PDP11/70

CPU INSTRUC EXERCISER

MD-11-DEQKC-B

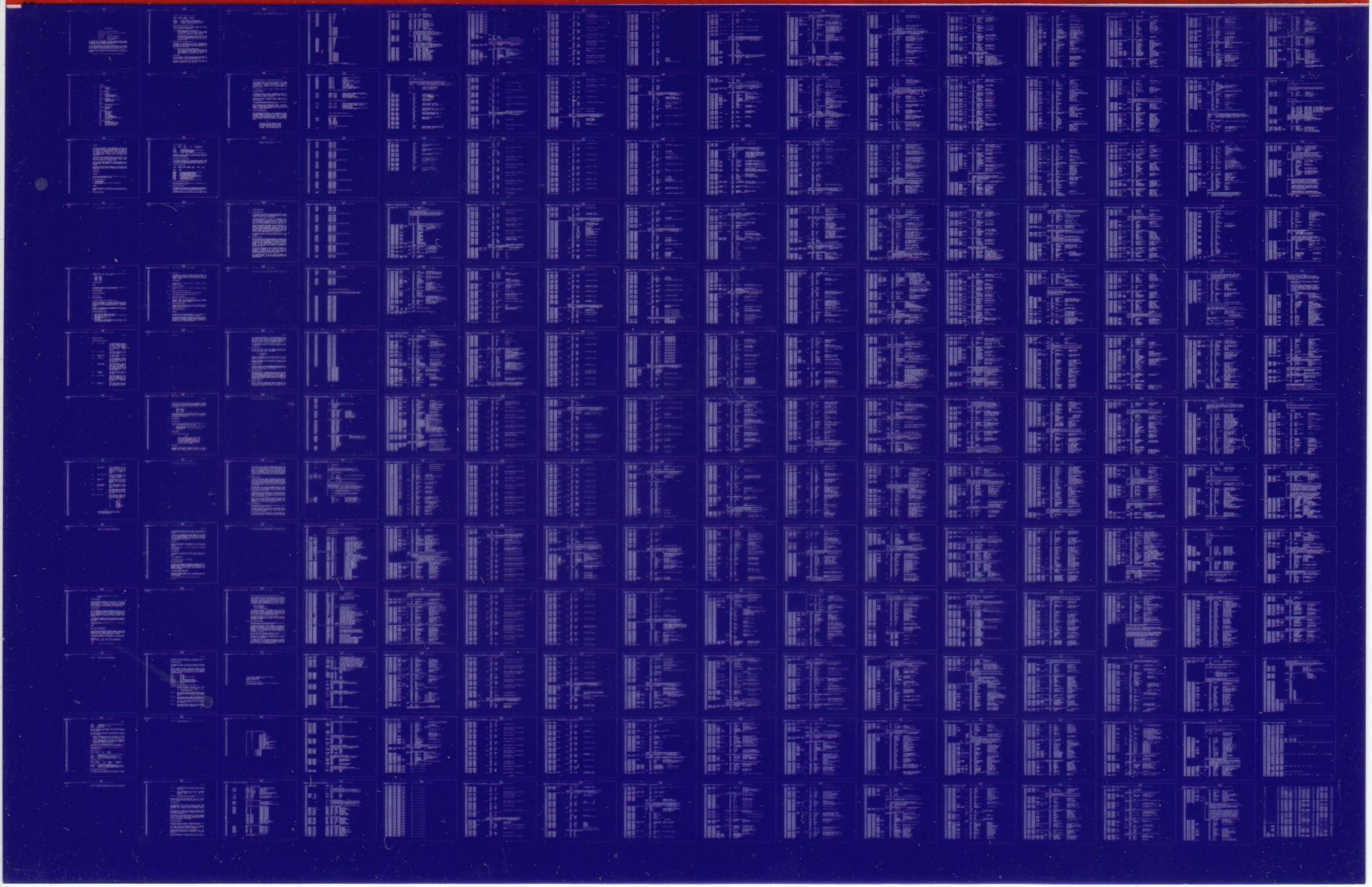
EP-DEQKC-B-DL-A

FICHE 1 OF 2

NOV 1976

digital

MADE IN USA



PDP11/70

CPU INSTRUCTION EXER.

MD-11-DEQKC-B

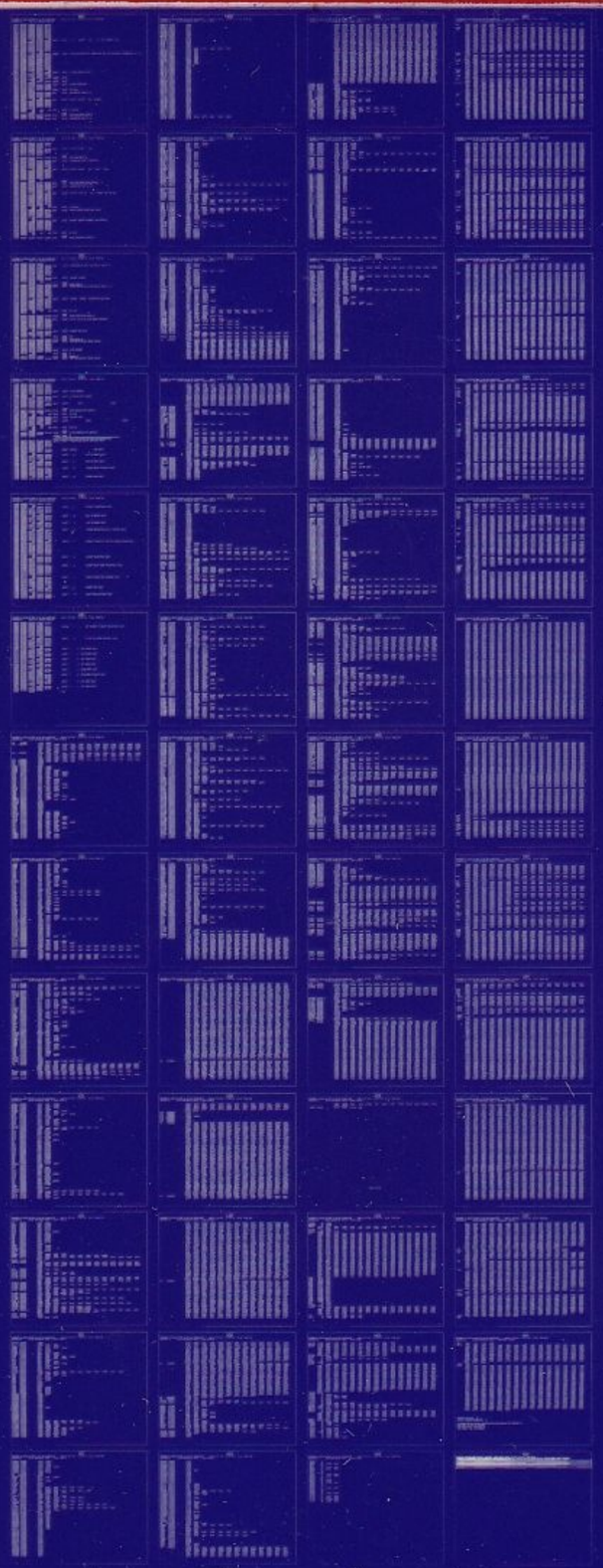
EP-DEQKC-B-DL-A

FICHE 2 OF 2

NOV 1976

digital

MADE IN USA



```
                                .REM    %
```

IDENTIFICATION

PRODUCT CODE: MAINDEC-11-DEQKC-B-D

PRODUCT NAME: PDP11/70 CPU INSTRUCTION EXERCISER

DATE CREATED: MAY 21,1976

MAINTAINER: DIAGNOSTIC ENGINEERING

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

CONTENTS

1.0 ABSTRACT

THIS PROGRAM IS DESIGNED TO BE A COMPREHENSIVE CHECK OF THE PDP-11/70 CPU CLUSTER. THE PROGRAM EXECUTES EACH INSTRUCTION IN ALL ADDRESS MODES AND INCLUDES TESTS FOR TRAPS, INTERRUPTS, THE MAPPING BOX, MEMORY MANAGEMENT, MEMORY, THE UNIBUS, AND THE MASS BUS. IF NOT DESELECTED, THE PROGRAM RELOCATES THE TEST CODE THROUGHOUT MEMORY (0-2M). ALSO, IF NOT DESELECTED, THE PROGRAM WILL RELOCATE USING AVAILABLE DISKS (RP03,RK05,RP04,RS03/4). SEE SECTION 9.5 FOR A DESCRIPTION OF RELOCATION.

THE MAIN DIFFERENCES BETWEEN REVISION A AND REVISION B ARE ROUTINES TO USE THE UBE AND MBT (MANUFACTURING ONLY), WORST CASE TESTING OCCURS WITH ALL SWITCHES DOWN, STANDARD SYSMAC MACROS, AND FLOATING POINT PROCESSOR TESTS.

ALSO, THE DISK DRIVER WAS REWRITTEN TO MAKE EACH DEVICE HAVE A MODULAR DRIVER AND TO CAUSE I/O TO OCCUR CONCURRENTLY ON THE AVAILABLE DISKS. (SEE SECTION 9.5.4 FOR A DESCRIPTION OF DISK DRIVERS)

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

2.0 REQUIREMENTS

2.1 EQUIPMENT

PDP-11/70 CENTRAL PROCESSOR WITH 16K OF MEMORY, A LINE CLOCK, AND AN LA30 (OR EQUIVALENT) CONSOLE.

2.1.1 OPTIONAL EQUIPMENT USED

1. UNIBUS EXERCISER
2. MASS BUS TESTER
3. RP11/RP03, RK11/RK05, RH70/RP04, RH70/RS03/RS04
4. FP11-B, FP11-C

2.2 STORAGE

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

2.3 PRELIMINARY PROGRAMS

```
155
156                                     ALTHOUGH THIS PROGRAM IS A TEST OF  THE  CPU  CLUSTER,  IT  IS
```

ADVISABLE THAT THE CPU CLUSTER (AND FLOATING POINT) DIAGNOSTICS RUN FIRST. THESE CONSIST OF:

DEKBA DEKBF
DEKBB DEKBG
DEKBC DEMJA
DEKBD DEFPA
DEKBE DEFPB

3.0 LOADING PROCEDURE

3.1 METHOD

THE PROGRAM IS SUPPLIED ON THE DIAGNOSTIC MEDIA. REFER TO THE XXDP OPERATING MANUAL FOR FURTHER INFORMATION.

4.0 STARTING PROCEDURE

4.1 CONSOLE SWITCH SETTINGS

SEE SECTION 5.1

4.2 STARTING ADDRESSES

THE STARTING ADDRESS FOR THE EXERCISER IS 200.

BY STARTING AT ADDRESS 210, THE SWITCH REGISTER AND DISPLAY LIGHTS CAN BE CHECKED. THIS ROUTINE JUST MOVES THE SWITCHES TO THE DISPLAY REGISTER ALLOWING THE OPERATOR TO TOGGLE THE SWITCHES AND SEE THE CORRESPONDING LIGHTS IN THE DISPLAY REGISTER.

BY STARTING AT ADDRESS 214, THE MICRO-BREAK REGISTER CAN BE CHECKED. THIS TEST REQUIRES A MAINTENANCE CARD. SEE SECTION 9.0 FOR FURTHER DETAILS.

4.3 PROGRAM AND OPERATOR ACTION

1. LOAD PROGRAM INTO MEMORY (SEE SECTION 3)
2. CHECK FOR ANY SYSTEM DISK PACKS OR CONFIGURATION EXCEPTIONS AS DESCRIBED IN SECTION 7.0.
3. LOAD ADDRESS 200
4. SET SWITCHES (SEE SECTION 5.1)
5. PRESS START
6. THE PROGRAM WILL LOOP AND MESSAGES WILL BE TYPED AT THE END OF EACH SUB-PASS AND EACH PASS. (SEE SECTION 8.3 FOR A DESCRIPTION OF THE MESSAGES)

5.0 OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

SW15	HALT ON ERROR	THIS SWITCH WHEN SET WILL HALT THE PROCESSOR WHEN AN ERROR IS DETECTED. PRESSING CONTINUE WILL CAUSE AN ERROR MESSAGE TO BE TYPED AND THE PROCESSOR WILL AGAIN HALT. PRESSING CONTINUE AGAIN WILL RESUME TESTING.
SW14	LOOP ON TEST	THIS SWITCH WHEN SET WILL CAUSE THE PROGRAM TO LOOP ON THE CURRENT SUBTEST.
SW13	INHIBIT ERROR TYPEOUT	THIS SWITCH WHEN SET INHIBITS THE ERROR TYPEOUT.
SW12	INHIBIT UBE	THIS SWITCH WHEN SET INHIBITS THE INITIALIZATION OF THE UNIBUS EXERCISER. SEE SECTION 9.2 FOR A DESCRIPTION OF THE UBE FUNCTION.
SW11	INHIBIT SUB-TEST ITERATION	THIS SWITCH WHEN SET INHIBITS SUBTEST ITERATION AFTER THE FIRST PASS. EACH SUBTEST IS EXECUTED 10 TIMES BEFORE THE NEXT SUBTEST IS RUN. SETTING SW11 CAUSES EACH TEST TO BE EXECUTED ONCE BEFORE STARTING THE NEXT SUBTEST.
SW10	RING BELL ON ERROR	THIS SWITCH WHEN SET WILL RING THE BELL WHEN AN ERROR IS DETECTED.
SW9	LOOP ON ERROR	THIS SWITCH WHEN SET WILL CAUSE THE PROGRAM TO LOOP ON THE FIRST FAILURE EVEN IF THE FAILURE IS INTERMITTANT. SEE SECTION 6.1 FOR A DESCRIPTION OF LOOPING ON RELOCATION ERRORS.
SW8	RELOCATE WITH CPU ONLY	THIS SWITCH WHEN SET WILL CAUSE RELOCATION TO BE DONE BY THE CPU INSTEAD OF A DISK. SEE SECTION 9.5 FOR A

269 DESCRIPTION OF RELOCATION.

SW7	INHIBIT SYSTEM SIZE TYPEOUT	THIS SWITCH WHEN SET WILL INHIBIT THE TYPEOUT OF THE SWITCH DEFINITIONS AND THE DISKS THAT WILL BE USED FOR RELOCATION. (TYPEOUT ONLY OCCURS WHEN THE PROGRAM IS DUMPED)
SW6	INHIBIT RELOCATION	THIS SWITCH WHEN SET WILL INHIBIT ALL RELOCATION. DO NOT CHANGE THIS SWITCH WHILE THE PROGRAM IS RUNNING.
SW5	INHIBIT ROUND ROBIN	THIS SWITCH WHEN SET WILL ONLY RELOCATE USING THE DEVICE SELECTED BY SWITCHES <2:0> RATHER THAN ALL AVAILABLE DEVICES.
SW4	INHIBIT RANDOM DISK ADDRESS	THIS SWITCH WHEN SET WILL CAUSE RELOCATION TO ALWAYS START AT ADDRESS 0 ON THE DISK(S).
SW3	INHIBIT MBT	THIS SWITCH WHEN SET INHIBITS THE INITIALIZATION OF THE MASS BUS TESTER. SEE SECTION 9.3 FOR A DESCRIPTION OF THE MBT FUNCTION.
SW2-SW0	DEVICE CODES	THESE SWITCHES (ALONG WITH SW5) CAUSE THE PROGRAM TO RELOCATE THE TEST CODE USING THE DEVICE SPECIFIED BELOW:

VALUE	DEVICE
0	RP11/RP03
1	RK05
2	NOT USED
3	NOT USED
4	RH70/RP04
5	RH70/RS03/RS04
6	NOT USED
7	NOT USED

NOTE

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

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

MODE, AND THEREFORE WILL NOT BE USED TO RELOCATE.

5.2 DISPLAY REGISTER

WHILE THE PROGRAM IS RUNNING, THE LOW BYTE OF THE DISPLAY REGISTER CONTAINS THE SUBTEST NUMBER AND THE HIGH BYTE CONTAINS BITS <14:7> OF KERNEL PAR0. THESE BITS, OF KERNEL PAR0, CORRESPOND TO BITS <20:13> OF THE PHYSICAL ADDRESS OF THE RELOCATED CODE. WHEN AN ERROR IS DETECTED AND LOOP ON ERROR IS SELECTED, THE HIGH BYTE CONTAINS THE ERROR COUNT.

5.3 OPERATOR ACTION

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

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

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

6.0 ERRORS

6.1 ERROR HALTS AND DESCRIPTION

IF AN ERROR IS DETECTED, THE PROGRAM WILL TRAP TO THE ERROR HANDLING ROUTINE ($ERROR). IF HALT ON ERROR IS ENABLED, THE PROCESSOR WILL HALT. PRESSING CONTINUE WILL CAUSE AN ERROR MESSAGE TO BE TYPED AND THE PROCESSOR WILL HALT AGAIN.

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

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

WHERE:

UUUUUU = VIRTUAL PC OF THE ERROR CALL.

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

XXXXXX = CONTENTS OF THE MAINTENANCE REGISTER(17777750).
YYYYYY = TEST NUMBER.
SSSSSS = SUB-PASS COUNT (0 THRU 5)
PPPPPP = PASS COUNT

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

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

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

2. IF MEMORY MANAGEMENT IS ON THE CONTENTS OF THE APPROPRIATE PAR IS SHIFTED AND ADDED TO THE VIRTUAL PC TO GENERATE A PHYSICAL 22 BIT ADDRESS. IN THIS CASE THE VIRTUAL PC CORRESPONDS TO THE NON-RELOCATED CODE.

THE CONTENTS OF THE MAINTENANCE REGISTER WILL INDICATE WHAT MEMORY MARGIN WAS BEING PERFORMED WHEN THE ERROR OCCURRED.

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

6.1.1 UNEXPECTED TRAP TO 4

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

VVVVVV = VIRTUAL PC THAT WAS PUSHED ON THE STACK WHEN THE TRAP OCCURRED.
PPPPPPPP = PHYSICAL PC CALCULATED AS DESCRIBED ABOVE.
YYYYYY = PSW THAT WAS PUSHED ON THE STACK.
ZZZZZZ = CONTENTS OF THE CPU ERROR REGISTER(17777766).

6.1.2 UNEXPECTED TRAP TO 114

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

V, P, AND Y = ARE THE SAME AS DESCRIBED IN 6.1.1.
ZZZZZZ = CONTENTS OF THE MEMORY ERROR REGISTER (777744).
EEEEEEEE = CONTENTS OF THE ERROR ADDRESS REGISTERS COMBINED INTO A 22 BIT ADDRESS (777740 & 777742).

6.1.3 PARITY ERROR DURING DATA CHECK

THIS ERROR CAN ONLY OCCUR DURING THE DATA CHECK THAT IS MADE ON THE RELOCATED TEST CODE BEFORE IT IS EXECUTED. THIS CHECK

IS MADE BY COMPARING THE UNRELOCATED CODE WITH THE RELOCATED
CODE. THE SOURCE DATA REFERS TO THE UNRELOCATED CODE AND THE

DESTINATION DATA TO THE RELOCATED CODE.

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

SSSSSS     = VIRTUAL ADDRESS OF THE SOURCE DATA.
DDDDDDDD   = PHYSICAL ADDRESS OF THE DESTINATION DATA.
EEEEEEEE   = CONTENTS OF THE ERROR ADDRESS REGISTERS.
ZZZZZZ     = CONTENTS OF MEMORY ERROR REGISTER (777744).
```

6.1.4 ERROR DURING DATA CHECK-RELOC WAS BY CP

THIS ERROR IS SIMILAR TO 6.1.3 EXCEPT INSTEAD OF A PARITY ERROR, IT IS A DATA COMPARISON ERROR. REFER TO SECTION 9.5.3 FOR A DESCRIPTION OF CP RELOCATION.

LOOP ON ERROR (SW<9>) HAS THE FOLLOWING EFFECT:

1. MEMORY MANAGEMENT OFF- IF SWITCH<9> IS SET, LOOPING WILL BE PERFORMED ON THE SECTION RELOCATION (SEE SECTION 9.5.1). IF SW<9> IS NOT SET, EXECUTION WILL CONTINUE AT THE BEGINNING OF THE NEXT SECTION.

2. MEMORY MANAGEMENT ON- IF SW<9> IS SET, LOOPING WILL BE PERFORMED ON THE PROGRAM RELOCATION (SEE SECTION 9.5.2) TO THE SAME MEMORY SPACE THAT FAILED. IF SW<9> IS NOT SET, PROGRAM RELOCATION WILL BE RETRIED IN THE SAME MEMORY SPACE.

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

THIS ERROR IS THE SAME AS 6.1.4 EXCEPT RELOCATION WAS PERFORMED VIA A DISK RATHER THAN THE CP. THE ERROR PRINTOUT WILL IDENTIFY WHICH DEVICE AND DRIVE NUMBER TRANSFERRED THE PARTICULAR WORD THAT FAILED. REFER TO SECTION 9.5.4 FOR A DESCRIPTION OF I/O RELOCATION.

LOOP ON ERROR (SW<9>) HAS THE FOLLOWING EFFECT:

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

6.1.6 DEVICE ERROR

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

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

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

6.1.7 UNIBUS EXERCISER FAILED

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

```
XXXXXX      = CYCLE COUNT.
VVVVVV      = VIRTUAL BUS ADDRESS THAT THE UBE FAILED AT
WWWWWW      = CONTROL REGISTER NUMBER 2
YYYYYY      = CONTROL REGISTER NUMBER 1
ZZZZZZZZ    = PHYSICAL MEMORY ADDRESS THAT THE UBE FAILED AT
```

THE PHYSICAL MEMORY ADDRESS IS CALCULATED BY ADDING THE APPROPRIATE MAP REGISTER TO THE VIRTUAL BUS ADDRESS, FORMING A REAL 22 BIT MEMORY ADDRESS.

6.1.8 UBE NON-EXISTANT MEMORY ERROR

THIS ERROR ONLY OCCURS WHEN THE "NO SLAVE SYNC" ERROR OCCURS IN THE UNIBUS EXERCISER. ONLY THE PHYSICAL ADDRESS THAT TIMED OUT IS TYPED. THIS ERROR MIGHT INDICATE THAT THERE IS A HOLE IN MEMORY OR THAT THE SIZE REGISTER (777760) IS SET WRONG.

6.1.9 MASS BUS TESTER FAILED

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

  ER        CS3
HHHHHH     JJJJJJ
```

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

6.1.10 MBT NON-EXISTANT MEMORY ERROR

THIS IS THE SAME AS 6.1.7 EXCEPT THAT IT IS DETECTED BY THE NEXM BIT IN CS2 OF THE MBT.

6.1.11 FLOATING POINT ERROR

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

THIS ERROR SHOULD ONLY BE A FUNCTION OF THE FLOATING POINT PROCESSOR AND THE FPP DIAGNOSTICS (DEFPA DEFPB) SHOULD BE USED

```
   557                                  TO ISOLATE THE PROBLEM.
```

6.1.12 DEVICE HUNG

THIS ERROR WILL OCCUR IF A DEVICE DOES NOT FINISH ITS RELOCATION FUNCTION WITHIN 2 SECONDS AFTER ITS INITIATION. IF A LINE CLOCK IS NOT INSTALLED, A HUNG DEVICE WILL HANG THE PROGRAM. REFER TO SECTION 9.5.4.4 TO DETERMINE WHICH DEVICE AND DRIVE IS HUNG.

6.2 ERROR RECOVERY

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

6.2.1 ERRORS WITHIN SUBTESTS

EXECUTION STARTS WITH THE INSTRUCTION FOLLOWING THE ERROR CALL.

6.2.2 RELOCATION WITH MEMORY MGMT. OFF

EXECUTION STARTS AT THE BEGINNING OF THE NEXT SECTION.

6.2.3 DEVICE ERROR OR CP RELOCATION WITH MEMORY MGMT. ON

RELOCATION IS RESTARTED.

6.2.4 UNEXPECTED TRAPS EXCEPT PARITY (4,10,250)

EXECUTION STARTS AT THE ADDRESS POINTED TO BY LOCATION "$LPERR". THIS LOCATION CONTAINS THE ADDRESS+2 OF THE MOST RECENTLY EXECUTED "SCOPE" INSTRUCTION.

6.2.5 UNEXPECTED PARITY ERROR

IF THE PARITY ERROR IS FATAL (BIT 2 OR 3 SET IN ERROR REG) THE PROGRAM TYPES A RESTART MESSAGE AT RESTARTS. OTHERWISE, EXECUTION STARTS AS IN 6.2.4.

7.0 WARNINGS AND EXCEPTIONS

7.1 WARNINGS

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

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

```
614
615                                7.2     EXCEPTIONS
```

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

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

IF THE SYSTEM HAS BOTH AN RP03 AND AN RP04, THE BRANCH INSTRUCTION AT 100$, IN THE "SIZE ROUTINE" MUST BE REPLACED BY A NOP (240) FOR BOTH DEVICES TO BE USED. THIS BRANCH IS APPROXIMATELY AT ADDRESS 4552.

8.0 MISCELLANEOUS

8.1 EXECUTION TIME

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

1. MANUFACTURING BASIC LINE-32K MEMORY,UBE, MBT, AND NO DISKS---3 MINUTES.
2. SYSTEM-128K MEMORY, 2 RK05'S, RP04, AND 2 RS04'S ---9 MINUTES.

8.2 STACK POINTER

THE STACK POINTER IS SET TO 700.

NOTE

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

8.3 PASS COUNT

THERE ARE TWO WORDS USED FOR EFFECTIVE PASS COUNT. LOCATION "SUBPASS" AND "$PASS". SUBPASS CONTAINS THE ASCII REPRESENTATION OF THE SUBPASS COUNT. THIS IS USED TO INDEX

672 THE PSW TABLE AND MARGIN TABLE (SEE SECTION 8.7).

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

AT THE END OF EACH SUBPASS THE SUBPASS NUMBER (THAT IS BEING STARTED) IS TYPED FOLLOWED BY "THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789". IF RUNNING ON ACT11 QV OR AA, ONLY THE SUB-PASS NUMBER IS TYPED. AT THE END OF EACH PASS THE ELAPSED RUN TIME AND THE MESSAGE "END PASS X TOTAL ERRORS SINCE LAST REPORT Y" IS TYPED.

8.4 ITERATIONS

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

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

8.5 T-BIT TRAPPING

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

8.6 ACT-11 COMPATABILITY

THE PROGRAM IS FULLY ACT-11 COMPATABLE.

8.7 PSW AND MARGIN TABLES

AT THE END OF THE PROGRAM, JUST BEFORE THE MESSAGES, ARE THE PSW AND MARGIN TABLES. THESE TABLES CONTROL WHAT MODE AND REGISTER SET AND WHICH MEMORY MARGIN WILL BE EXECUTED ON A SUBPASS. REFER TO SECTION 9.5.2 FOR A DESCRIPTION OF HOW THESE TABLES ARE USED BY THE PROGRAM. THESE TABLES MAY BE MODIFIED IF DESIRED.

8.8 I/O DEVICE ADDRESS MODIFICATION

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

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

8.9 POWER FAIL

```
   729                                          ----------
```

IF A POWER FAIL OCCURS (FOLLOWED BY A POWER UP), THE WORD "POWER" IS TYPED ON THE TERMINAL AND THE PROGRAM RESTARTS.

9.0 PROGRAM DESCRIPTION

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

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

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

FOLLOWING IS A BRIEF DESCRIPTION OF EACH SECTION:

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

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

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

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

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

SECTION 4 — THIS SECTION CHECKS THAT EACH BIT IN THE PROCESSOR STATUS WORD (PSW) CAN BE SET CLEARED, RESERVED INSTRUCTIONS, AND ODD ADDRESS TRAPS.

```
                        SECTION 5   THIS SECTION CHECKS THE SXT, XOR, SOB,  MARK,  RTT
                                    AND RTT INSTRUCTIONS.
```

SECTION 6 THIS SECTION CHECKS THE ASH, ASHC, MUL, DIV, SPL INSTRUCTIONS AND THE PROGRAM INTERRUPT REQUEST (PIRQ) LOGIC.

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

SECTION 8 THIS SECTION CHECKS THE FLOATING POINT OPTION, (FP11-B OR FP11-C) IF AVAILABLE.

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

9.1 MICRO-BREAK TEST

THE MICRO-BREAK TEST IS USED TO TEST THE MICRO BREAK COMPARATORS AND THE STOP ON MICRO MATCH FUNCTION OF THE MAINTENANCE CARD. TO RUN THIS TEST THE OPERATOR MUST HAVE A MAINTENANCE CARD INSTALLED AND START THE PROGRAM AT ADDRESS 214.

