     ;ERROR! INCORRECT CC'S AFTER COM

007214  000261                          SEC
007216  005500                          ADC     R0              ;R0=000000,CC'S=0101
007220  103003                          BCC     ADCO
007222  102402                          BVS     ADCO
007224  001001                          BNE     ADCO
007226  002001                          BGE     .+4
007230  104000                  ADCO:   HLT                     ;ERROR! INCORRECT CC'S AFTER ADC

007232  000261                          SEC
007234  006000                          ROR     R0              ;R0=100000,CC'S=1010
007236  103404                          BCS     RORO
007240  102003                          BVC     RORO
007242  001402                          BEQ     RORO
007244  100001                          BPL     RORO
007246  003001                          BGT     .+4
007250  104000                  RORO:   HLT                     ;ERROR! INCORRECT CC'S AFTER ROR
007252  000277                          SCC
007254  000242                          CLV
007256  005300                          DEC     R0              ;R0=077777,CC'S=0011
007260  103004                          BCC     DECO
007262  102003                          BVC     DECO
007264  001402                          BEQ     DECO
007266  100401                          BMI     DECO
007270  003401                          BLE     .+4
007272  104000                  DECO:   HLT                     ;ERROR! INCORRECT CC'S AFTER DEC

007274  000257                          CCC
007276  005200                          INC     R0              ;R0=100000,CC'S=1010
007300  103404                          BCS     INCO
007302  102003                          BVC     INCO
007304  001402                          BEQ     INCO
007306  100001                          BPL     INCO
007310  003001                          BGT     .+4
```

F06


```
007312  104000                          INCO:   HLT                     ;ERROR! INCORRECT CC'S AFTER INC

007314  000277                                  SCC
007316  000242                                  CLV
007320  005400                                  NEG     R0              ;R0=100000,CC'S=1011
007322  103003                                  BCC     NEGO
007324  102002                                  BVC     NEGO
007326  001401                                  BEQ     NEGO
007330  002001                                  BGE     .+4
007332  104000                          NEGO:   HLT                     ;ERROR! INCORRECT CC'S AFTER NEG

007334  000261                                  SEC
007336  006300                                  ASL     R0              ;R0=000000,CC'S=0111
007340  103004                                  BCC     ASLO
007342  102003                                  BVC     ASLO
007344  001002                                  BNE     ASLO
007346  100401                                  BMI     ASLO
007350  101401                                  BLOS    .+4
007352  104000                          ASLO:   HLT                     ;ERROR! INCORRECT CC'S AFTER ASL

007354  006100                                  ROL     R0              ;R0=000001,CC'S=0000
007356  103402                                  BCS     ROLO
007360  003401                                  BLE     ROLO
007362  002001                                  BGE     .+4
007364  104000                          ROLO:   HLT                     ;ERROR! INCORRECT CC'S AFTER ROL

007366  006200                                  ASR     R0              ;R0=000000,CC'S=0111
007370  103003                                  BCC     ASRO
007372  102002                                  BVC     ASRO
007374  001001                                  BNE     ASRO
007376  002401                                  BLT     .+4
007400  104000                          ASRO:   HLT                     ;ERROR! INCORRECT CC'S AFTER ASR

007402  000277                                  SCC
007404  005600                                  SBC     R0              ;R0=-1,CC'S=1001
007406  103002                                  BCC     SBCO
007410  102401                                  BVS     SBCO
007412  003401                                  BLE     .+4
007414  104000                          SBCO:   HLT                     ;ERROR! INCORRECT CC'S AFTER SBC

007416  005400                                  NEG     R0              ;R0=000001,CC'S=00001
007420  000300                                  SWAB    R0              ;R0=000400 ,CC'S=0100
007422  103403                                  BCS     SWABO
007424  102402                                  BVS     SWABO
007426  001001                                  BNE     SWABO
007430  002001                                  BGE     .+4
007432  104000                          SWABO:  HLT                     ;ERROR! INCORRECT CC'S AFTER SWAB
                                        ;;************************************************************
                                        ;*TEST 4        CHECK REGISTER SELECTION
                                        ;;************************************************************
007434                                  TST4:
007434  000004                                  SCOPE
007436  112737  000004  001202                  MOVB    #4,@#$TSTNM
007444  012737  000005  001324                  MOV     #5,@#$TIMES
```

```
007452  005000                          CLR     R0
007454  000277                          SCC
007456  006100                          ROL     R0                      ;R0=1
007460  010002                          MOV     R0,R2
007462  006302                          ASL     R2                      ;R2=2
007464  010203                          MOV     R2,R3
007466  006303                          ASL     R3                      ;R3=4
007470  010304                          MOV     R3,R4
007472  006304                          ASL     R4                      ;R4=10
007474  010405                          MOV     R4,R5
007476  006305                          ASL     R5                      ;R5=20
007500  010546                          MOV     R5,-(SP)                ;SET BITS SET IN REGISTERS
007502  050416                          BIS     R4,(SP)                 ;INTO STACK ADDRESS
007504  050316                          BIS     R3,(SP)
007506  050216                          BIS     R2,(SP)
007510  050016                          BIS     R0,(SP)
007512  022726  000037                  CMP     #37,(SP)+
007516  001401                          BEQ     .+4                     ;WERE SET
007520  104000                          HLT                             ;MISSING BIT(S) REPRESENT
                                                                        ;INCORRECT REGISTER SELECTION

                                ;CHECK THAT ALL BITS CAN BE SET & CLEARED IN ALL REGISTERS
007522  000257                          CCC
007524  112700  000377                  MOVB    #377,R0                 ;SET ALL BITS (MOVB EXTENDS SIGN)
007530  006100                  1$:     ROL     R0                      ;ROTATE A 0 THROUGH ALL BIT
007532  103776                          BCS     1$                      ;POSITIONS
007534  005200                          INC     R0                      ;FINAL RESULT IS -1
007536  001401                          BEQ     .+4
007540  104000                          HLT                             ;ERROR!

007542  012700  000020                  MOV     #16.,R0                 ;SET SHIFT COUNT
007546  005002                          CLR     R2
007550  000261                  2$:     SEC
007552  006002                          ROR     R2                      ;ROTATE 1 THROUGH ALL BIT POSITS
007554  005300                          DEC     R0                      ;DECREMENT SHIFT COUNT
007556  001374                          BNE     2$
007560  005102                          COM     R2                      ;R2 SHOULD CONTAIN -1
007562  001401                          BEQ     .+4
007564  104000                          HLT                             ;ERROR! CHECK R2 SHOULD = 0

007566  012703  100000                  MOV     #100000,R3
007572  006203                  3$:     ASR     R3                      ;EXTEND 1 BIT THROUGH ALL POSITIONS
007574  103376                          BCC     3$
007576  005203                          INC     R3
007600  001401                          BEQ     .+4
007602  104000                          HLT                             ;ERROR!
```

```
007604  112704  177401                  MOVB    #177401,R4      ;R4=1
007610  060404                  4$:     ADD     R4,R4           ;HAS THE AFFECT OF SHIFTING A BIT
007612  103376                          BCC     4$              ;THROUGH ALL POSITIONS
007614  005704                          TST     R4              ;RESULT SHOULD BE 0
007616  001401                          BEQ     .+4
007620  104000                          HLT

007622  012705  000001                  MOV     #1,R5
007626  006305                  5$:     ASL     R5
007630  102376                          BVC     5$
007632  006305                          ASL     R5
007634  103002                          BCC     6$
007636  005705                          TST     R5
007640  001401                          BEQ     .+4
007642  104000                  6$:     HLT

                                ;CHECK REGISTER VOLITILITY
007644  005002                          CLR     R2
007646  005102                          COM     R2              ;R2=-1
007650  010203                          MOV     R2,R3
007652  000257                          CCC
007654  006002                          ROR     R2              ;R2=LOOP COUNT
007656  006202                          ASR     R2
007660  010304                  7$:     MOV     R3,R4
007662  005302                          DEC     R2              ;DECREMENT LOOP COUNT
007664  001375                          BNE     7$
007666  005203                          INC     R3              ;CHECK R3
007670  001002                          BNE     8$
007672  005204                          INC     R4              ;CHECK R4
007674  001401                          BEQ     .+4
007676  104000                  8$:     HLT

                                ;CHECK TRANSFER OF REGISTER DATA BETWEEN THE GS AND GD REGISTERS
007700  032737  000020  177776  GSTST:  BIT     #20,@#PSW       ;CHECK IF 'T' BIT IS SET
007706  001050                          BNE     7$              ;SKIP TEST IF 'T' BIT SET
007710  010627                          MOV     SP,(PC)+        ;SAVE STACK PTR
007712  000000                  1$:     .WORD   0               ;CONTAINS SAVED STACK PTR
007714  010727                          MOV     PC,(PC)+        ;LOAD DATA. THE CURRENT PC IS USED AS
007716  000000                  2$:     .WORD   0               ;DATA. IF THIS TEST FAILS 2$ CON-
                                                                ;TAINS THE DATA BEING USED.
007720  005267  177772                  INC     2$              ;MAKE ODD TO CHECK BIT 0
007724  016700  177766          3$:     MOV     2$,R0           ;LOAD GD REGISTER 0
007730  010001                          MOV     R0,R1           ;TRANSFER GS REG 0 TO GD REG 1
007732  010102                          MOV     R1,R2           ;AND GS REG 1 TO GD REG 2
007734  010203                          MOV     R2,R3           ;ETC...
007736  010304                          MOV     R3,R4
007740  010405                          MOV     R4,R5
007742  152737  000340  177776          BISB    #340,@#PSW      ;SET PRIORITY LEVEL 7
007750  010506                          MOV     R5,SP           ;TRANSFER GS REG 5 TO GD STK PTR
007752  010627                          MOV     SP,(PC)+        ;TRANSFER GS STK PTR TO MEMORY
007754  000000                  4$:     .WORD   0               ;CONTAINS GS STACK PTR
007756  016706  177730                  MOV     1$,SP           ;RESTORE STK PTR NEEDED FOR HLT/SCOPE
007762  142737  000340  177776          BICB    #340,@#PSW      ;SET PRIORITY LEVEL 0
007770  026700  177760                  CMP     4$,R0           ;COMPARE GS/GD STACK WITH GS REG 0
007774  001004                          BNE     5$              ;BRANCH IF THEY WERE NOT =
```

```
007776  006367  177714                  ASL     2$                      ;SHIFT TEST DATA UNTIL = 000000
010002  001350                          BNE     3$
010004  000411                          BR      6$
010006  010046                  5$:     MOV     R0,-(SP)                ;GET GS REG 0
010010  010146                          MOV     R1,-(SP)                ;ETC...
010012  010246                          MOV     R2,-(SP)
010014  010346                          MOV     R3,-(SP)
010016  010446                          MOV     R4,-(SP)
010020  010546                          MOV     R5,-(SP)
010022  104000                          HLT                             ;ERROR! DATA IN GS STK PTR NOT = GS REG 0
                                                                        ;GS REG 0-GS REG 5 ARE ON THE STACK
010024  016706  177662                  MOV     1$,SP                   ;RESTORE STACK PTR
010030                          6$:
010030                          7$:
                                ;;*********************************************************************
                                ;*TEST 5        TEST UNIARY WORD INSTRUCTIONS USING ADDRESS MODE 1
                                ;;*********************************************************************
010030                          TST5:
010030  000004                          SCOPE
010032  112737  000005  001202          MOVB    #5,@#$TSTNM
010040  012737  000005  001324          MOV     #5,@#$TIMES
010046  000401                          BR      .+4
010050  000000                          .WORD   0                       ;RESERVE ADDRESS FOR TESTS
010052  010702                          MOV     PC,R2
010054  162702  000004                  SUB     #4,R2                   ;R2 POINTS TO RESERVED WORD
010060  005012                          CLR     (R2)                    ;PRESET (R2)

010062  000261                          SEC
010064  006012                          ROR     (R2)                    ;(R2)=100000,CC=1010
010066  101402                          BLOS    ROR1
010070  100001                          BPL     ROR1
010072  002001                          BGE     .+4
010074  104000                  ROR1:   HLT                             ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010076  000257                          CCC
010100  000261                          SEC
010102  005312                          DEC     (R2)                    ;(R2)=077777,CC=0011
010104  103001                          BCC     DEC1
010106  003401                          BLE     .+4
010110  104000                  DEC1:   HLT                             ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010112  000257                          CCC
010114  000261                          SEC
010116  005512                          ADC     (R2)                    ;(R2)=100000,CC=1010
010120  103403                          BCS     ADC1
010122  102002                          BVC     ADC1
010124  100001                          BPL     ADC1
010126  001001                          BNE     .+4
010130  104000                  ADC1:   HLT                             ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010132  006112                          ROL     (R2)                    ;(R2)=000000,CC=0111
010134  103003                          BCC     ROL1
010136  102002                          BVC     ROL1
010140  001001                          BNE     ROL1
010142  100001                          BPL     .+4
010144  104000                  ROL1:   HLT                             ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
```

```

010146  006112                  ROL     (R2)            ;(R2)=000001,CC=0000
010150  101402                  BLOS    ROL1A           ;BRANCH IF C OR Z IS SET
010152  102401                  BVS     ROL1A
010154  100001                  BPL     .+4
010156  104000          ROL1A:  HLT

010160  006212                  ASR     (R2)            ;(R2)=000000,CC=0111
010162  103003                  BCC     ASR1
010164  102002                  BVC     ASR1
010166  001001                  BNE     ASR1
010170  100001                  BPL     .+4
010172  104000          ASR1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010174  006012                  ROR     (R2)            ;(R2)=100000,CC=1010
010176  103403                  BCS     ROR1A
010200  102002                  BVC     ROR1A
010202  001401                  BEQ     ROR1A
010204  100401                  BMI     .+4
010206  104000          ROR1A:  HLT

010210  000261                  SEC
010212  005212                  INC     (R2)            ;(R2)=100001,CC=1001
010214  103003                  BCC     INC1
010216  102402                  BVS     INC1
010220  001401                  BEQ     INC1
010222  100401                  BMI     .+4
010224  104000          INC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010226  005612                  SBC     (R2)            ;(R2)=100000,CC=1000
010230  103403                  BCS     SBC1
010232  102402                  BVS     SBC1
010234  001401                  BEQ     SBC1
010236  100401                  BMI     .+4
010240  104000          SBC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010242  000261                  SEC
010244  005612                  SBC     (R2)            ;(R2)=077777,CC=0010
010246  103403                  BCS     SBC1A
010250  102002                  BVC     SBC1A
010252  001401                  BEQ     SBC1A
010254  100001                  BPL     .+4
010256  104000          SBC1A:  HLT                     ;ERROR! INCORRECT CC'S AS SHOEN ABOVE

010260  000261                  SEC
010262  005512                  ADC     (R2)            ;(R2)=100000,CC=1010
010264  100401                  BMI     .+4
010266  104000                  HLT

010270  000261                  SEC
010272  006312                  ASL     (R2)            ;(R2)=000000,CC=0111
010274  103003                  BCC     ASL1
010276  102002                  BVC     ASL1
010300  001001                  BNE     ASL1
010302  100001                  BPL     .+4
010304  104000          ASL1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
```

```

010306  005112                          COM     (R2)            ;(R2)=177777,CC=1001
010310  103002                          BCC     COM1
010312  102401                          BVS     COM1
010314  100401                          BMI     .+4
010316  104000                  COM1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010320  000250                          CLN
010322  005712                          TST     (R2)            ;(R2)=177777,CC=1000
010324  103403                          BCS     TEST1
010326  102402                          BVS     TEST1
010330  100001                          BPL     TEST1
010332  001001                          BNE     .+4
010334  104000                  TEST1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010336  000262                          SEV
010340  005412                          NEG     (R2)            ;(R2)=000001,CC=0000
010342  103002                          BCC     NEG1
010344  102401                          BVS     NEG1
010346  001001                          BNE     .+4
010350  104000                  NEG1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010352  005312                          DEC     (R2)            ;(R2)=000000,CC=0101
010354  103001                          BCC     DEC1A
010356  001401                          BEQ     .+4
010360  104000                  DEC1A:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
                                ;;*************************************************************
                                ;*TEST 6        CHECK UNIARY BYTE INSTRUCTIONS USING ADDRESS MODE 1
                                ;;*************************************************************
010362                          TST6:
010362  000004                          SCOPE
010364  112737  000006  001202          MOVB    #6,@#$TSTNM
010372  000401                          BR      .+4             ;RESERVE A WORD
010374  000000                          .WORD   0               ;ADDRESS RESERVED FOR TESTS
010376  010703                          MOV     PC,R3
010400  162703  000004                  SUB     #4,R3           ;R3 POINTS TO EVEN BYTE OF WORD
010404  010304                          MOV     R3,R4           ;R4 POINTS TO ODD BYTE OF WORD
010406  005204                          INC     R4
010410  005013                          CLR     (R3)            ;PRESET DATA

010412  000261                  1$:     SEC
010414  105513                          ADCB    (R3)            ;ADD CARRY TO EVEN BYTE
010416  100402                          BMI     2$              ;UNTIL EVEN BYTE BECOMES NEGATIVE
010420  105214                          INCB    (R4)            ;INCREMENT ODD BYTE
010422  000773                          BR      1$
010424  102401                  2$:     BVS     .+4             ;(R3)=077600=[0774][200],CC=1010
010426  104000                          HLT
010430  000242                          CLV
010432  105214                          INCB    (R4)            ;(R3)=100200=[1000][200],CC=1010
010434  103402                          BCS     INCB1
010436  102001                          BVC     INCB1
010440  100401                          BMI     .+4
010442  104000                  INCB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010444  106114                          ROLB    (R4)            ;(R3)=000200=[0000][200],CC=0111
010446  103002                          BCC     ROLB1
```

```
010450  102001                  BVC     ROLB1
010452  001401                  BEQ     .+4
010454  104000          ROLB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010456  105614                  SBCB    (R4)            ;(R3)=177600=[1774][200], CC=1001
010460  103002                  BCC     SBCB1
010462  102401                  BVS     SBCB1
010464  100401                  BMI     .+4
010466  104000          SBCB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010470  106313                  ASLB    (R3)            ;(R3)=177400,CC=0111
010472  103002                  BCC     ASLB1
010474  102001                  BVC     ASLB1
010476  001401                  BEQ     .+4
010500  104000          ASLB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010502  105413                  NEGB    (R3)            ;(R3)=177400,CC=0100
010504  103402                  BCS     NEGB1
010506  102401                  BVS     NEGB1
010510  001401                  BEQ     .+4
010512  104000          NEGB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010514  000277                  SCC
010516  105313                  DECB    (R3)            ;(R3)=177777,CC=1001
010520  103002                  BCC     DECB1
010522  102401                  BVS     DECB1
010524  001001                  BNE     .+4
010526  104000          DECB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010530  000241                  CLC
010532  106013                  RORB    (R3)            ;(R3)=177577,CC=0011
010534  103002                  BCC     RORB1
010536  102001                  BVC     RORB1
010540  100001                  BPL     .+4
010542  104000          RORB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010544  000241                  CLC
010546  105114                  COMB    (R4)            ;(R3)=000177,CC=0101
010550  103002                  BCC     COMB1
010552  102401                  BVS     COMB1
010554  001401                  BEQ     .+4
010556  104000          COMB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010560  106213          1$:     ASRB    (R3)            ;SHIFT EVEN BYTE UNTIL V CLEARS
010562  102002                  BVC     2$
010564  105514                  ADCB    (R4)            ;AND ADD CARRY TO ODD BYTE
010566  000774                  BR      1$
010570  103401          2$:     BCS     ASRB1
010572  001401                  BEQ     .+4
010574  104000          ASRB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010576  106214                  ASRB    (R4)
010600  106214                  ASRB    (R4)            ;(R3)=000400,CC=0011
010602  103002                  BCC     ASRB1A
010604  102001                  BVC     ASRB1A
010606  001001                  BNE     .+4
```

```
010610  104000          ASRB1A: HLT                         ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010612  105314                  DECB    (R4)                ;(R3)=000000,CC=0100
010614  001401                  BEQ     .+4
010616  104000                  HLT                         ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010620  000261                  SEC
010622  106014                  RORB    (R4)                ;(R3)=100000,CC=1010
010624  103402                  BCS     RORB1A
010626  102001                  BVC     RORB1A
010630  100401                  BMI     .+4
010632  104000          RORB1A: HLT                         ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010634  000242                  CLV
010636  105314                  DECB    (R4)                ;(R3)=077400,CC=0100
010640  102401                  BVS     .+4
010642  104000                  HLT

010644  000261                  SEC
010646  105313                  DECB    (R3)                ;(R3)=077777,CC=1001
010650  103002                  BCC     DECB1A
010652  102401                  BVS     DECB1A
010654  100401                  BMI     .+4
010656  104000          DECB1A: HLT                         ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010660  000277                  SCC
010662  000313                  SWAB    (R3)                ;(R3)=177577=[1774][177],CC=0000
010664  103402                  BCS     SWAB1
010666  102401                  BVS     SWAB1
010670  100001                  BPL     .+4
010672  104000          SWAB1:  HLT                         ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010674  105714                  TSTB    (R4)                ;(R3)=177577=[1774][177],CC=1000
010676  103402                  BCS     TSTB1
010700  102401                  BVS     TSTB1
010702  100401                  BMI     .+4
010704  104000          TSTB1:  HLT                         ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010706  105014                  CLRB    (R4)                ;(R3)=000177=[0000][177],CC=0100
010710  001401                  BEQ     .+4
010712  104000                  HLT
010714  106313                  ASLB    (R3)                ;(R3)=000376 ,CC=1010
010716  103402                  BCS     ASLB1A
010720  102001                  BVC     ASLB1A
010722  100401                  BMI     .+4
010724  104000          ASLB1A: HLT                         ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010726  105113                  COMB    (R3)                ;(R3)=000001,CC=0001
010730  103002                  BCC     COMB1A
010732  102401                  BVS     COMB1A
010734  100001                  BPL     .+4
010736  104000          COMB1A: HLT                         ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

010740  000313                  SWAB    (R3)                ;(R3)=000400, CC=0100
010742  001401                  BEQ     .+4
010744  104000                  HLT
```

```
010746  105213                          INCB    (R3)
010750  000261                          SEC
010752  105613                          SBCB    (R3)            ;(R3)=000400,CC=0100
010754  001401                          BEQ     .+4
010756  104000                          HLT
010760  022713  000400                  CMP     #400,(R3)       ;CHECK REMAINING RESULT
010764  001401                          BEQ     .+4
010766  104000                          HLT
                                ;;*******************************************************************
                                ;*TEST 7        CHECK UNIARY WORD OPS USING ADDRESS MODES 2 & 4
                                ;;*******************************************************************
010770                          TST7:
010770  000004                          SCOPE
010772  112737  000007  001202          MOVB    #7,@#$TSTNM
011000  000401                          BR      .+4
011002  000000                          .WORD   0               ;ADDRESS RESERVED FOR TESTS
011004  010704                          MOV     PC,R4
011006  162704  000004                  SUB     #4,R4           ;R4 AND R5 POINT TO
011012  010405                          MOV     R4,R5           ;RESERVED WORD
011014  005015                          CLR     (R5)            ;PRESET DATA=0

011016  000277                          SCC
011020  000244                          CLZ
011022  005725                          TST     (R5)+           ;(R5)=000000,CC=0100
011024  103402                          BCS     TEST2
011026  102401                          BVS     TEST2
011030  001401                          BEQ     .+4
011032  104000                  TEST2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011034  005145                          COM     -(R5)           ;(R5)=177777,CC=1001
011036  103001                          BCC     COM4
011040  100401                          BMI     .+4
011042  104000                  COM4:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011044  000241                          CLC
011046  006024                          ROR     (R4)+           ;(R4)=077777,CC=0011
011050  103002                          BCC     ROR2
011052  102001                          BVC     ROR2
011054  100001                          BPL     .+4
011056  104000                  ROR2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011060  000257                          CCC
011062  005244                          INC     -(R4)           ;(R4)=100000,CC=1010
011064  102002                          BVC     INC4
011066  001401                          BEQ     INC4
011070  100401                          BMI     .+4
011072  104000                  INC4:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011074  000261                          SEC
011076  000324                          SWAB    (R4)+           ;(R4)=000200,CC=1000
011100  103401                          BCS     SWAB2
011102  100401                          BMI     .+4
011104  104000                  SWAB2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011106  005425                          NEG     (R5)+           ;(R5)=177600,CC=1001
```

```
011110  103001                      BCC     NEG2
011112  100401                      BMI     .+4
011114  104000              NEG2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011116  005044                      CLR     -(R4)           ;(R4)=000000,CC=0100
011120  001401                      BEQ     .+4
011122  104000                      HLT

011124  000261                      SEC
011126  006045                      ROR     -(R5)           ;(R5)=100000,CC=1010
011130  000261                      SEC
011132  005525                      ADC     (R5)+           ;(R5)=100001,CC=1000
011134  102401                      BVS     ADC2
011136  100401                      BMI     .+4
011140  104000              ADC2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011142  000262                      SEV
011144  006224                      ASR     (R4)+           ;(R4)=140000,CC=1001
011146  103002                      BCC     ASR2
011150  102401                      BVS     ASR2
011152  100401                      BMI     .+4
011154  104000              ASR2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011156  000262                      SEV
011160  006144                      ROL     -(R4)           ;(R4)=100001, CC=1001
011162  103002                      BCC     ROL4
011164  102401                      BVS     ROL4
011166  100401                      BMI     .+4
011170  104000              ROL4:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011172  005645                      SBC     -(R5)           ;(R5)=100000,CC=1000
011174  103001                      BCC     .+4
011176  104000                      HLT                     ;ERROR! 'C' BIT FAILED TO CLEAR

011200  005325                      DEC     (R5)+           ;(R5)=077777,CC=0010
011202  103402                      BCS     DEC2
011204  102001                      BVC     DEC2
011206  100001                      BPL     .+4
011210  104000              DEC2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011212  006324                      ASL     (R4)+           ;(R4)=177776,CC=1010
011214  102401                      BVS     .+4
011216  104000                      HLT
011220  006344                      ASL     -(R4)           ;(R4)=177774,CC=1001
011222  103003                      BCC     ASL4
011224  102402                      BVS     ASL4
011226  001401                      BEQ     ASL4
011230  100401                      BMI     .+4
011232  104000              ASL4:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011234  022724  177774              CMP     #177774,(R4)+
011240  001401                      BEQ     .+4
011242  104000                      HLT
011244  020405                      CMP     R4,R5
011246  001401                      BEQ     .+4
011250  104000                      HLT
```

```
                                        ;;*****************************************************************
                                        ;*TEST 10        CHECK UNIARY BYTE OPS USING ADDRESS MODES 2 & 4
                                        ;;*****************************************************************
011252                                  TST10:
011252  000004                                  SCOPE
011254  112737  000010  001202                  MOVB    #10,@#STSTNM
011262  000401                                  BR      .+4             ;RESERVE A WORD
011264  000000                                  .WORD   0               ;RESERVED WORD
011266  010705                                  MOV     PC,R5
011270  162705  000004                          SUB     #4,R5           ;R5 POINTS TO EVEN BYTE OF RESERVED WORD
011274  010500                                  MOV     R5,R0
011276  010002                                  MOV     R0,R2
011300  005202                                  INC     R2              ;R2 POINTS TO ODD BYTE OF RESERVED WORD
011302  005010                                  CLR     (R0)            ;PRESET

011304  000277                                  SCC
011306  000241                                  CLC
011310  105125                                  COMB    (R5)+           ;(R0)=000377,CC=1001
011312  103002                                  BCC     COMB2
011314  102401                                  BVS     COMB2
011316  100401                                  BMI     .+4
011320  104000                          COMB2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011322  105542                                  ADCB    -(R2)           ;(R0)=000000,CC=0101
011324  001401                                  BEQ     .+4
011326  104000                                  HLT                     ;ERROR! INCORRECT RESULT AS SHOWN ABOVE
011330  105525                                  ADCB    (R5)+           ;(R0)=000400,CC=0000
011332  103401                                  BCS     ADCB2
011334  001001                                  BNE     .+4
011336  104000                          ADCB2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011340  000263                                  +SEC!SEV
011342  106045                                  RORB    -(R5)           ;(R0)=100000,CC=1001
011344  103003                                  BCC     RORB4
011346  102402                                  BVS     RORB4
011350  001401                                  BEQ     RORB4
011352  100401                                  BMI     .+4
011354  104000                          RORB4:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011356  000277                                  SCC
011360  106122                                  ROLB    (R2)+           ;(R0)=100001,CC=0000
011362  103403                                  BCS     ROLB2
011364  102402                                  BVS     ROLB2
011366  001401                                  BEQ     ROLB2
011370  100001                                  BPL     .+4
011372  104000                          ROLB2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011374  000257                                  CCC
011376  106225                                  ASRB    (R5)+           ;(R0)=140001, CC=1010
011400  103402                                  BCS     ASRB2
011402  102001                                  BVC     ASRB2
011404  100401                                  BMI     .+4
011406  104000                          ASRB2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011410  105242                                  INCB    -(R2)           ;(R0)=140002,CC=0000
011412  000277                                  SCC
```

```
011414  106222             ASRB    (R2)+          ;(R0)=140001,CC=0000
011416  103402             BCS     ASRB2A
011420  102401             BVS     ASRB2A
011422  100001             BPL     .+4
011424  104000  ASRB2A:    HLT                    ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011426  000266             +SEZ!SEV               ;SET Z,V
011430  106345             ASLB    -(R5)          ;(R0)=100001,CC=1001
011432  103003             BCC     ASLB4
011434  102402             BVS     ASLB4
011436  001401             BEQ     ASLB4
011440  100401             BMI     .+4
011442  104000  ASLB4:     HLT                    ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011444  105322             DECB    (R2)+          ;(R0)=077401=[0774][001] ,CC=0010
011446  103002             BCC     DECB2
011450  102001             BVC     DECB2
011452  100001             BPL     .+4
011454  104000  DECB2:     HLT                    ;ERROR! INCORRECT CC'S AS SHOWNABOVE

011456  105645             SBCB    -(R5)          ;(R0)=077400, CC=0100
011460  103402             BCS     SBCB4
011462  102401             BVS     SBCB4
011464  001401             BEQ     .+4
011466  104000  SBCB4:     HLT                    ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011470  105442             NEGB    -(R2)          ;(R0)=10400,CC=1001
011472  103002             BCC     NEGB4
011474  102401             BVS     NEGB4
011476  100401             BMI     .+4
011500  104000  NEGB4:     HLT                    ;ERROR! INCORRECT CC'S AS SHOWN ABOVE

011502  105725             TSTB    (R5)+          ;(R0)=100400,CC=0100
011504  103401             BCS     TSTB2
011506  001401             BEQ     .+4
011510  104000  TSTB2:     HLT

011512  105722             TSTB    (R2)+          ;(R0)=100400,CC=1000
011514  001401             BEQ     TSTB2A
011516  100401             BMI     .+4
011520  104000  TSTB2A:    HLT

011522  000261             SEC
011524  000342             SWAB    -(R2)          ;(R0)=000201,CC=1000
011526  103401             BCS     SWAB4
011530  100401             BMI     .+4
011532  104000  SWAB4:     HLT

011534  000277             SCC
011536  105225             INCB    (R5)+          ;(R0)=000601=[0004][201],CC=0000
011540  103003             BCC     INCB2
011542  102402             BVS     INCB2
011544  001401             BEQ     INCB2
011546  100001             BPL     .+4
011550  104000  INCB2:     HLT
```

```
 011552  022227  000601                 CMP     (R2)+,#000601   ;CHECK END RESULT
 011556  001401                         BEQ     .+4
 011560  104000                         HLT
 011562  020205                         CMP     R2,R5           ;CHECK REGISTERS
 011564  001401                         BEQ     .+4
 011566  104000                         HLT
                                ;;**************************************************************
                                ;*TEST 11        CHECK UNIARY WORD OPS USING ADDRESS MODES 3 & 5
                                ;;**************************************************************
 011570                         TST11:
 011570  000004                         SCOPE
 011572  112737  000011  001202         MOVB    #11,@#$TSTNM
 011600  000402                         BR      .+6             ;RESERVE 2 WORDS
 011602  000000                         .WORD   0               ;1 FOR THE ADDRESS
 011604  000000                         .WORD   0               ;AND 1 FOR DATA
 011606  010703                         MOV     PC,R3
 011610  162703  000004                 SUB     #4,R3
 011614  005013                         CLR     (R3)            ;PRESET DATA
 011616  010300                         MOV     R3,R0           ;R0 POINTS TO DATA WORD
 011620  005743                         TST     -(R3)
 011622  010013                         MOV     R0,(R3)
 011624  010304                         MOV     R3,R4

 011626  000257                         CCC
 011630  005733                         TST     @(R3)+          ;(R0)=000000,CC=0100
 011632  001401                         BEQ     .+4
 011634  104000                         HLT

 011636  000261                         SEC
 011640  006053                         ROR     @-(R3)          ;(R0)=100000,CC=1010
 011642  103402                         BCS     RORS
 011644  102001                         BVC     RORS
 011646  100401                         BMI     .+4
 011650  104000                 RORS:   HLT

 011652  000257                         CCC
 011654  006234                         ASR     @(R4)+          ;(R0)=140000,CC=1010
 011656  102001                         BVC     ASR3
 011660  100401                         BMI     .+4
 011662  104000                 ASR3:   HLT

 011664  000250                         CLN
 011666  006333                         ASL     @(R3)+          ;(R0)=100000,CC=1001
 011670  103002                         BCC     ASL3
 011672  102401                         BVS     ASL3
 011674  100401                         BMI     .+4
 011676  104000                 ASL3:   HLT

 011700  000277                         SCC
 011702  005354                         DEC     @-(R4)          ;(R0)=077777, CC=0010
 011704  103003                         BCC     DEC5
 011706  102002                         BVC     DEC5
 011710  001401                         BEQ     DEC5
 011712  100001                         BPL     +4
 011714  104000                 DEC5:   HLT

```

```
011716  005453                          NEG     @-(R3)          ;(R0)=100001, CC=1001
011720  103002                          BCC     NEG5
011722  102401                          BVS     NEG5
011724  100401                          BMI     .+4
011726  104000                  NEG5:   HLT

011730  000262                          SEV
011732  005134                          COM     @(R4)+          ;(R0)=077776, CC=0001
011734  103001                          BCC     COM3
011736  102001                          BVC     .+4
011740  104000                  COM3:   HLT

011742  005233                          INC     @(R3)+          ;(R0)=077777, CC=0001
011744  103001                          BCC     INC3
011746  100001                          BPL     .+4
011750  104000                  INC3:   HLT

011752  005554                          ADC     @-(R4)          ;(R0)=100000, CC=1010
011754  103402                          BCS     ADC5
011756  102001                          BVC     ADC5
011760  100401                          BMI     .+4
011762  104000                  ADC5:   HLT

011764  000257                          CCC
011766  006134                          ROL     @(R4)+          ;(R0)=000000,CC=0111
011770  103002                          BCC     ROL3
011772  102001                          BVC     ROL3
011774  001401                          BEQ     .+4
011776  104000                  ROL3:   HLT

012000  005253                          INC     @-(R3)          ;(R0)=000001, CC=0001
012002  005654                          SBC     @-(R4)          ;(R0)=000000, CC=0100
012004  103401                          BCS     SBC5
012006  001401                          BEQ     .+4
012010  104000                  SBC5:   HLT
                                ;;*************************************************************
                                ;*TEST 12       CHECK UNIARY BYTE OPS USING ADDRESS MODES 3 & 5
                                ;;*************************************************************
012012                          TST12:
012012  000004                          SCOPE
012014  112737  000012  001202          MOVB    #12,@#$TSTNM
012022  000403                          BR      .+10            ;RESERVE 3 WORDS
012024  000000                          .WORD   0               ;1 FOR EVEN BYTE ADDRESS
012026  000000                          .WORD   0               ;1 FOR ODD BYTE ADDRESS
012030  000000                          .WORD   0               ;AND 1 FOR DATA
012032  010702                          MOV     PC,R2
012034  005742                          TST     -(R2)           ;BACK R2 UP TO
012036  005742                          TST     -(R2)           ;DATA WORD
012040  010200                          MOV     R2,R0           ;R0 POINTS TO THE DATA WORD
012042  005010                          CLR     (R0)            ;PRESET DATA
012044  005742                          TST     -(R2)           ;BACK R2 UP TO
012046  005742                          TST     -(R2)           ;EVEN BYTE ADDRESS WORD
012050  010022                          MOV     R0,(R2)+        ;LOAD ADDRESS
012052  005200                          INC     R0              ;ODD BYTE ADDRESS
012054  010022                          MOV     R0,(R2)+        ;LOAD ODD BYTE ADDRESS
012056  010200                          MOV     R2,R0           ;RESET R0
```

```
012060  010205                  MOV     R2,R5
012062  105152                  COMB    @-(R2)          ;(R0)=177400,CC=1001
012064  103001                  BCC     COMB5
012066  100401                  BMI     .+4
012070  104000          COMB5:  HLT
012072  105752                  TSTB    @-(R2)          ;(R0)=177400, CC=0100
012074  001401                  BEQ     .+4
012076  104000                  HLT
012100  000262                  SEV
012102  106255                  ASRB    @-(R5)          ;(R0)=177400, CC=1001
012104  103002                  BCC     ASRB5
012106  102401                  BVS     ASRB5
012110  100401                  BMI     .+4
012112  104000          ASRB5:  HLT

012114  105232                  INCB    @(R2)+          ;(R0)=177401, CC=000
012116  103001                  BCC     INCB3
012120  100001                  BPL     .+4
012122  104000          INCB3:  HLT

012124  000241                  CLC
012126  106055                  RORB    @-(R5)          ;(R0)=177400, CC=0111
012130  103003                  BCC     RORB5
012132  102002                  BVC     RORB5
012134  001001                  BNE     RORB5
012136  100001                  BPL     .+4
012140  104000          RORB5:  HLT

012142  106332                  ASLB    @(R2)+          ;(R0)=177000, CC=1001
012144  103002                  BCC     ASLB3
012146  102401                  BVS     ASLB3
012150  100401                  BMI     .+4
012152  104000          ASLB3:  HLT

012154  105552                  ADCB    @-(R2)          ;(R0)=177400, CC=1000
012156  103401                  BCS     ADCB5
012160  100401                  BMI     .+4
012162  104000          ADCB5:  HLT

012164  000277                  SCC
012166  106135                  ROLB    @(R5)+          ;(R0)=177401, CC=0000
012170  101402                  BLOS    ROLB3           ;BRANCH IF C OR Z IS SET
012172  102401                  BVS     ROLB3
012174  100001                  BPL     .+4
012176  104000          ROLB3:  HLT

012200  000352                  SWAB    @-(R2)          ;(R0)=000777, CC=1000
012202  100401                  BMI     .+4
012204  104000                  HLT

012206  000261                  SEC
012210  105635                  SBCB    @(R5)+          ;(R0)=000377, CC=0100
012212  103401                  BCS     SBCB3
012214  001401                  BEQ     .+4
012216  104000          SBCB3:  HLT

```

```
012220  105432                          NEGB    @(R2)+          ;(R0)=000001
012222  105352                          DECB    @-(R2)          ;(R0)=000000, CC=0101
012224  103001                          BCC     DECBS
012226  001401                          BEQ     .+4
012230  104000                  DECBS:  HLT
                                ;;*******************************************************************
                                ;*TEST 13        CHECK UNIARY WORD OPS USING ADDRESS MODE 6 (PC)
                                ;;*******************************************************************
012232                          TST13:
012232  000004                          SCOPE
012234  112737  000013  001202          MOVB    #13,@#STSTNM
012242  005027                          CLR     (PC)+           ;PRESET DATA = 0
012244  000000                  UWM6:   .WORD   0               ;RESERVED FOR DATA
012246  010700                          MOV     PC,R0
012250  024040                          CMP     -(R0),-(R0)     ;R0 POINTS TO DATA WORD
012252  000277                          SCC
012254  006167  177764                  ROL     UWM6            ;(R0)=000001,CC=0000
012260  103403                          BCS     ROL6
012262  102402                          BVS     ROL6
012264  001401                          BEQ     ROL6
012266  100001                          BPL     .+4
012270  104000                  ROL6:   HLT

012272  005167  177746                  COM     UWM6            ;(R0)=177776, CC=1001
012276  103002                          BCC     COM6
012300  102401                          BVS     COM6
012302  100401                          BMI     .+4
012304  104000                  COM6:   HLT
012306  006267  177732                  ASR     UWM6            ;(R0)=177777, CC=1010
012312  103402                          BCS     ASR6
012314  102001                          BVC     ASR6
012316  100401                          BMI     .+4
012320  104000                  ASR6:   HLT

012322  000277                          SCC
012324  005467  177714                  NEG     UWM6            ;(R0)=000001, CC=0001
012330  103003                          BCC     NEG6
012332  102402                          BVS     NEG6
012334  001401                          BEQ     NEG6
012336  100001                          BPL     .+4
012340  104000                  NEG6:   HLT

012342  000277                          SCC
012344  006067  177674                  ROR     UWM6            ;(R0)=100000, CC=1001
012350  103003                          BCC     ROR6
012352  102402                          BVS     ROR6
012354  001401                          BEQ     ROR6
012356  100401                          BMI     .+4
012360  104000                  ROR6:   HLT

012362  005667  177656                  SBC     UWM6            ;(R0)=077777, CC=0010
012366  103402                          BCS     SBC6
012370  102001                          BVC     SBC6
012372  100001                          BPL     .+4
012374  104000                  SBC6:   HLT
```

```
012376  000242                                  CLV
012400  005267  177640                          INC     UWM6            ;(R0)=100000, CC=1011
012404  103403                                  BCS     INC6
012406  102002                                  BVC     INC6
012410  001401                                  BEQ     INC6
012412  100401                                  BMI     .+4
012414  104000                          INC6:   HLT
012416  006267  177622                          ASR     UWM6            ;(R0)=140000, CC=1010
012422  000261                                  SEC
012424  006367  177614                          ASL     UWM6            ;(R0)=100000, CC=1001
012430  103002                                  BCC     ASL6
012432  102401                                  BVS     ASL6
012434  100401                                  BMI     .+4
012436  104000                          ASL6:   HLT
012440  005367  177600                          DEC     UWM6            ;(R0)=077777, CC=0011
012444  103002                                  BCC     DEC6
012446  102001                                  BVC     DEC6
012450  100001                                  BPL     .+4
012452  104000                          DEC6:   HLT
012454  005567  177564                          ADC     UWM6            ;(R0)=100000, CC=1010
012460  103402                                  BCS     ADC6
012462  102001                                  BVC     ADC6
012464  100401                                  BMI     .+4
012466  104000                          ADC6:   HLT
012470  000242                                  CLV
012472  000367  177546                          SWAB    UWM6
012476  100401                                  BMI     .+4
012500  104000                                  HLT
012502  022710  000200                          CMP     #200,(R0)
012506  001401                                  BEQ     .+4
012510  104000                                  HLT
                                        ;;******************************************************************
                                        ;*TEST 14        CHECK UNIARY BYTE OPS (EVEN/ODD) USING ADDRESS MODE 6 (PC)
                                        ;;******************************************************************
012512                                  TST14:
012512  000004                                  SCOPE
012514  112737  000014  001202                  MOVB    #14,@#$TSTNM
012522  012700  C13064                          MOV     #UBM6,R0
012526  063700  001550                          ADD     @#FACTOR,R0     ;R0 POINTS TO ADDRESS OF DATA
012532  005067  000326                          CLR     UBM6            ;CLEAR DATA
012536  000277                                  SCC
0125[illegible]0  000244                        CLZ
012542  105767  000316                          TSTB    UBM6
012546  103403                                  BCS     TSTB6
012550  102402                                  BVS     TSTB6
012552  001001                                  BNE     TSTB6
012554  100001                                  BPL     .+4
012556  104000                          TSTB6:  HLT
012560  000257                                  CCC
012562  105767  000277                          TSTB    UBM6+1          ;TEST ODD BYTE
012566  001401                                  BEQ     .+4
012570  104000                                  HLT
```

```

012572  105667  000266            SBCB    UBM6            ;(R0)=000000, CC=0100
012576  103402                    BCS     SBCB6
012600  102401                    BVS     SBCB6
012602  001401                    BEQ     .+4
012604  104000           SBCB6:   HLT

012606  000261           1$:      SEC
012610  10[illegible]67  000250            INCB    UBM6            ;LOOP UNTIL (R0)=077600, CC=1011
012614  100403                    BMI     2$
012616  105567  000243            ADCB    UBM6+1          ;INCB INST INCREMENTS EVEN BYTE
01[illegible]2  000771                    BR      1$              ;ADCB INCREMENTS ODD BYTE
01[illegible]4  103001           2$:      BCC     INCB6
012626  102401                    BVS     .+4
012630  104000           INCB6:   HLT

012632  106367  000226            ASLB    UBM6            ;(R0)=077400, CC=0111
012[illegible]6  103003                    BCC     ASLB6
012640  102002                    BVC     ASLB6
012642  001001                    BNE     ASLB6
012644  100001                    BPL     .+4
012646  104000           ASLB6:   HLT

012650  000242                    CLV
012652  105567  000207            ADCB    UBM6+1          ;(R0)=100000, CC=1010
012656  103402                    BCS     ADCB6
012660  102001                    BVC     ADCB6
012662  100401                    BMI     .+4
012664  104000           ADCB6:   HLT

012666  000261                    SEC
012670  10[illegible]067  000171           RORB    UBM6+1          ;(R0)=140000, CC=1010
012674  103402                    BCS     RORB6
012676  102001                    BVC     RORB6
012700  100401                    BMI     .+4
012702  104000           RORB6:   HLT

012704  105167  000154            COMB    UBM6            ;(R0)=140377 CC=1001
012710  103002                    BCC     COMB6
012712  102401                    BVS     COMB6
012714  100401                    BMI     .+4
012716  104000           COMB6:   HLT

012720  000262                    SEV
012722  105467  000137            NEGB    UBM6+1          ;(R0)=040377, CC=0001
012726  103002                    BCC     NEGB6
012730  102401                    BVS     NEGB6
012732  100001                    BPL     .+4
012734  104000           NEGB6:   HLT

012736  106167  000123            ROLB    UBM6+1          ;(R0)=100777, CC=1010
012742  103402                    BCS     ROLB6
012744  102001                    BVC     ROLB6
012746  100401                    BMI     .+4
012750  104000           ROLB6:   HLT

```

```
012752  106267  000106                     ASRB    UBM6            ;(R0)=100777, CC=1001
012756  103002                             BCC     ASRB6
012760  102401                             BVS     ASRB6
012762  100401                             BMI     .+4
012764  104000                     ASRB6:  HLT

012766  105267  000072                     INCB    UBM6            ;(R0)=100400, CC=0101
012772  103002                             BCC     INCB6A
012774  102401                             BVS     INCB6A
012776  001401                             BEQ     .+4
013000  104000                     INCB6A: HLT

013002  105367  000057                     DECB    UBM6+1          ;(R0)=100000, CC=1001
013006  103003                             BCC     DECB6A
013010  102402                             BVS     DECB6A
013012  001401                             BEQ     DECB6A
013014  100401                             BMI     .+4
013016  104000                     DECB6A: HLT

013020  000367  000040                     SWAB    UBM6            ;(R0)=000200, CC=1000
013024  103401                             BCS     SWAB6
013026  100401                             BMI     .+4
013030  104000                     SWAB6:  HLT

013032  106167  000026                     ROLB    UBM6            ;(R0)=000000, CC=0111
013036  103002                             BCC     ROLB6A
013040  102001                             BVC     ROLB6A
013042  001401                             BEQ     .+4
013044  104000                     ROLB6A: HLT

013046  005767  000012                     TST     UBM6            ;(R0)=000000, CC=0100
013052  103402                             BCS     TEST6
013054  102401                             BVS     TEST6
013056  001401                             BEQ     .+4
013060  104000                     TEST6:  HLT

013062  000401                             BR      .+4             ;RESERVE A WORD
013064  000000                     UBM6:   .WORD   0               ;WORD RESERVED FOR DATA
013066  000004                     RELE1:  SCOPE
013070  010702                             MOV     PC,R2
013072  062702  000012                     ADD     #12,R2
013076  012707  036574                     MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
013102  000000                     REL11:  .WORD   0
                                   ;1111111111111 LAST ADDRESS OF CODE TO BE RELOCATED 11111111111


                                   ;;*****************************************************************
                                   ;*TEST 15        CHECK UNIARY WORD OPS USING ADDRESS MODE 7
                                   ;;*****************************************************************
013104                             TST15:
013104  012767  000001  166212             MOV     #1,$TIMES       ;;DO 1 ITERATION
013112  000004                             SCOPE
013114  112737  000015  001202             MOVB    #15,@#$TSTNM

                                           .SBTTL  START OF SECTION 2
                                   ;2222222222222 FIRST ADDRESS TO BE RELOCATED 222222222
013122  112737  000015  001202     REL2:   MOVB    #$TN-1,@#$TSTNM
```

```
013130  010700                          MOV     PC,R0           ;GET PC
013132  005740                          TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL2
013134  010037  001554                  MOV     R0,@#FRSTAD     ;SAVE
013140  010700                          MOV     PC,R0           ;GET CURRENT PC
013142  162700  013142                  SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
013146  010037  001550                  MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
013152  010737  001212                  MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
013156  062737  000026  001212          ADD     #26,@#$LPERR    ;ADJUST
013164  013737  001212  001210          MOV     @#$LPERR,@#$LPADR
013172  105737  001544                  TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
013176  001402                          BEQ     .+6
013200  000167  004062                  JMP     RELE2
013204  000403                          BR      UW7             ;RESERVE 3 WORDS FOR ADDRESSES & DATA
013206  000000                          .WORD   0               ;CONTAINS ADDRESS OF UWM7
013210  000000                  UWM7:   .WORD   0               ;CONTAINS DATA
013212  000000                          .WORD   0               ;CONTAINS ADDRESS OF UWM7

013214  010700                  UW7:    MOV     PC,R0
013216  005740                          TST     -(R0)
013220  005740                          TST     -(R0)
013222  005040                          CLR     -(R0)           ;CLEAR TEST DATA
013224  010002                          MOV     R0,R2
013226  010240                          MOV     R2,-(R0)        ;SET UP ADDRESS
013230  005720                          TST     (R0)+           ;MOVE R0 TO NEXT ADDRESS
013232  005720                          TST     (R0)+
013234  010210                          MOV     R2,(R0)         ;SET NEXT ADDRESS
013236  010200                          MOV     R2,R0           ;SET R0 POINTING TO DATA
013240  000277                          SCC
013242  000244                          CLZ
013244  005772  000002                  TST     @2(2)           ;(R0)=000000, CC=0100
013250  001401                          BEQ     .+4
013252  104000                          HLT

013254  000277                          SCC
013256  005672  177776                  SBC     @-2(2)          ;(R0)=177777, CC=1001
013262  103002                          BCC     SBC7
013264  102401                          BVS     SBC7
013266  100401                          BMI     .+4
013270  104000                  SBC7:   HLT

013272  000277                          SCC
013274  000241                          CLC
013276  006372  000002                  ASL     @2(2)           ;(R0)=177776, CC=1001
013302  103002                          BCC     ASL7
013304  102401                          BVS     ASL7
013306  100401                          BMI     .+4
013310  104000                  ASL7:   HLT

013312  000257                          CCC
013314  005372  000002                  DEC     @2(2)           ;(R0)=177775, CC=1000
013320  103402                          BCS     DEC7
013322  102401                          BVS     DEC7
013324  100401                          BMI     .+4
013326  104000                  DEC7:   HLT

013330  000262                          SEV
```

```
013332 006272 177776            ASR     @-2(2)          ;(R0)=177776, CC=1001
013336 103002                   BCC     ASR7
013340 102401                   BVS     ASR7
013342 100401                   BMI     .+4
013344 104000          ASR7:    HLT

013346 000241                   CLC
013350 000262                   SEV
013352 006072 177776            ROR     @-2(2)          ;(R0)=077777, CC=0000
013356 101402                   BLOS    ROR7            ;BRANCH IF C OR Z IS SET
013360 102401                   BVS     ROR7
013362 100001                   BPL     .+4
013364 104000          ROR7:    HLT

013366 000262                   SEV
013370 005472 000002            NEG     @2(2)           ;(R0)=100001, CC=1001
013374 103002                   BCC     NEG7
013376 102401                   BVS     NEG7
013400 100401                   BMI     .+4
013402 104000          NEG7:    HLT

013404 000250                   CLN
013406 000372 177776            SWAB    @-2(2)          ;(R0)=000600, CC=1000
013412 103401                   BCS     SWAB7
013414 100401                   BMI     .+4
013416 104000          SWAB7:   HLT

013420 000262                   SEV
013422 005172 000002            COM     @2(2)           ;(R0)=177177, CC=1001
013426 103002                   BCC     COM7
013430 102401                   BVS     COM7
013432 100401                   BMI     .+4
013434 104000          COM7:    HLT

013436 000372 000002            SWAB    @2(2)           ;(R0)=077776, CC=1000
013442 100401                   BMI     .+4
013444 104000                   HLT

013446 000277                   SCC
013450 005572 177776            ADC     @-2(2)          ;(R0)=077777, CC=0000
013454 103402                   BCS     ADC7
013456 102401                   BVS     ADC7
013460 100001                   BPL     .+4
013462 104000          ADC7:    HLT

013464 005272 000002            INC     @2(2)           ;(R0)=100000, CC=1010
013470 102001                   BVC     INC7
013472 100401                   BMI     .+4
013474 104000          INC7:    HLT

013476 000257                   CCC
013500 006172 177776            ROL     @-2(2)          ;(R0)=000000, CC=0111
013504 103002                   BCC     ROL7
013506 102001                   BVC     ROL7
013510 001401                   BEQ     .+4
013512 104000          ROL7:    HLT
```

```
                                        ;;*****************************************************************
                                        ;*TEST 16       CHECK UNIARY BYTE OPS USING ADDRESS MODE 7
                                        ;;*****************************************************************
013514                                  TST16:
013514  000004                                  SCOPE
013516  112737  000016  001202                  MOVB    #16,@#$TSTNM
013524  012700  013210                          MOV     #UWM7,R0
```

```
013530  063700  001550            ADD     @#FACTOR,R0
013534  010002                    MOV     R0,R2
013536  010067  177450            MOV     R0,UWM7+2
013542  005720                    TST     (R0)+
013544  005210                    INC     (R0)             ;WORD FOLLOWING UWM7 CONTAINS ADDRESS
013546  005740                    TST     -(R0)            ;OF ODD BYTE, R0 POINTS TO DATA WORD
013550  005010                    CLR     (R0)             ;PRESET DATA
013552  010067  177430            MOV     R0,UWM7-2
                          ;NOTE:  @2(2) REFERENCES THE ODD BYTE, AND @-2(2) REFERENCES THE EVEN BYTE.

013556  000263                    +SEC!SEV                 ;SET C AND V
013560  105672  000002            SBCB    @2(2)            ;(R0)=177400, CC=1001
013564  103003                    BCC     SBCB7
013566  102402                    BVS     SBCB7
013570  001401                    BEQ     SBCB7
013572  100401                    BMI     .+4
013574  104000            SBCB7:  HLT

013576  000277                    SCC                      ;SET CONDITION CODES
013600  105572  177776            ADCB    @-2(2)           ;(R0)=177401, CC=0000
013604  103403                    BCS     ADCB7
013606  102402                    BVS     ADCB7
013610  001401                    BEQ     ADCB7
013612  100001                    BPL     .+4
013614  104000            ADCB7:  HLT

013616  105172  177776            COMB    @-2(2)           ;(R0)=177776, CC=1001
013622  103002                    BCC     COMB7
013624  102401                    BVS     COMB7
013626  100401                    BMI     .+4
013630  104000            COMB7:  HLT

013632  000241                    CLC                      ;CLEAR CARRY
013634  106072  000002            RORB    @2(2)            ;(R0)=077776, CC=0011
013640  103002                    BCC     RORB7
013642  102001                    BVC     RORB7
013644  100001                    BPL     .+4
013646  104000            RORB7:  HLT

013650  105272  000002            INCB    @2(2)            ;(R0)=100376, CC=1011
013654  103002                    BCC     INCB7
013656  102001                    BVC     INCB7
013660  100401                    BMI     .+4
013662  104000            INCB7:  HLT

013664  105372  177776            DECB    @-2(2)           ;(R0)=100375, CC=1001
013670  103002                    BCC     DECB7
013672  102401                    BVS     DECB7
013674  100401                    BMI     .+4
013676  104000            DECB7:  HLT

013700  106372  000002            ASLB    @2(2)            ;(R0)=000375, CC=0111
013704  103002                    BCC     ASLB7
013706  102001                    BVC     ASLB7
013710  001401                    BEQ     .+4
013712  104000            ASLB7:  HLT
```

```

 013714  000241                            CLC                           ;CLEAR CARRY
 013716  106272  177776                    ASRB    @-2(2)                ;(R0)=000376, CC=1001
 013722  103002                            BCC     ASRB7
 013724  102401                            BVS     ASRB7
 013726  100401                            BMI     .+4
 013730  104000                  ASRB7:    HLT

 013732  105472  000002                    NEGB    @2(2)                 ;(R0)=000376, CC=0100
 013736  103402                            BCS     NEGB7
 013740  102401                            BVS     NEGB7
 013742  001401                            BEQ     .+4
 013744  104000                  NEGB7:    HLT

 013746  000262                            SEV
 013750  106172  177776                    ROLB    @-2(2)                ;(R0)=00374, CC=1001
 013754  103002                            BCC     ROLB7
 013756  102401                            BVS     ROLB7
 013760  100401                            BMI     .+4
 013762  104000                  ROLB7:    HLT

 013764  105272  177776                    INCB    @-2(2)                ;(R0)=000375, CC=1001
 013770  105272  177776                    INCB    @-2(2)                ;(R0)=000376, CC=1001
 013774  105572  177776                    ADCB    @-2(2)                ;(R0)=000377, CC=1000
 014000  105172  177776                    COMB    @-2(2)                ;(R0)=000000, CC=0100
 014004  001401                            BEQ     .+4
 014006  104000                            HLT
                                 ;;************************************************************
                                 ;*TEST 17        CHECK BINARY OPS USING ADDRESS MODE 0
                                 ;;************************************************************
 014010                          TST17:
 014010  000004                            SCOPE
 014012  112737  000017  001202            MOVB    #17,@#$TSTNM
 014020  000277                            SCC                           ;SET CONDITION CODES
 014022  010700                            MOV     PC,R0                 ;R0=PC, CC=X001
 014024  103002                            BCC     MOV0
 014026  102401                            BVS     MOV0
 014030  001001                            BNE     .+4
 014032  104000                  MOV0:     HLT

 014034  010002                            MOV     R0,R2                 ;R2=R0
 014036  000262                            SEV                           ;SET V
 014040  160002                            SUB     R0,R2                 ;R2=000000, CC=0100
 014042  103402                            BCS     SUB0
 014044  102401                            BVS     SUB0
 014046  001401                            BEQ     .+4
 014050  104000                  SUB0:     HLT

 014052  000244                            CLZ
 014054  010203                            MOV     R2,R3                 ;R2=R3=000000, CC=0100
 014056  103401                            BCS     MOV0A
 014060  001401                            BEQ     .+4
 014062  104000                  MOV0A:    HLT

 014064  000257                            CCC
 014066  000272                            +SEV!SEN                      ;SET V & N
```

```
014070 020203                   CMP     R2,R3           ;R2=R3=000000, CC=0100
014072 103403                   BCS     CMP0
014074 102402                   BVS     CMP0
014076 001001                   BNE     CMP0
014100 100001                   BPL     .+4
014102 104000           CMP0:   HLT

014104 010002                   MOV     R0,R2           ;R0=R2
014106 010203                   MOV     R2,R3           ;R0=R2=R3
014110 060203                   ADD     R2,R3           ;R3=2*R0
014112 006302                   ASL     R2              ;R2=2*R0
014114 020203                   CMP     R2,R3           ;R2=R3=2*R0
014116 001401                   BEQ     .+4
014120 104000                   HLT                     ;ERROR! CHECK ADD INSTRUCTION

                        ;THE FOLLOWING SUBTEST SHIFTS A BIT THROUGH R2 AND R5 AND DOES A
                        ;BIT TEST (BIT) USING R2 AND R5.
014122 005002                   CLR     R2
014124 005202                   INC     R2
014126 000402                   BR      2$
014130 006302           1$:     ASL     R2
014132 100407                   BMI     4$
014134 010205           2$:     MOV     R2,R5
014136 000277                   SCC
014140 030205                   BIT     R2,R5           ;R2=R5
014142 103002                   BCC     3$
014144 102401                   BVS     3$
014146 001370                   BNE     1$
014150 104000           3$:     HLT
014152 010205           4$:     MOV     R2,R5
014154 000257                   CCC
014156 030205                   BIT     R2,R5
014160 100401                   BMI     .+4
014162 104000                   HLT

014164 005002                   CLR     R2
014166 000277                   SCC
014170 050002                   BIS     R0,R2
014172 103002                   BCC     BIS0
014174 102401                   BVS     BIS0
014176 001001                   BNE     .+4
014200 104000           BIS0:   HLT

014202 010003                   MOV     R0,R3
014204 000277                   SCC
014206 000244                   CLZ
014210 040003                   BIC     R0,R3
014212 103003                   BCC     BIC0
014214 102402                   BVS     BIC0
014216 001001                   BNE     BIC0
014220 100001                   BPL     .+4
014222 104000           BIC0:   HLT

014224 010004                   MOV     R0,R4
014226 005104                   COM     R4
014230 040004                   BIC     R0,R4
```

```
 014232  005104                           COM     R4
 014234  020004                           CMP     R0,R4
 014236  001401                           BEQ     .+4
 014240  104000                           HLT

 014242  010004                           MOV     R0,R4
 014244  005104                           COM     R4
 014246  010403                           MOV     R4,R3
 014250  050003                           BIS     R0,R3
 014252  103001                           BCC     BISOA
 014254  100401                           BMI     .+4
 014256  104000                   BISOA:  HLT
 014260  005203                           INC     R3
 014262  001401                           BEQ     .+4
 014264  104000                           HLT
 014266  010304                           MOV     R3,R4               ;R3=R4=0
 014270  005103                           COM     R3                  ;R3=177777
 014272  000261                           SEC                         ;SET C
 014274  006004                           ROR     R4                  ;R4=100000
 014276  060304                           ADD     R3,R4               ;R3=177777,R4=077777, CC=0011
 014300  103003                           BCC     ADDO
 014302  102002                           BVC     ADDO
 014304  001401                           BEQ     ADDO
 014306  100001                           BPL     .+4
 014310  104000                   ADDO:   HLT
 014312  010700                           MOV     PC,R0
 014314  022020                           CMP     (R0)+,(R0)+
 014316  020007                           CMP     R0,PC
 014320  001401                           BEQ     .+4
 014322  104000                           HLT

 014324  010700                           MOV     PC,R0
 014326  062700  000010                   ADD     #10,R0
 014332  010002                           MOV     R0,R2
 014334  020700                           CMP     PC,R0
 014336  001002                           BNE     CMPOA
 014340  020200                           CMP     R2,R0
 014342  001401                           BEQ     .+4
 014344  104000                   CMPOA:  HLT
                                  ;;*************************************************************
                                  ;*TEST 20        CHECK BINARY OPS USING ADDRESS MODE 1
                                  ;;*************************************************************
 014346                           TST20:
 014346  000004                           SCOPE
 014350  112737  000020  001202           MOVB    #20,@#$TSTNM
 014356  000402                           BR      .+6                 ;RESERVE TWO WORDS
 014360  000000                           .WORD   0                   ;RESERVED FOR SOURCE DATA
 014362  000000                           .WORD   0                   ;RESERVED FOR DESTINATION DATA
 014364  010704                           MOV     PC,R4
 014366  005744                           TST     -(R4)
 014370  005044                           CLR     -(R4)               ;R4 POINTS TO DESTINATION DATA
 014372  010403                           MOV     R4,R3
 014374  005043                           CLR     -(R3)               ;R3 POINTS TO SOUCE DATA

 014376  005113                           COM     (R3)                ;(R3)=177777
 014400  005214                           INC     (R4)                ;(R4)=000001
```

```
014402  000262                  SEV                     ;SET V
014404  061314                  ADD     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=0101
014406  103002                  BCC     ADD1
014410  102401                  BVS     ADD1
014412  001401                  BEQ     .+4
014414  104000          ADD1:   HLT

014416  000277                  SCC
014420  000250                  CLN
014422  021314                  CMP     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=1000
014424  103403                  BCS     CMP1
014426  102402                  BVS     CMP1
014430  001401                  BEQ     CMP1
014432  100401                  BMI     .+4
014434  104000          CMP1:   HLT

014436  000277                  SCC
014440  000244                  CLZ
014442  031314                  BIT     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=0101
014444  103002                  BCC     BITT1
014446  102401                  BVS     BITT1
014450  001401                  BEQ     .+4
014452  104000          BITT1:  HLT

014454  000277                  SCC
014456  000245                  +CLC!CLZ
014460  005114                  COM     (R4)            ;(R4)=177777
014462  161314                  SUB     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=0100
014464  103402                  BCS     SUB1
014466  102401                  BVS     SUB1
014470  001401                  BEQ     .+4
014472  104000          SUB1:   HLT

014474  105013                  CLRB    (R3)            ;(R3)=177400
014476  000313                  SWAB    (R3)            ;(R3)=000377
014500  000270                  SEN
014502  011314                  MOV     (R3),(R4)       ;(R3)=(R4)=000377
014504  100001                  BPL     .+4
014506  104000                  HLT
014510  000314                  SWAB    (R4)            ;(R3)=000377,(R4)=177400
014512  000263                  +SEC!SEV                ;SET C & V
014514  051314                  BIS     (R3),(R4)       ;(R3)=000377,(R4)=177777, CC=1001
014516  103002                  BCC     BIS1
014520  102401                  BVS     BIS1
014522  100401                  BMI     .+4
014524  104000          BIS1:   HLT

014526  041314                  BIC     (R3),(R4)       ;(R3)=000377,(R4)=177400, CC=1001
014530  103002                  BCC     BIC1
014532  102401                  BVS     BIC1
014534  100401                  BMI     .+4
014536  104000          BIC1:   HLT

014540  000262                  SEV                     ;SET V
014542  021314                  CMP     (R3),(R4)       ;(R3)=000377,(R4)=177400, CC=0001
014544  103003                  BCC     CMP1A
```

```
014546  102402                          BVS     CMP1A
014550  001401                          BEQ     CMP1A
014552  100001                          BPL     .+4
014554  104000                 CMP1A:   HLT

014556  005013                          CLR     (R3)            ;(R3)=000000
014560  000261                          SEC
014562  006013                          ROR     (R3)            ;(R3)=100000
014564  011314                          MOV     (R3),(R4)       ;(R3)=(R4)=100000
014566  005114                          COM     (R4)            ;(R4)=077777
014570  161314                          SUB     (R3),(R4)       ;(R3)=100000,(R4)=177777, CC=1011
014572  103002                          BCC     SUB1A
014574  102001                          BVC     SUB1A
014576  100401                          BMI     .+4
014600  104000                 SUB1A:   HLT

014602  000277                          SCC
014604  161314                          SUB     (R3),(R4)       ;(R3)=100000,(R4)=077777, CC=0000
014606  101402                          BLOS    SUB1B           ;BRANCH IF C OR Z IS SET
014610  102401                          BVS     SUB1B
014612  100001                          BPL     .+4
014614  104000                 SUB1B:   HLT

014616  011314                          MOV     (R3),(R4)       ;(R3)=100000,(R4)=100000, CC=1000
014620  001401                          BEQ     MOV1
014622  100401                          BMI     .+4
014624  104000                 MOV1:    HLT

014626  061314                          ADD     (R3),(R4)       ;(R3)=100000,(R4)=000000, CC=0111
014630  103003                          BCC     ADD1A
014632  102002                          BVC     ADD1A
014634  001001                          BNE     ADD1
014636  100001                          BPL     .+4
014640  104000                 ADD1A:   HLT

014642  005113                          COM     (R3)            ;(R3)=077777
014644  011314                          MOV     (R3),(R4)       ;(R4)=077777
014646  061314                          ADD     (R3),(R4)       ;(R3)=077777,(R4)=177776, CC=1010
014650  103402                          BCS     ADD1B
014652  102001                          BVC     ADD1B
014654  100401                          BMI     .+4
014656  104000                 ADD1B:   HLT

014660  062714  000002                  ADD     #2,(R4)
014664  005714                          TST     (R4)            ;CHECK FINAL RESULT
014666  001401                          BEQ     .+4
014670  104000                          HLT
                               ;;*****************************************************
                               ;*TEST 21        CHECK BINARY BYTE OPS USING ADDRESS MODE 1
                               ;;*****************************************************
014672                         TST21:
014672  000004                          SCOPE
014674  112737  000021  001202          MOVB    #21,@#$TSTNM
014702  000402                          BR      .+6
014704  000000                          .WORD   0
014706  000000                          .WORD   0
```

```
014710 010705                  MOV     PC,R5
014712 005745                  TST     -(R5)
014714 005045                  CLR     -(R5)           ;(R5)=000000
014716 010502                  MOV     R5,R2
014720 005042                  CLR     -(R2)           ;(R2)=000000
014722 005202                  INC     R2              ;R2 POINTS TO ODD BYTE
014724 105112                  COMB    (R2)            ;(R2)=177400

014726 000277                  SCC
014730 111215                  MOVB    (R2),(R5)       ;(R2)=177400,(R5)=000377,CC=1001
014732 103005                  BCC     MOVB1
014734 102404                  BVS     MOVB1
014736 001403                  BEQ     MOVB1
014740 100002                  BPL     MOVB1
014742 105215                  INCB    (R5)            ;CHECK RESULT
014744 001401                  BEQ     .+4
014746 104000          MOVB1:  HLT

014750 106312                  ASLB    (R2)            ;SHIFT (R2) UNTIL
014752 102376                  BVC     .-2             ;(R2)=000000
014754 106012                  RORB    (R2)            ;(R2)=100000
014756 105315                  DECB    (R5)            ;(R5)=00377
014760 106015                  RORB    (R5)            ;(R5)=000177
014762 000257                  CCC
014764 121512                  CMPB    (R5),(R2)       ;(R5)=000177,(R2)=100000, CC=1010
014766 102001                  BVC     CMPB1
014770 100401                  BMI     .+4
014772 104000          CMPB1:  HLT

014774 005003                  CLR     R3
014776 000261                  SEC
015000 006003                  ROR     R3              ;R3=100000
015002 050315                  BIS     R3,(R5)         ;(R5)=100177
015004 000273                  +SEC!SEV!SEN            ;SET C,V, & N
015006 131215                  BITB    (R2),(R5)       ;(R2)=100000,(R5)=100177, CC=0101
015010 103002                  BCC     BITB1
015012 102401                  BVS     BITB1
015014 001401                  BEQ     .+4
015016 104000          BITB1:  HLT

015020 151215                  BISB    (R2),(R5)       ;(R2)=100000,(R5)=100377, CC=1001
015022 103001                  BCC     BISB1
015024 100401                  BMI     .+4
015026 104000          BISB1:  HLT

015030 141215                  BICB    (R2),(R5)       ;(R2)=100000,(R5)=100177, CC=0001
015032 103002                  BCC     BICB1
015034 001401                  BEQ     BICB1
015036 100001                  BPL     .+4
015040 104000          BICB1:  HLT

015042 105112                  COMB    (R2)            ;(R2)=077400,(R5)=100177
015044 121215                  CMPB    (R2),(R5)
015046 001401                  BEQ     .+4
015050 104000                  HLT
```

```
015052  141512                 BICB    (R5),(R2)       ;(R5)=100177,(R2)=000000, CC=0100
015054  001002                 BNE     BICB1A
015056  105712                 TSTB    (R2)
015060  001401                 BEQ     .+4
015062  104000         BICB1A: HLT

015064  000402                 BR      .+6             ;RESERVE TWO WORDS FOR DATA
015066  000000                 .WORD   0               ;SOURCE DATA
015070  000000                 .WORD   0               ;DEST DATA
015072  010705                 MOV     PC,R5
015074  005745                 TST     -(R5)
015076  105045                 CLRB    -(R5)           ;R5 POINTS TO DEST ODD BYTE
015100  010504                 MOV     R5,R4
015102  105044                 CLRB    -(R4)           ;R4 POINTS TO DEST EVEN BYTE
015104  010403                 MOV     R4,R3
015106  105043                 CLRB    -(R3)           ;R3 POINTS TO SOURCE ODD BYTE
015110  010302                 MOV     R3,R2
015112  105042                 CLRB    -(R2)           ;R2 POINTS TO SOURCE EVEN BYTE

                       ;COMMENTS ARE LEAST SIGNIFICANT 4 BITS OF BYTES POINTED TO BY R2,R3
                       ;R4, AND R5 RESPECTIVELY AND THE REMAINING BITS ARE 0'S.
015114  000261                 SEC                     ;SET CARRY
                                                       ;(R2),(R3),(R4),(R5)
015116  106112                 ROLB    (R2)            ;0001,0000,0000,0000
015120  111214                 MOVB    (R2),(R4)       ;0001,0000,0001,0000
015122  106112                 ROLB    (R2)            ;0010,0000,0001,0000
015124  111213                 MOVB    (R2),(R3)       ;0010,0010,0001,0000
015126  106112                 ROLB    (R2)            ;0100,0010,0001,0000
015130  111315                 MOVB    (R3),(R5)       ;0100,0010,0001,0010
015132  106112                 ROLB    (R2)            ;1000,0010,0001,0010
015134  106113                 ROLB    (R3)            ;1000,0100,0001,0010
015136  151215                 BISB    (R2),(R5)       ;1000,0100,0001,1010
015140  131512                 BITB    (R5),(R2)       ;1000,0100,0001,1010
015142  001426                 BEQ     BIN1
015144  151314                 BISB    (R3),(R4)       ;1000,0100,0101,1010
015146  131413                 BITB    (R4),(R3)       ;1000,0100,0101,1010
015150  001423                 BEQ     BIN1
015152  105213                 INCB    (R3)            ;1000,0101,0101,1010
015154  121314                 CMPB    (R3),(R4)       ;1000,0101,0101,1010
015156  001020                 BNE     BIN1
015160  106113                 ROLB    (R3)            ;1000,1010,0101,1010
015162  121315                 CMPB    (R3),(R5)       ;1000,1010,0101,1010
015164  001015                 BNE     BIN1
015166  106212                 ASRB    (R2)            ;0100,1010,0101,1010
015170  131214                 BITB    (R2),(R4)       ;0100,1010,0101,1010
015172  001412                 BEQ     BIN1
015174  106015                 RORB    (R5)            ;0100,1010,0101,0101
015176  121415                 CMPB    (R4),(R5)       ;0100,1010,0101,0101
015200  001007                 BNE     BIN1
015202  105314                 DECB    (R4)            ;0100,1010,0100,0101
015204  141214                 BICB    (R2),(R4)       ;0100,1010,0000,0101
015206  001004                 BNE     BIN1
015210  111314                 MOVB    (R3),(R4)       ;0100,1010,1010,0101
015212  106213                 ASRB    (R3)            ;0100,0101,1010,0101
015214  141315                 BICB    (R3),(R5)       ;0100,0101,1010,0101
015216  001401                 BEQ     .+4
```

```
015220  104000                          BIN1:   HLT
                                        ;;*****************************************************************
                                        ;*TEST 22       CHECK BINARY WORD OPS USING ADDRESS MODE 2 & 4
                                        ;;*****************************************************************
015222                                  TST22:
015222  000004                                  SCOPE
015224  112737  000022  001202                  MOVB    #22,@#STSTNM
015232  012704  015070                          MOV     #BICB1A+6,R4
015236  012702  015066                          MOV     #BICB1A+4,R2
015242  063702  001550                          ADD     @#FACTOR,R2
015246  063704  001550                          ADD     @#FACTOR,R4
015252  010405                                  MOV     R4,R5           ;SET DESTINATION REGISTER
015254  012715  000001                          MOV     #1,(R5)
015260  012712  177777                          MOV     #-1,(R2)
015264  000257                                  CCC
015266  000262                                  SEV
015270  062225                                  ADD     (R2)+,(R5)+     ;(R2)=177777,(R5)=000000, CC=0101
015272  103002                                  BCC     ADD2
015274  102401                                  BVS     ADD2
015276  001401                                  BEQ     .+4
015300  104000                          ADD2:   HLT

015302  000262                                  SEV                     ;SET V
015304  024527  000001                          CMP     -(R5),#1        ;(R5)=000000, CC=1001
015310  103002                                  BCC     CMP2
015312  102401                                  BVS     CMP2
015314  100401                                  BMI     .+4
015316  104000                          CMP2:   HLT

015320  054225                                  BIS     -(R2),(R5)+     ;(R2)=177777,(R5)=177777,CC=1001
015322  103001                                  BCC     BIS2
015324  100401                                  BMI     .+4
015326  104000                          BIS2:   HLT
015330  000277                                  SCC
015332  000244                                  CLZ
015334  162245                                  SUB     (R2)+,-(R5)     ;(R2)=177777,(R5)=000000, CC=0100
015336  103402                                  BCS     SUB2
015340  102401                                  BVS     SUB2
015342  001401                                  BEQ     .+4
015344  104000                          SUB2:   HLT

015346  005442                                  NEG     -(R2)           ;(R2)=000001
015350  005115                                  COM     (R5)            ;(R5)=177777
015352  000277                                  SCC
015354  000250                                  CLN
015356  042225                                  BIC     (R2)+,(R5)+     ;(R2)=000001,(R5)=177776, CC=1001
015360  103003                                  BCC     BIC2
015362  102402                                  BVS     BIC2
015364  001401                                  BEQ     BIC2
015366  100401                                  BMI     .+4
015370  104000                          BIC2:   HLT

015372  012742  125252                          MOV     #125252,-(R2)
015376  012245                                  MOV     (R2)+,-(R5)
015400  005125                                  COM     (R5)+           ;(R5)=052525
015402  000262                                  SEV
```

```
015404  034245                           BIT     -(R2),-(R5)       ;(R2)=125252,(R5)=052525, CC=0101
015406  103002                           BCC     BITT2
015410  102401                           BVS     BITT2
015412  001401                           BEQ     .+4
015414  104000                  BITT2:   HLT

015416  000262                           SEV
015420  052225                           BIS     (R2)+,(R5)+       ;(R2)=125252,(R5)=177777, CC=1001
015422  103002                           BCC     BIS2A
015424  102401                           BVS     BIS2A
015426  100401                           BMI     .+4
015430  104000                  BIS2A:   HLT

015432  042745  125252                   BIC     #125252,-(R5)     ;(R5)=052525
015436  005125                           COM     (R5)+             ;(R5)=125252
015440  024245                           CMP     -(R2),-(R5)
015442  001401                           BEQ     .+4
015444  104000                           HLT