THE PROGRAM ASKS THE OPERATOR TO TURN ON THE STOP ON MICRO MATCH SWITCH. IT THEN CHECKS CERTAIN BIT PATTERNS IN THE MICRO BREAK REGISTER TO ENSURE THE PROCESSOR DOES NOT STOP WHEN IT IS NOT SUPPOSED TO.

THE PROCESSOR WILL THEN STOP WITH ZERO IN THE MICRO ADDRESS LIGHTS. THE OPERATOR THEN HITS CONTINUE, AND THE PROCESSOR WILL STOP WITH ONE (1) IN THE LIGHTS. THIS SEQUENCE CONTINUES WITH 2, 4, 10, 20, 40, AND 200 APPEARING IN THE LIGHTS. THE PROGRAM TYPES DONE WHEN IT IS FINISHED.

9.2 UNIBUS EXERCISER(UBE)

ANY ONE OF 4 UBE'S WILL BE USED. THE PROGRAM LOOKS FOR A UBE AT ADDRESSES 17770004, 17770024, 17770034, AND 17770044.

TEST 75 WILL INITIATE THE UNIBUS EXERCISER IF IT IS PRESENT. THIS IS ONLY DONE ON PASS 1 - SUBPASS 1, SINCE FROM THAT POINT ON, THE SERVICE ROUTINE TAKES CARE OF RESTARTING IT.

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

WHEN AN INTERRUPT OCCURS A CHECK IS MADE TO SEE IF IT WAS CAUSED BY AN ERROR. IF THERE WAS NO ERROR, 776 IS LOADED AS

```
844                     THE BUS ADDRESS AND THE UBE IS STARTED  AGAIN.   ON  THE  NEXT
845                     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 ADDRESS IN THE SYSTEM SIZE REGISTERS. IF THEY ARE THE SAME (NO HOLES IN MEMORY) THE UBE IS RESTARTED AT ADDRESS 0 AND THE ABOVE SEQUENCE IS REPEATED. IF THE ADDRESSES ARE NOT THE SAME A MEMORY-HOLE ERROR IS REPORTED.

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

9.3 MASS BUS TESTER(MBT)

ANY ONE OF 4 MBT'S WILL BE USED. THE PROGRAM LOOKS FOR AN MBT AT ADDRESSES 17770100, 17770200, 17770300, AND 17770400. IF AN MBT IS FOUND, THE DRIVE TYPE REGISTER (17770X26) IS CHECKED TO MAKE SURE THAT IT REALLY IS AN MBT.

TEST 75 ALSO INITIATES THE MASS BUS TESTER. AGAIN, THIS IS ONLY DONE ON PASS 1 - SUBPASS 1 SINCE THE SERVICE ROUTINE KEEPS IT RUNNING.

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

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

9.4 LINE CLOCK INITIALIZATION

TEST 75 TURNS ON THE LINE CLOCK. TWO LOCATIONS IN "COMMON TAGS" KEEP TRACK OF THE ELAPSED RUN TIME OF THE PROGRAM. WHEN THE CLOCK INTERRUPTS, THE LOW BYTE OF LOCATION "LTICKS"IS INCREMENTED. WHEN THIS BYTE GETS TO 60(DECIMAL) IT IS CLEARED AND THE HIGH BYTE IS INCREMENTED(SECONDS). WHEN THE SECOND COUNT GETS TO 60(DECIMAL) LOCATION "MTICKS" IS INCREMENTED AND LTICKS IS CLEARED. THIS GIVES THE TIMER A 64K DECIMAL MINUTE RANGE.

NOTE

FOR THE UBE, MBT, AND LINE CLOCK, WHEN AN INTERRUPT OCCURS, PROGRAM EXECUTION RETURNS TO KERNEL MODE AND THE KERNEL PAR'S ARE MAPPED DOWN TO THE 0-12K BANK OF MEMORY. UPON RETURNING FROM THE INTERRUPT THE PAR'S ARE MAPPED BACK TO

```
902                                     WHERE  THEY  WERE   AND   THE   PREVIOUS
903                                     PROCESSOR MODE IS RESTORED.
```

9.5 RELOCATION ALGORITHM

9.5.1 SECTION RELOCATION

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

AT THE END OF EACH SECTION, CONTROL IS TRANSFERED TO THE "RELOCATION ROUTINE". IF SW<8> IS SET, THIS ROUTINE WILL RELOCATE THE SECTION VIA THE CP (SEE 9.5.3). IF SW<8> IS NOT SET, THE LENGTH OF THE SECTION IS CALCULATED, SAVED AS A WORD COUNT, AND CONTROL IS TRANSFERED TO THE "I/O MONITOR" (SEE SECTION 9.5.4) WHICH RELOCATES THE SECTION BY USING A DISK.

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

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

9.5.2 PROGRAM RELOCATION

WHEN ALL NINE SECTIONS HAVE BEEN RELOCATED AND EXECUTED THRU 28K (SEE SECTION 9.5.1), MEMORY MANAGEMENT IS SETUP ACCORDING TO THE VALUE IN LOCATION "NEXPAR". THIS VALUE IS INITIALIZED TO 600 (OR 1600 IF RUNNING UNDER THE XXDP MONITOR), MAKING RELOCATION START AT ADDRESS 60000 (OR 160000). THE "I/O MONITOR" IS THEN ENTERED (SEE SECTION 9.5.4) TO RELOCATE THE PROGRAM. WHEN THE I/O MONITOR COMPLETES THE RELOCATION, EXECUTION IS TRANSFERED TO THE START OF THE PROGRAM AT THE RELOCATED POSITION.

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

THIS CONTINUES UNTIL THE END OF MEMORY IS REACHED AND CONSTITUTES A SUB-PASS. THE PSW AND MAINTENANCE REGISTER (FOR MEMORY MARGINS) ARE THEN SETUP FOR THE NEXT SUB-PASS AND THE PROGRAM RESTARTS.

```
960
961                                     THE VALUE FOR THE PSW AND MAINTENANCE REGISTERS IS TAKEN  FROM
```

THE TABLES (SEE SECTION 8.7). THE PARTICULAR ENTRY THAT IS USED IS OBTAINED BY INDEXING THE TABLE BY THE SUB-PASS NUMBER (SEE SECTION 8.3). FOR EXAMPLE, SUB-PASS 3 USES WORD 3 (THE FIRST WORD IS COUNTED AS ZERO) OF EACH TABLE. THEREFORE, TO CHANGE THE VALUE IN THE PSW OR MAINTENANCE REGISTER ONLY REQUIRES CHANGING THE VALUE IN THE APPROPRIATE TABLE.

THE COMPLETION OF 6 SUB-PASSES CONSTITUTES A PASS AND AN END OF PASS MESSAGE IS TYPED. THE PROGRAM THEN RESTARTS IN PASS 2, SUB-PASS 0.

9.5.3 RELOCATION VIA CP

IF SW<8> IS SET, BOTH SECTION AND PROGRAM RELOCATION (SEE SECTIONS 9.5.1 AND 9.5.2), ARE PERFORMED BY AN INSTRUCTION MOVE LOOP RATHER THAN A DISK. FOR EXAMPLE:

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

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

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

9.5.4 RELOCATION VIA I/O

IF SW<8> IS NOT SET, BOTH SECTION AND PROGRAM RELOCATION (SEE SECTION 9.5.1 AND 9.5.2), ARE PERFORMED BY WRITING THE DATA TO A DISK AND READING IT BACK TO THE RELOCATED POSITION. THIS RELOCATION IS CONTROLLED BY THE "I/O MONITOR".

9.5.4.1 SECTION RELOCATION

WHEN THE I/O MONITOR IS ENTERED FROM THE "RELOCATION ROUTINE" (SEE SECTION 9.5.1) A DEVICE IS SELECTED (SEE 9.5.4.3), THE MEMORY ADDRESSES (FROM AND TO) AND WORD COUNT ARE PASSED TO THE DEVICE HANDLER (SEE SECTION 9.5.4.4), AND THE HANDLER IS CALLED. WHEN THE HANDLER FINISHES, THE I/O MONITOR CHECKS THE RELOCATED DATA WITH AN INSTRUCTION COMPARE LOOP TO ENSURE THE RELOCATED DATA IS CORRECT, AND RETURNS TO THE "RELOCATION ROUTINE" (SEE 9.5.1).

9.5.4.2 PROGRAM RELOCATION

WHEN THE I/O MONITOR IS ENTERED FOR PROGRAM RELOCATION (SEE SECTION 9.5.2) THE BASE ADDRESS FOR THE RELOCATION IS CALCULATED FROM THE CONTENTS OF KERNEL PAR3 WHICH WAS SET UP WITH MEMORY MANAGEMENT (SEE 9.5.2). IF SW<8> IS SET,

1018 RELOCATION IS PERFORMED VIA THE CP (SEE SECTION 9.5.3).

IF SW<8> IS NOT SET, A DEVICE IS SELECTED (SEE 9.5.4.3), THE WORD COUNT IS SET TO 2K, AND THE MEMORY ADDRESSES (FROM AND TO) AND WORD COUNT ARE PASSED TO THE DEVICE HANDLER (SEE 9.5.4.4), AND THE HANDLER IS CALLED. THE I/O MONITOR THEN ADDS 2K TO THE MEMORY ADDRESSES, SELECTS ANOTHER DEVICE, PASSES THE ADDRESSES TO THE DEVICE HANDLER, AND CALLS THE HANDLER. THIS CONTINUES UNTIL ALL 12K HAS BEEN RELOCATED. THE RELOCATED DATA IS THEN CHECKED WITH AN INSTRUCTION COMPARE LOOP. THE RELOCATED PROGRAM IS THEN EXECUTED AS DESCRIBED IN 9.5.2.

9.5.4.3 DEVICE SELECTION

IF SW<5> IS NOT SET, AN INDEX IS PICKED UP FROM LOCATION "DEVINDX". THIS INDEX IS USED TO INDEX THE SYSTEM SIZE TABLE. THE SYSTEM SIZE TABLE CONSISTS OF 8 WORDS (ONE FOR EACH DEVICE TYPE). BITS <7:0> OF EACH WORD ARE USED TO INDICATE THE DRIVE NUMBERS THAT ARE AVAILABLE ON THE DEVICE, AND ARE INITIALIZED IN THE SIZE ROUTINE. BITS <15:8> OF EACH WORD ARE USED TO INDICATE WHETHER THE DRIVE HAS BEEN USED FOR A DATA TRANSFER (UNIT USED BIT).

THE SYSTEM SIZE TABLE IS THEN SEARCHED, USING THE INDEX DESCRIBED ABOVE, FOR A DRIVE THAT HAS NOT BEEN USED. WHEN A DRIVE IS FOUND, THE "UNIT USED BIT" IS SET, THE CURRENT INDEX IS PUT BACK IN LOCATION DEVINDX, AND EXECUTION CONTINUES AS DESCRIBED IN 9.5.4.1 OR 9.5.4.2.

IF AN UNUSED UNIT IS NOT FOUND, ALL THE "UNIT USED" BITS ARE CLEARED AND THE SEARCH IS RESTARTED. IF THE SEARCH FINDS THE SYSTEM SIZE TABLE EMPTY (NO DEVICES ON THE SYSTEM), THE MESSAGE "NO I/O DEVICES" IS TYPED AND RELOCATION IS PERFORMED VIA THE CP AS DESCRIBED IN 9.5.3.

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

9.5.4.4 DEVICE HANDLERS

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

THE HANDLER IS CALLED BY THE I/O MONITOR (SEE SECTION 9.5.4). IT FIRST CLEARS THE DONE BIT (BIT 7) IN THE HANDLER STATUS WORD. THIS PREVENTS THE MONITOR FROM CALLING THIS HANDLER AGAIN BEFORE IT IS FINISHED.

IF A "DEVICE HUNG" ERROR (SEE SECTION 6.1.12) IS DETECTED, THE HANDLER STATUS WORDS CAN BE EXAMINED TO DETERMINE WHICH DEVICE DID NOT FINISH (SET BIT 7). THE DRIVE CAN THEN BE DETERMINED

```
1075                    BY  LOOKING  IN  THE  "DEVICE HANDLER UNIT NUMBER" TABLE.  THE
1076                    HANDLER STATUS WORDS AND DEVICE HANDLER  UNIT  NUMBER  TABLES,
```

ARE LOCATED IN THE "COMMON TAGS" AREA OF THE LISTING.

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

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

THE TRACK-SECTOR ADDRESS OF THE TRANSFER IS SAVED IN THE "RUN TRACK TABLE" (RUNTRAK). THE POSITION IN THIS TABLE IS AS DESCRIBED ABOVE. THE BIT CONFIGURATION OF EACH WORD IS THE SAME AS THAT FOR THE DISK ADDRESS REGISTER FOR THE PARTICULAR DEVICE. BIT 15 IS USED TO INDICATE A DEVICE ERROR. IT IS SET BY THE DEVICE SERVICE ROUTINE.(SEE SECTION 9.5.4.5)

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

9.5.4.5 DEVICE SERVICE ROUTINES

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

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

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

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

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

```
1133                                    %
1134
1135
1136
1137
1138
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1149
1150
1151
1152
1153
1154
1155                                    .TITLE  MAINDEC-11-DEQKC-B PDP 11/70 CPU EXERCISOR
1156                                    ;*COPYRIGHT (C) SEPTEMBER 21, 1975
1157                                    ;*DIGITAL EQUIPMENT CORP.
1158                                    ;*MAYNARD, MASS. 01754
1159                                    ;*
1160                                    ;*PROGRAM BY DONALD W. MONROE
1161                                    ;*
1162                                    ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
1163                                    ;*PACKAGE (MAINDEC-11-DZQAC-A5).
1164                                    ;*
```

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

```
1204
1205                                    .SBTTL  BASIC DEFINITIONS
1206
1207                                    ;*INITIAL ADDRESS OF THE STACK POINTER *** 1200 ***
1208    001200                          STACK=  1200            ;;FIRST ADDRESS OF THE STACK
1209    001200                          KERSTK= STACK           ;;KERNEL STACK
1210    000700                          SUPSTK= STACK-300       ;;SUPERVISOR STACK
1211    000600                          USESTK= STACK-400       ;;USER STACK
1212                                    .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
1213                                    .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL
1214    177776                          PS=     177776          ;;PROCESSOR STATUS WORD
1215                                    .EQUIV  PS,PSW
1216    177774                          STKLMT= 177774          ;;STACK LIMIT REGISTER
1217    177772                          PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
1218    177570                          SWR=    177570          ;;SWITCH REGISTER
1219    177570                          DISPLAY=SWR
1220
1221                                    ;*MISCELLANEOUS DEFINITIONS
1222    000011                          HT=     11              ;;CODE FOR HORIZONTAL TAB
1223    000012                          LF=     12              ;;CODE LINE FEED
1224    000015                          CR=     15              ;;CODE CARRIAGE RETURN
1225    000200                          CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
1226
1227                                    ;*GENERAL PURPOSE REGISTER DEFINITIONS
1228    000000                          R0=     %0              ;;GENERAL REGISTER
1229    000001                          R1=     %1              ;;GENERAL REGISTER
1230    000002                          R2=     %2              ;;GENERAL REGISTER
1231    000003                          R3=     %3              ;;GENERAL REGISTER
1232    000004                          R4=     %4              ;;GENERAL REGISTER
1233    000005                          R5=     %5              ;;GENERAL REGISTER
1234    000006                          R6=     %6              ;;GENERAL REGISTER
1235    000007                          R7=     %7              ;;GENERAL REGISTER
1236                                    .EQUIV  R0,R10          ;;GENERAL REGISTER
1237                                    .EQUIV  R1,R11          ;;GENERAL REGISTER
1238                                    .EQUIV  R2,R12          ;;GENERAL REGISTER
1239                                    .EQUIV  R3,R13          ;;GENERAL REGISTER
1240                                    .EQUIV  R4,R14          ;;GENERAL REGISTER
1241                                    .EQUIV  R5,R15          ;;GENERAL REGISTER
1242                                    .EQUIV  R6,SP           ;;STACK POINTER
1243                                    .EQUIV  SP,KSP          ;;KERNEL STACK POINTER
1244                                    .EQUIV  SP,SSP          ;;SUPERVISOR STACK POINTER
1245                                    .EQUIV  SP,USP          ;;USER STACK POINTER
1246                                    .EQUIV  R7,PC           ;;PROGRAM COUNTER
1247
1248                                    ;*PRIORITY LEVEL DEFINITIONS
1249    000000                          PR0=    0               ;;PRIORITY LEVEL 0
1250    000040                          PR1=    40              ;;PRIORITY LEVEL 1
1251    000100                          PR2=    100             ;;PRIORITY LEVEL 2
1252    000140                          PR3=    140             ;;PRIORITY LEVEL 3
1253    000200                          PR4=    200             ;;PRIORITY LEVEL 4
1254    000240                          PR5=    240             ;;PRIORITY LEVEL 5
1255    000300                          PR6=    300             ;;PRIORITY LEVEL 6
1256    000340                          PR7=    340             ;;PRIORITY LEVEL 7
1257
1258                                    ;*"SWITCH REGISTER" SWITCH DEFINITIONS
1259    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
```

```
1316    000010          RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
1317    000014          TBITVEC=14              ;;"T" BIT
1318    000014          TRTVEC= 14              ;;TRACE TRAP
1319    000014          BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
1320    000020          IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
1321    000024          PWRVEC= 24              ;;POWER FAIL
1322    000030          EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
1323    000034          TRAPVEC=34              ;;"TRAP" TRAP
1324    000060          TKVEC=  60              ;;TTY KEYBOARD VECTOR
1325    000064          TPVEC=  64              ;;TTY PRINTER VECTOR
1326    000114          CACHVEC=114             ;;CACHE ERROR INTERRUPT VECTOR
1327    000240          PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
1328    000250          MMVEC=  250             ;;MEMORY MANAGEMENT VECTOR
1329
1330                    .SBTTL  CACHE   REGISTER DEFINITIONS
1331
1332
1333    177740          LOADRS = 177740         ;;LOWER 16 BITS OF ADDRESS THAT CAUSED ERROR
1334    177742          HIADRS = 177742         ;;UPPER SIX BITS OF ADDRESS THAT CAUSED ERROR
1335    177744          MEMERR = 177744         ;;CACHE ERROR REGISTER
1336    177746          CONTRL = 177746         ;;MEMORY CONTROL REGISTER
1337    177750          MAINT  = 177750         ;;MEMORY MAINTENENCE REGISTER
1338    177752          HITMIS = 177752         ;;HIT MISS REGISTER "1" IMPLIES HIT IN CACHE
1339
1340
1341                    .SBTTL  CPU REGISTER DEFINITIONS
1342
1343
1344    177760          SIZELO = 177760         ;;MEMORY SIZE REGISTER NUMBER TO PUT INTO A PAR
1345                                            ;;TO GET TO THE LAST 32 WORDS OF MEMORY
1346    177762          SIZEHI = 177762         ;;HIGH SIZE REGISTER, RESERVED FOR FUTURE USE
1347                                            ;;CURRENTLY ALL ZERO
1348    177764          SYSTID = 177764         ;;SYSTEM ID REGISTER
1349    177766          CPUERR = 177766         ;;CPU ERROR REGISTER HOLDS CONDITION THAT CAUSED
1350                                            ;;THE TRAP TO ERRVEC (000004)
1351
1352
1353
1354
1355                    .SBTTL  MEMORY MANAGEMENT DEFINITIONS
1356
1357
1358                    ;*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES
1359
1360    177572          MMR0=   177572
1361    177574          MMR1=   177574
1362    177576          MMR2=   177576
1363    172516          MMR3=   172516
1364                    .EQUIV  MMR0,SR0
1365                    .EQUIV  MMR1,SR1
1366                    .EQUIV  MMR2,SR2
1367                    .EQUIV  MMR3,SR3
1368
1369                    ;*USER "I" PAGE DESCRIPTOR REGISTERS
1370
1371    177600          UIPDR0= 177600
```

```
1372            177602                  UIPDR1= 177602
1373            177604                  UIPDR2= 177604
1374            177606                  UIPDR3= 177606
1375            177610                  UIPDR4= 177610
1376            177612                  UIPDR5= 177612
1377            177614                  UIPDR6= 177614
1378            177616                  UIPDR7= 177616
1379
1380                                    ;*USER "D" PAGE DESCRIPTOR REGISTORS
1381
1382            177620                  UDPDR0= 177620
1383            177622                  UDPDR1= 177622
1384            177624                  UDPDR2= 177624
1385            177626                  UDPDR3= 177626
1386            177630                  UDPDR4= 177630
1387            177632                  UDPDR5= 177632
1388            177634                  UDPDR6= 177634
1389            177636                  UDPDR7= 177636
1390
1391                                    ;*USER "I" PAGE ADDRESS REGISTERS
1392
1393            177640                  UIPAR0= 177640
1394            177642                  UIPAR1= 177642
1395            177644                  UIPAR2= 177644
1396            177646                  UIPAR3= 177646
1397            177650                  UIPAR4= 177650
1398            177652                  UIPAR5= 177652
1399            177654                  UIPAR6= 177654
1400            177656                  UIPAR7= 177656
1401
1402                                    ;*USER "D" PAGE ADDRESS REGISTERS
1403
1404            177660                  UDPAR0= 177660
1405            177662                  UDPAR1= 177662
1406            177664                  UDPAR2= 177664
1407            177666                  UDPAR3= 177666
1408            177670                  UDPAR4= 177670
1409            177672                  UDPAR5= 177672
1410            177674                  UDPAR6= 177674
1411            177676                  UDPAR7= 177676
1412
1413                                    ;*SUPERVISOR "I" PAGE DESCRIPTOR REGISTERS
1414
1415            172200                  SIPDR0= 172200
1416            172202                  SIPDR1= 172202
1417            172204                  SIPDR2= 172204
1418            172206                  SIPDR3= 172206
1419            172210                  SIPDR4= 172210
1420            172212                  SIPDR5= 172212
1421            172214                  SIPDR6= 172214
1422            172216                  SIPDR7= 172216
1423
1424                                    ;*SUPERVISOR "D" PAGE DESCRIPTOR REGISTERS
1425
1426            172220                  SDPDR0= 172220
1427            172222                  SDPDR1= 172222
```

```
1428            172224                  SDPDR2= 172224
1429            172226                  SDPDR3= 172226
1430            172230                  SDPDR4= 172230
1431            172232                  SDPDR5= 172232
1432            172234                  SDPDR6= 172234
1433            172236                  SDPDR7= 172236
1434
1435                                    ;*SUPERVISOR "I" PAGE ADDRESS REGISTERS
1436
1437            172240                  SIPAR0= 172240
1438            172242                  SIPAR1= 172242
1439            172244                  SIPAR2= 172244
1440            172246                  SIPAR3= 172246
1441            172250                  SIPAR4= 172250
1442            172252                  SIPAR5= 172252
1443            172254                  SIPAR6= 172254
1444            172256                  SIPAR7= 172256
1445
1446                                    ;*SUPERVISOR "D" PAGE ADDRESS REGISTERS
1447
1448            172260                  SDPAR0= 172260
1449            172262                  SDPAR1= 172262
1450            172264                  SDPAR2= 172264
1451            172266                  SDPAR3= 172266
1452            172270                  SDPAR4= 172270
1453            172272                  SDPAR5= 172272
1454            172274                  SDPAR6= 172274
1455            172276                  SDPAR7= 172276
1456
1457                                    ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS
1458
1459            172300                  KIPDR0= 172300
1460            172302                  KIPDR1= 172302
1461            172304                  KIPDR2= 172304
1462            172306                  KIPDR3= 172306
1463            172310                  KIPDR4= 172310
1464            172312                  KIPDR5= 172312
1465            172314                  KIPDR6= 172314
1466            172316                  KIPDR7= 172316
1467
1468                                    ;*KERNEL "D" PAGE DESCRIPTOR REGISTERS
1469
1470            172320                  KDPDR0= 172320
1471            172322                  KDPDR1= 172322
1472            172324                  KDPDR2= 172324
1473            172326                  KDPDR3= 172326
1474            172330                  KDPDR4= 172330
1475            172332                  KDPDR5= 172332
1476            172334                  KDPDR6= 172334
1477            172336                  KDPDR7= 172336
1478
1479                                    ;*KERNEL "I" PAGE ADDRESS REGISTERS
1480
1481            172340                  KIPAR0= 172340
1482            172342                  KIPAR1= 172342
1483            172344                  KIPAR2= 172344
```

```
                        KIPAR3= 172346
                        KIPAR4= 172350
                        KIPAR5= 172352
                        KIPAR6= 172354
                        KIPAR7= 172356

                        ;*KERNEL "D" PAGE ADDRESS REGISTERS

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



                        .SBTTL  UNIBUS MAP REGISTER DEFINITIONS

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

                        MAPL00 = 170200
                        MAPH00 = 170202
                        MAPL01 = 170204
                        MAPH01 = 170206
                        MAPL02 = 170210
                        MAPH02 = 170212
                        MAPL03 = 170214
                        MAPH03 = 170216
                        MAPL04 = 170220
                        MAPH04 = 170222
                        MAPL05 = 170224
                        MAPH05 = 170226
                        MAPL06 = 170230
                        MAPH06 = 170232
                        MAPL07 = 170234
                        MAPH07 = 170236
                        MAPL10 = 170240
                        MAPH10 = 170242
                        MAPL11 = 170244
                        MAPH11 = 170246
                        MAPL12 = 170250
                        MAPH12 = 170252
                        MAPL13 = 170254
                        MAPH13 = 170256
                        MAPL14 = 170260
                        MAPH14 = 170262
                        MAPL15 = 170264
                        MAPH15 = 170266
                        MAPL16 = 170270
```

```
1540            170272                  MAPH16 = 170272
1541            170274                  MAPL17 = 170274
1542            170276                  MAPH17 = 170276
1543            170300                  MAPL20 = 170300
1544            170302                  MAPH20 = 170302
1545            170304                  MAPL21 = 170304
1546            170306                  MAPH21 = 170306
1547            170310                  MAPL22 = 170310
1548            170312                  MAPH22 = 170312
1549            170314                  MAPL23 = 170314
1550            170316                  MAPH23 = 170316
1551            170320                  MAPL24 = 170320
1552            170320                  MAPH24 = 170320
1553            170324                  MAPL25 = 170324
1554            170326                  MAPH25 = 170326
1555            170330                  MAPL26 = 170330
1556            170332                  MAPH26 = 170332
1557            170334                  MAPL27 = 170334
1558            170336                  MAPH27 = 170336
1559            170340                  MAPL30 = 170340
1560            170342                  MAPH30 = 170342
1561            170344                  MAPL31 = 170344
1562            170346                  MAPH31 = 170346
1563            170350                  MAPL32 = 170350
1564            170352                  MAPH32 = 170352
1565            170354                  MAPL33 = 170354
1566            170356                  MAPH33 = 170356
1567            170360                  MAPL34 = 170360
1568            170362                  MAPH34 = 170362
1569            170364                  MAPL35 = 170364
1570            170366                  MAPH35 = 170366
1571            170370                  MAPL36 = 170370
1572            170372                  MAPH36 = 170372
1573            170374                  MAPL37 = 170374
1574            170376                  MAPH37 = 170376
1575                                    .EQUIV  MAPL00,MAPL0
1576                                    .EQUIV  MAPH00,MAPH0
1577                                    .EQUIV  MAPL01,MAPL1
1578                                    .EQUIV  MAPH01,MAPH1
1579                                    .EQUIV  MAPL02,MAPL2
1580                                    .EQUIV  MAPH02,MAPH2
1581                                    .EQUIV  MAPL03,MAPL3
1582                                    .EQUIV  MAPH03,MAPH3
1583                                    .EQUIV  MAPL04,MAPL4
1584                                    .EQUIV  MAPH04,MAPH4
1585                                    .EQUIV  MAPL05,MAPL5
1586                                    .EQUIV  MAPH05,MAPH5
1587                                    .EQUIV  MAPL06,MAPL6
1588                                    .EQUIV  MAPH06,MAPH6
1589                                    .EQUIV  MAPL07,MAPL7
1590                                    .EQUIV  MAPH07,MAPH7
1591
1592
1593
1594
1595            000000                  AC0=    %0
```

```
1596    000001                          AC1=    %1
1597    000002                          AC2=    %2
1598    000003                          AC3=    %3
1599    000004                          AC4=    %4
1600    000005                          AC5=    %5
1601                            ;LINE CLOCK AND PROGRAMMABLE LINE CLOCK REGISTERS
1602    172540                          PLKCSR=172540
1603    172542                          PLKCSB=172542
1604    000104                          PLKVEC=104
1605    177546                          LKS=177546
1606    000100                          LKVEC=100
1607
1608                            ;UNIBUS EXERCISOR REGISTER
1609    170000                          UBEDB=  170000          ;DATA BUFFER
1610    170002                          UBECC=  170002          ;CYCLE COUNT
1611    170004                          UBEBA=  170004          ;BUS ADDRESS
1612    170006                          UBECR1= 170006          ;CRONTROL REGISTER 1
1613    170010                          UBECLR= 170010          ;ERROR CLEAR
1614    170014                          UBEGO=  170014          ;MULTI-EXERCISOR GO
1615    170016                          UBECR2= 170016          ;CONTROL REGISTER 2
1616    000510                          UBEVEC= 510             ;INTERRUPT VECTOR
1617
1618                            ;MASS BUS TESTER REGISTERS
1619    160100                          MBTCS1= 160100
1620    160102                          MBTWC=  160102
1621    160104                          MBTBA=  160104
1622    160106                          MBTMR2= 160106
1623    160110                          MBTCS2= 160110
1624    160112                          MBTST=  160112
1625    160114                          MBTER=  160114
1626    160116                          MBTAS=  160116
1627    160120                          MBTDB=  160120
1628    160124                          MBTMR1= 160124
1629    160126                          MBTDT=  160126
1630    160174                          MBTBAE= 160174
1631    160176                          MBTCS3= 160176
1632    000774                          MBTVEC= 774
1633    000776                          MBTPSW= 776
1634
1635                            ;MISCELLANEOUS BIT ASSIGNMENTS (USED IN OPT.CP)
1636    100000                          KTOPT=  100000          ;BELOW BIT ASSIGNMENTS ARE USED
1637    040000                          EISOPT= 040000          ;IN THE CPCHK ROUTINE
1638    020000                          FPOPT=  020000          ;A BIT FOR EACH OPTION PRESENT
1639    002000                          MBTOPT= 002000
1640    001000                          LKOPT=  001000
1641    000400                          TTOPT=  000400
1642    000200                          UBEOPT= 000200
1643                                    .EQUIV  ERROR,HLT
1644                                    .EQUIV  BIT14,SM
1645                                    .EQUIV  BIT12,PSM
1646                                    .EQUIV  BIT11,REG
1647    000010                          CALLHANDLER=10
1648    000000                          KM=0
1649    140000                          UM=140000
1650    000000                          PKM=0
1651    030000                          PUM=30000
```

```
                177770                          UBREAK=177770

                                        .SBTTL  TRAP CATCHER

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

                                        .SBTTL  STARTING ADDRESS(ES)
                000200                  .=200

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

                                        .SBTTL          ACT11 HOOKS

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

                000220                  $SVPC=.                 ;;SAVE LOCATION COUNTER
                000046                  .=46                    ;;SET LOCATION COUNTER
000046  036750                  .WORD   $ENDAD                  ;;SET LOC.46 TO ADDRESS $ENDAD
                000052                  .=52                    ;;SET LOCATION COUNTER
000052  040000                  .WORD   40000                   ;;SET LOC.52 TO 40000
                000220                  .=$SVPC                 ;; RESTORE LOCATION COUNTER
```

```
                                ;;**********************************************************************
                                .SBTTL  COMMON TAGS

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

                                        .=1200

001200                          $CMTAG:                         ;;START OF COMMON TAGS
001200  000000                  $PASS:  .WORD   0                ;;CONTAINS PASS COUNT
001202  000000                  $TSTNM: .WORD   0                ;;CONTAINS THE TEST NUMBER
001204  000                     $ERFLG: .BYTE   0                ;;CONTAINS ERROR FLAG
001206                                  .EVEN
001206  000000                  $ICNT:  .WORD   0                ;;CONTAINS SUBTEST ITERATION COUNT
001210  000000                  $LPADR: .WORD   0                ;;CONTAINS SCOPE LOOP 1200
001212  000000                  $LPERR: .WORD   0                ;;CONTAINS SCOPE RETURN FOR ERRORS
001214  000000                  $ERTTL: .WORD   0                ;;CONTAINS TOTAL ERRORS DETECTED
001216  000                     $ITEMB: .BYTE   0                ;;CONTAINS ITEM CONTROL BYTE
001217  001                     $ERMAX: .BYTE   1                ;;CONTAINS MAX. ERRORS PER TEST
001220  000000                  $ERRPC: .WORD   0                ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001222  000000                  $GDADR: .WORD   0                ;;CONTAINS 1200 OF 'GOOD' DATA
001224  000000                  $BDADR: .WORD   0                ;;CONTAINS 1200 OF 'BAD' DATA
001226  000000                  $GDDAT: .WORD   0                ;;CONTAINS 'GOOD' DATA
001230  000000                  $BDDAT: .WORD   0                ;;CONTAINS 'BAD' DATA
001232  000000  000000  000000          .WORD   0,0,0            ;;RESERVED--NOT TO BE USED
001240  177560                  $TKS:   177560                   ;;TTY KBD STATUS
001242  177562                  $TKB:   177562                   ;;TTY KBD BUFFER
001244  177564                  $TPS:   177564                   ;;TTY PRINTER STATUS REG. 1200
001246  177566                  $TPB:   177566                   ;;TTY PRINTER BUFFER REG. 1200
001250  000                     $NULL:  .BYTE   0                ;;CONTAINS NULL CHARACTER FOR FILLS
001251  002                     $FILLS: .BYTE   2                ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001252  012                     $FILLC: .BYTE   12               ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001253  000                     $TPFLG: .BYTE   0                ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001254  000000                  $REGAD: .WORD   0                ;;CONTAINS THE 1200 FROM
                                                                 ;;WHICH  ($REG0) WAS OBTAINED
001256  000000                  $REG0:  .WORD   0                ;;CONTAINS (($REGAD)+0)
001260  000000                  $REG1:  .WORD   0                ;;CONTAINS (($REGAD)+2)
001262  000000                  $REG2:  .WORD   0                ;;CONTAINS (($REGAD)+4)
001264  000000                  $REG3:  .WORD   0                ;;CONTAINS (($REGAD)+6)
001266  000000                  $REG4:  .WORD   0                ;;CONTAINS (($REGAD)+10)
001270  000000                  $REG5:  .WORD   0                ;;CONTAINS (($REGAD)+12)
001272  000000                  $REG6:  .WORD   0                ;;CONTAINS (($REGAD)+14)
001274  000000                  $REG7:  .WORD   0                ;;CONTAINS (($REGAD)+16)
001276  000000                  $TMP0:  .WORD   0                ;;USER DEFINED
001300  000000                  $TMP1:  .WORD   0                ;;USER DEFINED
001302  000000                  $TMP2:  .WORD   0                ;;USER DEFINED
001304  000000                  $TMP3:  .WORD   0                ;;USER DEFINED
001306  000000                  $TMP4:  .WORD   0                ;;USER DEFINED
001310  000000                  $TMP5:  .WORD   0                ;;USER DEFINED
001312  000000                  $TMP6:  .WORD   0                ;;USER DEFINED
001314  000000                  $TMP7:  .WORD   0                ;;USER DEFINED
001316  000000                  $TIMES: 0                        ;;MAX. NUMBER OF ITERATIONS
001320  000000                  $ESCAPE:0                        ;;ESCAPE ON ERROR 1200
001322  177607  000377           $BELL:  .ASCIZ  <207><377><377>  ;;CODE FOR BELL
001326  077                     $QUES:  .ASCII  /?/              ;;QUESTION MARK
```

```
 1751  001327  015                    $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
 1752  001330  000012                   $LF:    .ASCIZ  <12>            ;;LINE FEED
 1753  001332  000000                   ERRRTN: .WORD
 1754  001334  000044                   $FLBUFF:.BLKB   44      ;BUFFER FOR FLOATING POINT CONVERSION
 1755  001400  000000                   $BUFF:  .WORD           ;
 1756  001402  000000                   $AC0:   .WORD           ;EXTENDED EXPONENT VALUES
 1757  001404  000000                   $AC1:   .WORD           ;FOR THE SIX FLOATING POINT
 1758  001406  000000                   $AC2:   .WORD           ;ACCUMULATORS
 1759  001410  000000                   $AC3:   .WORD
 1760  001412  000000                   $AC4:   .WORD
 1761  001414  000000                   $AC5:   .WORD
 1762  001416  000000                   $$TMP4: .WORD
 1763  001420  000000                   $$TMP6: .WORD
 1764  001422  000004                   FLTMP0: .BLKW   4       ;FLOATING POINT DBL PREC BUFFER
 1765  001432  000004                   FLTMP1: .BLKW   4
 1766  001442  001444                   TKBFRP: .WORD   TKBFR   ;POINTER FOR KEYBOARD BUFFER
 1767  001444  000011                   TKBFR:  .BLKW   11      ;KEYBOARD BUFFER
 1768  001466  000000                   NOTYPE: .WORD           ;NO TYPEOUT FLAG (INHIBIT WHEN SET)
 1769  001470  000000                   OPT.CP: .WORD           ;CPU OPTION FLAGS
 1770  001472  000006                   $RTRN:  RTT             ;RETURN FOR T-BIT TRAP
 1771  001474  000000                   VADR:   .WORD           ;BUFFER FOR VIRTUAL ADDRESS
 1772  001476  000000                   PA1500: .WORD           ;BUFFER FOR PHYSICAL ADDRESS BITS<15:00>
 1773  001500  000000                   PA2116: .WORD           ;PHYSICAL ADDRESS BITS<21:16>
 1774  001502  000                      NEXEC:  .BYTE           ;NO EXECUTE FLAG(NO TEST EXECUTION WHEN SET)
 1775  001503  000                      MMON:   .BYTE           ;MEMORY MGMT FLAG(MGMT IS ON WHEN NON-ZERO)
 1776  001504  000                      QV:     .BYTE           ;QV FLAG(QV PASS WHEN SET)
 1777  001505  000                      AA:     .BYTE           ;AUTO ACCEPT FLAG (AA PASS WHEN SET)
 1778  001506  000000                   FACTOR: .WORD           ;RELOCATION FACTOR(NUMBER OF
 1779  001510  000000                   $FACTOR:.WORD           ;BYTES ABOVE BASE CODE)
 1780  001512  000000                   FRSTAD: .WORD           ;FIRST ADDRESS OF SECTION BEING EXECUTED
 1781  001514  000000                   FRSTMEM:.WORD           ;ADDRESS OF FIRST FREE MEMORY
 1782  001516  000000                   LSTMEM: .WORD           ;ADDRESS OF LAST FREEE MEMORY(IN 28K)
 1783  001520  000000                   NEXPAR: .WORD           ;NEXT VALUE TO PUT IN PARD
 1784  001522  123456                   $LONUM: .WORD   123456  ;LOW 16 BITS OF RANDOM NUMBER
 1785  001524  065432                   $HINUM: .WORD   65432   ;HIGH 16 BITS OF RANDOM NUMBER
 1786  001526  001     001     001      NULLS:  .BYTE   1,1,1,0 ;BUFFER FOR PRINTER TEST
 1787  001531  000
 1788  001532  000060                   SUBPASS:.WORD   60      ;SUB-PASS COUNT IN ASCII
 1789  001534  000000                   $ERPSW: .WORD           ;ERROR PSW FOR TYPEOUT
 1790  001536  000000                   EXITFL: .WORD
 1791  001540  000000                   OLDBASE:.WORD           ;SOURCE BASE ADDRESS FOR DEVICE RELOCATION
 1792  001542  000000                   NWBASL: .WORD           ;DEST ADDRESS FOR DEVICE RELOC BITS<15:00>
 1793  001544  000000                   NWBASH: .WORD           ;DEST ADDRESS FOR DEVICE RELOC BITS<21:16>
 1794  001546  000000                   IOWC:   .WORD           ;TWO'S COMPLIMENT WORD COUNT FOR DEVICE RELOC
 1795  001550  000000                   DEVICE: .WORD
 1796  001552  000000                   DEVINDX:.WORD           ;DEVICE INDEX (0 TO 7)
 1797  001554  000000                   UNITNO: .WORD           ;DEVICE UNIT NUMBER
 1798  001556  000000                   RNTBINX:.WORD           ;INDEX TO RUN TABLE
 1799  001560  000000                   MXMMHI: .WORD           ;BITS<21:16> OF LAST MEM ADDRESS ON SYSTEM
 1800  001562  000000                   MXMMLO: .WORD           ;BITS<15:00> OF LAST MEM ADDRESS ON SYSTEM
 1801  001564  000000                   RP310:  .WORD           ;DATA TO LOAD INTO RP03 CS REGISTER
 1802  001566  000000                   RP311:  .WORD           ;RP03 FLAG FOR FIRST 2K OF PROGRAM
 1803  001570  000000                   RK10:   .WORD           ;DATA TO LOAD INTO RK05 CS REGISTER
 1804  001572  000000                   RK11:   .WORD           ;RK05 FLAG FOR FIRST 2K OF PROGRAM
 1805  001574  000000                   RP411:  .WORD           ;RP04 FLAG FOR FIRST 2K OF PROGRAM
 1806  001576  000000                   RS11:   .WORD           ;RS04 FLAG FOR FIRST 2K OF PROGRAM
```

```
1807  001600  000000                  MTICKS: .WORD           ;ELAPSED RUN TIME IN MINUTES
1808  001602  000000                  LTICKS: .WORD           ;LOW BYTE=NUMBER OF CLOCK INTERRUPTS (0 TO 59)
1809                                                          ;HIGH BYTE=ELAPSED RUN TIME IN SECONDS(0 TO 59)
1810  001604  000000                  $MAINT: .WORD           ;CURRENT VALUE IN MAINTENANCE REGISTER
1811  001606  000010                  SYSSIZE:.BLKW   10      ;SYSTEM SIZE TABLE(ONE ENTRY FOR EACH DEVICE)
1812  001626  000006                  RUNTBL: .BLKW   6       ;RUN TIME TABLE(ONE ENTRY FOR EACH 2K BLOCK)
1813  001642  000006                  RUNTRAK:.BLKW   6       ;RUN TRACK TABLE(ONE ENTRY FOR EACH 2K BLOCK)
1814  001656  177777                  MAPTBL: .WORD   -1      ;MAP TABLE(ONE BYTE FOR EACH UNIBUS DEVICE)
1815  001660  177777                          .WORD   -1      ;UNUSED=377, USED=LOW 5 BITS OF MAP ADDRESS
1816  001662  000002                  UBESAV: .BLKW   2       ;BASE ADDRESS OF UBE TRANSFER IN PROGRESS
1817  001666  000002                  UBEADR: .BLKW   2       ;ADDRESS THAT GETS LOADED INTO UBE BA REG
1818  001672  000002                  ERRBA:  .BLKW   2       ;18 BIT UNIBUS ADDRESS WHEN DEVICE DETECTED AN ERROR
1819                                  .SBTTL  DEVICE HANDLER STATUS WORDS
1820                                  ;*      EACH WORD HAS THE FOLLOWING BIT ASSIGNMENTS:
1821                                  ;*              7       HANDLER READY
1822                                  ;*              8       REPEAT LAST FUNCTION
1823                                  ;*              15      ERROR
1824  001676  000200                  RP3HSTAT:.WORD  200     ;RP03
1825  001700  000200                  RKHSTAT:.WORD   200     ;RK05
1826  001702  000200                  SPARE0: .WORD   200
1827  001704  000200                  SPARE1: .WORD   200
1828  001706  000200                  RP4HSTAT:.WORD  200     ;RP04
1829  001710  000200                  RSHSTAT:.WORD   200     ;RS04
1830  001712  000200                          .WORD   200     ;SPARE
1831  001714  000200                          .WORD   200     ;SPARE
1832
1833                                  .SBTTL  DEVICE HANDLER WORD COUNTS
1834                                  ;*      THIS TABLE GETS LOADED BY THE I/O
1835                                  ;*      RELOCATION ROUTINE WITH THE TWO'S COMPLIMENT WORD
1836                                  ;*      COUNT FOR THE TRANSFER FOR THE PARTICULAR DEVICE.
1837  001716  000000                  RP3HWC: .WORD           ;RP03
1838  001720  000000                  RKHWC:  .WORD           ;RK05
1839  001722  000000                          .WORD           ;SPARE
1840  001724  000000                          .WORD           ;SPARE
1841  001726  000000                  RP4HWC: .WORD           ;RP04
1842  001730  000000                  RSHWC:  .WORD           ;RS04
1843
1844                                  .SBTTL  DEVICE HANDLER OLD BASE ADDRESS
1845                                  ;*      THIS TABLE GETS LOADED BY THE I/O RELOCATION ROUTINE
1846                                  ;*      WITH THE BASE ADDRESS OF THE SOURCE DATA FOR THE
1847                                  ;*      DEVICE THAT IS GOING TO TRANSFER THE DATA.
1848  001732  000000                  RP3OLD: .WORD           ;RP03
1849  001734  000000                          .WORD
1850  001736  000000                  RKOLD:  .WORD           ;RK05
1851  001740  000000                          .WORD
1852  001742  000000                          .WORD           ;SPARE
1853  001744  000000                          .WORD           ;SPARE
1854  001746  000000                          .WORD
1855  001750  000000                          .WORD           ;SPARE
1856  001752  000000                  RP4OLD: .WORD           ;RP04
1857  001754  000000                          .WORD
1858  001756  000000                  RSOLD:  .WORD           ;RS04
1859  001760  000000                          .WORD
1860
1861                                  .SBTTL  DEVICE HANDLER NEW BASE ADDRESSES
1862                                  ;*      THIS TABLE GETS LOADED BY THE I/O RELOCATION ROUTINE
```

```
1863                                    ;*      WITH THE BASE ADDRESS OF THE DESTINATION FOR THE
1864                                    ;*      PARTICULAR DEVICE THAT IS GOING TO DO THE TRANSFER.
1865  001762  000000                    RP3NWL: .WORD           ;RP03
1866  001764  000000                    RP3NWH: .WORD
1867  001766  000000                    RKNEWL: .WORD           ;RK05
1868  001770  000000                    RKNEWH: .WORD
1869  001772  000000                            .WORD           ;SPARE
1870  001774  000000                            .WORD
1871  001776  000000                            .WORD           ;SPARE
1872  002000  000000                            .WORD
1873  002002  000000                    RP4NWL: .WORD           ;RP04
1874  002004  000000                    RP4NWH: .WORD
1875  002006  000000                    RSNEWL: .WORD           ;RS04
1876  002010  000000                    RSNEWH: .WORD
1877
1878                                    .SBTTL  DEVICE HANDLER UNIT NUMBER
1879                                    ;*      THIS TABLE GETS LOADED BY THE I/O RELOCATION ROUTINE.
1880                                    ;*      IT TELLS THE DEVICE HANDLER WHICH UNIT NUMBER IS
1881                                    ;*      TO DO THE TRANSFER.
1882  002012  000000                    RP3UNIT:.WORD           ;RP03
1883  002014  000000                    RKUNIT: .WORD           ;RK05
1884  002016  000000                            .WORD           ;SPARE
1885  002020  000000                            .WORD           ;SPARE
1886  002022  000000                    RP4UNIT:.WORD           ;RP04
1887  002024  000000                    RSUNIT: .WORD           ;RS04
1888
1889                                    .SBTTL  ADDRESS OF THE DEVICE HANDLERS
1890                                    ;*      THIS TABLE CONTAINS THE ADDRESS OF THE DEVICE HANDLER
1891                                    ;*      ROUTINES. IT IS USED BY THE I/O RELOCATION ROUTINE
1892                                    ;*      TO TRANSFER CONTROL TO THE DEVICE HANDLER.
1893  002026  036770                    RP3HANA:.WORD   RP3DRV  ;RP03
1894  002030  037406                    RKHANA: .WORD   RKDRV   ;RK05
1895  002032  000000                            .WORD           ;SPARE
1896  002034  000000                            .WORD           ;SPARE
1897  002036  040002                    RP4HANA:.WORD   RP4DRV  ;RP04
1898  002040  040352                    RSHANA: .WORD   RSDRV   ;RS04
1899
1900                                    .SBTTL  DEVICE HANDLER DISK ADDRESS TABLE
1901                                    ;*      THIS TABLE GETS LOADED BY THE DEVICE HANDLER WITH THE
1902                                    ;*      DISK ADDRESS(SECTOR AND CYLINDER) OF THE CURRENT
1903                                    ;*      TRANSFER.
1904  002042  000000                    RP3HDA: .WORD           ;RP03 DISK ADDRESS
1905  002044  000000                    RP3HDC: .WORD           ;RP03 DESIRED CYLINDER
1906  002046  000000                    RKHDA:  .WORD           ;RK05 DISK ADDRESS
1907  002050  000000                            .WORD           ;SPARE
1908  002052  000000                    RP4HDA: .WORD
1909  002054  000000                    RP4HDC: .WORD           ;RP04 DESIRED CYLINDER
1910  002056  000000                    RSHDA:  .WORD           ;RS04 DISK ADDRESS
1911
1912                                    .SBTTL  DEVICE HANDLER FUNCTION TABLE
1913                                    ;*      THIS TABLE GETS LOADED BY THE DEVICE HANDLERS
1914                                    ;*      AND THE DEVICE SERVICE ROUTINES. IT TELLS THE ROUTINES
1915                                    ;*      WHICH FUNCTION TO DO NEXT.
1916  002060  000000                    RP3FUN: .WORD           ;RP03
1917  002062  000000                    RKFUN:  .WORD           ;RK05
1918  002064  000000                            .WORD           ;SPARE
```

```
                                RP4FUN: .WORD           ;RP04
                                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.
                                RP3TRY: .BYTE           ;RP03
                                RKTRY:  .BYTE           ;RK05
                                        .BYTE           ;SPARE
                                RP4TRY: .BYTE           ;RP04
                                RSTRY:  .BYTE           ;RS04
                                        .EVEN

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

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

                                .SBTTL  RH70/RP04 REGISTERS
002142  176700                  RP4CS1: .WORD   176700  ;CONTROL AND STATUS #1
002144  176702                  RP4WC:  .WORD   176702  ;WORD COUNT
002146  176704                  RP4BA:  .WORD   176704  ;BUS ADDRESS
002150  176750                  RP4BAE: .WORD   176750  ;BUS ADDRESS EXTENDED
002152  176706                  RP4DA:  .WORD   176706  ;DISK ADDRESS
002154  176710                  RP4CS2: .WORD   176710  ;CONTROL AND STATUS #2
002156  176752                  RP4CS3: .WORD   176752  ;CONTROL AND STATUS #3
002160  176712                  RP4DS:  .WORD   176712  ;DRIVE STATUS
002162  176714                  RP4ER1: .WORD   176714  ;ERROR REG #1
002164  176734                  RP4DC:  .WORD   176734  ;DESIRED CYLINDER
002166  176740                  RP4ER2: .WORD   176740  ;ERROR REG #2
```