015446  005012                           CLR     (R2)
015450  005122                           COM     (R2)+             ;(R2)=177777
015452  162742  000001                   SUB     #1,-(R2)          ;(R2)=177776, CC=1000
015456  103402                           BCS     SUB2A
015460  102401                           BVS     SUB2A
015462  100401                           BMI     .+4
015464  104000                  SUB2A:   HLT
015466  010702                           MOV     PC,R2             ;GET CURRENT PC
015470  010205                           MOV     R2,R5             ;MOVE TO R5
015472  124245                  1$:      CMPB    -(R2),-(R5)       ;COMPARE ALL PREVIOUS MEMORY ADDRESSES
015474  001401                           BEQ     .+4
015476  104000                           HLT                       ;ERROR!
015500  020237  001554                   CMP     R2,@#FRSTAD       ;CHECK FOR LOW LIMIT
015504  001372                           BNE     1$
                                ;;****************************************************************
                                ;*TEST 23        CHECK BINARY BYTE OPS USING ADDRESS MODE 2 & 4
                                ;;****************************************************************
015506                          TST23:
015506  000004                           SCOPE
015510  112737  000023  001202           MOVB    #23,@#$TSTNM
015516  000402                           BR      .+6               ;RESERVE TWO WORDS
015520  000000                           .WORD   0                 ;SOURCE DATA
015522  000000                           .WORD   0                 ;DESTINATION DATA
015524  010703                           MOV     PC,R3
015526  005743                           TST     -(R3)

                                ;FIRST CHECK AUTO INCREMENT/DECREMENT
015530  010300                           MOV     R3,R0
015532  010002                           MOV     R0,R2
015534  005302                           DEC     R2
015536  010604                           MOV     SP,R4
015540  010605                           MOV     SP,R5
015542  005745                           TST     -(R5)

015544  114046                           MOVB    -(R0),-(SP)
015546  020506                           CMP     R5,SP
015550  001021                           BNE     BINB
```

```
015552  020200                          CMP     R2,R0
015554  001017                          BNE     BINB
015556  122026                          CMPB    (R0)+,(SP)+
015560  020406                          CMP     R4,SP
015562  001014                          BNE     BINB
015564  020003                          CMP     R0,R3
015566  001012                          BNE     BINB
015570  154640                          BISB    -(SP),-(R0)
015572  020506                          CMP     R5,SP
015574  001007                          BNE     BINB
015576  020200                          CMP     R2,R0
015600  001005                          BNE     BINB
015602  142620                          BICB    (SP)+,(R0)+
015604  020406                          CMP     R4,SP
015606  001002                          BNE     BINB
015610  020003                          CMP     R0,R3
015612  001401                          BEQ     .+4
015614  104000                  BINB:   HLT
015616  010003                          MOV     R0,R3
015620  112743  000200                  MOVB    #200,-(R3)
015624  112743  000377                  MOVB    #377,-(R3)      ;(R3)=100377
015630  010304                          MOV     R3,R4
015632  112744  000177                  MOVB    #177,-(R4)
015636  112744  000000                  MOVB    #0,-(R4)        ;(R4)=077400
015642  001401                          BEQ     .+4
015644  104000                          HLT

015646  152324                          BISB    (R3)+,(R4)+     ;(R3)=100377,(R4)=077777
015650  100401                          BMI     .+4
015652  104000                          HLT

015654  122324                          CMPB    (R3)+,(R4)+
015656  103402                          BCS     CMPB2
015660  102001                          BVC     CMPB2
015662  100001                          BPL     .+4
015664  104000                  CMPB2:  HLT

015666  000261                          SEC
015670  134344                          BITB    -(R3),-(R4)
015672  103002                          BCC     BITB2
015674  102401                          BVS     BITB2
015676  001401                          BEQ     .+4
015700  104000                  BITB2:  HLT

015702  000244                          CLZ
015704  144344                          BICB    -(R3),-(R4)     ;(R3)=100377,(R4)=077400
015706  001401                          BEQ     .+4
015710  104000                          HLT
                                ;;**************************************************************
                                ;*TEST 24       CHECK BINARY WORD OPS USING ADDRESS MODES 3 & 5
                                ;;**************************************************************
015712                          TST24:
015712  000004                          SCOPE
015714  112737  000024  001202          MOVB    #24,@#$TSTNM
015722  000404                          BR      2$              ;RESERVE SPACE FOR DATA AND ADDRESSES
015724  000000                          .WORD   0               ;CONTAINS ADDRESS OF SOURCE DATA
```

```
015726  000000                      .WORD   0                   ;CONTAINS ADDRESS OF DEST DATA
015730  000000                      .WORD   0                   ;CONTAINS SOURCE DATA
015732  000000                      .WORD   0                   ;CONTAINS DEST DATA
015734  010701              2$:     MOV     PC,R1
015736  010100                      MOV     R1,R0               ;SET SCOPE PTR
015740  024040                      CMP     -(R0),-(R0)         ;ADJUST R0
015742  010005                      MOV     R0,R5               ;R5 POINTS TO DEST DATA
015744  024545                      CMP     -(R5),-(R5)         ;SUB 4 FROM R5
015746  010015                      MOV     R0,(R5)             ;R5 POINTS TO ADDRESS OF DEST DATA
015750  010502                      MOV     R5,R2
015752  010004                      MOV     R0,R4               ;R4 POINTS TO DEST DATA
015754  005740                      TST     -(R0)
015756  010003                      MOV     R0,R3               ;R3 POINTS TO SOURCE DATA
015760  010042                      MOV     R0,-(R2)            ;R2 POINTS TO ADDRESS OF SOURCE DATA
015762  005013                      CLR     (R3)                ;PRESET SOURCE DATA
015764  005014                      CLR     (R4)                ;PRESET DEST DATA

015766  000277                      SCC
015770  000244                      CLZ
015772  163235                      SUB     @(R2)+,@(R5)+       ;(R3)=000000,(R4)=000000, CC=0100
015774  103402                      BCS     SUB3
015776  102401                      BVS     SUB3
016000  001401                      BEQ     .+4
016002  104000              SUB3:   HLT

016004  052752  100000              BIS     #100000,@-(R2)      ;(R3)=100000
016010  062755  000001              ADD     #1,@-(R5)           ;(R4)=000001
016014  163235                      SUB     @(R2)+,@(R5)+       ;(R3)=100000,(R4)=100001, CC=1011
016016  103002                      BCC     SUB3A
016020  102001                      BVC     SUB3A
016022  100401                      BMI     .+4
016024  104000              SUB3A:  HLT

016026  005414                      NEG     (R4)                ;(R4)=077777
016030  035255                      BIT     @-(R2),@-(R5)       ;(R3)=100000,(R4)=077777
016032  001401                      BEQ     .+4
016034  104000                      HLT
016036  023235                      CMP     @(R2)+,@(R5)+
016040  102401                      BVS     .+4
016042  104000                      HLT
016044  005152                      COM     @-(R2)
016046  000257                      CCC
016050  063255                      ADD     @(R2)+,@-(R5)
016052  102001                      BVC     ADD3
016054  100401                      BMI     .+4
016056  104000              ADD3:   HLT
016060  000261                      SEC
016062  045235                      BIC     @-(R2),@(R5)+       ;(R3)=077777,(R4)=100000
016064  103001                      BCC     BIC3
016066  100401                      BMI     .+4
016070  104000              BIC3:   HLT

016072  005155                      COM     @-(R5)              ;(R4)=077777
016074  023235                      CMP     @(R2)+,@(R5)+       ;(R3)=077777,(R4)=077777
016076  001401                      BEQ     .+4
016100  104000                      HLT
```

```
                                    ;;*************************************************************
                                    ;*TEST 25       CHECK BINARY BYTE OPS USING ADDRESS MODES 3 & 5
                                    ;;*************************************************************
016102                              TST25:
016102  000004                              SCOPE
016104  112737  000025  001202              MOVB    #25,@#$TSTNM
016112  000406                              BR      1$              ;RESERVE SPACE FOR ADDRESS AND DATA
016114  000000                              .WORD   0               ;CONTAINS ADDRESS OF SOURCE DATA (EVEN BYTE)
016116  000000                              .WORD   0               ;CONTAINS ADDRESS OF SOURCE DATA (ODD BYTE)
016120  000000                              .WORD   0               ;CONTAINS ADDRESS OF DEST DATA (EVEN BYTE)
016122  000000                              .WORD   0               ;CONTAINS ADDRESS OF DEST DATA (ODD BYTE)
016124  000000                              .WORD   0               ;CONTAINS SOURCE DATA
016126  000000                              .WORD   0               ;CONTAINS DEST DATA

016130  010700                      1$:     MOV     PC,R0
016132  024040                              CMP     -(R0),-(R0)     ;R0=ADDRESS OF DEST DATA
016134  010003                              MOV     R0,R3           ;R3     "       "
016136  010305                              MOV     R3,R5           ;R5     "       "
016140  005743                              TST     -(R3)           ;SUB 2 FROM R3
016142  010043                              MOV     R0,-(R3)        ;R3 POINTS TO ADDRESS OF DEST DATA
016144  005213                              INC     (R3)            ;ODD BYTE
016146  010043                              MOV     R0,-(R3)        ;EVEN BYTE
016150  010304                              MOV     R3,R4
016152  005740                              TST     -(R0)           ;R0=ADDRESS OF SOURCE DATA
016154  010044                              MOV     R0,-(R4)        ;R4 POINTS TO ADDRESS OF SOURCE DATA
016156  005214                              INC     (R4)            ;ODD BYTE
016160  010044                              MOV     R0,-(R4)        ;EVEN BYTE

016162  000261                              SEC                     ;SET CARRY
016164  012734  177001                      MOV     #177001,@(R4)+
016170  112734  000200                      MOVB    #200,@(R4)+     ;SOURCE DATA=100001
016174  115433                              MOVB    @-(R4),@(R3)+
016176  115433                              MOVB    @-(R4),@(R3)+   ;DEST DATA=000600
016200  103401                              BCS     .+4
016202  104000                              HLT                     ;ERROR! MOV DOES AFFECT C BIT IN PSW
016204  022715  000600                      CMP     #600,(R5)       ;CHECK DEST DATA
016210  001401                              BEQ     .+4
016212  104000                              HLT                     ;ERROR! INCORRECT RESULT
016214  024343                              CMP     -(R3),-(R3)     ;POINT R4 BACK TO EVEN BYTE
016216  153433                              BISB    @(R4)+,@(R3)+
016220  153433                              BISB    @(R4)+,@(R3)+   ;DEST DATA=100601
016222  022715  100601                      CMP     #100601,(R5)    ;CHECK RESULT
016226  001401                              BEQ     .+4
016230  104000                              HLT                     ;ERROR! INCORRECT DEST DATA AFTER BISB
016232  145453                              BICB    @-(R4),@-(R3)
016234  145453                              BICB    @-(R4),@-(R3)
016236  133433                              BITB    @(R4)+,@(R3)+
016240  001002                              BNE     BITB3
016242  135433                              BITB    @-(R4),@(R3)+
016244  001001                              BNE     .+4
016246  104000                      BITB3:  HLT

016250  123453                              CMPB    @(R4)+,@-(R3)
016252  001002                              BNE     CMPB3
016254  123453                              CMPB    @(R4)+,@-(R3)
016256  001401                              BEQ     .+4
```

```
016260  104000                    CMPB3:  HLT
                                  ;;**********************************************************
                                  ;*TEST 26       CHECK BINARY OPS USING ADDRESS MODE 6
                                  ;;**********************************************************
016262                            TST26:
016262  000004                            SCOPE
016264  112737  000026  001202            MOVB    #26,@#$TSTNM
016272  000402                            BR      .+6                     ;RESERVE TWO LOCATIONS
016274  000000                    SDATA:  .WORD   0                       ;RESERVED FOR SOURCE DATA
016276  000000                    DDATA:  .WORD   0                       ;RESERVED FOR DESTINATION DATA

016300  013702  001550                    MOV     @#FACTOR,R2             ;GET RELOCATION FACTOR AND USE AS AN
016304  010205                            MOV     R2,R5                   ;INDEX VALUE TO POINT TO DATA
016306  005065  016276                    CLR     DDATA(5)                ;PRESET DESTINATION DATA
016312  012762  000001  016274            MOV     #1,SDATA(2)             ;THIS ROUTSINE PUT A 1 BIT INTO EVERY
016320  056265  016274  016276    1$:     BIS     SDATA(2),DDATA(5)               ;OTHER BIT POSITION IN THE DEST-
016326  006362  016274                    ASL     SDATA(2)                        ;INATION ADDRESS (52525)
016332  006362  016274                    ASL     SDATA(2)
016336  103370                            BCC     1$
016340  022765  052525  016276            CMP     #52525,DDATA(5) ;CHECK RESULT
016346  001401                            BEQ     .+4
016350  104000                            HLT                             ;ERROR! INCORRECT RESULT
016352  012762  177777  016274            MOV     #-1,SDATA(2)
016360  046562  016276  016274            BIC     DDATA(5),SDATA(2)               ;SOURCE DATA=125252
016366  036265  016274  016276            BIT     SDATA(2),DDATA(5)
016374  001401                            BEQ     .+4
016376  104000                            HLT                             ;ERROR! BIT INST FAILED
016400  006365  016276                    ASL     DDATA(5)                ;DDATA=125252
016404  026265  016274  016276            CMP     SDATA(2),DDATA(5)
016412  001401                            BEQ     .+4

016414  104000                            HLT                             ;ERROR! CMP INST FAILED
016416  000257                            CCC
016420  066265  016274  016276            ADD     SDATA(2),DDATA(5)
016426  103002                            BCC     ADD6
016430  102001                            BVC     ADD6
016432  100001                            BPL     .+4
016434  104000                    ADD6:   HLT

016436  006362  016274                    ASL     SDATA(2)                ;SDATA=52524
016442  166265  016274  016276            SUB     SDATA(2),DDATA(5)
016450  103401                            BCS     SUB6
016452  001401                            BEQ     .+4
016454  104000                    SUB6:   HLT

016456  112700  000377                    MOVB    #377,R0                 ;R0=177777 (MOVB %R EXTENDS SIGN)
016462  010062  016274                    MOV     R0,SDATA(2)
016466  012765  177777  016276            MOV     #-1,DDATA(5)
016474  166500  016276                    SUB     DDATA(5),R0
016500  001401                            BEQ     .+4
016502  104000                            HLT
016504  066265  016274  016276    1$:     ADD     SDATA(2),DDATA(5)
016512  006362  016274                    ASL     SDATA(2)
016516  005162  016274                    COM     SDATA(2)
016522  036265  016274  016276            BIT     SDATA(2),DDATA(5)
016530  001401                            BEQ     .+4
```

```
016532  104000                                   HLT
016534  005162  016274                           COM     SDATA(2)
016540  026265  016274  016276                   CMP     SDATA(2),DDATA(5)
016546  001401                                   BEQ     .+4
016550  104000                                   HLT
016552  026200  016274                           CMP     SDATA(2),R0
016556  001352                                   BNE     1$
                                         ;;*********************************************************
                                         ;*TEST 27        CHECK BINARY BYTE OPS USING ADDRESS MODE 6
                                         ;;*********************************************************
016560                                   TST27:
016560  000004                                   SCOPE
016562  112737  000027  001202                   MOVB    #27,@#$TSTNM
                                         ;NOTE: SDATAB(2), AND DDATAB(4) REFERENCE EVEN BYTE OF SOURCE & DEST DATA
                                         ;AND SDATAB(3), AND DDATAB(5) FEFERENCE ODD BYTE OF SOURCE & DEST DATA

016570  013702  001550                           MOV     @#FACTOR,R2         ;GET INDEX VALUE
016574  010204                                   MOV     R2,R4               ;R2 FOR SOUPCE EVEN BYTE INDEX, R4 FOR
016576  010403                                   MOV     R4,R3               ;DEST ODD BYTE, R3 FOR SOURCE EVEN
016600  005203                                   INC     R3                  ;AND R5 FOR DEST ODD BYTE
016602  010305                                   MOV     R3,R5
016604  000261                                   SEC                                 ;SET CARRY
016606  012762  125252  016730                   MOV     #125252,SDATAB(2)
016614  112763  177125  016730                   MOVB    #177125,SDATAB(3)           ;SOURCE DATA = 052652
016622  016264  016730  016732                   MOV     SDATAB(2),DDATAB(4)
016630  052764  125125  016732                   BIS     #125125,DDATAB(4)           ;DEST DATA = 177777
016636  136263  016730  016730                   BITB    SDATAB(2),SDATAB(3)
016644  001401                                   BEQ     .+4
016646  104000                           BITB6:  HLT

016650  146264  016730  016732                   BICB    SDATAB(2),DDATAB(4)
016656  103401                                   BCS     .+4
016660  104000                                   HLT                 ;ERROR MOV,BIS,BIT;BIC DO NOT AFFECT 'C'
016662  126364  016730  016732                   CMPB    SDATAB(3),DDATAB(4)
016670  001401                                   BEQ     .+4
016672  104000                                   HLT

016674  146365  016730  016732                   BICB    SDATAB(3),DDATAB(5)
016702  126265  016730  016732                   CMPB    SDATAB(2),DDATAB(5)
016710  001401                                   BEQ     .+4
016712  104000                                   HLT

016714  136564  016732  016732                   BITB    DDATAB(5),DDATAB(4)
016722  001401                                   BEQ     .+4
016724  104000                                   HLT
016726  000402                                   BR      TST30               ;;SKIP TO NEXT TEST
016730  000000                           SDATAB: .WORD   0                   ;RESERVED FOR SOURCE DATA
016732  000000                           DDATAB: .WORD   0                   ;RESERVED FOR DEST DATA

                                         ;;*********************************************************
                                         ;*TEST 30        CHECK BINARY WORD OPS USING ADDRESS MODE 7
                                         ;*       R2=ADDRESS OF SOURCE DATA, AND R3= ADDRESS OF DEST DATA
                                         ;;*********************************************************
016734                                   TST30:
016734  000004                                   SCOPE
016736  112737  000030  001202                   MOVB    #30,@#$TSTNM
```

```
016744 000404                            BR     UB7             ;RESERVE FOUR WORDS
016746 000000                    SBIN7:  .WORD  0               ;CONTAINS ADDRESS OF SOURCE DATA
016750 000000                    DBIN7:  .WORD  0               ;CONTAINS ADDRESS OF DEST DATA
016752 000000                            .WORD  0               ;CONTAINS SOURCE DATA
016754 000000                            .WORD  0               ;CONTAINS DEST DATA

016756 010700                    UB7:    MOV    PC,R0
016760 024040                            CMP    -(R0),-(R0)
016762 010002                            MOV    R0,R2
016764 024242                            CMP    -(R2),-(R2)
016766 010012                            MOV    R0,(R2)
016770 010203                            MOV    R2,R3
016772 024043                            CMP    -(R0),-(R3)
016774 010013                            MOV    R0,(R3)

016776 000261                            SEC
017000 012777 100000 177740              MOV    #100000,@SBIN7  ;SOURCE DATA = 100000
017006 017777 177734 177734              MOV    @SBIN7,@DBIN7   ;DEST DATA = 100000
017014 103001                            BCC    MOV7
017016 100401                            BMI    .+4
017020 104000                    MOV7:   HLT
017022 006377 177722                     ASL    @DBIN7          ;DEST DATA = 000000
017026 102001                            BVC    .+4
017030 001401                            BEQ    .+4
017032 104000                            HLT

017034 027777 177706 177706              CMP    @SBIN7,@DBIN7   ;(R2)=100000,(R3)=000000
017042 103402                            BCS    CMP7
017044 102401                            BVS    CMP7
017046 100401                            BMI    .+4
017050 104000                    CMP7:   HLT

017052 167777 177670 177670              SUB    @SBIN7,@DBIN7   ;(R2)=100000,(R3)=100000
017060 103003                            BCC    SUB7
017062 102002                            BVC    SUB7
017064 001401                            BEQ    SUB7
017066 100401                            BMI    .+4
017070 104000                    SUB7:   HLT

017072 006277 177650                     ASR    @SBIN7          ;(R2)=140000
017076 067777 177644 177644              ADD    @SBIN7,@DBIN7   ;(R2)=140000,(R3)=040000
017104 103003                            BCC    ADD7
017106 102002                            BVC    ADD7
017110 001401                            BEQ    ADD7
017112 100001                            BPL    .+4
017114 104000                    ADD7:   HLT

017116 047777 177624 177624              BIC    @SBIN7,@DBIN7   ;(R2)=140000,(R3)=000000
017124 001401                            BEQ    .+4
017126 104000                            HLT

017130 057777 177612 177612              BIS    @SBIN7,@DBIN7   ;(R2)=140000,(R3)=140000
017136 100401                            BMI    .+4
017140 104000                            HLT

017142 027777 177600 177600              CMP    @SBIN7,@DBIN7
```

```
017150  001401                                  BEQ     .+4
017152  104000                                  HLT
                                        ;;*************************************************************
                                        ;*TEST 31        SOME MISCELLANEOUS OPERATIONS INVOLVING THE PC
                                        ;*       NOTE: NONE OF THESE OPERATIONS SHOULD AFFECT THE PC
                                        ;;*************************************************************
017154                                  TST31:
017154  000004                                  SCOPE
017156  112737  000031  001202                  MOVB    #31,@#$TSTNM
017164  005000                                  CLR     R0
017166  005067  000072                          CLR     1$
017172  010707                                  MOV     PC,PC
017174  120707                                  CMPB    PC,PC
017176  030707                                  BIT     PC,PC
017200  060007                                  ADD     R0,PC
017202  105707                                  TSTB    PC
017204  005507                                  ADC     PC
017206  021007                                  CMP     (R0),PC
017210  131007                                  BITB    (R0),PC
017212  062707  000000                          ADD     #0,PC
017216  023707  001550                          CMP     @#FACTOR,PC
017222  133707  001550                          BITB    @#FACTOR,PC
017226  000240                                  NOP
                                        ;THE NEXT TWO INSTRUCTION CAUSE THE PROGRAM TO JUMP TO THE UNRELOCATED
                                        ;CODE AND TO RETURN ON THE FOLLOWING INST (IF THE CODE IS RELOCATED)
017230  163707  001550                          SUB     @#FACTOR,PC     ;JUMPS TO UNRELOCATED CODE
017234  063707  001550                          ADD     @#FACTOR,PC     ;RETURNS
017240  000240                                  NOP
017242  024607                                  CMP     -(SP),PC
017244  132607                                  BITB    (SP)+,PC
017246  026707  000012                          CMP     1$,PC
017252  166707  000006                          SUB     1$,PC
017256  046707  000002                          BIC     1$,PC
017262  000401                                  BR      .+4             ;BRANCH OVER 1$
017264  000000                          1$:     0
017266  000004                          RELE2:  SCOPE
017270  010702                                  MOV     PC,R2
017272  062702  000012                          ADD     #12,R2
017276  012707  036574                          MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
017302  000000                          REL22:  .WORD   0
                                        ;2222222222222 LAST ADDRESS OF CODE TO BE RELOCATED 2222222222

                                        ;;*************************************************************
                                        ;*TEST 32        CHECK BINARY BYTE OPS USING ADDRESS MODE 0
                                        ;;*************************************************************
017304                                  TST32:
017304  012767  000001  162012                  MOV     #1,$TIMES       ;;DO 1 ITERATION
017312  000004                                  SCOPE
017314  112737  000032  001202                  MOVB    #32,@#$TSTNM

                                                .SBTTL  START OF SECTION 3
                                        ;3333333333333 FIRST ADDRESS TO BE RELOCATED 333333333
017322  112737  000032  001202          REL3:   MOVB    #$TN-1,@#$TSTNM
017330  010700                                  MOV     PC,R0           ;GET PC
017332  005740                                  TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL3
017334  010037  001554                          MOV     R0,@#FRSTAD     ;SAVE
```

```
017340  010700                           MOV     PC,R0               ;GET CURRENT PC
017342  162700  017342                   SUB     #.,R0               ;SUBTRACT RELOCATION FACTOR
017346  010037  001550                   MOV     R0,@#FACTOR         ;SAVE RELOCATION FACTOR
017352  010737  001212                   MOV     PC,@#$LPERR         ;SET LOOP ADDRESS
017356  062737  000026  001212           ADD     #26,@#$LPERR        ;ADJUST
017364  013737  001212  001210           MOV     @#$LPERR,@#$LPADR
017372  105737  001544                   TSTB    @#NEXEC             ;BR IF TEST CODE TO BE EXECUTED
017376  001402                           BEQ     .+6
017400  000167  002230                   JMP     RELE3
017404  012703  125252                   MOV     #125252,R3
017410  010304                           MOV     R3,R4               ;R3=R4=125252
017412  140304                           BICB    R3,R4               ;R3=125252,R4=125000
017414  022704  125000                   CMP     #125000,R4          ;CHECK RESULT
017420  001401                           BEQ     .+4
017422  104000                           HLT

017424  005004                           CLR     R4                  ;R3=125252,R4=0
017426  150304                           BISB    R3,R4               ;R3=125252,R4=000252
017430  022704  000252                   CMP     #252,R4             ;CHECK RESULT
017434  001401                           BEQ     .+4
017436  104000                           HLT

017440  110404                           MOVB    R4,R4               ;R4=177652
017442  022704  177652                   CMP     #177652,R4          ;CHECK RESULT
017446  001401                           BEQ     .+4
017450  104000                           HLT

017452  132704  177525                   BITB    #177525,R4
017456  001401                           BEQ     .+4
017460  104000                           HLT

017462  105104                           COMB    R4                  ;R4=177525
017464  110404                           MOVB    R4,R4               ;R4=000125
017466  022704  000125                   CMP     #125,R4             ;CHECK RESULT
017472  001401                           BEQ     .+4
017474  104000                           HLT

017476  150304                           BISB    R3,R4               ;R3=125252,R4=000377
017500  105204                           INCB    R4
017502  001401                           BEQ     .+4
017504  104000                           HLT
                                ;;*****************************************************
                                ;*TEST 33        CHECK BINARY BYTE OPS USING ADDRESS MODE 7
                                ;;*****************************************************
017506                          TST33:
017506  000004                           SCOPE
017510  112737  000033  001202           MOVB    #33,@#$TSTNM
017516  000406                           BR      BINB7               ;RESERVE SPACE FOR ADDRESSES & DATA
017520  000000                  SBINB7:  .WORD   0                   ;CONTAINS ADDRESS OF SOURCE EVEN BYTE
017522  000000                           .WORD   0                   ;CONTAINS ADDRESS OF SOURCE ODD BYTE
017524  000000                           .WORD   0                   ;CONTAINS ADDRESS OF DEST EVEN BYTE
017526  000000                           .WORD   0                   ;CONTAINS ADDRESS OF DEST ODD BYTE
017530  000000                  DBINB7:  .WORD   0                   ;CONTAINS SOURCE DATA
017532  000000                           .WORD   0                   ;CONTAINS DEST DATA

017534  010700                  BINB7:   MOV     PC,R0
```

```
017536  024040                          CMP     -(R0),-(R0)         ;R0 = ADDRESS OF DEST DATA
017540  010060  177772                  MOV     R0,-6(R0)           ;LOAD ADDRES OF DEST EVEN BYTE DATA
017544  010060  177774                  MOV     R0,-4(R0)
017550  005260  177774                  INC     -4(R0)              ;LOAD ADDRESS OF DEST ODD BYTE DATA
017554  005740                          TST     -(R0)               ;R0=ADDRESS OF SOURCE DATA
017556  010060  177770                  MOV     R0,-10(R0)          ;LOAD ADDRESS OF SOURCE EVEN BYTE DATA
017562  010060  177772                  MOV     R0,-6(R0)
017566  005260  177772                  INC     -6(R0)              ;LOAD ADDRESS OF SOURCE ODD BYTE DATA

017572  005002                          CLR     R2                  ;SET INDEX REGISTERS
017574  012703  000002                  MOV     #2,R3               ;@SBINB7(2);@SBINB7(3) REFERENCE EVEN &
017600  012704  177774                  MOV     #-4,R4              ;ODD BYTE SOURCE DATA; @DBINB7(4);@DBINB7(5)
017604  012705  177776                  MOV     #-2,R5              ;REFERENCE DEST EVEN& ODD BYTE DATA


017610  005020                          CLR     (R0)+               ;PRESET SOURCE DATA
017612  005010                          CLR     (R0)                ;PRESET DEST DATA
017614  013746  001550                  MOV     @#FACTOR,-(SP)      ;GET RELOCATION FACTOR
017620  061602                          ADD     (SP),R2             ;AND ADD TO INDEX VALUES
017622  061603                          ADD     (SP),R3
017624  061604                          ADD     (SP),R4
017626  062605                          ADD     (SP)+,R5

017630  112773  177777  017520          MOVB    #-1,@SBINB7(3)      ;SRC DATA = 177400
017636  132772  000377  017520          BITB    #377,@SBINB7(2)     ;CHECK THAT EVEN BYTE WAS NOT AFFECTED
017644  001401                          BEQ     .+4                 ;BY MOVB INSTRUCTION
017646  104000                          HLT

017650  157374  017520  017530          BISB    @SBINB7(3),@DBINB7(4)
017656  105274  017530                  INCB    @DBINB7(4)          ;CHECK THAT BIS SET ALL BITS
017662  001401                          BEQ     .+4
017664  104000                          HLT

017666  105375  017530                  DECB    @DBINB7(5)          ;DEST DATA = 177400
017672  005274  017530                  INC     @DBINB7(4)          ;DEST DATA = 177401
017676  127375  017520  017530          CMPB    @SBINB7(3),@DBINB7(5)
017704  001401                          BEQ     .+4
017706  104000                          HLT

017710  147375  017520  017530          BICB    @SBINB7(3),@DBINB7(5)
017716  001401                          BEQ     .+4
017720  104000                          HLT

017722  105073  017520                  CLRB    @SBINB7(3)          ;SRC DATA = 000000
                                ;THIS ROUTINE SETS ALL BITS IN THE SOURCE ODD BYTE BY BISING A BIT FROM
                                ;THE DEST EVEN BYTE INTO THE SOURCE ODD BYTE
017726  157473  017530  017520  BIS7:   BISB    @DBINB7(4),@SBINB7(3)
017734  106174  017530                  ROLB    @DBINB7(4)
017740  103372                          BCC     BIS7
017742  022772  177400  017520          CMP     #177400,@SBINB7(2)      ;CHECK RESULT
017750  001401                          BEQ     .+4
017752  104000                          HLT

017754  000372  017520                  SWAB    @SBINB7(2)          ;SRC DATA = 000377
017760  112775  000200  017530          MOVB    #200,@DBINB7(5)     ;DEST DATA = 100000
```

```
017766  147572  017530  017520  BIC7:   BICB    @DBINB7(5),@SBINB7(2)
017774  106075  017530                  RORB    @DBINB7(5)
020000  103372                          BCC     BIC7
020002  005772  017520                  TST     @SBINB7(2)
020006  001401                          BEQ     .+4
020010  104000                          HLT

020012  012702  000001          OAERR:  MOV     #1,R2              ;LOAD R2 WITH ODD #
020016  010703                          MOV     PC,R3
020020  000401                          BR      .+4                ;RESERVE SPACE FOR A WORD
020022  000000                          .WORD   0                  ;WILL CONTAIN AN ODD ADDRESS
020024  005723                          TST     (R3)+              ;STEP R3 TO POINT TO WORD ABOVE
020026  010313                          MOV     R3,(R3)
020030  005213                          INC     (R3)               ;AND MAKE ODD
020032  012737  020160  000004          MOV     #1$,@#ERRVEC       ;SET ODD ADDRESS & RESERVED INSTRUCTION
020040  063737  001550  000004          ADD     @#FACTOR,@#ERRVEC
020046  013737  000004  000010          MOV     @#ERRVEC,@#RESVEC          ;TO TRAP TO 1$ BELOW

```

```
  4989  020054  000277                    SCC                           ;SET ALL CC'S
```

```
020056  160212                          SUB     R2,(R2)
020060  104000                          HLT
020062  060222                          ADD     R2,(R2)+
020064  104000                          HLT
020066  006342                          ASL     -(R2)
020070  104000                          HLT
020072  106512                          MFPD    (R2)
020074  104000                          HLT
020076  170412                          CLRF    (R2)
020100  104000                          HLT
020102  042202                          BIC     (R2)+,R2
020104  104000                          HLT
020106  164202                          SUB     -(R2),R2
020110  104000                          HLT
020112  155202                          BISB    @-(R2),R2
020114  104000                          HLT
020116  105532                          ADCB    @(R2)+
020120  104000                          HLT
020122  163302                          SUB     @(R3)+,R2
020124  104000                          HLT
020126  005733                          TST     @(R3)+
020130  104000                          HLT
020132  106533                          MFPD    @(R3)+
020134  104000                          HLT
020136  170453                          CLRD    @-(R3)
020140  104000                          HLT
020142  137702  177775                  BITB    @.+1,R2
020146  104000                          HLT
020150  105477  177773                  NEGB    @.-1
020154  104000                          HLT
020156  000406                          BR      2$

020160  062716  000002          1$:     ADD     #2,(SP)         ;ADJUST RETURN PC
020164  052766  000017  000002          BIS     #17,2(SP)       ;SET CONDITION CODES ON RETURN
020172  000002                          RTI

020174  012706  000700          2$:     MOV     #USESTK,SP      ;RESET STACK PTR
020200  012737  055064  000004          MOV     #ERPRT,@#ERRVEC ;RESET TIME OUT VECTOR
020206  012737  054774  000010          MOV     #RESERR,@#RESVEC
                                ;;*****************************************************************
                                ;*TEST 34       CHECK JUMP INSTRUCTIONS
                                ;;*****************************************************************
020214                          TST34:
020214  000004                          SCOPE
020216  112737  000034  001202          MOVB    #34,@#$TSTNM
020224  010700                          MOV     PC,R0
020226  062700  000012                  ADD     #12,R0          ;SET ADDRESS FOR JMP INST
020232  000277                          SCC                     ;SET CC'S
020234  000110                          JMP     (R0)
020236  000402                          BR      .+6
020240  000250                          CLN                     ;JMP INST JUMPS HERE
020242  000775                          BR      .-4

020244  103003                          BCC     JMP1
020246  102002                          BVC     JMP1
020250  001001                          BNE     JMP1
```

```
020252  100001                         BPL     .+4
020254  104000                 JMP1:   HLT                        ;ERROR! INCORRECT CC'S AFTER JMP

020256  005002                         CLR     R2                 ;SET INDICATOR
0[illegible]50  010703                         MOV     PC,R3
0[illegible]52  000401                         BR      .+4                ;RESERVE WORD FOR JMP ADDRESS
0[illegible]54  0[illegible]0000                         .WORD   0                  ;CONTAINS ADDRESS FOR JMP INST
0[illegible]56  005723                         TST     (R3)+
0[illegible]70  010313                         MOV     R3,(R3)
0[illegible]72  010300                         MOV     R3,R0
0[illegible]274  062713  000022                ADD     #22,(R3)           ;(R3) IS JMP ADDRESS
020300  010300                         MOV     R3,R0
020302  000133                         JMP     @(R3)+             ;JUMP TO ADDRESS CONTAINED IN R3
020304  000402                         BR      .+6
020306  005102                         COM     R2                 ;COMPLEMENT INDICATOR
020310  000775                         BR      .-4
020312  005202                         INC     R2                 ;CHECK INDICATOR
020314  001003                         BNE     JMP3
020316  005720                         TST     (R0)+
0[illegible]320  020003                         CMP     R0,R3              ;CHECK AUTO-INC R3
0[illegible]2  001401                         BEQ     .+4
020324  104000                 JMP3:   HLT

020326  005002                         CLR     R2                 ;SET INDICATOR
0[illegible]0330  010704                         MOV     PC,R4              ;SET UP JMP REGISTER
0[illegible]0332  010400                         MOV     R4,R0              ;SET UP CHECK REGISTER
0[illegible]0334  000402                         BR      1$
0[illegible]0336  0[illegible]5102                         COM     R2                 ;COMPLEMENT INDICATOR
[illegible]20340  0[illegible]0403                         BR      2$
[illegible]20342  0[illegible]2424                 1$:     CMP     (R4)+,(R4)+
0[illegible]344  005724                         TST     (R4)+              ;R4=JMP ADDRESS
0[illegible]46  000144                         JMP     -(R4)              ;USE R4 AS ADDRESS
[illegible]0  005[illegible]02                 2$:     INC     R2                 ;CHECK INDICATOR
[illegible]2  001003                         BNE     JMP4
0[illegible]54  022020                         CMP     (R0)+,(R0)+
0[illegible]56  020004                         CMP     R0,R4              ;CHECK AUTO-DEC R4
0[illegible]0  001401                         BEQ     .+4
0[illegible]362  104000                 JMP4:   HLT

020364  010703                         MOV     PC,R3
0[illegible]6  000401                         BR      .+4                ;RESERVE WORD FOR JMP ADDRESS
0[illegible]0  00[illegible]000                 1$:     .WORD   0                  ;CONTAINS JUMP ADDRESS
0[illegible]372  005723                         TST     (R3)+
[illegible]374  010313                         MOV     R3,(R3)
0[illegible]376  062723  000016                ADD     #16,(R3)+
0[illegible]402  010300                         MOV     R3,R0              ;LOAD CHECK REGISTER
0[illegible]404  000402                         BR      3$
0[illegible]406  005102                 2$:     COM     R2
020410  000401                         BR      4$
0[illegible]412  000153                 3$:     JMP     @-(R3)             ;JUMP TO 2$ VIA 1$ ABOVE
0[illegible]414  005202                 4$:     INC     R2                 ;CHECK INDICATOR
020416  001003                         BNE     JMP5
020420  005740                         TST     -(R0)
[illegible]422  020003                         CMP     R0,R3              ;CHECK AUTO-DEC R3
020424  001401                         BEQ     .+4
020426  104000                 JMP5:   HLT
```

```
020430  000402                          BR      2$
020432  005102                  1$:     COM     R2                      ;COMPLEMENT INDICATOR
020434  000402                          BR      3$
020436  000167  177770          2$:     JMP     1$
020442  005202                  3$:     INC     R2
020444  001401                          BEQ     .+4
020446  104000                  JMP6:   HLT

020450  012767  020466  000020          MOV     #1$,7$                  ;SET UP JMP ADDRESS
020456  063767  001550  000012          ADD     @#FACTOR,7$             ;ADD RELOCATION FACTOR
020464  000402                          BR      2$                      ;GO TO JMP  @7$ INST
020466  005102                  1$:     COM     R2                      ;COMPLEMENT INDICATOR
020470  000403                          BR      3$                      ;GO TO CHECK ROUTINE
020472  000177  000000          2$:     JMP     @7$                     ;JMP TO 1$ ABOVE VIA 7$
0[illegible]476  000000                  7$:     .WORD   0                       ;CONTAINS JMP ADDRESS
0[illegible]500  005202                  3$:     INC     R2                      ;CHECK INDICATOR
0[illegible]502  001401                          BEQ     .+4
020504  104000                  JMP7:   HLT
                                ;;*****************************************************************
                                ;*TEST 35        CHECK JSR INSTRUCTIONS
                                ;;*****************************************************************
020506                          TST35:
020506  000004                          SCOPE
020510  112737  000035  001202          MOVB    #35,@#$TSTNM
020516  013705  001550          JSR1:   MOV     @#FACTOR,R5             ;GET RELOCATION FACTOR
020522  012702  020554                  MOV     #3$,R2                  ;FORM DEST ADRS
020526  060502                          ADD     R5,R2                   ;ADD RELOCATION FACTOR
020530  000277                          SCC                             ;PRESET CC'S
020532  000242                          CLV
020534  004512                          JSR     R5,(R2)                 ;GO TO 3$ VIA R2
020536  005702                  1$:     TST     R2                      ;CHECK INDICATOR
020540  001017                          BNE     4$                      ;R2 SHOULD=0
020542  023705  001550                  CMP     @#FACTOR,R5             ;CHECK THAT RTS  R5 RESTORED R5
0[illegible]46  001014                          BNE     4$
0[illegible]0   000414                          BR      JSR3                    ;GO TO NEXT TEST
0[illegible]2   000205                  2$:     RTS     R5                      ;RETURN FROM SUBROUTINE
0[illegible]54  103011                  3$:     BCC     4$                      ;CHECK THAT JSR DID NOT
0[illegible]6   102410                          BVS     4$
0[illegible]0   001007                          BNE     4$                      ;AFFECT CC'S
0[illegible]2   100006                          BPL     4$
0[illegible]54  005002                          CLR     R2                      ;CLEAR INDICATOR
020566  012704  020536                  MOV     #1$,R4                  ;GET UNRELOCATED RETURN ADDRESS
020572  061604                          ADD     (SP),R4                 ;ADD RELOCATION FACTOR (OLD R5)
020574  020405                          CMP     R4,R5                   ;CHECK THAT OLD R5 WAS PLACED ON THE
020576  001765                          BEQ     2$                      ;STACK & THAT NEW R5 CONTAINS RETURN PC
020600  104000                  4$:     HLT                             ;ERROR! ABOVE

                                ;CHECK JSR INSTRUCTION ADDRESS MODE 3
020602  013704  001550          JSR3:   MOV     @#FACTOR,R4             ;GET RELOCATION FACTOR
020606  005000                          CLR     R0                      ;SET INDICATOR
020610  012705  020630                  MOV     #1$,R5
020614  060405                          ADD     R4,R5                   ;SET UP JSR DEFERRED ADRS
020616  010502                          MOV     R5,R2
020620  012715  020646                  MOV     #5$,(R5)
020624  060415                          ADD     R4,(R5)                 ;(R5)=DEST ADRS
```

```
020626  000401                         BR      2$                  ;RESERVE WORD FOR ADDRESS
0[illegible]30  000000          1$:    .WORD   0                   ;CONTAINS DEST ADRS FOR JSR
[illegible]2  004435          2$:    JSR     R4,@(R5)+           ;JSR TO 5$ VIA 1$ ABOVE
[illegible]34  005200          3$:    INC     R0                  ;CHECK INDICATOR
[illegible]36  001013                  BNE     6$
[illegible]40  000413                  BR      JSR4
0[illegible]42  005100          4$:    COM     R0                  ;COMPLEMENT INDICATOR
[illegible]44  000204                  RTS     4                   ;RETURN FROM SUBROUTINE
[illegible]646  012703  020634  5$:    MOV     #3$,R3              ;GET UNRELOCATED RETURN ADDRESS
0[illegible]652  061603                  ADD     (SP),R3             ;ADD RELOCATION FACTOR (OLD R4)
[illegible]54  020403                  CMP     R4,R3
[illegible]6  001003                  BNE     6$
[illegible]0  005722                  TST     (R2)+
0[illegible]662  020205                  CMP     R2,R5               ;CHECK AUTO-INC R5
0[illegible]64  001766                  BEQ     4$                  ;GO TO RTS
0[illegible]66  104000          6$:    HLT                         ;ERROR ABOVE

                                ;CHECK JSR INST ADDRESS MODE 4
020670  013704  001550  JSR4:  MOV     @#FACTOR,R4
020674  010405                  MOV     R4,R5
020676  010703                  MOV     PC,R3
020700  000401                  BR      2$
020702  000405          1$:    BR      4$
020704  022323          2$:    CMP     (R3)+,(R3)+
020706  000277                  SCC
020710  004443                  JSR     R4,-(R3)            ;GO TO 2$
020712  104000          3$:    HLT
020714  000414                  BR      JSR6                ;GO TO NEXT TEST
020716  103012          4$:    BCC     5$
020720  102011                  BVC     5$
020722  001010                  BNE     5$
020724  100007                  BPL     5$
020726  012702  020712          MOV     #3$,R2              ;GET UNRELOCATED RETURN ADDRESS
020732  061602                  ADD     (SP),R2             ;ADD RELOCATION FACTOR (OLD R4)
020734  020204                  CMP     R2,R4               ;CHECK THAT CALCULATED RETURN
020736  001002                  BNE     5$                  ;PC = NEW R4
020740  005724                  TST     (R4)+
020742  000204                  RTS     R4
020744  104000          5$:    HLT

                                ;TEST JSR INST ADDRESS MODE 6
020746  000401          JSR6:  BR      2$
020750  000405          1$:    BR      3$
020752  010700          2$:    MOV     PC,R0
020754  004767  177770          JSR     PC,1$
020760  100407                  BMI     JSR7                ;GO TO NEXT TEST
020762  104000                  HLT                         ;ERROR ON CC'S
020764  022020          3$:    CMP     (R0)+,(R0)+
020766  020016                  CMP     R0,(SP)             ;CHECK THAT RETURN ADDRESS IS ON THE
0[illegible]770  001401                  BEQ     .+4                 ;STACK
0[illegible]772  104000                  HLT
0[illegible]774  000270                  SEN                         ;SET N
020776  000207                  RTS     PC

                                ;TEST JSR INST ADDRESS MODE 7
021000  013746  001550  JSR7:  MOV     @#FACTOR,-(SP)      ;GET RELOCATION FACTOR
```

```
021004  062716  021024                  ADD     #1$,(SP)        ;FORM ADDRESS OF 1$ BELOW
021010  000277                          SCC                     ;SET ALL CC'S
021012  004076  000000                  JSR     R0,@(SP)        ;JSR TO 1$
021016  003003                          BGT     3$
021020  102002                          BVC     3$
021022  000402                          BR      4$

021024  000200                  1$:     RTS     R0              ;RETURN
021026  104000                  3$:     HLT                     ;ERROR!! INCORRECT CC'S
021030                          4$:
                                ;;*****************************************************************
                                ;*TEST 36       CHECK IOT TRAP (AND ROLB/ASLB)
                                ;*      THIS TEST CHECKS THAT THE PSW IS CORRECT AFTER THE IOT AND THAT THE
                                ;*      'NEW'PSW (FROM IOTVEC+2) IS CORRECT.
                                ;;*****************************************************************
021030                          TST36:
021030  000004                          SCOPE
021032  112737  000036  001202          MOVB    #36,@#$TSTNM
021040  012705  000022          IOTTST: MOV     #IOTVEC+2,R5
021044  005000                          CLR     R0
021046  052740  000200                  BIS     #PR4,-(R0)              ;SET PRIORITY LEVEL 4 IN PSW
021052  011015                          MOV     (R0),(R5)               ;SET IOTVEC+2 = PSW
021054  011504                          MOV     (R5),R4                 ;SAVE IN R4
021056  010746                          MOV     PC,-(SP)
021060  062716  000036                  ADD     #1$-.,(SP)
021064  012645                          MOV     (SP)+,-(R5)             ;LOAD IOT TRAP VECTOR
021066  042710  000357                  BIC     #PR7+17,(R0)
021072  052710  000244                  BIS     #PR5+4,(R0)             ;PSW=X XXX X00 101 1X1 000
021076  012003                          MOV     (R0)+,R3                ;R3 = PSW ABOVE
021100  010340                          MOV     R3,-(R0)                ;RESTORE PSW (MOV CHANGED IT)
021102  000004                          IOT
021104  012737  046720  000020  10$:    MOV     #$SCOPE,@#IOTVEC        ;RESTORE IOT VECTOR
021112  104000                          HLT                             ;ERROR! IOT FAILED TO TRAP
021114  000447                          BR      TST37           ;;GO TO NEXT TEST

021116  012002                  1$:     MOV     (R0)+,R2                ;GET PSW AFTER IOT TRAP
                                                                        ;NOTE: R0=0
021120  012725  046720                  MOV     #$SCOPE,(R5)+           ;RESTORE IOTVEC
021124  012715  000200                  MOV     #PR4,(R5)               ;AND IOTVEC+2
021130  010746                          MOV     PC,-(SP)                ;FORM PC OF 10$ ABOVE
021132  062716  177752                  ADD     #10$-.,(SP)
021136  022626                          CMP     (SP)+,(SP)+             ;CHECK RETURN PC ON STACK
021140  001026                          BNE     99$
021142  022603                          CMP     (SP)+,R3                ;CHECK SAVED PSW
021144  001024                          BNE     99$
021146  032703  140000                  BIT     #UM,R3                  ;BRANCH TO 3$ IF IN USER MODE
021152  100403                          BMI     3$
021154  020204                          CMP     R2,R4                   ;CHECK PSW AFTER IOT
021156  001017                          BNE     99$
021160  000404                          BR      4$


021162  052704  030000          3$:     BIS     #PUM,R4                 ;SET PREV USER MODE
021166  020204                          CMP     R2,R4                   ;CHECK PSW AFTER IOT
021170  001012                          BNE     99$
```

```
021172  005002                          4$:     CLR     R2
021174  000261                                  SEC
021176  106100                                  ROLB    R0                          ;ROTATE R0
021200  102376                                  BVC     .-2                         ;UNTIL V SETS (R0=200)

021202  106300                                  ASLB    R0                          ;SHIFT SHOULD SET CARRY
021204  103004                                  BCC     99$
021206  102003                                  BVC     99$
021210  001002                                  BNE     99$
021212  005700                                  TST     R0
021214  001401                                  BEQ     .+4
021216  104000                          99$:    HLT                                 ;ERROR! ROL/ASL FAILED TO SET
                                                                                    ;CC'S PROPERLY (IF R2=0) OR IN-
                                                                                    ;CORRECT PSW AFTER IOT (IF R2 NOT 0)
021220  042704  000340                          BIC     #PR7,R4
021224  010437  177776                          MOV     R4,@#PSW                    ;RESTORE PSW
021230  012706  000700                          MOV     #USESTK,SP                  ;RESTORE STACK PTR
                                        ;;*****************************************************************
                                        ;*TEST 37       CHECK EMT TRAP SEQUENCE
                                        ;;*****************************************************************
021234                                  TST37:
021234  000004                                  SCOPE
021236  112737  000037  001202                  MOVB    #37,@#$TSTNM
                                                .EQUIV  IOT,HLT                     ;REDEFINE HLT CALL
021244  012737  047160  000020                  MOV     #$ERROR,@#IOTVEC            ;SETUP VECTOR
021252  012737  000340  000022                  MOV     #PR7,@#IOTVEC+2
021260  005000                                  CLR     R0
021262  010746                                  MOV     PC,-(SP)
021264  062716  000030                          ADD     #EMT1-.,(SP)
021270  012637  000030                          MOV     (SP)+,@#EMTVEC
021274  000262                                  SEV                                 ;SET V
021276  013737  177776  000032                  MOV     @#PSW,@#EMTVEC+2            ;RETAIN CURRENT PSW ON TRAP
021304  000265                                  +SEZ!SEC
021306  104000                                  EMT                         ;TRAP TO EMT1
021310  001433                                  BEQ     EMT1C               ;GO TO EMT1C
021312  000004                                  HLT                         ;ERROR! INCORRECT CC'S WERE SET ON RETURN
021314  102027                          EMT1:   BVC     EMT1B               ;'V' SHOULD'VE SET ON EMT TRAP
021316  105100                                  COMB    R0                  ;R0=000377,CC'S=1001
021320  105500                                  ADCB    R0                  ;R0=000000,CC'S=0101
021322  106000                                  RORB    R0                  ;R0=000200,CC'S=1010
021324  102023                                  BVC     EMT1B
021326  100022                                  BPL     EMT1B
021330  000257                                  CCC
021332  105400                                  NEGB    R0                  ;R0=000200,CC'S=1010
021334  102017                                  BVC     EMT1B
021336  100016                                  BPL     EMT1B
021340  000242                                  CLV                         ;CLEAR 'V'
021342  000261                                  SEC                         ;AND SET 'C'
021344  105300                                  DECB    R0                  ;R0=000177,CC'S=0011
021346  102012                                  BVC     EMT1B
021350  100411                                  BMI     EMT1B
021352  000242                                  CLV                         ;CLEAR 'V'
021354  105200                                  INCB    R0                  ;R0=000200,CC'S=1011
021356  103006                                  BCC     EMT1B
021360  102005                                  BVC     EMT1B
021362  100004                                  BPL     EMT1B
```

```
021364  000242                                  CLV                             ;CLEAR 'V'
021366  106200                                  ASRB    R0                      ;SHIFT R0 UNTIL 'V' CLEARS
021370  102776                                  BVS     .-2
021372  000401                                  BR      .+4
021374  000004                          EMT1B:  HLT                             ;ERROR!
021376  000002                                  RTI                             ;EXIT WITH R0=000377
021400  105500                          EMT1C:  ADCB    R0                      ;R0=000000
021402  103003                                  BCC     EMT1D
021404  001002                                  BNE     EMT1D
021406  005700                                  TST     R0
021410  001401                                  BEQ     .+4
021412  000004                          EMT1D:  HLT
021414  012737  047160  000030                  MOV     #$ERROR,@#EMTVEC                ;RESTORE EMT TO ERROR
021422  012737  000340  000032                  MOV     #PR7,@#EMTVEC+2                 ;SET PRIORITY 7 ON ERROR
021430  012737  046720  000020                  MOV     #$SCOPE,@#IOTVEC                ;RESTORE IOT VECTOR
021436  005037  000022                          CLR     @#IOTVEC+2
                                                .EQUIV  ERROR,HLT                       ;REDEFINE HLT CALL
                                        ;;*********************************************************************
                                        ;*TEST 40       CHECK TRAP INSTRUCTION TRAP SEQUENCE
                                        ;;*********************************************************************
021442                                  TST40:
021442  000004                                  SCOPE
021444  112737  000040  001202                  MOVB    #40,@#$TSTNM
021452  052737  000340  177776                  BIS     #PR7,@#PSW                      ;LOCK OUT LINE CLOCK
021460  052737  000340  000016                  BIS     #PR7,@#TBITVEC+2
021466  010746                                  MOV     PC,-(SP)
021470  062716  000056                          ADD     #TRAP1-.,(SP)
021474  012637  000034                          MOV     (SP)+,@#TRAPVEC
021500  000270                                  SEN                                     ;SET N
021502  013737  177776  000036                  MOV     @#PSW,@#TRAPVEC+2               ;RETAIN CURRENT PSW ON TRAP
021510  000261                                  SEC                                     ;SET CARRY
021512  010700                                  MOV     PC,R0
021514  000264                                  SEZ                             ;SET Z BIT
021516  104400                                  TRAP                            ;TRAP TO TRAP1
021520  103404                                  BCS     .+12
021522  012737  053566  000034                  MOV     #$TRAP,@#TRAPVEC                ;RESTORE TRAP VECTOR
021530  104000                                  HLT
021532  001404                                  BEQ     .+12
021534  012737  053566  000034                  MOV     #$TRAP,@#TRAPVEC                ;RESTORE TRAP VECTOR
021542  104000                                  HLT
021544  000420                                  BR      TRAP1C
021546  100404                          TRAP1:  BMI     .+12                    ;N BIT GOT SET ON TRAP
021550  012737  053566  000034                  MOV     #$TRAP,@#TRAPVEC                ;RESTORE TRAP VECTOR
021556  104000                                  HLT
021560  062700  000004                          ADD     #4,R0
021564  020016                                  CMP     R0,(SP)                 ;CHECK LOW BYTE OF RETURN PC ON
021566  001404                                  BEQ     .+12                    ;STACK
021570  012737  053566  000034                  MOV     #$TRAP,@#TRAPVEC                ;RESTORE TRAP VECTOR
021576  104000                                  HLT
021600  124646                                  CMPB    -(SP),-(SP)
021602  032626                                  BIT     (SP)+,(SP)+
021604  000002                                  RTI                             ;RETURN TO INST FOLLOWING TRAP (1$)

021606  012702  000036                  TRAP1C: MOV     #TRAPVEC+2,R2                   ;RESTORE VECTORS
021612  012712  000340                          MOV     #PR7,(R2)
021616  012742  053566                          MOV     #$TRAP,-(R2)
```

```
021622  042737  000340  000016          BIC     #PR7,@#TBITVEC+2
021630  105037  177776                  CLRB    @#PSW                           ;GO BACK TO PRIORITY 0

021634  000004                  RELE3:  SCOPE
021636  010702                          MOV     PC,R2
021640  062702  000012                  ADD     #12,R2
021644  012707  036574                  MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
021650  000000                  REL33:  .WORD   0
                                ;3333333333333 LAST ADDRESS OF CODE TO BE RELOCATED 33333333333


                                ;;*******************************************************************
                                ;*TEST 41       CHECK STACK OVERFLOW
                                ;;*******************************************************************
021652                          TST41:
021652  012767  000001  157444          MOV     #1,$TIMES       ;;DO 1 ITERATION
021660  000004                          SCOPE
021662  112737  000041  001202          MOVB    #41,@#$TSTNM

                                        .SBTTL  START OF SECTION 4
                                ;444444444444 FIRST ADDRESS TO BE RELOCATED 444444444
021670  112737  000041  001202  REL4:   MOVB    #$TN-1,@#$TSTNM
021676  010700                          MOV     PC,R0           ;GET PC
021700  005740                          TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL4
021702  010037  001554                  MOV     R0,@#FRSTAD     ;SAVE
021706  010700                          MOV     PC,R0           ;GET CURRENT PC
021710  162700  021710                  SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
021714  010037  001550                  MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
021720  010737  001212                  MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
021724  062737  000026  001212          ADD     #26,@#$LPERR    ;ADJUST
021732  013737  001212  001210          MOV     @#$LPERR,@#$LPADR
021740  105737  001544                  TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
021744  001402                          BEQ     .+6
021746  000167  001326                  JMP     RELE4

021752  013767  177776  000330  OVFLW:  MOV     @#PSW,7$        ;SAVE STATUS IN 7$ BELOW
021760  005037  177776                  CLR     @#PSW           ;SET KERNEL MODE
021764  004737  054112                  JSR     PC,@#CLRTBIT    ;GO CLEAR 'T' BIT IF SET
021770  052737  000340  177776          BIS     #PR7,@#PSW      ;SET PRIORITY LEVEL 7 TO BLOCK CLOCK
021776  010746                          MOV     PC,-(SP)        ;PUSH CURRENT PC ONTO STACK
022000  062716  000146                  ADD     #2$-.,(SP)      ;FORM ADDRESS OF 2$ BELOW
022004  011637  000004                  MOV     (SP),@#ERRVEC   ;SET ERROR VECTOR
022010  012737  000340  000006          MOV     #340,@#ERRVEC+2 ;SET PRIORITY LEVEL 7 ON TRAP
022016  013727  000014                  MOV     @#BPTVEC,(PC)+  ;SAVE BPT VECTOR ADRS
022022  000000                  43$:    .WORD   0
022024  062716  000100                  ADD     #41$-2$,(SP)    ;FORM ADDRESS OF 41$ BELOW
022030  012637  000014                  MOV     (SP)+,@#BPTVEC  ;SET BPT TRAP VECTOR TO 41$
022034  012737  000340  000016          MOV     #340,@#RPTVEC+2

022042  012703  000376                  MOV     #376,R3
022046  010313                          MOV     R3,(R3)         ;LOAD 376 INTO ADDRESS 376
022050  010306                          MOV     R3,SP           ;SET STACK PTR AT BOUNDARY
022052  032767  140000  000230          BIT     #UM,7$          ;CHECK IF ENTERED TEST IN KERNEL
022060  001015                          BNE     1$              ;MODE. BRANCH IF NOT IN KERNEL

                                ;THE BELOW INSTRUCTIONS SHOULD NOT CAUSE AN OVERFLOW TRAP
022062  005716                          TST     (SP)            ;BECAUSE TST IS A NON MODIFYING INST
```

```
022064  021666  177776                  CMP     (SP),-2(SP)     ;SO IS COMPARE
022070  012656                          MOV     (SP)+,@-(SP)    ;BECAUSE OF ADDRESS MODE 5
022072  057636  000000                  BIS     @(SP),@(SP)+    ;BECAUSE OF ADDRESS MODE 3
022076  054676  000000                  BIS     -(SP),@(SP)     ;BECAUSE OF ADDRESS MODE 7
022102  005006                          CLR     SP
022104  013766  020000  020000          MOV     @#20000,20000(SP)
022112  000423                          BR      3$              ;BRANCH OVER NON KERNEL MODE TESTS

                                ;NOTE:  NO OVERFLOW TRAP WILL OCCUR IF NOT IN KERNEL MODE!!!
022114  156737  000171  177777  1$:     BISB    7$+1,@#PSW+1    ;RESTORE MODE BITS IN PSW
022122  012706  000376                  MOV     #376,SP         ;SET STACK PTR
022126  016646  177776                  MOV     -2(SP),-(SP)    ;SHOULD NOT TRAP
022132  051616                          BIS     (SP),(SP)
022134  061666  177776                  ADD     (SP),-2(SP)
022140  105037  177777                  CLRB    @#PSW+1         ;SET KERNEL MODE
022144  000451                          BR      6$              ;EXIT TEST

                                ;ERROR SERVICE ROUTINE
022146  012600                  2$:     MOV     (SP)+,R0        ;SAVE PC OF INSTRUCTION THAT TRAPPED
022150  012602                          MOV     (SP)+,R2        ;SAVE PSW
022152  012706  000700                  MOV     #USESTK,SP      ;SET STACK PTR
022156  104000                          HLT                     ;ERROR! AN INSTRUCTION THAT WAS NOT
                                                                ;SUPPOESED TO TRAP TRAPPED
                                                                ;RO CONTAINS PC, R2 CONTAINS PSW
022160  000443                          BR      6$              ;EXIT TEST
                                ;THE BELOW INSTRUCTIONS WILL CAUSE A STACK OVERFLOW
                                ;STACK PTR IS AT 376
022162  062737  000066  000004  3$:     ADD     #4$-2$,@#ERRVEC ;SET ERROR VECTOR TO 4$
022170  010306                          MOV     R3,SP           ;SET STACK PTR AT 376
022172  112702  000001                  MOVB    #1,R2
022176  005000                          CLR     R0
022200  005016                          CLR     (SP)            ;SETS BIT 0 IN R0
022202  006302                          ASL     R2              ;SHIFT INDICATOR BIT
022204  105226                          INCB    (SP)+           ;SETS BIT 1 IN R0
022206  006302                          ASL     R2
022210  060746                          ADD     PC,-(SP)        ;SETS BIT 2 IN R0
022212  006302                          ASL     R2
022214  000003                          BPT                     ;SETS BIT 3 IN R0
022216  006302                          ASL     R2
022220  004767  000014                  JSR     PC,40$          ;SETS BIT 4 IN R0
022224  006302                          ASL     R2
022226  050666  177776                  BIS     SP,-2(SP)       ;SETS BIT 5 IN R0
022232  000410                          BR      5$

                                ;PROGRAM WILL TRAP HERE ON OVERFLOW TRAP
022234  050200                  4$:     BIS     R2,R0           ;SET APPROPRIATE BIT IN R0
022236  000002                          RTI                     ;RETURN FROM TRAP

022240  052700  001000          40$:    BIS     #1000,R0                ;SET IND THAT JSR WAS EXECUTED
022244  000207                          RTS     PC

022246  052700  000400          41$:    BIS     #400,R0                 ;SET IND THAT BPT WAS EXECUTED
022252  000002                          RTI

                                ;CHECK THAT ABOVE INSRUCTIONS DID TRAP
022254  012706  000700          5$:     MOV     #USESTK,SP      ;SET STACK PTR
```

```
022260  022700  001477                  CMP     #1477,R0        ;EACH INSTRUCTION SET A BIT IN R0
022264  001401                          BEQ     .+4             ;R0= 1477
022266  104000                          HLT


                                ;EXIT ROUTINE
022270  012706  001200          6$:     MOV     #KERSTK,SP      ;SET KERNEL STACK PTR
022274  016737  177522  000014          MOV     43$,@#BPTVEC    ;RESTORE BPT VECTOR
022302  005037  000016                  CLR     @#BPTVEC+2
022306  012746                          MOV     (PC)+,-(SP)     ;PUSH OLD PSW ONTO STACK
022310  000000                  7$:     .WORD   0               ;CONTAINS SAVED PSW
022312  010746                          MOV     PC,-(SP)        ;PUSH CURRENT PC ONTO STACK
022314  062716  000006                  ADD     #6,(SP)         ;ADD OFFSET
022320  000002                          RTI
022322  012706  000700                  MOV     #USESTK,SP      ;SET STACK PTR
022326  012737  055064  000004          MOV     #ERPRT,@#ERRVEC ;RESET TIME OUT VECTOR
022334  013737  177776  000006          MOV     @#PSW,@#ERRVEC+2
022342  052737  000340  000006          BIS     #PR7,@#ERRVEC+2
                                ;;*****************************************************************
                                ;*TEST 42       CHECK THAT ALL RESERVED INSTRUCTIONS TRAP
                                ;;*****************************************************************
022350                          TST42:
022350  000004                          SCOPE
022352  112737  000042  001202          MOVB    #42,@#$TSTNM
022360  012702  022450          RESTRP: MOV     #5$,R2          ;GET ADDRESS OF RESERVED INSTRUCTION TABLE
022364  063702  001550                  ADD     @#FACTOR,R2
022370  012737  022426  000010          MOV     #4$,@#RESVEC    ;SET RESERVED INSTRUCTION TRAP
022376  063737  001550  000010          ADD     @#FACTOR,@#RESVEC
022404  012203                  1$:     MOV     (R2)+,R3        ;GET FIRST RESERVED INSTRUCTION
022406  001437                          BEQ     7$              ;0 TERMINATES THE TABLE
022410  012204                          MOV     (R2)+,R4        ;GET LAST RESERVED INSTRUCTION IN GROUP
022412  010317                  2$:     MOV     R3,(PC)         ;EXECUTE RESERVED INSTRUCTION
022414  000000                  3$:     .WORD   0               ;CONTAINS RESERVED INSTRUCTION
022416  104000                          HLT                     ;ERROR! INSTRUCTION IN R3
022420  104000                          HLT                     ;(2$) ABOVE FAILED TO CAUSE A
022422  104000                          HLT                     ;RESERVED INSTRUCTION TRAP
022424  000405                          BR      41$
022426  012716  022440          4$:     MOV     #41$,(SP)       ;ADJUST RETURN PC
022432  063716  001550                  ADD     @#FACTOR,(SP)   ;TO RETURN TO 41$
022436  000002                          RTI                     ;RETURN TO 41$
022440  020304                  41$:    CMP     R3,R4           ;HAS GROUP OF RESERVED INSTRUCTIONS
022442  001760                          BEQ     1$              ;BEEN EXECUTED
022444  005203                          INC     R3              ;INCREMENT THIS RESERVED INSTRUCTION
022446  000761                          BR      2$              ;TO NEXT ONE AND EXECUTE
                                ;TABLE OF 11/6X RESERVED INSTRUCTIONS (0 TERMINATES THE TABLE)
022450  000007                  5$:     7                       ;GROUP 1
022452  000077                          77                      ;    "
022454  000210                          210                     ;GROUP 2
022456  000237                          237                     ;    "
022460  007000                          7000                    ;GROUP 3
022462  007777                          7777                    ;    "
022464  075040                          75040                   ;GROUP 4
022466  076577                          76577                   ;    "
                                                                ;MED INST. = 076600
022470  076601                          76601                   ;GROUP 5
022472  076677                          76677                   ;    "
                                                                ;XFC INSTS. = 0767XX
```

```
                                                                  ;(76700-76777)
 022474  106400                               106400              ;GROUP 6
 022476  106477                               106477              ;   "
 022500  106700                               106700              ;GROUP 7
 022502  107777                               107777              ;   "
 022504  000000                               0                   ;0 TERMINATES THE TABLE

 022506  012737  054774  000010   7$:   MOV     #RESERR,@#RESVEC          ;RESTORE RESERVED TRAP
                                  ;;*****************************************************************
                                  ;;*TEST 43       CHECK THAT ALL BITS IN THE PSW CAN BE SET AND CLEARED
                                  ;;*****************************************************************
 022514                           TST43:
 022514  000004                         SCOPE
 022516  112737  000043  001202         MOVB    #43,@#$TSTNM
 022524  105737  001545           PSWCHK: TSTB  @#MMON              ;IF MEM MGMT IS ON SKIP THIS TEST
 022530  001103                         BNE     4$
 022532  013767  177776  000166         MOV     @#PSW,3$            ;SAVE STATUS
 022540  005037  177776                 CLR     @#PSW               ;CLEAR MODE BITS IN PSW
 022544  004737  054112                 JSR     PC,@#CLRTBIT        ;GO CLEAR 'T' BIT IF SET
 022550  013746  000016                 MOV     @#TBITVEC+2,-(SP)
 022554  012704  177776                 MOV     #PSW,R4             ;LOAD ADDRESS OF PSW INTO R4
 022560  000250                         CLN
 022562  005714                         TST     (R4)                ;CHECK THAT PSW WAS CLEARED
 022564  001401                         BEQ     .+4
 022566  104000                         HLT                         ;ERROR! PSW FAILED TO CLEAR
 022570  012700  170357                 MOV     #170357,R0
 022574  052700  170000                 BIS     #170000,R0          ;SET BITS 15-12 IF MEM MGMT
 022600  012702  000001           10$:  MOV     #1,R2               ;R2 = TEST BIT
 022604  030200                   1$:   BIT     R2,R0               ;CHECK IF BIT CAN BE SET/CLEARED
 022606  001423                         BEQ     2$
 022610  005037  000016                 CLR     @#TBITVEC+2
 022614  030227  000020                 BIT     R2,#20              ;CHECK IF TEST WILL SET 'T' BIT
 022620  001403                         BEQ     20$
 022622  012737  000002  000016         MOV     #RTI,@#TBITVEC+2;SET RTI INTO RETURN
 022630  005014                   20$:  CLR     (R4)                ;CLEAR PSW
 022632  050214                         BIS     R2,(R4)             ;SET R2 INTO PSW
 022634  011403                         MOV     (R4),R3             ;GET BIT
 022636  020203                         CMP     R2,R3               ;CHECK THAT BIT WAS SET IN PSW
 022640  001401                         BEQ     .+4
 022642  104000                         HLT                         ;ERROR! BIT IN R2 FAILED TO SET IN PSW
 022644  000244                         CLZ                         ;CLEAR Z BIT
 022646  040214                         BIC     R2,(R4)             ;CLEAR BIT IN PSW
 022650  011403                         MOV     (R4),R3             ;GET PSW RESULT
 022652  001401                         BEQ     2$                  ;BRANCH IF BIC ABOVE CLEARED BIT IN PSW
 022654  104000                         HLT                         ;ERROR! BIT IN R2 FAILED TO CLEAR IN PSW
 022656  020227  004000           2$:   CMP     R2,#4000            ;READY TO TEST BIT 12 YET?
 022662  001003                         BNE     12$                 ;BRANCH IF NO
 022664  012702  030000                 MOV     #30000,R2           ;IF YES, SET BITS 12 & 13 TOGETHER
 022670  000745                         BR      1$                  ;GO TEST BITS 12 & 13 OF THE PSW
 022672  020227  030000           12$:  CMP     R2,#30000           ;READY TO TEST BIT 14?
 022676  001003                         BNE     13$                 ;BRANCH IF NO
 022700  012702  140000                 MOV     #140000,R2          ;IF YES, SET BITS 14 & 15 TOGETHER
 022704  000737                         BR      1$                  ;GO TEST BITS 14 & 15 OF THE PSW
 022706  006302                   13$:  ASL     R2                  ;SHIFT TEST BIT
 022710  022702  100000                 CMP     #100000,R2          ;BRANCH IF ALL BITS NOT TESTED
 022714  001333                         BNE     1$
```

```
022716  005014                          CLR     (R4)                    ;CLEAR STATUS
022720  012637  000016                  MOV     (SP)+,@#TBITVEC+2               ;RESTORE T BIT RETURN
022724  012746                          MOV     (PC)+,-(SP)             ;PUSH ORIGINAL STATUS ON STACK
022726  000000                  3$:     .WORD   0                       ;CONTAINS ORIGINAL PSW
022730  010746                          MOV     PC,-(SP)                ;SET RETURN PC
022732  062716  000006                  ADD     #6,(SP)
022736  000002                          RTI                             ;RETURN
022740  013704  177776          4$:     MOV     @#PSW,R4                ;SAVE PSW IN R4
022744  112737  000340  177776          MOVB    #340,@#PSW              ;SET PRIORITY LEVEL 6
022752  004737  054112                  JSR     PC,@#CLRTBIT            ;GO CLEAR 'T' BIT IF SET
                                ;;*****************************************************************
                                ;*TEST 44        CHECK THAT ALL BITS IN THE CURRENT STACK PTR CAN BE SET CLEARED
                                ;;*****************************************************************
022756                          TST44:
022756  000004                          SCOPE
022760  112737  000044  001202          MOVB    #44,@#$TSTNM
022766  010603                  CHKSP:  MOV     SP,R3                   ;SAVE STACK PTR
022770  000257                          CCC
022772  112706  000377                  MOVB    #377,SP                 ;SET STACK PTR = -1
022776  006006                  1$:     ROR     SP                      ;ROTATE 0 BIT THROUGH ALL BIT
023000  103776                          BCS     1$                      ;BIT POSITIONS
023002  005206                          INC     SP                      ;SHOULD INCREMENT SP TO 0
023004  001403                          BEQ     2$
023006  010602                          MOV     SP,R2                   ;SAVE ERROR STACK PTR
023010  010306                          MOV     R3,SP                   ;SET STACK PTR FOR TRAP
023012  104000                          HLT                             ;ERROR!