```
1975  002170  176742                  RP4ER3: .WORD   176742  ;ERROR REG #3
1976  002172  176736                  RPCC:   .WORD   176736  ;CURRENT CYLINDER
1977  002174  176732                  RP4OF:  .WORD   176732  ;OFFSET REGISTER
1978  002176  000254                  RP4VEC: .WORD   254     ;INTERRUPT VECTOR
1979  002200  000256                  RP4PSW: .WORD   256     ;INTERRUPT VECTOR+2
1980
1981                                  .SBTTL  RH70/RS04 REGISTERS
1982  002202  172040                  RSCS1:  .WORD   172040  ;CONTROL AND STATUS #1
1983  002204  172042                  RSWC:   .WORD   172042  ;WORD COUNT
1984  002206  172044                  RSBA:   .WORD   172044  ;BUS ADDRESS
1985  002210  172070                  RSBAE:  .WORD   172070  ;BUS ADDRESS EXTENDED
1986  002212  172046                  RSDA:   .WORD   172046  ;DISK ADDRESS
1987  002214  172050                  RSCS2:  .WORD   172050  ;CONTROL AND STATUS #2
1988  002216  172072                  RSCS3:  .WORD   172072  ;CONTROL AND STATUS #3
1989  002220  172052                  RSDS:   .WORD   172052  ;DRIVE STATUS
1990  002222  172054                  RSER:   .WORD   172054  ;ERROR REG
1991  002224  000204                  RSVEC:  .WORD   204     ;INTERRUPT VECTOR
1992  002226  000206                  RSPSW:  .WORD   206     ;INTERRUPT VECTOR+2
1993
1994                                  .SBTTL  UNIBUS EXERCISER REGISTER ADDRESS TABLE
1995                                  ;*      THIS TABLE IS ASSEMBLED FOR UBE #0. IF THE UBE
1996                                  ;*      ADDRESSES ARE CUT FOR OTHER THAN UNIT #0, THE PROGRAM
1997                                  ;*      WILL CHANGE THIS TABLE. THE PROGRAM LOOKS FOR A
1998                                  ;*      UBE AT ADDRESSES 770002, 770022, 770032, AND 770042.
1999  002230  170002                  UBETBL: .WORD   UBECC    ;CYCLE COUNT
2000  002232  170004                          .WORD   UBEBA    ;BUS ADDRESS REG
2001  002234  170016                          .WORD   UBECR2   ;CONTROL REGISTER #2
2002  002236  170006                          .WORD   UBECR1   ;CONTROL REGISTER #1
2003  002240  170010                          .WORD   UBECLR   ;UBE CLEAR ADDRESS
2004  002242  000510                          .WORD   UBEVEC   ;INTERRUPT VECTOR
2005  002244  000512                          .WORD   UBEVEC+2         ;INTERRUPT VECTOR +2
2006
2007                                  .SBTTL  MASS BUS TESTER REGISTER ADDRESSES
2008                                  ;*      THE PROGRAM IS ASSEMBLED WITH ADDRESSES FOR A MBT
2009                                  ;*      AT 770100. IF THE MBT IS AT ANOTHER ADDRESS THE PROGRAM
2010                                  ;*      WILL CHANGE THIS TABLE. THE PROGRAM LOOKS FOR A UBE
2011                                  ;*      AT ADDRESSES 770100, 770200, 770300, AND 770400.
2012  002246  160100                  MBTTBL: .WORD   MBTCS1   ;CONTROL AND STATUS #1
2013  002250  160102                          .WORD   MBTWC    ;WORD COUNT
2014  002252  160104                          .WORD   MBTBA    ;BUS ADDRESS
2015  002254  160174                          .WORD   MBTBAE   ;BUS ADDRESS EXTENDED
2016  002256  160106                          .WORD   MBTMR2   ;MAINTENANCE REGISTER #2
2017  002260  160110                          .WORD   MBTCS2   ;CONTROL REGISTER #2
2018  002262  160112                          .WORD   MBTST    ;STATUS REGISTER
2019  002264  160114                          .WORD   MBTER    ;ERROR REGISTER
2020  002266  160176                          .WORD   MBTCS3   ;CONTROL REGISTER #3
2021  002270  000774                          .WORD   MBTVEC   ;INTERRUPT VECTOR
2022  002272  000776                          .WORD   MBTPSW   ;INTERRUPT VECTOR+2
2023  002274  160126                          .WORD   MBTDT    ;DRIVE TYPE REGISTER
2024  002276  160200                  MBTN2:  .WORD   160200  ;MASS BUS TESTER #2
2025  002300  160300                  MBTN3:  .WORD   160300  ;MASS BUS TESTER #3
2026  002302  160400                  MBTN4:  .WORD   160400  ;MASS BUS TESTER #4
2027
```

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

                        .SBTTL  ERROR POINTER TABLE

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

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


002304                  $ERRTB:
                                ;ITEM 1
002304  054772                  EM1             ;UNEXPECTED TRAP TO 4
002306  055017                  DH1             ;PCOFTP  PHYSC      PSW  CPUERR
002310  055064                  DT1             ;VADR,VADR,$TMP0,$TMP2
002312  055056                  DF1             ;0,1,0,0,0
                                ;ITEM 2
002314  055076                  EM2             ;UNEXPECTED TRAP TO 10
002316  055124                  DH2             ;PCOFTP  PHYSPC  PSW
002320  055152                  DT2             ;VADR,VADR,$TMP0
002322  055056                  DF1
                                ;ITEM 3
002324  055162                  EM3             ;UNEXPECTED TRAP TO 250(MGMT)
002326  055217                  DH3             ;PCOFTP  PHYSPC    PSW    MMR0    MMR2
002330  055266                  DT3             ;VADR,VADR,$TMP0,,$TMP2,,$TMP3
002332  055056                  DF1
                                ;ITEM 4
002334  055302                  EM4             ;UNEXPECTED TRAP TO 114
002336  055331                  DH4             ;PCOFTP  PHYSPC   PSW    ERADREG MEMERRREG
002340  055266                  DT3             ;VADR,VADR,$TMP0,$TMP3,$TMP2
002342  055407                  DF4             ;0,1,0,2,0
                                ;ITEM 5
002344  055414                  EM5             ;PARITY ERROR DURING DATA CHECK
002346  055453                  DH5             ;SRCADR DSTADR ERRADREG  MEM ERR REG
002350  055524                  DT5             ;$TMP0,PA1500,$TMP3,$TMP2
002352  055520                  DF5
                                ;ITEM 6
002354  055536                  EM6             ;ERROR DURING CHECK OF RELOCATED DATA
002356  055606                  DH6             ;SRCADR  DSTADR
002360  055626                  DT6             ;$TMP0,PA1500
002362  055407                  DF4
                                ;ITEM 7
002364  000000                  0
002366  000000                  0
002370  000000                  0
002372  000000                  0
                                ;ITEM 10
002374  055634                  EM10            ;ERROR DURING DATA CHECK-RELOC WAS BY I/O
002376  055705                  DH10            ;SRCADR   DSTADR   DEVICE THAT DID XFER
002400  055760                  DT10            ;$TMP0,VADR,$TMP2,$TMP3
```

```
  2084  002402  055754                DF10            ;0,1,3,0
  2085                                ;ITEM 11
  2086  002404  055772                EM11            ;BIT(S) STUCK IN MICRO-BREAK REG
  2087  002406  056037                DH11            ;GOOD DAT  BAD DAT
  2088  002410  056062                DT11            ;$TMP0,$TMP1
  2089  002412  056060                DF11            ;0,0
  2090                                ;ITEM 12
  2091  002414  056070                EM12            ;UNIBUS EXERCISOR NON-EXISTANT MEMOREY
  2092  002416  056126                DH12            ;PHYSICAL ADDRESS
  2093  002420  056144                DT12            ;PA1500
  2094  002422  056142                DF12            ;2
  2095                                ;ITEM 13
  2096  002424  056150                EM13            ;MASS BUS TESTER NON-EXISTANT MEMORY
  2097  002426  056206                DH13            ;PHYSICAL ADDRESS
  2098  002430  056144                DT12
  2099  002432  056142                DF12
  2100                                ;ITEM 14
  2101  002434  056223                EM14            ;FLOATING POINT ERROR
  2102  002436  056250                DH14            ;       DATA1           DATA2
  2103  002440  056270                DT14            ;$TMP4,$REG2,$TMP6,$REG3
  2104  002442  056302                DF14            ;4,0,4,0
  2105                                ;ITEM 15
  2106  002444  056306                EM15            ;DEVICE HUNG
  2107  002446  000000                0
  2108  002450  000000                0
  2109  002452  000000                0
  2110                                ;ITEM 16
  2111  002454  056223                EM14            ;FLOATING POINT ERROR
  2112  002456  056322                DH16
  2113  002460  056354                DT16            ;FLTMP0,$REG2,FLTMP1,$REG3
  2114  002462  056347                DF16            ;5,0,5,0
```

```
2115  002464  013737  177570  177570  START1: MOV     @#SWR,@#SWR
2116  002472  000774                          BR      START1
2117                                          .SBTTL  PROGRAM INITIALIZATION
2118
2119                                  ;;*****************************************************************
2120                                  .SBTTL  MICRO-BREAK REGISTER TEST
2121                                  ;*THIS TEST IS EXECUTED BY STARTING THE PROGRAM AT ADDRESS 214.
2122                                  ;*THIS TEST REQUIRES A MAINTENANCE CARD AND OPERATOR INTERVENTION.
2123                                  ;*THE PROCESSOR SHOULD STOP 8 TIMES. FOLLOWING IS THE DATA
2124                                  ;*THAT SHOULD BE IN THE MICRO-ADRESS DATA LIGHTS EACH TIME:
2125                                  ;*      1       000
2126                                  ;*      2       001
2127                                  ;*      3       002
2128                                  ;*      4       004
2129                                  ;*      5       010
2130                                  ;*      6       020
2131                                  ;*      7       040
2132                                  ;*      8       200
2133                                  ;;*****************************************************************
2134  002474  012706  001100          START2: MOV     #1100,SP        ;SETUP THE SP
2135  002500  012737  051372  000034          MOV     #$TRAP,@#TRAPVEC        ;SETUP TRAP VECTOR
2136  002506  012737  044772  000030          MOV     #$ERROR,@#EMTVEC;SETUP EMT VECTOR
2137  002514  012700  000377                  MOV     #377,R0         ;PUT MICRO-BREAK DATA IN R0
2138  002520  010037  177770                  MOV     R0,@#UBREAK     ;LOAD U BREAK REG
2139  002524  020037  177770                  CMP     R0,@#UBREAK     ;LOAD OK?
2140  002530  001024                          BNE     UBRERR  ;BRANCH IF NO
2141  002532  005000                          CLR     R0
2142  002534  010037  177770                  MOV     R0,@#UBREAK
2143  002540  020037  177770                  CMP     R0,@#UBREAK
2144  002544  001016                          BNE     UBRERR
2145  002546  012700  000125                  MOV     #125,R0
2146  002552  010037  177770                  MOV     R0,@#UBREAK
2147  002556  020037  177770                  CMP     R0,@#UBREAK
2148  002562  001007                          BNE     UBRERR
2149  002564  012700  000252                  MOV     #252,R0
2150  002570  010037  177770                  MOV     R0,@#UBREAK
2151  002574  020037  177770                  CMP     R0,@#UBREAK
2152  002600  001411                          BEQ     UBRK2
2153  002602  010067  176470          UBRERR: MOV     R0,$TMP0
2154  002606  013737  177770  001300          MOV     @#UBREAK,@#$TMP1
2155  002614  012737  002474  001212          MOV     #START2,@#$LPERR
2156  002622  104011                          ERROR   11
2157                                  ;TEST TO ENSURE U BREAK COMPARATORS DO NOT COME ON.
2158  002624  012737  000100  177770  UBRK2:  MOV     #100,@#UBREAK           ;PUT SAFE VALUE IN REG
2159  002632  104400  002640                  TYPE    ,65$             ;;TYPE ASCIZ STRING
2160  002636  000421                          BR      64$              ;;GET OVER THE ASCIZ
2161                                  ;;65$:  .ASCIZ  /SET MAINT TO STOP ON MICRO-BREAK/<CRLF>
2162  002702                          64$:
2163  002702  104400  002710                  TYPE    ,67$             ;;TYPE ASCIZ STRING
2164  002706  000407                          BR      66$              ;;GET OVER THE ASCIZ
2165                                  ;;67$:  .ASCIZ  /HIT CONTINUE/<CRLF>
2166  002726                          66$:
2167  002726  000000                          HALT
2168  002730  012737  000012  000010          MOV     #12,@#RESVEC
2169  002736  012737  000002  000012          MOV     #2,@#RESVEC+2
2170  002744  012705  003004                  MOV     #2$,R5                  ;SET UP R5 FOR MARK INSTR
```

```
2171  002750  012701  000010              MOV     #10,R1          ;SET SOB COUNT
2172  002754  012702  003161              MOV     #UBRTBL+1,R2    ;GET ADRS OF UBREAK DATA TABLE
2173  002760  112237  177770      1$:     MOVB    (R2)+,@#UBREAK  ;LOAD MICRO-BREAK FROM TABLE
2174  002764  000010                      10                      ;EXEC RES INSTR (ROM ADRS 000)
2175  002766  005037  177770              CLR     @#UBREAK
2176  002772  077106                      SOB     R1,1$           ;CONTINUE
2177  002774  012737  000125  177770      MOV     #125,@#UBREAK   ;SET MICRO-BREAK DATA PATTERN
2178  003002  006400                      MARK    0               ;EXEC MARK (ROM ADRS 252)
2179  003004  005037  177770      2$:     CLR     @#UBREAK
2180  003010  012706  001100              MOV     #1100,SP        ;RESTORE SP
2181  003014  012737  000006  000004      MOV     #6,@#ERRVEC
2182  003022  012737  000002  000006      MOV     #2,@#ERRVEC+2
2183  003030  052737  040000  177776      BIS     #BIT14,@#PSW    ;GO TO SUPER MODE
2184  003036  012706  000700              MOV     #700,SP         ;SET SUPER SP
2185  003042  012746  003064              MOV     #3$,-(SP)       ;SETUP STACKK FOR JSR INSTR
2186  003046  005000                      CLR     R0              ;SETUP R0
2187  003050  012701  000007              MOV     #7,R1           ;SET SOB COUNT
2188  003054  012702  003174              MOV     #INSTBL+2,R2    ;GET ADRS OF TABLE OF INSTRUCTIONS
2189  003060  012217          4$:         MOV     (R2)+,(PC)      ;GET INSTRUCTION
2190  003062  000000                      .WORD                   ;EXECUTE INSTRUCTION
2191  003064  077103          3$:         SOB     R1,4$           ;CONTINUE
2192  003066  012737  000100  177770      MOV     #100,@#UBREAK   ;PUT SAFE VALUE IN UBREAK REG
2193  003074  005000                      CLR     R0
2194  003076  012702  003160              MOV     #UBRTBL,R2
2195  003102  012703  003172              MOV     #INSTBL,R3
2196  003106  012701  000010              MOV     #10,R1
2197  003112  012746  003126              MOV     #5$,-(SP)
2198  003116  112237  177770      6$:     MOVB    (R2)+,@#UBREAK  ;LOAD UBREAK REG FROM TABLE
2199  003122  012317                      MOV     (R3)+,(PC)      ;GET INSTR FROM TABLE
2200  003124  000000                      .WORD                   ;EXECUTE INSTR. PROCESSOR SHOULD STOP
2201                                                              ;WITH THE CORRECT ROM ADR IN THE LIGHTS
2202  003126  077105          5$:         SOB     R1,6$           ;CONTINUE
2203  003130  111237  177770              MOVB    (R2),@#UBREAK   ;PUT SAFE VALUE IN UBREAK REG
2204  003134  005037  177776              CLR     @#PSW           ;GO BACK TO KERNEL MODE
2205  003140  104400  003146              TYPE    ,69$            ;;TYPE ASCIZ STRING
2206  003144  000403                      BR      68$             ;;GET OVER THE ASCIZ
2207                          ;;69$:      .ASCIZ  /DONE/<CRLF>
2208  003154                  68$:
2209  003154  000000                      HALT
2210  003156  000415                      BR      START
2211  003160     000     001     002  UBRTBL: .BYTE   0,1,2,4,10,20,40,200,100
2212  003163     004     010     020
2213  003166     040     200     100
2214          003172                  .EVEN
2215  003172  000010  005010  005020  INSTBL: .WORD   10,5010,5020,5040,0,5200,207,5010
2216  003200  005040  000000  005200
2217  003206  000207  005010
```

```
2218  003212  012706  001200          START:  MOV     #KERSTK,SP      ;SET KERNEL STACK PTR
2219  003216  012737  076543  001524          MOV     #76543,@#$HINUM         ;INITIALIZE RANDOM NUM GEN
2220  003224  012737  123456  001522          MOV     #123456,@#$LONUM
2221
2222                                  ;DETERMINE HOW PROGRAM WAS LOADED AND WHAT MODE (IF ACT11)
2223                                  ;AND SET MEMORY PROTECTION.
2224  003232  005037  001504                  CLR     @#QV                    ;SET NOT QV NOR AA MODE
2225  003236  005027                          CLR     (PC)+                   ;SET NOT XXDP
2226  003240     000          XXDP:   .BYTE   0                       ;XXDP INDICATOR
2227  003241     000          XXDPC:  .BYTE   0                       ;XXDP CHAIN MODE INDICATOR
2228  003242  005027                          CLR     (PC)+                   ;CLEAR MEMORY PROTECTION LIMIT
2229  003244  000000          PROT:   .WORD   0                       ;WILL CONTAIN MEM PROT LIMIT
2230  003246  005737  036754                  TST     @#$ENDAD+4              ;BRANCH IF NOT QV
2231  003252  100003                          BPL     1$
2232  003254  110637  001504                  MOVB    SP,@#QV                 ;SET ACT11 QV MODE
2233  003260  000411                          BR      3$
2234
2235  003262  001003          1$:     BNE     2$
2236  003264  110637  001505                  MOVB    SP,@#AA                 ;SET ACT11 AA MODE
2237  003270  000405                          BR      3$
2238
2239  003272  005737  000042          2$:     TST     @#42                    ;BRANCH IF NOT IN CHAIN MODE
2240  003276  001402                          BEQ     3$
2241  003300  110637  003241                  MOVB    SP,@#XXDPC              ;SET CHAIN MODE INDICATOR
2242
2243                                  ;SET MEMORY PROTECTION LIMITS
2244  003304  005737  001504          3$:     TST     @#QV                    ;BRANCH IF QV OR AA
2245  003310  001006                          BNE     MEMSIZ
2246  003312  005737  003240                  TST     @#XXDP                  ;BRANCH IF NOT VIA XXDP
2247  003316  001403                          BEQ     MEMSIZ
2248  003320  012737  005700  003244          MOV     #5700,@#PROT            ;PROTECT XXDP MONITOR
2249  003326  012737  157776  001516  MEMSIZ: MOV     #157776,@#LSTMEM        ;SET VALUE INTO LSTMEM
2250  003334  163737  003244  001516          SUB     @#PROT,@#LSTMEM         ;SET PROTECTION
2251  003342  012737  056370  001514          MOV     #ENDTAG+2,@#FRSTMEM     ;SET FIRST RELOCATION ADDRESS
2252
2253                                  ;GET ADDRESS OF THE LAST MEMORY LOCATION ON THE SYSTEM
2254  003350  005002                          CLR     R2
2255  003352  013703  177760                  MOV     @#SIZELO,R3     ;GET ADDRESS OF SIZE REG LO
2256  003356  073227  000006                  ASHC    #6,R2           ;SHIFT TO FORM ADDRESS
2257  003362  052703  000077                  BIS     #77,R3          ;ENSURE LOWER SIX BITS SET
2258  003366  062703  000001                  ADD     #1,R3
2259  003372  005502                          ADC     R2
2260  003374  010237  001560                  MOV     R2,@#MXMMHI     ;SAVE UPPER SIX BITS
2261  003400  010337  001562                  MOV     R3,@#MXMMLO     ;SAVE LOWER 16 BITS
2262
2263  003404  012706  001200                  MOV     #KERSTK,SP              ;SET STACK PTR
2264  003410  005037  001200                  CLR     @#$PASS                 ;CLEAR PASS COUNT
2265  003414  105037  001503                  CLRB    @#MMON                  ;SET MEM MGMT ON IND=NOT ON
2266  003420  012737  000600  001520          MOV     #600,@#NEXPAR           ;SET FIRST 'PAR' VALUE
2267  003426  005737  003244                  TST     @#PROT
2268  003432  001403                          BEQ     1$
2269  003434  012737  001600  001520          MOV     #1600,@#NEXPAR
2270  003442                          1$:
2271  003442  012700  000027                  MOV     #27,R0                  ;SET SOB COUNT
2272  003446  005001                          CLR     R1                      ;SETUP INDEX
2273  003450  005061  001600          2$:     CLR     MTICKS(R1)              ;CLEAR TABLES
```

```
2274  003454  062701  000002                  ADD     #2,R1
2275  003460  077005                          SOB     R0,2$                   ;CONTINUE
2276  003462  012737  177777  001656          MOV     #-1,@#MAPTBL            ;INITIALIZE MAP TABLE
2277  003470  012737  177777  001660          MOV     #-1,@#MAPTBL+2
2278  003476  012700  000010                  MOV     #10,R0                  ;SET SOB COUNT
2279  003502  012701  001676                  MOV     #RP3HSTAT,R1            ;GET ADDRESS OF HANDLER STAT
2280  003506  012721  000200          3$:     MOV     #200,(R1)+              ;INITIALIZE STATUS TABLE
2281  003512  077003                          SOB     R0,3$                   ;CONTINUE
2282  003514  012737  000060  001532          MOV     #60,@#SUBPASS           ;INIT SUBPASS TO ASCII 0
2283  003522  012700  047104                  MOV     #TIMEBUF,R0             ;GET ADR OF TIME BUFFER
2284  003526  012701  000012                  MOV     #12,R1                  ;SET SOB COUNT
2285  003532  112720  000060          4$:     MOVB    #60,(R0)+               ;INIT TIME BUFFER
2286  003536  077103                          SOB     R1,4$
2287  003540  105040                          CLRB    -(R0)                   ;INSERT TERMINATOR
2288  003542  112737  000072  047107          MOVB    #72,@#TIMEBUF+3         ;INSERT COLON
2289  003550  112737  000072  047112          MOVB    #72,@#TIMEBUF+6
2290  003556  012737  000340  177776          MOV     #340,@#PS               ;;LOCK OUT ALL INTERRUPTS
2291  003564  012706  001200                  MOV     #$CMTAG,R6              ;;FIRST LOCATION TO BE CLEARED
2292  003570  005026                          CLR     (R6)+                   ;;CLEAR MEMORY LOCATION
2293  003572  022706  001240                  CMP     #$TKS,R6                ;;DONE?
2294  003576  001374                          BNE     .-6                     ;;LOOP BACK IF NO
2295  003600  012706  001200                  MOV     #STACK,SP               ;;SETUP THE STACK POINTER
2296  003604  012737  044532  000020          MOV     #$SCOPE,@#IOTVEC        ;;IOT VECTOR FOR SCOPE ROUTINE
2297  003612  012737  000340  000022          MOV     #340,@#IOTVEC+2         ;;LEVEL 7
2298  003620  012737  044772  000030          MOV     #$ERROR,@#EMTVEC        ;;EMT VECTOR FOR ERROR ROUTINE
2299  003626  012737  000340  000032          MOV     #340,@#EMTVEC+2         ;;LEVEL 7
2300  003634  012737  051372  000034          MOV     #$TRAP,@#TRAPVEC        ;;TRAP VECTOR FOR TRAP CALLS
2301  003642  012737  000340  000036          MOV     #340,@#TRAPVEC+2        ;;LEVEL 7
2302  003650  012737  051226  000024          MOV     #$PWRDN,@#PWRVEC        ;;POWER FAILURE VECTOR
2303  003656  012737  000340  000026          MOV     #340,@#PWRVEC+2         ;;LEVEL 7
2304  003664  016767  032724  032714          MOV     $ENDCT,$EOPCT           ;;SETUP END-OF-PROGRAM COUNTER
2305  003672  005067  175420                  CLR     $TIMES                  ;;INITIALIZE NUMBER OF ITERATIONS
2306  003676  005067  175416                  CLR     $ESCAPE                 ;;CLEAR THE ESCAPE ON ERROR ADDRESS
2307  003702  112767  000001  175307          MOVB    #1,$ERMAX               ;;ALLOW ONE ERROR PER TEST
2308  003710  012767  003710  175272          MOV     #.,$LPADR               ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
2309  003716  012767  003716  175266          MOV     #.,$LPERR               ;;SETUP THE ERROR LOOP ADDRESS
2310
2311                                  ;CLEAR PROGRAM INDICATORS
2312  003724  052777  000100  175306          BIS     #100,@$TKS              ;SET IE BIT IN KEYBOARD STATUS REG
2313  003732  012737  052376  000060          MOV     #TKISR,@#TKVEC          ;SETUP KEYBOARD VECTOR
2314  003740  012737  000200  000062          MOV     #PR4,@#TKVEC+2
2315  003746  012737  052610  000064          MOV     #TPISR,@#TPVEC
2316  003754  012737  000200  000066          MOV     #PR4,@#TPVEC+2
2317  003762  005037  001466                  CLR     @#NOTYPE                ;CLEAR 'NO TYPING' INDICATOR
2318
2319                                  ;THE BELOW ROUTINE ASCERTAINS WHICH CP & CP OPTIONS THE PROGRAM IS RUN-
2320                                  ;NING ON AND SETS AN INDICATOR IN OPT.CP ACCORDINGLY.
2321  003766  012737  000006  000004  CPCHK:  MOV     #ERRVEC+2,@#ERRVEC      ;SET UP ERROR TRAP TO RETURN
2322  003774  012737  000002  000006          MOV     #2,@#ERRVEC+2
2323  004002  012737  000012  000010          MOV     #RESVEC+2,@#RESVEC      ;AND ALSO RESERVED INST TRAP
2324  004010  012737  000002  000012          MOV     #2,@#RESVEC+2
2325  004016  012702  144006                  MOV     #144006,R2              ;SET 11/70 NON-OPTION BITS
2326  004022  000261                          SEC
2327  004024  170500                          TSTF    R0                      ;WILL CLEAR CARRY IF 11/45 FLOATING POINT
2328  004026  170000                          CFCC                            ;IS AVAIL. COPY FLOATING CC'S INTO PSW
2329  004030  103402                          BCS     6$                      ;BRANCH IF NO FLOATING POINT
```

```
2330  004032  052702  020000            BIS     #FPOPT,R2               ;SET FP OPTION AVAIL INDICATOR
2331  004036  000261            6$:     SEC
2332  004040  005737  177546            TST     @#LKS                   ;BRANCH IF NO KW11-L
2333  004044  103402                    BCS     7$
2334  004046  052702  001000            BIS     #LKOPT,R2               ;SET OPTION INDICATOR
2335  004052  000261            7$:     SEC
2336  004054  005777  175164            TST     @$TPS                   ;BRANCH IF NO CONSOLE TTY
2337  004060  103402                    BCS     9$
2338  004062  052702  000400            BIS     #TTOPT,R2
2339  004066  005003            9$:     CLR     R3
2340  004070  000261                    SEC
2341  004072  005737  170000            TST     @#UBEDB         ;IS UBE1 THERE?
2342  004076  103410                    BCS     12$             ;BRANCH IF NO
2343  004100  105037  170006            CLRB    @#UBECR1        ;IS THIS A TESTER OR EXERCISOR?
2344  004104  105737  170006            TSTB    @#UBECR1
2345  004110  100045                    BPL     15$             ;BRANCH IF TESTER
2346  004112  052702  000200    16$:    BIS     #UBEOPT,R2      ;SET INDICATOR
2347  004116  000425                    BR      17$
2348  004120  000261            12$:    SEC
2349  004122  005737  170020            TST     @#UBEDB+20              ;IS UBE2 THERE?
2350  004126  103403                    BCS     13$             ;BRANCH IF NO
2351  004130  012703  000020            MOV     #20,R3          ;SET OFFSET IN R3
2352  004134  000766                    BR      16$
2353  004136  000261            13$:    SEC
2354  004140  005737  170040            TST     @#UBEDB+40      ;IS UBE3 THERE?
2355  004144  103403                    BCS     14$             ;BRANCH IF NO
2356  004146  012703  000040            MOV     #40,R3          ;PUT OFFSET IN R3
2357  004152  000757                    BR      16$
2358  004154  000261            14$:    SEC
2359  004156  005737  170060            TST     @#UBEDB+60      ;IS UBE4 THERE?
2360  004162  103420                    BCS     15$             ;BRANCH IF NO
2361  004164  012703  000060            MOV     #60,R3          ;PUT OFFSET IN R3
2362  004170  000750                    BR      16$
2363  004172  005227  177777    17$:    INC     #-1
2364  004176  001012                    BNE     15$
2365  004200  012704  002230            MOV     #UBETBL,R4      ;GET ADDRESS OF UBE TABLE
2366  004204  012705  000005            MOV     #5,R5           ;SET SOB COUNT
2367  004210  060324            18$:    ADD     R3,(R4)+        ;ADJUST UBE TABLE ENTRIES
2368  004212  077502                    SOB     R5,18$          ;CONTINUE
2369  004214  006003                    ROR     R3
2370  004216  006003                    ROR     R3              ;ADJUST OFFSET FOR UBE VECTOR
2371  004220  060324                    ADD     R3,(R4)+        ;ADJUST UBEVEC ENTRY
2372  004222  060314                    ADD     R3,(R4)         ;ADJUST UBEVEC PSW ENTRY
2373  004224  005003            15$:    CLR     R3                      ;INIT R3
2374  004226  000261                    SEC
2375  004230  005777  176012            TST     @MBTTBL         ;IS MASS BUS TESTER THERE?
2376  004234  103403                    BCS     20$             ;BRANCH IF NO
2377  004236  052702  002000    21$:    BIS     #MBTOPT,R2      ;SET OPTION AVAILABLE
2378  004242  000422                    BR      24$
2379  004244  005777  176026    20$:    TST     @MBTN2          ;IS MBT2 THERE?
2380  004250  103403                    BCS     22$             ;BRANCH IF NO
2381  004252  012703  000100            MOV     #100,R3         ;SETUP R3
2382  004256  000767                    BR      21$
2383  004260  005777  176014    22$:    TST     @MBTN3          ;IS MBT3 THERE?
2384  004264  103403                    BCS     23$             ;BRANCH IF NO
2385  004266  012703  000200            MOV     #200,R3
```

```
                                        BR      21$
                                        TST     @MBTN4          ;IS MBT4 THERE?
                                        BCS     30$             ;BRANCH IF NO
                                        MOV     #300,R3
                                        BR      21$
                                24$:    INC     #-1
                                        BNE     30$
                                        MOV     #MBTTBL,R4      ;GET ADDRESS OF MBT TABLE
                                        MOV     #11,R5          ;SET SOB COUNT
                                25$:    ADD     R3,(R4)+        ;ADJUST MBT TABLE
                                        SOB     R5,25$          ;CONTINUE
                                        ADD     R3,@#MBTTBL+26  ;ADJUST DRIVE TYPE ADDRESS
                                        MOVB    #7,@MBTTBL+12   ;SET UNIT NUMBER
                                        CMPB    #40,@MBTTBL+26  ;IS THIS REALLY A MBT?
                                        BEQ     30$             ;BRANCH IF YES
                                        BIC     #MBTOPT,R2      ;CLEAR OPTION AVAILABLE BIT
                                30$:    MOV     #ERPRT,@#ERRVEC ;RESTORE ERROR TRAP
                                        MOV     #RESERR,@#RESVEC;AND ALSO RESERVED INST TRAP
                                        MOV     R2,@#OPT.CP     ;LOAD INDICATOR
                                        INC     #-1             ;;FIRST TIME?
                                        BNE     64$             ;;BRANCH IF NO
                                        CMP     #$ENDAD,@#42    ;;ACT-11?
                                        BEQ     64$             ;;BRANCH IF YES
                                        TYPE    ,65$            ;;TYPE ASCIZ STRING
                                        BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  <CRLF>"MAINDEC-11-DEQKC-B...PDP 11/70 CPU EXERCISOR"<CRLF>
                                64$:
                                ;;******************************************************************
                                .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.
                                ;;******************************************************************
004504  012737  004616  000004  SIZE:   MOV     #21$,@#ERRVEC           ;SETUP TIMEOUT VECTOR
004512  005037  001276          CLR     @#$TMPO                 ;ENSURE $TMPO CLEAR
004516  005000                  CLR     R0                      ;USED TO SET THE UNIT AVAIL BITS
004520  012701  000010          MOV     #10,R1                  ;SOB COUNT
004524  013777  001276  175402  9$:     MOV     @#$TMPO,@RKDA           ;SET UNIT NUMBER
004532  012777  000015  175366          MOV     #15,@RKCS               ;SEND DRIVE RESET
004540  032777  000200  175356          BIT     #BIT7,@RKER             ;NON EXISTANT DISK?
004546  001011                  BNE     7$                      ;BRANCH IF YES
004550  017702  175346          MOV     @RKDS,R2                ;GET DRIVE STATUS
004554  042702  177537          BIC     #177537,R2              ;GET BITS 5 & 7 ONLY
004560  022702  000200          CMP     #200,R2                 ;IS DRIVE READY?
004564  001002                  BNE     7$                      ;BRANCH IF NO
004566  052700  000400          BIS     #BIT8,R0                ;SET UNIT AVAILABLE
004572  006000                  7$:     ROR     R0
004574  012777  000001  175324          MOV     #1,@RKCS                ;CLEAR THE ERRORS
004602  062737  020000  001276          ADD     #20000,@#$TMPO          ;SELECT NEXT UNIT
004610  077133                  SOB     R1,9$                   ;CONTINUE