023014  010306                  2$:     MOV     R3,SP                   ;RESTORE ORIGINAL STACK PTR

                                ;CHECK BYTE OPERATIONS USING THE STACK
023016  010600                  SPCHK:  MOV     SP,R0                   ;SAVE STACK PTR
023020  010003                          MOV     R0,R3

023022  005043                          CLR     -(R3)
023024  112746  177777                  MOVB    #-1,-(SP)               ;(SP) = 377
023030  022713  000377                  CMP     #377,(R3)               ;CHECK THAT ONLY EVEN BYTE WAS AFFECTED
023034  001002                          BNE     1$
023036  020306                          CMP     R3,SP                   ;CHECK AUTO-DEC
023040  001401                          BEQ     .+4
023042  104000                  1$:     HLT

023044  105226                          INCB    (SP)+
023046  005723                          TST     (R3)+                   ;CHECK RESULT
023050  001002                          BNE     2$
023052  020006                          CMP     R0,SP                   ;CHECK AUTO-INC
023054  001401                          BEQ     .+4
023056  104000                  2$:     HLT

023060  005143                          COM     -(R3)                   ;(R3)=177777
023062  144613                          BICB    -(SP),(R3)
023064  022713  177400                  CMP     #177400,(R3)            ;CHECK RESULT
023070  001002                          BNE     3$
023072  020603                          CMP     SP,R3
023074  001401                          BEQ     .+4
023076  104000                  3$:     HLT

```

```
023100  132627  000377                  BITB    (SP)+,#377
023104  001002                          BNE     4$
023106  020600                          CMP     SP,R0
023110  001401                          BEQ     .+4
023112  104000                  4$:     HLT

023114  012746  000001                  MOV     #1,-(SP)
023120  062706  000002                  ADD     #2,SP
023124  012702  177401                  MOV     #177401,R2
023130  120246                          CMPB    R2,-(SP)
023132  001004                          BNE     5$
023134  122602                          CMPB    (SP)+,R2
023136  001002                          BNE     5$
023140  020006                          CMP     R0,SP
023142  001401                          BEQ     .+4
023144  104000                  5$:     HLT
023146  105037  177776                  CLRB    @#PSW
023152  010446                          MOV     R4,-(SP)        ;RESTORE ORIGINAL PSW TO STACK
023154  010746                          MOV     PC,-(SP)
023156  062716  000006                  ADD     #6,(SP)
023162  000002                          RTI
                                ;;*****************************************************************
                                ;*TEST 45        CHECK THAT 'C' BIT SETS/CLEARS PROPERLY
                                ;;*****************************************************************
023164                          TST45:
023164  000004                          SCOPE
023166  112737  000045  001202          MOVB    #45,@#$TSTNM
023174  012727  177776          CBIT:   MOV     #177776,(PC)+   ;LOAD CONSTANT
023200  000000                  1$:     .WORD   0
023202  010700                          MOV     PC,R0           ;GET CURRENT PC
023204  162700  000004                  SUB     #4,R0           ;POINT R0 TO 1$ ABOVE
023210  005520                  2$:     ADC     (R0)+           ;ADD 'C' BIT TO 1$ ABOVE
023212  006340                          ASL     -(R0)           ;SHIFT 1$
023214  102375                          BVC     2$              ;UNTIL 'V' BIT SETS
023216  022767  077776  177754          CMP     #077776,1$      ;CHECK RESULT
023224  001401                          BEQ     .+4
023226  104000                          HLT                     ;ERROR! INCORRECT RESULT IN 1$ ABOVE
                                                                ;R0=ADDRESS OF DATA

                                ;CHECK THAT CONDITION CODES ARE SET PROPERLY WHEN A NUMBER (CURRENT PC)
                                ;AND THAT NUMBER +1 ARE COMPARED, AND VICE VERSA.
023230  010700                  CMPN:   MOV     PC,R0           ;GET CURRENT PC
023232  010002                          MOV     R0,R2           ;SAVE IN R2
023234  005202                          INC     R2              ;MAKE R2 = R0+1
023236  000277                          SCC
023240  000251                          +CLC!CLN                ;CLEAR C & N BITS
023242  020002                          CMP     R0,R2           ;COMPARE # WITH #+1
023244  103003                          BCC     1$              ;CARRY BIT SHOULD SET
023246  102402                          BVS     1$              ;V BIT SHOULD CLEAR
023250  001401                          BEQ     1$              ;Z BIT SHOULD CLEAR
023252  100401                          BMI     .+4             ;N BIT SHOULD SET
023254  104000                  1$:     HLT                     ;ERROR! COMPARE # WITH #+1 FAILED TO
                                                                ;SET CONDITION CODES IN PSW CORRECTLY

023256  000277                          SCC                     ;SET CONDITION CODES IN PSW
023260  120200                          CMPB    R2,R0           ;COMPARE #+1 WITH #
```

```
023262  103403                         BCS     2$              ;C BIT SHOULD CLEAR
02  64  102402                         BVS     2$              ;V BIT SHOULD CLEAR
02   6  001401                         BEQ     2$              ;Z BIT SHOULD CLEAR
02  70  100001                         BPL     .+4             ;N BIT SHOLD CLEAR
023272  104000                 2$:     HLT                     ;ERROR! COMPARE #+1 WITH # FAILED TO SET
                                                               ;CONDITION CODES IN PSW CORRECTLY
023274  105037  177776                 CLRB    @#PSW           ;ENSURE PRIORITY 0
023300  000004                 RELE4:  SCOPE
023302  010702                         MOV     PC,R2
023304  062702  000012                 ADD     #12,R2
023310  012707  036574                 MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
023314  000000                 REL44:  .WORD   0
                               ;444444444444 LAST ADDRESS OF CODE TO BE RELOCATED 4444444444

                               ;;*************************************************************
                               ;*TEST 46        CHECK EXTENDED INSTRUCTION SET
                               ;;*************************************************************
023316                         TST46:
023316  012767  000001  156000         MOV     #1,$TIMES       ;;DO 1 ITERATION
023324  000004                         SCOPE
023326  112737  000046  001202         MOVB    #46,@#$TSTNM

                                       .SBTTL  START OF SECTION 5
                               ;5555555555555 FIRST ADDRESS TO BE RELOCATED 555555555
023334  112737  000046  001202 REL5:   MOVB    #$TN-1,@#$TSTNM
023342  010700                         MOV     PC,R0           ;GET PC
023344  005740                         TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL5
023346  010037  001554                 MOV     R0,@#FRSTAD     ;SAVE
023352  010700                         MOV     PC,R0           ;GET CURRENT PC
023354  162700  023354                 SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
02 360  010037  001550                 MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
023364  010737  001212                 MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
023370  062737  000026  001212         ADD     #26,@#$LPERR    ;ADJUST
0 3376  013737  001212  001210         MOV     @#$LPERR,@#$LPADR
023404  105737  001544                 TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
023410  001402                         BEQ     .+6
0 3412  000167  001366                 JMP     RELE5
0 3416  005000                 EXTINST:CLR     R0
0  3420 000277                         SCC                     ;PRESET CC'S
0  3422 006700                         SXT     R0              ;EXTEND SIGN (1) INTO R0
0  3424 103005                         BCC     SXT0            ;CHECK RESULT CC'S
0    26 102404                         BVS     SXT0
0  3430 001403                         BEQ     SXT0
0  3432 100002                         BPL     SXT0
023434  005200                         INC     R0              ;CHECK RESULT
0  3436 001401                         BEQ     .+4
023440  104000                 SXT0:   HLT

023442  010700                         MOV     PC,R0
0 3444  010002                         MOV     R0,R2
0 3446  012703  177777                 MOV     #-1,R3
0    2  005102                         COM     R2
0     4 000243                         +CLV!CLC                ;CLEAR C AND V BITS
023456  074003                         XOR     R0,R3           ;R3 SHOULD CONTAIN COMPLEMENT OF R0
023460  103404                         BCS     XOR0            ;CHECK THAT C WAS NOT AFFECTED
023462  102403                         BVS     XOR0            ;AND THAT V WAS CLEARED
```

```
023464  001402                           BEQ     XOR0
023466  020203                           CMP     R2,R3              ;CHECK RESULT
023470  001401                           BEQ     .+4
023472  104000                  XOR0:    HLT                        ;ERROR!  XOR FAILED

023474  010700                           MOV     PC,R0
[illegible]  022020                      CMP     (R0)+,(R0)+        ;SET ADDRESS REGISTER
[illegible]  000401                      BR      1$                 ;RESERVE WORD FOR TEST DATA
[illegible]  000000                      .WORD   0                  ;CONTAINS TEST DATA
[illegible]  005700             1$:      TST     R0                 ;EXTEND SIGN OF ADDRESS INTO
[illegible]  006710                      SXT     (R0)               ;ADDRESS (R0)=-1 IF MSB R0=1
[illegible]  [illegible]                 CLR     R2                 ;OTHERWISE, (R0)=0
[illegible]  [illegible]                 TST     R0                 ;CHECK SIGN OF ADDRESS
[illegible]  [illegible]                 BPL     .+4
[illegible]  005102                      COM     R2                 ;COMPLEMENT CHECK REG IF NEG
[illegible]  021002                      CMP     (R0),R2            ;CHECK RESULT OF SXT
[illegible]  001401                      BEQ     .+4
023524  104000                  SXT1:    HLT                        ;ERROR!  SXT FAILED TO EXTEND SIGN PROPERLY

023526  012710  100000                   MOV     #100000,(R0)       ;PRESET DATA
023532  011002                           MOV     (R0),R2
[illegible]  000277                      SCC                        ;PRESET CC'S
[illegible]  074210                      XOR     R2,(R0)            ;XOR 100000 WITH 100000 RESULT = 0
[illegible]  103007                      BCC     XOR1               ;CHECK CC'S AFTER XOR
[illegible]  102406                      BVS     XOR1
[illegible]  001005                      BNE     XOR1
[illegible]  100404                      BMI     XOR1
[illegible]  005710                      TST     (R0)               ;CHECK RESULT (0)
[illegible]  001002                      BNE     XOR1
[illegible]  005402                      NEG     R2                 ;CHECK THAT REG WAS NOT AFFECTED
[illegible]  102401                      BVS     .+4
[illegible]  104000             XOR1:    HLT

023562  010702                           MOV     PC,R2
023564  022222                           CMP     (R2)+,(R2)+
023566  000401                           BR      SXT4               ;PRESERVE WORD FOR DATA
023570  000000                           .WORD   0                  ;RESERVED FOR DATA
023572  012722  125252          SXT4:    MOV     #125252,(R2)+      ;PRESET DATA
023576  006742                           SXT     -(R2)              ;EXTEND SIGN
023600  074722                           XOR     PC,(R2)+
023602  010700                           MOV     PC,R0              ;GET PC
023604  005740                           TST     -(R0)              ;SUBTRACT 2 FROM PC
023606  005100                           COM     R0                 ;R0=RESULT OF XOR PC-1 ABOVE
023610  074042                           XOR     R0,-(R2)           ;CHECK RESULT OF SXT AND XOR ABOVE
023612  001401                           BEQ     .+4
023614  104000                  XOR24:   HLT                        ;ERROR!  SXT & XOR ABOVE INCORRECT

023616  012704  000001                   MOV     #1,R4              ;SET R4
023622  006767  000060                   SXT     XOR6A              ;PRESET DATA=0
023626  074467  000054          2$:      XOR     R4,XOR6A
[illegible]  100423                      BMI     XOR6
[illegible]  006304                      ASL     R4                 ;SHIFT R4
[illegible]  102373                      BVC     2$                 ;UNTIL V SETS (R4=100000)
023640  100020                           BPL     XOR6               ;BRANCH IF 'N' IS CLEAR
023642  074467  000040                   XOR     R4,XOR6A           ;XOR6A=177777
023646  100015                           BPL     XOR6
```

```
023650  074767  000032                   XOR     PC,XOR6A          ;XOR PC WITH XOR6A (177777)
023654  010767  000030                   MOV     PC,XOR6B          ;FORM PC AS USED IN XOR ABOVE
023660  162767  000004  000022           SUB     #4,XOR6B
023666  005167  000016                   COM     XOR6B
023672  026767  000012  000006           CMP     XOR6B,XOR6A       ;XOR6A SHOULD = COMPLEMENT OF PC
023700  001401                           BEQ     .+4
023702  104000                   XOR6:   HLT                       ;ERROR!  XOR TESTS ABOVE FAILED

023704  000402                           BR      .+6

023706  000000                   XOR6A:  .WORD   0                 ;CONTAINS DATA USED BY TEST ABOVE
023710  000000                   XOR6B:  .WORD   0


023712  012700  077777                   MOV     #077777,R0        ;SET SOURCE OPERAND FOR ADD
023716  006767  177764                   SXT     XOR6A             ;CLEAR XOR6A
023722  001004                           BNE     SXT6              ;CHECK CC'S AFTER EXTENDING ZERO'S
023724  100403                           BMI     SXT6
023726  103402                           BCS     SXT6
023730  102401                           BVS     SXT6
023732  000401                           BR      .+4
023734  104000                   SXT6:   HLT                       ;ERROR!  SXT FAILED

023736  012702  000001                   MOV     #1,R2             ;SET DEST OPERAND FOR ADD
023742  013703  001550                   MOV     @#FACTOR,R3       ;LOAD INDEX REGISTER
023746  060002                           ADD     R0,R2             ;RESULT OF ADD=100000
023750  006763  023706                   SXT     XOR6A(3)          ;EXTEND SIGN OF ADD ABOVE
023754  001403                           BEQ     SXT6A
023756  005267  177724                   INC     XOR6A             ;CHECK RESULT OF SXT
023762  001401                           BEQ     .+4
023764  104000                   SXT6A:  HLT                       ;ERROR!  SXT ABOVE FAILED TO EXTEND
                                                                   ;SIGN
023766  010703                           MOV     PC,R3
023770  000402                           BR      .+6               ;PRESERVE 2 WORDS FOR DATA
023772  000000                   SXRA:   .WORD   0                 ;RESERVED WORD FOR DATA
023774  000000                   SXRB:   .WORD   0                 ;RESERVED WORD FOR DATA
023776  005723                           TST     (R3)+
024000  010304                           MOV     R3,R4             ;R3 = ADDRESS OF SXRA
024002  000250                           CLN                       ;CLEAR N BIT
024004  006724                           SXT     (R4)+             ;EXTEND ZEROS INTO SXRA
024006  001401                           BEQ     .+4
024010  104000                   SXT2:   HLT                       ;ERROR!  SXT FAILED

024012  010467  177754                   MOV     R4,SXRA           ;SXRA = ADDRESS OF SXRB
024016  000257                           CCC                       ;CLEAR CONDITION CODES
024020  006733                           SXT     @(R3)+            ;EXTEND ZEROS INTO SXRB
024022  001401                           BEQ     .+4
024024  104000                   SXT3:   HLT                       ;ERROR!

024026  000270                           SEN                       ;SET N BIT
024030  006753                           SXT     @-(R3)            ;EXTEND ONES INTO SXRB
024032  100401                           BMI     .+4
024034  104000                   SXT5:   HLT                       ;ERROR!

024036  012704  025252                   MOV     #025252,R4        ;R4 = 025252
024042  074433                           XOR     R4,@(R3)+         ;SXRB = 152525 (COMPLEMENT OF R4)
```

```
024044  005002                          CLR     R2
024046  074253                          XOR     R2,@-(R3)       ;SXRB REMAINS UNCHANGED
024050  001405                          BEQ     XOR35           ;CHECK CONDITION CODES
024052  100004                          BPL     XOR35
024054  005104                          COM     R4              ;R4 = 152525
024056  020467  177712                  CMP     R4,SXRB         ;CHECK XOR
024062  001401                          BEQ     .+4
024064  104000                  XOR35:  HLT                     ;ERROR!  XOR FAILED

024066  005743                          TST     -(R3)           ;R3 = ADDRESS OF SXRA-2
024070  000250                          CLN                     ;CLEAR N BIT
024072  006773  000002                  SXT     @2(R3)          ;SXRB = 0
024076  001401                          BEQ     .+4
024100  104000                  SXT7:   HLT                     ;ERROR!  SXT FAILED

024102  074473  000002                  XOR     R4,@2(R3)       ;SXRB = R4
024106  020473  000002                  CMP     R4,@2(R3)       ;CHECK XOR
024112  001401                          BEQ     .+4
024114  104000                  XOR7:   HLT                     ;ERROR!  XOR FAILED
                                ;;*************************************************************
                                ;*TEST 47       SOB TEST
                                ;*      NOTE:   DO NOT INSERT ANY CODE IN FOLLOWING SOB TESTS
                                ;               SINCE IT TESTS THE MAXIMUM BRANCH WIDTH OF THE INSTRUCTION.
                                ;;*************************************************************
024116                          TST47:
024116  000004                          SCOPE
024120  112737  000047  001202          MOVB    #47,@#$TSTNM

024126  005005                          CLR     R5              ;CLEAR ERROR INDICATOR
024130  000407                          BR      SOB0            ;BRANCH TO SOB TEST

024132  005004                  SOB10:  CLR     R4              ;R4 = 0
024134  005705                          TST     R5              ;CHECK ERROR INDICATOR
024136  001401                          BEQ     .+4             ;SOB BRANCHED CORRECTLY
024140  104000                          HLT                     ;ERROR!

024142  005005                  SOB9:   CLR     R5              ;CLEAR INDICATOR (R5)
024144  006004                          ROR     R4              ;ROTATE RIGHT R4
024146  000467                          BR      SOB8

024150  012700  000010          SOB0:   MOV     #10,R0          ;R0=10
024154  000277                          SCC                     ;SET CONDITION CODES
024156  001012                  SOB1:   BNE     SOB2            ;CHECK CONDITION CODES AFTER SOB
024160  100011                          BPL     SOB2            ;SOB SHOULD NOT EFFECT THE
024162  102010                          BVC     SOB2            ;CONDITION CODES.
024164  103007                          BCC     SOB2
024166  077005                          SOB     R0,SOB1
024170  001005                          BNE     SOB2            ;CHECK CONDITION CODES AFTER
024172  100004                          BPL     SOB2            ;SOB FALLS THROUGH.
024174  102003                          BVC     SOB2            ;SOB SHOULD NOT EFFECT
024176  103002                          BCC     SOB2            ;CONDITION CODES.
024200  005700                          TST     R0              ;CHECK IF R0=0
024202  001401                          BEQ     .+4
024204  104000                  SOB2:   HLT                     ;ERROR!

024206  012702  000100                  MOV     #100,R2         ;R2=100
```

```
024212  012700  000101            MOV     #101,R0           ;SET CHECK REGISTER, R0=101
024216  001414            SOB3:   BEQ     SOB4              ;CHECK CONDITION CODES AFTER
024220  100413                    BMI     SOB4              ;SOB BRANCH.
024222  102412                    BVS     SOB4              ;SOB SHOULD NOT EFFECT
024224  103411                    BCS     SOB4              ;CONDITION CODES.
024226  005300                    DEC     R0                ;DECREMENT CHECK REGISTER
024230  020002                    CMP     R0,R2             ;CHECK THAT SOB DECREMENTS
024232  001006                    BNE     SOB4
024234  000257                    CCC                       ;SET CONDITION CODES BEFORE SOB
024236  077211                    SOB     R2,SOB3           ;BRANCH TO SOB3 UNTIL R2=0
024240  001403                    BEQ     SOB4              ;CHECK CONDITION CODES AFTER
024242  100402                    BMI     SOB4              ;SOB FALLS THROUGH
024244  005702                    TST     R2                ;CHECK IF R2=0
024246  001401                    BEQ     .+4
024250  104000            SOB4:   HLT                       ;ERROR!

024252  012700  000001    SOB5:   MOV     #1,R0             ;R0=1
024256  000401                    BR      .+4
024260  104000                    HLT                       ;ERROR!
024262  077002                    SOB     R0,.-2            ;SOB SHOULD NOT BRANCH

024264  005700                    TST     R0                ;CHECK IF R0=0 AFTER SOB
024266  001401                    BEQ     .+4
024270  104000                    HLT                       ;ERROR!

024272  012704  100000    SOB5A:  MOV     #100000,R4        ;R4=100000
024276  000403                    BR      1$
024300  005204            3$:     INC     R4                ;R4=100000
024302  100403                    BMI     2$                ;N BIT SHOULD BE SET
024304  104000                    HLT                       ;ERROR!   SOB DID NOT
                                                            ;INCREMENT PROPERLY

024306  077404            1$:     SOB     R4,3$             ;SOB SHOULD BRANCH
024310  104000                    HLT                       ;ERROR!  SOB DID NOT BRANCH

024312  012703  000100    2$:     MOV     #100,R3           ;R3=100
024316  077301            SOB6:   SOB     R3,SOB6           ;USE SOB TO BRANCH TO ITSELF
024320  005703                    TST     R3                ;CHECK IF R3=0
024322  001703                    BEQ     SOB10
024324  104000            SOB7:   HLT                       ;ERROR!

024326  005705            SOB9:   TST     R5                ;CHECK INDICATOR (R5)
                                                            ;IF SOB BRANCHES INCORRECTLY
                                                            ;WHEN CHECKING MAX. BRANCH,
                                                            ;R5 WILL NOT BE CLEARED AT
                                                            ;THIS POINT INDICATING AN ERROR.

024330  001401                    BEQ     .+4               ;BRANCH IF SOB BRANCHES CORRECTLY
024332  104000                    HLT                       ;ERROR!

024334  005205                    INC     R5                ;SET INDICATOR (R5)
024336  077477                    SOB     R4,SOB9           ;TEST MAX. BRANCH OF SOB
024340  005704                    TST     R4                ;CHECK IF R4=0
024342  001401                    BEQ     .+4
024344  104000                    HLT                       ;ERROR!
                          ;;*********************************************************************
```

```
                                                  ;*TEST 50       CHECK THE MARK INSTRUCTION
                                                  ;;*************************************************************
024346                                            TST50:
024346  000004                                            SCOPE
024350  112737  000050  001202                            MOVB    #50,@#$TSTNM
024356  010602                                    MRKTST: MOV     SP,R2
024360  010705                                            MOV     PC,R5           ;THE STACK LOOKS LIKE THIS AFTER
024362  010500                                            MOV     R5,R0           ;THE JSR INSTRUCTION
024364  010546                                            MOV     R5,-(SP)        ; -2(SP)= R0    THIS IS A
024366  010746                                            MOV     PC,-(SP)        ; -4(SP)= PC    STRING
024370  010746                                            MOV     PC,-(SP)        ; -6(SP)= PC+2  OF
024372  010746                                            MOV     PC,-(SP)        ;-10(SP)= PC+4  FIVE
024374  010746                                            MOV     PC,-(SP)        ;-12(SP)= PC+6  DUMMY
024376  010746                                            MOV     PC,-(SP)        ;-14(SP)= PC+10 ARGUMENTS
024400  012746  006405                                    MOV     #MARK+5,-(SP)   ;-16(SP)= MARK 5
024404  010605                                            MOV     SP,R5           ;-20(SP)= PC PUSHED BY JSR
024406  004767  000002                                    JSR     PC,MARK1
024412  000403                                            BR      .+10
024414  000205                                    MARK1:  RTS     R5
024416  104000                                            HLT                     ;ERROR! SHOULD BE DOING MARK 5 INST.
024420  000407                                            BR      MARKEX
024422  020602                                            CMP     SP,R2
024424  001402                                            BEQ     .+6
024426  104000                                            HLT                     ;ERROR! SP NOT RETURNED TO PROPER
024430  000403                                            BR      MARKEX          ;VALUE BY MARK INSTRUCTION
024432  020005                                            CMP     R0,R5
024434  001401                                            BEQ     .+4
024436  104000                                            HLT                     ;ERROR! DID NOT RESTORE R5 FROM STACK
024440  010206                                    MARKEX: MOV     R2,SP           ;RESTORE SP
                                                  ;;*************************************************************
                                                  ;*TEST 51        RTT/RTI TEST
                                                  ;*       RTT/RTI TEST INSURES THAT CP DOES THE INSTRUCTION FOLLOWING
                                                  ;*       AN RTT IF THE "T"BIT IS SET IN THE PSW,BUT DOES HONOR
                                                  ;*       THE TRAP IMMEDIATELY IF IT EXECUTES AN RTI
                                                  ;*       INSTRUCTION SEQUENCE-RTT
                                                  ;*       2$:     RTT                     ;NO 'T' TRAP AFTER RTT
                                                  ;*               INC     R0              ;R0=000001
                                                  ;*                                       ;'T' TRAP TO 5$ AFTER INC
                                                  ;*       5$:     COM     R0              ;R0=177776
                                                  ;*               MOV     SAVPSW,2(SP)    ;CLEAR 'T' BIT IN RETURN PSW
                                                  ;*               RTI                     ;RETURN TO INSTRUCTION FOLLOWING INC
                                                  ;*               CMP     #RTT,2$         ;CHECK
                                                  ;*                       ETC

                                                  ;*       INSTRUCTION SQUENCE-RTI
                                                  ;*       2$:     RTI                     ;'T' TRAP AFTER RTI
                                                  ;*       5$:     COM     R0              ;R0=177777
                                                  ;*               MOV     SAVPSW,2(SP)    ;CLEAR 'T' BIT IN RETURN PSW
                                                  ;*               RTI                     ;RETURN TO INC INSTRUCTION
                                                  ;*               INC     R0              ;R0=000000
                                                  ;*               CMP     #RTT,2$         ;CHECK
                                                  ;*                       ETC
                                                  ;;*************************************************************
024442                                            TST51:
024442  000004                                            SCOPE
024444  112737  000051  001202                            MOVB    #51,@#$TSTNM
```

```
024452 013767 177776 000174  RTT1:   MOV     @#PSW,SAVPSW            ;SAVE PSW
024460 032767 000020 000166          BIT     #20,SAVPSW              ;CHECK IF "T"BIT SET
024466 001143                        BNE     RTT2EX                  ;BRANCH TO EXIT
024470 010746                1$:     MOV     PC,-(SP)                ;GET CURRENT PC
024472 062716 000116                 ADD     #5$-.,(SP)              ;FORM RELOCATED PC
024476 012637 000014                 MOV     (SP)+,@#TBITVEC         ;LOAD INTO TRAP VECTOR
024502 016746 000146                 MOV     SAVPSW,-(SP)            ;GET CURRENT PSW
024506 011637 000016                 MOV     (SP),@#TBITVEC+2
024512 052737 000340 177776          BIS     #PR7,@#PSW              ;SET PRIORITY LEVEL 7
024520 005000                        CLR     R0
024522 052716 000360                 BIS     #PR7+20,(SP)            ;SET "T"BIT IN PSW ON STACK
024526 010746                        MOV     PC,-(SP)                ;PUT THE PC ON THE STACK
024530 062716 000006                 ADD     #6,(SP)                 ;ADJUST PC FOR NEXT INSTRUCTION
024534 000006                2$:     RTT
024536 005200                        INC     R0                      ;DONE TO SEE IF INSTR. FOLLOWING
                                                                     ;RTT IS EXECUTED IF T-BIT SET
024540 042737 000340 177776          BIC     #PR7,@#PSW              ;SET PRIORITY LEVEL 0
024546 022767 000006 177760          CMP     #RTT,2$
024554 001005                        BNE     3$                      ;CHECK IF INC WAS EXECUTED
024556 022700 177776                 CMP     #177776,R0              ;CHECK IF COM-R0 EXECUTED
024562 001406                        BEQ     4$
024564 104000                        HLT                             ;ERROR!R0 NOT COMPLIMENTED
024566 000415                        BR      6$                      ;EXIT TEST
024570 005700                3$:     TST     R0                      ;TEST IF TRAPED BEFORE INC INST.
                                                                     ;WAS EXECUTED
024572 001413                        BEQ     6$
024574 104000                        HLT                             ;ERROR!
024576 000411                        BR      6$                      ;EXIT TEST
024600 012767 000002 177726  4$:     MOV     #RTI,2$
024606 000730                        BR      1$
024610 005100                5$:     COM     R0                      ;RTT CHECK
024612 016766 000036 000002          MOV     SAVPSW,2(SP)
024620 000002                        RTI
024622 012767 000006 177704  6$:     MOV     #RTT,2$
024630 012737 001534 000014          MOV     #$RTRN,@#TBITVEC                ;RESTORE 'T' TRAP VECTOR
024636 005037 000016                 CLR     @#TBITVEC+2
024642                       RTT1EX:
                             ;;*******************************************************************
                             ;*TEST 52        SECOND RTT TEST
                             ;;*******************************************************************
024642                       TST52:
024642 000004                        SCOPE
024644 112737 000052 001202          MOVB    #52,@#$TSTNM
024652 000401                        BR      RTT2A
024654 000000                SAVPSW: .WORD   0
024656 016700 177772         RTT2A:  MOV     SAVPSW,R0               ;GET SAVED PSW
024662 105000                        CLRB    R0                      ;CLEAR PRIOITY LEVEL,T, AND COND CODES
024664 012702 140000                 MOV     #UM,R2
```

```
  6102  024670  074002                  XOR     R0,R2
  6103  024672  001423                  BEQ     2$                      ;USER MODE
  6104  024674  032700  140000          BIT     #UM,R0
  6105  024700  001036                  BNE     RTT2EX
```

```

                                                ;TEST THAT RTT CLEARS BITS 12,13 & PRIORITY LEVEL BITS IN KERNEL MODE
024702  012737  030240  177776                  MOV     #PUM+PR5,@#PSW          ;GO TO KERNEL MODE
024710  012746  000100                          MOV     #PR2,-(SP)
024714  010746                                  MOV     PC,-(SP)
024716  062716  000006                          ADD     #1$-.,(SP)              ;FORM NEW PC
024722  000006                                  RTT
024724  013700  177776          1$:             MOV     @#PSW,R0                ;NOW USING REG SET 0
024730  022700  000100                          CMP     #PR2,R0         ;TESTS THE PSW AFTER THE RTT
024734  001420                                  BEQ     RTT2EX
024736  104000                                  HLT                             ;ERROR! INCORRECT PSW AFTER THE RTT
024740  000416                                  BR      RTT2EX

                                                ;TEST TO INSURE THAT RTI DOES NOT CLEAR BITS 12-15 IN USER MODE
024742  052737  030340  177776  2$:             BIS     #PUM+PR7,@#PSW
024750  005046                                  CLR     -(SP)
024752  010746                                  MOV     PC,-(SP)
024754  062716  000006                          ADD     #5$-.,(SP)
024760  000002                                  RTI                             ;ATTEMPS TO INSERT A PSW OF 0
024762  022737  170340  177776  5$:             CMP     #UM+PUM+PR7,@#PSW
024770  001402                                  BEQ     RTT2EX
024772  104000                                  HLT                             ;ERROR! RTI CLEARED BITS IN PSW
024774  000400                                  BR      RTT2EX

024776  016737  177652  177776  RTT2EX:         MOV     SAVPSW,@#PSW
025004  000004                  RELES:          SCOPE
025006  010702                                  MOV     PC,R2
025010  062702  000012                          ADD     #12,R2
025014  012707  036574                          MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
025020  000000                  REL55:          .WORD   0
                                ;5555555555555 LAST ADDRESS OF CODE TO BE RELOCATED 55555555555

                                ;;************************************************************
                                ;*TEST 53       CHECK ASH, ASHC, MUL, AND DIV INSTRUCTIONS
                                ;;************************************************************
025022                          TST53:
025022  012767  000001  154274                  MOV     #1,$TIMES       ;;DO 1 ITERATION
025030  000004                                  SCOPE
025032  112737  000053  001202                  MOVB    #53,@#$TSTNM

                                                .SBTTL  START OF SECTION 6
                                ;6666666666666 FIRST ADDRESS TO BE RELOCATED 666666666
025040  112737  000053  001202  REL6:           MOVB    #$TN-1,@#$TSTNM
025046  010700                                  MOV     PC,R0           ;GET PC
025050  005740                                  TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL6
025052  010037  001554                          MOV     R0,@#FRSTAD     ;SAVE
025056  010700                                  MOV     PC,R0           ;GET CURRENT PC
025060  162700  025060                          SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
025064  010037  001550                          MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
025070  010737  001212                          MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
025074  062737  000026  001212                  ADD     #26,@#$LPERR    ;ADJUST
025102  013737  001212  001210                  MOV     @#$LPERR,@#$LPADR
025110  105737  001544                          TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
025114  001402                                  BEQ     .+6
025116  000167  004402                          JMP     RELE6
025122  012700  000001          ASHLO:          MOV     #1,R0           ;R0 WILL BE THE SHIFT COUNT
```

```
025126  012703  000021                  MOV     #17.,R3         ;MAX SHIFT COUNT
025132  005067  000014          1$:     CLR     2$              ;PRESET SAVED CC'S LOCATION=0
025136  010002                          MOV     R0,R2           ;GET SHIFT COUNT FOR PASS
025140  010705                          MOV     PC,R5           ;R5 & R4 WILL BE DATA SHIFTED BY
025142  010504                          MOV     R5,R4           ;ASH & ASL INSTRUCTIONS
025144  072502                          ASH     R2,R5           ;SHIFT R5
025146  113727  177776                  MOVB    @#PSW,(PC)+     ;SAVE CC'S
025152  000000                  2$:     .WORD   0               ;CONTAINS ASH CC'S IN EVEN BYTE
                                                                ;ASL CC'S IN ODD BYTE
025154  006304                  3$:     ASL     R4              ;SHIFT R4
025156  113746  177776                  MOVB    @#PSW,-(SP)     ;SAVE PSW ON STACK
025162  132716  000002                  BITB    #2,(SP)         ;CHECK IF ASL SET V BIT
025166  001403                          BEQ     30$
025170  152767  000002  177755          BISB    #2,2$+1         ;IF ASL SET V THEN SET V IN 2$+1
025176  112637  177776          30$:    MOVB    (SP)+,@#PSW     ;RESTORE ORIGINAL PSW
025202  077214                          SOB     R2,3$           ;SHIFT R4 R2 TIMES
025204  153767  177776  177741          BISB    @#PSW,2$+1      ;SAVE CC'S AFTER ASL
025212  020504                          CMP     R5,R4           ;CHECK ASH & ASL RESULTS
025214  001004                          BNE     4$
025216  126767  177730  177727          CMPB    2$,2$+1         ;CHECK ASH & ASL CC'S
025224  001401                          BEQ     .+4
025226  104000                  4$:     HLT                     ;ERROR! INCORRECT RESULT OR CC'S
025230  005200                          INC     R0              ;INCREMENT PASS SHIFT COUNT
025232  020003                          CMP     R0,R3
025234  001336                          BNE     1$

025236  012700  177777          ASHR0:  MOV     #-1,R0          ;R0 = RIGHT SHIFT COUNT FOR PASS
025242  012703  177757                  MOV     #-17.,R3        ;MAX SHIFT COUNT
025246  010002                  1$:     MOV     R0,R2           ;GET SHIFT COUNT FOR PASS
025250  010705                          MOV     PC,R5           ;R5 & R4 = DATA TO BE SHIFTED
025252  010504                          MOV     R5,R4           ;BY ASH & ASR INSTRUCTIONS
025254  072502                          ASH     R2,R5           ;SHIFT R5 R2 TIMES
025256  113727  177776                  MOVB    @#PSW,(PC)+     ;SAVE CC'S IN EVEN BYTE
025262  000000                  2$:     .WORD   0               ;CONTAINS ASH CC'S IN EVEN BYTE
                                                                ;ASR CC'S IN ODD BYTE
025264  005402                          NEG     R2
025266  006204                  3$:     ASR     R4              ;SHIFT R4
025270  077202                          SOB     R2,3$           ;SHIFT R4 R2 TIMES
025272  113767  177776  177763          MOVB    @#PSW,2$+1      ;SAVE CC'S AFTER ASR
025300  142767  000002  177755          BICB    #2,2$+1         ;ASH RIGHT WILL NOT SET V ASR MAY SET V
025306  020504                          CMP     R5,R4           ;CHECK ASH & ASR RESULTS
025310  001004                          BNE     4$
025312  126767  177744  177743          CMPB    2$,2$+1         ;CHECK ASH & ASR CC'S
025320  001401                          BEQ     .+4
025322  104000                  4$:     HLT
025324  005300                          DEC     R0              ;DECREMENT PASS SHIFT COUNT
025326  020003                          CMP     R0,R3
025330  001346                          BNE     1$

025332  012746  000037          ASHCL0: MOV     #31.,-(SP)      ;PUT MAX SHIFT COUNT ON STACK
025336  012746  000001                  MOV     #1,-(SP)        ;PUT LEFT SHIFT COUNT ON STACK
025342  011600                  1$:     MOV     (SP),R0         ;GET PASS SHIFT COUNT
025344  010705                          MOV     PC,R5           ;CURRENT PC IS DATA TO BE SHIFTED
025346  010503                          MOV     R5,R3           ;ASHC SHIFTS R4,R5;ASL,ROL SHIFTS R2,R3
025350  005004                          CLR     R4
025352  005002                          CLR     R2
```

G11


```
025354  073400                          ASHC    R0,R4           ;SHIFT R4 LEFT AS SPECIFIED BY R0
0[illegible]56  006303          2$:     ASL     R3              ;SHIFT R2,R3 LEFT
0[illegible]0  006102                   ROL     R2              ;AS SPECIFIED BY R0
025362  077003                          SOB     R0,2$
025364  020402                          CMP     R4,R2           ;CHECK RESULTS
0[illegible]6  001002                   BNE     3$
0[illegible]70  020503                  CMP     R5,R3
025372  001401                          BEQ     .+4
025374  104000                  3$:     HLT
025376  005216                          INC     (SP)            ;INCREMENT NEXT PASS SHIFT COUNT
025400  021666  000002                  CMP     (SP),2(SP)      ;REACHED MAX COUNT (31.)
025404  001356                          BNE     1$
025406  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK PTR

025410  012746  177740          ASHCR0: MOV     #-32.,-(SP)     ;PUT MAX RIGHT SHIFT COUNT ON STACK
025414  012746  177777                  MOV     #-1,-(SP)       ;PUT PASS SHIFT COUNT ON STACK
0[illegible]420  011600         1$:     MOV     (SP),R0         ;GET PASS SHIFT COUNT
025422  010702                          MOV     PC,R2           ;R2,R3 & R4,R5 ARE THE DATA REGISTERS
025424  010204                          MOV     R2,R4           ;TO BE SHIFTED BY TEST
0[illegible]4[illegible]6  005003       CLR     R3
0[illegible]430  005005                 CLR     R5
025432  000262                          SEV                     ;SET V BIT IN PSW
0[illegible]434  073200                 ASHC    R0,R2           ;SHIFT R2,R3 RIGHT R0 TIMES
025436  102410                          BVS     3$              ;SHIFT RIGHT CLEARS V
025440  005400                          NEG     R0              ;NEGATE SHIFT COUNT FOR SOB
025442  006204                  2$:     ASR     R4              ;SHIFT R4,R5 RIGHT R0 TIMES
025444  006005                          ROR     R5
025446  077003                          SOB     R0,2$
025450  020204                          CMP     R2,R4           ;CHECK RESULT
025452  001002                          BNE     3$
025454  020305                          CMP     R3,R5
025456  001401                          BEQ     .+4
025460  104000                  3$:     HLT
025462  005316                          DEC     (SP)            ;SET SHIFT COUNT FOR NEXT PASS
025464  021666  000002                  CMP     (SP),2(SP)      ;CHECK IF MAX SHIFT COUNT
025470  001353                          BNE     1$
025472  022626                          CMP     (SP)+,(SP)+     ;RESTORE STACK PTR
                                ;;*************************************************************
                                ;*TEST 54        CHECK MUL
                                ;*       THE BELOW TEST OF THE MUL INSTRUCTION MULTIPLIES THE CURRENT PC
                                ;*       BY 1,2,4,8 ETC AND SHIFTS THE SAME PC VALUE USING AN ASHC LEFT BY
                                ;*       0,1,2,3,ETC AND COMPARES THE RESULTS. CONDITION CODE RESULTS ARE NOT CHECKED.
                                ;;*************************************************************
025474                          TST54:
025474  000004                          SCOPE
025476  112737  000054  001202          MOVB    #54,@#$TSTNM
025504  012700  000001          MUL0:   MOV     #1,R0           ;R0 CONTAINS MULTIPLIER FOR MUL
025510  012706  000700                  MOV     #USESTK,SP      ;SETUP THE STACK
025514  005016                          CLR     (SP)            ;(SP) CONTAINS SHIFT VALUE FOR ASHC
025516  010702                  1$      MOV     PC,R2           ;R3,R2 & R5,R4 ARE DATA REGISTERS
025520  010227                          MOV     R2,(PC)+        ;SAVE MULTIPICAND
025522  000000                          .WORD   0               ;CONTAINS ORIGINAL MULTIPICAND
025524  005003                          CLR     R3
025526  005004                          CLR     R4
025530  010205                          MOV     R2,R5           ;FOR MUL AND ASHC
025532  100001                          BPL     .+4             ;IF MULTPICAND IS NEG THEN SET R4 = -1
```

```
025534  005104                          COM     R4              ;FOR ASHC
025536  000277                          SCC                     ;PRESET CC'S
025540  070200                          MUL     R0,R2           ;MULTIPLY R2 BY R0 LEAVE PRODUCT
                                                                ;IN R2,R3 MSH IN R2,LSH IN R3
025542  102406                          BVS     2$
025544  001405                          BEQ     2$              ;PRODUCT WILL NEVER BE = 0
025546  073416                          ASHC    (SP),R4         ;'MULTIPLY' R4,R5 BY (SP) LEAVE PRODUCT
                                                                ;IN R4,R5 MSH IN R4,LSH IN R5
025550  020204                          CMP     R2,R4           ;CHECK MSH RESULT
025552  001002                          BNE     2$
025554  020305                          CMP     R3,R5           ;CHECK LSH RESULT
025556  001401                          BEQ     .+4
025560  104000                  2$:     HLT
025562  005216                          INC     (SP)            ;INCREMENT ASHC SHIFT COUNT
025564  006300                          ASL     R0              ;SHIFT MUL MULTIPLIER
025566  102353                          BVC     1$
                                ;CHECK MUL INST WITH MULTIPIER (R0) = 100000
025570  010702                          MOV     PC,R2           ;R2 = MULTIPICAND
025572  005202                          INC     R2
025574  010227                          MOV     R2,(PC)+        ;SAVE MULTIPICAND
025576  000000                          .WORD   0               ;CONTAINS ORIGINAL MULTIPICAND
025600  005103                          COM     R3
025602  010204                          MOV     R2,R4           ;R4 WILL BE MSH 'PRODUCT'
025604  006204                          ASR     R4              ;FORM 'PRODUCT'
025606  005104                          COM     R4              ;COMPLEMENT MSH 'PRODUCT'
025610  070200                          MUL     R0,R2           ;MULTIPLY R2 BY 100000 LEAVING
                                                                ;R2 = MSH, R3 = LSH PRODUCT
025612  020204                          CMP     R2,R4           ;COMPARE MSH PRODUCTS
025614  001002                          BNE     3$
025616  020003                          CMP     R0,R3           ;CHECK LSH PRODUCT
025620  001401                          BEQ     .+4
025622  104000                  3$:     HLT
                                ;;*************************************************************
                                ;*TEST 55        CHECK THE DIV INSTRUCTION
                                ;*       THE BELOW TEST OF THE DIV INSTRUCTION DIVIDES THE CURRENT PC BY
                                ;*       1,2,4,8,ETC LEAVING THE QUOTIENT/REMAINDER IN R2/R3. NEXT THE QUOTIENT
                                ;*       IS MULTPLIED BY 1,2,4,8,ETC AND THE REMAINDER ADDED. THE RESULT IS
                                ;*       THEN COMPARED WITH THE ORIGINAL CURRENT PC.
                                ;;*************************************************************
025624                          TST55:
025624  000004                          SCOPE
025626  112737  000055  001202          MOVB    #55,@#$TSTNM
025634  012700  000001          DIV0:   MOV     #1,R0           ;R0=DIVISOR
025640  010716                          MOV     PC,(SP)         ;SAVE DATA ON STACK
025642  011603                  1$:     MOV     (SP),R3         ;GET DATA
025644  005002                          CLR     R2              ;CLEAR MSH DIVIDEND
025646  000277                          SCC
025650  071200                          DIV     R0,R2           ;DIVIDE R2 BY R0 LEAVING QUOTIENT IN R2
                                                                ;AND REMAINDER IN R3
025652  103417                          BCS     2$
025654  100416                          BMI     2$
0[illegible]56  102007                  BVC     20$             ;BRANCH IF DIVIDE WORKED
[illegible]60  022700  000001           CMP     #1,R0           ;V BIT SHOULD ONLY SET IF DIVIDING BY 1
025664  001012                          BNE     2$              ;AND THE LSH OF DIVIDEND
025666  032716  100000                  BIT     #100000,(SP)    ;IS NEGATIVE
025672  001407                          BEQ     2$
```

```
025674  000407                                 BR      3$
025676  010204                         20$:    MOV     R2,R4              ;GET QUOTIENT
025700  070400                                 MUL     R0,R4              ;MULTIPLY QUOTIENT BY DIVISOR
025702  060305                                 ADD     R3,R5              ;ADD REMAINDER TO LSH PRODUCT
025704  103402                                 BCS     2$                 ;SHOULD BE NO CARRY
025706  021605                                 CMP     (SP),R5            ;CHECK RESULT
025710  001401                                 BEQ     .+4
025712  104000                         2$:     HLT                        ;ERROR! DIVIDE FAILED
                                                                          ;QUOTIENT IS IN R2,REMAINDER IN R3
                                                                          ;ORIGINAL PC IS ON STACK AND FINAL
                                                                          ;PRODUCT IN R4,R5 [MSH][LSH]
025714  006300                         3$:     ASL     R0                 ;GET NEXT DIVISOR
025716  102351                                 BVC     1$

                                       ;CHECK ASH,ASHC,MUL, AND DIV INSTRUCTIONS USING ADDRESS MODE 1
025720  005016                         ASHL1:  CLR     (SP)               ;(SP) = SHIFT COUNT
025722  005000                                 CLR     R0                 ;R0 = SHIFT COUNT FOR CHECK ASH
025724  012702  000020                         MOV     #16.,R2            ;R2 = MAX LEFT SHIFT COUNT
025730  005067  000012                 1$:     CLR     2$                 ;CLEAR CC'S HOLDING ADDRESS
025734  010703                                 MOV     PC,R3              ;R3,R4 = DATA TO BE SHIFTED
025736  010304                                 MOV     R3,R4
025740  072316                                 ASH     (SP),R3            ;SHIFT R3 LEFT (SP) TIMES
025742  013727  177776                         MOV     @#PSW,(PC)+        ;SAVE CC'S
025746  000000                         2$:     .WORD   0                  ;CONTAINS ASH (SP),R3 CC'S IN EVEN BYTE
                                                                          ;AND ASH R0,R4 CC'S IN ODD BYTE
025750  072400                                 ASH     R0,R4              ;SHIFT R4 LEFT R0 TIMES
025752  113767  177776  177767                 MOVB    @#PSW,2$+1         ;SAVE CC'S IN ODD BYTE OF 2$
025760  020304                                 CMP     R3,R4              ;COMPARE RESULTS
025762  001004                                 BNE     3$                 ;BRANCH IF THEY DO NOT COMPARE
025764  126767  177756  177755                 CMPB    2$,2$+1            ;CHECK CC'S AFTER ASH INSTRUCTIONS
025772  001401                                 BEQ     .+4
025774  104000                         3$:     HLT                        ;ERROR! EITHER RESULTS OF SHIFT OR
                                                                          ;RESULT CC'S ARE INCORRECT
025776  005200                                 INC     R0                 ;INCREMENT SHIFT COUNT FOR ASH R0,R4
026000  005216                                 INC     (SP)               ;INCREMENT SHIFT COUNT FOR ASH (SP),R3
026002  020200                                 CMP     R2,R0              ;CHECK FOR MAX SHIFT COUNT
026004  001351                                 BNE     1$

026006  005016                         ASHR1:  CLR     (SP)               ;(SP) = SHIFT COUNT FOR ASH (SP),R4
026010  005000                                 CLR     R0                 ;R0 = SHIFT COUNT FOR ASH R0,R5
026012  005402                                 NEG     R2                 ;R2 = MAX RIGHT SHIFT COUNT (SET BY
                                                                          ;ABOVE TEST TO 16. NOW = -16.
026014  005067  000012                 1$:     CLR     2$                 ;CLEAR CC'S HOLDING ADDRESS
026020  010704                                 MOV     PC,R4              ;R4,R5 = DATA TO BE SHIFTED RIGHT
026022  010405                                 MOV     R4,R5
026024  072416                                 ASH     (SP),R4            ;SHIFT R4 RIGHT (SP) TIMES
026026  013727  177776                         MOV     @#PSW,(PC)+        ;SAVE CC'S
026032  000000                         2$:     .WORD   0                  ;CONTAINS ASH (SP),R4 CC'S IN EVEN BYTE
                                                                          ;AND ASH R0,R5 CC'S IN ODD BYTE
026034  072500                                 ASH     R0,R5              ;SHIFT R5 RIGHT R0 TIMES
026036  113767  177776  177767                 MOVB    @#PSW,2$+1         ;SAVE CC'S IN ODD BYTE 2$
026044  020405                                 CMP     R4,R5              ;CHECK RESULTS
026046  001004                                 BNE     3$
026050  126767  177756  177755                 CMPB    2$,2$+1            ;CHECK RESULT CC'S
026056  001401                                 BEQ     .+4
026060  104000                         3$:     HLT                        ;ERROR! EITHER RESULTS OR RESULT CC'S
```

```
                                                                   ;DID NOT COMPARE
026062  005300                              DEC     R0             ;DECREMENT SHIFT COUNT
026064  005316                              DEC     (SP)           ;DECREMENT SHIFT COUNT FOR ASH (SP),R4
026066  020002                              CMP     R0,R2          ;CHECK FOR MAX RIGHT SHIFT
026070  001351                              BNE     1$
                                ;;***********************************************************
                                ;*TEST 56       DIVIDE AGAIN
                                ;*      THE BELOW TEST CHECKS THE DIVIDE INSTRUCTION BY DIVIDING
                                ;*      THE CURRENT PC BY ITSELF+1. THE QUOTIENT (IN R2) ALWAYS = 0,
                                ;*      AND THE REMAINDER (IN R3) ALWAYS = THE CURRENT PC.
                                ;;***********************************************************
026072                          TST56:
026072  000004                              SCOPE
026074  112737  000056  001202              MOVB    #56,@#$TSTNM
026102  010703                  DIV1:       MOV     PC,R3          ;CURRENT PC IS LSH DIVIDEND
026104  006702                              SXT     R2             ;EXTEND SIGN TO R2 (MSH DIVIDEND)
026106  010304                              MOV     R3,R4          ;SAVE ORIGINAL DIVIDEND
026110  010316                              MOV     R3,(SP)        ;PUT ON STACK
026112  005216                              INC     (SP)           ;ADD 1 (WILL BE DIVISOR)
026114  100002                              BPL     1$             ;BRANCH IF POSITIVE
026116  162716  000002                      SUB     #2,(SP)        ;MAKE DIVISOR 1 LESS THAN DIVIDEND
026122  071216                  1$:         DIV     (SP),R2        ;DIVIDE R2 BY (SP)
026124  103410                              BCS     2$             ;CHECK CONDITION CODES
026126  102407                              BVS     2$
026130  001006                              BNE     2$
026132  100405                              BMI     2$
026134  005702                              TST     R2             ;CHECK QUOTIENT (R2 = 0)
026136  001361                              BNE     DIV1
026140  010416                              MOV     R4,(SP)        ;GET ORIGINAL DIVISOR
026142  020316                              CMP     R3,(SP)        ;CHECK REMAINDER
026144  001401                              BEQ     .+4
026146  104000                  2$:         HLT                    ;REPORT ERROR





                                ;*
                                ;*      THIS SECTION OF THE MED TESTS EXERCISES CERTAIN SCRATCH
                                ;*      PAD REGISTERS USING MED READS AND WRITES.  THEIR ORIGINAL
                                ;*      CONTENTS ARE RESTORED BUT:
                                ;*
                                ;*      ****************** IMPORTANT NOTE *******************
                                ;*      *                                                   *
                                ;*      *  THE CONSOLE MUST NOT !!! BE USED DURING THESE MED  *
                                ;*      *  TESTS. NO INTERRUPTS OR TRAPS CAN BE ALLOWED EITHER*
                                ;*      *  (THE T-BIT IS CLEARED FOR THESE TESTS)            *
                                ;*      *****************************************************
                                ;*

                                ;;***********************************************************
                                ;*TEST 57       CHECK MED IS ILLEGAL IN USER
                                ;*      THE NEXT TWO TESTS BELOW CHECK TO SEE THAT THE "MED"
                                ;*      (MAINTENANCE, EXAM, AND DEPOSIT) INSTRUCTION WILL EXECUTE
                                ;*      WHEN IN KERNEL MODE AND THAT IT IS ILLEGAL IN
                                ;*      USER MODE
```

```
                                        ;;*************************************************************
026150                                  TST57:
026150  000004                                  SCOPE
026152  112737  000057  001202                  MOVB    #57,@#$TSTNM
026160  012737  000001  001324                  MOV     #1,@#$TIMES             ;DO 1 ITERATION
026166  013737  177776  002362          MED1:   MOV     @#PSW,@#PSWHOL          ;SAVE PSW
026174  004737  054112                          JSR     PC,@#CLRTBIT            ;GO CLEAR "T-BIT" IF SET
026200  012700  000340                          MOV     #340,R0                 ;SET "VECTOR+2'S" UP FOR MED TESTS
026204  010037  000006                          MOV     R0,@#6
026210  010037  000012                          MOV     R0,@#12
026214  010037  000022                          MOV     R0,@#22
026220  010037  000032                          MOV     R0,@#32
026224  010037  000116                          MOV     R0,@#116
026230  012737  140340  177776                  MOV     #140340,@#PSW           ;GO TO USER MODE
026236  012706  000700                          MOV     #USESTK,SP              ;SETUP USER STACK PTR.
026242  010746                                  MOV     PC,-(SP)                ;GET CURRFNT PC
026244  062716  000032                          ADD     #2$-.,(SP)
026250  011637  000004                          MOV     (SP),@#ERRVEC           ;SET ERROR TRAP VECTOR TO 2$ BELOW
026254  012637  000010                          MOV     (SP)+,@#RESVEC          ;LOAD RESERVED INST. TRAP VECTOR
026260  012701  177777                          MOV     #-1,R1                  ;LOAD R1 WITH A -1
026264  005000                                  CLR     R0                      ;CLEAR R0
026266  076600                                  MED                             ;TRY TO DO MAINT. EXAMINE
026270  000041                                  .WORD   041                     ;MED READ CODE FOR R1
026272  104000                          1$:     HLT                             ;ERROR - MED INST. NOT ILLEGAL IN USER
026274  000407                                  BR      4$
026276  005700                          2$:     TST     R0                      ;IS R0 UNCHANGED?
026300  001401                                  BEQ     3$                      ;BRANCH IF YES
026302  104000                                  HLT                             ;ERROR - MED INSTRUCTION WAS EXECUTED
                                                                                ;BEFORE TRAPPING
026304  010716                          3$:     MOV     PC,(SP)                 ;REPLACE RETURN PC WITH
026306  062716  000006                          ADD     #4$-.,(SP)              ;ADDRESS OF 4$ BELOW
026312  000002                                  RTI                             ;RETURN (TO 4$)
026314  012737  055064  000004          4$:     MOV     #ERPRT,@#ERRVEC;RESTORE ERROR TRAP VECTOR
026322  012737  054774  000010                  MOV     #RESERR,@#RESVEC;RESTORE RESERVED INST. TRAP VECTOR
                                        ;;*************************************************************
                                        ;*TEST 60        CHECK MED DOESN'T AFFECT PSW
                                        ;;*************************************************************
026330                                  TST60:
026330  000004                                  SCOPE
026332  112737  000060  001202                  MOVB    #60,@#$TSTNM

026340  005037  177776                  MED0:   CLR     @#PSW                   ;GO TO KERNEL MODE
026344  000257                                  CCC                             ;CLEAR CONDITION CODE BITS
026346  076600                                  MED                             ;DO MAINT. EXAMINE OF R1
026350  000041                                  .WORD   041                     ;MED READ CODE FOR R1
026352  103403                                  BCS     MEDHLT
026354  102402                                  BVS     MEDHLT
026356  100401                                  BMI     MEDHLT
026360  001001                                  BNE     .+4
026362  104000                          MEDHLT: HLT                             ;ERROR CC-BITS IN PSW AFFECTED BY MED

                                        ;;*************************************************************
                                        ;*TEST 61        MED TEST - R/W DATA PATTERNS TO REGS
                                        ;*       THIS PARTICULAR MED TEST WRITES DATA PATTERNS
                                        ;*       TO THOSE INTERNAL REGS. WHICH CAN BE WRITTEN
                                        ;*       AND READ WITHOUT SPECIAL CONSIDERAIONS.  REGISTERS
```

L11


```
                                         ;*      REQUIRING SPECIAL TESTS ARE TESTED IN LATER
                                         ;*      MED TESTS.
                                         ;*      TABLE II CONTAINS THE REGISTER ADDRESSES.
                                         ;*
                                         ;;*************************************************************
026364                                   TST61:
026364  000004                                   SCOPE
026366  112737  000061  001202                   MOVB    #61,@#$TSTNM
0[illegible]374  012737  000340  177776  MEDT1:  MOV     #340,@#PSW          ;KERNEL MODE-PRIORITY 7
0[illegible]402  012701  002502                  MOV     #TBL2,R1            ;INITIALIZE ADDRESS POINTER
0[illegible]406  012737  125252  002476  1$:     MOV     #125252,@#MEDTP2
0[illegible]414  111167  00[illegible]036                  MOVB    (R1),11$            ;PUT WRITE CODE BY "WRITE-MED'S"
[illegible]0  112167  00[illegible]054                  MOVB    (R1)+,13$           ;AND POINT R1 TO READ CODE
[illegible]4  111167  00[illegible]006                  MOVB    (R1),10$            ;PUT READ CODE BY "READ-MED'S"
[illegible]430  112167  000030                  MOVB    (R1)+,12$           ;R1 NOW POINTS TO NEXT REG.
[illegible]434  076600                  2$:     MED                         ;MED-READ THE INTERNAL REG.
[illegible]6  00[illegible]0                  10$:    .WORD   0                   ;MED-READ CODE
0[illegible]440  010037  002472                  MOV     R0,@#MEDTP0         ;SAVE ITS ORIGINAL CONTENTS
0[illegible]444  010137  002474                  MOV     R1,@#MEDTP1         ;SAVE ADDR. PTR. VALUE
0[illegible]450  013700  002476                  MOV     @#MEDTP2,R0         ;LOAD R0 WITH DATA TO BE WRITTEN
0[illegible]454  076600                          MED                         ;MED-WRITE THE TEST DATA
0[illegible]456  000000                  11$:    .WORD   0                   ;MED-WRITE CODE
0[illegible]460  005000                          CLR     R0                  ;CLEAR R0
0[illegible]6462  076600                          MED                         ;MED-READ THE DATA BACK
0[illegible]6464  000000                  12$:    .WORD   0                   ;MED-READ CODE
0[illegible]6466  010037  002500                  MOV     R0,@#MEDTP3         ;SAVE DATA READ FOR COMPARISON
0[illegible]6472  013700  002472                  MOV     @#MEDTP0,R0         ;LOAD ORIGINAL DATA IN R0
0[illegible]6476  076600                          MED                         ;MED-WRITE ORG. DATA TO REG.
0[illegible]500  000000                  13$:    .WORD   0                   ;MED-WRITE CODE
0[illegible]6502  023737  002476  002500          CMP     @#MEDTP2,@#MEDTP3   ;DID DATA READ=DATA WRITTEN?
026510  001401                                   BEQ     3$                  ;BRANCH IF YES
026512  104000                                   HLT                         ;INT. REG. READ BACK WRONG DATA
                                                                             ;MEDCODE WILL BE AT 12$
                                                                             ;DATA EXPECTED IS IN MEDTP2
                                                                             ;DATA RECEIVED IS IN MEDTP3
026514  005137  002476                  3$:     COM     @#MEDTP2            ;CHANGE DATA PATTERN
026520  013701  002474                          MOV     @#MEDTP1,R1         ;RESTORE ADDR. POINTER
026524  022737  125252  002476                  CMP     #125252,@#MEDTP2    ;BOTH DATA PATTERNS BEEN USED?
0[illegible]532  001340                                   BNE     2$                  ;BRANCH IF NO
026534  005711                                   TST     (R1)                ;END OF ADDR. TABLE?
026536  001323                                   BNE     1$                  ;BRANCH IF NO



                                         ;;*************************************************************
                                         ;*TEST 62       MED TEST - CSP CONSTANTS CHECK
                                         ;*
                                         ;*      THIS TEST CHECKS THE CONSTANT VALUES LOCATED
                                         ;*      IN THE C SCRATCH PAD.  THE CONSTANTS ARE READ
                                         ;*      WITH A MED INSTRUCTION AND COMPARED TO THEIR
                                         ;*      EXPECTED VALUE.  THE ADDRESSES OF THESE CONSTANTS
                                         ;*      AND THE VALUES EXPECTED ARE IN TABLE VII.
                                         ;*
                                         ;;*************************************************************
026540                                   TST62:
```

```
 026540  000004                               SCOPE
 026542  112737  000062  001202               MOVB    #62,@#STSTNM

 026550  170000                       MEDT10: CFCC                            ;EXECUTE FLT. PT INST. SO FLT. PT.
                                                                              ;CONSTANTS ARE LOADED INTO CSP
 026552  012701  003034                       MOV     #TBL7,R1                ;SETUP TABLE POINTER
 026556  012167  000006               10$:    MOV     (R1)+,1$                ;LOAD MED READ CODE AT 1$
 026562  001412                               BEQ     11$                     ;EXIT IF END OF TEST
 026564  005000                               CLR     R0
 026566  076600                               MED                             ;READ INTERNAL CONTENTS INTO R0
 026570  000000                       1$:     .WORD   0
 026572  020021                               CMP     R0,(R1)+                ;DID I READ THE CONSTANT I EXPECTED
 026574  001770                               BEQ     10$                     ;BRANCH IF YES
 026576  016137  177776  002476               MOV     -2(R1),@#MEDTP2         ;SAVE CONSTANT VALUE EXPECTED
 026604  104000                               HLT                             ;CSP LOCATION HELD WRONG VALUE
                                                                              ;MEDCODE IS AT 1$
                                                                              ;DATA EXPECTED IS IN MEDTP2
                                                                              ;DATA RECEIVED IS IN R0 OR THE LOC. ITSELF
 026606  000763                               BR      10$                     ;BRANCH BACK TO TEST NEXT CONSTANT
 026610                               11$:



                                      ;;*****************************************************************
                                      ;*TEST 63        MED TEST - MICROBK CHECK OF MICRO-POINTS
                                      ;*
                                      ;*        THIS TEST USES THE MICROBREAK REGISTER AND THE
                                      ;*        INFORMATION IN TABLE V TO CHECK THAT THE
                                      ;*        CORRECT MED-FLOW IS ENTERED WHEN EACH
                                      ;*        REGISTER IS ACCESSED BY A MED INSTRUCTION.
                                      ;*        THE MICROBREAK REG. IS SETUP TO CAUSE A TRAP TO
                                      ;*        LOC. 4 WHEN ITS CONTENTS EQUAL THE ADDRESS
                                      ;*        OF THE MIRCOWORD BEING EXECUTED.
                                      ;*
                                      ;*            NOTE:   THE MICRO BREAK - TRAP-TO-4 CAPABILITY
                                      ;*                    IS TRIED AT THE BEGINNING OF THE TEST.
                                      ;*                    IF IT DOESN'T WORK, AN ERROR IS PRINTED
                                      ;*                    AND THE TEST IS SKIPPED
                                      ;*
                                      ;;*****************************************************************
 026610                               TST63:
 026610  000004                               SCOPE
 026612  112737  000063  001202               MOVB    #63,@#STSTNM
 026620  012737  000071  177770       MEDT11: MOV     #SWB01,@#UBREAK ;LOAD MICROBK. REG. WITH AN MICRO ADDR.
 026626  012737  055172  000004               MOV     #BKROUT,@#4     ;LOAD ADDR. OF MICROBK. ROUTINE IN 4
 026634  005037  055200                       CLR     @#BKFLAG        ;CLEAR MICROBK. TRAP FLAG
 026640  076600                               MED
 026642  000022                               RDWHAMI
 026644  042700  100001                       BIC     #BIT15+BIT0,R0  ;ENSURE LOG "CONTINUOUS" MODE
 026650  052700  001000                       BIS     #BIT9,R0
 026654  076600                               MED                     ;MED-WRITE THE WHAMI REG TO
 026656  000222                       10$:    WRWHAMI                 ;ENABLE MICROBK-TRAP-TO-4
 026660  076600                               MED
 026662  000144                               RDFLAG
 026664  052700  100000                       BIS     #BIT15,R0
 026670  076600                               MED                     ;MED-WRITE THE FLAG REG TO
 026672  000344                       11$:    WRFLAG                  ;ENABLE MICROBK TRAPPING
```

```
026674 000300                           SWAB    R0              ;MICROBK TRAP SHOULD OCCUR ON SWAB
026676 005737 055200                    TST     @#BKFLAG        ;DID TRAP TO 4 OCCUR?
026702 001020                           BNE     1$              ;BRANCH IF YES
026704 104000                           HLT                     ;MICROBREAK TRAP DIDN'T WORK
                                                                ;WHEN TRIED TO TRAP IN "SWAB" ROUTINE
                                                                ;THEREFORE REST OF TEST IS SKIPPED
026706 104401 026714                    TYPE    ,65$            ;;TYPE ASCIZ STRING
026712 000413                           BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$: .ASCIZ  <15><12>?SKIP TO NEXT TEST?<15><12>
026742                          64$:
026742 000444                           BR      50$             ;SKIP TO END OF TEST


026744 076600                   1$:     MED                     ;READ THE LOG CUA REG.
026746 000103                           RDLCUA
026750 042700 100007                    BIC     #100007,R0      ;MASK OFF NON-ADDRESS BITS
026754 020027 000710                    CMP     R0,#SWB1A       ;WAS CORRECT MICRO ADDR. LOGGED?
                                                                ;(MICRO ADDR. TRAPPED ON SHIFTED RIGHT)
026760 001401                           BEQ     16$             ;BRANCH IF YES
026762 104000                           HLT                     ;LOG CUA DID NOT CONTAIN MICRO
026764                          16$:                            ;ADDR. IT SHOULD HAVE CONTAINED AFTER
                                                                ;MICROBREAKING ON "SWB01"
026764 012701 002734                    MOV     #TBL5,R1        ;INTIALIZE TABLE PTR. (R1)
026770 012702 002762                    MOV     #TBL6,R2
026774 105711                   2$:     TSTB    (R1)            ;IS OPERATION CODE = 0
026776 001426                           BEQ     50$             ;BRANCH IF YES, END OF TABLE
027000 111167 000030                    MOVB    (R1),12$        ;IF NO, LOAD WRITE CODE AFTER MED
027004 011237 177770            4$:     MOV     (R2),@#UBREAK   ;LOAD MICROBK REG. WITH MICROADDR.
027010 005037 055200                    CLR     @#BKFLAG        ;CLEAR MICROBK TRAP-TO-4 FLAG
027014 076600                           MED
027016 000144                           RDFLAG
027020 052700 100000                    BIS     #BIT15,R0
027024 076600                           MED                     ;MED WRITE TO FLAG REG TO
027026 000344                   15$:    WRFLAG                  ;ENABLE MICROBK TRAPPING
027030 005000                           CLR     R0              ;CLEAR R0 IN CASE MICROBK DOESN'T OCCUR
027032 076600                           MED
027034 000000                   12$:    .WORD   0
027036 005737 055200                    TST     @#BKFLAG        ;DID WE TRAP-TO-4? (FLAG NOT = 0)
027042 001001                           BNE     20$             ;BRANCH IF YES TO NEXT ENTRY
027044 104000                           HLT                     ;MICROBK. TRAP-TO-4 DID NOT OCCUR
                                                                ;MEDCODE IS AT 12$
                                                                ;MICROADDRESS IS AT ADDRESS CONTAINED IN R2)

027046 105721                   20$:    TSTB    (R1)+
027050 005722                           TST     (R2)+
027052 000750                           BR      2$              ;BRANCH BACK TO 2$

027054 076600                   50$:    MED                     ;TEST IS OVER, DISABLE TRAPPING
027056 000022                           RDWHAMI
027060 042700 001000                    BIC     #BIT9,R0
027064 076600                           MED                     ;CLEAR THE WHAMI REG. TO
027066 000222                   14$:    WRWHAMI                 ;    DISABLE MICROBK. TRAP-TO-4
027070 012737 055064 000004             MOV     #ERPRT,@#4      ;RESTORE NORMAL ERROR ROUTINE


                                ;*      THE FOLLOWING TESTS WILL BE EXECUTED WHEN RELOCATED
```

```
                                            ;*


                                            ;;*********************************************************
                                            ;*TEST 64        PHYSICAL ADDRESS & ODD ADDRESS ERROR LOGGING
                                            ;*        THIS TEST CHECKS THAT THE PROPER PHYSICAL ADDRESS BITS
                                            ;*        <17:00> ARE LOGGED UPON ERROR.  THE ERROR IS CAUSED BY
                                            ;*        FORCING AN ODD ADDRESS TRAP.  THE ERROR LOG MODE USED
                                            ;*        IS "LOG FIRST".  ALSO, THE ODD ADDRESS ERROR BITS IN
                                            ;*        THE LOG JAM AND CPU ERROR REGISTER ARE CHECKED.
                                            ;;*********************************************************
 027076                                     TST64:
 027076  000004                                     SCOPE
 027100  112737  000064  001202                     MOVB    #64,@#$TSTNM
 027106  010700                                     MOV     PC,R0                ;SETUP NEW PC & PSW FOR THE
 027110  062700  000042                             ADD     #2$-.,R0             ;ODD ADDRESS SERVICE ROUTINE
 027114  010701                                     MOV     PC,R1                ;GET CURRENT PC
 027116  062701  000005                             ADD     #1$+1-.,R1           ;FORM ADDRESS OF 1$+1
 027122  010037  000004             1$:             MOV     R0,@#4
 027126  076600                                     MED
 027130  000022                                     RDWHAMI
 027132  052700  100001                             BIS     #BIT15+BIT0,R0       ;SETUP "LOG FIRST" MODE
 027136  076600                                     MED
 027140  000222                                     WRWHAMI
 027142  005767  177755                             TST     1$+1                 ;DO ODD ADDRESS INSTRUCTION TO FORCE
                                                                                 ;A JAMUPP & TRAP TO 4
 027146  104000                                     HLT                          ;*** ODD ADDR. TRAP DID NOT OCCUR
 027150  000454                                     BR      10$                  ;EXIT TEST
 027152  022626                     2$:             CMP     (SP)+,(SP)+          ;RESTORE STACK
 027154  012737  055064  000004                     MOV     #ERPRT,@#4           ;RESTORE OLD PC
 027162  032737  000100  177766                     BIT     #BIT6,@#CPUERR       ;WAS ODD ADDR. ERROR RECORDED BY
                                                                                 ;THE CPU ERROR REGISTER?
 027170  001001                                     BNE     3$                   ;BRANCH IF YES
 027172  104000                                     HLT                          ;*** CPU ERROR REG. DID NOT
                                                                                 ;REPORT ODD ADDRESS ERROR
                                                                                 ;READ THE LOG JAM REGISTER
 027174  076600                     3$:             MED                          ;READ LOG JAM REG.
 027176  000100                                     RDLJAM
 027200  032700  100004                             BIT     #BIT15+BIT2,R0       ;WAS ODD ADDR. ERROR LOGGED BY LOG JAM
 027204  001001                                     BNE     4$                   ;BRANCH IF YES
 027206  104000                                     HLT                          ;*** LOG JAM REG.  DID NOT LOG
                                                                                 ;ODD ADDRESS ERROR CORRECTLY
 027210  010137  001536             4$:             MOV     R1,@#VADR            ;STORE VIRTUAL ADDR. OF ODD ADDR. INSTRUCTION
 027214  004737  053724                             JSR     PC,@#CNVADR          ;CONVERT IT TO AN 18-BIT PHYSICAL ADDR.
 027220  005005                                     CLR     R5
 027222  076600                                     MED                          ;READ THE LOG PBA REGISTER
 027224  000102                                     RDLPBA
 027226  010037  002472                             MOV     R0,@#MEDTP0
 027232  063737  001550  001540                     ADD     @#FACTOR,@#PA1500    ;ADD RELOCATION FACTOR
 027240  020037  001540                             CMP     R0,@#PA1500          ;WERE BITS <15:00> OF THE PHYSICAL
                                                                                 ;BUS ADDR. LOGGED CORRECTLY?
 027244  001401                                     BEQ     5$                   ;BRANCH IF YES
 027246  005205                                     INC     R5
 027250  076600                     5$:             MED                          ;READ THE LOG SERVICE REGISTER
 027252  000101                                     RDLSERVICE
 027254  000300                                     SWAB    R0                   ;GET "PBA 17&16" DOWN TO BIT POSITION 0&1
```

```
027256  042700  177774                          BIC     #177774,R0
027262  010037  002476                          MOV     R0,@#MEDTP2
027266  020037  001542                          CMP     R0,@#PA1716     ;PBA <17:16> LOGGED CORRECTLY?
027272  001002                                  BNE     11$             ;BRANCH IF YES
027274  005705                                  TST     R5
027276  001401                                  BEQ     10$
027300                                  11$:
027300  104000                                  HLT                     ;*** PHYSICAL BUS ADDR. <17:00>
                                                                        ;NOT LOGGED CORRECTLY WHEN
                                                                        ;ODD ADDRESS TRAP OCCURRED
                                                                        ;EXPECTED PBA IS IN "PA1716" & "PA1500"
027302  012737  055064  000004          10$:    MOV     #ERRPRT,@#4     ;RESTORE OLD PC  AT 4
027310  076600                                  MED
027312  000022                                  RDWHAMI
027314  042700  100001                          BIC     #BIT15+BIT0,R0
027320  076600                                  MED                     ;DISABLE "LOG FIRST" MODE
027322  000222                                  WRWHAMI

                                        ;*
                                        ;*
                                        ;*THE FOLLOWING TESTS WILL NOT BE EXECUTED WHEN RELOCATED
                                        ;*




                                        ;;*****************************************************************
                                        ;*TEST 65        CHECK DISABLE PARITY ERROR TRAP
                                                ;*THIS TEST CHECKS THAT PARITY ERROR TRAPS TO LOCATION 114
                                                ;*ARE DISABLED WHEN BIT0 OF THE CACHE CONTROL REGISTER IS
                                                ;*SET (=1).  A TRAP TO 114 SHOULD NOT OCCUR AND ERROR
                                                ;*INFORMATION SHOULD NOT BE LOGGED IN THE LOG PBA, LOG
                                                ;*CACHE DATA, OR LOG TAG DATA REGISTERS.  WRONG PARITY IS
                                                ;*WRITTEN INTO A TEST LOCATION TO CAUSE THE PARITY ERROR
                                                ;*NEEDED IN THIS TEST.
                                        ;;*****************************************************************
027324                                  TST65:
027324  000004                                  SCOPE
027326  112737  000065  001202                  MOVB    #65,@#$TSTNM
027334  012737  000001  001324                  MOV     #1,@#$TIMES     ;DO 1 ITERATION

027342  005737  001550                          TST     @#FACTOR        ;IS RELOCATION FACTOR=0?
027346  001402                                  BEQ     14$             ;BRANCH IF YES
027350  000167  001560                          JMP     MEDEX           ;EXIT IF RELOCATED
027354  105737  001545                  14$:    TSTB    @#MMON          ;IS MEM. MGMT. ON?
027360  001402                                  BEQ     12$             ;BRANCH IF NO
027362  000167  001546                          JMP     MEDEX           ;EXIT IF RELOCATED
027366                                  12$:                            ;DO NEXT 7 TESTS - NOT RELOCATED

027366  012701  002360                          MOV     #TLOC1,R1       ;GET POINTER TO TEST LOCATION
027372  005711                                  TST     (R1)            ;MAKE IT A HIT
027374  012737  000100  177746                  MOV     #WWP,@#CCR      ;SET WRITE WRONG PARITY BIT
027402  012711  125252                          MOV     #125252,(R1)    ;WRITE TO TEST LOC. WITH WRONG PARITY
027406  012737  000001  177746                  MOV     #DPTRP,@#CCR    ;DISABLE PARITY ERROR TRAPS
                                                                        ;AND CLEAR WWP
```

```
027414  012737  027446  000114           MOV     #1$,@#114          ;SETUP PARITY ERROR VECTOR
027422  005000                           CLR     R0
027424  076600                           MED                        ;CLEAR LOG PBA REGISTER
027426  000302                           WRLPBA
027430  076600                           MED                        ;CLEAR LOG CACHE DATA REGISTER
027432  000306                           WRLDATA
027434  076600                           MED                        ;CLEAR LOG CACHE TAG REGISTER
027436  000307                           WRLTAG
027440  005767  152714                   TST     TLOC1              ;READ TEST LOC1 TO FORCE PARITY ERROR
027444  000406                           BR      2$                 ;BRANCH IF NO TRAP OCCURS
027446  012700  000200           1$:     MOV     #200,R0
027452  076600                           MED                        ;CLEAN UP THE CACHE
027454  000352                           352                        ;INITIALIZATION CODE
027456  022626                           CMP     (SP)+,(SP)+        ;CLEAN UP STACK
027460  104000                           HLT                        ;*** PARITY TRAP TO 114 OCCURRED
                                                                    ;WHEN IT SHOULD HAVE BEEN DISABLED
027462  012700  000200           2$:     MOV     #200,R0
027466  076600                           MED
027470  000352                           352                        ;CLEAN UP CACHE WITH INITIALIZATION CODE
027472  012711  125252                   MOV     #125252,(R1)       ;WRITE BACK GOOD PARITY IN TST LOC
027476  012737  054414  000114           MOV     #.PARSRV,@#114     ;RESTORE ORIGINAL PARITY HANDLER
027504  005005                           CLR     R5                 ;CLEAR ERROR FLAG
027506  076600                           MED                        ;READ LOG PBA REGISTER
027510  000102                           RDLPBA
027512  010067  152754                   MOV     R0,MEDTP0          ;SAVE COPY
                                                                    ;LOG PBA REG. STILL CLEAR?
027516  001401                           BEQ     3$                 ;BRANCH IF YES
027520  005205                           INC     R5                 ;OTHERWISE SET ERROR FLAG
027522  076600                   3$:     MED                        ;READ LOG CACHE DATA REG.
027524  000106                           RDLDATA
027526  010067  152742                   MOV     R0,MEDTP1          ;SAVE COPY
                                                                    ;LOG CACHE DATA REG. STILL CLEAR?
027532  001401                           BEQ     4$                 ;BRANCH IF YES
027534  005205                           INC     R5                 ;OTHERWISE SET ERROR FLAG
027536  076600                   4$:     MED                        ;READ LOG CACHE TAG REG.
027540  000107                           RDLTAG
027542  010067  152730                   MOV     R0,MEDTP2          ;SAVE COPY
                                                                    ;LOG CACHE TAG REG. STILL CLEAR?
027546  001401                           BEQ     5$                 ;BRANCH IF YES
027550  005205                           INC     R5                 ;OTHERWISE SET ERROR FLAG
027552  005705                   5$:     TST     R5                 ;WERE ANY OF LOG REGISTERS CHANGED
027554  001401                           BEQ     6$                 ;BRANCH IF NO
027556  104000                           HLT                        ;*** ONE OF LOG REGISTERS CHANGED
                                                                    ;WHEN ERROR SHOULD NOT HAVE BEEN LOGGED
                                                                    ;LOG PBA, LOG DATA & LOG TAG
                                                                    ;REGISTER SHOULD BE CLEAR.
027560  005037  177746           6$:     CLR     @#CCR              ;ENABLE PARITY ERROR TRAPS

                                 ;;*************************************************************
                                 ;*TEST 66       CHECK PARITY ERROR BITS IN MEMERR REG. IN BACKUP MODE OF CACHE (TRAP)

                                         ;*THIS TEST CHECKS THAT ALL OF THE PARITY ERROR BITS (5,6,7)
                                         ;*OF THE MEMORY ERROR REGISTER ARE SET TO "1" WHEN A CACHE
                                         ;*PARITY ERROR OCCURS IN THE BACKUP MODE.
                                 ;;*************************************************************
027564                           TST66:
```

```
027564  000004                                      SCOPE
027566  112737  000066  001202                      MOVB    #66,@#$TSTNM
027574  012701  002360                              MOV     #TLOC1,R1               ;GET POINTER TO TEST LOCATION
027600  005711                                      TST     (R1)                    ;MAKE IT A HIT
027602  012737  000100  177746                      MOV     #WWP,@#CCR              ;SET WRITE WRONG PARITY BIT
027610  012711  125252                              MOV     #125252,(R1)            ;WRITE TO TEST LOC. WITH WRONG PARITY
027614  042737  000100  177746                      BIC     #WWP,@#CCR              ;CLEAR WWP
027622  012737  027650  000114                      MOV     #1$,@#114               ;SETUP NEW TEST HANDLER AT PARITY VECTOR
027630  005767  152524                              TST     TLOC1                   ;READ TEST LOC. TO FORCE PARITY ERROR
027634  012700  000200                              MOV     #200,R0
027640  076600                                      MED                             ;CLEAN UP THE CACHE
027642  000352                                      352                             ;INITIALIZATION CODE
027644  104000                                      HLT                             ;*** PARITY ERROR DID NOT CAUSE TRAP
027646  000405                                      BR      2$                      ;BRANCH TO 2$
027650  012700  000200                      1$:     MOV     #200,R0
027654  076600                                      MED                             ;CLEAN UP CACHE WITH INITIALIZATION CODE
027656  000352                                      352
027660  022626                                      CMP     (SP)+,(SP)+             ;CLEAN UP STACK
027662  022737  000340  177744              2$:     CMP     #000340,@#MEMERR        ;WERE PARITY ERROR BITS (5,6,7) SET
                                                                                    ;AND CPU ABORT BIT (15) LEFT CLEAR
                                                                                    ;IN MEMORY ERROR REGISTER?
027670  001401                                      BEQ     3$                      ;BRANCH IF YES
027672  104000                                      HLT                             ;*** MEMORY ERROR REGISTER BITS
                                                                                    ;WERE SET INCORRECTLY
                                                                                    ;PARITY ERROR BITS SHOULD BE SET
027674  012737  054414  000114              3$:     MOV     #.PARSRV,@#114          ;RESTORE OLD PARITY HANDLER PC

                                            ;;*****************************************************************
                                            ;*TEST 67       CHECK UNIBUS TIMEOUT, ODD ADDRESS AND LOG CONTINUOUS MODE

                                                    ;*THIS TEST CHECKS THAT THE "UNIBUS TIMEOUT" BIT (BIT4)
                                                    ;*GETS SET IN THE CPU ERROR REGISTER WHEN A TIMEOUT OCCURS.
                                                    ;*A TIMEOUT TRAP IS FORCED BY REFERENCING BUS ADDRESS 760000.
                                                    ;*THEN AN ODD ADDRESS ERROR IS FORCED AND IT
                                                    ;*IS CHECKED IF ONLY BIT (6)-ODD ADDRESS ERROR IS SET
                                                    ;*(IN CPUERR).  THIS CHECKS THAT THE ERROR LOG IS
                                                    ;*CONTINUOSLY UPDATED IN THE "LOG CONTINUOUS" MODE.
                                            ;;*****************************************************************
027702                                      TST67:
027702  000004                                      SCOPE
027704  112737  000067  001202                      MOVB    #67,@#$TSTNM
027712  012737  027726  000004                      MOV     #1$,@#4                 ;SETUP NEW PC
027720  005737  160000                              TST     @#160000                ;FORCE A TIMEOUT TRAP TO 4 BY
                                                                                    ;REFERENCING NON-EXISTENT ADDRESS
027724  000452                                      BR      6$
027726  022626                              1$:     CMP     (SP)+,(SP)+             ;RESTORE STACK
027730  012737  055064  000004                      MOV     #ERPRT,@#4              ;RESTORE OLD PC
027736  022737  000020  177766                      CMP     #BIT4,@#CPUERR          ;DID "UNIBUS TIMEOUT" BIT IN CPU ERROR
                                                                                    ;REGISTER GET SET?
027744  001401                                      BEQ     2$                      ;BRANCH IF YES
027746  104000                                      HLT                             ;*** "UNIBUS TIMEOUT" BIT (BIT4) IN CPU
                                                                                    ;ERROR REG. DID NOT SET WHEN A
                                                                                    ;TIMEOUT WAS FORCED
027750  076600                              2$:     MED                             ;READ THE LOG JAM REG.
027752  000100                                      RDLJAM
027754  022700  021200                              CMP     #BIT13+BIT9+BIT7,R0     ;DID "UNIBUS TIMEOUT" BIT (BIT7) SET?
```

```
6890                                                ;BIT 9= POWER STATUS, ALWAYS SET
6891  027760  001401          BEQ     3$            ;BRANCH IF YES
6892  027762  104000          HLT                   ;*** "UNIBUS TIMEOUT" BIT (BIT7)
6893                                                ;DID NOT SET IN LOG JAM REGISTER
```

```
                                                                  ;WHEN UNIBUS TIMEOUT WAS FORCED
 027764  076600                           3$:     MED             ;READ LOG PBA
 027766  000102                                   RDLPBA
 027770  020027  160000                           CMP     R0,#160000      ;WAS PHYS BA LOGGED CORRECTLY?
 027774  001401                                   BEQ     5$
 027776  104000                                   HLT                     ;PHYSICAL BUS ADDRESS WAS
                                                                          ;LOGGED WRONG ON A UNIBUS
                                                                          ;TIMEOUT
                                                                          ;LOG PBA SHOULD HOLD ADDR. 160000
 030000  012737  030014  000004           5$:     MOV     #4$,@#4         ;SET UP PC FOR ODD ADDRESS
 030006  005767  177753                           TST     3$+1            ;FORCE ODD ADDRESS ERROR
 030012  000417                                   BR      6$
 030014  022626                           4$:     CMP     (SP)+,(SP)+     ;RESTORE STACK
 030016  012737  055064  000004                   MOV     #ERPRT,@#4
 030024  022737  000100  177766                   CMP     #BIT6,@#CPUERR  ;ODD ADDR. BUT SET 3
 030032  001401                                   BEQ     7$
 030034  104000                                   HLT                     ;ODD ADDRESS BIT WAS
                                                                          ;NOT SET IN THE CPU
                                                                          ;ERROR REGISTER.  IN LOG
                                                                          ;CONTINUOUS MADE THE
                                                                          ;LATEST ERROR SHOULD
                                                                          ;BE LOGGED
 030036  076600                           7$:     MED                     ;READ LOG JAM REG.
 030040  000100                                   RDLJAM
 030042  032700  000004                           BIT     #BIT2,R0        ;ODD ADR. BIT SET IN
 030046  001001                                   BNE     6$              ;LOG JAM?
 030050  104000                                   HLT                     ;ODD ADDRESS BIT WAS
                                                                          ;NOT SET IN THE LOG
                                                                          ;JAM REGISTER ON A
                                                                          ;ODD ADDRESS ERROR
 030052  076600                           6$:     MED                     ;CHECK IF LAST INTERRUPT VECTOR
 030054  000104                                   RDLFGINT                ;WAS LOGGED?
 030056  120027  000004                           CMPB    R0,#4
 030062  001401                                   BEQ     8$
 030064  104000                                   HLT                     ;LAST ERROR VECTOR WAS NOT LOGGED
                                                                          ;LOW BYTE OF LOG FLAG/INT REG.
                                                                          ;SHOULD CONTAIN LAST VEC. =004
 030066  012737  055064  000004           8$:     MOV     #ERPRT,@#4



                                          ;;*********************************************************************
                                          ;*TEST 70        CHECK ILLEGAL INTERNAL ADDRESS TRAP

                                                  ;*THIS TEST CHECKS THAT A TRAP OCCURS UPON REFERENCING AN
                                                  ;*ILLEGAL INTERNAL ADDRESS AND THAT "ILLEGAL INTERNAL ADDRESS"
                                                  ;*BIT (BIT0) OF THE CPU ERROR REGISTER AND BITS OF LOG JAM
                                                  ;*REGISTER GET SET.  IT ALSO CHECKS IF THE INTERRUPT VECTOR
                                                  ;*(4) IS SAVED AS THE "LAST INTERRUPT VECTOR" IN THE LOG
                                                  ;*FLAG/INTERRUPT REG.
                                          ;;*********************************************************************
 030074                                   TST70:
 030074  000004                                   SCOPE
 030076  112737  000070  001202                   MOVB    #70,@#$TSTNM
 030104  012737  030126  000004                   MOV     #1$,@#4         ;SETUP NEW HANDLER PC
 030112  005037  177746                           CLR     @#CCR
```

```
030116  012707  177746                  MOV     #CCR,PC         ;ILLEGAL INTERNAL ADDRESS TRAP SHOULD OCCUR
030122  104000                          HLT                     ;*** ILLEGAL INTERNAL ADDRESS
                                                                ;DID NOT RESULT IN A TRAP
030124  000417                          BR      3$              ;BRANCH TO EXIT IF NO TRAP
030126  022626                  1$:     CMP     (SP)+,(SP)+     ;RESTORE STACK
030130  012737  055064  000004          MOV     #ERPRT,@#4      ;RESTORE OLD HANDLER PC
030136  032737  000001  177766          BIT     #BIT0,@#CPUERR  ;DID "ILLEGAL INTERNAL ADDRESS" BIT (0)
                                                                ;IN CPU ERROR REGISTER GET SET?
030144  001001                          BNE     2$              ;BRANCH IF YES
030146  104000                          HLT                     ;*** ILLEGAL INTERNAL ADDRESS
                                                                ;BIT DID NOT SET IN CPU ERROR REG.
030150  076600                  2$:     MED                     ;READ LOG JAM REG.
030152  000100                          RDLJAM
030154  032700  000040                  BIT     #BIT5,R0        ;DID "ILLEGAL INTERNAL ADDRESS" BIT (5)
                                                                ;IN LOG JAM REG. GET SET
030160  001001                          BNE     3$              ;BRANCH IF YES
030162  104000                          HLT                     ;*** ILLEGAL INTERNAL ADDRESS BIT
                                                                ;DID NOT SET IN LOG JAM REG.
030164                          3$:



                                ;;**************************************************************
                                ;*TEST 71        CHECK LOG SERVICE & MEMERR LOGS LO-HI BYTE & TAG BITS, IN ABORT MODE
                                        ;THIS TEST CHECKS THAT "LO BYTE PARITY" AND "TAG PARITY"
                                        ;BITS CAN SET IN "LOG SERVICE" REGISTERS.  IT IS ALSO
                                        ;CHECKED THAT THE PROPER TAG AND DATA BITS GET STORED
                                        ;IN THE "LOG CACHE DATA, "LOG CACHE TAG/CPU" AND THE
                                        ;"MEMORY ADDRESS REGISTER" WHEN A PARITY ERROR IS
                                        ;FORCED.
                                        ;IT IS CHECKED IF THE INSTRUCTION WAS ABORTED AND THE
                                        ;LOG FLAG/INTERRUPT REGISTER LOGGED THE LAST INTERRUPT
                                        ;VECTOR.
                                ;;**************************************************************
030164                          TST71:
030164  000004                          SCOPE
030166  112737  000071  001202          MOVB    #71,@#$TSTNM

030174  012737  000201  177746          MOV     #DPTRP+PABORT,@#CCR  ;DISABLE PARITY TRAPS (CACHE)
030202  005067  152264                  CLR     MEDTPO
030206  012701  002360                  MOV     #TLOC1,R1       ;GET POINTER TO TEST LOC.
030212  012711  111000                  MOV     #111000,(R1)
030216  005711                          TST     (R1)            ;MAKE IT A HIT
030220  052737  000100  177746          BIS     #WWP,@#CCR      ;WRITE WRONG PARITY SET
030226  012711  000252                  MOV     #252,(R1)       ;WRITE TEST LOCATION
                                                                ;WITH WRONG PARITY
030232  042737  000100  177746          BIC     #WWP,@#CCR      ;CLEAR WWP
030240  076600                          MED
030242  000022                          RDWHAMI
030244  052700  100001                  BIS     #BIT15+BIT0,R0  ;ENABLE "LOG FIRST" MODE AND
030250  076600                          MED                     ;ERROR LOGGING
030252  000222                          WRWHAMI
030254  042737  000001  177746          BIC     #DPTRP,@#CCR    ;ENABLE CACHE PARITY TRAPS
030262  012737  030310  000114          MOV     #PTRP1,@#114    ;NEW PARITY TRAP SERVICE
030270  016767  152064  152174          MOV     TLOC1,MEDTPO    ;READ TEST LOC, FORCE PARITY ERROR
030276  012700  000200                  MOV     #200,R0
030302  076600                          MED                     ;CLEAN UP THE CACHE
```

```
030304  000352                              352                                 ;INITIALIZATION CODE
030306  104000                              HLT                                 ;*** CACHE PARITY ERROR TRAP
                                                                                ;DID NOT OCCUR WHEN
                                                                                ;TEST LOC WITH BAD PARITY
                                                                                ;WAS READ

                                                                                ;ENTER HERE IF PARITY TRAP OCCURRED
030310  012700  000200              PTRP1:  MOV     #200,R0
030314  076600                              MED                                 ;CLEAN UP THE CACHE
030316  000352                              352                                 ;INITIALIZATION CODE
030320  012737  000001  177746              MOV     #DPTRP,@#CCR                ;DISABLE CACHE PARITY ERROR TRAPS
030326  012737  054414  000114              MOV     #.PARSRV,@#114              ;REESTABLISH OLD SERVICE VECTORS
030334  022626                              CMP     (SP)+,(SP)+
030336  005767  152130                      TST     MEDTPO                      ;WAS THE INSTRUCTION ABORTED ON
                                                                                ;CACHE PARITY ERROR (ABORT MODE)?
030342  001401                              BEQ     1$                          ;YES
030344  104000                              HLT                                 ;INSTRUCTION HAVING CACHE PARITY
                                                                                ;ERROR WAS NOT ABORTED, IN THE
                                                                                ;CACHE ABORT MODE.
030346  076600                      1$:     MED                                 ;READ THE "LOG SERVICE" REGISTER
030350  000101                              RDLSERVICE
030352  042700  177435                      BIC     #177435,R0                  ;MASK ALL BITS BUT LO,HI,TAG,CACHE BITS
030356  022700  000342                      CMP     #342,R0                     ;LO, HI,TAG, CACHE PARITY BITS SET? IN "SERVICE"
030362  001401                              BEQ     2$                          ;YES
030364  104000                              HLT                                 ;*** "LO BYTE" PARITY ERROR
                                                                                ;AND "TAG" PARITY ERROR BITS
                                                                                ;WERE NOT LOGGED CORRECTLY IN "LOG
                                                                                ;SERVICE" REGISTER, WHEN PARITY
                                                                                ;ERROR TRAP WAS FORCED.
                                                                                ;CLEAR BITS ARE ACTIVE.
030366  022737  100340  177744      2$:     CMP     #BIT7+BIT6+BIT5+BIT15,@#MEMERR  ;DID"LO BYTE" AND "TAG"
                                                                                ;PARITY ERROR BITS SET IN
                                                                                ;THE MEMORY ERROR REGISTER?
030374  001401                              BEQ     3$                          ;YES
030376  104000                              HLT                                 ;*** "LO BYTE" AND "TAG" PARITY
                                                                                ;ERROR BITS DID NOT SET
                                                                                ;CORRECTLY IN THE MEMORY
                                                                                ;ERROR REGISTER
030400  076600                      3$:     MED                                 ;READ "LOG PBA" REGISTER
030402  000102                              RDLPBA
030404  020027  002360                      CMP     R0,#TLOC1                   ;DID "LOG PBA" CONTAIN CORRECT
                                                                                ;PHYSICAL BUS ADDRESS-WHERE
                                                                                ;THE PARITY ERROR OCCURRED?
030410  001401                              BEQ     4$                          ;YES
030412  104000                              HLT                                 ;*** PHYSICAL BUS ADDRESS
                                                                                ;(WHERE PARITY ERROR OCCURRED)
                                                                                ;WAS NOT LOGGED CORRECTLY
                                                                                ;WHEN CACHE PARITY ERROR WAS FORCED
                                                                                ;LOG PBA SHOULD CONTAIN ADDR. OF "TLOC1"
030414  076600                      4$:     MED
030416  000107                              RDLTAG                              ;READ "LOG CACHE TAG" REGISTER
030420  010002                              MOV     R0,R2
030422  000302                              SWAB    R2
030424  012701  002360                      MOV     #TLOC1,R1                   ;SHIFT RIGHT (3 TIMES) THE 16 BIT
030430  000301                              SWAB    R1
030432  106201                              ASRB    R1                          ;PHYSICAL BUS ADDRESS OF THE
```

```
030434  106201                          ASRB    R1                      ;TEST LOCATION
030436  106201                          ASRB    R1
030440  052701  000200                  BIS     #BIT7,R1                ;FUDGE TAG BIT
030444  120102                          CMPB    R1,R2                   ;WAS THE CORRECT TAG LOGGED?
030446  001405                          BEQ     5$                      ;YES
030450  010167  152016                  MOV     R1,MEDTP0               ;EXPECTED TAG
030454  010267  152014                  MOV     R2,MEDTP1               ;TAG RECEIVED
030460  104000                          HLT                             ;TAG BITS WERE NOT LOGGED
                                                                        ;CORRECTLY, WHEN CACHE
                                                                        ;PARITY ERROR WAS FORCED
                                                                        ;EXPECTED TAG IS IN MEDTP0
                                                                        ;TAG RECEIVED IS IN MEDTP1
030462  076600                  5$:     MED                             ;READ CACHE DATA
030464  000106                          RDLDATA
030466  020027  000252                  CMP     R0,#252                 ;CACHE DATA LOGGED CORRECTLY?
030472  001401                          BEQ     6$
030474  104000                          HLT                             ;CACHE DATA WAS NOT LOGGED
                                                                        ;CORRECTLY, LOG DATA REG.
                                                                        ;SHOULD CONTAIN DATA = 252

030476  076600                  6$:     MED                             ;RETURN TO "LOG CONTINUOUS" MODE
030500  000022                          RDWHAMI
030502  042700  100001                  BIC     #BIT15+BIT0,R0
030506  076600                          MED
030510  000222                          WRWHAMI
030512  012737  030524  000004          MOV     #7$,@#4
030520  005737  160000                  TST     @#160000                ;INTERRUPT THRU 4 SHOULD LOAD "FLAG"
030524  022626                  7$:     CMP     (SP)+,(SP)+
030526  012737  055064  000004          MOV     #ERPRT,@#4              ;RESTORE LOC. 4
030534  076600                          MED                             ;READ LOG FLAG/INTERRUPT REGISTER
030536  000104                          RDLFGINT
030540  120027  000114                  CMPB    R0,#114                 ;DID LO BYTE CONTAIN VECTOR 114?
030544  001401                          BEQ     8$
030546  104000                          HLT                             ;LAST INTERRUPT VECTOR WAS NOT
                                                                        ;LOGGED CORRECTLY IN FLAG REGISTER
                                                                        ;WHEN A CACHE PARITY ERROR WAS
                                                                        ;FORCED.
                                                                        ;LOW BYTE OF LOG FLAG/INT REG.
                                                                        ;SHOULD BE LAST VEC. = 114
                                                                        ;ANY ERROR CALL DURING TEST MIGHT CAUSE
                                                                        ;LAST VECTOR TO LOOK WRONG
030550                          8$:


                                ;;*****************************************************************
                                ;*TEST 72        CHECK "LOG FIRST" MODE OF ERROR LOGGING
                                        ;*THIS TEST CHECKS THE "LOG FIRST" MODE OF ERROR LOGGING.
                                        ;*THE "LOG FIRST" MODE IS ENABLED.  THEN A TIME-OUT TRAP
                                        ;*IS FORCED, BIT 4 OF CPU ERROR REGISTER SHOULD BE SET.
                                        ;*THEN AN ODD ADDRESS TRAP IS FORCED.  HOWEVER, THIS
                                        ;*TIME THE ERROR SHOULD NOT BE LOGGED; BIT 6 (ODD
                                        ;*ADDRESS) SHOULD NOT BE SET BECAUSE THE ERROR LOG
                                        ;*IS LOCKED UP AFTER THE FIRST ERROR.

                                        ;*THEN, THE ERROR LOG IS ENABLED (BY SETTING BIT 0 OF
                                        ;*WHAMI).  AN ODD ADDRESS ERROR IS FORCED AGAIN AND IT IS
```

```
                                                 ;*CHECKED THAT THIS TIME THE ERROR IS LOGGED. (BIT 6-ODD
                                                 ;*ADDRESS SHOULD BE SET IN CPU ERROR REGISTER).
                                         ;;*************************************************************
030550                                   TST72:
030550  000004                                   SCOPE
030552  112737  000072  001202                   MOVB    #72,@#$TSTNM

030560  076600                                   MED
030562  000022                                   RDWHAMI
030564  052700  100001                           BIS     #BIT15+BIT0,R0    ;SET UP "LOG FIRST MODE
030570  076600                                   MED
030572  000222                                   WRWHAMI
030574  012737  030610  000004                   MOV     #1$,@#4           ;SET UP NEW PC FOR TIMEOUT
030602  005737  160000                           TST     @#160000          ;FORCE A TIMEOUT
030606  000454                                   BR      5$                ;SKIP TEST IF NO TIMEOUT

030610  022626                           1$:     CMP     (SP)+,(SP)+       ;RESTORE STACK
                                                                           ;BIT 4 OF CPU ERROR REGISTER
                                                                           ;SHOULD HAVE SET
030612  012737  030626  000004                   MOV     #2$,@#4           ;SET UP NEW PC FOR ODD ADDRESS
030620  005767  177765                           TST     1$+1              ;FORCE ODD ADDRESS TRAP
030624  000445                                   BR      5$                ;SKIP TEST IF NO ODD ADDRESS TRAP

030626  022626                           2$:     CMP     (SP)+,(SP)+       ;RESTORE STACK
030630  012737  055064  000004                   MOV     #ERPRT,@#4
030636  022737  000020  177766                   CMP     #BIT4,@#CPUERR    ;"TIMEOUT" BIT SHOULD BE STILL
                                                                           ;SET, CHECK?
030644  001402                                   BEQ     3$
030646  104000                                   HLT                       ;*** SECOND ERROR (ODD ADDRESS)
                                                                           ;UPDATED THE ERROR LOG IN
                                                                           ;THE LOG FIRST MODE.  BIT 4
                                                                           ;(UNIBUS TIMEOUT) SHOULD BE
                                                                           ;STILL SET FROM THE FIRST
                                                                           ;ERROR
030650  000433                                   BR      5$                ;SKIP THE REST
030652  076600                           3$:     MED                       ;READ LOG JAM REG.
030654  000100                                   RDLJAM
030656  032700  100004                           BIT     #BIT2+BIT15,R0    ;CHECK THAT ODD ADRES ERROR BITS NOT
030662  001401                                   BEQ     6$                ;SET IN LOG JAM.  NOTE LOG FIRST
                                                                           ;MODE SHOULD INHIBIT FURTHER
                                                                           ;ERROR LOGGING
030664  104000                                   HLT                       ;ODD ADDRESS ERROR BITS GOT SET IN LOG JAM
                                                                           ;THEY SHOULD NOT BE SINCE LOG FIRST MODE
                                                                           ;INHIBITS ERROR LOGGING AFTER THE FIRST ERROR
030666  076600                           6$:     MED
030670  000022                                   RDWHAMI
030672  052700  100001                           BIS     #BIT15+BIT0,R0    ;ENABLE ERROR LOG AGAIN IN
                                                                           ;LOG FIRST MODE
030676  076600                                   MED
030700  000222                                   WRWHAMI
030702  012737  030716  000004                   MOV     #4$,@#4           ;SET UP NEW PC
030710  005767  177737                           TST     3$+1              ;FORCE ODD ADDRESS TRAP
030714  000411                                   BR      5$                ;SKIP IF NO TRAP
030716  022626                           4$:     CMP     (SP)+,(SP)+       ;RESTORE STACK
                                                                           ;RESTORE OLD PC(4)
030720  012737  055064  000004                   MOV     #ERPRT,@#4
```

```
030726  022737  000100  177766          CMP     #BIT6,@#CPUERR  ;THE ERROR LOG FROM PREVIOUS
                                                                ;ERROR SHOULD BE OVER WRITTEN.
                                                                ;ODD ADDRESS BIT SHOULD
                                                                ;BE SET, BECAUSE THE ERROR
030734  001401                          BEQ     5$              ;LOG WAS ENABLED.
                                                                ;OK, IF YES
030736  104000                          HLT                     ;THE ERROR LOG WAS NOT UPDATED
                                                                ;(UPON AN ODD ADDRESS ERROR)
                                                                ;AFTER THE LOG WAS ENABLED.
                                                                ;AT THIS FORMAT BIT 6 OF
                                                                ;CPU ERROR REGISTER SHOULD
                                                                ;BE SET.  IT WAS NOT.
030740  012737  055064  000004  5$:     MOV     #ERPRT,@#4      ;RESTORE OLD PC(4)
030746  076600                          MED
030750  000022                          RDWHAMI
030752  042700  100001                  BIC     #BIT15+BIT0,R0
030756  076600                          MED                     ;PUT THE LOGGING BACK INTO
030760  000222                          WRWHAMI                 ;"CONTINUOUS" MODE


                                ;;*************************************************************
                                ;*TEST 73       CHECK LAST INTERRUPT VECTOR IS LOGGED IN FLAG REG.
                                ;;*************************************************************
030762                          TST73:
030762  000004                          SCOPE
030764  112737  000073  001202          MOVB    #73,@#$TSTNM
030772  012737  000001  001324          MOV     #1,@#$TIMES     ;DO 1 ITERATION

031000  012737  031010  000030          MOV     #1$,@#30        ;LOAD LOC. 30 WITH 1$ (EMT VEC.)
031006  104000                          EMT                     ;FIRST INTERRUPT-EMT
031010  022626                  1$:     CMP     (SP)+,(SP)+     ;CLEAN UP STACK
031012  012737  047160  000030          MOV     #$ERROR,@#30    ;RESTORE LOC. 30
031020  012737  031032  000004          MOV     #2$,@#4         ;LOAD LOC. 4 WITH 2$
031026  005737  160000                  TST     @#160000        ;SECOND INTERRUPT-SHOULD LOAD LOG FLAG REG.
031032  022626                  2$:     CMP     (SP)+,(SP)+     ;CLEAN UP STACK
031034  012737  055064  000004          MOV     #ERPRT,@#4      ;RESTORE LOC.  4
031042  076600                          MED                     ;READ LOG FLAG/INT REG.
031044  000104                          RDLFGINT
031046  120027  000030                  CMPB    R0,#30          ;WAS 30 LOGGED AS LAST INTRPT. VEC?
031052  001401                          BEQ     3$              ;BRANCH IF YES
031054  104000                          HLT                     ;LOG FLAG/INT REG DID NOT LOG VECTOR
                                                                ;LOW BYTE OF LOG FLAG/INT REG.
                                                                ;SHOULD BE LAST VEC. = 30
031056  012737  031066  000020  3$:     MOV     #6$,@#20        ;LOAD LOC. 20  WITH 6$
031064  000004                          IOT                     ;FIRST INTERRUP -  IOT
031066  022626                  6$:     CMP     (SP)+,(SP)+     ;CLEAN UP STACK
031070  012737  046720  000020          MOV     #$SCOPE,@#20    ;RESTORE CONTENTS OF LOC 20
031076  012737  031110  000004          MOV     #4$,@#4         ;LOAD LOC.  4 WITH 4$
031104  005737  160000                  TST     @#160000        ;INTERRUPT SHOULD LOAD LOG FLAG REG.
031110  022626                  4$:     CMP     (SP)+,(SP)+     ;CLEAN UP  STACK
031112  012737  055064  000004          MOV     #ERPRT,@#4      ;RESTORE LOC. 4
031120  076600                          MED                     ;READ LOG FLAG/INT REG.
031122  000104                          RDLFGINT
031124  122700  000020                  CMPB    #20,R0          ;WAS 20 LOGGED AS LAST INTRPT VEC?
031130  001401                          BEQ     5$              ;BRANCH IF YES
031132  104000                          HLT                     ;LOG FLAG/INT REG. DID NOT LOG VECTOR
```

```
                                                                        ;LOW BYTE OF LOG FLAG/INT REG.
                                                                        ;SHOULD BE LAST VEC. = 20
 031134                                 5$:                             ;ANY ERROR CALL DURING THE TEST MIGHT
                                                                        ;CAUSE LAST VECTOR TO BE WRONG





 031134  013746  002362                 MEDEX:  MOV     @#PSWHOL,-(SP)  ;PUSH OLD PSW ON STACK
 031140  011600                                 MOV     (SP),R0         ;RESET "VECTOR + 2'S"
 031142  042700  000020                         BIC     #BIT4,R0
 031146  052700  000340                         BIS     #PR7,R0
 031152  010037  000006                         MOV     R0,@#6
 031156  010037  000012                         MOV     R0,@#12
 031162  010037  000032                         MOV     R0,@#32
 031166  010037  000116                         MOV     R0,@#116
 031172  005037  000022                         CLR     @#22
 031176  010746                                 MOV     PC,-(SP)        ;PUSH CURRENT PC ON STACK
 031200  062716  000006                         ADD     #6,(SP)         ;ADD OFFSET TO PC
 031204  000002                                 RTI                     ;RESTORE ORIGINAL PSW AND CONTINUE
 031206  012706  000700                         MOV     #USESTK,SP      ;SET STACK POINTER
                                        ;;*****************************************************************
                                        ;*TEST 74       CHECK MFPI/MTPI INSTRUCTIONS
                                        ;;*****************************************************************
 031212                                 TST74:
 031212  000004                                 SCOPE
 031214  112737  000074  001202                 MOVB    #74,@#$TSTNM
 031222  032737  140000  177776         MPI:    BIT     #UM,@#PSW       ;KERNEL MODE?
 031230  001535                                 BEQ     ENDCP           ;IF YES, EXIT TEST
 031232  010746                                 MOV     PC,-(SP)
 031234  062716  000120                         ADD     #5$-.,(SP)
 031240  012637  000250                         MOV     (SP)+,@#MMVEC   ;SET MEM MGMT ABORT VECTOR
 031244  005046                                 CLR     -(SP)           ;CLEAR CHECK WORD
 031246  010603                                 MOV     SP,R3
 031250  010346                                 MOV     R3,-(SP)        ;PUT ADDRESS OF CHECK WORD ON THE STACK
 031252  105737  001545                         TSTB    @#MMON          ;CHECK IF MEM MGMT IS ENABLED
 031256  001411                                 BEQ     1$              ;BRANCH IF OFF
 031260  013737  177640  177654                 MOV     @#UIPAR0,@#UIPAR6       ;SET UP USER PAGE ADDR. REG.
 031266  012737  006006  177614                 MOV     #6006,@#UIPDR6          ;SET USER PAGE DESC REG R/W UP 6 PAGES
 031274  062706  140000                 10$:    ADD     #140000,SP      ,SET CURRENT MODE'S STACK POINTER
 031300  000240                                 NOP
 031302  010746                         1$:     MOV     PC,-(SP)
 031304  062716  000024                         ADD     #3$-.,(SP)
 031310  012637  000020                         MOV     (SP)+,@#IOTVEC  ;SET IOT TRAP VECTOR
 031314  00[illegible]04                                 IOT                     ;TRAP TO 3$ BELOW
 031316  00[illegible]66  000002                         INC     2(SP)           ;INCREMENT CHECK WORD
 031322  001417                                 BEQ     6$
 031324  104000                         4$:     HLT                     ;ERROR! MFPI,MTPI FAILURE-FOR BETTER
 031326  000415                                 BR      6$              ;ISOLATION SUGGEST RUNNING MEMORY MGMT. DIAG.
 031330  000240                         3$:     NOP                     ;PSW=KERNEL MODE,PREV USER MODE
 031332  006506                                 MFPI    SP              ;GET PREV. MODES STACK POINTER
 031334  006536                                 MFPI    @(SP)+          ;GET DATA (AN ADDRESS) ON PREV MODE'S STACK
 031336  006576  000000                         MFPI    @(SP)           ;GET DATA (=0) FROM PREV MODES ADDRESS
 031342  000240                                 NOP                     ;SPACE AND PUSH ONTO KERNEL STACK
 031344  001367                                 BNE     4$              ;ERROR IF BRANCH TAKEN! SHOULD HAVE A ZERO ON THE STACK
```

```
031346  005116                                  COM     (SP)                    ;COMPLEMENT OPERAND
031350  006636                                  MTPI    @(SP)+                  ;POP OPERAND OFF KERNEL STACK AND MOVE
                                                                                ;IT TO PREV MODE'S SPACE
031352  000002                                  RTI                             ;RETURN TO INST FOLLOWING IOT ABOVE
031354  104000                          5$:     HLT                             ;ERROR! MEMORY MANG. ABORT
031356  105037  177776                          CLRB    @#PSW                   ;SET PRIORITY LEVEL BACK TO 0
031362  012737  054662  000250          6$:     MOV     #KTABRT,@#MMVEC         ;RESTORE VECTOR
031370  012737  046720  000020                  MOV     #$SCOPE,@#IOTVEC
031376  012706  000700                          MOV     #USESTK,SP              ;RESTORE STACK POINTER
                                        ;;*****************************************************************
                                        ;*TEST 75       CHECK ILLEGAL HALT
                                        ;;*****************************************************************
031402                                  TST75:
031402  000004                                  SCOPE
031404  112737  000075  001202                  MOVB    #75,@#$TSTNM
031412  010746                          HALT1:  MOV     PC,-(SP)                ;GET CURRENT PC
031414  062716  000022                          ADD     #2$-.,(SP)
031420  011637  000004                          MOV     (SP),@#ERRVEC           ;SET ERROR TRAP VECTOR TO 2$ BELOW
031424  012637  000010                          MOV     (SP)+,@#RESVEC          ;LOAD RESERVED INST TRAP VECTOR
031430  000000                                  HALT                            ;SHOULD TRAP TO 4 IN USER MODE
031432  104000                          1$:     HLT                             ;ERROR! HALT ABOVE FAILED IN USER MODE
031434  000404                                  BR      3$
031436  010716                          2$:     MOV     PC,(SP)                 ;REPLACE RETURN PC WITH
031440  062716  000006                          ADD     #3$-.,(SP)              ;ADDRESS OF 3$ BELOW
031444  000002                                  RTI                             ;RETURN (TO 3$)

031446  012737  055064  000004          3$:     MOV     #ERPRT,@#ERRVEC         ;RESTORE ERROR TRAP VECTOR
031454  012737  054774  000010                  MOV     #RESERR,@#RESVEC
031462  105037  177776                          CLRB    @#PSW
                                        ;;*****************************************************************
                                        ;*TEST 76       CHECK RESET IN USER MODE
                                        ;;*****************************************************************
031466                                  TST76:
031466  000004                                  SCOPE
031470  112737  000076  001202                  MOVB    #76,@#$TSTNM
031476  000277                          RESET1: SCC
031500  013700  177776                          MOV     @#PSW,R0                ;GET CURRENT PSW
031504  000277                                  SCC
031506  000005                                  RESET
031510  023700  177776                          CMP     @#PSW,R0                ;CHECK THAT PSW UNCHANGED BY RESET ABOVE
031514  001401                                  BEQ     .+4
031516  104000                                  HLT                             ;ERROR! RESET CLEARED MODE BITS IN PSW
031520  010037  177776                          MOV     R0,@#PSW                ;RESTORE PSW (FOR ERROR)
031524                                  ENDCP:
031524  000004                          RELE6:  SCOPE
031526  010702                                  MOV     PC,R2
031530  062702  000012                          ADD     #12,R2
031534  012707  036574                          MOV     #RELOC,PC               ;GO RELOCATE PROGRAM CODE
031540  000000                          REL66:  .WORD   0
                                        ;66666666666666 LAST ADDRESS OF CODE TO BE RELOCATED 66666666666

                                        ;;*****************************************************************
                                        ;*TEST 77       TEST STACK LIMIT REGISTER
                                        ;;*****************************************************************
031542                                  TST77:
031542  012767  000001  147554                  MOV     #1,$TIMES               ;;DO 1 ITERATION
```

```
031550  000004                                  SCOPE
031552  112737  000077  001202                  MOVB    #77,@#$TSTNM

                                                .SBTTL  START OF SECTION 7
                                        ;7777777777777 FIRST ADDRESS TO BE RELOCATED 777777777
031560  112737  000077  001202  REL7:   MOVB    #$TN-1,@#$TSTNM
031566  010700                          MOV     PC,R0           ;GET PC
031570  005740                          TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL7
031572  010037  001554                  MOV     R0,@#FRSTAD     ;SAVE
031576  010700                          MOV     PC,R0           ;GET CURRENT PC
031600  162700  031600                  SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
031604  010037  001550                  MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
031610  010737  001212                  MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
031614  062737  000026  001212          ADD     #26,@#$LPERR    ;ADJUST
031622  013737  001212  001210          MOV     @#$LPERR,@#$LPADR
031630  105737  001544                  TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
031634  001402                          BEQ     .+6
031636  000167  000726                  JMP     RELE7
                                        ;THIS TEST SHIFTS A '1' BIT THROUGH ALL BIT POSITIONS
031642  012702  177774                  MOV     #STKLMT,R2                      ;GET ADDRESS OF STACK LIM REG
031646  005022                          CLR     (R2)+                   ;CLEAR STACK LIMIT REG
031650  032712  000020                  BIT     #20,(R2)                        ;EXIT TEST IF 'T' BIT IS SET
031654  001111                          BNE     101$
031656  052712  000340                  BIS     #340,(R2)               ;SET PRIORITY LEVEL 7 TO PREVENT
                                                                        ;ANY INTERRUPTS FROM OCCURRING
031662  012700  000400                  MOV     #400,R0                 ;SET CHECK DATA
031666  010042                  1$:     MOV     R0,-(R2)                ;MOVE TO STACK LIMIT REG
031670  022200                          CMP     (R2)+,R0                ;AND CHECK RESULT
031672  001401                          BEQ     2$
031674  104000                          HLT                             ;ERROR! STACK LIMIT DID NOT
                                                                        ;LOAD CORRECTLY. CORRECT RESULT
                                                                        ;IS IN R0
031676  006300                  2$:     ASL     R0                      ;SHIFT '1' BIT LEFT
031700  103372                          BCC     1$                      ;LOOP UNTIL 1 BIT SHIFTS OUT
031702  005042                          CLR     -(R2)                   ;CLEAR STACK LIMIT REG

                                        ;THIS TEST CHECKS THAT A PROPER 'RED' ZONE VIOLATION OCCURS, NOTE THAT
                                        ;NO 'RED ZONE' VIOLATION WILL OCCUR IF IN USER MODES.
                                        ;A RED ZONE VIOLATION PUSHES THE CURRENT PSW,PC ON [illegible] STACK AT 2 AND 0
                                        ;AND TAKES THE NEXT INSTRUCTION FROM THE PC IN LOC [illegible]04. THE INST-
                                        ;RUCTION CAUSING THE RED ZONE VIOLATION IS 'ABORTE[illegible]
031704  010746                          MOV     PC,-(SP)                ;GET CURRENT PC
031706  062716  000060                  ADD     #4$-.,(SP)              ;FORM ADDRESS OF 4$ BELOW
031712  012637  000004                  MOV     (SP)+,@#ERRVEC          ;SET ERROR TRAP VECTOR TO 4$ BELOW
031716  013737  177776  000006          MOV     @#PSW,@#ERRVEC+2        ;RETAIN CURRENT STATUS ON TRAP
031724  010712                          MOV     PC,(R2)                 ;SET STACK LIMIT TO CURRENT PC
                                                                        ;+400
031726  011206                          MOV     (R2),SP                 ;AND STACK PTR = STACK LIMIT REG
031730  010603                          MOV     SP,R3                   ;SAVE STACK PTR
031732  016304  000336                  MOV     336(R3),R4              ;SAVE MEMORY LOC CONTENTS
                                                                        ;AT 'RED ZONE' BOUNDARY
031736  032737  140000  177776          BIT     #UM,@#PSW               ;BRANCH IF IN KERNEL MODE
031744  001403                          BEQ     20$
031746  010466  000336                  MOV     R4,336(SP)              ;SHOULD NOT CAUSE TRAP
031752  000432                          BR      100$
```

```
031754  005066  000336          20$:    CLR     336(SP)                 ;SHOULD CAUSE 'RED ZONE' TRAP
031760  012706  000700          3$:     MOV     #USESTK,SP              ;RESTORE THE STACK
031764  104000                          HLT                             ;ERROR! FAILED TO TRAP

031766  032737  140000  000002  4$:     BIT     #UM,@#2                 ;CHECK IF TRAPPED WHEN IN USER
                                                                        ;MODE (2 CONTAINS OLD PSW)
031774  001013                          BNE     99$                     ;GO TO ERROR CALL
031776  010600                          MOV     SP,R0                   ;STACK PTR SHOULD = 0
032000  001011                          BNE     99$                     ;GO TO ERROR CALL IF NOT 0
032002  026304  000336                  CMP     336(R3),R4              ;CHECK THAT INST WAS ABORTED
032006  001006                          BNE     99$                     ;GO REPORT ERRPR
032010  005012                  5$:     CLR     (R2)                    ;CLEAR STACK LIMIT REG
032012  010705                          MOV     PC,R5                   ;GET CURRENT PC
032014  062705  177744                  ADD     #3$-.,R5                ;FORM ADDRESS OF  3$ ABOVE
032020  020516                          CMP     R5,(SP)                 ;CHECK THAT RETURN PC IS ON
                                                                        ;THE STACK (AT 0)
032022  001406                          BEQ     100$                    ;EXIT TEST

                                ;ERROR
032024  005012                  99$:    CLR     (R2)                    ;CLEAR STACK LIMIT REG
032026  010463  000336                  MOV     R4,336(R3)              ;RESTORE MEM LOCATION
032032  012706  000700                  MOV     #USESTK,SP              ;SET STACK PTR
032036  104000                          HLT                             ;ERROR!
032040  010463  000336          100$:   MOV     R4,336(R3)              ;RESTORE MEM LOCATION
032044  005022                          CLR     (R2)+                   ;CLEAR STACK LIM REG
032046  012706  000700                  MOV     #USESTK,SP              ;SET STACK PTR
032052  042712  000340                  BIC     #340,(R2)               ;SET PRIORITY LEVEL BACK TO 0
032056  012737  055064  000004          MOV     #ERPRT,@#ERRVEC         ;RESTORE ERROR TRAP VECTOR
032064  013737  177776  000006          MOV     @#PSW,@#EPRVEC+2
032072  112737  000340  000006          MOVB    #PR7,@#ERRVEC+2
032100                          101$:
                                ;;************************************************************
                                ;*TEST 100      MEMORY MANAGEMENT REGISTER TESTS
                                ;*      PDR TEST - THIS TEST WRITES 64. RANDOM #'S INTO EACH PDR REGISTER
                                ;*      NOTE:  IF MEM MGMT IS ENABLED ONLY PDR/PAR PAIRS 3-5 ARE TESTED.
                                ;;************************************************************
032100                          TST100:
032100  000004                          SCOPE
032102  112737  000100  001202          MOVB    #100,@#$TSTNM

032110  012702  032340          KTPDR:  MOV     #PDRTBL,R2      ;SET TABLE ADDRESS OF PDR'S
032114  012705  100361                  MOV     #100361,R5      ;SET BIT MASK
032120  012200                  1$:     MOV     (R2)+,R0        ;GET PDR ADDRESS
032122  001435                          BEQ     100$            ;EXIT ON '0' TERMINATOR
032124  012716  000010          2$:     MOV     #8.,(SP)        ;SET LOOP COUNT (FOR 8 REGS)
032130  105737  001545                  TSTB    @#MMON          ;BRANCH IF MEM MGMT DISABLED
032134  001404                          BEQ     3$
032136  062700  000006                  ADD     #6,R0           ;SET R0 TO PDR3
032142  012716  000003                  MOV     #3,(SP)         ;AND LIMIT TO TEST 3 PDRS
032146  012703  000040          3$:     MOV     #32.,R3         ;SET DATA COUNT
032152  005004                          CLR     R4              ;INITIALIZE DATA TO BE WRITTEN
032154  040504                  4$:     BIC     R5,R4           ;CLEAR NON-SETTABLE BITS
032156  010410                          MOV     R4,(R0)         ;WRITE INTO PDR
032160  021004                          CMP     (R0),R4         ;AND CHECK DATA READ BACK
032162  001013                          BNE     99$             ;GO TO ERROR CALL
032164  005104                          COM     R4              ;COMPLEMENT DATA
```

```
032166 040504                          BIC     R5,R4            ;CLEAR NON-SETTABLE BITS
032170 010410                          MOV     R4,(R0)          ;WRITE COMPLEMENT DATA INTO PDR
032172 021004                          CMP     (R0),R4          ;AND CHECK
032174 001006                          BNE     99$              ;GO TO ERROR CALL
032176 060104                          ADD     R1,R4            ;STEP DATA
032200 077313                          SOB     R3,4$
032202 005020                  5$:     CLR     (R0)+            ;STEP TO NEXT REGISTER
032204 005316                          DEC     (SP)             ;DECREMENT REGISTER COUNT
032206 001357                          BNE     3$
032210 000743                          BR      1$               ;GET NEXT SET OF 8 REGISTERS

032212 104000                  99$:    HLT                      ;ERROR! INCORRECT DATA READ
                                                                ;BACK FROM PDR. ADDRESS OF
                                                                ;PDR IS IN R0, DATA IS IN R4
032214 000772                          BR      5$               ;STEP TO NEXT REGISTER
032216                         100$:
                               ;;*********************************************************
                               ;*TEST 101      PAR TEST
                               ;*       PAR TEST - THIS TEST WRITES 64. COMPLEMENTING RANDOM #'S INTO EACH PAR.
                               ;;*********************************************************
032216                         TST101:
032216 000004                          SCOPE
032220 112737 000101 001202            MOVB    #101,@#$TSTNM
032226 012702 032346           KTPAR:  MOV     #PARTBL,R2       ;GET TABLE ADDRESS OF PAR'S
032232 005005                          CLR     R5
032234 012705 170000                   MOV     #170000,R5
032240 012200                  1$:     MOV     (R2)+,R0         ;GET PAR ADDRESS
032242 001435                          BEQ     100$             ;EXIT ON '0' TERMINATOR
032244 012716 000010           2$:     MOV     #8.,(SP)         ;SET LOOP COUNT (FOR 8 REGS.)
032250 105737 001545                   TSTB    @#MMON           ;BRANCH IF MEM MGMT DISABLED
032254 001404                          BEQ     3$
032256 062700 000006                   ADD     #6,R0            ;SET R0 TO PAR3
032262 012716 000003                   MOV     #3,(SP)          ;AND LIMIT TEST TO 3 PARS
032266 012703 000040           3$:     MOV     #32.,R3          ;SET DATA COUNT
032272 005004                          CLR     R4               ;INITIALIZE DATA
032274 040504                  4$:     BIC     R5,R4            ;CLEAR NON-SETTABLE BITS
032276 010410                          MOV     R4,(R0)          ;WRITE INTO PAR
032300 021004                          CMP     (R0),R4          ;AND CHECK
032302 001013                          BNE     99$              ;TAKE ERROR EXIT
032304 005104                          COM     R4               ;COMPLEMENT DATA
032306 040504                          BIC     R5,R4            ;CLEAR NON-SETTABLE BITS
032310 010410                          MOV     R4,(R0)          ;WRITE COMPLEMENT DATA
032312 021004                          CMP     (R0),R4          ;AND CHECK
032314 001006                          BNE     99$              ;TAKE ERROR EXIT
032316 060104                          ADD     R1,R4            ;STEP DATA
032320 077313                          SOB     R3,4$            ;LOOP UNTIL FINISHED

032322 005020                  5$:     CLR     (R0)+
032324 005316                          DEC     (SP)             ;DECREMENT REGISTER COUNT
032326 001357                          BNE     3$               ;BRANCH IF 8 REGS NOT DONE
032330 000743                          BR      1$

032332 104000                  99$:    HLT                      ;ERROR! INCORRECT DATA READ BACK
                                                                ;FROM PAR. ADDRESS OF PAR IS IN
                                                                ;R0, DATA IS IN R4
032334 000772                          BR      5$               ;DO NEXT REGISTER
```

```
032336                                100$:
032336  000406                                BR      TST102          ;;GO TO NEXT TEST
                                      ;TABLES FOR PDR & PAR TESTS ABOVE
032340  172300                        PDRTBL: .WORD   KIPDR0
032342  177600                                .WORD   UIPDR0
032344  000000                                .WORD   0               ;TERMINATOR

032346  172340                        PARTBL: .WORD   KIPAR0
032350  177640                                .WORD   UIPAR0
032352  000000                                .WORD   0               ;TERMINATOR

                                      ;;*************************************************************
                                      ;*TEST 102        CHECK KT ABORT LOGIC
                                      ;*       THIS TEST CHECKS KT ABORT LOGIC. TEST CREATES AN ABORT CONDITION
                                      ;*       AND INSURES THAT ABORT IS TAKEN PROPERLY. NOTE: TEST IS EXECUTED ONLY
                                      ;*       IF TEST IS ENTERED WITH MEM MGMT ENABLED.
                                      ;;*************************************************************
032354                                TST102:
032354  000004                                SCOPE
032356  112737  000102  001202                MOVB    #102,@#$TSTNM
032364  105737  001545                KTABT:  TSTB    @#MMON          ;BRANCH IF MEM MGMT DISABLED
032370  001477                                BEQ     KTEX
032372  005037  172350                        CLR     @#KIPAR4        ;SET UP MEM MGMT REGISTERS
032376  005037  172310                        CLR     @#KIPDR4        ;TO ABORT IF A MEMORY
032402  005037  177650                        CLR     @#UIPAR4        ;REFERENCE IS MADE TO
032406  005037  177610                        CLR     @#UIPDR4        ;ADDRESSES (VIRTUAL) BETWEEN
032412  013746  000250                1$:     MOV     @#MMVEC,-(SP)   ;SAVE MEM MGMT VECTOR
032416  013746  000252                        MOV     @#MMVEC+2,-(SP) ;AND PRIORITY
032422  010746                                MOV     PC,-(SP)        ;SET MEM MGMT
032424  062716  000040                        ADD     #4$-.,(SP)      ;VECTOR TO 4$ BELOW
032430  012637  000250                        MOV     (SP)+,@#MMVEC
032434  013737  177776  000252                MOV     @#PSW,@#MMVEC+2
032442  005000                                CLR     R0              ;CLEAR ABORT INDICATOR
032444  010702                                MOV     PC,R2           ;SET R2 AND R3 NOTE:
032446  012703  100000                        MOV     #100000,R3      ;THE REF VIA R3 CAUSES THE
032452  014223                        2$:     MOV     -(R2),(R3)+     ;ABORT
032454  005700                        3$:     TST     R0              ;BRANCH IF THE ABORT OCCURRED
032456  001001                                BNE     .+4
032460  104000                                HLT                     ;REPORT ERROR
032462  000433                                BR      100$
                                      ;ABORT HERE
032464  013700  177776                4$:     MOV     @#PSW,R0        ;SR0 SHOULD CONTAIN
032470  000300                                SWAB    R0              ;CAUSE FOR ABORT AND
032472  006200                                ASR     R0              ;ALSO WHICH SEGMENT
032474  042700  177637                        BIC     #177637,R0      ;WAS IN USE WHEN ABORT
032500  062700  100011                        ADD     #100011,R0      ;OCCURRED.
032504  020037  177572                        CMP     R0,@#SR0
032510  001013                                BNE     99$
032512  012700  032452                        MOV     #2$,R0          ;GET ADDRESS OF INST THAT ABORTED
032516  020037  177576                        CMP     R0,@#SR2        ;THAT ABORTED
032522  001006                                BNE     99$
032524  012700  032452                        MOV     #2$,R0
032530  005720                        5$:     TST     (R0)+           ;R0=ADDRESS OF INST FOLLOWING ABORT
032532  020016                                CMP     R0,(SP)         ;(3$)
032534  001001                                BNE     99$
032536  000002                                RTI                     ;RETURN
```

```
                                     ;ENTER HERE ON ERROR
032540  104000                       99$:    HLT                             ;REPORT ERROR
032542  010716                               MOV     PC,(SP)
032544  062716  177710                       ADD     #3$-.,(SP)
032550  000002                               RTI                             ;RETURN
032552  012637  000252               100$:   MOV     (SP)+,@#MMVEC+2 ;RESTORE ABORT VECTOR
032556  012637  000250                       MOV     (SP)+,@#MMVEC   ;& PRIORITY.
032562  012737  000001  177572               MOV     #1,@#SR0        ;CLEAR ERROR CONDITIONS
032570                               KTEX:
032570  000004                       RELE7:  SCOPE
032572  010702                               MOV     PC,R2
032574  062702  000012                       ADD     #12,R2
032600  012707  036574                       MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
032604  000000                       REL77:  .WORD   0
                                     ;777777777777 LAST ADDRESS OF CODE TO BE RELOCATED 77777777777

                                     ;;*****************************************************************
                                     ;*TEST 103       FLOATING POINT TEST 1
                                     ;*
                                     ;*       THIS TEST TAKES TWO RANDOM NUMBERS (A AND B) AND
                                     ;*       COMPARES THE RESULTS OF TWO EQUAL CALCULATIONS.
                                     ;*       EACH SECTION EVALUATES A DIFFERENT EQUATION AS DESCRIBED BELOW:
                                     ;*       SECT1   (A+B)**2=A**2+2*A*B+B**2
                                     ;*       SECT2   (A+B)*(A-B)=A**2-B**2
                                     ;*       SECT3   A/B*B=A
                                     ;;*****************************************************************
032606                               TST103:
032606  012767  000001  146510               MOV     #1,$TIMES       ;;DO 1 ITERATION
032614  000004                               SCOPE
032616  112737  000103  001202               MOVB    #103,@#$TSTNM

                                             .SBTTL  START OF SECTION 8
                                     ;888888888888 FIRST ADDRESS TO BE RELOCATED 888888888
032624  112737  000103  001202       REL8:   MOVB    #$TN-1,@#$TSTNM
032632  010700                               MOV     PC,R0           ;GET PC
032634  005740                               TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL8
032636  010037  001554                       MOV     R0,@#FRSTAD     ;SAVE
032642  010700                               MOV     PC,R0           ;GET CURRENT PC
032644  162700  032644                       SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
032650  010037  001550                       MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
032654  010737  001212                       MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
032660  062737  000026  001212               ADD     #26,@#$LPERR    ;ADJUST
032666  013737  001212  001210               MOV     @#$LPERR,@#$LPADR
032674  105737  001544                       TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
032700  001402                               BEQ     .+6
032702  000167  002400                       JMP     RELE8
032706  004737  053056               100$:   JSR     PC,@#FLTSGL     ;GET RANDOM OPERANDS
032712  170127  000000                       LDFPS   #0              ;INIT FPS
032716  172537  001424                       LDF     @#FTMP0,AC1     ;LOAD A OPERAND
032722  172437  001430                       LDF     @#FTMP2,AC0     ;LOAD B OPERAND
032726  013737  001444  001412               MOV     @#FREG0,@#$AC1  ;SETUP EXTENDED
032734  013737  001446  001410               MOV     @#FREG1,@#$AC0  ;EXPONENTS
032742  004767  002202                       JSR     PC,FLTADD       ;PERFORM THE ADD
032746  174100                               STF     AC1,AC0         ;SETUP AC0 TO
032750  013737  001412  001410               MOV     @#$AC1,@#$AC0   ;PERFORM THE SQUARE
032756  004767  002116                       JSR     PC,FLTMPY       ;DO THE MULTIPLY
```

```
	032762	174137	001434			STF	AC1,@#FTMP4		;SAVE RESULT
	032766	013737	001412	001450		MOV	@#$AC1,@#FREG2		;AND SOFTWARE EXP
					;
					;NOW DO THE RIGHT HAND SIDE OF THE EQUATION
					;DO THE A*A FIRST
	032774	013737	001444	001410		MOV	@#FREG0,@#$AC0		;GET EXT EXPONENT
	033002	172437	001424			LDF	@#FTMP0,AC0		;LOAD OPERAND A
	033006	013737	001410	001412		MOV	@#$AC0,@#$AC1		;SET OPERAND B EXT EXPONENT
	033014	172500				LDF	AC0,AC1			;LOAD B OPERAND
	033016	004767	002056			JSR	PC,FLTMPY		;EXECUTE THE MULTIPLY
	033022	174102				STF	AC1,AC2			;SAVE RESULT
	033024	013737	001412	001414		MOV	@#$AC1,@#$AC2
					;
					;NOW DO THE B*B
	033032	172437	001430			LDF	@#FTMP2,AC0		;LOAD B OPERAND
	033036	172500				LDF	AC0,AC1
	033040	013737	001446	001410		MOV	@#FREG1,@#$AC0		;AND EXT EXPONENT
	033046	013737	001410	001412		MOV	@#$AC0,@#$AC1
	033054	004767	002020			JSR	PC,FLTMPY		;DO THE MULTIPLY
	033060	174103				STF	AC1,AC3			;SAVE THE RESULT
	033062	013737	001412	001416		MOV	@#$AC1,@#$AC3
					;
					;NOW DO THE 2*B*A
	033070	012701	001430			MOV	#FTMP2,R1
	033074	172411				LDF	(R1),AC0		;LOAD THE B OPERAND
	033076	172541				LDF	-(R1),AC1		;LOAD THE A OPERAND
	033100	013737	001446	001410		MOV	@#FREG1,@#$AC0		;AND THE EXT EXPONENTS
	033106	013737	001444	001412		MOV	@#FREG0,@#$AC1
	033114	004767	001760			JSR	PC,FLTMPY		;DO THE MULTIPLY
	033120	172427	040000			LDF	#^O40000,AC0		;SETUP TO MULTIPLY BY TWO
	033124	012737	000002	001410		MOV	#2,@#$AC0
	033132	004767	001742			JSR	PC,FLTMPY		;DO THE MULTIPLY
					;
					;NOW SUM THE RESULTS
	033136	013737	001416	001410		MOV	@#$AC3,@#$AC0
	033144	172403				LDF	AC3,AC0			;GET RESULT OF B*B
	033146	004767	001776			JSR	PC,FLTADD		;ADD THE RESULT
	033152	172402				LDF	AC2,AC0			;GET RESULT OF A*A
	033154	013737	001414	001410		MOV	@#$AC2,@#$AC0
	033162	004767	001762			JSR	PC,FLTADD		;ADD THIS RESULT
	033166	174137	001440			STF	AC1,@#FTMP6		;SAVE FINAL RESULT
	033172	013737	001412	001452		MOV	@#$AC1,@#FREG3
					;
					;NOW CHECK BOTH SIDES OF THE EQUATION
					;CALCULATE THE NUMBER OF CORRECT BITS
					;PUT LARGEST EXPONENT OF A**2 OR B**2 IN $AC2
	033200	023737	001414	001416		CMP	@#$AC2,@#$AC3
	033206	002003				BGE	1$			;BRANCH IF $AC2 ALREADY HAS LARGEST
	033210	013737	001416	001414		MOV	@#$AC3,@#$AC2		;$AC3 WAS LARGER
	033216	163737	001412	001414	1$:	SUB	@#$AC1,@#$AC2		;NOW CALCULATE NUMBER
	033224	162737	000024	001414		SUB	#20.,@#$AC2		;OF CORRECT BITS WITHIN 2
	033232	005437	001414			NEG	@#$AC2			;MAKE RESULT POSITIVE
	033236	172437	001434			LDF	@#FTMP4,AC0		;LOAD RESULT OF LEFT HAND SIDE
	033242	013737	001450	001410		MOV	@#FREG2,@#$AC0		;AND EXTENDED EXPONENT
	033250	004767	001670			JSR	PC,FLTSUB		;SUBTRACT TO SEE HOW CLOSE THEY ARE
	033254	163737	001452	001412		SUB	@#FREG3,@#$AC1		;GET DIFFERENCE IN EXT EXPONENTS
```

```
                                                                        ;ACTUAL EXP'S ARE EQUAL TO 200
033262  100002                                  BPL     3$              ;ENSURE RESULT IS POSITIVE
033264  005437  001412                          NEG     @#SAC1
033270  023737  001414  001412  3$:             CMP     @#SAC2,@#SAC1   ;ANSWERS WITHIN ALLOWABLE NUMBER?
033276  003401                                  BLE     SECT2           ;BRANCH IF YES
033300  104014                          4$:     ERROR   14              ;RESULTS ARE WRONG
                                        ;;*******************************************************************
033302  170127  000000                  SECT2:  LDFPS   #0
                                        ;DO A+B
033306  172537  001424                          LDF     @#FTMP0,AC1     ;LOAD A OPERAND
033312  172437  001430                          LDF     @#FTMP2,AC0     ;LOAD B OPERAND
033316  013737  001444  001412                  MOV     @#FREG0,@#SAC1
033324  013737  001446  001410                  MOV     @#FREG1,@#SAC0
033332  004767  001612                          JSR     PC,FLTADD       ;ADD THEM
033336  174102                                  STF     AC1,AC2         ;SAVE IN AC2
033340  013737  001412  001414                  MOV     @#SAC1,@#SAC2   ;AND EXT EXPONENT
                                        ;NOW DO THE A-B
033346  172537  001424                          LDF     @#FTMP0,AC1     ;LOAD OPERAND A
033352  013737  001444  001412                  MOV     @#FREG0,@#SAC1  ;AND EXT EXPONENT
033360  172437  001430                          LDF     @#FTMP2,AC0     ;LOAD OPERAND B
033364  013737  001446  001410                  MOV     @#FREG1,@#SAC0
033372  004767  001546                          JSR     PC,FLTSUB       ;SUBTRACT THEM
                                        ;NOW DO (A+B)*(A-B)
033376  172402                                  LDF     AC2,AC0         ;GET RESULT OF (A+B)
033400  013737  001414  001410                  MOV     @#SAC2,@#SAC0
033406  004767  001466                          JSR     PC,FLTMPY       ;FORM THE PRODUCT
033412  174137  001434                          STF     AC1,@#FTMP4     ;SAVE RESULT
033416  013737  001412  001450                  MOV     @#SAC1,@#FREG2  ;AND EXT EXPONENT
                                        ;NOW DO THE B*B
033424  172437  001430                          LDF     @#FTMP2,AC0     ;LOAD OPERAND B
033430  013737  001446  001410                  MOV     @#FREG1,@#SAC0
033436  172500                                  LDF     AC0,AC1         ;B OPERAND IS IN AC0
033440  013737  001410  001412                  MOV     @#SAC0,@#SAC1   ;AND EXT EXPONENT
033446  004767  001426                          JSR     PC,FLTMPY       ;
033452  174102                                  STF     AC1,AC2         ;SAVE RESULT IN AC2
033454  013737  001412  001414                  MOV     @#SAC1,@#SAC2
                                        ;NOW DO THE A*A
033462  172437  001424                          LDF     @#FTMP0,AC0     ;LOAD OPERAND A
033466  013737  001444  001410                  MOV     @#FREG0,@#SAC0
033474  172500                                  LDF     AC0,AC1
033476  013737  001410  001412                  MOV     @#SAC0,@#SAC1
033504  004767  001370                          JSR     PC,FLTMPY       ;EXECUTE THE MULTIPLY
033510  013737  001412  001416                  MOV     @#SAC1,@#SAC3   ;SAVE EXT EXPO OF A*A
                                        ;NOW DO A**2-B**2
033516  172402                                  LDF     AC2,AC0         ;GET B*B
033520  013737  001414  001410                  MOV     @#SAC2,@#SAC0   ;A*A IN AC1
033526  004767  001412                          JSR     PC,FLTSUB
033532  174137  001440                          STF     AC1,@#FTMP6     ;SAVE IN MEMORY
033536  013737  001412  001452                  MOV     @#SAC1,@#FREG3
                                        ;NOW COMPUTE THE RESULTS
                                        ;CALCULATE THE NUMBER OF CORRECT BITS
033544  023737  001414  001416                  CMP     @#SAC2,@#SAC3   ;DETERMINE WHICH EXP IS LARGER
033552  002003                                  BGE     2$              ;BRANCH IF AC2 LARGER
033554  013737  001416  001414                  MOV     @#SAC3,@#SAC2   ;PUT LARGEST IN AC2
033562  163737  001412  001414  2$:             SUB     @#SAC1,@#SAC2
033570  162737  000025  001414                  SUB     #21.,@#SAC2
```

```
 033576  005437  001414                   NEG     @#$AC2
 033602  172437  001434                   LDF     @#FTMP4,AC0       ;GET LEFT HAND SIDE
 033606  013737  001450  001410           MOV     @#FREG2,@#$AC0
 033614  004767  001324                   JSR     PC,FLTSUB         ;SUBTRACT TO SEE HOW CLOSE THEY ARE
 033620  163737  001452  001412           SUB     @#FREG3,@#$AC1    ;SUB EXT EXPONENTS
                                                                    ;ACTUAL EXPONENTS ARE EQUAL
 033626  100002                           BPL     1$                ;MAKE SURE RESULT IS POSITIVE
 033630  005437  001412                   NEG     @#$AC1
 033634  023737  001414  001412   1$:     CMP     @#$AC2,@#$AC1     ;RESULTS WITHIN RANGE ALLOWED?
 033642  003401                           BLE     SECT3             ;BRANCH IF YES
 033644  104014                           ERROR   14                ;RESULTS WRONG

                                  ;;*******************************************************
 033646  172537  001424           SECT3:  LDF     @#FTMP0,AC1       ;LOAD OPERAND A
 033652  172437  001430                   LDF     @#FTMP2,AC0       ;AND OPERAND B
 033656  013737  001444  001412           MOV     @#FREG0,@#$AC1
 033664  013737  001446  001410           MOV     @#FREG1,@#$AC0
 033672  004767  001224                   JSR     PC,FLTDIV         ;GO DIVIDE THEM
 033676  004767  001176                   JSR     PC,FLTMPY         ;MULTIPLY RESULT BY B
 033702  174137  001434                   STF     AC1,@#FTMP4       ;SAVE RESULT
 033706  013737  001412  001450           MOV     @#$AC1,@#FREG2
 033714  172437  001424                   LDF     @#FTMP0,AC0       ;LOAD OPERAND A
 033720  174037  001440                   STF     AC0,@#FTMP6       ;SAVE INCASE TYPE OUT
 033724  013737  001444  001410           MOV     @#FREG0,@#$AC0
 033732  013737  001444  001452           MOV     @#FREG0,@#FREG3
 033740  004767  001200                   JSR     PC,FLTSUB         ;SUBTRACT RIGHT AND LEFT HAND SIDES
 033744  163737  001444  001412           SUB     @#FREG0,@#$AC1    ;SEE IF RESULT OK
 033752  100002                           BPL     1$                ;ENSURE DIFFERANCE IS POSITIVE
 033754  005437  001412                   NEG     @#$AC1
 033760  022737  000027  001412   1$:     CMP     #23.,@#$AC1       ;RESULTS WITHIN 2 BITS?
 033766  003001                           BGT     2$                ;BRANCH IF NO
 033770  000401                           BR      TST104            ;;GO TO NEXT TEST
 033772  104014                   2$:     ERROR   14                ;RESULTS WRONG

                                  ;;*******************************************************
                                  ;*TEST 104      FLOATING POINT TEST 2
                                  ;*
                                  ;*      THIS TEST TAKES TWO RANDOM NUMBERS (A AND B) AND
                                  ;*      COMPARES THE RESULTS OF TWO EQUAL CALCULATIONS.
                                  ;*      EACH SECTION EVALUATES A DIFFERENT EQUATION AS DESCRIBED BELOW:
                                  ;*      SECT1   (A+B)**2=A**2+2*A*B+B**2
                                  ;*      SECT2   (A+B)*(A-B)=A**2-B**2
                                  ;*      SECT3   A/B*B=A
                                  ;;*******************************************************
 033774                           TST104:
 033774  000004                           SCOPE
 033776  112737  000104  001202           MOVB    #104,@#$TSTNM
 034004  012737  000001  001324           MOV     #1,@#$TIMES
 034012  004737  053050           100$:   JSR     PC,@#FLTDBL       ;GET RANDOM OPERANDS
 034016  170127  000200                   LDFPS   #200              ;INIT FPS
 034022  172537  001424                   LDF     @#FTMP0,AC1       ;LOAD A OPERAND
 034026  172437  001434                   LDF     @#FTMP4,AC0       ;LOAD B OPERAND
 034032  013737  001444  001412           MOV     @#FREG0,@#$AC1    ;SETUP EXTENDED
 034040  013737  001446  001410           MOV     @#FREG1,@#$AC0    ;EXPONENTS
 034046  004767  001076                   JSR     PC,FLTADD         ;PERFORM THE ADD
 034052  174100                           STF     AC1,AC0           ;SETUP AC0 TO
```

```
034054  013737  001412  001410          MOV     @#$AC1,@#$AC0   ;PERFORM THE SQUARE
034062  004767  001012                  JSR     PC,FLTMPY       ;DO THE MULTIPLY
034066  174137  001464                  STF     AC1,@#FLTMP0    ;SAVE RESULT
034072  013737  001412  001450          MOV     @#$AC1,@#FREG2  ;AND SOFTWARE EXP
                                ;
                                ;NOW DO THE RIGHT HAND SIDE OF THE EQUATION
                                ;DO THE A*A FIRST
034100  013737  001444  001410          MOV     @#FREG0,@#$AC0  ;GET EXT EXPONENT
034106  172437  001424                  LDF     @#FLTMP0,AC0    ;LOAD OPERAND A
034112  013737  001410  001412          MOV     @#$AC0,@#$AC1   ;SET OPERAND B EXT EXPONENT
034120  172500                          LDF     AC0,AC1         ;LOAD B OPERAND
034122  004767  000752                  JSR     PC,FLTMPY       ;EXECUTE THE MULTIPLY
034126  174102                          STF     AC1,AC2         ;SAVE RESULT
034130  013737  001412  001414          MOV     @#$AC1,@#$AC2
                                ;
                                ;NOW DO THE B*B
034136  172437  001434                  LDF     @#FTMP4,AC0     ;LOAD B OPERAND
034142  172500                          LDF     AC0,AC1
034144  013737  001446  001410          MOV     @#FREG1,@#$AC0  ;AND EXT EXPONENT
034152  013737  001410  001412          MOV     @#$AC0,@#$AC1
034160  004767  000714                  JSR     PC,FLTMPY       ;DO THE MULTIPLY
034164  174103                          STF     AC1,AC3         ;SAVE THE RESULT
034166  013737  001412  001416          MOV     @#$AC1,@#$AC3
                                ;
                                ;NOW DO THE 2*B*A
034174  012701  001434                  MOV     #FTMP4,R1
034200  172411                          LDF     (R1),AC0        ;LOAD THE B OPERAND
034202  172541                          LDF     -(R1),AC1       ;LOAD THE A OPERAND
034204  013737  001446  001410          MOV     @#FREG1,@#$AC0  ;AND THE EXT EXPONENTS
034212  013737  001444  001412          MOV     @#FREG0,@#$AC1
034220  004767  000654                  JSR     PC,FLTMPY       ;DO THE MULTIPLY
034224  172427  040000                  LDF     #^O40000,AC0    ;SETUP TO MULTIPLY BY TWO
034230  012737  000002  001410          MOV     #2,@#$AC0
034236  004767  000636                  JSR     PC,FLTMPY       ;DO THE MULTIPLY
                                ;
                                ;NOW SUM THE RESULTS
034242  013737  001416  001410          MOV     @#$AC3,@#$AC0
034250  172403                          LDF     AC3,AC0         ;GET RESULT OF B*B
034252  004767  000672                  JSR     PC,FLTADD       ;ADD THE RESULT
034256  172402                          LDF     AC2,AC0         ;GET RESULT OF A*A
034260  013737  001414  001410          MOV     @#$AC2,@#$AC0
034266  004767  000656                  JSR     PC,FLTADD       ;ADD THIS RESULT
034272  174137  001474                  STF     AC1,@#FLTMP1    ;SAVE FINAL RESULT
034276  013737  001412  001452          MOV     @#$AC1,@#FREG3
                                ;
                                ;NOW CHECK BOTH SIDES OF THE EQUATION
                                ;CALCULATE THE NUMBER OF CORRECT BITS
                                ;PUT LARGEST EXPONENT OF A**2 OR B**2 IN $AC2
034304  023737  001414  001416          CMP     @#$AC2,@#$AC3
034312  002003                          BGE     1$              ;BRANCH IF $AC2 ALREADY HAS LARGEST
034314  013737  001416  001414          MOV     @#$AC3,@#$AC2   ;$AC3 WAS LARGER
034322  163737  001412  001414  1$:     SUB     @#$AC1,@#$AC2   ;NOW CALCULATE NUMBER
034330  162737  000065  001414          SUB     #53.,@#$AC2     ;OF CORRECT BITS WITHIN 2
034336  005437  001414                  NEG     @#$AC2          ;MAKE RESULT POSITIVE
034342  172437  001464                  LDF     @#FLTMP0,AC0    ;LOAD RESULT OF LEFT HAND SIDE
034346  013737  001450  001410          MOV     @#FREG2,@#$AC0  ;AND EXTENDED EXPONENT
```

```
034354  004767  000564                   JSR     PC,FLTSUB          ;SUBTRACT TO SEE HOW CLOSE THEY ARE
034360  163737  001452  001412           SUB     @#FREG3,@#SAC1     ;GET DIFFERENCE IN EXT EXPONENTS
                                                                    ;ACTUAL EXP'S ARE EQUAL TO 200
034366  100002                           BPL     3$                 ;ENSURE RESULT IS POSITIVE
034370  005437  001412                   NEG     @#SAC1
034374  023737  001414  001412   3$:     CMP     @#SAC2,@#SAC1      ;ANSWERS WITHIN ALLOWABLE NUMBER?
034402  003401                           BLE     SECT2D             ;BRANCH IF YES
034404  104011                   4$:     ERROR   11                 ;RESULTS ARE WRONG
                                 ;;*****************************************************************
034406  170127  000200           SECT2D: LDFPS   #200
                                 ;DO A+B
034412  172537  001424                   LDF     @#FTMP0,AC1        ;LOAD A OPERAND
034416  172437  001434                   LDF     @#FTMP4,AC0        ;LOAD B OPERAND
034422  013737  001444  001412           MOV     @#FREG0,@#SAC1
034430  013737  001446  001410           MOV     @#FREG1,@#SAC0
034436  004767  000506                   JSR     PC,FLTADD          ;ADD THEM
034442  174102                           STF     AC1,AC2            ;SAVE IN AC2
034444  013737  001412  001414           MOV     @#SAC1,@#SAC2      ;AND EXT EXPONENT
                                 ;NOW DO THE A-B
034452  172537  001424                   LDF     @#FTMP0,AC1        ;LOAD OPERAND A
034456  013737  001444  001412           MOV     @#FREG0,@#SAC1     ;AND EXT EXPONENT
034464  172437  001434                   LDF     @#FTMP4,AC0        ;LOAD OPERAND B
034470  013737  001446  001410           MOV     @#FREG1,@#SAC0
034476  004767  000442                   JSR     PC,FLTSUB          ;SUBTRACT THEM
                                 ;NOW DO (A+B)*(A-B)
034502  172402                           LDF     AC2,AC0            ;GET RESULT OF (A+B)
034504  013737  001414  001410           MOV     @#SAC2,@#SAC0
034512  004767  000362                   JSR     PC,FLTMPY          ;FORM THE PRODUCT
034516  174137  001464                   STF     AC1,@#FLTMP0       ;SAVE RESULT
034522  013737  001412  001450           MOV     @#SAC1,@#FREG2     ;AND EXT EXPONENT
                                 ;NOW DO THE B*B
034530  172437  001434                   LDF     @#FTMP4,AC0        ;LOAD OPERAND B
034534  013737  001446  001410           MOV     @#FREG1,@#SAC0
034542  172500                           LDF     AC0,AC1            ;B OPERAND IS IN AC0
034544  013737  001410  001412           MOV     @#SAC0,@#SAC1      ;AND EXT EXPONENT
034552  004767  000322                   JSR     PC,FLTMPY          ;
034556  174102                           STF     AC1,AC2            ;SAVE RESULT IN AC2
034560  013737  001412  001414           MOV     @#SAC1,@#SAC2
                                 ;NOW DO THE A*A
034566  172437  001424                   LDF     @#FTMP0,AC0        ;LOAD OPERAND A
034572  013737  001444  001410           MOV     @#FREG0,@#SAC0
034600  172500                           LDF     AC0,AC1
034602  013737  001410  001412           MOV     @#SAC0,@#SAC1
034610  004767  000264                   JSR     PC,FLTMPY          ;EXECUTE THE MULTIPLY
034614  013737  001412  001416           MOV     @#SAC1,@#SAC3      ;SAVE EXT EXPO OF A*A
                                 ;NOW DO A**2-B**2
034622  172402                           LDF     AC2,AC0            ;GET B*B
034624  013737  001414  001410           MOV     @#SAC2,@#SAC0      ;A*A IN AC1
034632  004767  000306                   JSR     PC,FLTSUB
034636  174137  001474                   STF     AC1,@#FLTMP1       ;SAVE IN MEMORY
034642  013737  001412  001452           MOV     @#SAC1,@#FREG3
                                 ;NOW COMPUTE THE RESULTS
                                 ;CALCULATE THE NUMBER OF CORRECT BITS
034650  023737  001414  001416           CMP     @#SAC2,@#SAC3      ;DETERMINE WHICH EXP IS LARGER
```

```
 7900  034656  002003                    BGE     2$              ;BRANCH IF AC2 LARGER
```

```
034660  013737  001416  001414          MOV     @#$AC3,@#$AC2           ;PUT LARGEST IN AC2
034666  163737  001412  001414  2$:     SUB     @#$AC1,@#$AC2
034674  162737  000066  001414          SUB     #54.,@#$AC2
034702  005437  001414                  NEG     @#$AC2
034706  172437  001464                  LDF     @#FLTMP0,AC0            ;GET LEFT HAND SIDE
034712  013737  001450  001410          MOV     @#FREG2,@#$AC0
034720  004767  000220                  JSR     PC,FLTSUB               ;SUBTRACT TO SEE HOW CLOSE THEY ARE
034724  163737  001452  001412          SUB     @#FREG3,@#$AC1          ;SUB EXT EXPONENTS
                                                                        ;ACTUAL EXPONENTS ARE EQUAL
034732  100002                          BPL     1$                      ;MAKE SURE RESULT IS POSITIVE
034734  005437  001412                  NEG     @#$AC1
034740  023737  001414  001412  1$:     CMP     @#$AC2,@#$AC1           ;RESULTS WITHIN RANGE ALLOWED?
034746  003401                          BLE     SECT3D                  ;BRANCH IF YES
034750  104011                          ERROR   11                      ;RESULTS WRONG

                                ;;*************************************************************
034752  172537  001424          SECT3D: LDF     @#FTMP0,AC1             ;LOAD OPERAND A
034756  172437  001434                  LDF     @#FTMP4,AC0             ;AND OPERAND B
034762  013737  001444  001412          MOV     @#FREG0,@#$AC1
034770  013737  001446  001410          MOV     @#FREG1,@#$AC0
034776  004767  000120                  JSR     PC,FLTDIV               ;GO DIVIDE THEM
035002  004767  000072                  JSR     PC,FLTMPY               ;MULTIPLY RESULT BY B
035006  174137  001464                  STF     AC1,@#FLTMP0            ;SAVE RESULT
035012  013737  001412  001450          MOV     @#$AC1,@#FREG2
035020  172437  001424                  LDF     @#FTMP0,AC0             ;LOAD OPERAND A
035024  174037  001474                  STF     AC0,@#FLTMP1            ;SAVE INCASE TYPE OUT
035030  013737  001444  001410          MOV     @#FREG0,@#$AC0
035036  013737  001444  001452          MOV     @#FREG0,@#FREG3
035044  004767  000074                  JSR     PC,FLTSUB               ;SUBTRACT RIGHT AND LEFT HAND SIDES
035050  163737  001444  001412          SUB     @#FREG0,@#$AC1          ;SEE IF RESULT OK
035056  100002                          BPL     1$                      ;ENSURE DIFFERANCE IS POSITIVE
035060  005437  001412                  NEG     @#$AC1
035064  022737  000067  001412  1$:     CMP     #55.,@#$AC1             ;RESULTS WITHIN 2 BITS?
035072  003505                          BLE     RELE8                   ;BRANCH IF YES
035074  104011                          ERROR   11                      ;RESULTS WRONG
035076  000503                          BR      RELE8

                                ;;*************************************************************
                                .SBTTL  FLOATING POINT MULTIPLY ROUTINE
                                ;*      THIS ROUTINE MULTIPLIES THE CONTENTS OF AC0 AND AC1
                                ;*      AND LEAVES THE RESULT IN AC1. IT ALSO TAKES CARE OF
                                ;*      THE SOFTWARE EXPONENTS THAT ARE KEPT IN $AC0 AND $AC1.
                                ;;*************************************************************
035100  063737  001410  001412  FLTMPY: ADD     @#$AC0,@#$AC1           ;ADD SOFTWARE EXPONENTS
035106  171100                          MULF    AC0,AC1                 ;DO THE MULTIPLY
035110  012746  100400                  MOV     #100400,-(SP)           ;PUT CONTROL WORD ON STACK
035114  004737  053210                  JSR     PC,@#EXPEXT             ;CALCULATE EXT EXPONENT
035120  000207                  1$:     RTS     PC                      ;RETURN

                                ;;*************************************************************
                                .SBTTL  FLOATING POINT DIVIDE ROUTINE
                                ;*      THIS ROUTINE DIVIDES THE CONTENTS OF AC1 BY AC0
                                ;*      AND LEAVES THE RESULT IN AC1.
                                ;;*************************************************************
035122  163737  001410  001412  FLTDIV: SUB     @#$AC0,@#$AC1           ;ADJUST SOFTWARE EXPONENTS
035130  174500                          DIVF    AC0,AC1                 ;EXECUTE THE DIVIDE
```

```
035132  012746  100400                   MOV     #100400,-(SP)    ;PUT CONTROL WORD ON STACK
035136  004737  053210                   JSR     PC,@#EXPEXT      ;CALCULATE EXT EXPONENT
035142  000207                  1$:      RTS     PC               ;RETURN

                                ;;*******************************************************************
                                .SBTTL   FLOATING POINT ADD ROUTINE
                                ;*       THIS ROUTINE ADDS THE CONTENTS OF AC0 TO AC1.
                                ;*       THIS CAN ONLY BE DONE IF THE SOFTWARE EXPONENTS
                                ;*       ARE CLOSE ENOUGH TOGETHER SUCH THAT AN ADJUSTMENT
                                ;*       OF THE REAL EXPONENT LEAVES A NON-ZERO NUMBER.
                                ;;*******************************************************************
035144  010667  000134          FLTSUB:  MOV     SP,SUBFLG        ;SET SUBTRACT FLAG
035150  023737  001410  001412  FLTADD:  CMP     @#$AC0,@#$AC1    ;CHECK SOFTWARE EXPONENTS
035156  003016                           BGT     1$
035160  001434                           BEQ     2$
                                ;ACCUMULATOR 1 IS LARGER THAN ACCUMULATOR 0
035162  013702  001412                   MOV     @#$AC1,R2        ;GET OPERAND B SOFTWARE EXP
035166  163702  001410                   SUB     @#$AC0,R2        ;GET DIFFERENCE IN SOFTWARE EXP'S
035172  020227  000071                   CMP     R2,#57.          ;EXP WITHIN DBL PREC RANGE?
035176  002003                           BGE     7$               ;BRANCH IF ADD NOT REQUIRED
                                                                  ;RESULT IS OPERAND B
035200  005402                           NEG     R2
035202  176402                           LDEXP   R2,AC0           ;RELOAD THE EXPONENT
035204  000422                           BR      2$
035206  176427  177703          7$:      LDEXP   #-75,AC0         ;FAKE EXPONENT SO HARDWARE
035212  000417                           BR      2$               ;WILL DETECT OUT OF RANGE
                                ;
                                ;ACCUMULATOR 0 IS LARGER THAN ACCUMULATOR 1
035214  013702  001410          1$:      MOV     @#$AC0,R2        ;GET SOFTWARE EXP OF OPERAND A
035220  163702  001412                   SUB     @#$AC1,R2        ;GET DIFFERENCE IN EXP'S
035224  013737  001410  001412           MOV     @#$AC0,@#$AC1    ;MAKE SOFTWARE EXP'S EQUAL
035232  020227  000071                   CMP     R2,#57. ;EXP WITHIN DBL PREC RANGE?
035236  002003                           BGE     4$               ;BRANCH IF NO
035240  005402                           NEG     R2
035242  176502                           LDEXP   R2,AC1           ;RELOAD THE EXPONENT
035244  000402                           BR      2$
                                ;
                                ;ACCUMULATOR 0 IS MUCH LARGER THAN ACCUMULATOR 1 SO RESULT IS 0
035246  176527  177703          4$:      LDEXP   #-75,AC1         ;FAKE EXPONENT SO HARDWARE
                                                                  ;WILL DETECT OUT OF RANGE
035252  005767  000026          2$:      TST     SUBFLG           ;ADD OR SUBTRACT?
035256  001402                           BEQ     5$               ;BRANCH IF ADD
035260  173100                           SUBF    AC0,AC1
035262  000401                           BR      6$
035264  172100                  5$:      ADDF    AC0,AC1          ;EXECUTE THE ADD
035266  012746  100400          6$:      MOV     #100400,-(SP)    ;PUT CONTROL WORD ON STACK
035272  004737  053210                   JSR     PC,@#EXPEXT      ;CALCULATE EXT EXPONENT
035276  005067  000002          3$:      CLR     SUBFLG           ;INIT SUBTRACT FLAG
035302  000207                           RTS     PC               ;RETURN
035304  000000                  SUBFLG:  .WORD
035306  000004                  RELEB:   SCOPE
035310  010702                           MOV     PC,R2
035312  062702  000012                   ADD     #12,R2
035316  012707  036574                   MOV     #RELOC,PC        ;GO RELOCATE PROGRAM CODE
035322  000000                  RELE9:   .WORD   0
                                ;88888888888888 LAST ADDRESS OF CODE TO BE RELOCATED 88888888888
```

```
                                        ;;*****************************************************************
                                        ;*TEST 105      TELETYPE AND CLOCK TESTS
                                        ;;*****************************************************************
 035324                                 TST105:
 035324  000240                                 NOP
 035326  112737  000105  001202                 MOVB    #105,@#$TSTNM
 035334  113777  001202  143700                 MOVB    @#$TSTNM,@SWR
 035342  005037  001550                 TTYCHK: CLR     @#FACTOR
 035346  012704  000100                         MOV     #100,R4         ;SET R4 = CONSTANT 100
 035352  032737  000400  001532                 BIT     #TTOPT,@#OPT.CP ;BRANCH IF TTY
 035360  001002                                 BNE     1$              ;ON SYSTEM
 035362  000167  000216                         JMP     ARBFIN          ;JUMP IF NOT
 035366  122777  000200  143656 1$:             CMPB    #200,@$TPS      ;CHECK IF TTY IS READY
 035374  001374                                 BNE     1$
 035376  012737  001567  001276                 MOV     #NULLS-1,@#$REG5;SET ADDRESS OF ASCII STRING TO TYPE
 035404  106277  143642                         ASRB    @$TPS           ;SET IE BIT. SEE TPISR FOR INT SERVICE.
 035410  000001                                 WAIT                    ;WAIT FOR INTERRUPT


 035412                                 DUMMY:
                                        ;ROUTINE TO CHECK PRIORITY ARBITRATION LOGIC
                                        ;THE BELOW TEST WILL INHIBIT INTERRUPTS ON LEVEL 6 AND BELOW (LOCKING
                                        ;OUT THE LINE CLOCK) AND THEN SET UP THE TTY TO INTERRUPT. NEXT THE
                                        ;PRIORITY LEVEL WILL BE SET TO 0 ALLOWING INTERRUPTS IN WHICH CASE
                                        ;THE LINE CLOCK (AT LEVEL 6) SHOULD INTERRUPT BEFORE THE TTY (AT LEVEL 4).
```

```
035412  132737  000020  177776  1$:     BITB    #20,@#PSW
035420  001072                          BNE     ARBEX                   ;EXIT TEST IF 'T' BIT SET
035422  030477  143624          2$:     BIT     R4,@$TPS                ;WAIT FOR TTY TO BE NOT
035426  001375                          BNE     2$                      ;BUSY
035430  112737  000300  177776          MOVB    #300,@#PSW              ;SET PRIORITY LEVEL 6
035436  150477  143610          3$:     BISB    R4,@$TPS                ;SET IE BIT
035442  100375                          BPL     3$                      ;AND WAIT FOR READY
035444  032737  001000  001532          BIT     #LKOPT,@#OPT.CP         ;LINE CLOCK AVAILABLE?
035452  001450                          BEQ     ARBFIN                  ;BRANCH IF NO
035454  012737  035546  000064          MOV     #7$,@#TPVEC             ;SET TTY VECTOR
035462  012737  035560  000100          MOV     #8$,@#LKVEC             ;SET CLOCK VECTORS
035470  012737  000340  000102          MOV     #PR7,@#LKVEC+2
035476  005027                          CLR     (PC)+                   ;CLEAR CHECK WORD
035500  000000                  4$:     .WORD   0
035502  000240                          NOP
035504  000240                          NOP
035506  000240                          NOP
035510  010437  177546                  MOV     R4,@#LKS
035514  113700                  5$:     MOVB    @(PC)+,R0               ;GET CLOCK STATUS & BRANCH IF READY
035516  177546                  6$:     .WORD   LKS                     ;CONTAINS ADDRESS OF L CLOCK STAT
035520  100375                          BPL     5$
035522  000240                          NOP                             ;AT THIS TIME BOTH THE CLOCK
                                                                        ;ARE READY TO INTERRUPT
035524  105037  177776                  CLRB    @#PSW                   ;SET PRIORITY LEVEL 0
                                ;A CLOCK INTERRUPT WILL OCCUR (8$) AND LOC 4$ WILL BE INCREMENTED
                                ;AFTER THE CLOCK SERVICE A TTY INTERRUPT WILL OCCUR. THE TTY INT SERV
                                ;ICE WILL SHIFT LEFT 4$.