004612  110037  001610          MOVB    R0,@#SYSSIZE+2          ;STORE IN TABLE
```

```
2442                                    ;;*****************************************************************
2443                                    ;THIS CODE DETERMINES IF THERE IS AN RP03 OR AN RP04 OR BOTH.
2444                                    ;IF BOTH ARE ON THE SYSTEM, THE OPERATOR MUST CHANGE THE RP04
2445                                    ;ADDRESSES IN THE TABLE IN "COMMON TAGS" AND "NOP" THE BRANCH
2446                                    ;AT "100$".
2447
2448  004616  012737  005056  000004  21$:    MOV     #11$,@#ERRVEC           ;SET THE ERROR VECTOR
2449  004624  005737  176710           1$:     TST     @#176710                ;IS THERE AN RP ON THE SYSTEM?
2450                                                                          ;STAY HERE IF YES
2451  004630  012737  004644  000004          MOV     #1$,@#ERRVEC
2452  004636  005777  175300                  TST     @RP4CS1         ;IS THERE AN RP04 ON SYSTEM?
2453  004642  000441                  100$:   BR      10$                     ;BRANCH IF YES
2454                                    ;;*****************************************************************
2455
2456                                    ;;*****************************************************************
2457  004644  012737  004746  000004  1$:     MOV     #10$,@#ERRVEC           ;SETUP TIMEOUT VEC FOR RP03 TEST
2458  004652  012737  000001  001276          MOV     #1,@#$TMPO              ;SETUP TEMPO
2459  004660  005000                         CLR     R0                      ;USED TO SET UNIT AVAILABLE BITS
2460  004662  012701  000010                 MOV     #10,R1                  ;SOB COUNT
2461  004666  013777  001276  175210  3$:     MOV     @#$TMPO,@RP3CS          ;SET FUNCTION IDLE WITH UNIT NO
2462  004674  005777  175204                 TST     @RP3CS                  ;WAS THERE AN ERROR?
2463  004700  100006                         BPL     6$                      ;BRANCH IF NO
2464  004702  006000                 4$:     ROR     R0                      ;UNIT NOT AVAILABLE
2465  004704  062737  000400  001276         ADD     #400,@#$TMPO            ;SELECT NEXT UNIT
2466  004712  077113                         SOB     R1,3$                   ;CONTINUE
2467  004714  000412                         BR      5$
2468  004716  017702  175156          6$:    MOV     @RP3DS,R2               ;GET STATUS REGISTER
2469  004722  042702  136377                 BIC     #136377,R2              ;GET BITS 14, 9 & 8 ONLY
2470  004726  022702  041400                 CMP     #41400,R2               ;IS DRIVE READY?
2471  004732  001363                         BNE     4$                      ;BRANCH IF NO
2472  004734  052700  000400                 BIS     #BIT8,R0                ;SET DRIVE AVAILABLE BIT
2473  004740  000760                         BR      4$                      ;CONTINUE
2474  004742  110037  005506          5$:    MOVB    R0,@#SYSSIZE            ;STORE IN TABLE
2475
2476                                    ;;*****************************************************************
2477  004746  012737  005056  000004  10$:    MOV     #11$,@#ERRVEC           ;SETUP ERROR VEC FOR RP04 TEST
2478  004754  005037  001276                 CLR     @#$TMPO
2479  004760  005000                         CLR     R0                      ;UNIT AVAILABLE WORD
2480  004762  012701  000010                 MOV     #10,R1                  ;SOB COUNT
2481  004766  113777  001276  175160  14$:    MOVB    @#$TMPO,@RP4CS2         ;SET UNIT NUMBER
2482  004774  012777  000021  175140         MOV     #21,@RP4CS1             ;TRY READ-IN-PRESET
2483  005002  032777  010000  175144         BIT     #BIT12,@RP4CS2          ;NON EXISTANT DRIVE?
2484  005010  001011                         BNE     12$                     ;BRANCH IF YES
2485  005012  017702  175142                 MOV     @RP4DS,R2               ;GET DRIVE STATUS
2486  005016  042702  163277                 BIC     #163277,R2              ;GET BITS 12, 11, 8, & 6 ONLY
2487  005022  022702  010500                 CMP     #10500,R2               ;IS DRIVE READY?
2488  005026  001002                         BNE     12$                     ;BRANCH IF NO
2489  005030  052700  000400                 BIS     #BIT8,R0                ;SET UNIT AVAILABLE
2490  005034  006000                 12$:    ROR     R0
2491  005036  052777  000040  175110         BIS     #BIT5,@RP4CS2           ;CLEAR ERROR BITS
2492  005044  005237  001276                 INC     @#$TMPO                 ;SELECT NEXT DRIVE
2493  005050  077132                         SOB     R1,14$                  ;CONTINUE
2494  005052  110037  001616                 MOVB    R0,@#SYSSIZE+10         ;STORE IN TABLE
2495
2496                                    ;;*****************************************************************
2497  005056  012737  005166  000004  11$:    MOV     #15$,@#ERRVEC           ;SETUP ERROR VEC FOR RS04 TEST
```

```
2498  005064  005037  001276                  CLR     @#$TMPO
2499  005070  005000                          CLR     R0
2500  005072  012701  000010                  MOV     #10,R1                  ;SOB COUNT
2501  005076  113777  001276  175110  18$:    MOVB    @#$TMPO,@RSCS2          ;SET UNIT NUMBER
2502  005104  012777  000001  175070          MOV     #1,@RSCS1               ;TRY NOP OPERATION
2503  005112  032777  010000  175074          BIT     #BIT12,@RSCS2           ;NON EXISTANT DRIVE?
2504  005120  001011                          BNE     16$                     ;BRANCH IF YES
2505  005122  017702  175072                  MOV     @RSDS,R2                ;GET DRIVE STATUS
2506  005126  042702  163577                  BIC     #163577,R2              ;GET BITS 12, 11, & 7 ONLY
2507  005132  022702  010200                  CMP     #10200,R2               ;IS DRIVE READY?
2508  005136  001002                          BNE     16$                     ;BRANCH IF NO
2509  005140  052700  000400                  BIS     #BIT8,R0                ;SET DRIVE AVAILABLE BIT
2510  005144  006000                  16$:    ROR     R0
2511  005146  052777  000040  175040          BIS     #BIT5,@RSCS2            ;CLEAR ANY ERROR BITS
2512  005154  005237  001276                  INC     @#$TMPO                 ;SELECT NEXT UNIT
2513  005160  077132                          SOB     R1,18$                  ;CONTINUE
2514  005162  110037  001620                  MOVB    R0,@#SYSSIZE+12         ;STORE IN TABLE
2515                                  ;
2516                                  ;NEXT, DELETE XXDP UNIT 0 FROM TABLE
2517  005166  122737  000002  000041  15$:    CMPB    #2,@#41                 ;PK?
2518  005174  001004                          BNE     19$                     ;BRANCH IF NO
2519  005176  042737  000001  001610          BIC     #BIT0,@#SYSSIZE+2       ;MAKE UNIT ZERO NOT AVAILABLE
2520  005204  000420                          BR      20$
2521  005206  113700  000041          19$:    MOVB    @#41,R0                 ;GET LOCATION 41
2522  005212  042700  177770                  BIC     #177770,R0              ;GET LEAST SIG 3 BITS
2523  005216  000241                          CLC                             ;ENSURE C CLEAR
2524  005220  006100                          ROL     R0                      ;ADJUST
2525  005222  122700  000002                  CMPB    #2,R0                   ;
2526  005226  002404                          BLT     40$                     ;BRANCH IF NO
2527  005230  042737  000001  001606          BIC     #BIT0,@#SYSSIZE
2528  005236  000403                          BR      20$
2529  005240  042760  000001  001612  40$:    BIC     #BIT0,SYSSIZE+4(R0)
2530  005246  005227  177777          20$:    INC     #-1
2531  005252  001055                          BNE     LOOP                    ;;BRANCH IF NOT FIRST TIME
2532  005254  104400  054762                  TYPE    ,MSG25
2533  005260  013746  001470                  MOV     @#OPT.CP,-(SP)
2534  005264  104402                          TYPOC
2535  005266  104400  001327                  TYPE    ,$CRLF
2536  005272  005737  001504                  TST     @#QV                    ;ACT11?
2537  005276  001043                          BNE     LOOP                    ;BRANCH IF YES
2538  005300  105737  003241          50$:    TSTB    @#XXDPC                 ;XXDP CHAIN MODE?
2539  005304  001040                          BNE     LOOP                    ;;BRANCH IF YES
2540  005306  105737  001200                  TSTB    @#$PASS                 ;FIRST PASS?
2541  005312  001035                          BNE     LOOP                    ;;BRANCH IF NO
2542  005314  032737  000200  177570          BIT     #SW7,@#SWR              ;INHIBIT SIZE TYPEOUT?
2543  005322  001031                          BNE     LOOP                    ;;BRANCH IF YES
2544  005324  004767  041566                  JSR     PC,TYPSIZ               ;GO TYPE SYSTEM SIZE
2545  005330  104400  005336                  TYPE    ,65$                    ;;TYPE ASCIZ STRING
2546  005334  000417                          BR      64$                     ;;GET OVER THE ASCIZ
2547                                  ;;65$:  .ASCIZ  /TYPE A CHARACTER TO CONTINUE/<CRLF>
2548  005374                          64$:
2549  005374  005037  177776                  CLR     @#PSW
2550  005400  000001                          WAIT
2551  005402  000137  004504                  JMP     @#SIZE                  ;GO CHECK SYSTEM AGAIN
2552                                  ;PROGRAM RESTARTS HERE AFTER RELOCATION ABOVE 28K IS COMPLETE.
2553                                  ;INITIALIZE TRAP VECTORS
```

```
2554  005406  012706  000700          LOOP:   MOV     #SUPSTK,SP              ;SET THE STACK...WILL BE DIFFERENT
2555                                                                          ;THAN KERN STACK WHEN IN OUTER MODE
2556  005412  012700  000004                  MOV     #ERRVEC,R0
2557  005416  013701  177776                  MOV     @#PSW,R1                ;GET CURRENT PSW
2558  005422  012720  053440                  MOV     #ERPRT,(R0)+            ;SET ERROR VEC
2559  005426  052701  000340                  BIS     #PR7,R1         ;SET PRIORITY 7 IN CURRENT PSW
2560  005432  042701  000020                  BIC     #BIT4,R1        ;CLEAR T BIT
2561  005436  010120                          MOV     R1,(R0)+
2562  005440  012720  053366                  MOV     #RESERR,(R0)+           ;SET RESERVED INST TRAP VECTOR
2563  005444  010120                          MOV     R1,(R0)+
2564  005446  012720  001472                  MOV     #$RTRN,(R0)+            ;SET T BIT VEC
2565  005452  042701  000340                  BIC     #PR7,R1
2566  005456  005020                          CLR     (R0)+                   ;SET TBIT VEC+2
2567  005460  005720                          TST     (R0)+                   ;BUMP R0 TO SCOPE VEC+2
2568  005462  005020                          CLR     (R0)+                   ;SET SCOPE VEC+2
2569  005464  062700  000006                  ADD     #6,R0                   ;SET R0 TO ERROR TRAP VEC
2570  005470  012720  000340                  MOV     #PR7,(R0)+              ;SET ERROR VEC
2571  005474  005720                          TST     (R0)+
2572  005476  012720  000340                  MOV     #PR7,(R0)+              ;SET TRAP VEC+2
2573  005502  012737  052640  000114          MOV     #.PARSRV,@#CACHVEC      ;SET PARITY ERROR VECTOR
2574  005510  052701  000340                  BIS     #PR7,R1
2575  005514  010137  000116                  MOV     R1,@#CACHVEC+2
2576  005520  012737  053272  000250          MOV     #KTABRT,@#MMVEC         ;SET KT11 ABORT VECTOR
2577  005526  010137  000252                  MOV     R1,@#MMVEC+2
2578  005532  042737  000340  177776          BIC     #PR7,@#PSW
2579                                  ;;**************************************************************
2580                                  ;*TEST 1         MEMORY VERIFICATION TEST
2581                                  ;;**************************************************************
2582  005540  112737  000001  001202  TST1:   MOVB    #1,@#$TSTNM             ;LOAD TEST NUMBER
2583  005546  012767  000001  173542          MOV     #1,$TIMES       ;;DO 1 ITERATION
2584  005554  000004                          SCOPE
2585
2586                                          .SBTTL  START OF SECTION 0
2587                                  ;000000000000 FIRST ADDRESS TO BE RELOCATED 000000000
2588  005556  010700                  RELO:   MOV     PC,R0                   ;GET PC
2589  005560  005740                          TST     -(R0)                   ;R0 CONTAINS THE ADDRESS OF RELO
2590  005562  010037  001512                  MOV     R0,@#FRSTAD             ;SAVE
2591  005566  010700                          MOV     PC,R0                   ;GET CURRENT PC
2592  005570  162700  005570                  SUB     #.,R0                   ;SUBTRACT RELOCATION FACTOR
2593  005574  010037  001506                  MOV     R0,@#FACTOR             ;SAVE RELOCATION FACTOR
2594  005600  010737  001212                  MOV     PC,@#$LPERR             ;SET LOOP ADDRESS
2595  005604  062737  000030  001212          ADD     #30,@#$LPERR            ;ADJUST
2596  005612  013737  001212  001210          MOV     @#$LPERR,@#$LPADR
2597  005620  105737  001502                  TSTB    @#NEXEC                 ;BR IF TEST CODE TO BE EXECUTED
2598  005624  001402                          BEQ     .+6
2599  005626  000167  000720                  JMP     RELE0
2600                                  ;MEMORY AND DISK (IF SELECTED) VERIFICATION TEST.
2601  005632  000167  000714                  JMP     1$
2602  005636  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
2603  005644  177777  000000  000000
2604  005652  000000  000000
2605  005656  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
2606  005664  177777  000000  000000
2607  005672  000000  000000
2608  005676  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
2609  005704  177777  000000  000000
```

```
 2610  005712  000000  000000
 2611  005716  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2612  005724  177777  000000  000000
 2613  005732  000000  000000
 2614  005736  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2615  005744  177777  000000  000000
 2616  005752  000000  000000
 2617  005756  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2618  005764  177777  000000  000000
 2619  005772  000000  000000
 2620  005776  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2621  006004  177777  000000  000000
 2622  006012  000000  000000
 2623  006016  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2624  006024  177777  000000  000000
 2625  006032  000000  000000
 2626  006036  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2627  006044  177777  000000  000000
 2628  006052  000000  000000
 2629  006056  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2630  006064  177777  000000  000000
 2631  006072  000000  000000
 2632  006076  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2633  006104  177777  000000  000000
 2634  006112  000000  000000
 2635  006116  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2636  006124  177777  000000  000000
 2637  006132  000000  000000
 2638  006136  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2639  006144  177777  000000  000000
 2640  006152  000000  000000
 2641  006156  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2642  006164  177777  000000  000000
 2643  006172  000000  000000
 2644  006176  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2645  006204  177777  000000  000000
 2646  006212  000000  000000
 2647  006216  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2648  006224  177777  000000  000000
 2649  006232  000000  000000
 2650  006236  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2651  006244  177777  000000  000000
 2652  006252  000000  000000
 2653  006256  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2654  006264  177777  000000  000000
 2655  006272  000000  000000
 2656  006276  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2657  006304  177777  000000  000000
 2658  006312  000000  000000
 2659  006316  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2660  006324  177777  000000  000000
 2661  006332  000000  000000
 2662  006336  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
 2663  006344  177777  000000  000000
 2664  006352  000000  000000
 2665  006356  177777  177777  177777          .WORD   -1,-1,-1,-1,0,0,0,0
```

```
2666   006364   177777   000000   000000
2667   006372   000000   000000
2668   006376   177777   177777   177777          .WORD   -1,-1,-1,-1,0,0,0,0
2669   006404   177777   000000   000000
2670   006412   000000   000000
2671   006416   177777   177777   177777          .WORD   -1,-1,-1,-1,0,0,0,0
2672   006424   177777   000000   000000
2673   006432   000000   000000
2674   006436   177777   177777   177777          .WORD   -1,-1,-1,-1,0,0,0,0
2675   006444   177777   000000   000000
2676   006452   000000   000000
2677   006456   177777   177777   177777          .WORD   -1,-1,-1,-1,0,0,0,0
2678   006464   177777   000000   000000
2679   006472   000000   000000
2680   006476   177777   177777   177777          .WORD   -1,-1,-1,-1,0,0,0,0
2681   006504   177777   000000   000000
2682   006512   000000   000000
2683   006516   177777   177777   177777          .WORD   -1,-1,-1,-1,0,0,0,0
2684   006524   177777   000000   000000
2685   006532   000000   000000
2686   006536   177777   177777   177777          .WORD   -1,-1,-1,-1,0,0
2687   006544   177777   000000   000000
2688   006552                                  1$:
2689   006552   000004                         RELED:  SCOPE
2690   006554   010702                                 MOV     PC,R2
2691   006556   062702   000012                        ADD     #12,R2
2692   006562   012707   034240                        MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
2693   006566   000000                         RELOD:  .WORD   0
2694                                           ;000000000000 LAST ADDRESS OF CODE TO BE RELOCATED 000000000000
2695
2696                                           ;;*****************************************************************
2697                                           ;*TEST 2         CHECK BRANCH INSTRUCTIONS
2698                                           ;;*****************************************************************
2699   006570   112737   000002   001202      TST2:   MOVB    #2,@#$TSTNM             ;LOAD TEST NUMBER
2700   006576   012767   000001   172512              MOV     #1,$TIMES       ;;DO 1 ITERATION
2701   006604   000004                                SCOPE
2702
2703                                                   .SBTTL  START OF SECTION 1
2704                                           ;111111111111 FIRST ADDRESS TO BE RELOCATED 111111111
2705   006606   010700                         REL1:   MOV     PC,R0           ;GET PC
2706   006610   005740                                 TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL1
2707   006612   010037   001512                        MOV     R0,@#FRSTAD     ;SAVE
2708   006616   010700                                 MOV     PC,R0           ;GET CURRENT PC
2709   006620   162700   006620                        SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
2710   006624   010037   001506                        MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
2711   006630   010737   001212                        MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
2712   006634   062737   000030   001212              ADD     #30,@#$LPERR    ;ADJUST
2713   006642   013737   001212   001210              MOV     @#$LPERR,@#$LPADR
2714   006650   105737   001502                        TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
2715   006654   001402                                 BEQ     .+6
2716   006656   000167   004052                        JMP     RELE1
2717                                           ;
2718   006662   000257                                 CCC                     ;CC'S=0000
2719   006664   103407                                 BCS     CC0             ;SAME AS BLO
2720   006666   102406                                 BVS     CC0
2721   006670   001405                                 BEQ     CC0
```

```
2722  006672  100404                    BMI     CC0
2723  006674  002403                    BLT     CC0
2724  006676  003402                    BLE     CC0
2725  006700  101401                    BLOS    CC0
2726  006702  101001                    BHI     .+4
2727  006704  104000            CC0:    HLT                     ;ONE OF THE ABOVE BRANCHES FAILED
2728
2729                            ;CONTINUE
2730  006706  000270                    SEN                     ;CC'S=1000
2731  006710  100003                    BPL     CC1
2732  006712  002002                    BGE     CC1
2733  006714  003001                    BGT     CC1
2734  006716  002401                    BLT     .+4
2735  006720  104000            CC1:    HLT                     ;ONE OF THE ABOVE BRANCHES FAILED
2736
2737                            ;CONTINUE
2738  006722  000262                    SEV                     ;CC'S=1010
2739  006724  102003                    BVC     CC2
2740  006726  002402                    BLT     CC2
2741  006730  003401                    BLE     CC2
2742  006732  002001                    BGE     .+4
2743  006734  104000            CC2:    HLT                     ;ERROR! ONE OF THE ABOVE BRANCHES FAILED
2744
2745                            ;CONTINUE
2746  006736  000261                    SEC                     ;CC'S=1011
2747  006740  103002                    BCC     CC3
2748  006742  101001                    BHI     CC3
2749  006744  003001                    BGT     .+4
2750  006746  104000            CC3:    HLT                     ;ERROR! ONE OF THE ABOVE BRANCHES FAILED
2751
2752                            ;CONTINUE
2753  006750  000264                    SEZ                     ;CC'S=1111
2754  006752  001003                    BNE     CC4
2755  006754  003002                    BGT     CC4
2756  006756  101001                    BHI     CC4
2757  006760  003401                    BLE     .+4
2758  006762  104000            CC4:    HLT                     ;ERROR! ONE OF THE ABOVE BRANCHES FAILED
2759                            ;;*****************************************************************
2760                            ;*TEST 3        TEST UNIARY CONDITION CODES
2761                            ;;*****************************************************************
2762  006764  112737  000003  001202  TST3:   MOVB    #3,@#$TSTNM             ;LOAD TEST NUMBER
2763  006772  000004                    SCOPE
2764                            ;CLR    R0
2765  006774  000277                    SCC
2766  006776  000244                    CLZ
2767  007000  005000                    CLR     R0              ;R0=0,CC'S=0100
2768  007002  103404                    BCS     CLR0
2769  007004  102403                    BVS     CLR0
2770  007006  001002                    BNE     CLR0
2771  007010  100401                    BMI     CLR0
2772  007012  003401                    BLE     .+4
2773  007014  104000            CLR0:   HLT                     ;ERROR! INCORRECT CC'S AFTER CLR
2774
2775  007016  000277                    SCC
2776  007020  000244                    CLZ
2777  007022  005700                    TST     R0              ;R0=0,CC'S=0100
```

```
2778  007024  103404                    BCS     TSTO
2779  007026  102403                    BVS     TSTO
2780  007030  001002                    BNE     TSTO
2781  007032  100401                    BMI     TSTO
2782  007034  101401                    BLOS    .+4
2783  007036  104000            TSTO:   HLT                     ;ERROR! INCORRECT CC'S AFTER TST
2784
2785  007040  000257                    CCC
2786  007042  000266                    +SEZ!SEV
2787  007044  005100                    COM     R0              ;R0=-1,CC'S=1001
2788  007046  103004                    BCC     COM0
2789  007050  102403                    BVS     COM0
2790  007052  001402                    BEQ     COM0
2791  007054  100001                    BPL     COM0
2792  007056  002401                    BLT     .+4
2793  007060  104000            COM0:   HLT                     ;ERROR! INCORRECT CC'S AFTER COM
2794
2795  007062  000261                    SEC
2796  007064  005500                    ADC     R0              ;R0=000000,CC'S=0101
2797  007066  103003                    BCC     ADC0
2798  007070  102402                    BVS     ADC0
2799  007072  001001                    BNE     ADC0
2800  007074  002001                    BGE     .+4
2801  007076  104000            ADC0:   HLT                     ;ERROR! INCORRECT CC'S AFTER ADC
2802
2803  007100  000261                    SEC
2804  007102  006000                    ROR     R0              ;R0=100000,CC'S=1010
2805  007104  103404                    BCS     ROR0
2806  007106  102003                    BVC     ROR0
2807  007110  001402                    BEQ     ROR0
2808  007112  100001                    BPL     ROR0
2809  007114  003001                    BGT     .+4
2810  007116  104000            ROR0:   HLT                     ;ERROR! INCORRECT CC'S AFTER ROR
2811  007120  000277                    SCC
2812  007122  000242                    CLV
2813  007124  005300                    DEC     R0              ;R0=077777,CC'S=0011
2814  007126  103004                    BCC     DEC0
2815  007130  102003                    BVC     DEC0
2816  007132  001402                    BEQ     DEC0
2817  007134  100401                    BMI     DEC0
2818  007136  003401                    BLE     .+4
2819  007140  104000            DEC0:   HLT                     ;ERROR! INCORRECT CC'S AFTER DEC
2820
2821  007142  000257                    CCC
2822  007144  005200                    INC     R0              ;R0=100000,CC'S=1010
2823  007146  103404                    BCS     INC0
2824  007150  102003                    BVC     INC0
2825  007152  001402                    BEQ     INC0
2826  007154  100001                    BPL     INC0
2827  007156  003001                    BGT     .+4
2828  007160  104000            INC0:   HLT                     ;ERROR! INCORRECT CC'S AFTER INC
2829
2830  007162  000277                    SCC
2831  007164  000242                    CLV
2832  007166  005400                    NEG     R0              ;R0=100000,CC'S=1011
2833  007170  103003                    BCC     NEG0
```

```
                                        BVC     NEGO
                                        BEQ     NEGO
                                        BGE     .+4
                                NEGO:   HLT                     ;ERROR! INCORRECT CC'S AFTER NEG

                                        SEC
                                        ASL     R0              ;R0=000000,CC'S=0111
                                        BCC     ASLO
                                        BVC     ASLO
                                        BNE     ASLO
                                        BMI     ASLO
                                        BLOS    .+4
                                ASLO:   HLT                     ;ERROR! INCORRECT CC'S AFTER ASL

                                        ROL     R0              ;R0=000001,CC'S=0000
                                        BCS     ROLO
                                        BLE     ROLO
                                        BGE     .+4
                                ROLO:   HLT                     ;ERROR! INCORRECT CC'S AFTER ROL

                                        ASR     R0              ;R0=000000,CC'S=0111
                                        BCC     ASRO
                                        BVC     ASRO
                                        BNE     ASRO
                                        BLT     .+4
                                ASRO:   HLT                     ;ERROR! INCORRECT CC'S AFTER ASR

                                        SCC
                                        SBC     R0              ;R0=-1,CC'S=1001
                                        BCC     SBCO
                                        BVS     SBCO
                                        BLE     .+4
                                SBCO:   HLT                     ;ERROR! INCORRECT CC'S AFTER SBC

                                        NEG     R0              ;R0=000001,CC'S=00001
                                        SWAB    R0              ;R0=000400',CC'S=0100
                                        BCS     SWABO
                                        BVS     SWABO
                                        BNE     SWABO
                                        BGE     .+4
                                SWABO:  HLT                     ;ERROR! INCORRECT CC'S AFTER SWAB
                                ;;*****************************************************************
                                ;*TEST 4         CHECK REGISTER SELECTION
                                ;;*****************************************************************
000004  001202                TST4:   MOVB    #4,@#STSTNM             ;LOAD TEST NUMBER
                                        SCOPE
000005  001316                        MOV     #5,@#STIMES     ;SET ITERATION COUNT TO 5
                                        CLR     R0
                                        SCC
                                        ROL     R0              ;R0=1
                                        MOV     R0,R2
                                        ASL     R2              ;R2=2
                                        MOV     R2,R3
                                        ASL     R3              ;R3=4
                                        MOV     R3,R4
                                        ASL     R4              ;R4=10
```

```
                                        MOV     R4,R5
                                        ASL     R5              ;R5=20
                                        MOV     R5,-(SP)        ;SET BITS SET IN REGISTERS
                                        BIS     R4,(SP)         ;INTO STACK ADDRESS
                                        BIS     R3,(SP)
                                        BIS     R2,(SP)
                                        BIS     R0,(SP)
                                        CMP     #37,(SP)+
                                        BEQ     .+4             ;WERE SET
                                        HLT                     ;MISSING BIT(S) REPRESENT
                                                                ;INCORRECT REGISTER SELECTION

                                ;CHECK THAT ALL BITS CAN BE SET & CLEARED IN ALL REGISTERS
                                        CCC
                                        MOVB    #377,R0         ;SET ALL BITS (MOVB EXTENDS SIGN)
                                1$:     ROL     R0              ;ROTATE A 0 THROUGH ALL BIT
                                        BCS     1$              ;POSITIONS
                                        INC     R0              ;FINAL RESULT IS -1
                                        BEQ     .+4
                                        HLT                     ;ERROR!

                                        MOV     #16.,R0         ;SET SHIFT COUNT
                                        CLR     R2
                                2$:     SEC
                                        ROR     R2              ;ROTATE 1 THROUGH ALL BIT POSITS
                                        DEC     R0              ;DECREMENT SHIFT COUNT
                                        BNE     2$
                                        COM     R2              ;R2 SHOULD CONTAIN -1
                                        BEQ     .+4
                                        HLT                     ;ERROR! CHECK R2 SHOULD = 0

                                        MOV     #100000,R3
                                3$:     ASR     R3              ;EXTEND 1 BIT THROUGH ALL POSITIONS
                                        BCC     3$
                                        INC     R3
                                        BEQ     .+4
                                        HLT                     ;ERROR!

                                        MOVB    #177401,R4      ;R4=1
                                4$:     ADD     R4,R4           ;HAS THE AFFECT OF SHIFTING A BIT
                                        BCC     4$              ;THROUGH ALL POSITIONS
                                        TST     R4              ;RESULT SHOULD BE 0
                                        BEQ     .+4
                                        HLT

                                        MOV     #1,R5
                                5$:     ASL     R5
                                        BVC     5$
                                        ASL     R5