035530  000240                          NOP                             ;LEAVE TIME FOR INTERRUPTS
035532  022767  000002  177740          CMP     #2,4$                   ;CHECK THAT THE CLOCK
035540  001415                          BEQ     ARBFIN                  ;& TTY INTERRUPTED IN
035542  104000                          HLT                             ;THE PROPER SEQUENCE
035544  000413                          BR      ARBFIN

035546  005077  143500          7$:     CLR     @$TPS                   ;CLEAR IE BIT
035552  006367  177722                  ASL     4$                      ;SHIFT INDICATOR
035556  000002                          RTI                             ;RETURN

035560  005267  177714          8$:     INC     4$
035564  012737  046630  000100          MOV     #LKSRV,@#LKVEC          ;SET CLOCK VECTORS
035572  000002                          RTI


035574  012737  054364  000064  ARBFIN: MOV     #TPISR,@#TPVEC          ;RESTORE TTY VECTOR
035602  005077  143444                  CLR     @$TPS                   ;CLEAR IE BIT
035606                          ARBEX:
                                ;;*****************************************************************
                                ;*TEST 106      TURN ON UBE AND MBT
                                ;*      TURN ON THE MASS BUS TESTER AND UNIBUS EXERCISER IF PRESENT
                                ;;*****************************************************************
035606                          TST106:
035606  000240                          NOP
035610  112737  000106  001202          MOVB    #106,@#$TSTNM
035616  113777  001202  143416          MOVB    @#$TSTNM,@SWR

                                ;FIRST SETUP SOME MEM. MGMT. REGISTERS
```

```
035624  012700  077406                  MOV     #77406,R0       ;SET CONSTANT=R/W UP 4K WORDS
035630  010037  172300                  MOV     R0,@#KIPDR0     ;SET KIPDR0,1,2,3,& 7 R/W UP 4K WORDS
035634  010037  172302                  MOV     R0,@#KIPDR1
035640  010037  172304                  MOV     R0,@#KIPDR2
03 644  010037  172306                  MOV     R0,@#KIPDR3
035650  010037  172310                  MOV     R0,@#KIPDR4
035654  010037  172312                  MOV     R0,@#KIPDR5
035660  010037  172316                  MOV     R0,@#KIPDR7
035664  005037  172340                  CLR     @#KIPAR0
035670  012737  000200  172342          MOV     #200,@#KIPAR1
035676  012737  000400  172344          MOV     #400,@#KIPAR2
035704  012737  007600  172356          MOV     #7600,@#KIPAR7
035712  005737  177776                  TST     @#PSW           ;ARE WE IN USER MODE?
035716  100464                          BMI     11$             ;BRANCH IF YES
035720  005737  001624                  TST     @#MXMMLO        ;HAS MXMMLO BEEN FOUND YET?
035724  001061                          BNE     11$             ;BRANCH IF YES
035726  012737  000001  177572          MOV     #1,@#SR0
                                ;GET ADDRESS OF LAST MEMORY LOCATION ON THE SYSTEM
035734  012737  000600  172346          MOV     #600,@#KIPAR3   ;INITIALLY MAP PAR3 TO 12K
035742  012737  036006  000004          MOV     #10$,@#ERRVEC   ;EXIT TO 10$ ON FIRST TIMEOUT
035750  005037  000006                  CLR     @#ERRVEC+2      ;BE SURE IN KERNEL MODE
035754  012700  060000                  MOV     #60000,R0       ;SETUP STARTING VIRTUAL ADDR.
035760  012701  100000                  MOV     #100000,R1      ;SETUP VIRT. ADDR. THAT SELECTS PAR4
035764  005720                  5$:     TST     (R0)+           ;TEST MEMORY LOCATION
035766  020001                          CMP     R0,R1           ;DOES VIRT. ADDR. STILL SELECT PAR3
035770  103775                          BLO     5$              ;BRANCH IF IT DOES
035772  062737  000200  172346          ADD     #200,@#KIPAR3   ;MAP PAR3 UP TO THE NEXT 4K
036000  012700  060000                  MOV     #60000,R0       ;RESET R0
036004  000767                          BR      5$              ;CONTINUE TESTING EACH VIRT. ADDR.
                                                                ;UNTIL A TIMEOUT OCCURS
036006  062706  000004          10$:    ADD     #4,SP           ;CLEANUP THE STACK
036012  162700  000002                  SUB     #2,R0           ;GET VIRT. ADDR. OF "TIMEOUT" LOC.
036016  010003                          MOV     R0,R3           ;MOVE VIRT. ADDR. INTO R3
036020  072327  177772                  ASH     #-6,R3          ;SHIFT VIRT. ADDR. RIGHT SIX BITS
036024  042703  177600                  BIC     #177600,R3      ;CLEAR OFF THE TOP 9 BITS
036030  063703  172346                  ADD     @#KIPAR3,R3     ;ADD THE PAF TO THE BLOCK NO.
036034  005002                          CLR     R2              ;CLEAR R2
036036  073227  000006                  ASHC    #6,R2           ;FORM TOP 12 BITS OF PHYSICAL ADDR.
036042  042702  177774                  BIC     #177774,R2      ;CLEAR ALL BUT BITS 0 & 1 IN R2
036046  042700  177700                  BIC     #177700,R0      ;CLEAR BITS <15:06> IN VIRT. ADDR.
036052  050003                          BIS     R0,R3           ;SET BITS <05:00> IN PHYSICAL ADDR.
036054  010237  001622                  MOV     R2,@#MXMMHI     ;SAVE BITS 16 & 17 OF TIMEOUT ADDR.
036060  010337  001624                  MOV     R3,@#MXMMLO     ;SAVE BITS 15-0 OF TIMFOUT ADDR.
036064  005037  177572                  CLR     @#SR0           ;TURN MEM. MGMT. OFF AGAIN

036070  032737  001000  001532  11$:    BIT     #LKOPT,@#OPT.CP         ;BRANCH IF NOT AVAIL
036076  001411                          BEQ     UBESET
036100  012737  046630  000100          MOV     #LKSRV,@#LKVEC
036106  012737  000340  000102          MOV     #PR7,@#LKVEC+2
036114  052737  000100  177546          BIS     #100,@#LKS              ;SET IE BIT

                                ;;*********************************************************
                                ;TURN ON THE UNIBUS EXERCISER IF PRESENT
036122  105737  001532          UBESET: TSTB    @#OPT.CP        ;IS UBE OPTION AVAILABLE?
036126  100023                          BPL     MBTSET          ;BRANCH IF NO
036130  032777  010000  143104          BIT     #SW12,@SWR      ;INHIBIT UBE?
```

```
036136  001017                           BNE     MBTSET            ;BRANCH IF YES
036140  122737  000060  001574           CMPB    #60,@#SUBPASS     ;FIRST SUB-PASS?
036146  001056                           BNE     STMM              ;BRANCH IF NO
036150  105737  001200                   TSTB    @#$PASS           ;FIRST PASS?
036154  001053                           BNE     STMM              ;BRANCH IF NO
036156  105737  001545                   TSTB    @#MMON            ;MEM MGMT ON?
036162  001050                           BNE     STMM              ;BRANCH IF YES
036164  004737  053646                   JSR     PC,@#UBEINIT      ;INITIALIZE UBE
036170  012772  064545  000000           MOV     #64545,@(R2)      ;START UBE

                                 ;;*******************************************************
                                 ;TURN ON THE MASS BUS TESTER IF PRESENT
036176  032737  002000  001532   MBTSET: BIT     #MBTOPT,@#OPT.CP          ;IS MBT AVAILABLE?
036204  001437                           BEQ     STMM              ;BRANCH IF NO
036206  032777  000010  143026           BIT     #SW3,@SWR         ;INHIBIT MBT?
036214  001033                           BNE     STMM              ;BRANCH IF YES
036216  122737  000060  001574           CMPB    r60,@#SUBPASS     ;FIRST SUB-PASS?
036224  001027                           BNE     STMM              ;BRANCH IF NO
036226  105737  001200                   TSTB    @#$PASS           ;FIRST PASS?
036232  001024                           BNE     STMM              ;BRANCH IF NO
036234  105737  001545                   TSTB    @#MMON            ;MEM MGMT ON?
036240  001021                           BNE     STMM              ;BRANCH IF YES
036242  052777  000047  144064   MBT1:   BIS     #47,@MBTTBL+12            ;CLEAR THE MBT
036250  012777  000007  144056           MOV     #7,@MBTTBL+12             ;SELECT UNIT 7
036256  005077  144042                   CLR     @MBTTBL+2                 ;CLEAR THE WORD COUNT
036262  012777  046304  144054           MOV     #MBTSRV,@MBTTBL+22        ;SETUP INTERRUPT VECTOR
036270  012777  000240  144050           MOV     #PR5,@MBTTBL+24           ;SET VECTOR PSW
036276  112777  000161  144016           MOVB    #161,@MBTTBL              ;START MBT

                                 ;;*******************************************************
                                 .SBTTL  STMM ROUTINE
                                 ;ROUTINE TO SET UP MEMORY MANAGEMENT TO RELOCATE PROGRAM CODE ABOVE 16K
                                 ;CHECK IF PROGRAM IS TO BE RELOCATED.
                                 ;SW6=1=NO RELOCATION
                                 ;;*******************************************************
036304  032777  000100  142730   STMM:   BIT     #SW6,@SWR         ;RELOCATION DISABLED?
036312  001402                           BEQ     3$                ;BRANCH IF NO
036314  000167  002266                   JMP     ENDM

                                 ;THE PROGRAM IS GOING TO RELOCATE.
                                 ;RELOCATION WILL BE PERFORMED IN KERNEL MODE WITH PSW SET AT PRIORITY
                                 ;LEVEL 4 (TO PREVENT TTY INTERRUPT-WHICH CHANGES DATA IN PROGRAM)
                                 ;THE 'T' BIT IS CLEARED (IF SET). AFTER THE DATA HAS BEEN WRITTEN IT IS
                                 ;VERIFIED BEFORE EXECUTION.
036320  013727  177776           3$:     MOV     @#PSW,(PC)+       ;SAVE CURRENT PSW
036324  000000                   OLDPSW: .WORD   0
036326  012737  000200  177776           MOV     #PR4,@#PSW
036334  004767  015552                   JSR     PC,CLRTBIT        ;GO CLEAR 'T' BIT IF SET

                                 ;NOW SETUP MEMORY MANAGEMENT REGISTERS.

036340  005037  172340                   CLR     @#KIPAR0          ;NOTE: THESE 2 INSTRUCIONS EFFECTIVELY
036344  012737  000200  172342           MOV     #200,@#KIPAR1     ;RELOCATE PROGRAM EXECUTION
036352  012737  000400  172344           MOV     #400,@#KIPAR2
036360  013737  001562  172346           MOV     @#NEXPAR,@#KIPAR3         ;SET UP KIPAR3 & KIPAR4 & 5
036366  013737  172346  172350           MOV     @#KIPAR3,@#KIPAR4
```

```
036374  062737  000200  172350           ADD     #200,@#KIPAR4
036402  013737  172346  172352           MOV     @#KIPAR3,@#KIPAR5
036410  062737  000400  172352           ADD     #400,@#KIPAR5
036416  012737  007600  172356           MOV     #7600,@#KIPAR7;AND OF COUSE THE I/O PAGE
                                 ;NOW SETUP USER MEM MGMT REGISTERS
036424  012700  077406           1$:     MOV     #77406,R0       ;LOAD R0 WITH VALUE FOR PDRS
036430  010037  177600                   MOV     R0,@#UIPDR0     ;SET UP USER MEM MGMT REGS
036434  010037  177602                   MOV     R0,@#UIPDR1
036440  010037  177604                   MOV     R0,@#UIPDR2
036444  010037  177616                   MOV     R0,@#UIPDR7
036450  016737  143106  177640           MOV     NEXPAR,@#UIPAR0
036456  013737  177640  177642           MOV     @#UIPAR0,@#UIPAR1
036464  062737  000200  177642           ADD     #200,@#UIPAR1
036472  013737  177640  177644           MOV     @#UIPAR0,@#UIPAR2
036500  062737  000400  177644           ADD     #400,@#UIPAR2
036506  013737  172356  177656           MOV     @#KIPAR7,@#UIPAR7

036514  012737  000001  177572           MOV     #1,@#SR0        ;ENABLE MEM MGMT
036522  110637  001545                   MOVB    SP,@#MMON       ;SET MEM MGMT ON IND = ON
036526  013767  001202  021202           MOV     @#$TSTNM,ENDTAG ;LOAD LAST WORD XFERED
036534  012737  040604  000004   RETRY:  MOV     #ENDMEM,@#ERRVEC;SET TIME OUT TRAP VECTOR
036542  005037  000006                   CLR     @#ERRVEC+2
036546  012702  060000                   MOV     #60000,R2       ;SETUP GENERAL REGISTERS
036552  005000                           CLR     R0              ;DATA WILL BE RELOCATED FROM
                                                                 ;ADDRESS IN R0 TO ADDRESS IN R2
036554  012703  137776                   MOV     #137776,R3      ;GET 12K WORDS TO RELOCATE
036560  010013                           MOV     R0,(R3)         ;TRAP TO ENDMEM IF INSUFFICIENT MEMORY
036562  012737  055064  000004           MOV     #ERPRT,@#ERRVEC ;RESTORE ERROR TRAP VECTOR
036570  000137  036770                   JMP     @#IOMON
                                 ;;*****************************************************************
                                 .SBTTL  RELOCATION ROUTINE
                                 ;*      THIS ROUTINE IS USED TO RELOCATE THE 9 SUBTESTS UP TO 28K.
                                 ;*      IF RELOCATION BY AN I/O DEVICE IS SELEC ED, CONTROL IS PASSED
                                 ;*      TO THE I/O MONITOR.
                                 ;*      ENTER WITH:
                                 ;*              FRSTAD=PHYSICAL ADDRESS OF FIRST CODE
                                 ;*              FACTOR=NUMBER OF BYTES ABOVE BASE CODE
                                 ;*              R2    =LAST PHYSICAL ADDRESS OF THE SECTION
                                 ;*      EXIT TO I/O MONITOR WITH:
                                 ;*              OLDBASE=FIRST PHYSICAL ADDRESS TO BE RELOCATED
                                 ;*              NWBASL =FIRST PHYSICAL ADDRESS TO RELOCATE TO
                                 ;*              IOWC   =TWO'S COMPLIMENT WORD COUNT
                                 ;;*****************************************************************
036574  032777  000100  142440   RELOC:  BIT     #SW6,@SWR               ;IS RELOCATION DISABLED?
036602  001067                           BNE     EXITRE                  ;BRANCH IF YES
036604  105737  001545                   TSTB    @#MMON                  ;IS MEMORY MGMT ON?
036610  001064                           BNE     EXITRE                  ;BRANCH IF YES
036612  013700  001554                   MOV     @#FRSTAD,R0             ;GET FIRST ADDRESS TO BE RELOCATED
036616  010005                           MOV     R0,R5
                                 ;LAST ADDRESS IS IN R2
036620  010203                           MOV     R2,R3                   ;SAVE LAST ADDRESS
036622  010204                           MOV     R2,R4
036624  160004                           SUB     R0,R4                   ;R4 NOW HAS BYTE COUNT
036626  010437  001552                   MOV     R4,@#$FACTOR            ;SAVE BYTE COUNT
036632  005737  001550                   TST     @#FACTOR                ;FIRST RELOC IS TO ENDTAG+2
036636  001004                           BNE     1$                      ;BRANCH IF NOT EXECUTING BASE CODE
```

```
036640  010237  036766                   MOV     R2,@#RETPC             ;SAVE RETURN PC TO NEXT SECTION
036644  013702  001556                   MOV     @#FRSTMEM,R2           ;GET FIRST ADDRESS TO RELOCATE TO
036650  060204                   1$:     ADD     R2,R4                  ;R4 NOW CONTAINS LAST MEM ADDRESS
036652  020437  001560                   CMP     R4,@#LSTMEM            ;ENOUGH MEMORY?
036656  101042                           BHI     NOMEM                  ;BRANCH IF NO
036660  160204                           SUB     R2,R4                  ;R4 NOW HAS BYTE COUNT
036662  005037  001550                   CLR     @#FACTOR
036666  032777  000400  142346           BIT     #SW8,@SWR              ;INHIBIT RELOC BY I/O DEVICE?
036674  001014                           BNE     RELNIO                 ;BRANCH IF YES
036676  010037  001602                   MOV     R0,@#OLDBASE           ;SAVE START ADDRESS
036702  010237  001604                   MOV     R2,@#NWBASL            ;SAVE NEW BASE ADDRESS
036706  005037  001606                   CLR     @#NWBASH               ;
036712  006204                           ASR     R4                     ;MAKE IT A WORD COUNT
036714  005404                           NEG     R4                     ;GET TWO'S COMPLIMENT
036716  010437  001610                   MOV     R4,@#IOWC              ;SAVE R4 AS WORDCOUNT
036722  000167  000120                   JMP     ENTER2                 ;GO TO I/O MONITOR
                                 ;RELOCATE BY CPU-MEMORY MANAGEMENT OFF
036726  012022                   RELNIO: MOV     (R0)+,(R2)+            ;RELOCATE CODE
036730  020003                           CMP     R0,R3                  ;DONE YET?
036732  001375                           BNE     RELNIO                 ;BRANCH IF NO
036734  004737  054026                   JSR     PC,@#CHKDAT            ;GO CHECK DATA
036740  102010                           BVC     EXITRE
036742  010037  001304                   MOV     R0,@#$TMP0             ;SAVE R0 FOR TYPEOUT
036746  010237  001536                   MOV     R2,@#VADR              ;SAVE R2
036752  004737  053724                   JSR     PC,@#CNVADR            ;CONVERT R2 TO A PHYSICAL ADR
036756  104006                           ERROR   6
036760  000401                           BR      NOMEM
036762  010207                   EXITRE: MOV     R2,PC                  ;GO EXECUTE RELOCATED CODE
036764  011707                   NOMEM:  MOV     (PC),PC                ;GO TO NEXT SECTION
036766  000000                   RETPC:  .WORD   0                      ;CONTAINS PC OF NEXT SECTION
                                 ;;*************************************************************
                                 .SBTTL  I/O RELOCATION MONITOR
                                 ;*      THIS ROUTINE IS USED TO SCHEDULE I/O DEVICES FOR SUBTEST
                                 ;*      RELOCATION AND PROGRAM RELOCATION. THE I/O DEVICE UNIT
                                 ;*      NUMBER IS DETERMINED, THE BUS ADDRESS CALCULATED, THE WORD
                                 ;*      COUNT CALCULATED AND PASSED TO THE DEVICE HANDLER.
                                 ;;*************************************************************
036770  012737  036776  001212   IOMON:  MOV     #1$,@#$LPERR   ;SETUP ERROR LOOP
036776  005037  001602           1$:     CLR     @#OLDBASE
037002  013705  172346                   MOV     @#KIPAR3,R5            ;SETUP R4 AND R5
037006  005004                           CLR     R4                     ;TO FORM 18 BIT ADDRESS
037010  073427  000006                   ASHC    #6,R4                  ;FORM 18 BIT ADDRESS
037014  010537  001604                   MOV     R5,@#NWBASL            ;SAVE LOWER 16 BITS
037020  010437  001606                   MOV     R4,@#NWBASH            ;SAVE UPPER 2 BITS
037024  032777  000400  142210           BIT     #SW8,@SWR              ;RELOCATE VIA I/O?
037032  001402                           BEQ     2$                     ;BRANCH IF YES
037034  000167  001374                   JMP     RELOCP                 ;GO RELOCATE VIA CP
037040  012737  174000  001610   2$:     MOV     #174000,@#IOWC         ;SET WORD COUNT TO 2K
037046  005037  001310           ENTER2: CLR     @#$TMP2
037052  012737  177776  001620           MOV     #-2,@#RNTBINX          ;SETUP RUN TABLE INDEX
037060  005037  001304                   CLR     @#$TMP0
037064  005002                           CLR     R2                     ;CLEAR LEGAL DEV FLAG
037066  032777  000040  142146   41$:    BIT     #SW5,@SWR              ;INHIBIT ROUND ROBIN?
037074  001416                           BEQ     50$                    ;BRANCH IF NO
037076  005737  001304                   TST     @#$TMP0                ;FLAG SET?
037102  001027                           BNE     43$                    ;BRANCH IF YES
```

```
037104  117737  142132  001614          MOVB    @SWR,@#DEVINDX           ;GET DEVICE FROM SWITCHES
037112  042737  177770  001614          BIC     #177770,@#DEVINDX        ;MASK LOWER 3 BITS
037120  006337  001614                  ASL     @#DEVINDX                ;ADJUST FOR WORD INDEX
037124  005237  001304                  INC     @#$TMP0                  ;SET FLAG
037130  000414                          BR      43$                      ;CONTINUE
037132  012705  000010          50$:    MOV     #10,R5                   ;SET SOB COUNT
037136  022737  000016  001614  40$:    CMP     #16,@#DEVINDX            ;LAST DEVICE YET?
037144  001003                          BNE     42$                      ;BRANCH IF NO
037146  012737  177776  001614  48$:    MOV     #-2,@#DEVINDX            ;INIT DEVICE INDEX
037154  062737  000002  001614  42$:    ADD     #2,@#DEVINDX             ;INCREMENT INDEX
037162  013703  001614          43$:    MOV     @#DEVINDX,R3             ;GET INDEX
037166  012737  000401  001306          MOV     #401,@#$TMP1             ;INIT UNIT MASK
037174  012704  000010                  MOV     #10,R4          ;SET SOB COUNT
037200  133763  001306  001646  44$:    BITB    @#$TMP1,SYSSIZE(R3)      ;IS THIS UNIT EXISTENT?
037206  001405                          BEQ     52$                      ;BRANCH IF NO
037210  005202                          INC     R2                       ;SET LEGAL DEVICE FLAG
037212  133763  001307  001647          BITB    @#$TMP1+1,SYSSIZE+1(R3)  ;HAS IT BEEN USED?
037220  001520                          BEQ     11$                      ;BRANCH IF NO
037222  006337  001306          52$:    ASL     @#$TMP1                  ;SELECT NEXT UNIT
037226  077414                          SOB     R4,44$                   ;CONTINUE
037230  005737  001304                  TST     @#$TMP0                  ;INHIBIT ROUND ROBIN?
037234  001013                          BNE     45$                      ;BRANCH IF YES
037236  077541                          SOB     R5,40$                   ;CONTINUE
037240  005702                          TST     R2                       ;ANY DEVICES AT ALL?
037242  001442                          BEQ     46$                      ;BRANCH IF NO
037244  012704  000010                  MOV     #10,R4                   ;SET SOB COUNT
037250  012701  001647                  MOV     #SYSSIZE+1,R1            ;GET ADR OF SIZE TABLE
037254  105021                  47$:    CLRB    (R1)+                    ;CLEAR ALL USED BITS
037256  005201                          INC     R1                       ;IN ALL DEVICES
037260  077403                          SOB     R4,47$                   ;CONTINUE
037262  000701                          BR      41$
037264  005702                  45$:    TST     R2                       ;WAS IT A LEGAL DEVICE?
037266  001403                          BEQ     49$                      ;BRANCH IF NO
037270  105063  001647                  CLRB    SYSSIZE+1(R3)            ;CLEAR ALL USED BITS THIS DEV
037274  000732                          BR      43$
037276  010367  000016          49$:    MOV     R3,60$                   ;
037302  062767  055236  000010          ADD     #MSGINX,60$              ;GEN MESSAGE ADR
037310  017767  000004  000002          MOV     @60$,60$
037316  104401                          TYPE
037320  000000                  60$:    .WORD
037322  104401  037330                  TYPE    ,65$            ;;TYPE ASCIZ STRING
037326  000407                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /UNAVAILABLE/<CRLF>
037346                          64$:
037346  000637                          BR      ENTER2
037350  105737  001546          46$:    TSTB    @#QV                     ;ACT11?
037354  001016                          BNE     51$                      ;BRANCH IF YES
037356  005227  177777                  INC     #-1
037362  001013                          BNE     51$
037364  104401  037372                  TYPE    ,67$            ;;TYPE ASCIZ STRING
037370  000410                          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  ?NO I/O DEVICES?<CRLF>
037412                          66$:
037412  105737  001545          51$:    TSTB    @#MMON                   ;MGMT ON?
037416  001012                          BNE     61$                      ;BRANCH IF YES
037420  013700  001602                  MOV     @#OLDBASE,R0             ;RESTORE R0
```

```
037424  013702  001604                       MOV     @#NWBASL,R2                      ;RESTORE R2
037430  013703  001552                       MOV     @#$FACTOR,R3                     ;GET RELOCATION FACTOR
037434  060003                               ADD     R0,R3                            ;FORM LAST ADDRESS
037436  010005                               MOV     R0,R5                            ;SETUP R5
037440  000167  177262                       JMP     RELNIO                           ;GO RELOCATE WITH CP
037444  012702  060000               61$:    MOV     #60000,R2                        ;SETUP REGISTERS
037450  012703  137776                       MOV     #137776,R3                       ;WITH FROM
037454  005000                               CLR     R0                               ;AND TO ADDRESS
037456  000137  040434                       JMP     @#RELOCP                         ;RELOCATE VIA CP
037462  105763  001752               11$:    TSTB    RP3HSTAT(R3)                     ;IS HANDLER BUSY?
037466  100405                               BMI     8$                               ;BRANCH IF NO
037470  005737  001304                       TST     @#$TMP0                          ;ROUND ROBIN?
037474  001372                               BNE     11$                              ;BRANCH IF NO
037476  000167  177364                       JMP     41$
037502  005763  001752               8$:     TST     RP3HSTAT(R3)                     ;DID HANDLER FAIL?
037506  100005                               BPL     62$                              ;BRANCH IF NO
037510  005737  001304                       TST     @#$TMP0                          ;ROUND ROBIN
037514  001402                               BEQ     62$                              ;BRANCH IF YES
037516  000137  040414                       JMP     @#15$
037522  153763  001307  001647       62$:    BISB    @#$TMP1+1,SYSSIZE+1(R3)          ;SET UNIT USED BIT
037530  005002                               CLR     R2
037532  006037  001306               30$:    ROR     @#$TMP1                          ;ENCODE THE BIT POSITION
037536  005202                               INC     R2                               ;INTO A UNIT NUMBER
037540  103374                               BCC     30$
037542  005302                               DEC     R2
037544  010237  001616                       MOV     R2,@#UNITNO                      ;SAVE UNIT NUMBER
037550  013763  001610  001772       10$:    MOV     @#IOWC,RP34WC(R3)                ;GIVE WORD COUNT TO HANDLER
037556  010304                               MOV     R3,R4                            ;
037560  072427  000003                       ASH     #3,R4                            ;ENCODE DEVICE FOR RUNTABLE
037564  053704  001616                       BIS     @#UNITNO,R4                      ;ENCODE UNIT NUMBER
037570  006304                               ASL     R4
037572  062737  000002  001620               ADD     #2,@#RNTBINX                     ;INCREMENT RUN TABLE INDEX
037600  013702  001620                       MOV     @#RNTBINX,R2                     ;GET RUN TABLE INDEX
037604  110462  001667                       MOVB    R4,RUNTBL+1(R2)                  ;ENTER DEV & UNIT IN TABLE
037610  013763  001616  002066               MOV     @#UNITNO,RP3UNIT(R3)             ;GIVE HANDLER UNIT NUMBER
037616  012737  000240  000012               MOV     #PRS,@#RESVEC+2                  ;SETUP RESERVED VECTOR PSW
037624  016337  002102  000010               MOV     RP3HANA(R3),@#RESVEC             ;SETUP RESERVED VECTOR
037632  006303                               ASL     R3                               ;ADJUST INDEX
037634  013763  001602  002006               MOV     @#OLDBASE,RP3OLD(R3)             ;GIVE HANDLER OLD BASE ADDRESS
037642  013763  001604  002036               MOV     @#NWBASL,RP3NWL(R3)              ;GIVE HANDLER
037650  013763  001606  002040               MOV     @#NWBASH,RP3NWH(R3)              ;NEW BASE ADDRESS
037656  005063  002010                       CLR     RP3OLD+2(R3)                     ;ENSURE OLD BASE HIGH IS CLR
037662  000010                               CALLHANDLER
037664  105737  001545                       TSTB    @#MMON                           ;IS MEMORY MANAGEMENT ON?
037670  001416                               BEQ     13$                              ;BRANCH IF NO
037672  022737  000012  001620               CMP     #12,@#RNTBINX                    ;TRANSFERED 12K YET?
037700  001412                               BEQ     13$                              ;BRANCH IF YES
037702  062737  010000  001602               ADD     #10000,@#OLDBASE                 ;ADD 2K
037710  062737  010000  001604               ADD     #10000,@#NWBASL                  ;TO BASE
037716  005537  001606                       ADC     @#NWBASH                         ;ADDRESSES
037722  000137  037066                       JMP     @#41$
037726  113705  001645               13$:    MOVB    @#LTICKS+1,R5                    ;GET SECOND COUNT
037732  062705  000002                       ADD     #2,R5                            ;INCREMENT BY TWO
037736  162705  000074                       SUB     #60.,R5                          ;ENSURE RESULT IS 59 OR LESS
037742  100002                               BPL     31$                              ;
037744  062705  000074                       ADD     #60.,R5                          ;COUNT WAS LESS THAN 58-RESTORE
```

```
037750  012700  000010   31$:   MOV     #10,R0                  ;SET SOB COUNT
037754  005002                  CLR     R2
037756  005003                  CLR     R3
037760  005004                  CLR     R4
037762  066203  001752   14$:   ADD     RP3HSTAT(R2),R3         ;ADD ALL THE HANDLER
037766  005504                  ADC     R4                      ;STATUS WORDS. WHEN ALL
037770  062702  000002          ADD     #2,R2                   ;TRANSFERS ARE FINISHED
037774  077006                  SOB     R0,14$                  ;RESULT WILL BE 2000
037776  006103                  ROL     R3                      ;(WITHOUT ROTATE)
040000  005504                  ADC     R4
040002  022703  004000          CMP     #4000,R3                ;ALL DONE?
040006  001406                  BEQ     32$                     ;BRANCH IF YES
040010  123705  001645          CMPB    @#LTICKS+1,R5           ;TWO SECONDS ELAPSED YET?
040014  001355                  BNE     31$                     ;BRANCH IF NO
040016  104015                  ERROR   15                      ;DEVICE HUNG
040020  000177  141166          JMP     @$LPERR                 ;RESTART RELOCATION
040024  005704           32$:   TST     R4                      ;ANY DEVICE ERRORS?
040026  001172                  BNE     15$                     ;BRANCH IF YES
040030  105737  001545          TSTB    @#MMON                  ;MEM MGMT ON?
040034  001012                  BNE     25$                     ;BRANCH IF YES
040036  013705  001602          MOV     @#OLDBASE,R5            ;SETUP R5 FOR DATA CHECK
040042  010500                  MOV     R5,R0
040044  063700  001552          ADD     @#$FACTOR,R0            ;GET LAST ADDRESS
                                                                ;OF GOOD DATA
040050  013702  001604          MOV     @#NWBASL,R2
040054  063702  001552          ADD     @#$FACTOR,R2            ;GET LAST ADDRESS
                                                                ;OF DATA TO BE CHECKED
040060  000406                  BR      22$                     ;CONTINUE
040062  012700  057776   25$:   MOV     #57776,R0               ;GET LAST ADR OF GOOD DATA
040066  012702  137776          MOV     #137776,R2              ;GET LAST ADR OF DATA TO BE CHECKED
040072  012705  002500          MOV     #2500,R5                ;DON'T CHECK FIRST 2000 LOCATIONS
040076  004737  054026   22$:   JSR     PC,@#CHKDAT             ;GO CHECK DATA
040102  102173                  BVC     EXIT                    ;EXIT IF NO ERRORS
040104  005001                  CLR     R1
040106  010037  001304          MOV     R0,@#$TMP0
040112  010237  001536          MOV     R2,@#VADR
040116  010203                  MOV     R2,R3                   ;SAVE ERROR ADDRESS
040120  005004                  CLR     R4
040122  105737  001545          TSTB    @#MMON                  ;IS MEM MGMT ON?
040126  001406                  BEQ     16$                     ;BRANCH IF NO
040130  162703  010000   17$:   SUB     #10000,R3               ;SUBTRACT 2K FROM ERROR ADDRESS
040134  100403                  BMI     16$                     ;BRANCH IF BLOCK IS FOUND
040136  062704  000002          ADD     #2,R4                   ;COUNT ONE MORE BLOCK
040142  000772                  BR      17$                     ;CONTINUE
                         ;R4 NOW CONTAINS INDEX OF ERROR FOR RUN TIME TABLE
040144  116404  001667   16$:   MOVB    RUNTBL+1(R4),R4         ;GET DEVICE THAT FAILED
040150  042704  177400          BIC     #177400,R4              ;ENSURE HIGH BYTE CLEAR
040154  006004                  ROR     R4                      ;THROW AWAY LSB
040156  005005                  CLR     R5                      ;ENSURE R5 CLEAR
040160  073427  177775          ASHC    #-3,R4                  ;GET UNIT NUMBER IN R5
040164  010500                  MOV     R5,R0
040166  072027  177764          ASH     #-14,R0
040172  042700  177770          BIC     #177770,R0
040176  010037  001312          MOV     R0,@#$TMP3
040202  010403                  MOV     R4,R3                   ;AND DEVICE INDEX IN R4 & R3
040204  010337  001310          MOV     R3,@#$TMP2
```

```
040210  012737  000001  001306          MOV     #1,@#$TMP1                      ;ENCODE 3 BIT UNIT NO INTO
040216  162705  020000          19$:    SUB     #20000,R5               ;ONE BIT IN THE LOW BYTE OF $TMP1
04[illegible]2  103403                          BCS     18$                     ;BRANCH IF DONE
04[illegible]4  006137  001306                  ROL     @#$TMP1                 ;SELECT NEXT UNIT
04[illegible]0  000772                          BR      19$                     ;CONTINUE
04[illegible]32 012737  040350  001212  18$:    MOV     #20$,@#$LPERR           ;SETUP LOOP RETURN
04[illegible]0  005701                          TST     R1                      ;DEVICE ERROR?
04[illegible]2  001010                          BNE     100$                    ;BRANCH IF YES
04[illegible]   104010                          ERROR   10                      ;DATA CHECK ERROR
04[illegible]6  105737  001545                  TSTB    @#MMON                  ;MGMT ON?
04[illegible]2  001002                          BNE     70$                     ;BRANCH IF YES
040254  000137  037046          71$:    JMP     @#ENTER2
040260  000137  036770          70$:    JMP     @#IOMON
040264  104007                  100$:   ERROR   7
040266  042763  100000  001752          BIC     #BIT15,RP3HSTAT(R3)     ;CLEAR THE ERROR
040274  022703  000002                  CMP     #2,R3                   ;RK?
040300  002405                          BLT     90$                     ;BRANCH IF RH
040302  003016                          BGT     92$                     ;BRANCH IF RP03
040304  112777  000001  141670          MOVB    #1,@RKCS                ;CLEAR RK
040312  000412                          BR      92$

040314  022703  000012          90$:    CMP     #12,R3                  ;RS?
040320  001004                          BNE     91$                     ;BRANCH IF NO
040322  052777  000040  141740          BIS     #BIT5,@RSCS2            ;CLEAR RS
040330  000403                          BR      92$

040332  052777  000040  141670  91$:    BIS     #BIT5,@RP4CS2           ;CLEAR RP

040340                          92$:
040340  105737  001545                  TSTB    @#MMON                  ;MGMT ON?
040344  001345                          BNE     70$                     ;BRANCH IF YES
040346  000742                          BR      71$
040350  052763  000400  001752  20$:    BIS     #BIT8,RP3HSTAT(R3)      ;SET REPEAT FLAG IN HANDLER
040356  016337  002102  000010          MOV     RP3HANA(R3),@#RESVEC    ;SETUP RESERVED INSTRUCTION VECTOR
040364  000010                          CALLHANDLER
040366  105763  001752          21$:    TSTB    RP3HSTAT(R3)            ;HANDLER FINISHED?
040372  100375                          BPL     21$                     ;BRANCH IF NO
040374  005763  001752                  TST     RP3HSTAT(R3)            ;ANY ERROR?
040400  100714                          BMI     18$                     ;BRANCH IF YES
040402  032777  001000  140632          BIT     #BIT9,@SWR              ;STILL LOOPING?
040410  001357                          BNE     20$                     ;BRANCH IF YES
040412  000725                          BR      100$+2                  ;CONTINUE TEST
                                ;THE FOLLOWING CODE HANDLES DEVICE ERROR ON RELOCATION
040414  005004                  15$:    CLR     R4                      ;SET INDEX
040416  010601                          MOV     SP,R1
040420  005764  001702          24$:    TST     RUNTRAK(R4)             ;SEARCH FOR DEVICE ERROR
040424  100647                          BMI     16$                     ;BRANCH IF ERROR
040426  062704  000002                  ADD     #2,R4                   ;INCREMENT INDEX
040432  000772                          BR      24$                     ;CONTINUE SEARCH
                                ;RELOCATE BY CPU-MEMORY MANAGEMENT ON
040434  012022                  RELOCP: MOV     (R0)+,(R2)+             ;RELOCATE CODE
040436  020302                          CMP     R3,R2                   ;DONE YET?
040440  001375                          BNE     RELOCP                  ;BRANCH IF NO
040442  012705  002500                  MOV     #2500,R5
040446  004737  054026                  JSR     PC,@#CHKDAT             ;CHECK DATA
040452  102007                          BVC     EXIT
```

```
040454 010037 001304                     MOV     R0,@#$TMP0
040460 010237 001536                     MOV     R2,@#VADR
040464 104006                            ERROR   6
040466 000167 176042                     JMP     RETRY
040472 105737 001545             EXIT:   TSTB    @#MMON                  ;MEM MGMT ON?
040476 001002                            BNE     .+6                     ;BRANCH IF YES
040500 000137 036762                     JMP     @#EXITRE
040504 062737 000077 001562              ADD     #77,@#NEXPAR            ;SET VALUE FOR NEXT RELOCATION
040512 013737 172346 172340              MOV     @# IPAR3,@#KIPAR0
040520 013737 172350 172342              MOV     @#KIPAR4,@#KIPAR1
040526 013737 172352 172344              MOV     @#KIPAR5,@#KIPAR2
                                 ;*************************************************************************
                                 ;PROGRAM IS NOW EXECUTING IN KERNEL MODE RELOCATED TO ADDRESS AS SPEC-
                                 ;IFIED IN KIPAR0. FOR EX. IF KIPAR0=1600 THEN PROGRAM EXECUTING AT
                                 ;ADDRESS 160000+(PC)
040534 013700 172340                     MOV     @#KIPAR0,R0     ;GET PAR0
040540 072027 177774                     ASH     #-4,R0          ;GET BITS <11:4> IN LOW BYTE
040544 110037 001203                     MOVB    R0,@#$TSTNM+1   ;PUT IN DSPLAY REG HIGH BYTE
040550 005037 177776                     CLR     @#PSW
040554 012706 001200                     MOV     #KERSTK,SP      ;SET KERNEL STACK PTR
040560 016746 175540                     MOV     OLDPSW,-(SP)    ;RESTORE OLD PSW
040564 012746 005530                     MOV     #LOOP,-(SP)
040570 105737 001544                     TSTB    @#NEXEC         ;BRANCH IF TEST CODE TO
040574 001402                            BEQ     1$              ;BE EXECUTED
040576 012716 036304                     MOV     #STMM,(SP)
040602 000002                    1$:     RTI                     ;RESTART PROGRAM AT LOOP

                                 ;WHEN RELOCATION ABOVE 28K IS COMPLETE PROGRAM TRAPS TO ENDMEM.
040604 022626                    ENDMEM: CMP     (SP)+,(SP)+     ;POP STACK TWICE
040606 005037 177572             ENDM:   CLR     @#SR0           ;DISABLE MEM MGMT

                                 ;*************************************************************************
                                 ;AT THIS TIME A 'SUB-PASS' HAS BEEN COMPLETED.
                                 ;PROGRAM NOW EXECUTING IN KERNEL MODE AT PC AS SHOWN (NO RELOCATION)
040612 012737 000600 001562              MOV     #600,@#NEXPAR   ;RESET NEXT VALUE FOR PAR REGISTERS
040620 005737 003330                     TST     @#PROT
040624 001403                            BEQ     2$
040626 012737 001600 001562              MOV     #1600,@#NEXPAR
040634 105037 001545             2$:     CLRB    @#MMON          ;SET MEM MGMT ON IND = OFF
                                 ;;*************************************************************
                                 .SBTTL  END OF SUB-PASS ROUTINE
                                 ;*      THIS ROUTINE SETS UP THE PSW FROM THE PSW TABLE
                                 ;*      FOR THE NEXT SUB-PASS. IT THEN STARTS THE PRINTER
                                 ;*      (IF NOT ON ACT11) FOR TYPING THE END OF SUB-PASS MESSAGE.
                                 ;;*************************************************************
040640                           END:
040640 012737 055064 000004      END1:   MOV     #ERPRT,@#ERRVEC
040646 005037 177776                     CLR     @#PSW           ;CLEAR MODE BITS IN PSW
040652 004767 013234                     JSR     PC,CLRTBIT      ;GO CLEAR 'T' BIT IF SET
040656 012706 001200                     MOV     #KERSTK,SP      ;SET KERNEL STACK PTR
040662 032737 000400 001532              BIT     #BIT8,@#OPT.CP  ;TTY AVAILABLE?
040670 001404                            BEQ     1$              ;BRANCH IF NO
040672 032777 000100 140352              BIT     #100,@$TPS      ;CHECK IF OUTPUT DEVICE IS BUSY
040700 001374                            BNE     .-6             ;IS AVAILABLE
040702 105237 001574             1$:     INCB    @#SUBPASS
040706 113702 001574                     MOVB    @#SUBPASS,R2
```

```
 040712  162702  000060                   SUB     #60,R2
 040716  022702  000004                   CMP     #4,R2                  ;END OF TEST?
 040722  001007                           BNE     2$              ;BRANCH IF NOT AT END
 040724  012737  000060  001574           MOV     #60,@#SUBPASS          ;INIT SUBPASS COUNT TO ASCII 0
 040732  005046                           CLR     -(SP)
 040734  012746  041036                   MOV     #$EOP,-(SP)
 040740  000002                           RTI
 040742  006302                   2$:     ASL     R2
 040744  012737  001534  000014           MOV     #$RTRN,@#TBITVEC       ;SET 'T' TRAP VECTOR
 040752  012737  001573  001276           MOV     #SUBPASS-1,@#$REGS
 040760  106277  140266                   ASRB    @$TPS
 040764  016246  055202                   MOV     PSWTAB(2),-(SP) ;PUSH NEXT PASS PSW ON STACK
 040770  012746  005530                   MOV     #LOOP,-(SP)     ;RESTART PROGRAM AT LOOP
 040774  105737  001546           3$:     TSTB    @#QV            ;QV PASS?
 041000  001015                           BNE     RTI1            ;BRANCH IF YES
 041002  032737  000400  001532           BIT     #BIT8,@#OPT.CP  ;TTY AVAILABLE?
 041010  001411                           BEQ     RTI1            ;BRANCH IF NO
 041012  122777  000200  140232           CMPB    #200,@$TPS      ;IS PRINTER READY?
 041020  001365                           BNE     3$              ;BRANCH IF NO
 041022  012737  056217  001276           MOV     #MSG20-1,@#$REGS
 041030  106277  140216                   ASRB    @$TPS           ;TYPE END SUBPASS MESSAGE
 041034  000002                   RTI1:   RTI                     ;RESTART PROGRAM AT LOOP WITH NEW PSW
                                                                  ;(FROM TABLE NEAR END OF PROGRAM)

                                  .SBTTL  END OF PASS ROUTINE

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

 041036                           $EOP:
 041036  004737  051010                   JSR     PC,@#TYPTIME
 041042  005067  140134                   CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 041046  005067  140252                   CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 041052  005267  140122                   INC     $PASS           ;;INCREMENT THE PASS NUMBER
 041056  042767  100000  140114           BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 041064  005327                           DEC     (PC)+           ;;LOOP?
 041066  000001                   $EOPCT: .WORD   1
 041070  003063                           BGT     $DOAGN          ;;YES
 041072  012737                           MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 041074  000001                   $ENDCT: .WORD   1
 041076  041066                           $EOPCT
 041100  104401  041106                   TYPE    ,65$            ;;TYPE ASCIZ STRING
 041104  000407                           BR      64$             ;;GET OVER THE ASCIZ
                                  ;;65$:  .ASCIZ  <12><15>/END PASS #/
 041124                           64$:
 041124  016746  140050                   MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                                                  ;;TYPE PASS NUMBER
 041130  104405                           TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
 041132  104401  041140                   TYPE    ,67$            ;;TYPE ASCIZ STRING
 041136  000421                           BR      66$             ;;GET OVER THE ASCIZ
                                  ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
 041202                           66$:
```

```
041202  016746  140006                    MOV     $ERTTL,-(SP)        ;;SAVE $ERTTL FOR TYPEOUT
                                                                      ;;TOTAL NUMBER OF ERRORS
041206  104405                            TYPDS                       ;;GO TYPE--DECIMAL ASCII WITH SIGN
041210  104401  001335                    TYPE    ,$CRLF              ;;TYPE CARRIAGE RETURN, LINE FEED
041214  005067  137774                    CLR     $ERTTL              ;;CLEAR ERROR TOTAL
041220  013700  000042          $GET42:   MOV     @#42,R0             ;;GET MONITOR ADDRESS
041224  001405                            BEQ     $DOAGN              ;;BRANCH IF NO MONITOR
041226  000005                            RESET                       ;;CLEAR THE WORLD
041230  004710                  $ENDAD:   JSR     PC,(R0)             ;;GO TO MONITOR
041232  000240                            NOP                         ;;SAVE ROOM
041234  000240                            NOP                         ;;FOR
041236  000240                            NOP                         ;;ACT11
041240                          $DOAGN:
041240  000137                            JMP     @(PC)+              ;;RETURN
041242  005530                  $RTNAD:   .WORD   LOOP
041244     377     377     000  $ENULL:   .BYTE   -1,-1,0             ;;NULL CHARACTER STRING
        041250                            .EVEN
                                ;;*******************************************************************
                                .SBTTL  RP11/RP03 HANDLER
                                ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
                                ;;*******************************************************************
041250  104406                  RP3DRV:   SAVREG
041252  105037  001752                    CLRB    @#RP3HSTA                 ;CLEAR DONE FLAG
041256  032737  000400  001752            BIT     #BIT8,@#RP3HSTA           ;REPEAT FLAG SET?
041264  001403                            BEQ     8$                        ;BRANCH IF NO
041266  104407                            RESREG
041270  000137  043130                    JMP     @#RP3RPT
041274  013737  001620  001630  8$:       MOV     @#RNTBINX,@#RP311         ;SAVE RUN TABLE INDEX
041302  032777  000020  137732            BIT     #SW4,@SWR                 ;INHIBIT RND DSK ADR?
041310  001403                            BEQ     1$                        ;BRANCH IF NO
041312  005000                            CLR     R0
041314  005001                            CLR     R1
041316  000410                            BR      4$
041320  004737  052752          1$:       JSR     PC,@#$RAND                ;GO GET RANDOM NUMBER
041324  013700  001566                    MOV     @#$HINUM,R0               ;GET HI NUMBER
041330  013701  001564                    MOV     @#$LONUM,R1               ;GET LO NUMBER
041334  073027  177771                    ASHC    #-7,R0                    ;ADJUST TO FORM CYL ADR
041340  042700  177000          4$:       BIC     #177000,R0                ;GET RID OF UNUSED BITS
041344  022700  000624                    CMP     #624,R0                   ;LEGAL CYL?
041350  100003                            BPL     5$                        ;BRANCH IF YES
041352  062700  000624                    ADD     #624,R0                   ;MAKE IT LEGAL
041356  000770                            BR      4$
041360  013702  001630          5$:       MOV     @#RP311,R2                ;GET RUN TABLE INDEX
041364  016203  001666                    MOV     RUNTBL(R2),R3             ;GET DEVICE ID
041370  042703  000777                    BIC     #777,R3                   ;ID ONLY
041374  050300                            BIS     R3,R0                     ;COMBINE WITH CYL ADR
041376  010062  001666                    MOV     R0,RUNTBL(R2)             ;PUT BACK IN TABLE
041402  072127  177775                    ASH     #-3,R1                    ;GEN TRK-SECT ADR
041406  010103                            MOV     R1,R3                     ;SAVE
041410  042701  160377          6$:       BIC     #160377,R1                ;GET RID OF ALL BUT TRK
041414  022701  011400                    CMP     #11400,R1                 ;LEGAL TRAK?
041420  100003                            BPL     2$                        ;BRANCH IF YES
041422  062701  011400                    ADD     #11400,R1                 ;MAKE IT LEGAL
041426  000770                            BR      6$
041430  042703  177760          2$:       BIC     #177760,R3                ;GET SECTOR ADR
041434  022703  000011                    CMP     #11,R3                    ;IS IT LEGAL?
```

```
041440  100003                              BPL     3$                              ;BRANCH IF YES
041442  062703  000011                      ADD     #11,R3                          ;MAKE IT LEGAL
041446  000770                              BR      2$
041450  050301                     3$:      BIS     R3,R1                           ;COMBINE TRK-SECT
041452  010162  001702                      MOV     R1,RUNTRAK(R2)                  ;PUT IN TABLE
041456  010037  002120                      MOV     R0,@#RP3HDC                     ;SAVE DESIRED CYL
041462  010137  002166                      MOV     R1,@#RP3DA                      ;SAVE DSK ADR
041466  112737  177775  002146              MOVB    #-3,@#RP3TRY                    ;INIT TRY COUNT
041474  012737  000103  001626     7$:      MOV     #103,@#RP310                    ;GET FUNCTION
041502  013700  002010                      MOV     @#RP3OLD+2,R0                   ;GET BAE BITS
041506  072027  000004                      ASH     #4,R0                           ;SHIFT TO BITS 4 & 5
041512  050037  001626                      BIS     R0,@#RP310                      ;COMBINE WITH FUNCTION
041516  010037  002010                      MOV     R0,@#RP3OLD+2
041522  013700  002066                      MOV     @#RP3UNIT,R0
041526  072027  000010                      ASH     #10,R0                          ;SHIFT UNIT NO TO RIGHT BITS
041532  050037  001626                      BIS     R0,@#RP310                      ;COMBINE WITH FUNC & BAE
041536  010037  002066                      MOV     R0,@#RP3UNIT
041542  104407                              RESREG
041544  053777  002066  140406     RP3WTRY: BIS     @#RP3UNIT,@RP3CS                        ;SET UNIT BITS
041552  004737  041612                      JSR     PC,@#LDRP3                      ;LOAD RP3 REGISTERS
041556  012777  043160  140406              MOV     #RP3SRV,@RP3VEC                 ;SET VECTOR
041564  005077  140404                      CLR     @RP3PSW
041570  005037  002134                      CLR     @#RP3FUN                        ;SET FUNCTION TO WRITE
041574  005777  140354             1$:      TST     @RP3DS                          ;IS DRIVE READY?
041600  100375                              BPL     1$                              ;BRANCH IF NO
041602  013777  001626  140350              MOV     @#RP310,@RP3CS                  ;LOAD FUNCT AND GO
041610  000002                              RTI                                     ;RETURN
041612  013777  002116  140346     LDRP3:   MOV     @#RP3HDA,@RP3DA                 ;LOAD DSK ADR
041620  013777  002120  140342              MOV     @#RP3HDC,@RP3DC                 ;LOAD CYL ADR
041626  013777  001772  140326              MOV     @#RP3HWC,@RP3WC                 ;LOAD WORD COUNT
041634  013777  002006  140322              MOV     @#RP3OLD,@RP3BA                 ;LOAD BUS ADR
041642  000207                              RTS     PC                              ;RETURN

                                   ;;*******************************************************************
                                   .SBTTL   RK11/RK05 HANDLER
                                   ;*       SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
                                   ;;*******************************************************************
041644  104406                     RKDRV:   SAVREG
041646  105037  001754                      CLRB    @#RKHSTAT                       ;CLEAR DONE FLAG IN HANDLER STAT
041652  032737  000400  001754              BIT     #BIT8,@#RKHSTAT                 ;REPEAT FLAG SET?
041660  001403                              BEQ     5$                              ;BRANCH IF NO
041662  104407                              RESREG
041664  000137  043700                      JMP     @#RKRPT
041670  013737  001620  001634     5$:      MOV     @#RNTBINX,@#RK11                ;SAVE RUN TABLE INDEX
041676  105037  001754                      CLRB    @#RKHSTAT                       ;CLEAR DONE FLAG IN HANDLER STAT
041702  032777  000020  137332              BIT     #SW4,@SWR                       ;RANDOM DSK ADDRESS?
041710  001403                              BEQ     6$                              ;BRANCH IF YES
041712  005000                              CLR     R0                              ;CLEAR REGISTERS
041714  005001                              CLR     R1
041716  000406                              BR      7$                              ;FOR ADDRESS CHECKING
041720  004737  052752             6$:      JSR     PC,@#$RAND                      ;GET RANDOM NUMBER
041724  013700  001566                      MOV     @#$HINUM,R0                     ;GET HIGH NUMBER
041730  013701  001564                      MOV     @#$LONUM,R1                     ;GET LO NUMBER
041734  072027  177775             7$:      ASH     #-3,R0                          ;ADJUST TO FORM
                                                                                    ;CYLINDER ADDRESS
041740  010001                              MOV     R0,R1                           ;SAVE IN R1
```

```
041742 042701 160037          4$:     BIC     #160037,R1            ;GET RID OF SURF-SECT BITS
041746 022701 014300                  CMP     #14300,R1             ;IS IT A LEGAL TRAK?
041752 100003                         BPL     3$                    ;BRANCH IF YES
041754 062701 014340                  ADD     #14340,R1             ;ADD MAXIMUM TRAK
041760 000770                         BR      4$                    ;TRY AGAIN

041762 072127 177773          3$:     ASH     #-5,R1                ;ADJUST CYLINDER ADDRESS
041766 013702 001634                  MOV     @#RK11,R2             ;GET RUN TABLE INDEX
041772 016203 001666          DWORRY: MOV     RUNTBL(R2),R3
041776 042703 000777                  BIC     #777,R3
042002 050103                         BIS     R1,R3
042004 010362 001666                  MOV     R3,RUNTBL(R2)         ;ENTER CYLINDER ADR IN RUN TABLE
042010 072027 177770                  ASH     #-10,R0               ;GENER SECTOR-SURF ADDRESS
042014 042700 177740                  BIC     #177740,R0            ;GET RID OF EXTRA BITS
042020 010003                         MOV     R0,R3                 ;SAVE
042022 042700 000020                  BIC     #BIT4,R0              ;GET RID OF SURFACE BIT
042026 022700 000012                  CMP     #12,R0                ;IS SECTOR ADDRESS LEGAL?
042032 100004                         BPL     1$                    ;BRANCH IF YES
042034 062700 000012                  ADD     #12,R0                ;MAKE IT LEGAL
042040 042700 000020                  BIC     #BIT4,R0              ;GET RID OF CARRY FROM ADD
042044 042703 000017          1$:     BIC     #17,R3                ;GET SURFACE ADDRESS
042050 050300                         BIS     R3,R0                 ;GENER COMP SECT-SURF ADDRESS
042052 010062 001702                  MOV     R0,RUNTRAK(R2)        ;SAVE IN RUN TRAK TABLE
042056 072127 000005                  ASH     #5,R1                 ;ADJUST CYLINDER ADDRESS
042062 050100                         BIS     R1,R0                 ;CONCATINATE TRK & SECT ADDR
042064 013701 002070                  MOV     @#RKUNIT,R1           ;GET UNIT NUMBER
042070 072127 000015                  ASH     #15,R1        ;ADJUST
042074 050100                         BIS     R1,R0                 ;CONCATINATE UNIT,TRK,SURF,SECT
042076 010037 002122                  MOV     R0,@#RKHDA            ;SAVE
042102 112737 177775 002147           MOVB    #-3,@#RKTRY           ;SET RETRY COUNT
042110 012767 000103 137514   2$:     MOV     #103,RK10             ;SET FUNCTION
042116 013700 002014                  MOV     @#RKOLD+2,R0          ;GET BA EXTENDED
042122 072027 000004                  ASH     #4,R0                 ;ADJUST
042126 050037 001632                  BIS     R0,@#RK10             ;PUT IN WITH FUNCTION
042132 010037 002014                  MOV     R0,@#RKOLD+2          ;SAVE IN MEMORY
042136 104407                         RESREG
042140 013777 002122 140042   RKWTRY: MOV     @#RKHDA,@RKDA         ;LOAD DISK ADDRESS
042146 013777 001774 140030           MOV     @#RKHWC,@RKWC         ;LOAD WORD COUNT
042154 013777 002012 140024           MOV     @#RKOLD,@RKBA         ;LOAD BUS ADDRESS
042162 012777 043730 140022           MOV     #RKSRV,@RKVEC         ;LOAD INTERRUPT VECTOR
042170 005077 140020                  CLR     @RKPSW
042174 005037 002136                  CLR     @#RKFUN               ;SET FUNCTION TO WRITE
042200 032777 000100 137770           BIT     #BIT6,@RKDS           ;UNIT READY?
042206 001774                         BEQ     .-6
042210 013777 001632 137764           MOV     @#RK10,@RKCS          ;LOAD FUNCTION AND GO
042216 000006                         RTT                           ;RETURN

                              ;;***************************************************************
                              .SBTTL  RH11/RP04 HANDLER
                              ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
                              ;;***************************************************************
042220 104406                 RP4DRV: SAVREG
042222 105037 001762                  CLRB    @#RP4HSTA             ;CLEAR DONE FLAG
042226 032737 000400 001762           BIT     #BIT8,@#RP4HST        ;REPEAT FLAG SET?
042234 001403                         BEQ     6$                    ;BRANCH IF NO
042236 104407                         RESREG
```

```
042240  000137  044504                  JMP     @#RP4RPT
042244  013737  001620  001636  6$:     MOV     @#RNTBINX,@#RP411       ;SAVE RUN TABLE INDEX
042252  105037  001762                  CLRB    @#RP4HSTA               ;CLEAR DONE FLAG
042256  032777  000020  136756          BIT     #SW4,@SWR               ;RANDOM DSK ADDRESS?
042264  001403                          BEQ     1$                      ;BRANCH IF YES
042266  005000                          CLR     R0
042270  005001                          CLR     R1
042272  000410                          BR      4$
042274  004737  052752          1$:     JSR     PC,@#$RAND              ;GET RANDOM NUMBER
042300  013700  001566                  MOV     @#$HINUM,R0             ;GET HI NUMBER
042304  013701  001564                  MOV     @#$LONUM,R1             ;GET LO NUMBER
042310  073027  177771                  ASHC    #-7,R0                  ;ADJUST TO FORM CYL. ADR.
042314  042700  177000          4$:     BIC     #177000,R0              ;GET RID OF UNUSED BITS
042320  022700  000631                  CMP     #631,R0                 ;LEGAL CYLINDER
042324  100003                          BPL     5$                      ;BRANCH IF YES
042326  062700  000631                  ADD     #631,R0                 ;MAKE IT LEGAL
042332  000770                          BR      4$

042334  013702  001636          5$:     MOV     @#RP411,R2              ;GET RUN TABLE INDEX
042340  016203  001666                  MOV     RUNTBL(R2),R3           ;GET DEVICE ID
042344  042703  000777                  BIC     #777,R3                 ;SAVE ID ONLY
042350  050003                          BIS     R0,R3                   ;COMBINE WITH CYL ADR
042352  010362  001666                  MOV     R3,RUNTBL(R2)           ;PUT IN RUN TABLE
042356  072127  177775                  ASH     #-3,R1                  ;GEN TRAK-SECT ADR
042362  042701  160340                  BIC     #160340,R1              ;GET RID OF UNUSED BITS
042366  010103                          MOV     R1,R3                   ;SAVE
042370  042701  000037                  BIC     #37,R1                  ;GET RID OF SECT BITS
042374  022701  011000                  CMP     #11000,R1               ;LEGAL TRAK?
042400  100004                          BPL     2$                      ;BRANCH IF YES
042402  062701  011000                  ADD     #11000,R1               ;MAKE IT LEGAL
042406  042701  020000                  BIC     #BIT13,R1               ;GET RID OF ADD CARRY
042412  042703  177740          2$:     BIC     #177740,R3              ;GET SECTOR ADR
042416  022703  000025                  CMP     #25,R3                  ;LEGAL SECTOR
042422  100004                          BPL     3$                      ;BRANCH IF YES
042424  062703  000025                  ADD     #25,R3                  ;MAKE IT LEGAL
042430  042703  000040                  BIC     #BIT5,R3                ;GET RID OF ADD CARRY
042434  050301                  3$:     BIS     R3,R1                   ;COMBINE TRACK-SECTOR
042436  010162  001702                  MOV     R1,RUNTRAK(R2)          ;PUT TRAK-SECT IN TABLE
042442  010037  002130                  MOV     R0,@#RP4HDC             ;SAVE CYLINDER ADR
042446  010137  002126                  MOV     R1,@#RP4HDA             ;SAVE TRAK-SECTOR ADR
042452  112737  177775  002151          MOVB    #-3,@#RP4TRY            ;SET TRY COUNT
042460  104407                          RESREG
042462                          RP4WTRY:
042462  004767  000026                  JSR     PC,LDRP4                ;LOAD RP4 REGISTERS
042466  012777  044530  137556          MOV     #RP4SRV,@RP4VEC ;LOAD INTERRUPT VECTOR
042474  005077  137554                  CLR     @RP4PSW                 ;
042500  005037  002142                  CLR     @#RP4FUN                ;SET FUNCTION TO WRITE
042504  112777  000161  137504          MOVB    #161,@RP4CS1            ;LOAD FUNCTION AND GO
042512  000002                          RTI                             ;RETURN

042514  013777  002076  137506  LDRP4:  MOV     @#RP4UNIT,@RP4CS2       ;LOAD UNIT NUMBER
042522  012777  010000  137520          MOV     #BIT12,@RP4OF           ;SET FORMAT TO 16 BIT
042530  013777  002130  137502          MOV     @#RP4HDC,@RP4DC         ;LOAD CYLINDER ADR
042536  013777  002126  137462          MOV     @#RP4HDA,@RP4DA         ;LOAD TRAK-SECTOR
042544  013777  002002  137446          MOV     @#RP4HWC,@RP4WC         ;LOAD WORD COUNT
042552  010546                          MOV     R5,-(SP)                ;SAVE R5
```

```
042554  013705  002030                          MOV     @#RP4OLD+2,R5
042560  072527  000010                          ASH     #10,R5                  ;SHIFT EXTENDED ADDR. BITS
042564  042705  176377                          BIC     #176377,R5              ;       AND
042570  042777  001400  137420                  BIC     #1400,@RP4CS1
042576  050577  137414                          BIS     R5,@RP4CS1              ;LOAD INTO RP4CS1
042602  012605                                  MOV     (SP)+,R5                ;RESTORE R5
042604  013777  002026  137410                  MOV     @#RP4OLD,@RP4BA         ;LOAD BUS ADR
042612  000207                                  RTS     PC                      ;RETURN

                                        ;;*****************************************************************
                                        .SBTTL  RH11/RS04 HANDLER
                                        ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
                                        ;;*****************************************************************
042614  104406                          RSDRV:  SAVREG
042616  105037  001764                          CLRB    @#RSHSTAT               ;CLEAR DONE FLAG
042622  032737  000400  001764                  BIT     #BIT8,@#RSHSTAT         ;REPEAT FLAG SET?
042630  001403                                  BEQ     3$                      ;BRANCH IF NO
042632  104407                                  RESREG
042634  000137  045234                          JMP     @#RSRPT
042640  013737  001620  001640          3$:     MOV     @#RNTBINX,@#RS11        ;SAVE RUN TABLE INDEX
042646  032777  000020  136366                  BIT     #SW4,@SWR               ;RANDOM DSK ADR?
042654  001403                                  BEQ     1$                      ;BRANCH IF YES
042656  005000                                  CLR     R0
042660  005001                                  CLR     R1
042662  000407                                  BR      4$
042664  004737  052752                  1$:     JSR     PC,@#$RAND              ;GET RANDOM NUMBER
042670  013700  001566                          MOV     @#$HINUM,R0
042674  072027  177774                          ASH     #-4,R0
042700  010001                                  MOV     R0,R1                   ;SAVE RANDOM NUMBER
042702  042700  170077                  4$:     BIC     #170077,R0              ;GET TRACK ADR
042706  022700  007600                          CMP     #7600,R0                ;IS IT LEGAL?
042712  100003                                  BPL     5$                      ;BRANCH IF YES
042714  062700  007600                          ADD     #7600,R0                ;MAKE IT LEGAL
042720  000770                                  BR      4$
042722  013702  001640                  5$:     MOV     @#RS11,R2               ;GET RUN TABLE INDEX
042726  072027  177772                          ASH     #-6,R0                  ;ADJUST TRACK ADR
042732  110062  001666                          MOVB    R0,RUNTBL(R2)           ;SAVE TRAK ADR IN RUN TBL
042736  042701  177700                  6$:     BIC     #177700,R1              ;GET SECTOR ADR
042742  022701  000077                          CMP     #77,R1                  ;IS IT LEGAL?
042746  100003                                  BPL     2$                      ;BRANCH IF YES
042750  062701  000077                          ADD     #77,R1                  ;MAKE IT LEGAL
042754  000770                                  BR      6$
042756  010162  001702                  2$:     MOV     R1,RUNTRAK(R2)          ;SAVE IN RUN TRAK TABLE
042762  072027  000006                          ASH     #6,R0
042766  050100                                  BIS     R1,R0                   ;COMBINE SECTOR TRAK
042770  010037  002132                          MOV     R0,@#RSHDA              ;SAVE AS DSK ADR
042774  112737  177775  002152                  MOVB    #-3,@#RSTRY             ;SET TRY COUNT
043002  104407                                  RESREG
043004  004737  043044                  RSWTRY: JSR     PC,@#LDRS               ;GO LOAD REGISTERS
043010  012777  045260  137262                  MOV     #RSSRV,@RSVEC           ;SET INTERRUPT VECTOR
043016  005077  137260                          CLR     @RSPSW
043022  005037  002144                          CLR     @#RSFUN                 ;SET FUNCTION TO WRITE
043026  105777  137242                  1$:     TSTB    @RSDS                   ;IS DRIVE READY?
043032  001775                                  BEQ     1$                      ;BRANCH IF NO
043034  112777  000161  137214                  MOVB    #161,@RSCS1             ;LOAD FUNCTION AND GO
043042  000002                                  RTI
```

```
     043044  013777  002100  137216  LDRS:    MOV     @#RSUNIT,@RSCS2          ;LOAD UNIT NUMBER
     043052  013777  002132  137206           MOV     @#RSHDA,@RSDA            ;LOAD DSK ADR
     043060  013777  002004  137172           MOV     @#RSHWC,@RSWC            ;LOAD WORD COUNT
     043066  010546                           MOV     R5,-(SP)                 ;SAVE R5
     043070  013705  002034                   MOV     @#RSOLD+2,R5             ;SHIFT EXTENDED ADDR. BITS
     043074  072527  000010                   ASH     #10,R5                   ;        AND
     043100  042705  176377                   BIC     #176377,R5               ;        LOAD
     043104  042777  001400  137144           BIC     #1400,@RSCS1
     043112  050577  137140                   BIS     R5,@RSCS1                ;INTO RSCS1
     043116  012605                           MOV     (SP)+,R5                 ;RESTORE R5
     043120  013777  002032  137134           MOV     @#RSOLD,@RSBA            ;LOAD BUS ADDRESS
     043126  000207                           RTS     PC                       ;RETURN

                                     ;;*******************************************************************
                                     .SBTTL  RP11/RP03 SERVICE ROUTINE
                                     ;*       SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
                                     ;;*******************************************************************
     043130  000005                   RP3RPT: RESET
     043132  005337  002134                   DEC     @#RP3FUN                 ;RESTORE FUNCTION
     043136  022737  000001  002134           CMP     #1,@#RP3FUN              ;WHAT IS IT?
     043144  001472                           BEQ     RP31                     ;BRANCH IF WC
     043146  002402                           BLT     1$                       ;BRANCH IF WRITE
     043150  000137  041544                   JMP     @#RP3WTRY                ;BRANCH TO READ
     043154  000167  000366           1$:     JMP     RP33
     043160  005237  002134           RP3SRV: INC     @#RP3FUN                 ;INCREMENT FUNCTION
     043164  022737  000002  002134           CMP     #2,@#RP3FUN              ;WHAT IS IT?
     043172  001501                           BEQ     RP3WCK                   ;BRANCH TO WRITE CHECK
     043174  100002                           BPL     .+6
     043176  000137  043606                   JMP     @#RP3READ

                                     ;FUNCTION JUST EXECUTED WAS A WRITE
     043202  032737  000400  001752           BIT     #BIT8,@#RP3HSTAT         ;REPEAT FLAG SET?
     043210  001036                           BNE     RP3LOOP                  ;BRANCH IF YES
     043212  005777  136742                   TST     @RP3CS                   ;ANY ERRORS?
     043216  100045                           BPL     RP31                     ;BRANCH IF NO
     043220  105737  002146                   TSTB    @#RP3TRY                 ;TRIED 3 TIMES?
     043224  001415                           BEQ     RP3ERR                   ;BRANCH IF YES
     043226  112777  000001  136724           MOVB    #BIT0,@RP3CS             ;CLEAR THE DRIVE
     043234  105777  136720                   TSTB    @RP3CS                   ;CONTROLLER READY?
     043240  100375                           BPL     .-4                      ;BRANCH IF NO
     043242  105237  002146                   INCB    @#RP3TRY                 ;INCREMENT TRY COUNT
     043246  013746  177776                   MOV     @#PSW,-(SP)              ;MAINTAIN SAME PSW
     043252  012746  041544                   MOV     #RP3WTRY,-(SP)           ;SET RETRY ADDRESS
     043256  000002                           RTI                              ;RETURN
     043260  012737  100200  001752  RP3ERR:  MOV     #100200,@#RP3HSTA        ;SET ERROR BIT IN HAND. STA
     043266  010046                           MOV     R0,-(SP)                 ;SAVE R0
     043270  013700  001630                   MOV     @#RP311,R0               ;GET RUNTABLE INDEX
     043274  052760  100000  001702           BIS     #BIT15,RUNTRAK(R0)       ;SET ERROR BIT
     043302  012600                           MOV     (SP)+,R0                 ;RESTORE R0
     043304  000002                           RTI                              ;RETURN

     043306  012737  100200  001752  RP3LOOP:MOV     #100200,@#RP3HSTAT       ;SET DONE AND ERROR
     043314  005777  136640                   TST     @RP3CS                   ;ANY ERRORS?
     043320  100403                           BMI     1$                       ;BRANCH IF YES
     043322  042737  100000  001752           BIC     #BIT15,@#RP3HSTAT        ;CLEAR ERROR BIT
```

```
043330  000002                          1$:     RTI                                     ;RETURN
                                        ;WRITE WAS OK- NOW DO A WRITE CHECK
043332  112737  177775  002146          RP31:   MOVB    #-3,@#RP3TRY                    ;INIT TRY COUNT
043340  012737  000107  001626                  MOV     #107,@#RP310                    ;SET FUNCTION
043346  053737  002010  001626                  BIS     @#RP3OLD+2,@#RP310              ;SET BAE BITS
043354  053737  002066  001626                  BIS     @#RP3UNIT,@#RP310               ;SET UNIT BITS
043362  004737  041612                  RP32:   JSR     PC,@#LDRP3                      ;LOAD RP3 REGISTERS
043366  013777  001626  136564                  MOV     @#RP310,@RP3CS                  ;LOAD FUNCTION AND GO
043374  000002                                  RTI                                     ;RETURN

                                        ;FUNCTION JUST EXECUTED WAS A WRITE CHECK
043376  032737  000400  001752          RP3WCK: BIT     #BIT8,@#RP3HSTAT                        ;REPEAT FLAG SET?
043404  001340                                  BNE     RP3LOOP                         ;BRANCH IF YES
043406  005777  136546                          TST     @RP3CS                          ;ANY ERRORS?
043412  100031                                  BPL     1$                              ;BRANCH IF NO
043414  005737  001630                          TST     @#RP311                         ;FIRST 2K?
043420  001422                                  BEQ     4$                              ;BRANCH IF YES
043422  105737  002146                  5$:     TSTB    @#RP3TRY                        ;TRIED 3 TIMES?
043426  001714                                  BEQ     RP3ERR                          ;BRANCH IF YES
043430  005337  002134                          DEC     @#RP3FUN                        ;RESTORE FUNCTION
043434  112777  000001  136516                  MOVB    #BIT0,@RP3CS                    ;CLEAR THE DRIVE
043442  105777  136512                          TSTB    @RP3CS                          ;CONTROLLER READY?
043446  100375                                  BPL     .-4                             ;BRANCH IF NO
043450  105237  002146                          INCB    @#RP3TRY                        ;INCREMENT TRY COUNT
043454  013746  177776                          MOV     @#PSW,-(SP)
043460  012746  043362                          MOV     #RP32,-(SP)
043464  000002                                  RTI                                     ;GO TRY AGAIN
043466  032777  000010  136462          4$:     BIT     #BIT3,@RP3ER                    ;WRITE CHECK ERROR?
043474  001752                                  BEQ     5$                              ;BRANCH IF NO

                                        ;WRITE CHECK OK- NOW DO A READ
043476  112737  177775  002146          1$:     MOVB    #-3,@#RP3TRY                    ;RESTORE TRY COUNT
043504  010046                          2$:     MOV     R0,-(SP)                        ;SAVE R0
043506  013700  002040                          MOV     @#RP3NWH,R0                     ;GET BAE BITS
043512  072027  000004                          ASH     #4,R0                           ;ADJUST
043516  010037  002040                          MOV     R0,@#RP3NWH                     ;SAVE
043522  012600                                  MOV     (SP)+,R0                        ;RESTORE R0
043524  012737  000105  001626                  MOV     #105,@#RP310                    ;SET FUNCTION
043532  053737  002040  001626                  BIS     @#RP3NWH,@#RP310                ;SET BAE BITS
043540  053737  002066  001626                  BIS     @#RP3UNIT,@#RP310               ;SET UNIT NUMBER
043546  013777  002116  136412          RP33:   MOV     @#RP3HDA,@RP3DA                 ;LOAD DSK ADR
043554  013777  002120  136406                  MOV     @#RP3HDC,@RP3DC                 ;LOAD CYL
043562  013777  001772  136372                  MOV     @#RP3HWC,@RP3WC                 ;LOAD WORD COUNT
043570  013777  002036  136366                  MOV     @#RP3NWL,@RP3BA                 ;LOAD BUS ADR
043576  013777  001626  136354                  MOV     @#RP310,@RP3CS                  ;LOAD FUNCTION AND GO
043604  000002                                  RTI                                     ;RETURN

                                        ;FUNCTION JUST EXECUTED WAS A READ
043606  032737  000400  001752          RP3READ:BIT     #BIT8,@#RP3HSTAT                        ;REPEAT FLAG SET?
043614  001234                                  BNE     RP3LOOP                         ;BRANCH IF YES
043616  005777  136336                          TST     @RP3CS                          ;ANY ERRORS?
043622  100022                                  BPL     1$                              ;BRANCH IF NO
043624  105737  002146                          TSTB    @#RP3TRY                        ;TRIED 3 TIMES?
043630  001613                                  BEQ     RP3ERR                          ;BRANCH IF YES
043632  005337  002134                          DEC     @#RP3FUN                        ;RESTORE FUNCTION
043636  112777  000001  136314                  MOVB    #BIT0,@RP3CS                    ;CLEAR THE DRIVE
```

```
043644  105777  136310                  TSTB    @RP3CS                  ;CONTROLLER READY?
043650  100375                          BPL     .-4                     ;BRANCH OF NO
043652  105237  002146                  INCB    @#RP3TRY                ;INCREMENT TRY COUNT
043656  013746  177776                  MOV     @#PSW,-(SP)
043662  012746  043546                  MOV     #RP33,-(SP)
043666  000002                          RTI                             ;GO TRY AGAIN
043670  112737  000200  001752  1$:     MOVB    #200,@#RP3HSTA          ;SET DONE FLAG
043676  000002                          RTI                             ;RETURN
                                ;;*************************************************************
                                .SBTTL  RK11/RK05 SERVICE ROUTINE
                                ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
                                ;;*************************************************************
043700  000005                  RKRPT:  RESET
043702  005337  002136                  DEC     @#RKFUN                 ;RESTORE FUNCTION
043706  022737  000001  002136          CMP     #1,@#RKFUN              ;WHAT IS IT?
043714  001475                          BEQ     RK1                     ;BRANCH IF WC
043716  002402                          BLT     1$                      ;BRANCH IF WRITE
043720  000137  042140                  JMP     @#RKWTRY                ;IT WAS A WRITE
043724  000137  044332          1$:     JMP     @#RK3
043730  062737  000001  002136  RKSRV:  ADD     #1,@#RKFUN              ;FIND OUT WHAT FUNCTION
                                                                        ;WAS EXECUTED
043736  022737  000002  002136          CMP     #2,@#RKFUN              ;WAS IT A WRITE CHECK?
043744  001507                          BEQ     RKWRCK                  ;BRANCH IF YES
043746  100002                          BPL     .+6                     ;BRANCH IF IT WAS A WRITE
043750  000137  044364                  JMP     @#RKREAD

                                ;FUNCTION JUST EXECUTED WAS A WRITE. ANY ERRORS?
043754  032737  000400  001754          BIT     #BIT8,@#RKHSTAT         ;REPEAT FLAG SET?
043762  001040                          BNE     RKLOOP                  ;BRANCH IF YES
043764  005777  136212                  TST     @RKCS                   ;ANY ERRORS?
043770  100047                          BPL     RK1                     ;BRANCH IF NO
043772  105737  002147                  TSTB    @#RKTRY                 ;TRYED 3 TIMES?
043776  001417                          BEQ     RKERR                   ;BRANCH IF YES
044000  012777  000001  136174          MOV     #1,@RKCS                ;CLEAR THE ERROR
044006  004737  044466                  JSR     PC,@#TIMER              ;WAIT A LITTLE
044012  105777  136164                  TSTB    @RKCS                   ;WAIT FOR CONT CLR TO FINISH
044016  100375                          BPL     .-4
044020  105237  002147                  INCB    @#RKTRY                 ;INCREMENT TRY COUNT
044024  013746  177776                  MOV     @#PSW,-(SP)
044030  012746  042140                  MOV     #RKWTRY,-(SP)
044034  000002                          RTI
044036  012737  100200  001754  RKERR:  MOV     #100200,@#RKHSTAT       ;SET ERROR & DONE FLAG
044044  010046                          MOV     R0,-(SP)                ;SAVE R0
044046  013700  001634                  MOV     @#RK11,R0               ;GET SAVED RUN TABLE INDEX
044052  052760  100000  001702          BIS     #BIT15,RUNTRAK(R0)      ;SET ERROR BIT IN RUN TABLE
044060  012600                          MOV     (SP)+,R0                ;RESTORE R0
044062  000002                          RTI                             ;RETURN

044064  012737  100200  001754  RKLOOP:MOV      #100200,@#RKHSTAT       ;SET DONE AND ERROR BITS
044072  005777  136104                  TST     @RKCS                   ;ANY ERRORS?
044076  100403                          BMI     1$                      ;BRANCH IF YES
044100  042737  100000  001754          BIC     #BIT15,@#RKHSTAT        ;CLEAR ERROR BIT
044106  000002                  1$:     RTI                             ;RETURN
                                ;WRITE WAS OK, NOW DO A WRITE CHECK
044110  112737  177775  002147  RK1:    MOVB    #-3,@#RKTRY             ;RESTORE TRY COUNT
044116  012767  000507  135506          MOV     #507,RK10               ;SET FUNCTION TO WRITE
```

```
044124  053767  002014  135500          BIS     @#RKOLD+2,RK10          ;SET BA EXT BITS
044132  013777  002122  136050  RK2:    MOV     @#RKHDA,@RKDA           ;LOAD DISK ADDRESS
044140  013777  001774  136036          MOV     @#RKHWC,@RKWC           ;LOAD WORD COUNT
044146  013777  002012  136032          MOV     @#RKOLD,@RKBA           ;LOAD BUS ADDRESS
044154  016777  135452  136020          MOV     RK10,@RKCS              ;START FUNCTION
044162  000002                          RTI                             ;RETURN

                                ;FUNCTION JUST EXECUTED WAS A WRITE CHECK. ANY ERRORS?
```

```
 9111  044164  032737  000400  001754  RKWRCK:BIT    #BIT8,@#RKHSTAT         ;REPEAT FLAG SET?
 9112  044172  001334                         BNE    RKLOOP                  ;BRANCH IF YES
 9113  044174  005777  136002                 TST    @RKCS                   ;ANY ERRORS?
 9114  044200  100033                         BPL    1$                      ;BRANCH IF NO
 9115  044202  005737  001634                 TST    @#R''11                 ;FIRST 2K?
 9116  044206  001424                         BEQ    4$                      ;BRANCH IF YES
 9117  044210  105737  002147         5$:     TSTB   @#RKTRY                 ;TRYED 3 TIMES?
 9118  044214  001710                         BEQ    RKERR                   ;BRANCH IF YES
 9119  044216  005337  002136                 DEC    @#RKFUN                 ;SET FUNCTION BACK TO WC
 9120  044222  012777  000001  135752         MOV    #1,@RKCS                ;CLEAR THE ERROR
 9121  044230  004737  044466                 JSR    PC,@#TIMER              ;WAIT A LITTLE
```

```
044234  105777  135742                    TSTB    @RKCS                     ;WAIT FOR CLR TO FINISH
044240  100375                            BPL     .-4
044242  105237  002147                    INCB    @#RKTRY                   ;INCREMENT TRY COUNT
044246  013746  177776                    MOV     @#PSW,-(SP)
044252  012746  044132                    MOV     #RK2,-(SP)
044256  000002                            RTI
044260  032777  040000  135714  4$:       BIT     #BIT14,@RKCS              ;HARD ERROR?
044266  001350                            BNE     5$                        ;BRANCH IF YES

                                  ;WRITE CHECK WAS OK, NOW DO A READ.
044270  112737  177775  002147  1$:       MOVB    #-3,@#RKTRY               ;RESTORE TRY COUNT
044276  010046                  2$:       MOV     R0,-(SP)                  ;SAVE R0
044300  013700  002044                    MOV     @#RKNEWH,R0               ;GET BA EXT
044304  072027  000004                    ASH     #4,R0                     ;ADJUST
```

```
044310  010037  002044                    MOV     R0,@#RKNEWH             ;SAVE
044314  012600                            MOV     (SP)+,R0                ;RESTORE R0
044316  012767  000105  135306            MOV     #105,RK10               ;SET FUNCTION
044324  053767  002044  135300            BIS     @#RKNEWH,RK10           ;SET BA EXT BITS IN FUNCTION
044332  013777  002122  135650    RK3:    MOV     @#RKHDA,@RKDA           ;LOAD DISK ADDRESS
044340  013777  001774  135636            MOV     @#RKHWC,@RKWC           ;LOAD WORD COUNT
044346  013777  002042  135632            MOV     @#RKNEWL,@RKBA          ;LOAD BUS ADDRESS
044354  016777  135252  135620            MOV     RK10,@RKCS              ;LOAD FUNCTION AND GO
044362  000002                            RTI                             ;RETURN

                                  ;FUNCTION JUST EXECUTED WAS A READ. ANY ERRORS?
044364  032737  000400  001754    RKREAD: BIT     #BIT8,@#RKHSTAT                 ;REPEAT FLAG SET?
044372  001234                            BNE     RKLOOP                  ;BRANCH IF YES
044374  005777  135602                    TST     @RKCS                   ;ANY ERRORS?
044400  100026                            BPL     1$                      ;BRANCH IF NO
044402  105737  002147                    TSTB    @#RKTRY                 ;TRYED 3 TIMES?
044406  001002                            BNE     3$                      ;BRANCH IF NO
044410  000167  177422                    JMP     RKERR
044414  005337  002136            3$:     DEC     @#RKFUN                 ;SET FUNCTION BACK TO READ
044420  012777  000001  135554            MOV     #1,@RKCS                ;CLEAR THE ERROR
044426  004737  044466                    JSR     PC,@#TIMER              ;WAIT A LITTLE
044432  105777  135544                    TSTB    @RKCS                   ;WAIT FOR CLR TO FINISH
044436  100375                            BPL     .-4
044440  105237  002147                    INCB    @#RKTRY                 ;INCREMENT TRY COUNT
044444  013746  177776                    MOV     @#PSW,-(SP)
044450  012746  044332                    MOV     #RK3,-(SP)
044454  000002                            RTI
044456  112737  000200  001754    1$:     MOVB    #200,@#RKHSTAT          ;SET DON E FLAG
044464  000002                            RTI                             ;RETURN
044466  005067  000010            TIMER:  CLR     1$
044472  105267  000004            2$:     INCB    1$
044476  001375                            BNE     2$
044500  000207                            RTS     PC
044502  000000                    1$:     .WORD

                                  ;;*****************************************************************
                                  .SBTTL  RH11/RP04 SERVICE ROUTINE
                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
                                  ;;*****************************************************************
044504  000005                    RP4RPT: RESET
044506  005337  002142                    DEC     @#RP4FUN                ;RESTORE FUNCTION
044512  022737  000001  002142            CMP     #1,@#RP4FUN             ;WHAT IS IT?
044520  001501                            BEQ     RP41                    ;BRANCH IF WC
044522  002560                            BLT     RP43                    ;BRANCH IF READ
044524  000137  042462                    JMP     @#RP4WTRY               ;GO TO WRITE
044530  005237  002142            RP4SRV: INC     @#RP4FUN                ;FIND OUT WHAT FUNCTION
044534  022737  000002  002142            CMP     #2,@#RP4FUN             ;WAS JUST EXECUTED
044542  001504                            BEQ     RP4WCK
044544  100576                            BMI     RP4READ

                                  ;WRITE FUNCTION WAS JUST EXECUTED.
044546  032737  000400  001762            BIT     #BIT8,@#RP4HSTAT        ;REPEAT FLAG SET?
044554  001050                            BNE     RP4LOOP                 ;BRANCH IF YES
044556  032777  040000  135450            BIT     #BIT14,@RP4DS           ;ANY ERRORS
044564  001457                            BEQ     RP41                    ;BRANCH IF NO
044566  105737  002151                    TSTB    @#RP4TRY                ;TRIED 3 TIMES?
```

```
044572  001426                             BEQ     RP4ERR                  ;BRANCH IF YES
044574  052777  000040  135426             BIS     #BIT5,@RP4CS2           ;CLEAR ALL ERRORS
044602  004737  042514                     JSR     PC,@#LDRP4              ;RELOAD THE UNIT NO
044606  105237  002151                     INCB    @#RP4TRY                ;INCREMENT TRY COUNT
044612  013746  177776                     MOV     @#PSW,-(SP)             ;SETUP THE STACK TO
044616  012746  042462                     MOV     #RP4WTRY,-(SP)          ;TRY WRITE AGAIN
044622  032737  000400  001762             BIT     #BIT8,@#RP4HSTAT        ;REPEAT FLAG SET?
044630  001006                             BNE     2$                      ;BRANCH IF YES
044632  012777  000007  135356             MOV     #7,@RP4CS1              ;RECALIBRATE
044640  105777  135352          1$:        TSTB    @RP4CS1                 ;DRIVE READY?
044644  100375                             BPL     1$                      ;BRANCH IF NO
044646  000002                  2$:        RTI
044650  012737  100200  001762  RP4ERR:    MOV     #100200,@#RP4HSTA       ;SET ERROR & DONE BIT
044656  010046                             MOV     R0,-(SP)                ;SAVE R0
044660  013700  001636                     MOV     @#RP411,R0              ;GET RUN TABLE INDEX
044664  052760  100000  001702             BIS     #BIT15,RUNTRAK(R0)      ;SET ERROR BIT
044672  012600                             MOV     (SP)+,R0                ;RESTORE R0
044674  000002                             RTI                             ;RETURN

044676  012737  100200  001762  RP4LOOP:MOV        #100200,@#RP4HSTAT      ;SET DONE AND ERROR BITS
044704  032777  040000  135322             BIT     #BIT14,@RP4DS           ;ANY ERRORS?
044712  001003                             BNE     1$                              ;BRANCH IF YES
044714  042737  100000  001762             BIC     #BIT15,@#RP4HSTAT       ;CLEAR ERROR BIT
044722  000002                  1$:        RTI                             ;RETURN
                                ;WRITE OK...NOW DO A WRITE CHECK.
044724  112737  177775  002151  RP41:      MOVB    #-3,@#RP4TRY            ;INITIALIZE TRY COUNT
044732  105777  135276          RP42:      TSTB    @RP4DS                  ;IS DRIVE READY?
044736  001775                             BEQ     RP42                    ;BRANCH IF NO
044740  004737  042514                     JSR     PC,@#LDRP4
044744  112777  000151  135244             MOVB    #151,@RP4CS1            ;LOAD FUNCTION AND GO
044752  000002                             RTI

                                ;FUNCTION JUST EXECUTED WAS A WRITE CHECK
044754  032737  000400  001762  RP4WCK:    BIT     #BIT8,@#RP4HSTAT                ;REPEAT FLAG SET?
044762  001345                             BNE     RP4LOOP                 ;BRANCH IF YES
044764  032777  040000  135242             BIT     #BIT14,@RP4DS           ;ANY ERRORS?
044772  001421                             BEQ     1$                      ;BRANCH IF NO
044774  105737  002151          3$:        TSTB    @#RP4TRY                ;TRIED 3 TIMES?
045000  001723                             BEQ     RP4ERR                  ;BRANCH IF YES
045002  005337  002136                     DEC     @#RKFUN                 ;SET FUNCTION TO WC
045006  052777  000040  135214             BIS     #BIT5,@RP4CS2           ;CLEAR ALL ERRORS
045014  004737  042514                     JSR     PC,@#LDRP4              ;RELOAD THE UNIT NO
045020  105237  002151                     INCB    @#RP4TRY                ;INCREMENT TRY COUNT
045024  013746  177776                     MOV     @#PSW,-(SP)
045030  012746  044732                     MOV     #RP42,-(SP)
045034  000002                             RTI                             ;TRY AGAIN
045036  032777  040000  135164  1$:        BIT     #BIT14,@RP4CS2          ;WRITE CHECK ERROR?
045044  001404                             BEQ     2$                      ;BRANCH IF NO
045046  005737  001636                     TST     @#RP411                 ;FIRST 2K?
045052  001401                             BEQ     2$
045054  000747                             BR      3$

                                ;WRITE CHECK WAS OK...NOW DO A READ.
045056  112737  177775  002151  2$:        MOVB    #-3,@#RP4TRY            ;INITIALIZE TRY COUNT
045064  105777  135144          RP43:      TSTB    @RP4DS                  ;IS DRIVE READY?
045070  001775                             BEQ     RP43                    ;BRANCH IF NO
```

```
045072  004737  042514                    JSR     PC,@#LDRP4              ;LOAD REGISTERS
045076  010546                            MOV     R5,-(SP)                ;SAVE R5
045100  013705  002060                    MOV     @#RP4NWH,R5             ;SHIFT EXTENDED
045104  072527  000010                    ASH     #10,R5                  ;ADDRESS BITS
045110  042777  001400  135100            BIC     #1400,@RP4CS1           ;       AND
045116  050577  135074                    BIS     R5,@RP4CS1              ;LOAD INTO RP4CS1
045122  012605                            MOV     (SP)+,R5                ;RESTORE R5
045124  013777  002056  135070            MOV     @#RP4NWL,@RP4BA         ;LOAD BUS ADR
045132  112777  000171  135056            MOVB    #171,@RP4CS1            ;LOAD FUNCTION AND GO
045140  000002                            RTI                             ;RETURN

                                  ;FUNCTION JUST EXECUTED WAS A READ.
045142  032737  000400  001762  RP4READ:BIT     #BIT8,@#RP4HSTAT        ;REPEAT FLAG SET?
045150  001252                            BNE     RP4LOOP                 ;BRANCH IF YES
045152  032777  040000  135054            BIT     #BIT14,@RP4DS           ;ANY ERRORS?
045160  001421                            BEQ     1$                      ;BRANCH IF NO
045162  105737  002151                    TSTB    @#RP4TRY                ;TRIED 3 TIMES?
045166  001630                            BEQ     RP4ERR                  ;BRANCH IF YES
045170  005337  002142                    DEC     @#RP4FUN                ;SET FUNCTION TO A READ
045174  052777  000040  135026            BIS     #BIT5,@RP4CS2           ;CLEAR ALL ERRORS
045202  004737  042514                    JSR     PC,@#LDRP4              ;RELOAD THE UNIT NO
045206  105237  002151                    INCB    @#RP4TRY                ;INCREMENT TRY COUNT
045212  013746  177776                    MOV     @#PSW,-(SP)
045216  012746  045064                    MOV     #RP43,-(SP)
045222  000002                            RTI                             ;TRY AGAIN
045224  112737  000200  001762  1$:       MOVB    #200,@#RP4HSTA          ;SET DONE FLAG
045232  000002                            RTI                             ;RETURN
                                  ;;*************************************************************
                                  .SBTTL  RH11/RS04 SERVICE ROUTINE
                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
                                  ;;*************************************************************
045234  000005                  RSRPT:  RESET
045236  005337  002144                  DEC     @#RSFUN                 ;RESTORE FUNCTION
045242  022737  000001  002144          CMP     #1,@#RSFUN              ;WHAT IS IT?
045250  001465                          BEQ     RS41                    ;BRANCH IF WC
045252  002542                          BLT     RS43                    ;BRANCH IF WRITE
045254  000137  043004                  JMP     @#RSWTRY
045260  005237  002144          RSSRV:  INC     @#RSFUN                 ;FIND OUT WHAT FUNCTION
045264  022737  000002  002144          CMP     #2,@#RSFUN              ;WAS JUST EXECUTED
045272  001470                          BEQ     RSWCK
045274  100562                          BMI     RSREAD

                                ;WRITE FUNCTION WAS JUST EXECUTED
045276  032737  000400  001754          BIT     #BIT8,@#RSHSTAT ;REPEAT FLAG SET?
045304  001034                          BNE     RSLOOP                  ;BRANCH IF YES
045306  032777  040000  134760          BIT     #BIT14,@RSDS            ;ANY ERRORS?
045314  001443                          BEQ     RS41                    ;BRANCH IF NO
045316  105737  002152                  TSTB    @#RSTRY                 ;TRIED 3 TIMES?
045322  001412                          BEQ     RSERR                   ;BRANCH IF YES
045324  052777  000040  134736          BIS     #BIT5,@RSCS2            ;CLEAR ALL ERRORS
045332  105237  002152                  INCB    @#RSTRY                 ;INCREMENT TRY COUNT
045336  013746  177776                  MOV     @#PSW,-(SP)             ;SETUP THE STACK TO
045342  012746  043004                  MOV     #RSWTRY,-(SP)           ;TRY THE WRITE AGAIN
045346  000002                          RTI
045350  012737  100200  001764  RSERR:  MOV     #100200,@#RSHSTAT       ;SET ERROR AND DONE BIT
045356  010046                          MOV     R0,-(SP)                ;SAVE R0
```

```
045360  013700  001640                     MOV     @#RS11,R0               ;GET RUN TBL INDEX
045364  052760  100000  001702             BIS     #BIT15,RUNTRAK(R0)      ;SET ERROR BIT
045372  012600                             MOV     (SP)+,R0                ;RESTORE R0
045374  000002                             RTI

045376  012737  100200  001764     RSLOOP: MOV     #100200,@#RSHSTAT       ;SET DONE AND ERROR BITS
045404  032777  040000  134662             BIT     #BIT14,@RSDS            ;ANY ERRORS?
045412  001003                             BNE     1$                      ;BRANCH IF YES
045414  042737  100000  001764             BIC     #BIT15,@#RSHSTAT        ;CLEAR ERROR BIT
045422  000002                     1$:     RTI                             ;RETURN
                                   ;WRITE OK...NOW DO A WRITE CHECK
045424  112737  177775  002152     RS41:   MOVB    #-3,@#RSTRY             ;INIT TRY COUNT
045432  105777  134636             RS42:   TSTB    @RSDS                   ;IS DRIVE READY?
045436  001775                             BEQ     RS42                    ;BRANCH IF NO
045440  004737  043044                     JSR     PC,@#LDRS               ;LOAD RS REGISTERS
045444  112777  000151  134604             MOVB    #151,@RSCS1             ;LOAD FUNCTION AND GO
045452  000002                             RTI                             ;RETURN

                                   ;FUNCTION JUST EXECUTED WAS A WRITE CHECK
045454  032737  000400  001764     RSWCK:  BIT     #BIT8,@#RSHSTAT         ;REPEAT FLAG SET?
045462  001345                             BNE     RSLOOP                  ;BRANCH IF YES
045464  032777  040000  134602             BIT     #BIT14,@RSDS            ;ANY ERRORS?
045472  001417                             BEQ     1$                      ;BRANCH IF NO
045474  105737  002152             3$:     TSTB    @#RSTRY                 ;TRIED 3 TIMES?
045500  001723                             BEQ     RSERR                   ;BRANCH IF YES
045502  005337  002144                     DEC     @#RSFUN                 ;SET FUNCTION BACK TO WC
045506  052777  000040  134554             BIS     #BIT5,@RSCS2            ;CLEAR THE ERROR
045514  105237  002152                     INCB    @#RSTRY                 ;INCREMENT THE TRY COUNT
045520  013746  177776                     MOV     @#PSW,-(SP)
045524  016746  177702                     MOV     RS42,-(SP)
045530  000002                             RTI                             ;TRY AGAIN

045532  032777  040000  134530     1$:     BIT     #BIT14,@RSCS2           ;WRITE CHECK ERROR?
045540  001404                             BEQ     2$                      ;BRANCH IF NO
045542  005737  001640                     TST     @#RS11                  ;FIRST 2K?
045546  001401                             BEQ     2$                      ;BRANCH IF YES
045550  000751                             BR      3$

                                   ;WRITE CHECK WAS OK...NOW DO A READ.
045552  112737  177775  002152     2$:     MOVB    #-3,@#RSTRY             ;INIT TRY COUNT
045560  105777  134510             RS43:   TSTB    @RSDS                   ;IS DRIVE READY?
045564  001775                             BEQ     RS43                    ;BRANCH IF NO
045566  004737  043044                     JSR     PC,@#LDRS               ;LOAD RS REGISTERS
045572  010546                             MOV     R5,-(SP)                ;SAVE R5
045574  013705  002064                     MOV     @#RSNEWH,R5             ;SHIFT EXTENDED
045600  072527  000010                     ASH     #10,R5                  ;ADDRESS BITS
045604  042705  176377                     BIC     #176377,R5              ;       AND
045610  042777  001400  134440             BIC     #1400,@RSCS1
045616  050577  134434                     BIS     R5,@RSCS1               ;LOAD INTO RSCS1
045622  012605                             MOV     (SP)+,R5                ;RESTORE R5
045624  013777  002062  134430             MOV     @#RSNEWL,@RSBA          ;LOAD BUS ADR
045632  112777  000171  134416             MOVB    #171,@RSCS1             ;LOAD FUNCTION AND GO
045640  000002                             RTI                             ;RETURN

                                   ;FUNCTION JUST EXECUTED WAS A READ.
045642  032737  000400  001764     RSREAD: BIT     #BIT8,@#RSHSTAT         ;REPEAT FLAG SET?
```

```
045650  001252                          BNE     RSLOOP                  ;BRANCH IF YES
045652  032777  040000  134414          BIT     #BIT14,@RSDS            ;ANY ERRORS?
045660  001417                          BEQ     1$                      ;BRANCH IF NO
045662  105737  002152                  TSTB    @#RSTRY                 ;TRIED 3 TIMES?
045666  001630                          BEQ     RSERR                   ;BRANCH IF YES
045670  005337  002144                  DEC     @#RSFUN                 ;RESTORE FUN TO READ
045674  052777  000040  134366          BIS     #BIT5,@RSCS2            ;CLEAR ALL ERRORS
045702  105237  002152                  INCB    @#RSTRY                 ;INCREMENT TRY COUNT
045706  013746  177776                  MOV     @#PSW,-(SP)
045712  012746  045560                  MOV     #RS43,-(SP)
045716  000002                          RTI                             ;TRY AGAIN
045720  112737  000200  001764  1$:     MOVB    #200,@#RSHSTAT          ;SET DONE FLAG
045726  000002                          RTI                             ;RETURN
                                ;;*******************************************************************
                                .SBTTL  UNIBUS EXERCISER SERVICE ROUTINE
                                ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
                                ;;*******************************************************************
045730  104406                  UBESRV: SAVREG
045732  004737  054516                  JSR     PC,@#LDKT               ;GO TO LOW CORE
045736  012704  002312                  MOV     #UBETBL+6,R4            ;GET ADDRESS OF UBECR1
045742  005774  000000                  TST     @(R4)                   ;WAS THERE AN ERROR?
045746  100430                          BMI     UBE2                    ;BRANCH IF YES
045750  062737  000776  001716          ADD     #776,@#UBESAV           ;INCREMENT UBE BUS ADR
045756  005537  001720                  ADC     @#UBESAV+2
045762  013737  001716  001722          MOV     @#UBESAV,@#UBEADR
045770  013737  001720  001724          MOV     @#UBESAV+2,@#UBEADR+2
045776  013754  001724                  MOV     @#UBEADR+2,@-(R4)       ;LOAD UBECR2
046002  013754  001722                  MOV     @#UBEADR,@-(R4)         ;LOAD UBEBA
046006  012754  170000                  MOV     #170000,@-(R4)          ;LOAD UBECC
046012  004737  054604                  JSR     PC,@#RESKT              ;GO BACK TO ORIGINAL CORE
046016  104407                          RESREG
046020  012777  064545  134264          MOV     #64545,@UBETBL+6        ;RESTART UBE
046026  000002                          RTI                             ;RETURN

                                ;UBE ERROR-IS IT LAST MEMORY?
046030  005037  001732          UBE2:   CLR     @#MEMFLG
046034  162704  000004                  SUB     #4,R4                   ;ADJUST R4
046040  017403  000002                  MOV     @2(R4),R3               ;GET BECR2
046044  042703  000003                  BIC     #3,R3                   ;GET RID OF ADDRESS BITS
046050  022703  000400                  CMP     #400,R3                 ;WAS ERROR A TIMEOUT?
046054  001041                          BNE     UBEERR                  ;BRANCH IF NO
046056  017437  000000  001540          MOV     @(R4),@#PA1500          ;SAVE BUS ADR OF ERROR
046064  017437  000002  001542          MOV     @2(R4),@#PA1716
046072  042737  177774  001542          BIC     #177774,@#PA1716
046100  162737  000004  001540          SUB     #4,@#PA1500             ;ADJUST PHYSICAL ADR THAT FAILED
046106  005637  001542                  SBC     @#PA1716                ;UBE STOPS AT ADR+4
046112  023737  001540  001624          CMP     @#PA1500,@#MXMMLO       ;AT MAXIMUM MEMORY LO?
046120  001015                          BNE     MHOLE                   ;BRANCH IF NO
046122  023737  001542  001622          CMP     @#PA1716,@#MXMMHI       ;AT MAX MEMORY HI?
046130  001011                          BNE     MHOLE                   ;BRANCH IF NO
046132  004737  053646                  JSR     PC,@#UBEINIT
046136  004737  054604                  JSR     PC,@#RESKT
046142  104407                          RESREG
046144  012777  064545  134140          MOV     #64545,@UBETBL+6
046152  000002                          RTI
```

```
046154  010637  001732          MHOLE:  MOV     SP,@#MEMFLG
046160  013737  001212  001734  UBEERR: MOV     @#$LPERR,@#ERRADR;SAVE LOOP ARROR ADR
046166  012737  046230  001212          MOV     #UBE3,@#$LPERR          ;SET LOOP ADR
046174  012703  000022                  MOV     #22,R3
046200  005737  001732                  TST     @#MEMFLG
046204  001002                          BNE     1$
046206  104007                          ERROR   7
046210  000407                          BR      UBE3
046212  013737  001540  001226  1$:     MOV     @#PA1500,@#$GDDAT
046220  013737  001542  001230          MOV     @#PA1716,@#$BDDAT
046226  104012                          ERROR   12

                                ;RESTART UBE IN SAME MEMORY
046230  013737  001734  001212  UBE3:   MOV     @#ERRADR,@#$LPERR       ;RESTORE ERROR LOOP ADR
046236  010446                          MOV     R4,-(SP)                ;SAVE R4
046240  012704  002304                  MOV     #UBETBL,R4              ;GET ADDRESS OF UBE TABLE
046244  012734  170000                  MOV     #170000,@(R4)+          ;SET UBECC
046250  013734  001722                  MOV     @#UBEADR,@(R4)+         ;SET UBEBA <15:00>
046254  005074  000004                  CLR     @4(R4)                  ;CLEAR ALL ERRORS
046260  013734  001724                  MOV     @#UBEADR+2,@(R4)+       ;SET EXT ADR BITS
046264  012774  064545  000000          MOV     #64545,@(R4)            ;START UBE
046272  012604                          MOV     (SP)+,R4                ;RESTORE R4
046274  004737  054604                  JSR     PC,@#RESKT
046300  104407                          RESREG
046302  000002                          RTI                             ;RETURN

                                ;;*************************************************************
                                .SBTTL  MASS BUS TESTER SERVICE ROUTINE
                                ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
                                ;;*************************************************************
046304  104406                  MBTSRV: SAVREG
046306  004737  054516                  JSR     PC,@#LDKT               ;GO TO LOW CORE
046312  005037  001732                  CLR     @#MEMFLG
046316  012704  002322                  MOV     #MBTTBL,R4              ;GET ADDRESS OF ADDRESS OF CS1 REG
046322  032734  040000                  BIT     #BIT14,@(R4)+           ;ANY ERRORS?
046326  001007                          BNE     1$                      ;BRANCH IF YES
046330  004737  054604          2$:     JSR     PC,@#RESKT              ;GO BACK TO ORIGINAL CORE
046334  104407                          RESREG
046336  112777  000161  133756          MOVB    #161,@MBTTBL
046344  000002                          RTI                             ;RESTART MBT AND RETURN
046346  062704  000010          1$:     ADD     #10,R4                  ;ADJUST R4
046352  032774  004000  000000          BIT     #BIT11,@(R4)            ;NON-EXISTANT MEMORY ERROR?
046360  001443                          BEQ     MBTERR                  ;BRANCH IF NO
046362  162704  000006                  SUB     #6,R4                   ;ADJUST R4
046366  013437  001540                  MOV     @(R4)+,@#PA1500         ;GET BUS ADR
046372  013405                          MOV     @(R4)+,R5               ;GET BUS ADR EXT
046374  072527  000010                  ASH     #10,R5                  ;SHIFT EXTENDED BITS RIGHT
046400  042705  177774                  BIC     #177774,R5              ;CLEAR ALL BITS BUT 0 & 1
046404  010537  001542                  MOV     R5,@#PA1716
046410  162737  000004  001540          SUB     #4,@#PA1500             ;ADJUST BUS ADR
046416  005637  001542                  SBC     @#PA1716
046422  023737  001540  001624          CMP     @#PA1500,@#MXMMLO       ;IS IT LAST MEMORY?
046430  001015                          BNE     MEMHOLE                 ;BRANCH IF NO
046432  023737  001542  001622          CMP     @#PA1716,@#MXMMHI       ;CHECK EXT ADR BITS
046440  001011                          BNE     MEMHOLE
046442  005724                          TST     (R4)+                   ;INCREMENT R4
```

E16


```
046444  052774  000047  000000            BIS     #47,@(R4)               ;CLEAR THE ERROR
046452  012734  000007                    MOV     #7,@(R4)+               ;SELECT UNIT 7
046456  005074  177766                    CLR     @-12(R4)                ;CLEAR WORD COUNT
046462  000722                            BR      2$                      ;CONTINUE

046464  010637  001732          MEMHOLE:MOV       SP,@#MEMFLG
046470  013737  001212  001734  MBTERR: MOV       @#$LPERR,@#ERRADR       ;SAVE LOOP ADDRESS
046476  012737  046540  001212            MOV     #1$,@#$LPERR            ;SET NEW LOOP ADR
046504  012703  000020                    MOV     #20,R3                  ;PUT DEVICE ID IN R3
046510  005737  001732                    TST     @#MEMFLG
046514  001002                            BNE     2$
046516  104007                            ERROR   7
046520  000407                            BR      1$
046522  013737  001540  001226  2$:       MOV     @#PA1500,@#$GDDAT
046530  013737  001542  001230            MOV     @#PA1716,@#$BDDAT
046536  104013                            ERROR   13
046540  013737  001734  001212  1$:       MOV     @#ERRADR,@#$LPERR       ;RESTORE LOOP ADR
046546  012704  002332                    MOV     #MBTTBL+10,R4           ;GET ADR OF MBTTBL+10
046552  015400                            MOV     @-(R4),R0               ;GET BUS ADR EXTENDED
046554  015401                            MOV     @-(R4),R1               ;GET BUS ADR
046556  015402                            MOV     @-(R4),R2               ;GET WORD COUNT
046560  006302                            ASL     R2                      ;ADJUST WORD COUNT
046562  160201                            SUB     R2,R1                   ;FORM START ADR OF THIS XFER
046564  005600                            SBC     R0
046566  052774  000047  000010            BIS     #47,@10(R4)             ;CLEAR THE WORLD
046574  012774  000007  000010            MOV     #7,@10(R4)              ;SELECT UNIT 7
046602  005724                            TST     (R4)+                   ;ADJUST R4
046604  010134                            MOV     R1,@(R4)+               ;RESTORE BUS ADR
046606  010074  000000                    MOV     R0,@(R4)
046612  004737  054604                    JSR     PC,@#RESKT              ;GO BACK TO ORIGINAL CORE
046616  104407                            RESREG
046620  112777  000161  133474            MOVB    #161,@MBTTBL            ;START MBT AGAIN
046626  000002                            RTI                             ;RETURN
                                ;;*******************************************************************
                                .SBTTL  LINE CLOCK SERVICE ROUTINE
                                ;*      THIS ROUTINE FIRST REMAPS PROGRAM EXECUTION TO LOW
                                ;*      MEMORY. IT THEN INCREMENTS AND KEEPS TRACK OF THE
                                ;*      SECOND AND MINUTE COUNTS KEPT IN LOCATIONS "LTICKS"
                                ;*      AND "MTICKS" RESPECTIVELY.
                                ;;*******************************************************************
046630  104406                  LKSRV:    SAVREG
046632  004737  054516                    JSR     PC,@#LDKT               ;GO TO LOW CORE
046636  105237  001644                    INCB    @#LTICKS                ;INCREMENT TICK COUNT
046642  122737  000074  001644            CMPB    #60.,@#LTICKS           ;ONE SECOND YET?
046650  001014                            BNE     1$                      ;BRANCH IF NO
046652  105237  001645                    INCB    @#LTICKS+1              ;INCREMENT SECOND COUNT
046656  105037  001644                    CLRB    @#LTICKS                ;CLEAR SECOND COUNT
046662  122737  000074  001645            CMPB    #60.,@#LTICKS+1         ;ONE MINUTE YET?
046670  001004                            BNE     1$                      ;BRANCH IF NO
046672  105037  001645                    CLRB    @#LTICKS+1
046676  005237  001642                    INC     @#MTICKS                ;INCREMENT MINUTE COUNT
046702  004737  054604          1$:       JSR     PC,@#RESKT              ;RESTORE THE KT
046706  104407                            RESREG
046710  012737  000100  177546            MOV     #BIT6,@#LKS             ;CLEAR READY BIT IN CLOCK
046716  000002                            RTI                             ;RETURN
```

```
                                          .SBTTL  SCOPE HANDLER ROUTINE

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

 046720                                   $SCOPE:
 046720  032777  040000  132314           1$:     BIT     #BIT14,@SWR       ;;LOOP ON PRESENT TEST?
 046726  001077                                   BNE     $OVER             ;;YES IF SW14=1
                                          ;#####START OF CODE FOR THE XOR TESTER#####
 046730  000416                           $XTSTR: BR      6$                ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                            ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
 046732  013746  000004                           MOV     @#ERRVEC,-(SP)    ;;SAVE THE CONTENTS OF THE ERROR VECTOR
 046736  012737  046756  000004                   MOV     #5$,@#ERRVEC      ;;SET FOR TIMEOUT
 046744  005737  177060                           TST     @#177060          ;;TIME OUT ON XOR?
 046750  012637  000004                           MOV     (SP)+,@#ERRVEC    ;;RESTORE THE ERROR VECTOR
 046754  000453                                   BR      $SVLAD            ;;GO TO THE NEXT TEST
 046756  022626                           5$:     CMP     (SP)+,(SP)+       ;;CLEAR THE STACK AFTER A TIME OUT
 046760  012637  000004                           MOV     (SP)+,@#ERRVEC    ;;RESTORE THE ERROR VECTOR
 046764  000413                                   BR      7$                ;;LOOP ON THE PRESENT TEST
 046766                                   6$:;#####END OF CODE FOR THE XOR TESTER#####
 046766  105767  132212                   2$:     TSTB    $ERFLG            ;;HAS AN ERROR OCCURRED?
 046772  001421                                   BEQ     3$                ;;BR IF NO
 046774  126767  132217  132202                   CMPB    $ERMAX,$ERFLG     ;;MAX. ERRORS FOR THIS TEST OCCURRED?
 047002  101015                                   BHI     3$                ;;BR IF NO
 047004  032777  001000  132230                   BIT     #BIT09,@SWR       ;;LOOP ON ERROR?
 047012  001404                                   BEQ     4$                ;;BR IF NO
 047014  016767  132172  132166           7$:     MOV     $LPERR,$LPADR     ;;SET LOOP ADDRESS TO LAST SCOPE
 047022  000441                                   BR      $OVER
 047024  105067  132154                   4$:     CLRB    $ERFLG            ;;ZERO THE ERROR FLAG
 047030  005067  132270                           CLR     $TIMES            ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
 047034  000415                                   BR      1$                ;;ESCAPE TO THE NEXT TEST
 047036  032777  004000  132176           3$:     BIT     #BIT11,@SWR       ;;INHIBIT ITERATIONS?
 047044  001011                                   BNE     1$                ;;BR IF YES
 047046  005767  132126                           TST     $PASS             ;;IF FIRST PASS OF PROGRAM
 047052  001406                                   BEQ     1$                ;;      INHIBIT ITERATIONS
 047054  005267  132126                           INC     $ICNT             ;;INCREMENT ITERATION COUNT
 047060  026767  132240  132120                   CMP     $TIMES,$ICNT      ;;CHECK THE NUMBER OF ITERATIONS MADE
 047066  002017                                   BGE     $OVER             ;;BR IF MORE ITERATION REQUIRED
 047070  012767  000001  132110           1$:     MOV     #1,$ICNT          ;;REINITIALIZE THE ITERATION COUNTER
 047076  016767  000054  132220                   MOV     $MXCNT,$TIMES     ;;SET NUMBER OF ITERATIONS TO DO
 047104                                   $SVLAD:
 047104  011667  132100                           MOV     (SP),$LPADR       ;;SAVE SCOPE LOOP ADDRESS
 047110  011667  132076                           MOV     (SP),$LPERR       ;;SAVE ERROR LOOP ADDRESS
 047114  005067  132206                           CLR     $ESCAPE           ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 047120  112767  000001  132071                   MOVB    #1,$ERMAX         ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 047126  105767  132052                   $OVER:  TSTB    $ERFLG            ;;ANY ERRORS?
 047132  001403                                   BEQ     1$                ;;BRANCH IF NO
 047134  116767  132044  132041                   MOVB    $ERFLG,$TSTNM+1
 047142  016777  132034  132074           1$:     MOV     $TSTNM,@DISPLAY   ;;DISPLAY TEST NUMBER
```

```
047150  016716  132034                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
047154  000002                          RTI                     ;;FIXES PS
047156  000010                  $MXCNT: 10                      ;;MAX. NUMBER OF ITERATIONS
                                .SBTTL  ERROR HANDLER ROUTINE

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

047160                          $ERROR:
047160  116737  132020  001203          MOVB    $ERFLG,@#$TSTNM+1
047166  105267  132012          7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
047172  001775                          BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
047174  016777  132002  132042          MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
047202  032777  002000  132032          BIT     #BIT10,@SWR     ;;BELL ON ERROR?
047210  001402                          BEQ     1$              ;;NO - SKIP
047212  104401  001330                  TYPE    ,$BELL          ;;RING BELL
047216  005267  131772          1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
047222  011667  131772                  MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
047226  162767  000002  131764          SUB     #2,$ERRPC
047234  117767  131760  131754          MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
047242  032777  020000  131772          BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
047250  001004                          BNE     20$             ;;SKIP TYPEOUTS
047252  004767  000056                  JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
047256  104401  001335                  TYPE    ,$CRLF
047262                          20$:
047262  005777  131754          2$:     TST     @SWR            ;;HALT ON ERROR
047266  100001                          BPL     3$              ;;SKIP IF CONTINUE
047270  000000                          HALT                    ;;HALT ON ERROR!
047272  032777  001000  131742  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
047300  001402                          BEQ     4$              ;;BR IF NO
047302  016716  131704                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
047306  005767  132014          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
047312  001402                          BEQ     5$              ;;BR IF NONE
047314  016716  132006                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
047320                          5$:
047320  022737  041230  000042          CMP     #$ENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
047326  001001                          BNE     6$              ;;BRANCH IF NO
047330  000000                          HALT                    ;;YES
047332                          6$:
047332  000002                          RTI                     ;;RETURN
                                ;;*************************************************************
                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE

                                ;*THIS ROUTINE FIRST TYPES A STANDARD MESSAGE CONSISTING OF THE
                                ;*VIRTUAL PC, THE PHYSICAL PC, THE PSW AT THE TIME OF THE ERROR CALL,
                                ;*AND THE  SUB-PASS COUNT. THE SUB-PASS COUNT CONSISTS OF THE SUB PASS COUNT IN THE
                                ;*HIGH BYTE AND THE PASS COUNT IN THE LOW BYTE.
```

```
                                        ;*
                                        ;*IT THEN USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                        ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE"
                                        ;*THE ERROR MESSAGE POINTER AND TYPES THE ERROR MESSAGE. THE DATA
                                        ;*HEADER POINTER IS THEN OBTAINED AND A D^TA HEADER IS TYPED.
                                        ;*THE DATA POINTER AND DATA FORMAT ARE THEN OBTAINED. THERE ARE
                                        ;*FOUR TYPES OF DATA FORMAT, AS FOLLOWS:
                                        ;*
                                        ;*      0       TYPE THE CONTENTS OF THE DATA TABLE WORD IN
                                        ;*              6 DIGIT OCTAL FORMAT
                                        ;*      1       CONVERT THE CONTENTS OF THE DATA TABLE WORD TO
                                        ;*              18 BITS AND TYPE AN 6 DIGIT OCTAL NUMBER
                                        ;*      2       TYPE THE CONTENTS OF THE DATA TABLE WORD AND
                                        ;*              THE WORD+2 IN 6 DIGIT OCTAL FORMAT
                                        ;*      3       USE THE CONTENTS OF THE DATA TABLE WORD AS A
                                        ;*              DEVICE ID AND TYPE THE DEVICES NAME
                                        ;*      4       CONVERT THE TWO WORDS POINTED TO BY THE DATA
                                        ;*              TABLE TO FLOATING POINT FORMAT AND TYPE.
                                        ;*      5       CONVERT THE FOUR WORDS POINTED TO BY THE DATA
                                        ;*              TABLE TO FLOATING DOUBLE FORMAT AND TYPE.
                                        ;;*********************************************************

047334  104406                          $ERRTYP:SAVREG
047336  104401  001335                          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
047342  004737  051010                          JSR     PC,@#TYPTIME    ;GO TYPE THE TIME
047346  104401  055306                          TYPE    ,MSG3
047352  104401  001335                          TYPE    ,$CRLF
047356  016746  131636                          MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                        ;;TYPE THE VIRTUAL PC
047362  104402                                  TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
047364  104401  050556                          TYPE    ,8$
0"7370  013700  001536                          MOV     @#VADR,R0               ;SAVE VADR
047374  013737  001220  001536                  MOV     @#$ERRPC,@#VADR         ;SAVE THE VIR PC FOR CONVERSION
047402  122737  000014  001216                  CMPB    #14,@#$ITEMB
047410  003403                                  BLE     51$
047412  105737  001216                          TSTB    @#$ITEMB                ;ERROR ZERO?
047416  001005                                  BNE     42$                     ;BRANCH IF NO
047420  004737  053724          51$:            JSR     PC,@#CNVADR             ;CONVERT TO 18 BITS
047424  010037  001536                          MOV     R0,@#VADR
047430  000407                                  BR      41$
047432  013737  001536  001540  42$:            MOV     @#VADR,@#PA1500
047440  005037  001542                          CLR     @#PA1716
047444  010037  001536                          MOV     R0,@#VADR
047450  012746  001540          41$:            MOV     #PA1500,-(SP)           ;PUT ADDRESS OFPC ON STACK
047454  004737  052060                          JSR     PC,@#$DB20              ;CONVERT TO ASCII
047460  062716  000005                          ADD     #5,(SP)                 ;GET RID OF 5 MS DIGITS
047464  012667  000002                          MOV     (SP)+,30$               ;SAVE POINTER TO ASCII
047470  104401                                  TYPE                            ;TYPE IT
047472  000000                  30$:            .WORD
047474  104401  050556                          TYPE    ,8$
047500  016646  000030                          MOV     30(SP),-(SP)            ;GET PSW AT TIME OF ERROR
047504  104402                                  TYPOC                           ;TYPE IT
047506  104401  050556                          TYPE    ,8$
047512  116746  131464                          MOVB    $TSTNM,-(SP)
04.516  105066  000001                          CLRB    1(SP)
047522  104402                                  TYPOC                   ;TYPE THE TEST NUMBER
```

```
047524  104401  050556          TYPE    ,8$
047530  013746  001574          MOV     @#SUBPASS,-(SP)
047534  162716  000060          SUB     #60,(SP)
047540  104402                  TYPOC
047542  104401  050556          TYPE    ,8$
047546  016746  131426          MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                                        ;;TYPE THE PASS COUNT
047552  104405                  TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
047554  104401  001335          TYPE    ,$CRLF
047560  005000                  CLR     R0
047562  153700  001216          BISB    @#$ITEMB,R0             ;PICK UP THE INDEX
047566  001431                  BEQ     6$                      ;EXIT IF ZERO
047570  022700  000007  1$:     CMP     #7,R0           ;IS THIS ERROR 7?
047574  001551                  BEQ     15$             ;BRANCH IF YES
047576  005300                  DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
047600  006300                  ASL     R0              ;;      WORK FOR THE ERROR TABLE
047602  006300                  ASL     R0
047604  006300                  ASL     R0
047606  062700  003126          ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
047612  012067  000004          MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
047616  001404                  BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
047620  104401                  TYPE                    ;;TYPE THE "ERROR MESSAGE"
047622  000000          2$:     .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
047624  104401  001335          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
047630  012067  000004  3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
047634  001404                  BEQ     5$              ;;SKIP TYPEOUT IF 0
047636  104401                  TYPE                    ;;TYPE THE "DATA HEADER"
047640  000000          4$:     .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
047642  104401  001335          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
047646  012001          5$:     MOV     (R0)+,R1                ;;PICKUP "DATA TABLE" POINTER
047650  001004                  BNE     7$              ;;GO TYPE THE DATA
047652  104407          6$:     RESREG
047654  104401  001335          TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
047660  000207                  RTS     PC              ;;RETURN
047662  011002          7$:     MOV     (R0),R2         ;GET "DATA FORMAT" POINTER
047664  122712  000001  10$:    CMPB    #1,(R2)         ;DATA FORMAT 1?
047670  001424                  BEQ     9$              ;BRANCH IF YES
047672  122712  000002          CMPB    #2,(R2)         ;DATA FORMAT 2?
047676  001441                  BEQ     11$             ;BRANCH IF YES
047700  122712  000003          CMPB    #3,(R2)         ;DATA FORMAT 3?
047704  001445                  BEQ     24$             ;BRANCH IF YES
047706  122712  000004          CMPB    #4,(R2)         ;DATA FORMAT 4?
047712  001456                  BEQ     40$             ;BRANCH IF YES
047714  122712  000005          CMPB    #5,(R2)         ;DATA FORMAT 5?
047720  001465                  BEQ     60$             ;BRANCH IF YES
                        ;;***********************************************************************
                        ;DATA FORMAT 0
047722  005202                  INC     R2              ;INCREMENT FORMAT POINTER
047724  013146                  MOV     @(R1)+,-(SP)    ;PUSH DATA TO BE TYPED
047726  104402                  TYPOC
047730  005711          13$:    TST     (R1)            ;ANY MORE DATA?
047732  001747                  BEQ     $               ;BRANCH IF NO
047734  104401  050556          TYPE    ,8$             ;TYPE TWO SPACES
047740  000751                  BR      10$
                        ;;***********************************************************************
                        ;DATA FORMAT 1
```

```
047742  005202                  9$:     INC     R2                      ;INCREMENT FORMAT POINTER
047744  004737  053724                  JSR     PC,@#CNVADR             ;GET 18 BIT ADR
047750  012746  001540          14$:    MOV     #PA1500,-(SP)           ;PUSH ADR OF 18 BIT ADR
047754  004737  052060                  JSR     PC,@#$DB20              ;CONVERT TO ASCII
047760  062716  000005                  ADD     #5,(SP)                 ;DELETE LEADING ZEROS
047764  012667  000002                  MOV     (SP)+,12$               ;GET ADR OF ASCII STRING
047770  104401                          TYPE
047772  000000                  12$:    .WORD
047774  062701  000002                  ADD     #2,R1                   ;INCREMENT R1
050000  000753                          BR      13$
                                ;;*******************************************************************
                                ;DATA FORMAT 2
050002  005202                  11$:    INC     R2                      ;INCREMENT FORMAT POINTER
050004  011100                          MOV     (R1),R0
050006  012037  001540                  MOV     (R0)+,@#PA1500
050012  011037  001542                  MOV     (R0),@#PA1716
050016  000754                          BR      14$
                                ;;*******************************************************************
                                ;DATA FORMAT 3
050020  005202                  24$:    INC     R2                      ;INCREMENT FORMAT POINTER
050022  013167  000016                  MOV     @(R1)+,25$              ;GET DEVICE ID
050026  062767  055236  000010          ADD     #MSGINX,25$             ;FORM ADR OF ASCIZ ADR
050034  017767  000004  000002          MOV     @25$,25$                ;GET ADR OF ASCIZ
050042  104401                          TYPE
050044  000000                  25$:    .WORD
050046  000730                          BR      13$                     ;CONTINUE
                                ;;*******************************************************************
                                ;DATA FORMAT 4
050050  005202                  40$:    INC     R2
050052  012167  000002                  MOV     (R1)+,44$               ;GET ADDRESS OF DATA
050056  104410                          FL20                            ;CONVERT TO FLOATING FORMAT
050060  000000                  44$:    .WORD
050062  012667  000002                  MOV     (SP)+,45$               ;GET ADDRESS OF ASCIZ STRING
050066  104401                          TYPE                            ;TYPE THE DATA
050070  000000                  45$:    .WORD
050072  000716                          BR      13$
                                ;;*******************************************************************
                                ;DATA FORMAT 5
050074  005202                  60$:    INC     R2                      ;INCREMENT FORMAT POINTER
050076  012167  000002                  MOV     (R1)+,61$               ;GET ADDRESS OF DATA
050102  104411                          FLD20                           ;CONVERT TO FLOATING ASCII
050104  000000                  61$:    .WORD
050106  012667  000002                  MOV     (SP)+,62$               ;GET ADDRESS OF ASCII STRING
050112  104401                          TYPE                            ;TYPE THE DATA
050114  000000                  62$:    .WORD
050116  000704                          BR      13$
                                ;;*******************************************************************
                                ;ERROR 7 DECODE
050120  010300                  15$:    MOV     R3,R0                   ;SAVE R3
050122  062700  055236                  ADD     #MSGINX,R0              ;GEN ADRS OF ASCIZ
050126  011067  000002                  MOV     (R0),16$
050132  104401                          TYPE
050134  000000                  16$:    .WORD
050136  104401  050144                  TYPE    ,65$                    ;;TYPE ASCIZ STRING
050142  000404                          BR      64$                     ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /FAILED/<CRLF>
```

```
050154                          64$:
050154  010300                          MOV     R3,R0           ;SAVE DEVICE ID
050156  022700  000010                  CMP     #10,R0          ;MASS BUS DEVICE?
050162  003403                          BLE     17$             ;BRANCH IF YES
050164  104401  055533                  TYPE    ,MSG12
050170  000411                          BR      18$
                                ;;********************************************************
                                ;MASS BUS ERR
050172  022703  000020          17$:    CMP     #20,R3          ;MBT ERROR?
050176  001426                          BEQ     26$             ;BRANCH IF MBT ERROR
050200  002435                          BLT     27$             ;BRANCH IF UBE ERROR
050202  104401  055646                  TYPE    ,MSG13
050206  022700  000012                  CMP     #12,R0          ;WAS IT RS?
050212  001131                          BNE     29$             ;BRANCH IF NO
                                ;;********************************************************
                                ;UNIBUS ERROR OR RS04 ERROR
050214  062700  055212          18$:    ADD     #REGINX,R0      ;FORM ADR OF REG TABLE
050220  011000                          MOV     (R0),R0         ;GET ADR OF REG TABLE
050222  022703  000002                  CMP     #2,R3           ;RP3 OR RK?
050226  001404                          BEQ     20$             ;BRANCH IF RK
050230  100406                          BMI     21$             ;BRANCH IF NOT RP03
050232  012704  000007                  MOV     #7,R4           ;SET RP03 SOB COUNT
050236  000423                          BR      22$
050240  012704  000006          20$:    MOV     #6,R4           ;SET RK05 SOB COUNT
050244  000420                          BR      22$
050246  012704  000011          21$:    MOV     #11,R4          ;SET RS04 SOB COUNT
050252  000415                          BR      22$
050254  104401  056035          26$:    TYPE    ,MSG16
050260  012704  000011                  MOV     #11,R4          ;SET MBT SOB COUNT
050264  062700  055212          28$:    ADD     #REGINX,R0
050270  011000                          MOV     (R0),R0         ;GET ADR OF MBT TABLE
050272  000405                          BR      22$             ;GO TYPE REGISTERS
050274  104401  056144          27$:    TYPE    ,MSG17
050300  012704  000004                  MOV     #4,R4           ;SET UBE SOB COUNT
050304  000767                          BR      28$             ;GO TYPE UBE REGISTERS
050306  013046                  22$:    MOV     @(R0)+,-(SP)    ;GET DATA IN REG
050310  104402                          TYPOC                   ;TYPE IT
050312  104401  050556                  TYPE    ,8$             ;TYPE TWO SPACES
050316  077405                          SOB     R4,22$          ;CONTINUE
                                ;;********************************************************
                                ;THIS CODE TYPES A PHYSICAL BUS ADDRESS IF THE ERROR WAS AN RP03 OR RK05
050320  022703  000022                  CMP     #22,R3          ;UBE ERROR?
050324  001452                          BEQ     73$             ;BRANCH IF YES
050326  022703  000002                  CMP     #2,R3           ;RK05?
050332  002443                          BLT     32$             ;BRANCH IF NOT RK OR RP03
050334  001005                          BNE     70$             ;BRANCH IF RP03
050336  104401  056340                  TYPE    ,MSG22
050342  012700  002202                  MOV     #RKCS,R0        ;GET ADR OF ADR OF RKCS REG
050346  000404                          BR      71$
050350  012700  002160          70$:    MOV     #RP3CS,R0       ;GET ADR OF ADR OF RP3CS REG
050354  104401  056350                  TYPE    ,MSG23
050360  013001                  71$:    MOV     @(R0)+,R1       ;GET BUS ADR EXTENDED BITS
050362  005720                          TST     (R0)+           ;ADJUST R0
050364  013037  001540                  MOV     @(R0)+,@#PA1500 ;GET BUS ADRESS THAT FAILED
050370  072127  177774                  ASH     #-4,R1          ;GET BITS 4&5 INTO BITS 0&1
050374  042701  177774                  BIC     #177774,R1      ;GET RID OF UNUSED BITS
```

```
050400  010137  001542                  MOV     R1,@#PA1716         ;SAVE EXTENDED BITS
050404  162737  000002  001540  74$:    SUB     #2,@#PA1500         ;DECREMENT BUS ADR
050412  005637  001542                  SBC     @#PA1716
050416  012746  001540                  MOV     #PA1500,-(SP)
050422  004737  052060                  JSR     PC,@#$DB20          ;CONVERT TO ASCIZ STRING
050426  062716  000003                  ADD     #3,(SP)             ;GET RID OF LEADING ZEROS
050432  012667  000002                  MOV     (SP)+,72$
050436  104401                          TYPE
050440  000000                  72$:    .WORD
050442  104401  001335          32$:    TYPE    ,$CRLF
050446  000167  177200                  JMP     6$                  ;EXIT
050452  012700  002306          73$:    MOV     #UBETBL+2,R0        ;GET ADR OF UBE TABLE +2
050456  013037  001540                  MOV     @(R0)+,@#PA1500     ;GET BUS ADR THAT FAILED
050462  013037  001542                  MOV     @(R0)+,@#PA1716     ;GET BAE BITS
050466  042737  177774  001542          BIC     #177774,@#PA1716;MASK OFF ADR BITS
050474  000743                          BR      74$
                                ;;*****************************************************************
                                ;RP04 ERROR
050476  062700  055212          29$:    ADD     #REGINX,R0
050502  011000                          MOV     (R0),R0             ;FORM ADR OF RP04 TABLE
050504  012704  000011                  MOV     #11,R4              ;SET SOB COUNT
050510  013046                  31$:    MOV     @(R0)+,-(SP)        ;GET DATA TO BE TYPED
050512  104402                          TYPOC                       ;TYPE DATA
050514  104401  050556                  TYPE    ,8$
050520  077405                          SOB     R4,31$              ;CONTINUE
050522  104401  001335                  TYPE    ,$CRLF
050526  104401  001335                  TYPE    ,$CRLF
050532  012704  000004                  MOV     #4,R4               ;SET SOB COUNT
050536  104401  055756                  TYPE    ,MSG14
050542  013046                  50$:    MOV     @(R0)+,-(SP)        ;GET DTA TO BE TYPED
050544  104402                          TYPOC                       ;TYPE IT
050546  104401  050556                  TYPE    ,8$
050552  077405                          SOB     R4,50$              ;CONTINUE
050554  000732                          BR      32$
050556  020040     000          8$:     .ASCIZ  /  /                ;;TWO(2) SPACES
        050562                          .EVEN
                                .SBTTL  TYPE ROUTINE

                                ;;*****************************************************************
                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                                ;*NOTE1:         $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                                ;*NOTE2:         $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                                ;*NOTE3:         $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
                                ;*
                                ;*CALL:
                                ;*1) USING A TRAP INSTRUCTION
                                ;*       TYPE    ,MESADR             ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                                ;*OR
                                ;*       TYPE
                                ;*       MESADR
                                ;*

050562  105767  130473          $TYPE:  TSTB    $TPFLG              ;;IS THERE A TERMINAL?
050566  100002                          BPL     1$                  ;;BR IF YES
050570  000000                          HALT                        ;;HALT HERE IF NO TERMINAL
```

```
050572  000407                           BR      3$                      ;;LEAVE
050574  010046                   1$:     MOV     R0,-(SP)                ;;SAVE R0
050576  017600  000002                   MOV     @2(SP),R0               ;;GET ADDRESS OF ASCIZ STRING
050602  112046                   2$:     MOVB    (R0)+,-(SP)             ;;PUSH CHARACTER TO BE TYPED ONTO STACK
050604  001005                           BNE     4$                      ;;BR IF IT ISN'T THE TERMINATOR
050606  005726                           TST     (SP)+                   ;;IF TERMINATOR POP IT OFF THE STACK
050610  012600                   60$:    MOV     (SP)+,R0                ;;RESTORE R0
050612  062716  000002           3$:     ADD     #2,(SP)                 ;;ADJUST RETURN PC
050616  000002                           RTI                             ;;RETURN
050620  122716  000011           4$:     CMPB    #HT,(SP)                ;;BRANCH IF <HT>
050624  001430                           BEQ     8$
050626  122716  000200                   CMPB    #CRLF,(SP)              ;;BRANCH IF NOT <CRLF>
050632  001006                           BNE     5$
050634  005726                           TST     (SP)+                   ;;POP  <CR><LF> EQUIV
050636  104401                           TYPE                            ;;TYPE A CR AND LF
050640  001335                           $CRLF
050642  105067  000136                   CLRB    $CHARCNT                ;;CLEAR CHARACTER COUNT
050646  000755                           BR      2$                      ;;GET NEXT CHARACTER
050650  004767  000056           5$:     JSR     PC,$TYPEC               ;;GO TYPE THIS CHARACTER
050654  126726  130400           6$:     CMPB    $FILLC,(SP)+            ;;IS IT TIME FOR FILLER CHARS.?
050660  001350                           BNE     2$                      ;;IF NO GO GET NEXT CHAR.
050662  016746  130370                   MOV     $NULL,-(SP)             ;;GET # OF FILLER CHARS. NEEDED
                                                                         ;;AND THE NULL CHAR.
050666  105366  000001           7$:     DECB    1(SP)                   ;;DOES A NULL NEED TO BE TYPED?
050672  002770                           BLT     6$                      ;;BR IF NO--GO POP THE NULL OFF OF STACK
050674  004767  000032                   JSR     PC,$TYPEC               ;;GO TYPE A NULL
050700  105367  000100                   DECB    $CHARCNT                ;;DO NOT COUNT AS A COUNT
050704  000770                           BR      7$                      ;;LOOP

                                 ;HORIZONTAL TAB PROCESSOR

050706  112716  000040           8$:     MOVB    #' ,(SP)                ;;REPLACE TAB WITH SPACE
050712  004767  000014           9$:     JSR     PC,$TYPEC               ;;TYPE A SPACE
050716  132767  000007  000060           BITB    #7,$CHARCNT             ;;BRANCH IF NOT AT
050724  001372                           BNE     9$                      ;;TAB STOP
050726  005726                           TST     (SP)+                   ;;POP SPACE OFF STACK
050730  000724                           BR      2$                      ;;GET NEXT CHARACTER
050732  005737  001530           $TYPEC: TST     @#NOTYPE                ;;INHIBIT TYPING?
050736  100423                           BMI     $TYPEX                  ;;BRANCH IF YES
050740  105777  130306                   TSTB    @$TPS                   ;;WAIT UNTIL PRINTER IS READY
050744  100372                           BPL     $TYPEC
050746  116677  000002  130300           MOVB    2(SP),@$TPB             ;;LOAD CHAR TO BE TYPED INTO DATA REG.
050754  122766  000015  000002           CMPB    #CR,2(SP)               ;;IS CHARACTER A CARRIAGE RETURN?
050762  001003                           BNE     1$                      ;;BRANCH IF NO
050764  105067  000014                   CLRB    $CHARCNT                ;;YES--CLEAR CHARACTER COUNT
050770  000406                           BR      $TYPEX                  ;;EXIT
050772  122766  000012  000002   1$:     CMPB    #LF,2(SP)               ;;IS CHARACTER A LINE FEED?
051000  001402                           BEQ     $TYPEX                  ;;BRANCH IF YES
051002  105227                           INCB    (PC)+                   ;;COUNT THE CHARACTER
051004  000000                   $CHARCNT:.WORD  0                       ;;CHARACTER COUNT STORAGE
051006  000207                   $TYPEX: RTS     PC

                                 ;;*******************************************************************
                                 .SBTTL  ROUTINE TO TYPE THE ELAPSED RUN TIME OF THE PROGRAM
                                 ;*      THIS ROUTINE CONVERTS THE CONTENTS OF LOCATIONS "LTICI
                                 ;*      AND "MTICKS" TO SECONDS AND MINUTES/HOURS RESPECTIVELY
```

```
                                              ;*      AND TYPES THEM IN THE FOLLOWING FORMAT:
                                              ;*              HHH:MM:SS
                                              ;;*****************************************************
051010  104406                                TYPTIME:SAVREG
051012  004737  054516                                JSR     PC,@#LDKT       ;GO BACK TO LOW CORE
051016  113701  001645                                MOVB    @#LTICKS+1,R1   ;GET SECOND COUNT
051022  005000                                        CLR     R0
051024  071027  000012                                DIV     #10.,R0
051030  062701  000060                                ADD     #60,R1
051034  110137  051246                                MOVB    R1,@#TIMEBUF+10
051040  010001                                        MOV     R0,R1
051042  005000                                        CLR     R0
051044  071027  000006                                DIV     #6,R0
051050  062701  000060                                ADD     #60,R1
051054  110137  051245                                MOVB    R1,@#TIMEBUF+7
051060  013701  001642                                MOV     @#MTICKS,R1     ;GET MINUTE COUNT
051064  005000                                        CLR     R0
051066  071027  000012                                DIV     #10.,R0         ;GET HOURS AND MINUTES
051072  062701  000060                                ADD     #60,R1          ;MAKE REMAINDER ASCII
051076  110167  000141                                MOVB    R1,TIMEBUF+5    ;PUT IN BUFFER
051102  010001                                        MOV     R0,R1
051104  005000                                        CLR     R0
051106  071027  000006                                DIV     #6.,R0
051112  062701  000060                                ADD     #60,R1
051116  110167  000120                                MOVB    R1,TIMEBUF+4
051122  005700                                        TST     R0
051124  001434                                        BEQ     2$
051126  010001                                        MOV     R0,R1
051130  005000                                        CLR     R0
051132  071027  000012                                DIV     #10.,R0
051136  062701  000060                                ADD     #60,R1
051142  110167  000072                                MOVB    R1,TIMEBUF+2
051146  005700                                        TST     R0
051150  001422                                        BEQ     2$
051152  010001                                        MOV     R0,R1
051154  005000                                        CLR     R0
051156  071027  000010                                DIV     #10,R0
051162  062701  000060                                ADD     #60,R1
051166  110167  000045                                MOVB    R1,TIMEBUF+1
051172  005700                                        TST     R0
051174  001410                                        BEQ     2$
051176  010001                                        MOV     R0,R1
051200  005000                                        CLR     R0
051202  071027  000012                                DIV     #10.,R0
051206  062701  000060                                ADD     #60,R1
051212  110167  000020                                MOVB    R1,TIMEBUF
051216  104401  051236                        2$:     TYPE    ,TIMEBUF
051222  104401  001335                                TYPE    ,$CRLF
051226  004737  054604                                JSR     PC,@#RESKT      ;GO BACK TO ORIGINAL MEMORY
051232  104407                                        RESREG
051234  000207                                        RTS     PC
051236     001     001     001                TIMEBUF:.BYTE   1,1,1,72,1,1,72,60,60,0
051241     072     001     001
051244     072     060     060
051247     000
                                                      .EVEN
```

```
                                        ;;*************************************************************
                                        .SBTTL  ROUTINE TO TYPE THE AVAILABLE DEVICES AND UNIT NUMBERS
                                        ;*      THIS ROUTINE SEARCHES THE SYSTEM SIZE TABLE FOR NON-
                                        ;*      ZERO ENTRIES.WHEN IT FINDS ONE, IT TYPES THE NAME OF THE
                                        ;*      DEVICE AND THE UNIT NUMBERS THAT WERE FOUND TO BE
                                        ;*      AVAILABLE FOR THAT DEVICE.
                                        ;;*************************************************************
051250  104401  057740                  TYPSIZ: TYPE    ,SWITCH
051254  104401  056401                          TYPE    ,MSG25
051260  013746  001532                          MOV     @#OPT.CP,-(SP)  ;PUT CONTENTS OF OPT.CP ON STACK
051264  104402                                  TYPOC                   ;TYPE OPT.CP
051266  104401  001335                          TYPE    ,$CRLF
051272  104401  055366                          TYPE    ,MSG4
051276  012700  000010                          MOV     #10,R0          ;SET SOB COUNT
051302  005001                                  CLR     R1
051304  105761  001646                  1$:     TSTB    SYSSIZE(R1)     ;DEVICE AVAILABLE?
051310  001004                                  BNE     2$              ;BRANCH IF YES
051312  062701  000002                  7$:     ADD     #2,R1           ;INCREMENT INDEX
051316  077006                                  SOB     R0,1$           ;CONTINUE
051320  000207                                  RTS     PC              ;RETURN
051322  010102                          2$:     MOV     R1,R2           ;GET INDEX
051324  062702  055236                          ADD     #MSGINX,R2      ;GET ADR OF MESSAGE ADR
051330  011267  000002                          MOV     (R2),3$         ;GET ADDRESS OF MESSAGE
051334  104401                                  TYPE
051336  000000                          3$:     .WORD
051340  112767  000060  000034                  MOVB    #60,4$          ;INIT UNIT NO. BUFFER (ASCII)
051346  116102  001646                          MOVB    SYSSIZE(R1),R2  ;GET WORD WITH AVAILABLE UNITS
051352  012703  000010                          MOV     #10,R3          ;SET SOB COUNT
051356  006002                          6$:     ROR     R2              ;GET UNITS
051360  103002                                  BCC     5$              ;BRANCH IF NOT A UNIT
051362  104401  051402                          TYPE    ,4$
051366  005267  000010                  5$:     INC     4$
051372  077307                                  SOB     R3,6$           ;CONTINUE
051374  104401  001335                          TYPE    ,$CRLF
051400  000744                                  BR      7$
051402     000     054     040          4$:     .BYTE   0,54,40,0       ;NUMBER,COMMA,SPACE,TERMINATOR
051405     000
                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