                                        BCC     6$
                                        TST     R5
                                        BEQ     .+4
                                6$:     HLT

                                ;CHECK REGISTER VOLITILITY
                                        CLR     R2
```

```
2946  007514  005102                        COM     R2              ;R2=-1
2947  007516  010203                        MOV     R2,R3
2948  007520  000257                        CCC
2949  007522  006002                        ROR     R2              ;R2=LOOP COUNT
2950  007524  006202                        ASR     R2
2951  007526  010304                7$:     MOV     R3,R4
2952  007530  005302                        DEC     R2              ;DECREMENT LOOP COUNT
2953  007532  001375                        BNE     7$
2954  007534  005203                        INC     R3              ;CHECK R3
2955  007536  001002                        BNE     8$
2956  007540  005204                        INC     R4              ;CHECK R4
2957  007542  001401                        BEQ     .+4
2958  007544  104000                8$:     HLT
2959
2960                                  ;CHECK TRANSFER OF REGISTER DATA BETWEEN THE GS AND GD REGISTERS
2961  007546  032737  000020  177776 GSTST:  BIT     #20,@#PSW       ;CHECK IF 'T' BIT IS SET
2962  007554  001050                        BNE     7$              ;SKIP TEST IF 'T' BIT SET
2963  007556  010627                        MOV     SP,(PC)+        ;SAVE STACK PTR
2964  007560  000000                1$:     .WORD   0               ;CONTAINS SAVED STACK PTR
2965  007562  010727                        MOV     PC,(PC)+        ;LOAD DATA. THE CURRENT PC IS USED AS
2966  007564  000000                2$:     .WORD   0               ;DATA. IF THIS TEST FAILS 2$ CON-
2967                                                                ;TAINS THE DATA BEING USED.
2968  007566  005267  177772              INC     2$              ;MAKE ODD TO CHECK BIT 0
2969  007572  016700  177766        3$:     MOV     2$,R0           ;LOAD GD REGISTER 0
2970  007576  010001                        MOV     R0,R1           ;TRANSFER GS REG 0 TO GD REG 1
2971  007600  010102                        MOV     R1,R2           ;AND GS REG 1 TO GD REG 2
2972  007602  010203                        MOV     R2,R3           ;ETC...
2973  007604  010304                        MOV     R3,R4
2974  007606  010405                        MOV     R4,R5
2975  007610  152737  000340  177776        BISB    #340,@#PSW      ;SET PRIORITY LEVEL 7
2976  007616  010506                        MOV     R5,SP           ;TRANSFER GS REG 5 TO GD STK PTR
2977  007620  010627                        MOV     SP,(PC)+        ;TRANSFER GS STK PTR TO MEMORY
2978  007622  000000                4$:     .WORD   0               ;CONTAINS GS STACK PTR
2979  007624  016706  177730              MOV     1$,SP           ;RESTORE STK PTR NEEDED FOR HLT/SCOPE
2980  007630  142737  000340  177776        BICB    #340,@#PSW      ;SET PRIORITY LEVEL 0
2981  007636  026700  177760              CMP     4$,R0           ;COMPARE GS/GD STACK WITH GS REG 0
2982  007642  001004                        BNE     5$              ;BRANCH IF THEY WERE NOT =
2983  007644  006367  177714              ASL     2$              ;SHIFT TEST DATA UNTIL = 000000
2984  007650  001350                        BNE     3$
2985  007652  000411                        BR      6$
2986  007654  010046                5$:     MOV     R0,-(SP)        ;GET GS REG 0
2987  007656  010146                        MOV     R1,-(SP)        ;ETC...
2988  007660  010246                        MOV     R2,-(SP)
2989  007662  010346                        MOV     R3,-(SP)
2990  007664  010446                        MOV     R4,-(SP)
2991  007666  010546                        MOV     R5,-(SP)
2992  007670  104000                        HLT                     ;ERROR! DATA IN GS STK PTR NOT = GS REG 0
2993                                                                ;GS REG 0-GS REG 5 ARE ON THE STACK
2994  007672  016706  177662              MOV     1$,SP           ;RESTORE STACK PTR
2995  007676                        6$:
2996  007676                        7$:
2997                                  ;;*****************************************************************
2998                                  ;*TEST 5         TEST UNIARY WORD INSTRUCTIONS USING ADDRESS MODE 1
2999                                  ;;*****************************************************************
3000  007676  112737  000005  001202 TST5:   MOVB    #5,@#STSTNM             ;LOAD TEST NUMBER
3001  007704  000004                        SCOPE
```

```
3002  007706  012737  000005  001316          MOV     #5,@#$TIMES
3003  007714  000401                          BR      .+4
3004  007716  000000                          .WORD   0               ;RESERVE ADDRESS FOR TESTS
3005  007720  010702                          MOV     PC,R2
3006  007722  162702  000004                  SUB     #4,R2           ;R2 POINTS TO RESERVED WORD
3007  007726  005012                          CLR     (R2)            ;PRESET (R2)
3008
3009  007730  000261                          SEC
3010  007732  006012                          ROR     (R2)            ;(R2)=100000,CC=1010
3011  007734  101402                          BLOS    ROR1
3012  007736  100001                          BPL     ROR1
3013  007740  002001                          BGE     .+4
3014  007742  104000                  ROR1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3015
3016  007744  000257                          CCC
3017  007746  000261                          SEC
3018  007750  005312                          DEC     (R2)            ;(R2)=077777,CC=0011
3019  007752  103001                          BCC     DEC1
3020  007754  003401                          BLE     .+4
3021  007756  104000                  DEC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3022
3023  007760  000257                          CCC
3024  007762  000261                          SEC
3025  007764  005512                          ADC     (R2)            ;(R2)=100000,CC=1010
3026  007766  103403                          BCS     ADC1
3027  007770  102002                          BVC     ADC1
3028  007772  100001                          BPL     ADC1
3029  007774  001001                          BNE     .+4
3030  007776  104000                  ADC1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3031
3032  010000  006112                          ROL     (R2)            ;(R2)=000000,CC=0111
3033  010002  103003                          BCC     ROL1
3034  010004  102002                          BVC     ROL1
3035  010006  001001                          BNE     ROL1
3036  010010  100001                          BPL     .+4
3037  010012  104000                  ROL1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3038
3039  010014  006112                          ROL     (R2)            ;(R2)=000001,CC=0000
3040  010016  101402                          BLOS    ROL1A           ;BRANCH IF C OR Z IS SET
3041  010020  102401                          BVS     ROL1A
3042  010022  100001                          BPL     .+4
3043  010024  104000                  ROL1A:  HLT
3044
3045  010026  006212                          ASR     (R2)            ;(R2)=000000,CC=0111
3046  010030  103003                          BCC     ASR1
3047  010032  102002                          BVC     ASR1
3048  010034  001001                          BNE     ASR1
3049  010036  100001                          BPL     .+4
3050  010040  104000                  ASR1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3051
3052  010042  006012                          ROR     (R2)            ;(R2)=100000,CC=1010
3053  010044  103403                          BCS     ROR1A
3054  010046  102002                          BVC     ROR1A
3055  010050  001401                          BEQ     ROR1A
3056  010052  100401                          BMI     .+4
3057  010054  104000                  ROR1A:  HLT
```

```
3058
3059  010056  000261                    SEC
3060  010060  005212                    INC     (R2)            ;(R2)=100001,CC=1001
3061  010062  103003                    BCC     INC1
3062  010064  102402                    BVS     INC1
3063  010066  001401                    BEQ     INC1
3064  010070  100401                    BMI     .+4
3065  010072  104000              INC1: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3066
3067  010074  005612                    SBC     (R2)            ;(R2)=100000,CC=1000
3068  010076  103403                    BCS     SBC1
3069  010100  102402                    BVS     SBC1
3070  010102  001401                    BEQ     SBC1
3071  010104  100401                    BMI     .+4
3072  010106  104000              SBC1: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3073
3074  010110  000261                    SEC
3075  010112  005612                    SBC     (R2)            ;(R2)=077777,CC=0010
3076  010114  103403                    BCS     SBC1A
3077  010116  102002                    BVC     SBC1A
3078  010120  001401                    BEQ     SBC1A
3079  010122  100001                    BPL     .+4
3080  010124  104000              SBC1A: HLT                    ;ERROR! INCORRECT CC'S AS SHOEN ABOVE
3081
3082  010126  000261                    SEC
3083  010130  005512                    ADC     (R2)            ;(R2)=100000,CC=1010
3084  010132  100401                    BMI     .+4
3085  010134  104000                    HLT
3086
3087  010136  000261                    SEC
3088  010140  006312                    ASL     (R2)            ;(R2)=000000,CC=0111
3089  010142  103003                    BCC     ASL1
3090  010144  102002                    BVC     ASL1
3091  010146  001001                    BNE     ASL1
3092  010150  100001                    BPL     .+4
3093  010152  104000              ASL1: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3094
3095  010154  005112                    COM     (R2)            ;(R2)=177777,CC=1001
3096  010156  103002                    BCC     COM1
3097  010160  102401                    BVS     COM1
3098  010162  100401                    BMI     .+4
3099  010164  104000              COM1: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3100
3101  010166  000250                    CLN
3102  010170  005712                    TST     (R2)            ;(R2)=177777,CC=1000
3103  010172  103403                    BCS     TEST1
3104  010174  102402                    BVS     TEST1
3105  010176  100001                    BPL     TEST1
3106  010200  001001                    BNE     .+4
3107  010202  104000              TEST1: HLT                    ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3108
3109  010204  000262                    SEV
3110  010206  005412                    NEG     (R2)            ;(R2)=000001,CC=0000
3111  010210  103002                    BCC     NEG1
3112  010212  102401                    BVS     NEG1
3113  010214  001001                    BNE     .+4
```

```
3114  010216  104000                    NEG1:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3115
3116  010220  005312                            DEC     (R2)            ;(R2)=000000,CC=0101
3117  010222  103001                            BCC     DEC1A
3118  010224  001401                            BEQ     .+4
3119  010226  104000                    DEC1A:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3120                                      ;;*****************************************************************
3121                                      ;*TEST 6         CHECK UNIARY BYTE INSTRUCTIONS USING ADDRESS MODE 1
3122                                      ;;*****************************************************************
3123  010230  112737  000006  001202    TST6:   MOVB    #6,@#STSTNM             ;LOAD TEST NUMBER
3124  010236  000004                            SCOPE
3125  010240  000401                            BR      .+4             ;RESERVE A WORD
3126  010242  000000                            .WORD   0               ;ADDRESS RESERVED FOR TESTS
3127  010244  010703                            MOV     PC,R3
3128  010246  162703  000004                    SUB     #4,R3           ;R3 POINTS TO EVEN BYTE OF WORD
3129  010252  010304                            MOV     R3,R4           ;R4 POINTS TO ODD BYTE OF WORD
3130  010254  005204                            INC     R4
3131  010256  005013                            CLR     (R3)            ;PRESET DATA
3132
3133  010260  000261                    1$:     SEC
3134  010262  105513                            ADCB    (R3)            ;ADD CARRY TO EVEN BYTE
3135  010264  100402                            BMI     2$              ;UNTIL EVEN BYTE BECOMES NEGATIVE
3136  010266  105214                            INCB    (R4)            ;INCREMENT ODD BYTE
3137  010270  000773                            BR      1$
3138  010272  102401                    2$:     BVS     .+4             ;(R3)=077600=[0774][200],CC=1010
3139  010274  104000                            HLT
3140  010276  000242                            CLV
3141  010300  105214                            INCB    (R4)            ;(R3)=100200=[1000][200],CC=1010
3142  010302  103402                            BCS     INCB1
3143  010304  102001                            BVC     INCB1
3144  010306  100401                            BMI     .+4
3145  010310  104000                    INCB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3146
3147  010312  106114                            ROLB    (R4)            ;(R3)=000200=[0000][200],CC=0111
3148  010314  103002                            BCC     ROLB1
3149  010316  102001                            BVC     ROLB1
3150  010320  001401                            BEQ     .+4
3151  010322  104000                    ROLB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3152
3153  010324  105614                            SBCB    (R4)            ;(R3)=177600=[1774][200], CC=1001
3154  010326  103002                            BCC     SBCB1
3155  010330  102401                            BVS     SBCB1
3156  010332  100401                            BMI     .+4
3157  010334  104000                    SBCB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3158
3159  010336  106313                            ASLB    (R3)            ;(R3)=177400,CC=0111
3160  010340  103002                            BCC     ASLB1
3161  010342  102001                            BVC     ASLB1
3162  010344  001401                            BEQ     .+4
3163  010346  104000                    ASLB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3164
3165  010350  105413                            NEGB    (R3)            ;(R3)=177400,CC=0100
3166  010352  103402                            BCS     NEGB1
3167  010354  102401                            BVS     NEGB1
3168  010356  001401                            BEQ     .+4
3169  010360  104000                    NEGB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
```

```
3170
3171  010362  000277                    SCC
3172  010364  105313                    DECB    (R3)            ;(R3)=177777,CC=1001
3173  010366  103002                    BCC     DECB1
3174  010370  102401                    BVS     DECB1
3175  010372  001001                    BNE     .+4
3176  010374  104000            DECB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3177
3178  010376  000241                    CLC
3179  010400  106013                    RORB    (R3)            ;(R3)=177577,CC=0011
3180  010402  103002                    BCC     RORB1
3181  010404  102001                    BVC     RORB1
3182  010406  100001                    BPL     .+4
3183  010410  104000            RORB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3184
3185  010412  000241                    CLC
3186  010414  105114                    COMB    (R4)            ;(R3)=000177,CC=0101
3187  010416  103002                    BCC     COMB1
3188  010420  102401                    BVS     COMB1
3189  010422  001401                    BEQ     .+4
3190  010424  104000            COMB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3191
3192  010426  106213            1$:     ASRB    (R3)            ;SHIFT EVEN BYTE UNTIL V CLEARS
3193  010430  102002                    BVC     2$
3194  010432  105514                    ADCB    (R4)            ;AND ADD CARRY TO ODD BYTE
3195  010434  000774                    BR      1$
3196  010436  103401            2$:     BCS     ASRB1
3197  010440  001401                    BEQ     .+4
3198  010442  104000            ASRB1:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3199
3200  010444  106214                    ASRB    (R4)
3201  010446  106214                    ASRB    (R4)            ;(R3)=000400,CC=0011
3202  010450  103002                    BCC     ASRB1A
3203  010452  102001                    BVC     ASRB1A
3204  010454  001001                    BNE     .+4
3205  010456  104000            ASRB1A: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3206
3207  010460  105314                    DECB    (R4)            ;(R3)=000000,CC=0100
3208  010462  001401                    BEQ     .+4
3209  010464  104000                    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3210
3211  010466  000261                    SEC
3212  010470  106014                    RORB    (R4)            ;(R3)=100000,CC=1010
3213  010472  103402                    BCS     RORB1A
3214  010474  102001                    BVC     RORB1A
3215  010476  100401                    BMI     .+4
3216  010500  104000            RORB1A: HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3217
3218  010502  000242                    CLV
3219  010504  105314                    DECB    (R4)            ;(R3)=077400,CC=0100
3220  010506  102401                    BVS     .+4
3221  010510  104000                    HLT
3222
3223  010512  000261                    SEC
3224  010514  105313                    DECB    (R3)            ;(R3)=077777,CC=1001
3225  010516  103002                    BCC     DECB1A
```

```
3226  010520  102401                        BVS     DECB1A
3227  010522  100401                        BMI     .+4
3228  010524  104000              DECB1A:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3229
3230  010526  000277                        SCC
3231  010530  000313                        SWAB    (R3)            ;(R3)=177577=[1774][177],CC=0000
3232  010532  103402                        BCS     SWAB1
3233  010534  102401                        BVS     SWAB1
3234  010536  100001                        BPL     .+4
3235  010540  104000              SWAB1:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3236
3237  010542  105714                        TSTB    (R4)            ;(R3)=177577=[1774][177],CC=1000
3238  010544  103402                        BCS     TSTB1
3239  010546  102401                        BVS     TSTB1
3240  010550  100401                        BMI     .+4
3241  010552  104000              TSTB1:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3242
3243  010554  105014                        CLRB    (R4)            ;(R3)=000177=[0000][177],CC=0100
3244  010556  001401                        BEQ     .+4
3245  010560  104000                        HLT
3246  010562  106313                        ASLB    (R3)            ;(R3)=000376 ,CC=1010
3247  010564  103402                        BCS     ASLB1A
3248  010566  102001                        BVC     ASLB1A
3249  010570  100401                        BMI     .+4
3250  010572  104000              ASLB1A:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3251
3252  010574  105113                        COMB    (R3)            ;(R3)=000001,CC=0001
3253  010576  103002                        BCC     COMB1A
3254  010600  102401                        BVS     COMB1A
3255  010602  100001                        BPL     .+4
3256  010604  104000              COMB1A:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3257
3258  010606  000313                        SWAB    (R3)            ;(R3)=000400, CC=0100
3259  010610  001401                        BEQ     .+4
3260  010612  104000                        HLT
3261
3262  010614  105213                        INCB    (R3)
3263  010616  000261                        SEC
3264  010620  105613                        SBCB    (R3)            ;(R3)=000400,CC=0100
3265  010622  001401                        BEQ     .+4
3266  010624  104000                        HLT
3267  010626  022713  000400               CMP     #400,(R3)       ;CHECK REMAINING RESULT
3268  010632  001401                        BEQ     .+4
3269  010634  104000                        HLT
3270                          ;;*****************************************************************
3271                          ;*TEST 7       CHECK UNIARY WORD OPS USING ADDRESS MODES 2 & 4
3272                          ;;*****************************************************************
3273  010636  112737  000007  001202 TST7:   MOVB    #7,@#STSTNM             ;LOAD TEST NUMBER
3274  010644  000004                        SCOPE
3275  010646  000401                        BR      .+4
3276  010650  000000                        .WORD   0               ;ADDRESS RESERVED FOR TESTS
3277  010652  010704                        MOV     PC,R4
3278  010654  162704  000004               SUB     #4,R4           ;R4 AND R5 POINT TO
3279  010660  010405                        MOV     R4,R5           ;RESERVED WORD
3280  010662  005015                        CLR     (R5)            ;PRESET DATA=0
3281
```

```
3282  010664  000277                        SCC
3283  010666  000244                        CLZ
3284  010670  005725                        TST     (R5)+           ;(R5)=000000,CC=0100
3285  010672  103402                        BCS     TEST2
3286  010674  102401                        BVS     TEST2
3287  010676  001401                        BEQ     .+4
3288  010700  104000              TEST2:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3289
3290  010702  005145                        COM     -(R5)           ;(R5)=177777,CC=1001
3291  010704  103001                        BCC     COM4
3292  010706  100401                        BMI     .+4
3293  010710  104000              COM4:     HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3294
3295  010712  000241                        CLC
3296  010714  006024                        ROR     (R4)+           ;(R4)=077777,CC=0011
3297  010716  103002                        BCC     ROR2
3298  010720  102001                        BVC     ROR2
3299  010722  100001                        BPL     .+4
3300  010724  104000              ROR2:     HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3301
3302  010726  000257                        CCC
3303  010730  005244                        INC     -(R4)           ;(R4)=100000,CC=1010
3304  010732  102002                        BVC     INC4
3305  010734  001401                        BEQ     INC4
3306  010736  100401                        BMI     .+4
3307  010740  104000              INC4:     HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3308
3309  010742  000261                        SEC
3310  010744  000324                        SWAB    (R4)+           ;(R4)=000200,CC=1000
3311  010746  103401                        BCS     SWAB2
3312  010750  100401                        BMI     .+4
3313  010752  104000              SWAB2:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3314
3315  010754  005425                        NEG     (R5)+           ;(R5)=177600,CC=1001
3316  010756  103001                        BCC     NEG2
3317  010760  100401                        BMI     .+4
3318  010762  104000              NEG2:     HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3319
3320  010764  005044                        CLR     -(R4)           ;(R4)=000000,CC=0100
3321  010766  001401                        BEQ     .+4
3322  010770  104000                        HLT
3323
3324  010772  000261                        SEC
3325  010774  006045                        ROR     -(R5)           ;(R5)=100000,CC=1010
3326  010776  000261                        SEC
3327  011000  005525                        ADC     (R5)+           ;(R5)=100001,CC=1000
3328  011002  102401                        BVS     ADC2
3329  011004  100401                        BMI     .+4
3330  011006  104000              ADC2:     HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3331
3332  011010  000262                        SEV
3333  011012  006224                        ASR     (R4)+           ;(R4)=140000,CC=1001
3334  011014  103002                        BCC     ASR2
3335  011016  102401                        BVS     ASR2
3336  011020  100401                        BMI     .+4
3337  011022  104000              ASR2:     HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
```

```
3338
3339  011024  000262                          SEV
3340  011026  006144                          ROL     -(R4)           ;(R4)=100001, CC=1001
3341  011030  103002                          BCC     ROL4
3342  011032  102401                          BVS     ROL4
3343  011034  100401                          BMI     .+4
3344  011036  104000                  ROL4:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3345
3346  011040  005645                          SBC     -(R5)           ;(R5)=100000,CC=1000
3347  011042  103001                          BCC     .+4
3348  011044  104000                          HLT                     ;ERROR! 'C' BIT FAILED TO CLEAR
3349
3350  011046  005325                          DEC     (R5)+           ;(R5)=077777,CC=0010
3351  011050  103402                          BCS     DEC2
3352  011052  102001                          BVC     DEC2
3353  011054  100001                          BPL     .+4
3354  011056  104000                  DEC2:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3355
3356  011060  006324                          ASL     (R4)+           ;(R4)=177776,CC=1010
3357  011062  102401                          BVS     .+4
3358  011064  104000                          HLT
3359  011066  006344                          ASL     -(R4)           ;(R4)=177774,CC=1001
3360  011070  103003                          BCC     ASL4
3361  011072  102402                          BVS     ASL4
3362  011074  001401                          BEQ     ASL4
3363  011076  100401                          BMI     .+4
3364  011100  104000                  ASL4:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3365
3366  011102  022724  177774                CMP     #177774,(R4)+
3367  011106  001401                          BEQ     .+4
3368  011110  104000                          HLT
3369  011112  020405                          CMP     R4,R5
3370  011114  001401                          BEQ     .+4
3371  011116  104000                          HLT
3372                                  ;;**********************************************************************
3373                                  ;*TEST 10        CHECK UNIARY BYTE OPS USING ADDRESS MODES 2 & 4
3374                                  ;;**********************************************************************
3375  011120  112737  000010  001202  TST10:  MOVB    #10,@#$TSTNM            ;LOAD TEST NUMBER
3376  011126  000004                          SCOPE
3377  011130  000401                          BR      .+4             ;RESERVE A WORD
3378  011132  000000                          .WORD   0               ;RESERVED WORD
3379  011134  010705                          MOV     PC,R5
3380  011136  162705  000004                  SUB     #4,R5           ;R5 POINTS TO EVEN BYTE OF RESERVED WORD
3381  011142  010500                          MOV     R5,R0
3382  011144  010002                          MOV     R0,R2
3383  011146  005202                          INC     R2              ;R2 POINTS TO ODD BYTE OF RESERVED WORD
3384  011150  005010                          CLR     (R0)            ;PRESET
3385
3386  011152  000277                          SCC
3387  011154  000241                          CLC
3388  011156  105125                          COMB    (R5)+           ;(R0)=000377,CC=1001
3389  011160  103002                          BCC     COMB2
3390  011162  102401                          BVS     COMB2
3391  011164  100401                          BMI     .+4
3392  011166  104000                  COMB2:  HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3393
```

```
3394  011170  105542                        ADCB    -(R2)           ;(R0)=000000,CC=0101
3395  011172  001401                        BEQ     .+4
3396  011174  104000                        HLT                     ;ERROR! INCORRECT RESULT AS SHOWN ABOVE
3397  011176  105525                        ADCB    (R5)+           ;(R0)=000400,CC=0000
3398  011200  103401                        BCS     ADCB2
3399  011202  001001                        BNE     .+4
3400  011204  104000              ADCB2:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3401
3402  011206  000263                        +SEC!SEV
3403  011210  106045                        RORB    -(R5)           ;(R0)=100000,CC=1001
3404  011212  103003                        BCC     RORB4
3405  011214  102402                        BVS     RORB4
3406  011216  001401                        BEQ     RORB4
3407  011220  100401                        BMI     .+4
3408  011222  104000              RORB4:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3409
3410  011224  000277                        SCC
3411  011226  106122                        ROLB    (R2)+           ;(R0)=100001,CC=0000
3412  011230  103403                        BCS     ROLB2
3413  011232  102402                        BVS     ROLB2
3414  011234  001401                        BEQ     ROLB2
3415  011236  100001                        BPL     .+4
3416  011240  104000              ROLB2:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3417
3418  011242  000257                        CCC
3419  011244  106225                        ASRB    (R5)+           ;(R0)=140001, CC=1010
3420  011246  103402                        BCS     ASRB2
3421  011250  102001                        BVC     ASRB2
3422  011252  100401                        BMI     .+4
3423  011254  104000              ASRB2:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3424
3425  011256  105242                        INCB    -(R2)           ;(R0)=140002,CC=0000
3426  011260  000277                        SCC
3427  011262  106222                        ASRB    (R2)+           ;(R0)=140001,CC=0000
3428  011264  103402                        BCS     ASRB2A
3429  011266  102401                        BVS     ASRB2A
3430  011270  100001                        BPL     .+4
3431  011272  104000              ASRB2A:   HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3432
3433  011274  000266                        +SEZ!SEV                ;SET Z,V
3434  011276  106345                        ASLB    -(R5)           ;(R0)=100001,CC=1001
3435  011300  103003                        BCC     ASLB4
3436  011302  102402                        BVS     ASLB4
3437  011304  001401                        BEQ     ASLB4
3438  011306  100401                        BMI     .+4
3439  011310  104000              ASLB4:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3440
3441  011312  105322                        DECB    (R2)+           ;(R0)=077401=[0774][001] ,CC=0010
3442  011314  103002                        BCC     DECB2
3443  011316  102001                        BVC     DECB2
3444  011320  100001                        BPL     .+4
3445  011322  104000              DECB2:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWNABOVE
3446
3447  011324  105645                        SBCB    -(R5)           ;(R0)=077400, CC=0100
3448  011326  103402                        BCS     SBCB4
3449  011330  102401                        BVS     SBCB4
```

```
3450  011332  001401                        BEQ     .+4
3451  011334  104000              SBCB4:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3452
3453  011336  105442                        NEGB    -(R2)           ;(R0)=10400,CC=1001
3454  011340  103002                        BCC     NEGB4
3455  011342  102401                        BVS     NEGB4
3456  011344  100401                        BMI     .+4
3457  011346  104000              NEGB4:    HLT                     ;ERROR! INCORRECT CC'S AS SHOWN ABOVE
3458
3459  011350  105725                        TSTB    (R5)+           ;(R0)=100400,CC=0100
3460  011352  10340'                        BCS     TSTB2
3461  011354  001401                        BEQ     .+4
3462  011356  104000              TSTB2:    HLT
3463
3464  011360  105722                        TSTB    (R2)+           ;(R0)=100400,CC=1000
3465  011362  001401                        BEQ     TSTB2A
3466  011364  100401                        BMI     .+4