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

```
                                ;*
                                ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
                                ;*CALL:
                                ;*      MOV     NUM,-(SP)               ;;NUMBER TO BE TYPED
                                ;*      TYPOC                           ;;CALL FOR TYPEOUT

051406  017646  000000          $TYPOS: MOV     @(SP),-(SP)             ;;PICKUP THE MODE
051412  116667  000001  000211          MOVB    1(SP),$OFILL            ;;LOAD ZERO FILL SWITCH
051420  112667  000207                  MOVB    (SP)+,$OMODE+1          ;;NUMBER OF DIGITS TO TYPE
051424  062716  000002                  ADD     #2,(SP)                 ;;ADJUST RETURN ADDRESS
051430  000406                          BR      $TYPON
051432  112767  000001  000171  $TYPOC: MOVB    #1,$OFILL               ;;SET THE ZERO FILL SWITCH
051440  112767  000006  000165          MOVB    #6,$OMODE+1             ;;SET FOR SIX(6) DIGITS
051446  112767  000005  000154  $TYPON: MOVB    #5,$OCNT                ;;SET THE ITERATION COUNT
051454  010346                          MOV     R3,-(SP)                ;;SAVE R3
051456  010446                          MOV     R4,-(SP)                ;;SAVE R4
051460  010546                          MOV     R5,-(SP)                ;;SAVE R5
051462  116704  000145                  MOVB    $OMODE+1,R4             ;;GET THE NUMBER OF DIGITS TO TYPE
051466  005404                          NEG     R4
051470  062704  000006                  ADD     #6,R4                   ;;SUBTRACT IT FOR MAX. ALLOWED
051474  110467  000132                  MOVB    R4,$OMODE               ;;SAVE IT FOR USE
051500  116704  000125                  MOVB    $OFILL,R4               ;;GET THE ZERO FILL SWITCH
051504  016605  000012                  MOV     12(SP),R5               ;;PICKUP THE INPUT NUMBER
051510  005003                          CLR     R3                      ;;CLEAR THE OUTPUT WORD
051512  006105                  1$:     ROL     R5                      ;;ROTATE MSB INTO "C"
051514  000404                          BR      3$                      ;;GO DO MSB
051516  006105                  2$:     ROL     R5                      ;;FORM THIS DIGIT
051520  006105                          ROL     R5
051522  006105                          ROL     R5
051524  010503                          MOV     R5,R3
051526  006103                  3$:     ROL     R3                      ;;GET LSB OF THIS DIGIT
051530  105367  000076                  DECB    $OMODE                  ;;TYPE THIS DIGIT?
051534  100016                          BPL     7$                      ;;BR IF NO
051536  042703  177770                  BIC     #177770,R3              ;;GET RID OF JUNK
051542  001002                          BNE     4$                      ;;TEST FOR 0
051544  005704                          TST     R4                      ;;SUPPRESS THIS 0?
051546  001403                          BEQ     5$                      ;;BR IF YES
051550  005204                  4$:     INC     R4                      ;;DON'T SUPPRESS ANYMORE 0'S
051552  052703  000060                  BIS     #'0,R3                  ;;MAKE THIS DIGIT ASCII
051556  052703  000040          5$:     BIS     #' ,R3                  ;;MAKE ASCII IF NOT ALREADY
051562  110367  000040                  MOVB    R3,8$                   ;;SAVE FOR TYPING
051566  104401  051626                  TYPE    ,8$                     ;;GO TYPE THIS DIGIT
051572  105367  000032          7$:     DECB    $OCNT                   ;;COUNT BY 1
051576  003347                          BGT     2$                      ;;BR IF MORE TO DO
051600  002402                          BLT     6$                      ;;BR IF DONE
051602  005204                          INC     R4                      ;;INSURE LAST DIGIT ISN'T A BLANK
051604  000744                          BR      2$                      ;;GO DO THE LAST DIGIT
051606  012605                  6$:     MOV     (SP)+,R5                ;;RESTORE R5
051610  012604                          MOV     (SP)+,R4                ;;RESTORE R4
051612  012603                          MOV     (SP)+,R3                ;;RESTORE R3
051614  016666  000002  000004          MOV     2(SP),4(SP)             ;;SET THE STACK FOR RETURNING
051622  012616                          MOV     (SP)+,(SP)
051624  000002                          RTI                             ;;RETURN
051626     000                  8$:     .BYTE   0                       ;;STORAGE FOR ASCII DIGIT
051627     000                          .BYTE   0                       ;;TERMINATOR FOR TYPE ROUTINE
051630     000                  $OCNT:  .BYTE   0                       ;;OCTAL DIGIT COUNTER
```

```
051631     000                              $OFILL: .BYTE    0                   ;;ZERO FILL SWITCH
051632  000000                              $OMODE: .WORD    0                   ;;NUMBER OF DIGITS TO TYPE
                                            .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

                                            ;;*******************************************************
                                            ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
                                            ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
                                            ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
                                            ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
                                            ;*REPLACED WITH SPACES.
                                            ;*CALL:
                                            ;*       MOV     NUM,-(SP)           ;;PUT THE BINARY NUMBER ON THE STACK
                                            ;*       TYPDS                       ;;GO TO THE ROUTINE

051634                                      $TYPDS:
051634  010046                                      MOV     R0,-(SP)            ;;PUSH R0 ON STACK
051636  010146                                      MOV     R1,-(SP)            ;;PUSH R1 ON STACK
051640  010246                                      MOV     R2,-(SP)            ;;PUSH R2 ON STACK
051642  010346                                      MOV     R3,-(SP)            ;;PUSH R3 ON STACK
051644  010546                                      MOV     R5,-(SP)            ;;PUSH R5 ON STACK
051646  012746  020200                              MOV     #20200,-(SP)        ;;SET BLANK SWITCH AND SIGN
051652  016605  000020                              MOV     20(SP),R5           ;;GET THE INPUT NUMBER
051656  100004                                      BPL     1$                  ;;BR IF INPUT IS POS.
051660  005405                                      NEG     R5                  ;;MAKE THE BINARY NUMBER POS.
051662  112766  000055  000001                      MOVB    #'-,1(SP)           ;;MAKE THE ASCII NUMBER NEG.
051670  005000                              1$:     CLR     R0                  ;;ZERO THE CONSTANTS INDEX
051672  012703  052050                              MOV     #$DBLK,R3           ;;SETUP THE OUTPUT POINTER
051676  112723  000040                              MOVB    #' ,(R3)+           ;;SET THE FIRST CHARACTER TO A BLANK
051702  005002                              2$:     CLR     R2                  ;;CLEAR THE BCD NUMBER
051704  016001  052040                              MOV     $DTBL(R0),R1        ;;GET THE CONSTANT
051710  160105                              3$:     SUB     R1,R5               ;;FORM THIS BCD DIGIT
051712  002402                                      BLT     4$                  ;;BR IF DONE
051714  005202                                      INC     R2                  ;;INCREASE THE BCD DIGIT BY 1
051716  000774                                      BR      3$
051720  060105                              4$:     ADD     R1,R5               ;;ADD BACK THE CONSTANT
051722  005702                                      TST     R2                  ;;CHECK IF BCD DIGIT=0
051724  001002                                      BNE     5$                  ;;FALL THROUGH IF 0
051726  105716                                      TSTB    (SP)                ;;STILL DOING LEADING 0'S?
051730  100407                                      BMI     7$                  ;;BR IF YES
051732  106316                              5$:     ASLB    (SP)                ;;MSD?
051734  103003                                      BCC     6$                  ;;BR IF NO
051736  116663  000001  177777                      MOVB    1(SP),-1(R3)        ;;YES--SET THE SIGN
051744  052702  000060                      6$:     BIS     #'0,R2              ;;MAKE THE BCD DIGIT ASCII
051750  052702  000040                      7$:     BIS     #' ,R2              ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
051754  110223                                      MOVB    R2,(R3)+            ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
051756  005720                                      TST     (R0)+               ;;JUST INCREMENTING
051760  020027  000010                              CMP     R0,#10              ;;CHECK THE TABLE INDEX
051764  002746                                      BLT     2$                  ;;GO DO THE NEXT DIGIT
051766  003002                                      BGT     8$                  ;;GO TO EXIT
051770  010502                                      MOV     R5,R2               ;;GET THE LSD
051772  000764                                      BR      6$                  ;;GO CHANGE TO ASCII
051774  105726                              8$:     TSTB    (SP)+               ;;WAS THE LSD THE FIRST NON-ZERO?
051776  100003                                      BPL     9$                  ;;BR IF NO
052000  116663  177777  177776                      MOVB    -1(SP),-2(R3)       ;;YES--SET THE SIGN FOR TYPING
052006  105013                              9$:     CLRB    (R3)                ;;SET THE TERMINATOR
052010  012605                                      MOV     (SP)+,R5            ;;POP STACK INTO R5
```

```
052012  012603                          MOV     (SP)+,R3            ;;POP STACK INTO R3
052014  012602                          MOV     (SP)+,R2            ;;POP STACK INTO R2
052016  012601                          MOV     (SP)+,R1            ;;POP STACK INTO R1
052020  012600                          MOV     (SP)+,R0            ;;POP STACK INTO R0
052022  104401  052050                  TYPE    ,$DBLK              ;;NOW TYPE THE NUMBER
052026  016666  000002  000004          MOV     2(SP),4(SP)         ;;ADJUST THE STACK
052034  012616                          MOV     (SP)+,(SP)
052036  000002                          RTI                         ;;RETURN TO USER
052040  023420                  $DTBL:  10000.
052042  001750                          1000.
052044  000144                          100.
052046  000012                          10.
052050  000004                  $DBLK:  .BLKW   4
                                .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE

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


052060  104406                  $DB20:  SAVREG                      ;;SAVE ALL REGISTERS
052062  016601  000002                  MOV     2(SP),R1            ;;PICKUP THE POINTER TO LOW WORD
052066  012705  052215                  MOV     #$OCTVL+13.,R5      ;;POINTER TO DATA TABLE
052072  012704  000014                  MOV     #12.,R4             ;;DO ELEVEN CHARACTERS
052076  012703  177770                  MOV     #^C7,R3             ;;MASK
052102  012100                          MOV     (R1)+,R0            ;;LOWER WORD
052104  012101                          MOV     (R1)+,R1            ;;HIGH WORD
052106  005002                          CLR     R2                  ;;TERMINATOR
052110  110245                  1$:     MOVB    R2,-(R5)            ;;PUT CHARACTER IN DATA TABLE
052112  010002                          MOV     R0,R2               ;;GET THIS DIGIT
052114  005304                          DEC     R4                  ;;COUNT THIS CHARACTER
052116  003016                          BGT     3$                  ;;BR IF NOT THE LAST DIGIT
052120  001414                          BEQ     2$                  ;;BR IF IT IS THE LAST DIGIT
052122  005205                          INC     R5                  ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
052124  010566  000002                  MOV     R5,2(SP)            ;;ASCIZ CHAR. & PUT IT ON THE STACK
052130  122765  000061  000003          CMPB    #61,3(R5)           ;;LAST NUMBER LEGAL?
052136  002003                          BGE     4$                  ;;BRANCH IF YES
052140  112765  000060  000003          MOVB    #60,3(R5)           ;;MAKE IT A ZERO
052146  104407                  4$:     RESREG                      ;;RESTORE ALL REGISTERS
052150  000207                          RTS     PC                  ;;RETURN TO USER
052152  006203                  2$:     ASR     R3                  ;;POSITION THE MASK FOR THE LAST DIGIT
052154  006001                  3$:     ROR     R1                  ;;POSITION THE BINARY NUMBER FOR
052156  006000                          ROR     R0                  ;;      THE NEXT OCTAL DIGIT
052160  006001                          ROR     R1
052162  006000                          ROR     R0
052164  006001                          ROR     R1
052166  006000                          ROR     R0
052170  040302                          BIC     R3,R2               ;;MASK OUT ALL JUNK
052172  062702  000060                  ADD     #'0,R2              ;;MAKE THIS CHAR. ASCII
052176  000744                          BR      1$                  ;;GO PUT IT IN THE DATA TABLE
052200  000016                  $OCTVL: .BLKB   14.                 ;;RESERVE DATA TABLE
                                .SBTTL  SAVE AND RESTORE R0-R5 ROUTINES
```

```

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

052216                          $SAVREG:
052216  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
052220  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
052222  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
052224  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
052226  010446                          MOV     R4,-(SP)        ;;PUSH R4 ON STACK
052230  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
052232  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
052236  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
052242  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
052246  016646  000022                  MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
052252  000002                          RTI

                                ;*RESTORE R0-R5
                                ;*CALL:
                                ;*          RESREG
052254                          $RESREG:
052254  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
052260  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
052264  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
052270  012666  000022                  MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
052274  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
052276  012604                          MOV     (SP)+,R4        ;;POP STACK INTO R4
052300  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
052302  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
052304  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
052306  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
052310  000002                          RTI
                                ;;*************************************************************
                                .SBTTL  CONVERT FLOATING BINARY TO OCTAL ASCIZ
                                ;*
                                ;*THIS ROUTINE CONVERTS A 32 BIT FLOATING NUMBER TO AN OCTAL
                                ;*ASCIZ STRING IN THE FOLLOWING FORMAT:
                                ;*
                                ;*                   W  XXX  YYY ZZZZZZ
                                ;*
                                ;*        WHERE      W = SIGN BIT
                                ;*                   X = 8-BIT EXPONENT (RIGHT JUSTIFIED)
                                ;*                   Y = FRACTION BITS <57:51> (RIGHT JUSTIFIED)
                                ;*                   Z = FRACTION BITS <50:35>
```

```
                                   ;*
                                   ;*IT IS ENTERED BY A TRAP CALL WITH THE ADDRESS OF THE FLOATING
                                   ;*NUMBER IN THE WORD FOLLOWING THE CALL.
                                   ;*IT RETURNS WITH THE ADDRESS OF THE ASCIZ STRING ON THE STACK.
                                   ;;*********************************************************
052312  104406                     $FL20:  SAVREG
052314  017600  000000                     MOV     @(SP),R0            ;GET ADDRESS OF DATA
052320  062716  000002                     ADD     #2,(SP)             ;ADJUST RETURN PC
052324  016001  000002                     MOV     2(R0),R1            ;PUT SECOND DATA WORD IN R1
052330  011000                             MOV     (R0),R0             ;PUT FIRST DATA WORD IN R0
052332  012704  001365                     MOV     #$FLBUFF+23,R4      ;GET ADDRESS OF BUFFER END IN R4
052336  112744  000000                     MOVB    #0,-(R4)            ;PUT TERMINATOR IN BUFFER
052342  012705  000005                     MOV     #5,R5               ;SET SOB COUNT FOR FRACTION DIGITS
052346  010103                     1$:     MOV     R1,R3               ;GET LSB'S OF FRACTION
052350  042703  177770                     BIC     #^C7,R3             ;SAVE LS 3 BITS
052354  062703  000060                     ADD     #60,R3              ;MAKE THEM ASCII
052360  110344                             MOVB    R3,-(R4)            ;STORE IN BUFFER
052362  073027  177775                     ASHC    #-3,R0              ;SHIFT NUMBER TO NEXT 3 BITS
052366  077511                             SOB     R5,1$               ;CONTINUE FOR 7 DIGITS
052370  010103                             MOV     R1,R3               ;GET NEXT DIGITS
052372  042703  177776                     BIC     #^C1,R3             ;ONLY WANT 1 BIT
052376  062703  000060                     ADD     #60,R3              ;MAKE THEM ASCII
052402  110344                             MOVB    R3,-(R4)            ;STORE IN BUFFER
052404  112744  000040                     MOVB    #40,-(R4)           ;PUT SPACE IN BUFFER
052410  073027  177777                     ASHC    #-1,R0
052414  012705  000002                     MOV     #2,R5               ;SET SOB COUNT
052420  010103                     3$:     MOV     R1,R3               ;GET LOW WORD
052422  042703  177770                     BIC     #^C7,R3             ;MASK 3 BITS
052426  062703  000060                     ADD     #60,R3              ;MAKE THEM ASCII
052432  110344                             MOVB    R3,-(R4)            ;PUT IN BUFFER
052434  073027  177775                     ASHC    #-3,R0              ;GET NEXT 3 BITS
052440  077511                             SOB     R5,3$               ;CONVERT THEM
052442  010103                             MOV     R1,R3               ;
052444  042703  177776                     BIC     #^C1,R3             ;ONLY WANT 1 BIT
052450  062703  000060                     ADD     #60,R3              ;MAKE IT ASCII
052454  110344                             MOVB    R3,-(R4)            ;PUT IN BUFFER
052456  112744  000040                     MOVB    #40,-(R4)           ;PUT SPACE IN BUFFER
052462  112744  000040                     MOVB    #40,-(R4)
052466  072127  177777                     ASH     #-1,R1              ;GET FIRST 3 BITS OF EXPONENT
052472  012705  000002                     MOV     #2,R5               ;SET SOB COUNT FOR 2 DIGITS
052476  010103                     2$:     MOV     R1,R3               ;GET LSB'S OF EXPONENT
052500  042703  177770                     BIC     #^C7,R3             ;SAVE 3 BITS
052504  062703  000060                     ADD     #60,R3              ;MAKE THEM ASCII
052510  110344                             MOVB    R3,-(R4)            ;STORE IN BUFFER
052512  072127  177775                     ASH     #-3,R1              ;GET NEXT 3 BITS
052516  077511                             SOB     R5,2$               ;CONTINUE
052520  010103                             MOV     R1,R3               ;GET LAST 2 BITS OF EXPONENT
052522  042703  177774                     BIC     #^C3,R3             ;MAKE SURE ONLY 2 BITS
052526  062703  000060                     ADD     #60,R3              ;MAKE THEM ASCII
052532  110344                             MOVB    R3,-(R4)            ;STORE IN BUFFER
052534  112744  000040                     MOVB    #40,-(R4)           ;PUT SPACE IN BUFFER
052540  112744  000040                     MOVB    #40,-(R4)
052544  042700  177776                     BIC     #^C1,R0             ;GET SIGN BIT (IT WAS EXTENDED)
052550  062700  000060                     ADD     #60,R0              ;MAKE IT ASCII
052554  110044                             MOVB    R0,-(R4)            ;PUT IT IN THE BUFFER
052556  104407                             RESREG
```

```
052560  011646                          MOV     (SP),-(SP)          ;SAVE RETURN PC
052562  016666  000004  000002          MOV     4(SP),2(SP)         ;AND RETURN PSW
052570  012766  001342  000004          MOV     #$FLBUFF,4(SP)      ;PUT BUFFER ADDRESS ON STACK
052576  000006                          RTT                         ;RETURN
                                        .EVEN
                                ;;*************************************************************
                                .SBTTL  CONVERT FLOATING DOUBLE BINARY TO OCTAL ASCIZ
                                ;*
                                ;*THIS ROUTINE CONVERTS A 64 BIT FLOATING NUMBER TO AN OCTAL
                                ;*ASCIZ STRING IN THE FOLLOWING FORMAT:
                                ;*
                                ;*              U  VVV  WWW XXXXXX YYYYYY ZZZZZZ
                                ;*
                                ;*      WHERE   U = SIGN BIT
                                ;*              V = 8-BIT EXPONENT (RIGHT JUSTIFIED)
                                ;*              W = FRACTION BITS<57:51> (RIGHT JUSTIFIED)
                                ;*              X = FRACTION BITS <50:35>
                                ;*              Y = FRACTION BITS <34:19>
                                ;*              Z = FRACTION BITS <18:03>
                                ;*
                                ;*IT IS ENTERED BY A TRAP CALL WITH THE ADDRESS OF THE FLOATING
                                ;*NUMBER IN THE WORD FOLLOWING THE CALL.
                                ;*IT RETURNS WITH THE ADDRESS OF THE ASCIZ STRING ON THE STACK.
                                ;;*************************************************************
052600  104406                  $FLD20: SAVREG
052602  017667  000000  000006          MOV     @(SP),1$            ;GET ADDRESS OF DATA TO CONVERT
052610  062716  000002                  ADD     #2,(SP)             ;ADJUST RETURN PC
052614  104410                          FL20                        ;CONVERT MS 32 BITS
052616  000000                  1$:     .WORD
052620  012600                          MOV     (SP)+,R0            ;GET ADDRESS OF CONVERTED DATA
052622  010067  126560                  MOV     R0,$BUFF            ;SAVE IT
052626  062700  000041                  ADD     #41,R0              ;ADJUST TO END OF BUFFER
052632  105040                          CLRB    -(R0)               ;PUT TERMINATOR IN BUFFER
052634  016701  177756                  MOV     1$,R1               ;GET ADDRESS OF DATA TO CONVERT
052640  062701  000004                  ADD     #4,R1               ;ADJUST TO LOWER 32 BITS
052644  012102                          MOV     (R1)+,R2            ;SAVE THE DATA
052646  012103                          MOV     (R1)+,R3
052650  012701  000002                  MOV     #2,R1               ;SET LOOP COUNT
052654  012704  000005          3$:     MOV     #5,R4               ;SET LOOP COUNT
052660  010305                  4$:     MOV     R3,R5               ;GET LS 32 BITS OF DATA
052662  042705  177770                  BIC     #↑C7,R5             ;MASK 3 BITS
052666  062705  000060                  ADD     #60,R5              ;MAKE THEM ASCII
052672  110540                          MOVB    R5,-(R0)            ;PUT IN BUFFER
052674  073227  177775                  ASHC    #-3,R2              ;GET NEXT 3 BITS
052700  077411                          SOB     R4,4$               ;CONTINUE
052702  010305                          MOV     R3,R5               ;GET LS 32 BITS
052704  042705  177776                  BIC     #↑C1,R5             ;ONLY WANT 1 BIT
052710  062705  000060                  ADD     #60,R5              ;MAKE IT ASCII
052714  110540                          MOVB    R5,-(R0)            ;PUT IN TABLE
052716  112740  000040                  MOVB    #40,-(R0)           ;PUT SPACE IN TABLE
052722  073227  177777                  ASHC    #-1,R2
052726  077126                          SOB     R1,3$               ;CONVERT NEXT 16 BITS
052730  104407                          RESREG
052732  011646                          MOV     (SP),-(SP)          ;ADJUST STACK
052734  016666  000004  000002          MOV     4(SP),2(SP)         ;TO RETURN WITH ADDRESS
052742  016766  126440  000004          MOV     $BUFF,4(SP)         ;OF BUFFER ON STACK
```

```
052750  000006                                  RTT                             ;RETURN

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

                                .SBTTL  RANDOM NUMBER GENERATOR ROUTINE

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

052752                          $RAND:
052752  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
052754  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
052756  010246                          MOV     R2,-(SP)                ;;PUSH R2 ON STACK
052760  016700  126600                  MOV     $LONUM,R0               ;;SET R0 WITH LOW
052764  016701  126576                  MOV     $HINUM,R1               ;;SET R1 WITH HIGH
052770  012702  177771                  MOV     #-7,R2                  ;;SET SHIFT COUNT
052774  006300                  1$:     ASL     R0                      ;;SHIFT R0 LEFT AND
052776  006101                          ROL     R1                      ;;ROTATE CARRY INTO R1 AND
053000  005202                          INC     R2                      ;;CHECK FOR DONE
053002  001374                          BNE     1$                      ;;CONTINUE SHIFT LOOP
053004  066700  126554                  ADD     $LONUM,R0               ;;ADD NUMBER TO MAKE X 129
053010  005501                          ADC     R1                      ;;PROPOGATE CARRY
053012  066701  126550                  ADD     $HINUM,R1               ;;ADD NUMBER TO MAKE X 129
053016  062700  001057                  ADD     #1057,R0                ;;ADD LOW CONSTANT
053022  005501                          ADC     R1                      ;;PROPOGATE CARRY
053024  062701  047401                  ADD     #47401,R1               ;;ADD HIGH CONSTANT
053030  010067  126530                  MOV     R0,$LONUM               ;;SAVE R0
053034  010167  126526                  MOV     R1,$HINUM               ;;SAVE R1
053040  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
053042  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
053044  012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
053046  000207                          RTS     PC                      ;;RETURN
                                ;;*****************************************************************
                                .SBTTL  FLOATING POINT NUMBER GENERATOR
                                ;*      THIS ROUTINE GENERATES TWO RANDOM FLOATING POINT NUMBERS
                                ;*      IN EITHER SINGLE OR DOUBLE PRECISION. FOR SINGLE PRECISION
                                ;*      THE NUMBERS ARE STORED IN FTMP0 AND FTMP2. DOUBLE PRECISION
                                ;*      NUMBERS ARE STORED IN FTMP0 AND FTMP4.
                                ;*      IN EITHER SINGLE OR DOUBLE THE EXTENDED EXPONENT IS STORED
                                ;*      IN FREG0 AND FREG1.
                                ;;*****************************************************************
053050  012767  000002  000130  FLTDBL: MOV     #2,SOBDBL               ;SET LOOP FOR 2, FOUR WORD NUMBERS
053056  016700  000124          FLTSGL: MOV     SOBDBL,R0               ;SET WORD LENGTH LOOP
053062  012702  001424                  MOV     #FTMP0,R2               ;GET ADDRESS TO STORE WORDS IN
053066  012701  000002          2$:     MOV     #2,R1                   ;SET NUMBER OF WORDS TO 2
053072  004767  177654          1$:     JSR     PC,$RAND                ;GET RANDOM NUMBER
053076  022701  000002                  CMP     #2,R1                   ;FIRST TIME?
053102  001404                          BEQ     3$                      ;BRANCH IF YES
053104  022767  000002  000074          CMP     #2,SOBDBL               ;DOUBLE PRECISION?
053112  001407                          BEQ     4$                      ;BRANCH IF YES
053114  016703  126446          3$:     MOV     $HINUM,R3               ;GET EXPONENT PART
```

```
053120  042703  000177                          BIC     #177,R3             ;CHECK FOR MINUS ZERO
053124  022703  100000                          CMP     #BIT15,R3
053130  001760                                  BEQ     1$                  ;BRANCH IF MINUS ZERO
053132  016722  126430          4$:             MOV     $HINUM,(R2)+        ;SAVE HINUM
053136  016722  126422                          MOV     $LONUM,(R2)+        ;SAVE LONUM
053142  077125                                  SOB     R1,1$               ;CONTINUE
053144  077030                                  SOB     R0,2$               ;CONTINUE FOR DOUBLE PREC
053146  012746  001424                          MOV     #FTMPO,-(SP)        ;PUT ADDRESS OF NUMBER ON STACK
053152  012746  001002                          MOV     #1002,-(SP)         ;PUT CONTROL WORD ON STACK
053156  022767  000002  000022                  CMP     #2,SOBDBL           ;DOUBLE PREC?
053164  001002                                  BNE     5$                  ;BRANCH IF NO
053166  012716  001004                          MOV     #1004,(SP)          ;CHANGE CONTROL WORD
053172  004767  000012          5$:             JSR     PC,EXPEXT           ;CALCULATE EXT EXPONENTS
053176  012767  000001  000002                  MOV     #1,SOBDBL           ;INIT SOBDBL FOR SINGLE PREC
053204  000207                                  RTS     PC                  ;RETURN
053206  000001                  SOBDBL: .WORD   1
                                ;;*****************************************************************
                                .SBTTL  FLOATING POINT EXPONENT EXTENSION
                                ;*      THIS ROUTINE CONVERTS THE ACTUAL EXPONENT OF A FLOATING POINT
                                ;*      NUMBER INTO AN ACTUAL EXPONENT OF 200 AND AN EXTENDED
                                ;*      EXPONENT EQUAL TO THE DIFFERANCE BETWEEN THE ORIGINAL
                                ;*      ACTUAL EXPONENT AND 200.
                                ;*
                                ;*      THE ROUTINE IS ENTERED WITH A CONTROL WORD ON THE STACK.
                                ;*      BIT 15 OF THE CONTROL WORD INDICATES WHETHER THE NUMBER
                                ;*      IS IN MEMORY (<15>=0) OR IN AN ACCUMULATOR (<15>=1).
                                ;*      IF THE NUMBER IS IN AN ACCUMULATOR, BITS <9:8> INDICATE
                                ;*      THE ACCUMULATOR NUMBER. IF THE NUMBER(S) IS IN MEMORY,
                                ;*      BITS <9:8> INDICATE THE NUMBER OF NUMBERS TO CONVERT AND
                                ;*      BITS <2:0> INDICATE THE WORD LENGTH OF THE NUMBER(S).
                                ;*      IN THE CASE OF A MEMORY CONVERSION, THE ADDRESS OF THE
                                ;*      FIRST WORD TO CONVERT IS ALSO ON THE STACK (PRECEDING
                                ;*      THE CONTROL WORD).
                                ;;*****************************************************************
053210  012605                  EXPEXT: MOV     (SP)+,R5            ;SAVE RETURN PC
053212  012600                          MOV     (SP)+,R0            ;GET CONTROL WORD
053214  100437                          BMI     1$                  ;BRANCH IF ACC CONVERSION
053216  012601                          MOV     (SP)+,R1            ;GET START ADDRESS
053220  162700  000400                  SUB     #400,R0
053224  012702  001424                  MOV     #FTMPO,R2           ;GET OFFSET FROM FTMPO
05[illegible]30  160102                          SUB     R1,R2
053232  005402                          NEG     R2
053234  006202                          ASR     R2
053236  062702  001444                  ADD     #FREGO,R2           ;GEN ADDRESS OF EXT WORD
053242  011103                  3$:     MOV     (R1),R3             ;GET DATA
053244  042703  100177                  BIC     #100177,R3          ;GET EXPONENT
053250  072327  177771                  ASH     #-7,R3              ;RIGHT JUSTIFY EXPONENT
053254  162703  000200                  SUB     #200,R3             ;CONVERT TO 2'S COMPLIMENT
053260  010312                          MOV     R3,(R2)             ;ADD TO EXTENDED EXPONENT
053262  042711  077600                  BIC     #77600,(R1)         ;MAKE ACTUAL
053266  052711  040000                  BIS     #BIT14,(R1)         ;EXPONENT 200
053272  162700  000400                  SUB     #400,R0             ;ANY MORE WORDS?
053276  100435                          BMI     2$                  ;BRANCH IF NO
053300  110003                          MOVB    R0,R3               ;GET WORD LENGTH
053302  006303                          ASL     R3
053304  060301                          ADD     R3,R1               ;SELECT NEXT NUMBER ADDRESS
```

```
053306  062702  000002                  ADD     #2,R2               ;SELECT NEXT EXTENDED ADDRESS
053312  000753                          BR      3$                  ;CONTINUE
                                ;ACCUMULATOR CONVERSION
053314  072027  177776          1$:     ASH     #-2,R0              ;GET ACCUMULATOR NUMBER
053320  042700  177477                  BIC     #177477,R0          ;
053324  010002                          MOV     R0,R2               ;GENERATE
053326  072227  177773                  ASH     #-5,R2              ;ADDRESS OF
053332  062702  001410                  ADD     #$AC0,R2            ;EXTENDED EXPONENT
053336  042767  000300  000004          BIC     #300,5$             ;GENERATE INSTRUCTION
053344  050067  000000                  BIS     R0,5$               ;TO GET EXPONENT
053350  175003                  5$:     STEXP   AC0,R3              ;GET EXPONENT
053352  060312                          ADD     R3,(R2)             ;ADD TO EXTENDED EXPONENT
053354  005003                          CLR     R3
053356  042767  000300  000004          BIC     #300,4$             ;GENERATE INSTRUCTION
053364  050067  000000                  BIS     R0,4$               ;TO LOAD EXPONENT BACK TO ACC
053370  176403                  4$:     LDEXP   R3,AC0              ;LOAD EXPONENT OF 200
053372  010546                  2$:     MOV     R5,-(SP)            ;RESTORE RETURN PC
053374  000207                          RTS     PC                  ;RETURN
                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;;*****************************************************************
                                ;POWER DOWN ROUTINE
053376  012737  053550  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC    ;;SET FOR FAST UP
053404  012737  000340  000026          MOV     #340,@#PWRVEC+2     ;;PRIO:7
053412  010046                          MOV     R0,-(SP)            ;;PUSH R0 ON STACK
053414  010146                          MOV     R1,-(SP)            ;;PUSH R1 ON STACK
053416  010246                          MOV     R2,-(SP)            ;;PUSH R2 ON STACK
053420  010346                          MOV     R3,-(SP)            ;;PUSH R3 ON STACK
053422  010446                          MOV     R4,-(SP)            ;;PUSH R4 ON STACK
053424  010546                          MOV     R5,-(SP)            ;;PUSH R5 ON STACK
053426  017746  125610                  MOV     @SWR,-(SP)          ;;PUSH @SWR ON STACK
053432  010667  000116                  MOV     SP,$SAVR6           ;;SAVE SP
053436  012737  053450  000024          MOV     #$PWRUP,@#PWRVEC    ;;SET UP VECTOR
053444  000000                          HALT
053446  000776                          BR      .-2                 ;;HANG UP

                                ;;*****************************************************************
                                ;POWER UP ROUTINE
053450  012737  053550  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC    ;;SET FOR FAST DOWN
053456  016706  000072                  MOV     $SAVR6,SP           ;;GET SP
053462  005067  000066                  CLR     $SAVR6              ;;WAIT LOOP FOR THE TTY
053466  005267  000062          1$:     INC     $SAVR6              ;;WAIT FOR THE INC
053472  001375                          BNE     1$                  ;;OF  WORD
053474  011600                          MOV     (SP),R0             ;GET THE OLD SW. REG. VALUE
053476  076600                          MED                         ;WRITE THE CONSOLE SW. REG.
053500  000226                          WCNSSW                      ;  WITH ITS PREVIOUS VALUE
053502  012677  125534                  MOV     (SP)+,@SWR          ;;POP STACK INTO @SWR
053506  012605                          MOV     (SP)+,R5            ;;POP STACK INTO R5
053510  012604                          MOV     (SP)+,R4            ;;POP STACK INTO R4
053512  012603                          MOV     (SP)+,R3            ;;POP STACK INTO R3
053514  012602                          MOV     (SP)+,R2            ;;POP STACK INTO R2
053516  012601                          MOV     (SP)+,R1            ;;POP STACK INTO R1
053520  012600                          MOV     (SP)+,R0            ;;POP STACK INTO R0
053522  012737  053376  000024          MOV     #$PWRDN,@#PWRVEC    ;;SET UP THE POWER DOWN VECTOR
053530  012737  000340  000026          MOV     #340,@#PWRVEC+2     ;;PRIO:7
053536  104401                          TYPE                        ;REPORT THE POWER FAILURE
```

```
053540  053556                          $PWRMG: .WORD   $POWER          ;;POWER FAIL MESSAGE POINTER
053542  012716                                  MOV     (PC)+,(SP)      ;;RESTART AT START
053544  003276                          $PWRAD: .WORD   START           ;;RESTART ADDRESS
053546  000002                                  RTI
053550  000000                          $ILLUP: HALT                    ;;THE POWER UP SEQUENCE WAS STARTED
053552  000776                                  BR      .-2             ;; BEFORE THE POWER DOWN WAS COMPLETE
053554  000000                          $SAVR6: 0                       ;;PUT THE SP HERE
053556  005015  047520  042527          $POWER: .ASCIZ  <15><12>"POWER"
053564  000122
                                                .EVEN
                                        .SBTTL  TRAP DECODER

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

053566  010046                          $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
053570  016600  000002                          MOV     2(SP),R0        ;;GET TRAP ADDRESS
053574  005740                                  TST     -(R0)           ;;BACKUP BY 2
053576  111000                                  MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
053600  006300                                  ASL     R0              ;;POSITION FOR INDEXING
053602  016000  053622                          MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
053606  000200                                  RTS     R0              ;;GO TO ROUTINE


                                        ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

053610  011646                          $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
053612  016666  000004  000002                  MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
053620  000002                                  RTI                     ;;RESTORE THE PSW

                                        .SBTTL  TRAP TABLE

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

                                        ;       ROUTINE
                                                -------
053622  053610                          $TRPAD: .WORD   $TRAP2
053624  050562                                  $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
053626  051432                                  $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
053630  051406                                  $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
053632  051446                                  $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
053634  051634                                  $TYPDS  ;;CALL=TYPDS    TRAP+5(104405)  TYPE DECIMAL NUMBER (WITH SIGN)


053636  052216                                  $SAVREG ;;CALL=SAVREG   TRAP+6(104406)  SAVE R0-R5 ROUTINE
053640  052254                                  $RESREG ;;CALL=RESREG   TRAP+7(104407)  RESTORE R0-R5 ROUTINE
053642  052312                                  $FL20   ;;CALL=FL20     TRAP+10(104410)
053644  052600                                  $FLD20  ;;CALL=FLD20    TRAP+11(104411)
                                        ;;*************************************************************
                                        .SBTTL  UNIBUS EXERCISER INITIALIZATION ROUTINE
                                        ;*THIS ROUTINE INITIALIZES THE BASE ADDRESS FOR THE
                                        ;*UNIBUS EXERCISER AND LOADS UP THE EXERCISER REGISTERS.
```

```
                                        ;;************************************************************
053646  012703  001716                  UBEINIT:MOV     #UBESAV,R3          ;GET ADDRESS OF NEXT UBE ADRES
053652  005023                                  CLR     (R3)+               ;INITIALIZE
053654  005023                                  CLR     (R3)+
053656  005023                                  CLR     (R3)+
053660  005013                                  CLR     (R3)


                                        ;SET UP THE UBE AND START IT
053662  012702  002304                          MOV     #UBETBL,R2          ;GET ADDRESS OF UBE TABLE
053666  005072  000010                          CLR     @10(R2)             ;CLEAR ALL ERRORS
053672  012772  045730  000012                  MOV     #UBESRV,@12(R2)     ;SET UP UBE VECTOR
053700  012772  000340  000014                  MOV     #PR7,@14(R2)        ;SET UP UBE VECTOR PSW
053706  012732  170000                          MOV     #170000,@(R2)+      ;SET CC FOR 2K WORD TRANSFER
                                                                            ;UBE IS DOING BYTE TRANSFERS
053712  013732  001722                          MOV     @#UBEADR,@(R2)+     ;LOAD UBE BUS ADDRESS
053716  013732  001724                          MOV     @#UBEADR+2,@(R2)+   ;LOAD ADR BITS 16 & 17
053722  000207                                  RTS     PC                  ;RETURN
```

```
                                        ;;*********************************************************
                                        .SBTTL  CONVERT A VIRTUAL ADDRESS TO A PHYSICAL ADDRESS
                                        ;*      THIS ROUTINE CONVERTS A 16-BIT VIRTUAL ADDRESS TO A
                                        ;*      18-BIT PHYSICAL ADDRESS. THE VIRTUAL ADDRESS IS
                                        ;*      ASSUMED TO BE IN LOCATION "VADR" AND THE PHYSICAL
                                        ;*      ADDRESS IS PLACED IN LOCATIONS "PA1716" AND "PA1500".
                                        ;*
                                        ;*      IF MEMORY MANAGEMENT IS OFF THE PHYSICAL ADDRESS IS
                                        ;*      GENERATED BY SUBTRACTING THE CONTENTS OF LOCATION
                                        ;*      "$FACTOR" FROM THE VIRTUAL ADDRESS. THIS LOCATION
                                        ;*      CONTAINS THE BYTE OFFSET BETWEEN THE RELOCATED CODE
                                        ;*      AND THE NON-RELOCATED CODE.
                                        ;*
                                        ;*      IF MEMORY MANAGEMENT IS ON, THE CONTENTS OF THE
                                        ;*      APPROPRIATE PAR REGISTER IS ADDED(AFTER ADJUSTMENT)
                                        ;*      TO THE LEAST SIGNIFICANT 13 BITS OF THE VIRTUAL ADDRESS.
                                        ;;*********************************************************
053724  104406                          CNVADR: SAVREG
053726  013703  001536                          MOV     @#VADR,R3               ;GET VIRTUAL ADDRESS TO CONVERT
053732  105737  001545                          TSTB    @#MMON                  ;IS MEMORY MGMT ON?
053736  001426                                  BEQ     1$                      ;BRANCH IF NO
053740  005002                                  CLR     R2
053742  073227  000003                          ASHC    #3,R2                   ;GET PAR SELECT BITS
053746  072327  177775                          ASH     #-3,R3                  ;RETURN VIR ADDR TO ORIGINAL
053752  042703  160000                          BIC     #160000,R3              ;MAKE SURE SIGN DIDN'T EXTEND
053756  006102                                  ROL     R2                      ;MAKE R2 EVEN FOR WORD ADDRESSING
053760  062702  172340                          ADD     #KIPAR0,R2              ;GET ADDRESS OF PAR
053764  011205                                  MOV     (R2),R5                 ;GET PAR DATA
053766  005004                                  CLR     R4                      ;SETUP R4
053770  073427  000006                          ASHC    #6,R4                   ;SHIFT PAR DATA
053774  060305                                  ADD     R3,R5                   ;FORM PHYSICAL ADDRESS
053776  005504                                  ADC     R4
054000  010437  001542                  2$:     MOV     R4,@#PA1716             ;SAVE PHYSICAL
054004  010537  001540                          MOV     R5,@#PA1500             ;ADDRESS
054010  104407                                  RESREG
054012  000207                                  RTS     PC                      ;RETURN
054014  163703  001550                  1$:     SUB     @#FACTOR,R3             ;FORM PHYSICAL ADDRESS
054020  005004                                  CLR     R4
054022  010305                                  MOV     R3,R5
054024  000765                                  BR      2$                      ;RETURN

                                        ;;*********************************************************
                                        .SBTTL  ROUTINE TO CHECK RELOCATED DATA
                                        ;*ROUTINE TO CHECK DATA RELOCATED
                                        ;*CALL: R0=     HIGHEST ADDRESS +2 OF SOURCE DATA
                                        ;*      R2=     HIGHEST ADDRESS +2 OF DEST DATA
                                        ;*      R5=     LOWEST ADDRESS OF THE SOURCE DATA
                                        ;*
                                        ;*      THIS ROUTINE USES A COMPARE INSTRUCTION TO CHECK
                                        ;*      THE DATA THAT WAS RELOCATED. IF A PARITY ERROR OCCURS
                                        ;*      DURING THIS CHECK A SPECIAL ERROR MESSAGE IS TYPED
                                        ;*      INSTED OF THE UNEXPECTED TRAP MESSAGE.
                                        ;;*********************************************************
054026  012737  054070  000114          CHKDAT: MOV     #2$,@#CACHVEC           ;SETUP PARITY VECTOR.
054034  024042                                  CMP     -(R0),-(R2)             ;CHECK DATA
```

```
054036 001012                              BNE     99$
054040 005112                              COM     (R2)                         ;COMPLEMENT DEST DATA
054042 005112                              COM     (R2)                         ;TWICE
054044 021210                              CMP     (R2),(R0)                    ;CHECK DATA
054046 001006                              BNE     99$
054050 020005                      1$:     CMP     R0,R5                        ;BRANCH IF ALL DATA NOT CHECKED
054052 001365                              BNE     CHKDAT
054054 012737 054414 000114                MOV     #.PARSRV,@#CACHVEC                   ;RESTORE CACHVEC
054062 000207                              RTS     PC                           ;RETURN
054064 000262                      99$:    SEV
054066 000207                              RTS     PC
054070 013737 177744 001740        2$:     MOV     @#MEMERR,@#REPREG                    ;SAVE ERROR REG
054076 010237 001536                       MOV     R2,@#VADR
054102 010037 001750                       MOV     R0,@#SRCADR
054106 104005                              ERROR   5
054110 000765                              BR      99$                                  ;RETURN


                                   ;;*****************************************************
                                   .SBTTL  ROUTINE TO CLEAR 'T' BIT
                                   ;;*****************************************************
054112 013746 177776               CLRTBIT:MOV     @#PSW,-(SP)       ;PUSH PSW ONTO STACK
054116 011627                              MOV     (SP),(PC)+        ;SAVE IN RETPSW BELOW
054120 000000                      RETPSW: .WORD   0
054122 042716 000020                       BIC     #20,(SP)          ;CLEAR T BIT IN PSW ON STACK
                                   ;;*****************************************************
                                   .SBTTL  ROUTINE TO RESTORE THE T BIT
                                   ;;*****************************************************
054126 012746 054134               RESPSW: MOV     #1$,-(SP)         ;SET RETURN PC FOR RTI
054132 000002                              RTI                       ;CLEAR 'T' BIT IN PSW
054134 000207                      1$:     RTS     PC                ;RETURN

054136 042737 177400 177776        RESTPS: BIC     #177400,@#PSW     ;SET KERNEL MODE
054144 016746 177750                       MOV     RETPSW,-(SP)      ;PUSH ORIG PSW ONTO STACK
054150 000766                              BR      RESPSW

                                   ;;*****************************************************
                                   .SBTTL  KEYBOARD INT SERV ROUTINE
                                   ;*THIS ROUTINE HANDLES INTERRUPTS FROM THE KEYBOARD
                                   ;*
                                   ;*TYPING A CONTROL 'C' WILL CAUSE THE PROCESSOR TO  HALT
                                   ;*
                                   ;*TYPING A CARRAGE RETURN WILL CAUSE A CARRIAGE RETURN-LINE FEED
                                   ;*TO BE TYPED.
                                   ;*
                                   ;*TYPING A CONTROL 'O' WILL INHIBIT ANY FURTHER TYPEOUT. THE SECOND CONTROL 'O'
                                   ;*WILL ENABLE TYPEOUT AGAIN AND ECHO A CR-LF.
                                   ;*
                                   ;*ANY OTHER CHARACTER WILL JUST BE ECHOED.
                                   ;;*****************************************************

       000003                              CNTRLC=3
       000017                              CNTRLO=17

054152 017746 125072               TKISR:  MOV     @$TKB,-(SP)                  ;GET CHARACTER
054156 042716 177600                       BIC     #177600,(SP)                 ;STRIP UNUSED BITS
```

```
054162  022716  000003                  CMP     #CNTRLC,(SP)            ;BRANCH IF NOT CONTROL C (↑C)
054166  001010                          BNE     1$
054170  012737  001334  001276          MOV     #$CRLF-1,@#$REGS        ;ECHO CR LF
054176  106277  125050                  ASRB    @$TPS
054202  005726                          TST     (SP)+                   ;POP CHARACTER OFF THE STACK
054204  000000                          HALT
054206  000002                          RTI                             ;RETURN

054210  122716  000015          1$:     CMPB    #15,(SP)                ;BRANCH IF NOT <CR>
054214  001007                          BNE     2$
054216  012737  001334  001276          MOV     #$CRLF-1,@#$REGS        ;ECHO CR LF
054224  106277  125022                  ASRB    @$TPS
054230  005726                          TST     (SP)+                   ;POP CHARACTEROFF STACK
054232  000002                          RTI                             ;RETURN

054234  122716  000017          2$:     CMPB    #CNTRLO,(SP)            ;BRANCH IF NOT CONTROL O (↑O)
054240  001012                          BNE     3$
054242  005726                          TST     (SP)+
054244  005167  125260                  COM     NOTYPE
054250  100405                          BMI     7$
054252  012737  001334  001276          MOV     #$CRLF-1,@#$REGS        ;ECHO CR LF
054260  106277  124766                  ASRB    @$TPS
054264  000002                  7$:     RTI

054266  104406                  3$:     SAVREG
054270  011605                          MOV     (SP),R5         ;RETRIEVE CHARACTER
054272  004737  054516                  JSR     PC,@#LDKT               ;GO TO LOW CORE
054276  013700  001504                  MOV     @#TKBFRP,R0             ;GET BUFFER PTR
054302  110520                  4$:     MOVB    R5,(R0)+                ;LOAD CHAR INTO BFR
054304  105010                          CLRB    (R0)                    ;CLEAR NEXT LOC
054306  022700  001526          5$:     CMP     #TKBFR+20,R0            ;BRANCH IF NOT END OF BFR
054312  001002                          BNE     6$
054314  012700  001506                  MOV     #TKBFR,R0               ;RESET BUFFER PTR
054320  010037  001504          6$:     MOV     R0,@#TKBFRP             ;RESTORE BFR PTR
054324  004737  054604                  JSR     PC,@#RESKT              ;GO BACK TO ORIGINAL MEMORY
054330  104407                          RESREG
054332  005737  001530          ECHO:   TST     @#NOTYPE                ;TYPEOUT DISABLED?
054336  100004                          BPL     1$                      ;BRANCH IF NO
054340  005726                          TST     (SP)+                   ;FIX UP STACK
054342  105077  124704                  CLRB    @$TPS                   ;CLEAR IE BIT
054346  000002                          RTI                             ;RETURN
054350  105777  124676          1$:     TSTB    @$TPS           ;PRINTER READY?
054354  100375                          BPL     .-4                     ;BRANCH IF NO
054356  112677  124672                  MOVB    (SP)+,@$TPB             ;MOVE CHAR TO PRINTER
054362  000002                          RTI                             ;RETURN



                                ;;*************************************************************
                                .SBTTL  TELETYPE INTERRUPT SERVICE ROUTINE
                                ;*THIS ROUTINE TYPES A MESSAGE POINTED TO BY THE ACR STORED
                                ;*IN LOCATION $REGS. THIS ROUTINE IS INTERRUPT DRIVEN.
                                ;;*************************************************************
054364  005237  001276          TPISR:  INC     @#$REGS         ;STEP MESSAGE ADDRESS PTR
054370  117746  124702                  MOVB    @$REGS,-(SP)            ;GET CHAR TO BE TYPED
054374  001356                          BNE     ECHO                    ;GO TYPE CHAR IF NOT '0'
054376  005726                          TST     (SP)+                   ;POP STACK
```

```
054400  005077  124646                         CLR     @$TPS                   ;CLEAR IE BIT
054404  012737  001570  001276                 MOV     #NULLS,@#$REG5
054412  000002                                 RTI                             ;RETURN

                                       ;;*****************************************************************
                                       .SBTTL  PARITY ERROR SERVICE
                                       ;*      THIS ROUTINE FIELDS UNEXPECTED TRAPS TO 114. IT IS ASSUMED
                                       ;*      THAT THE ERROR WAS IN CACHE AND WAS CAUSED BY THE "OTHER
                                       ;*      WORD" RATHER THAN THE "WANTED WORD" WHICH MEANS THAT THE
                                       ;*      BAD DATA IS STILL IN THE CACHE. SO, TO CLEAR THE BAD DATA
                                       ;*      THE ERROR ADDRESS IS REFERENCED CAUSING THE CACHE TO GO
                                       ;*      TO MAIN MEMORY TO GET THE DATA. THIS PREVENTS AN
                                       ;*      ARBITRARY REFERENCE TO THE BAD WORD FROM TRAPPING.
                                       ;*
                                       ;*      AFTER THE ERROR IS REPORTED, BITS 2 AND 3 OF THE MEMORY
                                       ;*      ERROR REGISTER ARE TESTED TO SEE IF THE BAD DATA IS IN
                                       ;*      MAIN MEMORY. IF IT IS, THE PROGRAM RESTARTS SINCE THE
                                       ;*      GOOD DATA IS NOW LOST FOREVER. OTHERWISE THE PROGRAM
                                       ;*      RETURNS TO THE ADDRESS POINTED TO BY "$LPERR".
                                       ;;*****************************************************************
054414                                 PARSRV:
054414  005227                                 INC     (PC)+           ;MAKE FLAG ZERO FIRST TIME THRU
054416  177777                         PARFLG: .WORD   -1              ;NEGATIVE ONE FOR A FLAG
054420  001401                                 BEQ     5$              ;BRANCH IF FIRST TIME IN
054422  000000                                 HALT                    ;I HAVE ENTERED THIS ROUTINE BEFORE
                                                                       ;I FINISHED REPORTING THE FIRST ERROR
                                                                       ;THE SECOND ENTRY ADDRESS IS ON THE
                                                                       ;STACK
054424                                 5$:
054424  016637  000002  001736                 MOV     2(SP),@#REPPSW  ;SAVE ERROR PSW
054432  011637  001536                         MOV     (SP),@#VADR     ;SAVE PC
054436  162737  000002  001536                 SUB     #2,@#VADR       ;ADJUST ERROR PC
054444  013737  177744  001740         PERET:  MOV     @#MEMERR,@#REPREG       ;SAVE ERROR REG FOR TYPEOUT
054452  013737  001212  001742                 MOV     @#$LPERR,@#REPADR       ;SAVE LOOP ADDRESS
054460  012737  054470  001212                 MOV     #2$,@#$LPERR            ;SET RETURN ADDRESS IF LOOPING
054466  104004                                 ERROR   4
054470  013737  001742  001212         2$:     MOV     @#REPADR,@#$LPERR       ;RESTORE LOOP ADDRESS
054476  012767  177777  177712                 MOV     #-1,PARFLG      ;RESTORE NEGATIVE ONE FOR A FLAG
054504  012716  054512                 1$:     MOV     #X,(SP)         ;PUT ADDRESS ON STACK TO GET ORIGINAL
                                                                       ;PSW BACK
054510  000002                         RT1:    RTI                     ;GET OLD PSW
054512  000177  124474                 X:      JMP     @$LPERR         ;JUMP TO START OF TEST THAT HAD THE PE
                                       ;;*****************************************************************
                                       .SBTTL  CONTEXT SWITCH DOWN SUBROUTINE
                                       ;*      SUBROUTINE TO SAVE & LOAD KIPAR'S 0,1,AND 2 (IF MEM MGMT ENABLED)
                                       ;*      THIS ROUTINE IS CALLED BY THE KEYBOARD INTERRUPT, LINE CLOCK
                                       ;*      INTERRUPT, UBE SERVICE ROUTINE, MBT SERVICE ROUTINE, AND TYPE TIME ROUTINE.
                                       ;;*****************************************************************
054516  105737  001545                 LDKT:   TSTB    @#MMON                  ;BRANCH IF MEM MGMT DISABLED
054522  001427                                 BEQ     1$
054524  012604                                 MOV     (SP)+,R4        ;SAVE RETURN PC
054526  013737  177776  054660                 MOV     @#PSW,@#$SAVPSW ;SAVE THE CURRENT PSW
054534  042737  140000  177776                 BIC     #140000,@#PSW   ;GO TO KERNEL MODE
054542  012700  172340                         MOV     #KIPAR0,R0      ;GET ADDRESS OF PAR0
054546  012001                                 MOV     (R0)+,R1        ;GET PAR0
054550  012002                                 MOV     (R0)+,R2        ;GET PAR1
```

```
054552  012003                         MOV     (R0)+,R3            ;GET PAR2
054554  012740  000400                 MOV     #400,-(R0)          ;RELOC BACK TO LOW CORE
054560  012740  000200                 MOV     #200,-(R0)
054564  005040                         CLR     -(R0)
054566  012700  054652                 MOV     #$SAVPAR,R0         ;GET ADDRESS OF SAVE BUFFER
054572  010120                         MOV     R1,(R0)+            ;PUT PAR DATA IN MEMORY
054574  010220                         MOV     R2,(R0)+
054576  010320                         MOV     R3,(R0)+
054600  010446                         MOV     R4,-(SP)            ;PUT RETURN PC ON STACK
054602  000207                 1$:     RTS     PC

                               ;;*********************************************************
                               .SBTTL  CONTEXT SWITCH UP SUBROUTINE
                               ;SUBROUTINE TO RESTORE KIPAR0, 1, AND 2 (IF MGMT ENABLED)
                               ;;*********************************************************
054604  105737  001545         RESKT:  TSTB    @#MMON              ;BRANCH IF MEM MGMT DISABLED
054610  001417                         BEQ     1$
054612  012604                         MOV     (SP)+,R4            ;GET RETURN PC
054614  012700  054652                 MOV     #$SAVPAR,R0         ;GET ADDRESS OF SAVE BUFF
054620  012001                         MOV     (R0)+,R1            ;GET OLD PAR DATA
054622  012002                         MOV     (R0)+,R2
054624  012003                         MOV     (R0)+,R3
054626  012700  172340                 MOV     #KIPAR0,R0          ;GET ADDRESS OF PAR0
054632  010120                         MOV     R1,(R0)+            ;RELOCATE BACK
054634  010220                         MOV     R2,(R0)+
054636  010310                         MOV     R3,(R0)
054640  013737  054660  177776         MOV     @#$SAVPSW,@#PSW
054646  010446                         MOV     R4,-(SP)
054650  000207                 1$:     RTS     PC
054652  000003                 $SAVPAR:.BLKW   3
054660  000000                 $SAVPSW:.WORD
                               ;;*********************************************************
                               .SBTTL  KT ABORT SUBROUTINE
                               ;;*********************************************************
054662  005227                 KTABRT: INC     (PC)+               ;MAKE FLAG ZERO FIRST TIME THRU
054664  177777                 KTFLG:  .WORD   -1                  ;NEGATIVE ONE FOR A FLAG
054666  001401                         BEQ     5$                  ;BRANCH IF FIRST TIME IN
054670  000000                         HALT                        ;I HAVE ENTERED THIS ROUTINE BEFORE
                                                                   ;I FINISHED REPORTING THE FIRST ERROR
                                                                   ;THE SECOND ENTRY ADDRESS IS ON THE
                                                                   ;STACK
054672                         5$:
054672  016637  000002  001736         MOV     2(SP),@#REPPSW          ;SAVE ERROR PSW
054700  011637  001536                 MOV     (SP),@#VADR             ;SAVE ERROR PC
054704  162737  000002  001536         SUB     #2,@#VADR
054712  013737  177572  001744         MOV     @#MMR0,@#REPMR0         ;SAVE MMR0
054720  013737  177576  001746         MOV     @#MMR2,@#REPMR2         ;SAVE MMR2
054726  013737  001212  001742         MOV     @#$LPERR,@#REPADR       ;SAVE LOOP ADDRESS
054734  012737  054744  001212         MOV     #1$,@#$LPERR            ;SET RETURN ADR IF LOOPING
054742  104003                         ERROR   3
054744  013737  001742  001212 1$:     MOV     @#REPADR,@#$LPERR       ;RESTORE LOOP ADR
054752  042737  170000  177572         BIC     #170000,@#MMR0          ;CLEAR ERRORS
054760  012767  177777  177676         MOV     #-1,KTFLG               ;RESTORE NEGATIVE ONE FOR A FLAG
054766  013716  001212                 MOV     @#$LPERR,(SP)           ;GET LOOP ADDRESS
054772  000002                         RTI                             ;RETURN
```

```
                                          ;;**************************************************************
                                          .SBTTL  RESERVED INSTRUCTION ROUTINE
                                          ;;**************************************************************
054774                                    RESERR:
054774  005227                                    INC     (PC)+                   ;MAKE FLAG ZERO FIRST TIME THRU
054776  177777                            RESFLG: .WORD   -1                      ;NEGATIVE ONE FOR A FLAG
055000  001401                                    BEQ     5$                      ;BRANCH IF FIRST TIME IN
055002  000000                                    HALT                            ;I HAVE ENTERED THIS ROUTINE BEFORE
                                                                                  ;I FINISHED REPORTING THE FIRST ERROR
                                                                                  ;THE SECOND ENTRY ADDRESS IS ON THE
                                                                                  ;STACK
055004                                    5$:
055004  016637  000002  001736                    MOV     2(SP),@#REPPSW                  ;SAVE PSW
055012  011637  001536                            MOV     (SP),@#VADR                     ;SAVE ERROR PC
055016  162737  000002  001536                    SUB     #2,@#VADR
055024  013737  001212  001742                    MOV     @#$LPERR,@#REPADR               ;SAVE LOOP ADR
055032  012737  055042  001212                    MOV     #1$,@#$LPERR                    ;SET RETURN ADR IF LOOPING
055040  104002                                    ERROR   2
055042  013737  001742  001212            1$:     MOV     @#REPADR,@#$LPERR               ;RESTORE LOOP ADR
055050  012767  177777  177720                    MOV     #-1,RESFLG                      ;RESTORE NEGATIVE ONE FOR A FLAG
055056  013716  001212                            MOV     @#$LPERR,(SP)                   ;GET LOOP ADDRESS
055062  000002                                    RTI                                     ;RETURN

                                          ;;**************************************************************
                                          .SBTTL  TRAP TO 4 SERVICE ROUTINE
                                          ;;**************************************************************
055064  005227                            ERPRT:  INC     (PC)+                   ;MAKE FLAG ZERO FIRST TIME THRU
055066  177777                            ERFLG:  .WORD   -1                      ;NEGATIVE ONE FOR A FLAG
055070  001401                                    BEQ     5$                      ;BRANCH IF FIRST TIME IN
055072  000000                                    HALT                            ;I HAVE ENTERED THIS ROUTINE BEFORE
                                                                                  ;I FINISHED REPORTING THE FIRST ERROR
                                                                                  ;THE SECOND ENTRY ADDRESS IS ON THE
                                                                                  ;STACK
055074                                    5$:
055074  016637  000002  001736                    MOV     2(SP),@#REPPSW                  ;SAVE ERROR PSW
055102  011637  001536                            MOV     (SP),@#VADR                     ;SAVE ERROR PC
055106  162737  000002  001536                    SUB     #2,@#VADR
055114  012706  000700                            MOV     #USESTK,SP                      ;RESTORE SP
055120  013737  177766  001740                    MOV     @#CPUERR,@#REPREG               ;GET ERROR REG
055126  013737  001212  001740                    MOV     @#$LPERR,@#REPREG               ;SAVE LOOP ADR
055134  012737  055144  001212                    MOV     #1$,@#$LPERR                    ;SET RETURN ADR IF LOOPING
055142  104001                                    ERROR   1
055144  013737  001740  001212            1$:     MOV     @#REPREG,@#$LPERR               ;SET LOOP ADR
055152  012767  177777  177706                    MOV     #-1,ERFLG               ;RESTORE NEGATIVE ONE FOR A FLAG
055160  013746  001736                            MOV     @#REPPSW,-(SP)  ;SETUP STACK TO RETURN
055164  013746  001212                            MOV     @#$LPERR,-(SP)
055170  000002                                    RTI                                     ;RETURN

                                          .SBTTL  MICROBREAK TRAP SERVICE ROUTINE
                                          ;;**************************************************************
                                          ;*      THIS ROUTINE MERELY SETS A FLAG
                                          ;*      WHEN THE ROUTINE HAS BEEN ENTERED
                                          ;*
055172  005267  000002                    BKROUT: INC     BKFLAG                  ;SET MICROBREAK FLAG TO
                                                                                  ;INDICATE TRAP TO 4 OCCURRED
055176  000002                                    RTI                             ;RETURN FROM TRAP
```