3467  011366  104000              TSTB2A:   HLT
3468
3469  011370  000261                        SEC
3470  011372  000342                        SWAB    -(R2)           ;(R0)=000201,CC=1000
3471  011374  103401                        BCS     SWAB4
3472  011376  100401                        BMI     .+4
3473  011400  104000              SWAB4:    HLT
3474
3475  011402  000277                        SCC
3476  011404  105225                        INCB    (R5)+           ;(R0)=000601=[0004][201],CC=0000
3477  011406  103003                        BCC     INCB2
3478  011410  102402                        BVS     INCB2
3479  011412  001401                        BEQ     INCB2
3480  011414  100001                        BPL     .+4
3481  011416  104000              INCB2:    HLT
3482
3483  011420  022227  000601              CMP     (R2)+,#000601   ;CHECK END RESULT
3484  011424  001401                        BEQ     .+4
3485  011426  104000                        HLT
3486  011430  020205                        CMP     R2,R5           ;CHECK REGISTERS
3487  011432  001401                        BEQ     .+4
3488  011434  104000                        HLT
3489                              ;;=================================================================
3490                              ;*TEST 11       CHECK UNIARY WORD OPS USING ADDRESS MODES 3 & 5
3491                              ;;=================================================================
3492  011436  112737  000011  001202  TST11:  MOVB    #11,@#$TSTNM            ;LOAD TEST NUMBER
3493  011444  000004                        SCOPE
3494  011446  000402                        BR      .+6             ;RESERVE 2 WORDS
3495  011450  000000                        .WORD   0               ;1 FOR THE ADDRESS
3496  011452  000000                        .WORD   0               ;AND 1 FOR DATA
3497  011454  010703                        MOV     PC,R3
3498  011456  162703  000004              SUB     #4,R3
3499  011462  005013                        CLR     (R3)            ;PRESET DATA
3500  011464  010300                        MOV     R3,R0           ;R0 POINTS TO DATA WORD
3501  011466  005743                        TST     -(R3)
3502  011470  010013                        MOV     R0,(R3)
3503  011472  010304                        MOV     R3,R4
3504
3505  011474  000257                        CCC
```

```
3506  011476  005733                      TST     @(R3)+          ;(R0)=000000,CC=0100
3507  011500  001401                      BEQ     .+4
3508  011502  104000                      HLT
3509
3510  011504  000261                      SEC
3511  011506  006053                      ROR     @-(R3)          ;(R0)=100000,CC=1010
3512  011510  103402                      BCS     ROR5
3513  011512  102001                      BVC     ROR5
3514  011514  100401                      BMI     .+4
3515  011516  104000              ROR5:   HLT
3516
3517  011520  000257                      CCC
3518  011522  006234                      ASR     @(R4)+          ;(R0)=140000,CC=1010
3519  011524  102001                      BVC     ASR3
3520  011526  100401                      BMI     .+4
3521  011530  104000              ASR3:   HLT
3522
3523  011532  000250                      CLN
3524  011534  006333                      ASL     @(R3)+          ;(R0)=100000,CC=1001
3525  011536  103002                      BCC     ASL3
3526  011540  102401                      BVS     ASL3
3527  011542  100401                      BMI     .+4
3528  011544  104000              ASL3:   HLT
3529
3530  011546  000277                      SCC
3531  011550  005354                      DEC     @-(R4)          ;(R0)=077777, CC=0010
3532  011552  103003                      BCC     DEC5
3533  011554  102002                      BVC     DEC5
3534  011556  001401                      BEQ     DEC5
3535  011560  100001                      BPL     .+4
3536  011562  104000              DEC5:   HLT
3537
3538  011564  005453                      NEG     @-(R3)          ;(R0)=100001, CC=1001
3539  011566  103002                      BCC     NEG5
3540  011570  102401                      BVS     NEG5
3541  011572  100401                      BMI     .+4
3542  011574  104000              NEG5:   HLT
3543
3544  011576  000262                      SEV
3545  011600  005134                      COM     @(R4)+          ;(R0)=077776, CC=0001
3546  011602  103001                      BCC     COM3
3547  011604  102001                      BVC     .+4
3548  011606  104000              COM3:   HLT
3549
3550  011610  005233                      INC     @(R3)+          ;(R0)=077777, CC=0001
3551  011612  103001                      BCC     INC3
3552  011614  100001                      BPL     .+4
3553  011616  104000              INC3:   HLT
3554
3555  011620  005554                      ADC     @-(R4)          ;(R0)=100000, CC=1010
3556  011622  103402                      BCS     ADC5
3557  011624  102001                      BVC     ADC5
3558  011626  100401                      BMI     .+4
3559  011630  104000              ADC5:   HLT
3560
3561  011632  000257                      CCC
```

```
3562  011634  006134                        ROL     @(R4)+          ;(R0)=000000,CC=0111
3563  011636  103002                        BCC     ROL3
3564  011640  102001                        BVC     ROL3
3565  011642  001401                        BEQ     .+4
3566  011644  104000                ROL3:   HLT
3567
3568  011646  005253                        INC     @-(R3)          ;(R0)=000001, CC=0001
3569  011650  005654                        SBC     @-(R4)          ;(R0)=000000, CC=0100
3570  011652  103401                        BCS     SBC5
3571  011654  001401                        BEQ     .+4
3572  011656  104000                SBC5:   HLT
3573                                ;;*****************************************************************
3574                                ;*TEST 12       CHECK UNIARY BYTE OPS USING ADDRESS MODES 3 & 5
3575                                ;;*****************************************************************
3576  011660  112737  000012  001202 TST12:  MOVB    #12,@#$TSTNM            ;LOAD TEST NUMBER
3577  011666  000004                        SCOPE
3578  011670  000403                        BR      .+10            ;RESERVE 3 WORDS
3579  011672  000000                        .WORD   0               ;1 FOR EVEN BYTE ADDRESS
3580  011674  000000                        .WORD   0               ;1 FOR ODD BYTE ADDRESS
3581  011676  000000                        .WORD   0               ;AND 1 FOR DATA
3582  011700  010702                        MOV     PC,R2
3583  011702  005742                        TST     -(R2)           ;BACK R2 UP TO
3584  011704  005742                        TST     -(R2)           ;DATA WORD
3585  011706  010200                        MOV     R2,R0           ;R0 POINTS TO THE DATA WORD
3586  011710  005010                        CLR     (R0)            ;PRESET DATA
3587  011712  005742                        TST     -(R2)           ;BACK R2 UP TO
3588  011714  005742                        TST     -(R2)           ;EVEN BYTE ADDRESS WORD
3589  011716  010022                        MOV     R0,(R2)+        ;LOAD ADDRESS
3590  011720  005200                        INC     R0              ;ODD BYTE ADDRESS
3591  011722  010022                        MOV     R0,(R2)+        ;LOAD ODD BYTE ADDRESS
3592  011724  010200                        MOV     R2,R0           ;RESET R0
3593  011726  010205                        MOV     R2,R5
3594  011730  105152                        COMB    @-(R2)          ;(R0)=177400,CC=1001
3595  011732  103001                        BCC     COMB5
3596  011734  100401                        BMI     .+4
3597  011736  104000                COMB5:  HLT
3598  011740  105752                        TSTB    @-(R2)          ;(R0)=177400, CC=0100
3599  011742  001401                        BEQ     .+4
3600  011744  104000                        HLT
3601  011746  000262                        SEV
3602  011750  106255                        ASRB    @-(R5)          ;(R0)=177400, CC=1001
3603  011752  103002                        BCC     ASRB5
3604  011754  102401                        BVS     ASRB5
3605  011756  100401                        BMI     .+4
3606  011760  104000                ASRB5:  HLT
3607
3608  011762  105232                        INCB    @(R2)+          ;(R0)=177401, CC=000
3609  011764  103001                        BCC     INCB3
3610  011766  100001                        BPL     .+4
3611  011770  104000                INCB3:  HLT
3612
3613  011772  000241                        CLC
3614  011774  106055                        RORB    @-(R5)          ;(R0)=177400, CC=0111
3615  011776  103003                        BCC     RORB5
3616  012000  102002                        BVC     RORB5
3617  012002  001001                        BNE     RORB5
```

```
3618  012004  100001                        BPL     .+4
3619  012006  104000              RORB5:    HLT
3620
3621  012010  106332                        ASLB    @(R2)+          ;(R0)=177000, CC=1001
3622  012012  103002                        BCC     ASLB3
3623  012014  102401                        BVS     ASLB3
3624  012016  100401                        BMI     .+4
3625  012020  104000              ASLB3:    HLT
3626
3627  012022  105552                        ADCB    @-(R2)          ;(R0)=177400, CC=1000
3628  012024  103401                        BCS     ADCB5
3629  012026  100401                        BMI     .+4
3630  012030  104000              ADCB5:    HLT
3631
3632  012032  000277                        SCC
3633  012034  106135                        ROLB    @(R5)+          ;(R0)=177401, CC=0000
3634  012036  101402                        BLOS    ROLB3           ;BRANCH IF C OR Z IS SET
3635  012040  102401                        BVS     ROLB3
3636  012042  100001                        BPL     .+4
3637  012044  104000              ROLB3:    HLT
3638
3639  012046  000352                        SWAB    @-(R2)          ;(R0)=000777, CC=1000
3640  012050  100401                        BMI     .+4
3641  012052  104000                        HLT
3642
3643  012054  000261                        SEC
3644  012056  105635                        SBCB    @(R5)+          ;(R0)=000377, CC=0100
3645  012060  103401                        BCS     SBCB3
3646  012062  001401                        BEQ     .+4
3647  012064  104000              SBCB3:    HLT
3648
3649  012066  105432                        NEGB    @(R2)+          ;(R0)=000001
3650  012070  105352                        DECB    @-(R2)          ;(R0)=000000, CC=0101
3651  012072  103001                        BCC     DECB5
3652  012074  001401                        BEQ     .+4
3653  012076  104000              DECB5:    HLT
3654                              ;;*****************************************************************
3655                              ;*TEST 13       CHECK UNIARY WORD OPS USING ADDRESS MODE 6 (PC)
3656                              ;;*****************************************************************
3657  012100  112737  000013  001202  TST13:  MOVB    #13,@#$TSTNM            ;LOAD TEST NUMBER
3658  012106  000004                        SCOPE
3659  012110  005027                        CLR     (PC)+           ;PRESET DATA = 0
3660  012112  000000              UWM6:     .WORD   0               ;RESERVED FOR DATA
3661  012114  010700                        MOV     PC,R0
3662  012116  024040                        CMP     -(R0),-(R0)     ;R0 POINTS TO DATA WORD
3663  012120  000277                        SCC
3664  012122  006167  177764                ROL     UWM6            ;(R0)=000001,CC=0000
3665  012126  103403                        BCS     ROL6
3666  012130  102402                        BVS     ROL6
3667  012132  001401                        BEQ     ROL6
3668  012134  100001                        BPL     .+4
3669  012136  104000              ROL6:     HLT
3670
3671  012140  005167  177746                COM     UWM6            ;(R0)=177776, CC=1001
3672  012144  103002                        BCC     COM6
3673  012146  102401                        BVS     COM6
```

```
3674  012150  100401                    BMI     .+4
3675  012152  104000              COM6: HLT
3676  012154  006267  177732            ASR     UWM6            ;(R0)=177777, CC=1010
3677  012160  103402                    BCS     ASR6
3678  012162  102001                    BVC     ASR6
3679  012164  100401                    BMI     .+4
3680  012166  104000              ASR6: HLT
3681
3682  012170  000277                    SCC
3683  012172  005467  177714            NEG     UWM6            ;(R0)=000001, CC=0001
3684  012176  103003                    BCC     NEG6
3685  012200  102402                    BVS     NEG6
3686  012202  001401                    BEQ     NEG6
3687  012204  100001                    BPL     .+4
3688  012206  104000              NEG6: HLT
3689
3690  012210  000277                    SCC
3691  012212  006067  177674            ROR     UWM6            ;(R0)=100000, CC=1001
3692  012216  103003                    BCC     ROR6
3693  012220  102402                    BVS     ROR6
3694  012222  001401                    BEQ     ROR6
3695  012224  100401                    BMI     .+4
3696  012226  104000              ROR6: HLT
3697
3698  012230  005667  177656            SBC     UWM6            ;(R0)=077777, CC=0010
3699  012234  103402                    BCS     SBC6
3700  012236  102001                    BVC     SBC6
3701  012240  100001                    BPL     .+4
3702  012242  104000              SBC6: HLT
3703
3704  012244  000242                    CLV
3705  012246  005267  177640            INC     UWM6            ;(R0)=100000, CC=1011
3706  012252  103403                    BCS     INC6
3707  012254  102002                    BVC     INC6
3708  012256  001401                    BEQ     INC6
3709  012260  100401                    BMI     .+4
3710  012262  104000              INC6: HLT
3711
3712  012264  006267  177622            ASR     UWM6            ;(R0)=140000, CC=1010
3713  012270  000261                    SEC
3714  012272  006367  177614            ASL     UWM6            ;(R0)=100000, CC=1001
3715  012276  103002                    BCC     ASL6
3716  012300  102401                    BVS     ASL6
3717  012302  100401                    BMI     .+4
3718  012304  104000              ASL6: HLT
3719
3720  012306  005367  177600            DEC     UWM6            ;(R0)=077777, CC=0011
3721  012312  103002                    BCC     DEC6
3722  012314  102001                    BVC     DEC6
3723  012316  100001                    BPL     .+4
3724  012320  104000              DEC6: HLT
3725
3726  012322  005567  177564            ADC     UWM6            ;(R0)=100000, CC=1010
3727  012326  103402                    BCS     ADC6
3728  012330  102001                    BVC     ADC6
3729  012332  100401                    BMI     .+4
```

```
3730  012334  104000                ADC6:   HLT
3731  012336  000242                        CLV
3732  012340  000367  177546              SWAB    UWM6
3733  012344  100401                        BMI     .+4
3734  012346  104000                        HLT
3735  012350  022710  000200              CMP     #200,(R0)
3736  012354  001401                        BEQ     .+4
3737  012356  104000                        HLT
3738                                ;;*****************************************************************
3739                                ;*TEST 14        CHECK UNIARY BYTE OPS (EVEN/ODD) USING ADDRESS MODE 6 (PC)
3740                                ;;*****************************************************************
3741  012360  112737  000014  001202 TST14:  MOVB    #14,@#$TSTNM            ;LOAD TEST NUMBER
3742  012366  000004                        SCOPE
3743  012370  012700  012732              MOV     #UBM6,R0
3744  012374  063700  001506              ADD     @#FACTOR,R0     ;R0 POINTS TO ADDRESS OF DATA
3745  012400  005067  000326              CLR     UBM6            ;CLEAR DATA
3746  012404  000277                        SCC
3747  012406  000244                        CLZ
3748  012410  105767  000316              TSTB    UBM6
3749  012414  103403                        BCS     TSTB6
3750  012416  102402                        BVS     TSTB6
3751  012420  001001                        BNE     TSTB6
3752  012422  100001                        BPL     .+4
3753  012424  104000                TSTB6:  HLT
3754
3755  012426  000257                        CCC
3756  012430  105767  000277              TSTB    UBM6+1          ;TEST ODD BYTE
3757  012434  001401                        BEQ     .+4
3758  012436  104000                        HLT
3759
3760  012440  105667  000266              SBCB    UBM6            ;(R0)=000000, CC=0100
3761  012444  103402                        BCS     SBCB6
3762  012446  102401                        BVS     SBCB6
3763  012450  001401                        BEQ     .+4
3764  012452  104000                SBCB6:  HLT
3765
3766  012454  000261                1$:     SEC
3767  012456  105267  000250              INCB    UBM6            ;LOOP UNTIL (R0)=077600, CC=1011
3768  012462  100403                        BMI     2$
3769  012464  105567  000243              ADCB    UBM6+1          ;INCB INST INCREMENTS EVEN BYTE
3770  012470  000771                        BR      1$              ;ADCB INCREMENTS ODD BYTE
3771  012472  103001                2$:     BCC     INCB6
3772  012474  102401                        BVS     .+4
3773  012476  104000                INCB6:  HLT
3774
3775  012500  106367  000226              ASLB    UBM6            ;(R0)=077400, CC=0111
3776  012504  103003                        BCC     ASLB6
3777  012506  102002                        BVC     ASLB6
3778  012510  001001                        BNE     ASLB6
3779  012512  100001                        BPL     .+4
3780  012514  104000                ASLB6:  HLT
3781
3782  012516  000242                        CLV
3783  012520  105567  000207              ADCB    UBM6+1          ;(R0)=100000, CC=1010
3784  012524  103402                        BCS     ADCB6
3785  012526  102001                        BVC     ADCB6
```

```
3786  012530  100401                        BMI     .+4
3787  012532  104000              ADCB6:    HLT
3788
3789  012534  000261                        SEC
3790  012536  106067  000171                RORB    UBM6+1          ;(R0)=140000, CC=1010
3791  012542  103402                        BCS     RORB6
3792  012544  102001                        BVC     RORB6
3793  012546  100401                        BMI     .+4
3794  012550  104000              RORB6:    HLT
3795
3796  012552  105167  000154                COMB    UBM6            ;(R0)=140377 CC=1001
3797  012556  103002                        BCC     COMB6
3798  012560  102401                        BVS     COMB6
3799  012562  100401                        BMI     .+4
3800  012564  104000              COMB6:    HLT
3801
3802  012566  000262                        SEV
3803  012570  105467  000137                NEGB    UBM6+1          ;(R0)=040377, CC=0001
3804  012574  103002                        BCC     NEGB6
3805  012576  102401                        BVS     NEGB6
3806  012600  100001                        BPL     .+4
3807  012602  104000              NEGB6:    HLT
3808
3809  012604  106167  000123                ROLB    UBM6+1          ;(R0)=100777, CC=1010
3810  012610  103402                        BCS     ROLB6
3811  012612  102001                        BVC     ROLB6
3812  012614  100401                        BMI     .+4
3813  012616  104000              ROLB6:    HLT
3814
3815  012620  106267  000106                ASRB    UBM6            ;(R0)=100777, CC=1001
3816  012624  103002                        BCC     ASRB6
3817  012626  102401                        BVS     ASRB6
3818  012630  100401                        BMI     .+4
3819  012632  104000              ASRB6:    HLT
3820
3821  012634  105267  000072                INCB    UBM6            ;(R0)=100400, CC=0101
3822  012640  103002                        BCC     INCB6A
3823  012642  102401                        BVS     INCB6A
3824  012644  001401                        BEQ     .+4
3825  012646  104000              INCB6A:   HLT
3826
3827  012650  105367  000057                DECB    UBM6+1          ;(R0)=100000, CC=1001
3828  012654  103003                        BCC     DECB6A
3829  012656  102402                        BVS     DECB6A
3830  012660  001401                        BEQ     DECB6A
3831  012662  100401                        BMI     .+4
3832  012664  104000              DECB6A:   HLT
3833
3834  012666  000367  000040                SWAB    UBM6            ;(R0)=000200, CC=1000
3835  012672  103401                        BCS     SWAB6
3836  012674  100401                        BMI     .+4
3837  012676  104000              SWAB6:    HLT
3838
3839  012700  106167  000026                ROLB    UBM6            ;(R0)=000000, CC=0111
3840  012704  103002                        BCC     ROLB6A
3841  012706  102001                        BVC     ROLB6A
```

```
3842  012710  001401                        BEQ     .+4
3843  012712  104000              ROLB6A:   HLT
3844
3845  012714  005767  000012                TST     UBM6            ;(R0)=000000, CC=0100
3846  012720  103402                        BCS     TEST6
3847  012722  102401                        BVS     TEST6
3848  012724  001401                        BEQ     .+4
3849  012726  104000              TEST6:    HLT
3850
3851  012730  000401                        BR      .+4             ;RESERVE A WORD
3852  012732  000000              UBM6:     .WORD   0               ;WORD RESERVED FOR DATA
3853  012734  000004              RELE1:    SCOPE
3854  012736  010702                        MOV     PC,R2
3855  012740  062702  000012                ADD     #12,R2
3856  012744  012707  034240                MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
3857  012750  000000              REL11:    .WORD   0
3858                                ;111111111111 LAST ADDRESS OF CODE TO BE RELOCATED 11111111111
3859
3860                                ;;*****************************************************************
3861                                ;*TEST 15        CHECK UNIARY WORD OPS USING ADDRESS MODE 7
3862                                ;;*****************************************************************
3863  012752  112737  000015  001202 TST15:   MOVB    #15,@#$TSTNM            ;LOAD TEST NUMBER
3864  012760  012767  000001  166330        MOV     #1,$TIMES       ;;DO 1 ITERATION
3865  012766  000004                        SCOPE
3866
3867                                        .SBTTL  START OF SECTION 2
3868                                ;2222222222222 FIRST ADDRESS TO BE RELOCATED 222222222
3869  012770  010700              REL2:     MOV     PC,R0           ;GET PC
3870  012772  005740                        TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL2
3871  012774  010037  001512                MOV     R0,@#FRSTAD     ;SAVE
3872  013000  010700                        MOV     PC,R0           ;GET CURRENT PC
3873  013002  162700  013002                SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
3874  013006  010037  001506                MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
3875  013012  010737  001212                MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
3876  013016  062737  000030  001212        ADD     #30,@#$LPERR    ;ADJUST
3877  013024  013737  001212  001210        MOV     @#$LPERR,@#$LPADR
3878  013032  105737  001502                TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
3879  013036  001402                        BEQ     .+6
3880  013040  000167  004060                JMP     RELE2
3881  013044  000403                        BR      UW7             ;RESERVE 3 WORDS FOR ADDRESSES & DATA
3882  013046  000000                        .WORD   0               ;CONTAINS ADDRESS OF UWM7
3883  013050  000000              UWM7:     .WORD   0               ;CONTAINS DATA
3884  013052  000000                        .WORD   0               ;CONTAINS ADDRESS OF UWM7
3885
3886  013054  010700              UW7:      MOV     PC,R0
3887  013056  005740                        TST     -(R0)
3888  013060  005740                        TST     -(R0)
3889  013062  005040                        CLR     -(R0)           ;CLEAR TEST DATA
3890  013064  010002                        MOV     R0,R2
3891  013066  010240                        MOV     R2,-(R0)        ;SET UP ADDRESS
3892  013070  005720                        TST     (R0)+           ;MOVE R0 TO NEXT ADDRESS
3893  013072  005720                        TST     (R0)+
3894  013074  010210                        MOV     R2,(R0)         ;SET NEXT ADDRESS
3895  013076  010200                        MOV     R2,R0           ;SET R0 POINTING TO DATA
3896  013100  000277                        SCC
3897  013102  000244                        CLZ
```

```
3898  013104  005772  000002          TST     @2(2)           ;(R0)=000000, CC=0100
3899  013110  001401                  BEQ     .+4
3900  013112  104000                  HLT
3901
3902  013114  000277                  SCC
3903  013116  005672  177776          SBC     @-2(2)          ;(R0)=177777, CC=1001
3904  013122  103002                  BCC     SBC7
3905  013124  102401                  BVS     SBC7
3906  013126  100401                  BMI     .+4
3907  013130  104000          SBC7:   HLT
3908
3909  013132  000277                  SCC
3910  013134  000241                  CLC
3911  013136  006372  000002          ASL     @2(2)           ;(R0)=177776, CC=1001
3912  013142  103002                  BCC     ASL7
3913  013144  102401                  BVS     ASL7
3914  013146  100401                  BMI     .+4
3915  013150  104000          ASL7:   HLT
3916
3917  013152  000257                  CCC
3918  013154  005372  000002          DEC     @2(2)           ;(R0)=177775, CC=1000
3919  013160  103402                  BCS     DEC7
3920  013162  102401                  BVS     DEC7
3921  013164  100401                  BMI     .+4
3922  013166  104000          DEC7:   HLT
3923
3924  013170  000262                  SEV
3925  013172  006272  177776          ASR     @-2(2)          ;(R0)=177776, CC=1001
3926  013176  103002                  BCC     ASR7
3927  013200  102401                  BVS     ASR7
3928  013202  100401                  BMI     .+4
3929  013204  104000          ASR7:   HLT
3930
3931  013206  000241                  CLC
3932  013210  000262                  SEV
3933  013212  006072  177776          ROR     @-2(2)          ;(R0)=077777, CC=0000
3934  013216  101402                  BLOS    ROR7            ;BRANCH IF C'OR Z IS SET
3935  013220  102401                  BVS     ROR7
3936  013222  100001                  BPL     .+4
3937  013224  104000          ROR7:   HLT
3938
3939  013226  000262                  SEV
3940  013230  005472  000002          NEG     @2(2)           ;(R0)=100001, CC=1001
3941  013234  103002                  BCC     NEG7
3942  013236  102401                  BVS     NEG7
3943  013240  100401                  BMI     .+4
3944  013242  104000          NEG7:   HLT
3945
3946  013244  000250                  CLN
3947  013246  000372  177776          SWAB    @-2(2)          ;(R0)=000600, CC=1000
3948  013252  103401                  BCS     SWAB7
3949  013254  100401                  BMI     .+4
3950  013256  104000          SWAB7:  HLT
3951
3952  013260  000262                  SEV
3953  013262  005172  000002          COM     @2(2)           ;(R0)=177177, CC=1001
```

```
3954  013266  103002                        BCC     COM7
3955  013270  102401                        BVS     COM7
3956  013272  100401                        BMI     .+4
3957  013274  104000                COM7:   HLT
3958
3959  013276  000372  000002                SWAB    @2(2)           ;(R0)=077776, CC=1000
3960  013302  100401                        BMI     .+4
3961  013304  104000                        HLT
3962
3963  013306  000277                        SCC
3964  013310  005572  177776                ADC     @-2(2)          ;(R0)=077777, CC=0000
3965  013314  103402                        BCS     ADC7
3966  013316  102401                        BVS     ADC7
3967  013320  100001                        BPL     .+4
3968  013322  104000                ADC7:   HLT
3969
3970  013324  005272  000002                INC     @2(2)           ;(R0)=100000, CC=1010
3971  013330  102001                        BVC     INC7
3972  013332  100401                        BMI     .+4
3973  013334  104000                INC7:   HLT
3974
3975  013336  000257                        CCC
3976  013340  006172  177776                ROL     @-2(2)          ;(R0)=000000, CC=0111
3977  013344  103002                        BCC     ROL7
3978  013346  102001                        BVC     ROL7
3979  013350  001401                        BEQ     .+4
3980  013352  104000                ROL7:   HLT
3981                                ;;*****************************************************************
3982                                ;*TEST 16       CHECK UNIARY BYTE OPS USING ADDRESS MODE 7
3983                                ;;*****************************************************************
3984  013354  112737  000016  001202 TST16:  MOVB    #16,@#$TSTNM            ;LOAD TEST NUMBER
3985  013362  000004                        SCOPE
3986  013364  012700  013050                MOV     #UWM7,R0
3987  013370  063700  001506                ADD     @#FACTOR,R0
3988  013374  010002                        MOV     R0,R2
3989  013376  010067  177450                MOV     R0,UWM7+2
3990  013402  005720                        TST     (R0)+
3991  013404  005210                        INC     (R0)            ;WORD FOLLOWING UWM7 CONTAINS ADDRESS
3992  013406  005740                        TST     -(R0)           ;OF ODD BYTE, R0 POINTS TO DATA WORD
3993  013410  005010                        CLR     (R0)            ;PRESET DATA
3994  013412  010067  177430                MOV     R0,UWM7-2
3995                                ;NOTE:  @2(2) REFERENCES THE ODD BYTE, AND @-2(2) REFERENCES THE EVEN BYTE.
3996
3997  013416  000263                        +SEC!SEV                ;SET C AND V
3998  013420  105672  000002                SBCB    @2(2)           ;(R0)=177400, CC=1001
3999  013424  103003                        BCC     SBCB7
4000  013426  102402                        BVS     SBCB7
4001  013430  001401                        BEQ     SBCB7
4002  013432  100401                        BMI     .+4
4003  013434  104000                SBCB7:  HLT
4004
4005  013436  000277                        SCC                     ;SET CONDITION CODES
4006  013440  105572  177776                ADCB    @-2(2)          ;(R0)=177401, CC=0000
4007  013444  103403                        BCS     ADCB7
4008  013446  102402                        BVS     ADCB7
4009  013450  001401                        BEQ     ADCB7
```

```
4010  013452  100001                        BPL     .+4
4011  013454  104000              ADCB7:    HLT
4012
4013  013456  105172  177776                COMB    @-2(2)          ;(R0)=177776, CC=1001
4014  013462  103002                        BCC     COMB7
4015  013464  102401                        BVS     COMB7
4016  013466  100401                        BMI     .+4
4017  013470  104000              COMB7:    HLT
4018
4019  013472  000241                        CLC                     ;CLEAR CARRY
4020  013474  106072  000002                RORB    @2(2)           ;(R0)=077776, CC=0011
4021  013500  103002                        BCC     RORB7
4022  013502  102001                        BVC     RORB7
4023  013504  100001                        BPL     .+4
4024  013506  104000              RORB7:    HLT
4025
4026  013510  105272  000002                INCB    @2(2)           ;(R0)=100376, CC=1011
4027  013514  103002                        BCC     INCB7
4028  013516  102001                        BVC     INCB7
4029  013520  100401                        BMI     .+4
4030  013522  104000              INCB7:    HLT
4031
4032  013524  105372  177776                DECB    @-2(2)          ;(R0)=100375, CC=1001
4033  013530  103002                        BCC     DECB7
4034  013532  102401                        BVS     DECB7
4035  013534  100401                        BMI     .+4
4036  013536  104000              DECB7:    HLT
4037
4038  013540  106372  000002                ASLB    @2(2)           ;(R0)=000375, CC=0111
4039  013544  103002                        BCC     ASLB7
4040  013546  102001                        BVC     ASLB7
4041  013550  001401                        BEQ     .+4
4042  013552  104000              ASLB7:    HLT
4043
4044  013554  000241                        CLC                     ;CLEAR CARRY
4045  013556  106272  177776                ASRB    @-2(2)          ;(R0)=000376, CC=1001