H02


```
 11002  055200  000000                  BKFLAG: .WORD   0               ;MICROBREAK TRAP FLAG
 11003
```

```
                                       ;THE BELOW TABLE REPRESENTS THE 'NEW' PSW SET BY THE PROGRAM ON
                                       ;SUCCESSIVE SUB-PASSES.
                                       ;NOTE THE BELOW TABLE MAY BE MODIFIED TO CAUSE THE PROGRAM TO RUN
                                       ;UNDER USER DEFINED PARAMETERS BY PATCHING IN THE DESIRED PASS PARAMETER
                                       ;FOR EXAMPLE TO CAUSE THE PROGRAM TO RUN WITHOUT SETTING THE 'T' BIT
                                       ;IN ALL PASSES PATCH OUT THE 'T' BIT IN THE TABLE.
055202  000000                         PSWTAB: 000000                   ;
055204  000020                                 000020                   ;T-BIT TRAPPING
055206  140000                                 140000                   ;USER MODE
055210  140020                                 140020                   ;USER MODE, T-BIT TRAPPING
                                       ;MESSAGES
                                               .EVEN
055212  002154                         REGINX: RP3DS
055214  002176                                 RKDS
055216  000000                                 .WORD
055220  000000                                 .WORD
055222  002216                                 RP4CS1
055224  002256                                 RSCS1
055226  000000                                 .WORD
055230  000000                                 .WORD
055232  002322                                 MBTTBL
055234  002304                                 UBETBL
055236  055503                         MSGINX: .WORD   MSG5
055240  055511                                 .WORD   MSG6
055242  056320                                 .WORD   MSG21
055244  056320                                 .WORD   MSG21
055246  055517                                 .WORD   MSG10
055250  055525                                 .WORD   MSG11
055252  056320                                 .WORD   MSG21
055254  056320                                 .WORD   MSG21
055256  056014                                 .WORD   MSG15
055260  056357                                 .WORD   MSG24
055262  046200  053517  046040         MSG1:   .ASCIZ  <CRLF>'LOW LIM?'
055270  046511  000077
055274  044510  044107  046040         MSG2:   .ASCIZ  'HIGH LIM?'
055302  046511  000077
055306  051105  047522  050122         MSG3:   .ASCIZ  /ERRORPC PHYS-PC  PSW    TEST NO SUBPAS-PASS CNT/
055314  020103  044120  051531
055322  050055  020103  050040
055330  053523  020040  020040
055336  042524  052123  047040
055344  020117  052523  050102
055352  051501  050055  051501
055360  020123  047103  000124
055366  044124  020105  047506         MSG4:   .ASCII  /THE FOLLOWING DEVICES AND DRIVES WILL BE USED FOR RELOCATION:/<CRLF>
055374  046114  053517  047111
055402  020107  042504  044526
055410  042503  020123  047101
055416  020104  051104  053111
055424  051505  053440  046111
055432  020114  042502  052440
055440  042523  020104  047506
055446  020122  042522  047514
055454  040503  044524  047117
055462  100072
055464  042504  044526  042503                 .ASCIZ  /DEVICE DRIVES/<CRLF>
```

```
055472  042011  044522  042526
055500  100123     000
055503     122  030120  004463  MSG5:   .ASCIZ  ?RP03   ?
055510     000
055511     122  030113  004465  MSG6:   .ASCIZ  ?RK05   ?
055516     000
055517     122  030120  004464  MSG10:  .ASCIZ  ?RP04   ?
055524     000
055525     122  030123  004464  MSG11:  .ASCIZ  ?RS04   ?
055532     000
055533     104  053122  052123  MSG12:  .ASCIZ  /DRVSTA  ERRREG  CSREG   WRDCNT  BUSADR  DSKADR  CYLADR(RP03)  PHYS BUSA
055540  020101  042440  051122
055546  042522  020107  041440
055554  051123  043505  020040
055562  053440  042122  047103
055570  020124  041040  051525
055576  042101  020122  042040
055604  045523  042101  020122
055612  041440  046131  042101
055620  024122  050122  031460
055626  020051  050040  054510
055634  020123  052502  040523
055642  051104  000200
055646  041440  030523  020040  MSG13:  .ASCIZ  / CS1    WRDCNT  BUSADR  BADREX  DSKADR   CS2     CS3    DRVSTA  ERRREG/
055654  020040  051127  041504
055662  052116  020040  052502
055670  040523  051104  020040
055676  040502  051104  054105
055704  020040  051504  040513
055712  051104  020040  041440
055720  031123  020040  020040
055726  041440  031523  020040
055734  020040  051104  051526
055742  040524  020040  051105
055750  051122  043505  000200
055756  042504  041523  046131  MSG14:  .ASCIZ  /DESCYL   ER2     ER3    RPCC/<CRLF>
055764  020040  042440  031122
055772  020040  020040  042440
056000  031522  020040  020040
056006  050122  041503  000200
056014  040515  051523  041040  MSG15:  .ASCIZ  /MASS BUS TESTER /
056022  051525  052040  051505
056030  042524  020122     000
056035     040  051503  020061  MSG16:  .ASCIZ  / CS1    WRDCNT  BUSADR  BADREX   MR2     CS2      ST      ER      CS3/<
056042  020040  053440  042122
056050  047103  020124  041040
056056  051525  042101  020122
056064  041040  042101  042522
056072  020130  020040  046440
056100  031122  020040  020040
056106  041440  031123  020040
056114  020040  020040  052123
056122  020040  020040  042440
056130  020122  020040  020040
056136  041440  031523  000200
056144  020040  041503  020040  MSG17:  .ASCIZ  /  CC    BUSADR    CR2     CR1  PHYS BUSADR/<CRLF>
```

```
056152  020040  052502  040523
056160  051104  020040  020040
056166  051103  020062  020040
056174  020040  051103  020061
056202  050040  054510  020123
056210  052502  040523  051104
056216  000200
056220  044124  020105  052521  MSG20:  .ASCIZ  /THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789/<15><12>
056226  041511  020113  051102
056234  053517  020116  047506
056242  020130  052512  050115
056250  042105  047440  042526
056256  020122  044124  020105
056264  040514  054532  042040
056272  043517  020123  040502
056300  045503  030040  031061
056306  032063  033065  034067
056314  006471  000012
056320  046111  042514  040507  MSG21:  .ASCIZ  /ILLEGAL DEVICE/<CRLF>
056326  020114  042504  044526
056334  042503  000200
056340  020040  020040  020040  MSG22:  .ASCII  /       /
056346  020040
056350  020040  020040  020040  MSG23:  .ASCIZ  /      /
056356     000
056357     125  044516  052502  MSG24:  .ASCIZ  /UNIBUS EXERCISER /
056364  020123  054105  051105
056372  044503  042523  020122
056400     000
056401     117  052120  041456  MSG25:  .ASCIZ  /OPT.CP=/
056406  036520     000
056411     125  042516  050130  EM1:    .ASCIZ  /UNEXPECTED TRAP TO 4/
056416  041505  042524  020104
056424  051124  050101  052040
056432  020117  000064
056436  044526  052122  041520  DH1:    .ASCIZ  /VIRTPC  PHYSPC    PSW   CPUERR/
056444  020040  044120  051531
056452  041520  020040  020040
056460  051520  020127  020040
056466  050103  042525  051122
056474     000
056475     000     001     000  DF1:    .BYTE   0,1,0,0,0
056500     000     000
                                        .EVEN
056502  001536  001536  001736  DT1:    .WORD   VADR,VADR,REPPSW,REPREG,0
056510  001740  000000
056514  047125  054105  042520  EM2:    .ASCIZ  /UNEXPECTED TRAP TO 10/
056522  052103  042105  052040
056530  040522  020120  047524
056536  030440  000060
056542  044526  052122  041520  DH2:    .ASCIZ  /VIRTPC  PHYSPC    PSW/
056550  020040  044120  051531
056556  041520  020040  020040
056564  051520  000127
                                        .EVEN
056570  001536  001536  001736  DT2:    .WORD   VADR,VADR,REPPSW,0
```

```
056576  000000
056600  047125  054105  042520  EM3:    .ASCIZ  /UNEXPECTED TRAP TO 250(MGMT)/
056606  052103  042105  052040
056614  040522  020120  047524
056622  031040  030065  046450
056630  046507  024524     000
056635     126  051111  050124  DH3:    .ASCIZ  /VIRTPC  PHYSPC    PSW    MMR0    MMR2/
056642  020103  050040  054510
056650  050123  020103  020040
056656  050040  053523  020040
056664  020040  046515  030122
056672  020040  020040  046515
056700  031122     000
        056704                          .EVEN
056704  001536  001536  001736  DT3:    .WORD   VADR,VADR,REPPSW,REPMR0,REPMR2,0
056712  001744  001746  000000
056720  047125  054105  042520  EM4:    .ASCIZ  /UNEXPECTED TRAP TO 114/
056726  052103  042105  052040
056734  040522  020120  047524
056742  030440  032061     000
056747     126  051111  050124  DH4:    .ASCIZ  /VIRTPC  PHYSC PC  PSW   MEM ERR REG/
056754  020103  050040  054510
056762  041523  050040  020103
056770  050040  053523  020040
056776  046440  046505  042440
057004  051122  051040  043505
057012     000
        057014                          .EVEN
057014  001536  001536  001736  DT4:    .WORD   VADR,VADR,REPPSW,REPREG,0
057022  001740  000000
057026     000     001     000  DF4:    .BYTE   0,1,0,0
057031     000
057032  040520  044522  054524  EM5:    .ASCIZ  /PARITY ERROR DURING DATA CHECK/
057040  042440  051122  051117
057046  042040  051125  047111
057054  020107  040504  040524
057062  041440  042510  045503
057070     000
057071     123  041522  042101  DH5:    .ASCIZ  /SRCADR  DSTADR  MEM ERR REG/
057076  020122  042040  052123
057104  042101  020122  046440
057112  046505  042440  051122
057120  051040  043505     000
057125     000     001     000  DF5:    .BYTE   0,1,0
                                        .EVEN
057130  001750  001536  001740  DT5:    .WORD   SRCADR,VADR,REPREG,0
057136  000000
057140  051105  047522  020122  EM6:    .ASCIZ  /ERROR DURING DATA CHECK-RELOC WAS BY CP/
057146  052504  044522  043516
057154  042040  052101  020101
057162  044103  041505  026513
057170  042522  047514  020103
057176  040527  020123  054502
057204  041440  000120
057210  051123  040503  051104  DH6:    .ASCIZ  /SRCADR  DSTADR/
057216  020040  051504  040524
```

```
057224  051104     000
        057230                        .EVEN
057230  001304  001536  000000  DT6:  .WORD   $TMP0,VADR,0
057236  051105  047522  020122  EM10: .ASCIZ  'ERROR DURING DATA CHECK-RELOC WAS BY I/O'
057244  052504  044522  043516
057252  042040  052101  020101
057260  044103  041505  026513
057266  042522  047514  020103
057274  040527  020123  054502
057302  044440  047457     000
057307     123  041522  042101  DH10: .ASCIZ  /SRCADR   DSTADR   DEVICE THAT DID XFER/
057314  020122  020040  051504
057322  040524  051104  020040
057330  042040  053105  041511
057336  020105  044124  052101
057344  042040  042111  054040
057352  042506  000122
057356     000     001     003  DF10: .BYTE   0,1,3,0
057361     000
                                      .EVEN
057362  001304  001536  001310  DT10: .WORD   $TMP0,VADR,$TMP2,$TMP3,0
057370  001312  000000
057374  020040  020040  020040  DH11: .ASCIZ  /              DATA1                        DATA2/
057402  020040  020040  020040
057410  020040  042040  052101
057416  030501  020040  020040
057424  020040  020040  020040
057432  020040  020040  020040
057440  020040  020040  020040
057446  020040  020040  020040
057454  020040  042040  052101
057462  031101     000
057465     005     000     005  DF11: .BYTE   5,0,5,0
057470     000
        057472                        .EVEN
057472  001464  001450  001474  DT11: .WORD   FLTMP0,FREG2,FLTMP1,FREG3,0
057500  001452  000000
057504  041125  020105  047516  EM12: .ASCIZ  /UBE NON-EXISTANT MEMORY ERROR/
057512  026516  054105  051511
057520  040524  052116  046440
057526  046505  051117  020131
057534  051105  047522  000122
057542  044120  051531  041040  DH12: .ASCIZ  /PHYS BUSADR/
057550  051525  042101  000122
057556     002                  DF12: .BYTE   2
        057560                        .EVEN
057560  001226  000000          DT12: .WORD   $GDDAT,0
057564  041115  020124  047516  EM13: .ASCIZ  /MBT NON-EXISTANT MEMORY ERROR/
057572  026516  054105  051511
057600  040524  052116  046440
057606  046505  051117  020131
057614  051105  047522  000122
057622  044120  051531  040440  DH13: .ASCIZ  /PHYS ADDRESS/
057630  042104  042522  051523
057636     000
057637     106  047514  052101  EM14: .ASCIZ  /FLOATING POINT ERROR/
```

```
057644  047111  020107  047520
057652  047111  020124  051105
057660  047522  000122
057664  042011  052101  030501  DH14:   .ASCIZ  /       DATA1                           DATA2/
057672  004411  004411  040504
057700  040524  000062
                                        .EVEN
057704  001434  001450  001440  DT14:   .WORD   FTMP4,FREG2,FTMP6,FREG3,0
057712  001452  000000
057716     004     000     004  DF14:   .BYTE   4,0,4,0
057721     000
057722  042504  044526  042503  EM15:   .ASCIZ  /DEVICE HUNG/
057730  044040  047125  000107
                                        .EVEN

057736  000000                  ENDTAG: .WORD   0
                                ;;***********************************************************
                                ;THE FOLLOWING ASCII GETS OVERLAYED WHEN THE PROGRAM RUNS.
057740  050117  051105  052101  SWITCH: .ASCII  /OPERATIONAL SWITCH SETTINGS/<CRLF>
057746  047511  040516  020114
057754  053523  052111  044103
057762  051440  052105  044524
057770  043516  100123
057774  053523  052111  044103          .ASCII  /SWITCH                   USE/<CRLF>
060002  004411  052411  042523
060010     200
060011     040  030440  020065          .ASCII  /  15  = 100000           HALT ON ERROR/<CRLF>
060016  036440  030440  030060
060024  030060  004460  044011
060032  046101  020124  047117
060040  042440  051122  051117
060046     200
060047     040  030440  020064          .ASCII  /  14  = 040000           LOOP ON TEST/<CRLF>
060054  036440  030040  030064
060062  030060  004460  046011
060070  047517  020120  047117
060076  052040  051505  100124
060104  020040  031461  020040          .ASCII  /  13  = 020000           INHIBIT ERROR TYPEOUTS/<CRLF>
060112  020075  031060  030060
060120  030060  004411  047111
060126  044510  044502  020124
060134  051105  047522  020122
060142  054524  042520  052517
060150  051524     200
060153     040  030440  020062          .ASCII  /  12  = 010000           INHIBIT UBE/<CRLF>
060160  036440  030040  030061
060166  030060  004460  044411
060174  044116  041111  052111
060202  052440  042502     200
060207     040  030440  020061          .ASCII  /  11  = 004000           INHIBIT ITTERATIONS/<CRLF>
060214  036440  030040  032060
060222  030060  004460  044411
060230  044116  041111  052111
060236  044440  052124  051105
060244  052101  047511  051516
060252     200
```

```
060253     040 030440 020060          .ASCII  /  10  = 002000        BELL ON ERROR/<CRLF>
060260  036440 030040 031060
060266  030060 004460 041011
060274  046105 020114 047117
060302  042440 051122 051117
060310     200
060311     040 020040 020071          .ASCII  /   9  = 001000        LOOP ON ERROR/<CRLF>
060316  036440 030040 030460
060324  030060 004460 046011
060332  047517 020120 047117
060340  042440 051122 051117
060346     200
060347     040 020040 020070          .ASCII  ?   8  = 000400        INHIBIT RELOCATION VIA I/O DEVICE?<CRLF>
060354  036440 030040 030060
060362  030064 004460 044411
060370  044116 041111 052111
060376  051040 046105 041517
060404  052101 047511 020116
060412  044526 020101 027511
060420  020117 042504 044526
060426  042503    200
060431     040 020040 020067          .ASCII  /   7  = 000200        INHIBIT TYPEOUT OF THIS TEXT AND SYS SIZE/<CRLF>
060436  036440 030040 030060
060444  030062 004460 044411
060452  044116 041111 052111
060460  052040 050131 047505
060466  052125 047440 020106
060474  044124 051511 052040
060502  054105 020124 047101
060510  020104 054523 020123
060516  044523 042532    200
060523     040 020040 020066          .ASCII  /   6  = 000100        INHIBIT RELOCATION/<CRLF>
060530  036440 030040 030060
060536  030061 004460 044411
060544  044116 041111 052111
060552  051040 046105 041517
060560  052101 047511 100116
060566  020040 032440 020040          .ASCII  /   5  = 000040        INHIBIT ROUND ROBIN RELOCATION/<CRLF>
060574  020075 030060 030060
060602  030064 004411 047111
060610  044510 044502 020124
060616  047522 047125 020104
060624  047522 044502 020116
060632  042522 047514 040503
060640  044524 047117    200
060645     040 020040 020064          .ASCII  /   4  = 000020        INHIBIT RANDOM DISK ADDRESS/<CRLF>
060652  036440 030040 030060
060660  031060 004460 044411
060666  044116 041111 052111
060674  051040 047101 047504
060702  020115 044504 045523
060710  040440 042104 042522
060716  051523    200
060721     040 020040 020063          .ASCII  /   3  = 000010        INHIBIT MBT/<CRLF>
060726  036440 030040 030060
060734  030460 004460 044411
```

```
060742  044116  041111  052111
060750  046440  052102     200
060755     040  020040  004462        .ASCII  /   2        THESE THREE SWITCHES/<CRLF>
060762  052011  042510  042523
060770  052040  051110  042505
060776  051440  044527  041524
061004  042510  100123
061010  020040  030440  004411        .ASCII  /   1        ARE ENCODED TO SELECT RELOCATION/<CRLF>
061016  051101  020105  047105
061024  047503  042504  020104
061032  047524  051440  046105
061040  041505  020124  042522
061046  047514  040503  044524
061054  047117     200
061057     040  020040  004460        .ASCII  /   0        ON THE FOLLOWING DEVICES:/<CRLF>
061064  047411  020116  044124
061072  020105  047506  046114
061100  053517  047111  020107
061106  042504  044526  042503
061114  035123     200
061117     011  027060  027056        .ASCII  ?       0...RP11/RP03?<CRLF>
061124  050122  030461  051057
061132  030120  100063
061136  030411  027056  051056        .ASCII  ?       1...RK11/RK05?<CRLF>
061144  030513  027461  045522
061152  032460     200
061155     011  027062  027056        .ASCII  ?       2...NOT USED?<CRLF>
061162  047516  020124  051525
061170  042105     200
061173     011  027063  027056        .ASCII  ?       3...NOT USED?<CRLF>
061200  047516  020124  051525
061206  042105     200
061211     011  027064  027056        .ASCII  ?       4...RH11/RP04?<CRLF>
061216  044122  030461  051057
061224  030120  100064
061230  032411  027056  051056        .ASCII  ?       5...RH11/RS04 OR RS03?<CRLF>
061236  030510  027461  051522
061244  032060  047440  020122
061252  051522  031460     200
061257     011  027066  027056        .ASCII  ?       6...NOT USED?<CRLF>
061264  047516  020124  051525
061272  042105     200
061275     011  027067  027056        .ASCIZ  ?       7...NOT USED?<CRLF>
061302  047516  020124  051525
061310  042105  000200
        000001                        .END
```

```
AA       001547         1565#   2123#
ADCB2    011336         3311    3313#
ADCB5    012162         3543    3545#
ADCB6    012664         3701    3702    3704#
ADCB7    013614         3927    3928    3929    3931#
ADC0     007230         2705    2706    2707    2709#
ADC1     010130         2936    2937    2938    2940#
ADC2     011140         3240    3242#
ADC5     011762         3470    3471    3473#
ADC6     012466         3643    3644    3646#
ADC7     013462         3884    3885    3887#
ADD0     014310         4095    4096    4097    4099#
ADD1     014414         4133    4134    4136#
ADD1A    014640         4216    4217    4218    4220#
ADD1B    014656         4225    4226    4228#
ADD2     015300         4372    4373    4375#
ADD3     016056         4566    4568#
ADD6     016434         4669    4670    4672#
ADD7     017114         4788    4789    4790    4792#
ARBEX    035606         8040    8084#
ARBFIN   035574         8025    8047    8069    8071    8082#
ASHCL0   025332         6211#
ASHCR0   025410         6232#
ASHL0    025122         6161#
ASHL1    025720         6345#
ASHR0    025236         6188#
ASHR1    026006         6368#
ASLB1    010500         3071    3072    3074#
ASLB1A   010724         3158    3159    3161#
ASLB3    012152         3537    3538    3540#
ASLB4    011442         3348    3349    3350    3352#
ASLB6    012646         3693    3694    3695    3697#
ASLB7    013712         3959    3960    3962#
ASL0     007352         2749    2750    2751    2752    2754#
ASL1     010304         2999    3000    3001    3003#
ASL3     011676         3439    3440    3442#
ASL4     011232         3272    3273    3274    3276#
ASL6     012436         3631    3632    3634#
ASL7     013310         3831    3832    3834#
ASP1     002502         1850#
ASP2     002574         1880#
ASRB1    010574         3107    3109#
ASRB1A   010610         3113    3114    3116#
ASRB2    011406         3333    3334    3336#
ASRB2A   011424         3341    3342    3344#
ASRB5    012112         3518    3519    3521#
ASRB6    012764         3733    3734    3736#
ASRB7    013730         3966    3967    3969#
ASR0     007400         2763    2764    2765    2767#
ASR1     010172         2956    2957    2958    2960#
ASR2     011154         3246    3247    3249#
ASR3     011662         3433    3435#
ASR6     012320         3593    3594    3596#
ASR7     013344         3845    3846    3848#
BICB1    015040         4289    4290    4292#
BICB1A   015062         4300    4303#   4362    4363
```

```
BIC0      014222          4066    4067    4068    4070#
BIC1      014536          4179    4180    4182#
BIC2      015370          4401    4402    4403    4405#
BIC3      016070          4571    4573#
BIC7      017766          4971#   4973
BINB      015614          4466    4468    4471    4473    4476    4478    4481    4484#
BINB7     017534          4906    4914#
BIN1      015220          4332    4335    4338    4341    4344    4347    4350    4355#
BISB1     015026          4284    4286#
BIS0      014200          4057    4058    4060#
BIS0A     014256          4084    4086#
BIS1      014524          4173    4174    4176#
BIS2      015326          4385    4387#
BIS2A     015430          4419    4420    4422#
BIS7      017726          4961#   4963
BITB1     015016          4278    4279    4281#
BITB2     015700          4506    4507    4509#
BITB3     016246          4626    4629#
BITB6     016646          4719#
BITT1     014452          4150    4151    4153#
BITT2     015414          4412    4413    4415#
BIT0    = 000001          1265#   1437    2425    2433    2435    6601    6687    6736    6956    6998    7084    7127    7164
                          7189    8973    9011    9046
BIT00   = 000001          1255#   1265
BIT01   = 000002          1254#   1264
BIT02   = 000004          1253#   1263
BIT03   = 000010          1252#   1262
BIT04   = 000020          1251#   1261
BIT05   = 000040          1250#   1260
BIT06   = 000100          1249#   1259
BIT07   = 000200          1248#   1258
BIT08   = 000400          1247#   1257
BIT09   = 001000          1246#   1256    9559    9621
BIT1    = 000002          1264#
BIT10   = 002000          1245#   9606
BIT11   = 004000          1244#   9457    9566
BIT12   = 010000          1243#   2389    2409    8874
BIT13   = 020000          1242#   6889    8853    9613
BIT14   = 040000          1241#   9128    9189    9212    9227    9238    9262    9293    9310    9325    9336    9361    9450
                          9541    10530
BIT15   = 100000          1240#   6601    6607    6641    6687    6704    6736    6598    7036    7084    7127    7155    7164
                          7189    8501    8983    8990    9091    9098    9207    9214    9305    9312    10481
BIT2    = 000004          1263#   6704    6918    7155
BIT3    = 000010          1262#   9018
BIT4    = 000020          1261#   2464    6881    7143    7241    8782    8786
BIT5    = 000040          1260#   1441    2397    2417    6963    7036    8510    8513    8858    9193    9232    9267    9297
                          9330    9366
BIT6    = 000100          1259#   1436    1439    6696    6908    7036    7174    8809    9525
BIT7    = 000200          1258#   1438    1440    2335    6889    7036    7064
BIT8    = 000400          1257#   2341    2378    2395    2415    2446    8519    8593    8614    8678    8750    8820    8894
                          8967    9002    9039    9074    9111    9147    9187    9198    9225    9260    9291    9323    9359
BIT9    = 001000          1256#   6602    6659    6889    8526
BKFLAG    055200          6598*   6611    6638*   6647    10999*  11002#
BKROUT    055172          6597    10999#
BPTVEC= 000014            1272#   5424    5427*   5428*   5500*   5501*
BSP1      002576          1883#
```

```
BSP2     002664        1911#
CACHVE=  000114        1340#   2476*   2477*   10720*  10729*
CALLHA=  000010        1431#   8417    8521
CBIT     023174        5689#
CCNSC0   003070        2013#
CCNSC1   003074        2014#
CCNSC2   003100        2015#
CCNST0   003114        2018#
CCNST1   003120        2019#
CCNST2   003110        2017#
CCR    = 177746        1342#   6774*   6776*   6824*   6838*   6840*   6949*   6950    6987*   6992*   6995*   7001*   7016*
CCST20   003104        2016#
CC0      007036        2626    2627    2628    2629    2630    2631    2632    2634#
CC1      007052        2638    2639    2640    2642#
CC2      007066        2646    2647    2648    2650#
CC3      007100        2654    2655    2657#
CC4      007114        2661    2662    2663    2665#
CHKDAT   054026        8283    8462    8541    10720#  10728
CHKSP    022766        5622#
CLCUA    003050        2009#
CLFGIN   003054        2010#
CLJAM    003034        2006#
CLPBA    003044        2008#
CLRTBI   054112        5418    5568    5615    6448    8198    8591    10743#
CLRO     007146        2676    2677    2678    2679    2681#
CLSERV   003040        2007#
CLTAG    003064        2012#
CLWHAM   003060        2011#
CMPB1    014772        4268    4270#
CMPB2    015664        4499    4500    4502#
CMPB3    016260        4632    4635#
CMPN     023230        5703#
CMP0     014102        4020    4021    4022    4024#
CMP0A    014344        4110    4113#
CMP1     014434        4141    4142    4143    4145#
CMP1A    014554        4186    4187    4188    4190#
CMP2     015316        4379    4380    4382#
CMP7     017050        4774    4775    4777#
CNSCOR   002554        1872#
CNSCTL   002740        1954#
CNSC0    002704        1922#
CNSC1    002706        1923#
CNSC2    002710        1924#
CNSSW    002552        1871#
CNSTSW   002544        1868#
CNST0    002714        1926#
CNST1    002732        1945#
CNST2    002730        1944#
CNTRLC=  000003       10773#  10778
CNTRLO=  000017       10774#  10793
CNVADR   053724        6709    8287    9677    9753    10684#
COMB1    010556        3098    3099    3101#
COMB1A   010736        3164    3165    3167#
COMB2    011320        3302    3303    3305#
COMB5    012070        3510    3512#
COMB6    012716        3714    3715    3717#
```

```
COMB7    013630           3934    3935    3937#
COM0     007212           2696    2697    2698    2699    2701#
COM1     010316           3006    3007    3009#
COM3     011740           3460    3462#
COM4     011042           3203    3205#
COM6     012304           3588    3589    3591#
COM7     013434           3873    3874    3876#
COUNT    002753           1965#
CPCHK    004132           2229#
CPUERR=  177766           1345#   6696    6881    6908    6956    7143    7174    10984
CR     = 000015           1180#   9962    9972
CRLF   = 000200           1181#   1338#   2318    2452    8362    8371    9808    9931    9972    11036   11048   11059   11070
                         11083   11095   11103   11115   11134   11302   11307   11310   11316   11321   11328   11333   11340
                         11346   11352   11361   11371   11377   11385   11393   11398   11403   11410   11416   11419   11422
                         11425   11428   11431   11435   11438
CSP1     002666           1914#
CSP2     002714           1926#
CST200   002712           1925#
CUA      002745           1959#
DBINB7   017530           4911#   4943*   4944*   4948*   4949*   4950    4954*   4961    4962*   4969*   4971    4972*
DBIN7    016750           4749#   4764*   4768*   4773    4779*   4787*   4794*   4798*   4802
DCS0     002756           1968#
DCS1     002760           1970#
DDATA    016276           4644#   4648*   4650*   4654    4658    4659    4662*   4663    4668*   4675*   4682*   4683    4686*
                          4689    4693
DDATAB   016732           4715*   4716*   4721*   4724    4728*   4729    4733    4738#
DDISP  = 177570           1187#   1493    2203
DECB1    010526           3084    3085    3087#
DECB1A   010656           3136    3137    3139#
DECB2    011454           3355    3356    3358#
DECB5    012230           3566    3568#
DECB6A   013016           3745    3746    3747    3749#
DECB7    013676           3953    3954    3956#
DEC0     007272           2722    2723    2724    2725    2727#
DEC1     010110           2929    2931#
DEC1A    010360           3027    3029#
DEC2     011210           3263    3264    3266#
DEC5     011714           3446    3447    3448    3450#
DEC6     012452           3637    3638    3640#
DEC7     013326           3838    3839    3841#
DEVICE   001612           1583#
DEVIND   001614           1584#   8319*   8320*   8321*   8325    8327*   8328*   8329
DF1      056475           2040    2045    2050    11157#
DF10     057356           2075   11245#
DF11     057465           2080   11260#
DF12     057556           2085    2090   11272#
DF14     057716           2095   11293#
DF4      057026           2055    2065   11202#
DF5      057125           2060   11215#
DH1      056436           2038   11151#
DH10     057307           2073   11238#
DH11     057374           2078   11250#
DH12     057542           2083   11270#
DH13     057622           2088   11280#
DH14     057664           2093   11287#
DH2      056542           2043   11166#
```

```
DH3       056635        2048    11178#
DH4       056747        2053    11192#
DH5       057071        2058    11210#
DH6       057210        2063    11226#
DISPLA    001244        1493#   2203*   2211*   2480*   9583*   9605*
DISPRE    000174        1450#   2211
DIV0      025634        6316#
DIV1      026102        6400#   6413
DPTRP   = 000001        1437#   6776    6987    7001    7016
DREG      002750        1962#
DSWR    = 177570        1186#   1492    2202
DT1       056502        2039    11160#
DT10      057362        2074    11248#
DT11      057472        2079    11263#
DT12      057560        2084    2089    11274#
DT14      057704        2094    11291#
DT2       056570        2044    11171#
DT3       056704        2049    11186#
DT4       057014        2054    11200#
DT5       057130        2059    11217#
DT6       057230        2064    11230#
DUMMY     035412        8033#
DWORRY    041772        8775#
ECHO      054332        10814#  10832
EISOPT= 040000          1396#
EMTVEC= 000030          1275#   2186*   2187*   5299*   5301*   5338*   5339*
EMT1      021314        5298    5306#
EMT1B     021374        5306    5310    5311    5314    5315    5319    5320    5323    5324    5325    5330#
EMT1C     021400        5304    5332#
EMT1D     021412        5333    5334    5337#
EM1       056411        2037    11147#
EM10      057236        2072    11231#
EM12      057504        2082    11265#
EM13      057564        2087    11275#
EM14      057637        2077    2092    11283#
EM15      057722        2097    11295#
EM2       056514        2042    11162#
EM3       056600        2047    11173#
EM4       056720        2052    11188#
EM5       057032        2057    11204#
EM6       057140        2062    11219#
END       040640        8588#
ENDCP     031524        7259    7329#
ENDM      040606        8188    8572#
ENDMEM    040604        8227    8571#
ENDTAG    057736        2147    8226*   11299#
END1      040640        8589#
ENTER2    037046        8278    8311#   8363    8498
ERFLG     055066        10973#  10989*
ERPRT     055064        2143    2306    2462    5027    5508    6474    6662    6695    6733    6880    6907    6931    6955
                        7090    7142    7173    7186    7209    7224    7312    7425    8234    8589    10972#
ERRADR    001734        1605#   9417*   9429    9478*   9488
ERRBA     001726        1603#
ERRRTN    001340        1527#
ERRVEC= 000004          1268#   2138*   2143*   2200    2201*   2212*   2229*   2230*   2306*   2329*   2354*   2357*   2363*
                        2383*   2403*   2460    4985*   4986*   4987    5027*   5422*   5423*   5465*   5508*   5509*   5510*
```

```
                          6459*   6474*   7303*   7312*   7385*   7386*   7425*   7426*   7427*   8114*   8115*   8227*   8228*
                          8234*   8589*   9546    9547*   9549*   9552*
EXIT      040472          8463    8542    8547*
EXITFL    001600          1578*
EXITRE    036762          8251    8253    8284    8290*   8549
EXPEXT    053210          7947    7958    8003    10492   10514*
EXTINS    023416          5755*
FACTOR    001550          1566*   2498*   2617*   3661    3793*   3907    4364    4365    4646    4707    4823    4824    4828
                          4829    4861*   4932    4986    5112    5127    5135    5151    5176    5213    5408*   5518    5520
                          5531    5748*   5854    6154*   6714    6764    7353*   7605*   8021*   8261    8269*   10703
FAC0.0    002650          1905*
FAC0.1    002652          1906*
FAC0.2    002654          1907*
FAC0.4    002656          1908*
FAC0.5    002660          1909*
FAC1.0    002556          1873*
FAC1.1    002560          1874*
FAC1.2    002562          1875*
FAC1.3    002564          1876*
FAC1.4    002566          1877*
FAC1.5    002570          1878*
FAC2.0    002612          1890*
FAC2.1    002614          1891*
FAC2.2    002616          1892*
FAC2.3    002620          1893*
FAC2.4    002622          1894*
FAC2.5    002624          1895*
FAC3.0    002516          1857*
FAC3.1    002520          1858*
FAC3.2    002522          1859*
FAC3.3    002524          1860*
FAC3.4    002526          1861*
FAC3.5    002530          1862*
FDST0     002664          1911*
FDST1     002574          1880*
FDST2     002630          1897*
FDST3     002534          1864*
FEA       002662          1910*
FLAG      002746          1960*
FLD20  =  104411          9792    10643*
FLTADD    035150          7618    7658    7661    7691    7788    7828    7831    7861    7969*
FLTDBL    053050          7782    10470*
FLTDIV    035122          7751    7921    7955*
FLTMPY    035100          7621    7631    7640    7650    7653    7703    7711    7719    7752    7791    7801    7810    7820
                          7823    7873    7881    7889    7922    7944*
FLTMP0    001464          1552*   7792*   7844    7874*   7905    7923*   11263
FLTMP1    001474          1553*   7832*   7895*   7926*   11263
FLTSGL    053056          7612    10471*
FLTSUB    035144          7676    7699    7724    7737    7759    7846    7869    7894    7907    7929    7968*
FL20   =  104410          9782    10395   10642*
FPOPT  =  020000          1397*
FPSHI     002572          1879*
FREG0     001444          1544*   7616    7627    7649    7689    7696    7716    7749    7757    7758    7760    7786    7797
                          7819    7859    7866    7886    7919    7927    7928    7930    10523
FREG1     001446          1545*   7617    7638    7648    7690    7698    7708    7750    7787    7808    7818    7860    7868
                          7878    7920
```

```
FREG2    001450          1546#    7623*    7675     7705*    7736     7754*    7793*    7845     7875*    7906     7924*    11263    11291
FREG3    001452          1547#    7663*    7677     7726*    7738     7758*    7833*    7847     7896*    7908     7928*    11263    11291
FREG4    001454          1548#
FREG5    001456          1549#
FREG6    001460          1550#
FREG7    001462          1551#
FRSTAD   001554          1568#    2495*    2614*    3790*    4442     4858*    5405*    5745*    6151*    7350*    7602*    8254
FRSTME   001556          1569#    2147*    8264
FTMP0    001424          1536#    7614     7628     7687     7695     7715     7747     7755     7784     7798     7857     7865     7885
                         7917     7925     10472    10487    10519
FTMP1    001426          1537#
FTMP2    001430          1538#    7615     7636     7645     7688     7697     7707     7748
FTMP3    001432          1539#
FTMP4    001434          1540#    7622*    7674     7704*    7735     7753*    7785     7806     7815     7858     7867     7877     7918
                        11291
FTMP5    001436          1541#
FTMP6    001440          1542#    7662*    7725*    7756*    11291
FTMP7    001442          1543#
GNS    = ****** U        1449     2317     2451     6618     8361     8370     8646     8653     9807     10633    10634    10635    10636
                        10637    10640    10641    10642    10643
GNWHAM   002542          1867#
GSTST    007700          2870#
HALT1    031412          7301#
HI     = 000200          1440#
HITMIS=  177752          1343#
HT     = 000011          1178#    9929     9972
INCB1    010442          3053     3054     3056#
INCB2    011550          3390     3391     3392     3394#
INCB3    012122          3524     3526#
INCB6    012630          3688     3690#
INCB6A   013000          3739     3740     3742#
INCB7    013662          3947     3948     3950#
INC0     007312          2731     2732     2733     2734     2736#
INC1     010224          2971     2972     2973     2975#
INC3     011750          3465     3467#
INC4     011072          3216     3217     3219#
INC6     012414          3622     3623     3624     3626#
INC7     013474          3890     3892#
INIT     002757          1969#
IOMON    036770          8235     8300#    8499
IOTTST   021040          5232#
IOTVEC=  000020          1273#    2184*    2185*    5232     5245*    5294*    5295*    5340*    5341*    7274*    7293*
IOWC     001610          1582#    8277*    8310*    8401
JAM      002742          1956#
JMP1     020254          5043     5044     5045     5047#
JMP3     020324          5063     5067#
JMP4     020362          5079     5083#
JMP5     020426          5097     5101#
JMP6     020446          5109#
JMP7     020504          5120#
JSR1     020516          5127#
JSR3     020602          5137     5151#
JSR4     020670          5163     5176#
JSR6     020746          5185     5199#
JSR7     021000          5203     5213#
KERSTK=  001200          1351#    2105     2150     5499     8562     8592
```

```
KIPAR0= 172340          1328#    7517     8103#    8202#    8551#    8558     10693    10887    10912
KIPAR1= 172342          1329#    8104#    8203#    8552#
KIPAR2= 172344          1330#    8105#    8204#    8553#
KIPAR3= 172346          1331#    8113#    8121#    8130     8205#    8206     8208     8302     8551
KIPAR4= 172350          1332#    7532#    8206#    8207#    8552
KIPAR5= 172352          1333#    8208#    8209#    8553
KIPAR6= 172354          1334#
KIPAR7= 172356          1335#    8106#    8210#    8222
KIPDR0= 172300          1317#    7513     8096#
KIPDR1= 172302          1318#    8097#
KIPDR2= 172304          1319#    8098#
KIPDR3= 172306          1320#    8099#
KIPDR4= 172310          1321#    7533#    8100#
KIPDR5= 172312          1322#    8101#
KIPDR6= 172314          1323#
KIPDR7= 172316          1324#    8102#
KM    = 000000          1432#
KTABRT  054662          2478     7292     10924#
KTABT   032364          7530#
KTEX    032570          7531     7574#
KTFLG   054664          10925#   10942#
KTOPT = 100000          1395#
KTPAR   032226          7477#
KTPDR   032110          7438#
LCDTA   002734          1950#
LCUA    002674          1918#
LDKT    054516          9378     9447     9513     9980     10804    10882#
LDRP3   041612          8730     8738#    8997
LDRP4   042514          8866     8873#    9194     9220     9233     9248     9268
LDRS    043044          8927     8936#    9318     9346
LF    = 000012          1179#    9966     9972
LFGIN   002676          1919#
LJAM    002666          1915#
LKOPT = 001000          1399#    2238     8046     8140
LKS   = 177546          1364#    2236     8056#    8058     8144#    9525#
LKSRV   046630          8078     8142     9512#
LKVEC = 000100          1365#    8049#    8050#    8078#    8142#    8143#
LO    = 000100          1439#
LOOP    005530          2437     2439     2441     2443     2445     2447     2458#    8564     8611     8669
LPBA    002672          1917#
LSERV   002670          1916#
LSTMEM  001560          1570#    2145#    2146#    8266
LTAG    002702          1921#
LTICKS  001644          1596#    8426     8443     9514#    9515     9517#    9518#    9519     9521#    9981
LWHAM   002700          1920#
MARKEX  024440          6018     6022     6026#
MARK1   024414          6014     6016#
MBTAS = 160116          1385#
MBTBA = 160104          1380#    1807
MBTBAE= 160174          1389#
MBTCS1= 160100          1378#    1805     1808
MBTCS2= 160110          1382#    1810
MBTCS3= 160176          1390#    1813
MBTDB = 160120          1386#
MBTDT = 160126          1388#    1816
MBTER = 160114          1384#    1812
```

```
MBTERR  046470           9458    9478#
MBTMR1= 160124           1387#
MBTMR2= 160106           1381#   1809
MBTN2   002352           1817#   2283
MBTN3   002354           1818#   2287
MBTN4   002356           1819#   2291
MBTOPT= 002000           1398#   2281    2305    8163
MBTPSW= 000776           1392#   1815
MBTSET  036176           8149    8151    8163#
MBTSRV  046304           8176    9446#
MBTST = 160112           1383#   1811
MBTTBL  002322           1805#   2279    2297    2301*   2302*   2303    8173*   8174*   8175*   8176*   8177*   8178*   9449
                         9454*   9489    9503*   11024
MBTVEC= 000774           1391#   1814
MBTWC = 160102           1379#   1806
MBT1    036242           8173#
MD      002736           1952#
MED   = 076600           1339#   2215    6463    6485    6513    6518    6521    6525    6563    6599    6603    6605    6608
                         6623    6639    6642    6645    6657    6660    6685    6688    6702    6711    6719    6734    6737
                         6780    6782    6784    6789    6795    6800    6806    6812    6844    6849    6887    6895    6916
                         6924    6961    6996    6999    7005    7014    7025    7044    7055    7074    7082    7085    7091
                         7125    7128    7153    7162    7166    7187    7190    7210    7225    10580
MEDEX   031134           6766    6769    7239#
MEDHLT  026362           6487    6488    6489    6491#
MEDTP0  002472           1828#   6515*   6524    6713*   6802*   6988*   7003*   7019    7067*
MEDTP1  002474           1829#   6516*   6534    6808*   7068*
MEDTP2  002476           1830#   6508*   6517    6527    6533*   6535    6567*   6723*   6814*
MEDTP3  002500           1831#   6523*   6527
MEDT1   026374           6506#
MEDT10  026550           6557#
MEDT11  026620           6596#
MED0    026340           6483#
MED1    026166           6447#
MEMERR= 177744           1341#   6852    7036    10733   10866
MEMFLG  001732           1604#   9395*   9416*   9420    9448*   9477*   9481
MEMHOL  046464           9468    9470    9477#
MEMSIZ  003420           2133    2135    2137#
MHOLE   046154           9407    9409    9416#
MMON    001545           1563#   2152*   5564    6767    7266    7443    7483    7530    8156    8171    8225*   8252    8372
                         8418    8449    8469    8496    8516    8547    8581*   10686   10882   10905
MMR0  = 177572           1352#   10935   10941*
MMR2  = 177576           1353#   10936
MMVEC = 000250           1284#   2478*   2479*   7262*   7292*   7536    7537    7540*   7541*   7571*   7572*
MOVB1   014746           4253    4254    4255    4256    4259#
MOV0    014032           3998    3999    4001#
MOV0A   014062           4013    4015#
MOV1    014624           4211    4213#
MOV7    017020           4765    4767#
MPI     031222           7258#
MRKTST  024356           6003#
MSGINX  055236           8355    9773    9801    10053   11026#
MSG1    055262           11036#
MSG10   055517           11030   11066#
MSG11   055525           11031   11068#
MSG12   055533           9812    11070#
MSG13   055646           9819    11083#
```

```
MSG14    055756            9892     11095#
MSG15    056014           11034     11100#
MSG16    056035            9835     11103#
MSG17    056144            9840     11115#
MSG2     055274           11038#
MSG20    056220            8618     11123#
MSG21    056320           11028     11029     11032     11033     11134#
MSG22    056340            9854     11137#
MSG23    056350            9858     11139#
MSG24    056357           11035     11141#
MSG25    056401           10040     11145#
MSG3     055306            9665     11040#
MSG4     055366           10044     11048#
MSG5     055503           11026     11062#
MSG6     055511           11027     11064#
MTICKS   001642            1595#     2160*     9522*     9991
MULO     025504            6264#
MXMMHI   001622            1587#     8136*     9408      9469
MXMMLO   001624            1588#     2175*     8109      8137*     9406      9467
NEGB1    010512            3077      3078      3080#
NEGB4    011500            3367      3368      3370#
NEGB6    012734            3721      3722      3724#
NEGB7    013744            3972      3973      3975#
NEG0     007332            2741      2742      2743      2745#
NEG1     010350            3021      3022      3024#
NEG2     011114            3228      3230#
NEG5     011726            3453      3454      3456#
NEG6     012340            3600      3601      3602      3604#
NEG7     013402            3860      3861      3863#
NEXEC    001544            1562#     2502      2621      3797      4865      5412      5752      6158      7357      7609      8565
NEXPAR   001562            1571#     2153*     2156*     8205      8217      8550*     8577*     8580*
NOMEM    036764            8267      8289      8291#
NOTYPE   001530            1556#     2225*     9957     10796*    10814
NUA      002754            1966#
NULLS    001570            1574#     8028     10835
NWBASH   001606            1581#     8274*     8306*     8415      8424*
NWBASL   001604            1580#     8273*     8305*     8375      8414      8423*     8455
OAERR    020012            4978#
OLDBAS   001602            1579#     8272*     8301*     8374      8413      8422*     8451
OLDPSW   036324            8196#     8563
OPT.CP   001532            1557#     2308*     2446      8023      8046      8140      8148      8163      8593      8614     10041
OVFLW    021752            5416#
PABORT=  000200            1438#     6987
PARFLG   054416           10856#    10871*
PARTBL   032346            7477      7517#
PA1500   001540            1560#     6714*     6715      9401*     9404*     9406      9424      9460*     9465*     9467      9485      9680*     9683
                           9754      9766*     9861*     9865*     9867      9876*    10700*
PA1716   001542            1561#     6724      9402*     9403*     9405*     9408      9425      9464*     9466*     9469      9486      9681*     9767*
                           9864*     9866*     9877*     9878*    10699*
PBA      002744            1958#
PDRTBL   032340            7438      7513#
PERET    054444           10866#
PIRQ   = 177772            1185#
PIRQVE=  000240            1279#
PKM    = 000000            1434#
PLKCSB=  172542            1362#
```

```
PLKCSR= 172540          1361#
PLKVEC= 000104          1363#
PROT    003330          2116#   2136#   2146    2154    8578
PR0   = 000000          1202#
PR1   = 000040          1203#
PR2   = 000100          1204#   6109    6114
PR3   = 000140          1205#
PR4   = 000200          1206#   2222    2224    5234    5252    8197
PR5   = 000240          1207#   5241    6108    8177    8410
PR6   = 000300          1208#
PR7   = 000340          1209#   2463    2473    2475    2481    5240    5284    5295    5339    5349    5350    5380    5382
                        5419    5510    6062    6064    6070    6120    6125    7242    7427    8050    8143    10660
PS    = 177776          1182#   1183
PSW   = 177776          1183#   2453#   2461    2481#   2870    2884#   2889#   5285#   5301    5349#   5355    5383#   5416
                        5417#   5419#   5447#   5452#   5509    5566    5567#   5570    5613    5614#   5678#   5724#   6054
                        6062#   6070#   6108#   6113    6120#   6125    6130#   6168    6172    6176#   6178    6194    6200
                        6352    6356    6376    6380    6447    6455#   6483#   6506#   7258    7291#   7314#   7322    7325
                        7328#   7386    7393    7426    7541    7551    8039    8043#   8062#   8107    8195    8197#   8561#
                        8590#   8977    9015    9050    9085    9125    9160    9196    9235    9270    9299    9332    9368
                        10743   10754#  10885   10886#  10916#
PSWCHK  022524          5564#
PSWHOL  002362          1823#   6447#   7239
PSWTAB  055202          8610    11010#
PTRP1   030310          7002    7013#
PUM   = 030000          1435#   5266    6108    6120    6125
PWRVEC= 000024          1274#   2190#   2191#   10558#  10559#  10568#  10574#  10589#  10590#
QV      001546          1564#   2111#   2119#   2132    2436    8364    8612
RDFLAG= 000144          1405#   6606    6640
RDLCUA= 000103          1415#   6624
RDLDAT= 000106          1421#   6807    7075
RDLFGI= 000104          1417#   6925    7092    7211    7226
RDLJAM= 000100          1409#   6703    6888    6917    6962    7154
RDLPBA= 000102          1413#   6712    6801    6896    7045
RDLSER= 000101          1411#   6720    7026
RDLTAG= 000107          1423#   6813    7056
RDLWHA= 000105          1419#
RDWHAM= 000022          1404#   6600    6659    6686    6735    6997    7033    7125    7163    7188
REGINX  055212          9824    9837    9882    11016#
RELE0   006676          2504    2594#
RELE1   013066          2623    3770#
RELE2   017266          3799    4838#
RELE3   021634          4867    5385#
RELE4   023300          5414    5725#
RELE5   025004          5754    6131#
RELE6   031524          6160    7330#
RELE7   032570          7359    7575#
RELE8   035306          7611    7934    7936    8007#
RELNIO  036726          8271    8280#   8282    8379
RELOC   036574          2597    3773    4841    5388    5728    6134    7333    7578    8010    8250#
RELOCP  040434          8309    8383    8537#   8539
REL0    005674          2492#
REL00   006712          2598#
REL1    006732          2611#
REL11   013102          3774#
REL2    013122          3787#
REL22   017302          4842#
```

```
REL3     017322      4855#
REL33    021650      5389#
REL4     021670      5402#
REL44    023314      5729#
REL5     023334      5742#
REL55    025020      6135#
REL6     025040      6148#
REL66    031540      7334#
REL7     031560      7347#
REL77    032604      7579#
REL8     032624      7599#
REL88    035322      8011#
REPADR   001742      1608#   10867*  10870   10937*  10940   10961*  10964
REPMRO   001744      1609#   10935*  11186
REPMR2   001746      1610#   10936*  11186
REPPSW   001736      1606#   10863*  10932*  10958*  10980*  10990   11160   11171   11186   11200
REPREG   001740      1607#   10733*  10866*  10984*  10985*  10988   11160   11200   11217
RES      002755      1967#
RESERR   054774      2307    2466    5028    5557    6475    7313    10949#
RESET1   031476      7321#
RESFLG   054776     10951#   10965*
RESKT    054604      9389    9411    9438    9452    9501    9523    10024   10812   10905#
RESPSW   054126     10750#   10756
RESREG=  104407      8680    8728    8752    8802    8822    8864    8896    8926    9390    9412    9439    9453    9502
                     9524    9727    10025   10242   10367   10420   10641#  10701   10813
RESTPS   054136     10754#
RESTRP   022360      5517#
RESVEC=  000010      1269#   2231*   2232*   2307*   4987*   5028*   5519*   5520*   5557*   6460*   6475*   7304*   7313*
                     8410*   8411*   8520*
RETPC    036766      8263*   8292#
RETPSW   054120     10745#   10755
RETRY    036534      8227#   8546
REV      002751      1963#
RKBA     002206      1751#   8805*   9106*   9142*
RKCS     002202      1749#   2334*   2343*   8505*   8811*   9076    9080*   9082    9096    9107*   9113    9120*   9122
                     9128    9143*   9149    9155*   9157    9855
RKDA     002210      1752#   2333*   8803*   9104*   9140*
RKDRV    041644      1687    8748#
RKDS     002176      1747#   2337    8809    11017
RKER     002200      1748#   2335
RKERR    044036      9079    9088#   9118    9153
RKFUN    002136      1710#   8808*   9060*   9061    9066*   9068    9119*   9154*   9231*
RKHANA   002104      1687#
RKHDA    002122      1699#   8795*   8803    9104    9140
RKHSTA   001754      1618#   8749*   8750    8755*   9074    9088*   9095*   9098*   9111    9147    9163*
RKHWC    001774      1631#   8804    9105    9141
RKLOOP   044064      9075    9095#   9112    9148
RKNEWH   002044      1661#   9134    9136*   9139
RKNEWL   002042      1660#   9142
RKOLD    002012      1643#   8798    8801*   8805    9103    9106
RKPSW    002214      1754#   8807*
RKREAD   044364      9071    9147#
RKRPT    043700      8753    9059#
RKSRV    043730      8806    9066#
RKTRY    002147      1722#   8796*   9078    9084*   9101*   9117    9124*   9132*   9151    9159*
RKUNIT   002070      1676#   8792
```

```
RKVEC    002212          1753#   8806*
RKWC     002204          1750#   8804*   9105*   9141*
RKWRCK   044164          9069    9111#
RKWTRY   042140          8803#   9064    9086
RK1      044110          9062    9077    9101#
RK10     001632          1591#   8797*   8800*   8811    9102*   9103*   9107    9138*   9139*   9143
RK11     001634          1592#   8754*   8774    9090    9115
RK2      044132          9104#   9126
RK3      044332          9065    9140#   9161
RNTBIN   001620          1586#   8312*   8406*   8407    8420    8682    8754    8824    8898
ROLB1    010454          3059    3060    3062#
ROLB2    011372          3325    3326    3327    3329#
ROLB3    012176          3549    3550    3552#
ROLB6    012750          3727    3728    3730#
ROLB6A   013044          3757    3758    3760#
ROLB7    013762          3979    3980    3982#
ROLD     007364          2757    2758    2760#
ROL1     010144          2943    2944    2945    2947#
ROL1A    010156          2950    2951    2953#
ROL3     011776          3477    3478    3480#
ROL4     011170          3253    3254    3256#
ROL6     012270          3581    3582    3583    3585#
ROL7     013512          3896    3897    3899#
RORB1    010542          3091    3092    3094#
RORB1A   010632          3124    3125    3127#
RORB4    011354          3317    3318    3319    3321#
RORB5    012140          3530    3531    3532    3534#
RORB6    012702          3708    3709    3711#
RORB7    013646          3941    3942    3944#
RORD     007250          2713    2714    2715    2716    2718#
ROR1     010074          2921    2922    2924#
ROR1A    010206          2963    2964    2965    2967#
ROR2     011056          3209    3210    3212#
ROR5     011650          3426    3427    3429#
ROR6     012360          3608    3609    3610    3612#
ROR7     013364          3853    3854    3856#
RPCC     002246          1769#
RP3BA    002164          1740#   8741*   9034*
RP3CS    002160          1738#   2367*   2368    8729*   8736*   8969    8973*   8974    8988    8998*   9004    9011*   9012
                         9035*   9041    9046*   9047    9857
RP3DA    002166          1741#   8717*   8738*   9031*
RP3DC    002170          1742#   8739*   9032*
RP3DRV   041250          1686    8676#
RP3DS    002154          1736#   2374    8734    11016
RP3ER    002156          1737#   9018
RP3ERR   043260          8972    8980#   9009    9044
RP3FUN   002134          1709#   8733*   8954*   8955    8960*   8961    9010*   9045*
RP3HAN   002102          1686#   8411    8520
RP3HDA   002116          1697#   8738    9031
RP3HDC   002120          1698#   8716*   8739    9032
RP3HST   001752          1617#   2164    8384    8389    8435    8501*   8519*   8522    8524    8677*   8678    8967    8980*
                         8987*   8990*   9002    9039    9053*
RP3HWC   001772          1630#   8401*   8740    9033
RP3LOO   043306          8968    8987#   9003    9040
RP3NWH   002040          1659#   8415*   9024    9026*   9029
RP3NWL   002036          1658#   8414*   9034
```

```
RP3OLD   002006          1641#   8413*   8416*   8720    8723*   8741    8995
RP3PSW   002174          1744#   8732*
RP3REA   043606          8964    9039#
RP3RPT   043130          8681    8953#
RP3SRV   043160          8731    8960#
RP3TRY   002146          1721#   8718*   8971    8976*   8993*   9008    9014*   9022*   9043    9049*
RP3UNI   002066          1675#   8409*   8724     727*   8729    8996    9030
RP3VEC   002172          1743#   8731*
RP3WC    002162          1739#   8740*   9033*
RP3WCK   043376          8962    9002#
RP3WTR   041544          8729#   8958    8978
RP31     043332          8956    8970    8993#
RP310    001626          1589#   8719*   8722*   8726*   8736    8994*   8995*   8996*   8998    9028*   9029*   9030*   9035
RP311    001630          1590#   8682*   8697    8982    9006
RP32     043362          8997#   9016
RP33     043546          8959    9031#   9051
RP4BA    002222          1759#   8885*   9255*
RP4BAE   002224          1760#
RP4CS1   002216          1757#   2358    2388*   8870*   8882*   8883*   9200*   9201    9221*   9252*   9253*   9256*   11020
RP4CS2   002230          1762#   2387*   2389    2397*   8513*   8873*   9193*   9232*   9238    9267*
RP4CS3   002232          1763#
RP4DA    002226          1761#   8876*
RP4DC    002240          1766#   8875*
RP4DRV   042230          1690    8818#
RP4DS    002234          1764#   2391    9189    9212    9218    9227    9246    9262
RP4ERR   044650          9192    9204#   9230    9265
RP4ER1   002236          1765#
RP4ER2   002242          1767#
RP4ER3   002244          1768#
RP4FUN   002142          1712#   8869*   9176*   9177    9181*   9182    9266*
RP4HAN   002112          1690#
RP4HDA   002126          1701#   8862*   8876
RP4HDC   002130          1702#   8861*   8875
RP4HST   001762          1621#   8819*   8820    8825*   9187    9198    9204*   9211*   9214*   9225    9260    9273*
RP4HWC   002002          1634#   8877
RP4LOO   044676          9188    9211#   9226    9261
RP4NWH   002060          1667#   9250
RP4NWL   002056          1666#   9255
RP4OF    002250          1770#   8874*
RP4OLD   002026          1649#   8879    8885
RP4PSW   002254          1772#   8868*
RP4REA   045142          9184    9260#
RP4RPT   044504          8823    9175#
RP4SRV   044530          8867    9181#
RP4TRY   002151          1724#   8863*   9191    9195*   9217*   9229    9234*   9245*   9264    9269*
RP4UNI   002076          1679#   8873
RP4VEC   002252          1771#   8867*
RP4WC    002220          1758#   8877*
RP4WCK   044754          9183    9225#
RP4WTR   042462          8865#   9180    9197
RP41     044724          9178    9190    9217#
RP411    001636          1593#   8824*   8841    9206    9240
RP42     044732          9218#   9219    9236
RP43     045064          9179    9246#   9247    9271
RSBA     002262          1777#   8946*   9354*
RSBAE    002264          1778#
```

```
RSCS1   002256        1775#   2408*   8933*   8943*   8944*   9319*   9351*   9352*   9355*   11021
RSCS2   002270        1780#   2407*   2409    2417*   8510*   8936*   9297*   9330*   9336    9366*
RSCS3   002272        1781#
RSDA    002266        1779#   8937*
RSDRV   042614        1691    8892#
RSDS    002274        1782#   2411    8931    9293    9310    9316    9325    9344    9361
RSER    002276        1783#
RSERR   045350        9296    9302#   9328    9364
RSFUN   002144        1713#   8930*   9280*   9281    9285*   9286    9329*   9365*
RSHANA  002114        1691#
RSHDA   002132        1703#   8924*   8937
RSHSTA  001764        1622#   8893*   8894    9291    9302*   9309*   9312*   9323    9359    9371*
RSHWC   002004        1635#   8938
RSLOOP  045376        9292    9309#   9324    9360
RSNEWH  002064        1669#   9348
RSNEWL  002062        1668#   9354
RSOLD   002032        1651#   8940    8946
RSPSW   002302        1785#   8929*
RSREAD  045642        9288    9359#
RSRPT   045234        8897    9279#
RSSRV   045260        8928    9285#
RSTRY   002152        1725#   8925*   9295    9298*   9315*   9327    9331*   9343*   9363    9367*
RSUNIT  002100        1680#   8936
RSVEC   002300        1784#   8928*
RSWC    002260        1776#   8938*
RSWCK   045454        9287    9323#
RSWTRY  043004        8927#   9284    9300
RS11    001640        1594#   8898*   8913    9304    9338
RS41    045424        9282    9294    9315#
RS42    045432        9316#   9317    9333
RS43    045560        9283    9344#   9345    9369
RTI1    041034        8613    8615    8620#
RTT1    024452        6054#
RTT1EX  024642        6090#
RTT2A   024656        6097    6099#
RTT2EX  024776        6056    6105    6115    6117    6126    6128    6130#
RT1     054510       10874#
RT1A    002546        1869#
RT1B    002642        1902#
RT2A    002550        1870#
RT2B    002644        1903#
RUNTBL  001666        1599#   8408*   8476    8698    8701*   8775    8778*   8842    8845*   8915*
RUNTRA  001702        1600#   8532    8715*   8789*   8860*   8921*   8983*   9091*   9207*   9305*
RVECT   002646        1904#
RZERO   002640        1901#
R0A     002720        1940#
R0B     002724        1942#
R1A     002502        1851#
R1B     002576        1884#
R2A     002504        1852#
R2B     002600        1885#
R3A     002506        1853#
R3B     002602        1886#
R4A     002510        1854#
R4B     002604        1887#
R5A     002512        1855#
```

```
R5B       002606        1888#
R6A       002514        1856#
R6B       002610        1889#
R7A       002722        1941#
R7B       002726        1943#
SAVPSW    024654        6054*   6055    6060    6085    6098#   6099    6130
SAVREG=   104406        8676    8748    8818    8892    9377    9446    9512    9662    9979    10224   10317   10392   10640#
                        10684   10802
SBCB1     010466        3065    3066    3068#
SBCB3     012216        3560    3562#
SBCB4     011466        3361    3362    3364#
SBCB6     012604        3678    3679    3681#
SBCB7     013574        3919    3920    3921    3923#
SBC0      007414        2771    2772    2774#
SBC1      010240        2978    2979    2980    2982#
SBC1A     010256        2986    2987    2988    2990#
SBC5      012010        3484    3486#
SBC6      012374        3615    3616    3618#
SBC7      013270        3823    3824    3826#
SBINB7    017520        4907#   4938*   4939    4943    4950    4954    4958*   4961*   4964    4968*   4971*   4974
SBIN7     016746        4748#   4763*   4764    4773    4779    4786*   4787    4794    4798    4802
SDATA     016274        4643#   4649*   4650    4651*   4652*   4657*   4658*   4659    4663    4668    4674*   4675    4681*
                        4686    4687*   4688*   4689    4692*   4693    4696
SDATAB    016730        4713*   4714*   4715    4717    4721    4724    4728    4729    4737#
SECT2     033302        7682    7685#
SECT2D    034406        7852    7855#
SECT3     033646        7743    7747#
SECT3D    034752        7913    7917#
SERV      002743        1957#
SIZE      004634        2329#   2455
SOBDBL    053206        10470*  10471   10477   10489   10493*  10495#
SOB0      024150        5915    5926#
SOB1      024156        5928#   5932
SOB10     024132        5917#   5980
SOB2      024204        5928    5929    5930    5931    5933    5934    5935    5936    5939#
SOB3      024216        5943#   5951
SOB4      024250        5943    5944    5945    5946    5949    5952    5953    5956#
SOB5      024252        5958#
SOB5A     024272        5967#
SOB6      024316        5978#
SOB7      024324        5981#
SOB8      024326        5924    5983#
SOB9      024142        5922#   5993
SPARE0    001756        1619#
SPARE1    001760        1620#
SPCHK     023016        5636#
SRCADR    001750        1611#   10735*  11217
SREG      002752        1964#
SR0     = 177572        1288#   1352    7556    7573*   8111*   8138*   8224*   8572*
SR1     = 177574        1289#
SR2     = 177576        1290#   1353    7559
SR3     = 172516        1291#
STACK   = 001200        1173#   1337    1351    2182
START     003276        1453    2105#   10594
STGBLK    002370        1826#
STKLMT=   177774        1184#   7361
```

```
STMM     036304          8153     8155     8157     8164     8166     8168     8170     8172     8186#    8567
SUPFLG   035304          7968*    7997     8004*    8006#
SU[illegible]RS   001574          1576#    2167*    8152     8167     8597*    8598     8602*    8608     9697
SUB0     014050          4006     4007     4009#
SUB1     014472          4159     4160     4162#
SUB1A    014600          4198     4199     4201#
SUB1B    014614          4205     4206     4208#
SUB2     015344          4391     4392     4394#
SUB2A    015464          4433     4434     4436#
SUB3     016002          4543     4544     4546#
SUB3A    016024          4551     4552     4554#
SUB6     016454          4676     4678#
SUB7     017070          4780     4781     4782     4784#
SWAB0    007432          2778     2779     2780     2782#
SWAB1    010672          3143     3144     3146#
SWAB2    011104          3223     3225#
SWAB4    011532          3384     3386#
SWAB6    013030          3752     3754#
SWAB7    013416          3867     3869#
SWB01 =  000071          1426#    6596
SWB1A =  000710          1427#    6626
SWITCH   057740         10039    11302#
SWR      001242          1492#    2180     2202*    2204     2210*    2444     8020*    8092*    8150     8165     8186     8250     8270
                         8307     8315     8319     8526     8683     8756     8826     8899     9541     9559     9566     9606     9613
                         9618     9621    10566    10582*
SWREG    000176          1451#    2210
SW0    = 000001          1237#
SW00   = 000001          1227#    1237
SW01   = 000002          1226#    1236
SW02   = 000004          1225#    1235
SW03   = 000010          1224#    1234
SW04   = 000020          1223#    1233
SW05   = 000040          1222#    1232
SW06   = 000100          1221#    1231
SW07   = 000200          1220#    1230
SW08   = 000400          1219#    1229
SW09   = 001000          1218#    1228
SW1    = 000002          1236#
SW10   = 002000          1217#
SW11   = 004000          1216#
SW12   = 010000          1215#    8150
SW13   = 020000          1214#
SW14   = 040000          1213#
SW15   = 100000          1212#
SW2    = 000004          1235#
SW3    = 000010          1234#    8165
SW4    = 000020          1233#    8683     8756     8826     8899
SW5    = 000040          1232#    8315
SW6    = 000100          1231#    8186     8250
SW7    = 000200          1230#    2444
SW8    = 000400          1229#    8270     8307
SW9    = 001000          1228#
SXRA     023772          5864#    5873*
SXRB     023774          5865#    5891
SXT0     023440          5758     5759     5760     5761     5764#
SXT1     023524          5791#
```

H04


```
SXT2     024010            5871#
SXT3     024024            5877#
SXT4     023572            5809     5811#
SXT5     024034            5882#
SXT6     023734            5846     5847     5848     5849     5851#
SXT6A    023764            5857     5860#
SXT7     024100            5899#
SYSSIZ   001646            1598#    2346*    2380*    2400*    2420*    2425*    2433*    2435*    8332     8335     8345     8352*    8394*
                          10047    10058
TABBEG   002364            1824#
TABEND   002366            1825#
TAG    = 000040            1441#
TBITVE=  000014            1270#    5350*    5382*    5569     5580*    5583*    5607*    6059*    6061*    6088*    6089*    8607*
TBL2     002502            1847#    6507
TBL4     002720            1939#
TBL5     002734            1948#    6632
TBL6     002762            1976#    6633
TBL7     003034            2004#    6559
TEST1    010334            3013     3014     3015     3017#
TEST2    011032            3197     3198     3200#
TEST6    013060            3763     3764     3766#
TIMEBU   051236            2168     2173*    2174*    9985*    9990*    9995*   10000*   10007*   10014*   10021*   10022    10027#
TIMER    044466            9081     9121     9156     9165#
TKBFR    001506            1554     1555#   10808    10810
TKBFRP   001504            1554#   10805    10811*
TKISR    054152            2221    10776#
TKVEC  = 000060            1277#    2221*    2222*
TLOC1    002360            1822#    6772     6786     6836     6842     6989     7003     7046     7059
TLOC2    002470            1827#
TPISR    054364            2223     8082    10830#
TPVEC  = 000064            1278#    2223*    2224*    8048*    8082*
TRAPVE=  000034            1276#    2188*    2189*    5353*    5355*    5361*    5364*    5368*    5373*    5379
TRAP1    021546            5352     5367#
TRAP1C   021606            5366     5379#
TRTVEC=  000014            1271#
TSTB1    010704            3149     3150     3152#
TSTB2    011510            3373     3375#
TSTB2A   011520            3378     3380#
TSTB6    012556            3666     3667     3668     3670#
TST0     007170            2686     2687     2688     2689     2691#
TST1     005656            2485#
TST10    011252            3287#
TST100   032100            7434#
TST101   032216            7474#
TST102   032354            7511     7527#
TST103   032606            7592#
TST104   033774            7765     7778#
TST105   035324            8017#
TST106   035606            8089#
TST11    011570            3405#
TST12    012012            3490#
TST13    012232            3572#
TST14    012512            3657#
TST15    013104            3780#
TST16    013514            3903#
TST17    014010            3993#
```

```
TST2    006714          2604#
TST20   014346          4117#
TST21   014672          4237#
TST22   015222          4359#
TST23   015506          4447#
TST24   015712          4518#
TST25   016102          4582#
TST26   016262          4639#
TST27   016560          4701#
TST3    007116          2669#
TST30   016734          4736     4744#
TST31   017154          4809#
TST32   017304          4848#
TST33   017506          4903#
TST34   020214          5032#
TST35   020506          5124#
TST36   021030          5229#
TST37   021234          5247     5290#
TST4    007434          2786#
TST40   021442          5346#
TST41   021652          5395#
TST42   022350          5514#
TST43   022514          5561#
TST44   022756          5619#
TST45   023164          5686#
TST46   023316          5735#
TST47   024116          5910#
TST5    010030          2909#
TST50   024346          6000#
TST51   024442          6051#
TST52   024642          6094#
TST53   025022          6141#
TST54   025474          6261#
TST55   025624          6313#
TST56   026072          6397#
TST57   026150          6443#
TST6    010362          3033#
TST60   026330          6479#
TST61   026364          6503#
TST62   026540          6553#
TST63   026610          6593#
TST64   027076          6677#
TST65   027324          6759#
TST66   027564          6833#
TST67   027702          6872#
TST7    010770          3184#
TST70   030074          6945#
TST71   030164          6983#
TST72   030550          7121#
TST73   030762          7197#
TST74   031212          7255#
TST75   031402          7298#
TST76   031466          7318#
TST77   031542          7340#
TTOPT = 000400          1400#    2242     8023
TTYCHK  035342          8021#
```

```
TYPDS  = 104405      8650    8657    9703    10637#
TYPE   = 104401      2315    2449    6616    8357    8359    8368    8644    8651    8658    9608    9616    9663    9665
                     9666    9670    9687    9689    9692    9696    9700    9704    9717    9719    9722    9724    9728
                     9748    9758    9775    9785    9795    9803    9805    9812    9819    9835    9840    9845    9854
                     9858    9871    9873    9887    9889    9890    9892    9895    9934    10022   10023   10039   10040
                     10043   10044   10055   10062   10065   10129   10204   10591   10633#
TYPOC  = 104402      9669    9691    9695    9699    9745    9844    9886    9894    10042   10634#
TYPON  = 104404      10636#
TYPOS  = 104403      10635#
TYPSIZ   051250      2448    10039#
TYPTIM   051010      8633    9664    9979#
UBEADR   001722      1602#   9384*   9385*   9386    9387    9433    9435    10663   10664
UBEBA  = 170004      1370#   1793
UBECC  = 170002      1369#   1792
UBECLR = 170010      1372#   1796
UBECR1 = 170006      1371#   1795    2247*   2248
UBECR2 = 170016      1374#   1794
UBEDB  = 170000      1368#   2245    2253    2258    2263
UBEERR   046160      9400    9417#
UBEGO  = 170014      1373#
UBEINI   053646      8158    9410    10649#
UBEOPT = 000200      1401#   2250
UBESAV   001716      1601#   9382*   9383*   9384    9385    10649
UBESET   036122      8141    8148#
UBESRV   045730      9377#   10659
UBETBL   002304      1792#   2269    9379    9391*   9413*   9431    9875    10657   11025
UBEVEC = 000510      1375#   1797    1798
UBE2     046030      9381    9395#
UBE3     046230      9418    9423    9429#
UBM6     013064      3660    3662*   3665    3673    3677*   3684*   3686*   3692*   3700*   3707*   3713*   3720*   3726*
                     3732*   3738*   3744*   3751*   3756*   3762    3769#
UBREAK = 177770      1347#   6596*   6637*
UB7      016756      4747    4753#
UCNSCT   002772      1982#
UCOUNT   003020      1993#
UCUA     003004      1987#
UDCS0    003026      1996#
UDCS1    003032      1998#
UDREG    003012      1990#
UFLAG    003006      1988#
UINIT    003030      1997#
UIPAR0 = 177640      1306#   7268    7518    8217*   8218    8220
UIPAR1 = 177642      1307#   8218*   8219*
UIPAR2 = 177644      1308#   8220*   8221*
UIPAR3 = 177646      1309#
UIPAR4 = 177650      1310#   7534*
UIPAR5 = 177652      1311#
UIPAR6 = 177654      1312#   7268*
UIPAR7 = 177656      1313#   8222*
UIPDR0 = 177600      1295#   7514    8213*
UIPDR1 = 177602      1296#   8214*
UIPDR2 = 177604      1297#   8215*
UIPDR3 = 177606      1298#
UIPDR4 = 177610      1299#   7535*
UIPDR5 = 177612      1300#
UIPDR6 = 177614      1301#   7269*
```

```
UIPDR7= 177616          1302#   8216*
UJAM    002776          1984#
ULCDTA  002762          1978#
UM    = 140000          1433#   5259    5433    6101    6104    6125    7258    7393    7402
UMD     002766          1980#
UNITNO  001616          1585#   8400*   8404    8409
UNUA    003022          1994#
UPBA    003002          1986#
URES    003024          1995#
UREV    003014          1991#
UR6A    002532          1863#
UR6B    002626          1896#
USERV   003000          1985#
USESTK= 000700          1337#   2458    5026    5286    5458    5493    5507    6265    6456    7251    7294    7399    7419
                        7423    10983
USREG   003016          1992#
UWM6    012244          3576#   3580*   3587*   3592*   3599*   3607*   3614*   3621*   3628*   3630*   3636*   3642*   3648*
UWM7    013210          3802#   3906    3909*   3914*
UW7     013214          3807    3805#
VADR    001536          1559#   6708*   8286*   8466*   8544*   9671    9672*   9678*   9680    9682*   10685   10734*  10864*
                        10865*  10933*  10934*  10959*  10960*  10981*  10982*  11160   11171   11186   11200   11217   11230
                        11248
WCNSSW= 000226          1407#   10581
WCSADR  002636          1900#
WCSA.0  002536          1865#
WCSA.1  002540          1866#
WCSB.0  002632          1898#
WCSB.1  002634          1899#
WRFLAG= 000344          1408#   6609    6643
WRINIT= 000352          1425#   2216
WRLCUA= 000303          1416#
WRLDAT= 000306          1422#   6783
WRLFGI= 000304          1418#
WRLJAM= 000300          1410#
WRLPBA= 000302          1414#   6781
WRLSER= 000301          1412#
WRLTAG= 000307          1424#   6785
WRLWHA= 000305          1420#
WRWHAM= 000222          1406#   6604    6661    6689    6738    7000    7086    7129    7167    7191
WWP   = 000100          1436#   6774    6838    6840    6992    6995
X       054512          10872   10875#
XOR0    023472          5772    5773    5774    5777#
XOR1    023560          5797    5798    5799    5800    5802    5805#
XOR24   023614          5819#
XOR35   024064          5888    5889    5893#
XOR6    023702          5824    5827    5829    5836#
XOR6A   023706          5822*   5823*   5828*   5830*   5834    5840#   5845*   5856*   5858*
XOR6B   023710          5831*   5832*   5833*   5834    5841#
XOR7    024114          5904#
XXDP    003324          2113#   2126*   2134
XXDPC   003325          2114#   2129*   2438
$AC0    001410          1530#   7617*   7620*   7627*   7629    7638*   7639    7648*   7652*   7656*   7660*   7675*   7690*
                        7698*   7702*   7708*   7710    7716*   7718    7723*   7736*   7750*   7757*   7787*   7790*   7797*
                        7799    7808*   7809    7818*   7822*   7826*   7830*   7845*   7860*   7868*   7872*   7878*   7880
                        7886*   7888    7893*   7906*   7920*   7927*   7944    7955    7969    7974    7985    7987    10543
$AC1    001412          1531#   7616*   7620    7623    7629*   7633    7639*   7642    7649*   7663    7671    7677*   7680*
```

```
                              7681     7689*    7693     7696*    7705     7710*    7713     7718*    7720     7726     7732     7738*    7741*
                              7742     7749*    7754     7760*    7762*    7763     7786*    7790     7793     7799*    7803     7809*    7812
                              7819*    7833     7841     7847*    7850*    7851     7859*    7863     7866*    7875     7880*    7883     7888*
                              7890     7896     7902     7908*    7911*    7912     7919*    7924     7930*    7932*    7933     7944*    7955*
                              7969     7973     7986     7987*
$AC2      001414              1532#    7633*    7660     7668     7670*    7671*    7672*    7673*    7681     7693*    7702     7713*    7723
                              7729     7731*    7732*    7733*    7734*    7742     7803*    7830     7838     7840*    7841*    7842*    7843*
                              7851     7863*    7872     7883*    7893     7899     7901*    7902*    7903*    7904*    7912
$AC3      001416              1533#    7642*    7656     7668     7670     7720*    7729     7731     7812*    7826     7838     7840     7890*
                              7899     7901
$AC4      001420              1534#
$AC5      001422              1535#
$AUTOB    001236              1489#
$BDADR    001224              1484#
$BDDAT    001230              1486#    9425*    9486*
$BELL     001330              1522#    9608     9633
$BUFF     001406              1529#    10398*   10423
$CHARC    051004              9936*    9946*    9953     9964*    9969#
$CKSWR= ****** U              10640
$CMTAG    001200              1471#    2177     2178     2186     2192     2193     2194
$CM1    = 000010              1504#    1505#    1506#    1507#    1508#    1509#    1510#    1511#    1512#
$CM2    = 000020              1504#    1505#    1506#    1507#    1508#    1509#    1510#    1511#    1512#
$CM3    = 000010              1502#    1504
$CM4    = 000010              1512#    1513#    1514#    1515#    1516#    1517#    1518#    1519#    1520#
$CRLF     001335              1524#    8658     9616     9633     9663     9666     9704     9719     9724     9728     9873     9889     9890
                              9935     9972     10023    10043    10065    10780    10788    10798
$DBLK     052050              10170    10204    10212#
$DB20     052060              9684     9755     9868     10224#
$DOAGN    041240              8640     8661     8667#
$DTBL     052040              10173    10208#
$ENDAD    041230              1460     2117     2313     8663#    9628
$ENDCT    041074              2192     8642#
$ENULL    041244              8670#
$EOP      041036              8604     8632#
$EOPCT    041066              2192*    8639#    8643
$ERFLG    001204              1474#    9533     9555     9557     9563*    9580     9582     9587     9602     9603*    9633
$ERMAX    001217              1481#    2195*    9557     9579*    9587
$ERPSW    001576              1577#
$ERROR    047160              2186     5294     5338     7205     9601#
$ERRPC    001220              1482#    9610*    9611*    9612     9633     9667     9672
$[illegible]RTB   003126              2035#    9714
$[illegible]RTY   047334              9615     9662#
$ERTTL    001214              1479#    8655     8659*    9609*    9633
$ESCAP    001326              1521#    2194*    9578*    9624     9626     9633
$FACTO    001552              1567#    8260*    8376     8453     8456
$FILLC    00126[illegible]              1500#    9939     9972
$FILLS    001257              1499#    [illegible]972
$FLBUF    001342              1528#    10[illegible]22    10370
$FLD20    0[illegible]2600              10392#   106[illegible]3
$FL20     052312              10317#   10642
$GDADR    001222              1483#
$GDDAT    00122[illegible]              1485#    9424*    9485*    11274
$GET42    041220              8660#
$GTSWR= ****** U              10639
$HD     = 000000              1113
$HINUM    001566              1573#    2106*    8689     8762     8832     8905     10443    10451    10456*   10479    10483
```

```
$ICNT    001206          1476#    9570*    9571     9573*    9586
$ILLUP   053550         10558    10574    10596#
$INTAG   001237          1490#
$ITEMB   001216          1480#    9612*    9633     9673     9675     9706
$LF      001336          1525#    9633     9972
$LONUM   001564          1572#    2107*    8690     8763     8833    10442    10449    10455*   10484
$LPADR   001210          1477#    2196*    2501*    2620*    3796*    4864*    5411*    5751*    6157*    7356*    7608*    9561*    9576*
                         9584     9586
$LPERR   001212          1478#    2197*    2499*    2500*    2501     2618*    2619*    2620     3794*    3795*    3796     4862*    4863*
                         4864     5409*    5410*    5411     5749*    5750*    5751     6155*    6156*    6157     7354*    7355*    7356
                         7606*    7607*    7608     8300*    8446     8492*    9417     9418*    9429*    9478     9479*    9488*    9561
                         9577*    9586     9623    10867    10868*   10870*   10875    10937    10938*   10940*   10943    10961    10962*
                        10964*   10966    10985    10986*   10988*   10991
$MAIL  = ****** U        2214     9576     9618     9923
$MXCNT   047156          9574     9586#
$NULL    001256          1498#    9941     9972
$NWTST=  000001          2482#    2601#    2666#    2783#    2906#    3030#    3181#    3284#    3402#    3487#    3569#    3654#    3777#
                         3900#    3990#    4114#    4234#    4356#    4444#    4515#    4579#    4636#    4698#    4740#    4742     4805#
                         4807     4845#    4900#    5029#    5121#    5224#    5226     5287#    5343#    5392#    5511#    5558#    5616#
                         5683#    5732#    5905#    5907     5997#    6027#    6029     6091#    6138#    6255#    6257     6306#    6308
                         6391#    6393     6436#    6438     6476#    6493#    6495     6543#    6545     6576#    6578     6669#    6671
                         6749#    6751     6826#    6828     6861#    6863     6935#    6937     6971#    6973     7106#    7108     7194#
                         7252#    7295#    7315#    7337#    7429#    7431     7470#    7472     7521#    7523     7582#    7584     7768#
                         7770     8014#    8085#    8087
$OCNT    051630         10101*   10130*   10143#
$OCTVL   052200         10226    10254#
$OMODE   051632         10096*   10100*   10105    10108*   10119*   10145#
$OVER    047126          9542     9562     9572     9580#
$PASS    001200          1472#    2151*    2440     8157     8169     8636*    8637*    8648     8670     9568     9587     9701
$POWER   053556         10592    10599#
$PWRAD   053544         10594#
$PWRDN   053376          2190    10558#   10589
$PWRMG   053540         10592#
$PWRUP   053450         10568    10574#
$QUES    001334          1523#    9633     9972
$RAND    052752          8688     8761     8831     8904    10438#   10474
$RDCHR=  ****** U       10640
$RDDEC=  ****** U       10640
$RDLIN=  ****** U       10640
$RDOCT=  ****** U       10640
$REGAD   001262          1502#
$REG0    001264          1504#
$REG1    001266          1505#
$REG2    001270          1506#
$REG3    001272          1507#
$REG4    001274          1508#
$REG5    001276          1509#    8028*    8608*    8618*   10780*   10788*   10798*   10830*   10831    10835*
$REG6    001300          1510#
$REG7    001302          1511#
$RESRE   052254         10288#   10641
$RTNAD   041242          8669#
$RTRN    001534          1558#    2468     6088     8607
$R2A   = ****** U       10642
$SAVPA   054652         10894    10908    10919#
$SAVPS   054660         10885*   10916    10920#
$SAVRE   052216         10272#   10640
```

```
$SAVR6  053554        10567*  10575   10576*  10577*  10598*
$SCOPE  046720        2184    5245    5251    5340    7220    7293    9540*
$SETUP= 000037        2158*   2183    2184    2186    2188    2190    2192    2193    2194    2196    2313    2316    8634
                      9541    9602    9621    9628
$STUP = 177777        2158*
$SVLAD  047104        9550    9575*
$SVPC = 000204        1458*   1463
$SWR  = 167377        1104*   1113    1127    1128    1129    1130    1131    1132    1133    1134    1520    1521    1522
                      2193    2194    2196    2197    2489    2608    2672    2789    2912    3036    3187    3290    3408
                      3493    3575    3660    3784    3906    3996    4120    4240    4362    4450    4521    4585    4642
                      4704    4747    4812    4852    4906    5035    5127    5232    5293    5349    5339    5517    5564
                      5622    5689    5739    5913    6003    6054    6097    6145    6264    6316    6400    6446    6482
                      6506    6556    6596    6680    6762    6836    6875    6948    6986    7124    7200    7258    7301
                      7321    7344    7437    7477    7530    7596    7781    8020    8092    8629    8635    8662    8668
                      8670    9533    9534    9535    9536    9537    9541    9553    9555    9556    9557    9564    9565
                      9566    9577    9580    9586    9593    9594    9595    9596    9597    9606    9613    9618    9621
                      9633    10595
$SWRMK= 000000        9537
$TIMES  001324        1520*   2193*   2486*   2605*   2789*   2912*   3781*   4849*   5396*   5736*   6142*   6446*   6762*
                      7200*   7341*   7593*   7781*   8635*   9564*   9571    9574*   9586
$TKB    001250        1495*   10776
$TKS    001246        1494*   2220*
$TMP0   001304        1512*   2330*   2333    2344*   2364*   2367    2371*   2384*   2387    2398*   2404*   2407    2418*
                      8285*   8313*   8317    8322*   8339    8386    8391    8465*   8543*   11230   11248
$TMP1   001306        1513*   8330*   8332    8335    8337*   8394    8396*   8487*   8490*
$TMP2   001310        1514*   8311*   8486*   11248
$TMP3   001312        1515*   8484*   11248
$TMP4   001314        1516*
$TMP5   001316        1517*
$TMP6   001320        1518*
$TMP7   001322        1519*
$TN   = 000107        1104*   1113    2482    2489*   2492    2601    2608*   2611    2666    2672*   2783    2789*   2906
                      2912*   3030    3036*   3181    3187*   3284    3290*   3402    3408*   3487    3493*   3569    3575*
                      3654    3660*   3777    3784*   3787    3900    3906*   3990    3996*   4114    4120*   4234    4240*
                      4356    4362*   4444    4450*   4515    4521*   4579    4585*   4636    4642*   4698    4704*   4736
                      4740    4747*   4805    4812*   4845    4852*   4855    4900    4906*   5029    5035*   5121    5127*
                      5224    5232*   5247    5287    5293*   5343    5349*   5392    5399*   5402    5511    5517*   5558
                      5564*   5616    5622*   5683    5689*   5732    5739*   5742    5905    5913*   5997    6003*   6027
                      6054*   6091    6097*   6138    6145*   6148    6255    6264*   6306    6316*   6391    6400*   6436
                      6446*   6476    6482*   6493    6506*   6543    6556*   6576    6596*   6669    6680*   6749    6762*
                      6826    6836*   6861    6875*   6935    6948*   6971    6986*   7106    7124*   7194    7200*   7252
                      7258*   7295    7301*   7315    7321*   7337    7344*   7347    7429    7437*   7470    7477*   7511
                      7521    7530*   7582    7596*   7599    7765    7768    7781*   8014    8020*   8085    8092*
$TPB    001254        1497*   9961*   9972    10821*
$TPFLG  001261        1501*   9917    9972
$TPS    001252        1496*   2240    8026    8029*   8041    8044*   8073*   8083*   8595    8609*   8616    8619*   9959
                      9972    10781*  10789*  10799*  10817*  10819   10834*
$TRAP   053566        2188    5361    5364    5368    5373    5381    10610*
$TRAP2  053610        10621*  10632
$TRP  = 000012        10625*  10634*  10635*  10636*  10637*  10638*  10640   10641*  10642*  10643*  10644*
$TRPAD  053622        10615   10632*
$TSTNM  001202        1473*   2488*   2492*   2607*   2611*   2671*   2788*   2911*   3035*   3186*   3289*   3407*   3492*
                      3574*   3659*   3783*   3787*   3905*   3995*   4119*   4239*   4361*   4449*   4520*   4584*   4641*
                      4703*   4746*   4811*   4851*   4855*   4905*   5034*   5126*   5231*   5292*   5348*   5398*   5402*
                      5516*   5563*   5621*   5688*   5738*   5742*   5912*   6002*   6053*   6096*   6144*   6148*   6263*
                      6315*   6399*   6445*   6481*   6505*   6555*   6595*   6679*   6761*   6835*   6874*   6947*   6985*
```

B05


```
                          7123*   7199*   7257*   7300*   7320*   7343*   7347*   7436*   7476*   7529*   7595*   7599*   7780*
                          8019*   8020    8091*   8092    8226    8560*   8634*   9533    9582*   9583    9587    9602*   9605
                          9633    9693
$TYPBN= ****** U          10638
$TYPDS  051634            10158#  10637
$TYPE   050562            9917#   10625   10633
$TYPEC  050732            9938    9945    9952    9957#   9960
$TYPEX  051006            9958    9965    9967    9970#
$TYPOC  051432            10099#  10634
$TYPON  051446            10098   10101#  10636
$TYPOS  051406            10094#  10635
$XTSTR  046730            9544#
$$GET4= 000000            8662#
$OFILL  051631            10095*  10099*  10109   10144#
$4OCAT= ****** U          9541    9615
.     = 061314            1445#   1449#   1458    1459#   1461#   1463#   1470#   1475#   1526    1528#   1552#   1553#   1555#
                          1598#   1599#   1600#   1601#   1602#   1603#   1726#   1826#   2181    2196    2197    2318#   2497
                          2503    2616    2622    2633    2641    2649    2656    2664    2680    2690    2700    2708    2717
                          2726    2735    2744    2753    2759    2766    2773    2781    2807    2817    2827    2834    2841
                          2850    2866    2913    2923    2930    2939    2946    2952    2959    2966    2974    2981    2989
                          2994    3002    3008    3016    3023    3028    3036    3049    3055    3061    3067    3073    3079
                          3086    3093    3100    3108    3115    3119    3126    3131    3138    3145    3151    3155    3160
                          3166    3170    3176    3179    3187    3199    3204    3211    3218    3224    3229    3233    3241
                          3248    3255    3259    3265    3269    3275    3279    3282    3290    3304    3308    3312    3320
                          3328    3335    3343    3351    3357    3363    3369    3374    3379    3385    3393    3397    3400
                          3408    3421    3428    3434    3441    3449    3455    3461    3466    3472    3479    3485    3493
                          3511    3514    3520    3525    3533    3539    3544    3551    3555    3561    3567    3584    3590
                          3595    3603    3611    3617    3625    3633    3639    3645    3649    3652    3669    3674    3680
                          3689    3696    3703    3710    3716    3723    3729    3735    3741    3748    3753    3759    3765
                          3768    3792    3798    3818    3825    3833    3840    3847    3855    3862    3868    3875    3879
                          3886    3891    3898    3922    3930    3936    3943    3949    3955    3961    3968    3974    3981
                          3988    4000    4008    4014    4023    4031    4051    4059    4069    4077    4085    4088    4098
                          4103    4112    4120    4135    4144    4152    4161    4168    4175    4181    4189    4200    4207
                          4212    4219    4227    4232    4240    4258    4262    4269    4280    4285    4291    4296    4302
                          4305    4354    4374    4381    4386    4393    4404    4414    4421    4427    4435    4440    4450
                          4483    4491    4495    4501    4508    4513    4545    4553    4558    4561    4567    4572    4577
                          4612    4615    4621    4628    4634    4642    4655    4660    4F54    4671    4677    4684    4690
                          4694    4718    4722    4725    4730    4734    4766    4769    4,70    4776    4783    4791    4795
                          4799    4803    4836    4860    4866    4872    4878    4883    4887    4893    4898    4940    4945
                          4951    4955    4965    4975    4980    5016    5018*   5039    5041    5046    5051    5059    5061
                          5066    5082    5086    5100    5108    5119    5207    5238    5254    5273    5280    5298    5328
                          5329    5336    5352    5360    5363    5367    5372    5407    5413    5421    5495    5573    5588
                          5644    5651    5659    5665    5676    5697    5712    5721    5747    5753    5763    5776    5787
                          5790    5804    5818    5835    5838    5850    5859    5863    5870    5876    5881    5892    5898
                          5903    5919    5938    5955    5959    5961    5964    5989    5995    6015    6020    6024    6058
                          6111    6123    6153    6159    6182    6205    6225    6249    6273    6285    6304    6336    6360
                          6384    6416    6458    6472    6490    6681    6683    7261    7273    7302    7309    7326    7352
                          7359    7384    7411    7539    7547    7569    7604    7610    8362#   8548    8596    8647#   8670
                          8671#   8810    8963    8975    9013    9048    9070    9083    9123    9158    9586    9587    9633
                          9899#   9972    10212#  10254#  10570   10597   10820   10919#  11185#  11199#  11229#  11262#  11273#
.PARSR  054414            2476    6798    6859    7017    10729   10854#
```

```
COMMEN   1280#
ENDCOM   1280#
ERROR    1174#   1430    5342    7683    7744    7766    7853    7914    7935    8288    8445    8495    8500    8545    9422
         9426    9483    9487    10736   10869   10939   10963   10987
ESCAPE   1280#
FLTST1   7582#   7584    7770
GETPRI   1280#
GETSWR   1280#   2316#
HLT      1430#   2634    2642    2650    2657    2665    2681    2691    2701    2709    2718    2727    2736    2745    2754
         2760    2767    2774    2782    2808    2818    2828    2835    2842    2851    2867    2901    2924    2931    2940
         2947    2953    2960    2967    2975    2982    2990    2995    3003    3009    3017    3024    3029    3050    3056
         3062    3068    3074    3080    3087    3094    3101    3109    3116    3120    3127    3132    3139    3146    3152
         3156    3161    3167    3171    3177    3180    3200    3205    3212    3219    3225    3230    3234    3242    3249
         3256    3260    3266    3270    3276    3280    3283    3305    3309    3313    3321    3329    3336    3344    3352
         3358    3364    3370    3375    3380    3386    3394    3398    3401    3422    3429    3435    3442    3450    3456
         3462    3467    3473    3480    3486    3512    3515    3521    3526    3534    3540    3545    3552    3556    3562
         3568    3585    3591    3596    3604    3612    3618    3626    3634    3640    3646    3650    3653    3670    3675
         3681    3690    3697    3704    3711    3717    3724    3730    3736    3742    3749    3754    3760    3766    3819
         3826    3834    3841    3848    3856    3863    3869    3876    3880    3887    3892    3899    3923    3931    3937
         3944    3950    3956    3962    3969    3975    3982    3989    4001    4009    4015    4024    4032    4047    4052
         4060    4070    4078    4086    4089    4099    4104    4113    4136    4145    4153    4162    4169    4176    4182
         4190    4201    4208    4213    4220    4228    4233    4259    4270    4281    4286    4292    4297    4303    4355
         4375    4382    4387    4394    4405    4415    4422    4428    4436    4441    4484    4492    4496    4502    4509
         4514    4546    4554    4559    4562    4568    4573    4578    4613    4616    4622    4629    4635    4656    4661
         4666    4672    4678    4685    4691    4695    4719    4723    4726    4731    4735    4767    4771    4777    4784
         4792    4796    4800    4804    4873    4879    4884    4888    4894    4899    4941    4946    4952    4956    4966
         4976    4991    4993    4995    4997    4999    5001    5003    5005    5007    5009    5011    5013    5015    5017
         5019    5047    5067    5083    5101    5109    5120    5148    5173    5184    5196    5204    5208    5222    5246
         5281    5293#   5305    5330    5337    5342#   5362    5365    5369    5374    5459    5496    5526    5527    5528
         5574    5589    5594    5631    5645    5652    5660    5666    5677    5698    5713    5722    5764    5777    5791
         5805    5819    5836    5851    5860    5871    5877    5882    5893    5899    5904    5920    5939    5956    5960
         5965    5971    5975    5981    5990    5996    6017    6021    6025    6075    6080    6116    6127    6183    6206
         6226    6250    6286    6305    6337    6361    6385    6417    6465    6469    6491    6529    6568    6613    6629
         6649    6692    6699    6706    6729    6792    6820    6846    6856    6884    6892    6899    6910    6920    6928
         6951    6959    6966    7007    7022    7030    7040    7050    7069    7078    7095    7146    7159    7180    7214
         7229    7278    7290    7306    7327    7371    7400    7420    7465    7506    7548    7567    8070
MSG      6668#   6671    6748#   6751    6825#   6828    6860#   6863    6934#   6937    6970#   6973    7105#   7108
MSGA     4740#   4742
MSGB     4805#   4807
MSGC     5223#   5226
MSGD     5905#   5907
MSGE     6027#   6029
MSGF     6255#   6257
MSGG     6306#   6308
MSGH     6391#   6393
MSGJ     6436#   6438
MSGK     7428#   7431
MSGL     7469#   7472
MSGM     7521#   7523
MSGM1    6492#   6495
MSGM10   6542#   6545
MSGM11   6575#   6578
MSGO     8084#   8087
MULT     1280#
MYTAGS   1464#   1527
NEWTST   1280#   1464#   2482    2601    2666    2783    2906    3030    3181    3284    3402    3487    3569    3654    3777
```

```
          3900    3990    4114    4234    4356    4444    4515    4579    4636    4698    4740    4805    4845    4900    5029
          5121    5223    5287    5343    5392    5511    5558    5616    5683    5732    5905    5997    6027    6090    6138
          6255    6306    6391    6436    6476    6493    6543    6576    6669    6749    6826    6861    6935    6971    7106
          7194    7252    7295    7315    7337    7428    7469    7521    7582    7768    8014    8084
POP       12[illegible]0#  10199   10293   10457   10582   10583
PUSH      12[illegible]0#  10158   10273   10438   10560   10566
RELOCA    2102#   2489    2608    3784    4852    5399    5739    6145    7344    7596
RELOCB    2102#   2594    3770    4838    5385    5725    6131    7329    7575    8007
REPORT    1280#
SCOPE     1175#   2487    2594    2606    2670    2787    2910    3034    3185    3288    3406    3491    3573    3658    3770
          3782    3904    3994    4118    4238    4360    4448    4519    4583    4640    4702    4745    4810    4838    4850
          4904    5033    5125    5230    5291    5347    5385    5397    5515    5562    5620    5687    5725    5737    5911
          6001    6052    6095    6131    6143    6262    6314    6398    6444    6480    6504    6554    6594    6678    6760
          6834    6873    6946    6984    7122    7198    7256    7299    7319    7330    7342    7435    7475    7528    7575
          7594    7779    8007
SETPRI    1280#
SETTRA   10625#  10634   10635   10636   10637   10640   10641   10642   10643
SETUP     1280#   2176
SKIP      1280#   2437    2439    2441    2443    2445    2447    4736    5247    7510    7765
SLASH     1280#
SPACE     1280#
STARS     1280#   1456    1466    1526    2104    2319    2328    2348    2360    2362    2382    2402    2482    2484    2601
          2603    2666    2668    2783    2785    2906    2908    3030    3032    3181    3183    3284    3286    3402    3404
          3487    3489    3569    3571    3654    3656    3777    3779    3900    3902    3990    3992    4114    4116    4234
          4236    4356    43[illegible]3  4444    4446    4515    4517    4579    4581    4636    4638    4698    4700    4740    4743
          4805    4808    48[illegible]5  4847    4900    4902    5029    5031    5121    5123    5224    5228    5287    5289    5343
          5345    5392    5394    5511    5513    5558    5560    5616    5618    5683    5685    5732    5734    5905    5909
          5997    5999    6027    6050    6091    6093    6138    6140    6255    6260    6306    6312    6391    6396    6436
          6442    6476    6478    6493    6502    6543    6552    6576    6592    6669    6676    6749    6758    6826    6832
          6861    6871    6935    6944    6971    6982    7106    7120    7194    7196    7252    7254    7295    7297    7315
          7317    7337    7339    7429    7433    7470    7473    7521    7526    7582    7591    7684    7746    7768    7777
          7854    7916    7938    7943    7950    7954    7961    7967    8014    8016    8085    8088    8146    8161    8180
          8185    8236    8249    8293    8299    8582    8587    8625    8672    8675    8744    8747    8814    8817    8888
          8891    8949    8952    9055    9058    9171    9174    9275    9278    9373    9376    9442    9445    9505    9511
          9530    9589    9633    9660    9741    9750    9762    9769    9778    9788    9798    9814    9822    9847    9880
          9902    9972    9978   10032   10038   10071   10148   10215   10257   10300   10316   10373   10391   10426   10461
         10469   10496   10513   10556   10572   10604   10644   10648   10667   10683   10708   10719   10740   10742   10747
         10749   10758   10771   10825   10829   10838   10853   10876   10881   10901   10904   10921   10923   10946   10948
         10969   10971   10995   11300
SWRSU     1280#   2198#
TRMTRP   10625#
TYPBIN    1280#
TYPDEC    1280#   8648    8655    9701
TYPNAM    1104#   1280#   2309
TYPNUM    1280#
TYPOCS    1280#
TYPOCT    1280#   9667
TYPTXT    1280#   2449    6616    8359    8368    8644    8651    9805
UPCODE    1464#  10579
$$CMRE    1464#   1504    1505    1506    1507    1508    1509    1510    1511
$$CMTM    1464#   1512    1513    1514    1515    1516    1517    1518    1519
$$ESCA    1280#
$$NEWT    1280#   1464#   2482    2601    2666    2783    2906    3030    3181    3284    3402    3487    3569    3654    3777
          3900    3990    4114    4234    4356    4444    4515    4579    4636    4698    4740    4805    4845    4900    5029
          5121    5224    5287    5343    5392    5511    5558    5616    5683    5732    5905    5997    6027    6091    6138
          6255    6306    6391    6436    6476    6493    6543    6576    6669    6749    6826    6861    6935    6971    7106
```

E05


```
         7194    7252    7295    7315    7337    7429    7470    7521    7582    7768    8014    8085
$$SET   10625#  10634   10635   10636   10637   10640   10641   10642   10643
$$SKIP   1280#   4736    5247    7511    7765
.EQUAT   1104#   1170
.HEADE   1104#
.KT11    1104#   1280
.SETUP   1104#   2157
.SWRHI   1104#   1123
.SWRLO   1104#   1135#   1138    1141
.$ACT1   1104#   1454
.$CATC   1104#   1443
.$CMTA   1464#
.$DB20   1464#  10213
.$EOP    1104#   8623
.$ERRO   1104#   9587
.$POWE   1104#  10554
.$RAND   1464#  10426
.$RDOC   1104#
.$READ   1104#
.$SAVE   1104#  10255
.$SCOP   1464#   9528
.$TRAP   1104#  10602
.$TYPD   1104#  10146
.$TYPE   1464#   9900
.$TYPO   1104#  10069

. ABS.  061314     000

 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 DSKZ:DQKDCA,DSKZ:DQKDCA.SEQ/CRF/SOL/DS:ERFZ=DSKZ:DQKDCA.P11
 RUN-TIME: 27 31 3 SECONDS
 RUN-TIME RATIO: 680/63=10.7
 CORE USED:  31K  (62 PAGES)
```