4046  013562  103002                        BCC     ASRB7
4047  013564  102401                        BVS     ASRB7
4048  013566  100401                        BMI     .+4
4049  013570  104000              ASRB7:    HLT
4050
4051  013572  105472  000002                NEGB    @2(2)           ;(R0)=000376, CC=0100
4052  013576  103402                        BCS     NEGB7
4053  013600  102401                        BVS     NEGB7
4054  013602  001401                        BEQ     .+4
4055  013604  104000              NEGB7:    HLT
4056
4057  013606  000262                        SEV
4058  013610  106172  177776                ROLB    @-2(2)          ;(R0)=00374, CC=1001
4059  013614  103002                        BCC     ROLB7
4060  013616  102401                        BVS     ROLB7
4061  013620  100401                        BMI     .+4
4062  013622  104000              ROLB7:    HLT
4063
4064  013624  105272  177776                INCB    @-2(2)          ;(R0)=000375, CC=1001
4065  013630  105272  177776                INCB    @-2(2)          ;(R0)=000376, CC=1001
```

```
 4066  013634  105572  177776                ADCB    @-2(2)          ;(R0)=000377, CC=1000
 4067  013640  105172  177776                COMB    @-2(2)          ;(R0)=000000, CC=0100
 4068  013644  001401                        BEQ     .+4
 4069  013646  104000                        HLT
 4070                                ;;*****************************************************************
 4071                                ;*TEST 17       CHECK BINARY OPS USING ADDRESS MODE 0
 4072                                ;;*****************************************************************
 4073  013650  112737  000017  001202  TST17: MOVB    #17,@#$TSTNM            ;LOAD TEST NUMBER
 4074  013656  000004                        SCOPE
 4075  013660  000277                        SCC                             ;SET CONDITION CODES
 4076  013662  010700                        MOV     PC,R0                   ;R0=PC, CC=X001
 4077  013664  103002                        BCC     MOV0
 4078  013666  102401                        BVS     MOV0
 4079  013670  001001                        BNE     .+4
 4080  013672  104000                MOV0:   HLT
 4081
 4082  013674  010002                        MOV     R0,R2                   ;R2=R0
 4083  013676  000262                        SEV                             ;SET V
 4084  013700  160002                        SUB     R0,R2                   ;R2=000000, CC=0100
 4085  013702  103402                        BCS     SUB0
 4086  013704  102401                        BVS     SUB0
 4087  013706  001401                        BEQ     .+4
 4088  013710  104000                SUB0:   HLT
 4089
 4090  013712  000244                        CLZ
 4091  013714  010203                        MOV     R2,R3                   ;R2=R3=000000, CC=0100
 4092  013716  103401                        BCS     MOV0A
 4093  013720  001401                        BEQ     .+4
 4094  013722  104000                MOV0A:  HLT
 4095
 4096  013724  000257                        CCC
 4097  013726  000272                        +SEV!SEN                        ;SET V & N
 4098  013730  020203                        CMP     R2,R3                   ;R2=R3=000000, CC=0100
 4099  013732  103403                        BCS     CMP0
 4100  013734  102402                        BVS     CMP0
 4101  013736  001001                        BNE     CMP0
 4102  013740  100001                        BPL     .+4
 4103  013742  104000                CMP0:   HLT
 4104
 4105  013744  010002                        MOV     R0,R2                   ;R0=R2
 4106  013746  010203                        MOV     R2,R3                   ;R0=R2=R3
 4107  013750  060203                        ADD     R2,R3                   ;R3=2*R0
 4108  013752  006302                        ASL     R2                      ;R2=2*R0
 4109  013754  020203                        CMP     R2,R3                   ;R2=R3=2*R0
 4110  013756  001401                        BEQ     .+4
 4111  013760  104000                        HLT                             ;ERROR! CHECK ADD INSTRUCTION
 4112
 4113                                ;THE FOLLOWING SUBTEST SHIFTS A BIT THROUGH R2 AND R5 AND DOES A
 4114                                ;BIT TEST (BIT) USING R2 AND R5.
 4115  013762  005002                        CLR     R2
 4116  013764  005202                        INC     R2
 4117  013766  000402                        BR      2$
 4118  013770  006302                1$:     ASL     R2
 4119  013772  100407                        BMI     4$
 4120  013774  010205                2$:     MOV     R2,R5
 4121  013776  000277                        SCC
```

```
4122  014000  030205                        BIT     R2,R5           ;R2=R5
4123  014002  103002                        BCC     3$
4124  014004  102401                        BVS     3$
4125  014006  001370                        BNE     1$
4126  014010  104000              3$:       HLT
4127  014012  010205              4$:       MOV     R2,R5
4128  014014  000257                        CCC
4129  014016  030205                        BIT     R2,R5
4130  014020  100401                        BMI     .+4
4131  014022  104000                        HLT
4132
4133  014024  005002                        CLR     R2
4134  014026  000277                        SCC
4135  014030  050002                        BIS     R0,R2
4136  014032  103002                        BCC     BISO
4137  014034  102401                        BVS     BISO
4138  014036  001001                        BNE     .+4
4139  014040  104000              BISO:     HLT
4140
4141  014042  010003                        MOV     R0,R3
4142  014044  000277                        SCC
4143  014046  000244                        CLZ
4144  014050  040003                        BIC     R0,R3
4145  014052  103003                        BCC     BICO
4146  014054  102402                        BVS     BICO
4147  014056  001001                        BNE     BICO
4148  014060  100001                        BPL     .+4
4149  014062  104000              BICO:     HLT
4150
4151  014064  010004                        MOV     R0,R4
4152  014066  005104                        COM     R4
4153  014070  040004                        BIC     R0,R4
4154  014072  005104                        COM     R4
4155  014074  020004                        CMP     R0,R4
4156  014076  001401                        BEQ     .+4
4157  014100  104000                        HLT
4158
4159  014102  010004                        MOV     R0,R4
4160  014104  005104                        COM     R4
4161  014106  010403                        MOV     R4,R3
4162  014110  050003                        BIS     R0,R3
4163  014112  103001                        BCC     BISOA
4164  014114  100401                        BMI     .+4
4165  014116  104000              BISOA:    HLT
4166  014120  005203                        INC     R3
4167  014122  001401                        BEQ     .+4
4168  014124  104000                        HLT
4169  014126  010304                        MOV     R3,R4           ;R3=R4=0
4170  014130  005103                        COM     R3              ;R3=177777
4171  014132  000261                        SEC                     ;SET C
4172  014134  006004                        ROR     R4              ;R4=100000
4173  014136  060304                        ADD     R3,R4           ;R3=177777,R4=077777, CC=0011
4174  014140  103003                        BCC     ADDO
4175  014142  102002                        BVC     ADDO
4176  014144  001401                        BEQ     ADDO
4177  014146  100001                        BPL     .+4
```

```
 4178  014150  104000                  ADDO:   HLT
 4179  014152  010700                          MOV     PC,R0
 4180  014154  022020                          CMP     (R0)+,(R0)+
 4181  014156  020007                          CMP     R0,PC
 4182  014160  001401                          BEQ     .+4
 4183  014162  104000                          HLT
 4184
 4185  014164  010700                          MOV     PC,R0
 4186  014166  062700  000010                  ADD     #10,R0
 4187  014172  010002                          MOV     R0,R2
 4188  014174  020700                          CMP     PC,R0
 4189  014176  001002                          BNE     CMPOA
 4190  014200  020200                          CMP     R2,R0
 4191  014202  001401                          BEQ     .+4
 4192  014204  104000                  CMPOA:  HLT
 4193                                  ;;*****************************************************************
 4194                                  ;*TEST 20       CHECK BINARY OPS USING ADDRESS MODE 1
 4195                                  ;;*****************************************************************
 4196  014206  112737  000020  001202  TST20:  MOVB    #20,@#$TSTNM            ;LOAD TEST NUMBER
 4197  014214  000004                          SCOPE
 4198  014216  000402                          BR      .+6             ;RESERVE TWO WORDS
 4199  014220  000000                          .WORD   0               ;RESERVED FOR SOURCE DATA
 4200  014222  000000                          .WORD   0               ;RESERVED FOR DESTINATION DATA
 4201  014224  010704                          MOV     PC,R4
 4202  014226  005744                          TST     -(R4)
 4203  014230  005044                          CLR     -(R4)           ;R4 POINTS TO DESTINATION DATA
 4204  014232  010403                          MOV     R4,R3
 4205  014234  005043                          CLR     -(R3)           ;R3 POINTS TO SOUCE DATA
 4206
 4207  014236  005113                          COM     (R3)            ;(R3)=177777
 4208  014240  005214                          INC     (R4)            ;(R4)=000001
 4209  014242  000262                          SEV                     ;SET V
 4210  014244  061314                          ADD     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=0101
 4211  014246  103002                          BCC     ADD1
 4212  014250  102401                          BVS     ADD1
 4213  014252  001401                          BEQ     .+4
 4214  014254  104000                  ADD1:   HLT
 4215
 4216  014256  000277                          SCC
 4217  014260  000250                          CLN
 4218  014262  021314                          CMP     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=1000
 4219  014264  103403                          BCS     CMP1
 4220  014266  102402                          BVS     CMP1
 4221  014270  001401                          BEQ     CMP1
 4222  014272  100401                          BMI     .+4
 4223  014274  104000                  CMP1:   HLT
 4224
 4225  014276  000277                          SCC
 4226  014300  000244                          CLZ
 4227  014302  031314                          BIT     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=0101
 4228  014304  103002                          BCC     BITT1
 4229  014306  102401                          BVS     BITT1
 4230  014310  001401                          BEQ     .+4
 4231  014312  104000                  BITT1:  HLT
 4232
 4233  014314  000277                          SCC
```

```
4234  014316  000245                          +CLC!CLZ
4235  014320  005114                          COM     (R4)            ;(R4)=177777
4236  014322  161314                          SUB     (R3),(R4)       ;(R3)=177777,(R4)=000000, CC=0100
4237  014324  103402                          BCS     SUB1
4238  014326  102401                          BVS     SUB1
4239  014330  001401                          BEQ     .+4
4240  014332  104000              SUB1:       HLT
4241
4242  014334  105013                          CLRB    (R3)            ;(R3)=177400
4243  014336  000313                          SWAB    (R3)            ;(R3)=000377
4244  014340  000270                          SEN
4245  014342  011314                          MOV     (R3),(R4)       ;(R3)=(R4)=000377
4246  014344  100001                          BPL     .+4
4247  014346  104000                          HLT
4248  014350  000314                          SWAB    (R4)            ;(R3)=000377,(R4)=177400
4249  014352  000263                          +SEC!SEV                ;SET C & V
4250  014354  051314                          BIS     (R3),(R4)       ;(R3)=000377,(R4)=177777, CC=1001
4251  014356  103002                          BCC     BIS1
4252  014360  102401                          BVS     BIS1
4253  014362  100401                          BMI     .+4
4254  014364  104000              BIS1:       HLT
4255
4256  014366  041314                          BIC     (R3),(R4)       ;(R3)=000377,(R4)=177400, CC=1001
4257  014370  103002                          BCC     BIC1
4258  014372  102401                          BVS     BIC1
4259  014374  100401                          BMI     .+4
4260  014376  104000              BIC1:       HLT
4261
4262  014400  000262                          SEV                     ;SET V
4263  014402  021314                          CMP     (R3),(R4)       ;(R3)=000377,(R4)=177400, CC=0001
4264  014404  103003                          BCC     CMP1A
4265  014406  102402                          BVS     CMP1A
4266  014410  001401                          BEQ     CMP1A
4267  014412  100001                          BPL     .+4
4268  014414  104000              CMP1A:      HLT
4269
4270  014416  005013                          CLR     (R3)            ;(R3)=000000
4271  014420  000261                          SEC
4272  014422  006013                          ROR     (R3)            ;(R3)=100000
4273  014424  011314                          MOV     (R3),(R4)       ;(R3)=(R4)=100000
4274  014426  005114                          COM     (R4)            ;(R4)=077777
4275  014430  161314                          SUB     (R3),(R4)       ;(R3)=100000,(R4)=177777, CC=1011
4276  014432  103002                          BCC     SUB1A
4277  014434  102001                          BVC     SUB1A
4278  014436  100401                          BMI     .+4
4279  014440  104000              SUB1A:      HLT
4280
4281  014442  000277                          SCC
4282  014444  161314                          SUB     (R3),(R4)       ;(R3)=100000,(R4)=077777, CC=0000
4283  014446  101402                          BLOS    SUB1B           ;BRANCH IF C OR Z IS SET
4284  014450  102401                          BVS     SUB1B
4285  014452  100001                          BPL     .+4
4286  014454  104000              SUB1B:      HLT
4287
4288  014456  011314                          MOV     (R3),(R4)       ;(R3)=100000,(R4)=100000, CC=1000
4289  014460  001401                          BEQ     MOV1
```

```
 4290  014462  100401                        BMI     .+4
 4291  014464  104000              MOV1:     HLT
 4292
 4293  014466  061314                        ADD     (R3),(R4)       ;(R3)=100000,(R4)=000000, CC=0111
 4294  014470  103003                        BCC     ADD1A
 4295  014472  102002                        BVC     ADD1A
 4296  014474  001001                        BNE     ADD1A
 4297  014476  100001                        BPL     .+4
 4298  014500  104000              ADD1A:    HLT
 4299
 4300  014502  005113                        COM     (R3)            ;(R3)=077777
 4301  014504  011314                        MOV     (R3),(R4)       ;(R4)=077777
 4302  014506  061314                        ADD     (R3),(R4)       ;(R3)=077777,(R4)=177776, CC=1010
 4303  014510  103402                        BCS     ADD1B
 4304  014512  102001                        BVC     ADD1B
 4305  014514  100401                        BMI     .+4
 4306  014516  104000              ADD1B:    HLT
 4307
 4308  014520  062714  000002              ADD     #2,(R4)
 4309  014524  005714                        TST     (R4)            ;CHECK FINAL RESULT
 4310  014526  001401                        BEQ     .+4
 4311  014530  104000                        HLT
 4312                              ;;*****************************************************************
 4313                              ;*TEST 21       CHECK BINARY BYTE OPS USING ADDRESS MODE 1
 4314                              ;;*****************************************************************
 4315  014532  112737  000021  001202 TST21:   MOVB    #21,@#$TSTNM            ;LOAD TEST NUMBER
 4316  014540  000004                        SCOPE
 4317  014542  000402                        BR      .+6
 4318  014544  000000                        .WORD   0
 4319  014546  000000                        .WORD   0
 4320  014550  010705                        MOV     PC,R5
 4321  014552  005745                        TST     -(R5)
 4322  014554  005045                        CLR     -(R5)           ;(R5)=000000
 4323  014556  010502                        MOV     R5,R2
 4324  014560  005042                        CLR     -(R2)           ;(R2)=000000
 4325  014562  005202                        INC     R2              ;R2 POINTS TO ODD BYTE
 4326  014564  105112                        COMB    (R2)            ;(R2)=177400
 4327
 4328  014566  000277                        SCC
 4329  014570  111215                        MOVB    (R2),(R5)       ;(R2)=177400,(R5)=000377,CC=1001
 4330  014572  103005                        BCC     MOVB1
 4331  014574  102404                        BVS     MOVB1
 4332  014576  001403                        BEQ     MOVB1
 4333  014600  100002                        BPL     MOVB1
 4334  014602  105215                        INCB    (R5)            ;CHECK RESULT
 4335  014604  001401                        BEQ     .+4
 4336  014606  104000              MOVB1:    HLT
 4337
 4338  014610  106312                        ASLB    (R2)            ;SHIFT (R2) UNTIL
 4339  014612  102376                        BVC     .-2             ;(R2)=000000
 4340  014614  106012                        RORB    (R2)            ;(R2)=100000
 4341  014616  105315                        DECB    (R5)            ;(R5)=00377
 4342  014620  106015                        RORB    (R5)            ;(R5)=000177
 4343  014622  000257                        CCC
 4344  014624  121512                        CMPB    (R5),(R2)       ;(R5)=000177,(R2)=100000, CC=1010
 4345  014626  102001                        BVC     CMPB1
```

```
 4346  014630  100401                    BMI     .+4
 4347  014632  104000            CMPB1:  HLT
 4348
 4349  014634  005003                    CLR     R3
 4350  014636  000261                    SEC
 4351  014540  006003                    ROR     R3                      ;R3=100000
 4352  014642  050315                    BIS     R3,(R5)                 ;(R5)=100177
 4353  014644  000273                    +SEC!SEV!SEN                    ;SET C,V, & N
 4354  014646  131215                    BITB    (R2),(R5)               ;(R2)=100000,(R5)=100177, CC=0101
 4355  014650  103002                    BCC     BITB1
 4356  014652  102401                    BVS     BITB1
 4357  014654  001401                    BEQ     .+4
 4358  014656  104000            BITB1:  HLT
 4359
 4360  014660  151215                    BISB    (R2),(R5)               ;(R2)=100000,(R5)=100377, CC=1001
 4361  014662  103001                    BCC     BISB1
 4362  014664  100401                    BMI     .+4
 4363  014666  104000            BISB1:  HLT
 4364
 4365  014670  141215                    BICB    (R2),(R5)               ;(R2)=100000,(R5)=100177, CC=0001
 4366  014672  103002                    BCC     BICB1
 4367  014674  001401                    BEQ     BICB1
 4368  014676  100001                    BPL     .+4
 4369  014700  104000            BICB1:  HLT
 4370
 4371  014702  105112                    COMB    (R2)                    ;(R2)=077400,(R5)=100177
 4372  014704  121215                    CMPB    (R2),(R5)
 4373  014706  001401                    BEQ     .+4
 4374  014710  104000                    HLT
 4375
 4376  014712  141512                    BICB    (R5),(R2)               ;(R5)=100177,(R2)=000000, CC=0100
 4377  014714  001002                    BNE     BICB1A
 4378  014716  105712                    TSTB    (R2)
 4379  014720  001401                    BEQ     .+4
 4380  014722  104000            BICB1A: HLT
 4381
 4382  014724  000402                    BR      .+6                     ;RESERVE TWO WORDS FOR DATA
 4383  014726  000000                    .WORD   0                       ;SOURCE DATA
 4384  014730  000000                    .WORD   0                       ;DEST DATA
 4385  014732  010705                    MOV     PC,R5
 4386  014734  005745                    TST     -(R5)
 4387  014736  105045                    CLRB    -(R5)                   ;R5 POINTS TO DEST ODD BYTE
 4388  014740  010504                    MOV     R5,R4
 4389  014742  105044                    CLRB    -(R4)                   ;R4 POINTS TO DEST EVEN BYTE
 4390  014744  010403                    MOV     R4,R3
 4391  014746  105043                    CLRB    -(R3)                   ;R3 POINTS TO SOURCE ODD BYTE
 4392  014750  010302                    MOV     R3,R2
 4393  014752  105042                    CLRB    -(R2)                   ;R2 POINTS TO SOURCE EVEN BYTE
 4394
 4395                            ;COMMENTS ARE LEAST SIGNIFICANT 4 BITS OF BYTES POINTED TO BY R2,R3
 4396                            ;R4, AND R5 RESPECTIVELY AND THE REMAINING BITS ARE 0'S.
 4397  014754  000261                    SEC                             ;SET CARRY
 4398                                                                    ;(R2),(R3),(R4),(R5)
 4399  014756  106112                    ROLB    (R2)                    ;0001,0000,0000,0000
 4400  014760  111214                    MOVB    (R2),(R4)               ;0001,0000,0001,0000
 4401  014762  106112                    ROLB    (R2)                    ;0010,0000,0001,0000
```

```
4402  014764  111213                    MOVB    (R2),(R3)       ;0010,0010,0001,0000
4403  014766  106112                    ROLB    (R2)            ;0100,0010,0001,0000
4404  014770  111315                    MOVB    (R3),(R5)       ;0100,0010,0001,0010
4405  014772  106112                    ROLB    (R2)            ;1000,0010,0001,0010
4406  014774  106113                    ROLB    (R3)            ;1000,0100,0001,0010
4407  014776  151215                    BISB    (R2),(R5)       ;1000,0100,0001,1010
4408  015000  131512                    BITB    (R5),(R2)       ;1000,0100,0001,1010
4409  015002  001426                    BEQ     BIN1
4410  015004  151314                    BISB    (R3),(R4)       ;1000,0100,0101,1010
4411  015006  131413                    BITB    (R4),(R3)       ;1000,0100,0101,1010
4412  015010  001423                    BEQ     BIN1
4413  015012  105213                    INCB    (R3)            ;1000,0101,0101,1010
4414  015014  121314                    CMPB    (R3),(R4)       ;1000,0101,0101,1010
4415  015016  001020                    BNE     BIN1
4416  015020  106113                    ROLB    (R3)            ;1000,1010,0101,1010
4417  015022  121315                    CMPB    (R3),(R5)       ;1000,1010,0101,1010
4418  015024  001015                    BNE     BIN1
4419  015026  106212                    ASRB    (R2)            ;0100,1010,0101,1010
4420  015030  131214                    BITB    (R2),(R4)       ;0100,1010,0101,1010
4421  015032  001412                    BEQ     BIN1
4422  015034  106015                    RORB    (R5)            ;0100,1010,0101,0101
4423  015036  121415                    CMPB    (R4),(R5)       ;0100,1010,0101,0101
4424  015040  001007                    BNE     BIN1
4425  015042  105314                    DECB    (R4)            ;0100,1010,0100,0101
4426  015044  141214                    BICB    (R2),(R4)       ;0100,1010,0000,0101
4427  015046  001004                    BNE     BIN1
4428  015050  111314                    MOVB    (R3),(R4)       ;0100,1010,1010,0101
4429  015052  106213                    ASRB    (R3)            ;0100,0101,1010,0101
4430  015054  141315                    BICB    (R3),(R5)       ;0100,0101,1010,0101
4431  015056  001401                    BEQ     .+4
4432  015060  104000          BIN1:     HLT
4433                          ;;*****************************************************************
4434                          ;*TEST 22       CHECK BINARY WORD OPS USING ADDRESS MODE 2 & 4
4435                          ;;*****************************************************************
4436  015062  112737  000022  001202  TST22:  MOVB    #22,@#STSTNM            ;LOAD TEST NUMBER
4437  015070  000004                    SCOPE
4438  015072  012704  014730                   MOV     #BICB1A+6,R4
4439  015076  012702  014726                   MOV     #BICB1A+4,R2
4440  015102  063702  001506                   ADD     @#FACTOR,R2
4441  015106  063704  001506                   ADD     @#FACTOR,R4
4442  015112  010405                           MOV     R4,R5           ;SET DESTINATION REGISTER
4443  015114  012715  000001                   MOV     #1,(R5)
4444  015120  012712  177777                   MOV     #-1,(R2)
4445  015124  000257                           CCC
4446  015126  000262                           SEV
4447  015130  062225                           ADD     (R2)+,(R5)+     ;(R2)=177777,(R5)=000000, CC=0101
4448  015132  103002                           BCC     ADD2
4449  015134  102401                           BVS     ADD2
4450  015136  001401                           BEQ     .+4
4451  015140  104000                ADD2:      HLT
4452
4453  015142  000262                           SEV                     ;SET V
4454  015144  024527  000001                   CMP     -(R5),#1        ;(R5)=000000, CC=1001
4455  015150  103002                           BCC     CMP2
4456  015152  102401                           BVS     CMP2
4457  015154  100401                           BMI     .+4
```

```
4458  015156  104000                  CMP2:   HLT
4459
4460  015160  054225                          BIS     -(R2),(R5)+     ;(R2)=177777,(R5)=177777,CC=1001
4461  015162  103001                          BCC     BIS2
4462  015164  100401                          BMI     .+4
4463  015166  104000                  BIS2:   HLT
4464  015170  000277                          SCC
4465  015172  000244                          CLZ
4466  015174  162245                          SUB     (R2)+,-(R5)     ;(R2)=177777,(R5)=000000, CC=0100
4467  015176  103402                          BCS     SUB2
4468  015200  102401                          BVS     SUB2
4469  015202  001401                          BEQ     .+4
4470  015204  104000                  SUB2:   HLT
4471
4472  015206  005442                          NEG     -(R2)           ;(R2)=000001
4473  015210  005115                          COM     (R5)            ;(R5)=177777
4474  015212  000277                          SCC
4475  015214  000250                          CLN
4476  015216  042225                          BIC     (R2)+,(R5)+     ;(R2)=000001,(R5)=177776, CC=1001
4477  015220  103003                          BCC     BIC2
4478  015222  102402                          BVS     BIC2
4479  015224  001401                          BEQ     BIC2
4480  015226  100401                          BMI     .+4
4481  015230  104000                  BIC2:   HLT
4482
4483  015232  012742  125252                  MOV     #125252,-(R2)
4484  015236  012245                          MOV     (R2)+,-(R5)
4485  015240  005125                          COM     (R5)+           ;(R5)=052525
4486  015242  000262                          SEV
4487  015244  034245                          BIT     -(R2),-(R5)     ;(R2)=125252,(R5)=052525, CC=0101
4488  015246  103002                          BCC     BITT2
4489  015250  102401                          BVS     BITT2
4490  015252  001401                          BEQ     .+4
4491  015254  104000                  BITT2:  HLT
4492
4493  015256  000262                          SEV
4494  015260  052225                          BIS     (R2)+,(R5)+     ;(R2)=125252,(R5)=177777, CC=1001
4495  015262  103002                          BCC     BIS2A
4496  015264  102401                          BVS     BIS2A
4497  015266  100401                          BMI     .+4
4498  015270  104000                  BIS2A:  HLT
4499
4500  015272  042745  125252                BIC     #125252,-(R5)   ;(R5)=052525
4501  015276  005125                          COM     (R5)+           ;(R5)=125252
4502  015300  024245                          CMP     -(R2),-(R5)
4503  015302  001401                          BEQ     .+4
4504  015304  104000                          HLT
4505
4506  015306  005012                          CLR     (R2)
4507  015310  005122                          COM     (R2)+           ;(R2)=177777
4508  015312  162742  000001                  SUB     #1,-(R2)        ;(R2)=177776, CC=1000
4509  015316  103402                          BCS     SUB2A
4510  015320  102401                          BVS     SUB2A
4511  015322  100401                          BMI     .+4
4512  015324  104000                  SUB2A:  HLT
4513  015326  010702                          MOV     PC,R2           ;GET CURRENT PC
```

```
 4514  015330  010205                          MOV     R2,R5           ;MOVE TO R5
 4515  015332  124245                  1$:     CMPB    -(R2),-(R5)     ;COMPARE ALL PREVIOUS MEMORY ADDRESSES
 4516  015334  001401                          BEQ     .+4
 4517  015336  104000                          HLT                     ;ERROR!
 4518  015340  020237  001512                  CMP     R2,@#FRSTAD     ;CHECK FOR LOW LIMIT
 4519  015344  001372                          BNE     1$
 4520                                  ;;*****************************************************************
 4521                                  ;*TEST 23        CHECK BINARY BYTE OPS USING ADDRESS MODE 2 & 4
 4522                                  ;;*****************************************************************
 4523  015346  112737  000023  001202  TST23:  MOVB    #23,@#STSTNM            ;LOAD TEST NUMBER
 4524  015354  000004                          SCOPE
 4525  015356  000402                          BR      .+6             ;RESERVE TWO WORDS
 4526  015360  000000                          .WORD   0               ;SOURCE DATA
 4527  015362  000000                          .WORD   0               ;DESTINATION DATA
 4528  015364  010703                          MOV     PC,R3
 4529  015366  005743                          TST     -(R3)
 4530
 4531                                  ;FIRST CHECK AUTO INCREMENT/DECREMENT
 4532  015370  010300                          MOV     R3,R0
 4533  015372  010002                          MOV     R0,R2
 4534  015374  005302                          DEC     R2
 4535  015376  010604                          MOV     SP,R4
 4536  015400  010605                          MOV     SP,R5
 4537  015402  005745                          TST     -(R5)
 4538
 4539  015404  114046                          MOVB    -(R0),-(SP)
 4540  015406  020506                          CMP     R5,SP
 4541  015410  001021                          BNE     BINB
 4542  015412  020200                          CMP     R2,R0
 4543  015414  001017                          BNE     BINB
 4544  015416  122026                          CMPB    (R0)+,(SP)+
 4545  015420  020406                          CMP     R4,SP
 4546  015422  001014                          BNE     BINB
 4547  015424  020003                          CMP     R0,R3
 4548  015426  001012                          BNE     BINB
 4549  015430  154640                          BISB    -(SP),-(R0)
 4550  015432  020506                          CMP     R5,SP
 4551  015434  001007                          BNE     BINB
 4552  015436  020200                          CMP     R2,R0
 4553  015440  001005                          BNE     BINB
 4554  015442  142620                          BICB    (SP)+,(R0)+
 4555  015444  020406                          CMP     R4,SP
 4556  015446  001002                          BNE     BINB
 4557  015450  020003                          CMP     R0,R3
 4558  015452  001401                          BEQ     .+4
 4559  015454  104000          BINB:           HLT
 4560  015456  010003                          MOV     R0,R3
 4561  015460  112743  000200                  MOVB    #200,-(R3)
 4562  015464  112743  000377                  MOVB    #377,-(R3)      ;(R3)=100377
 4563  015470  010304                          MOV     R3,R4
 4564  015472  112744  000177                  MOVB    #177,-(R4)
 4565  015476  112744  000000                  MOVB    #0,-(R4)        ;(R4)=077400
 4566  015502  001401                          BEQ     .+4
 4567  015504  104000                          HLT
 4568
 4569  015506  152324                          BISB    (R3)+,(R4)+     ;(R3)=100377,(R4)=077777
```

```
 4570  015510  100401                          BMI     .+4
 4571  015512  104000                          HLT
 4572
 4573  015514  122324                          CMPB    (R3)+,(R4)+
 4574  015516  103402                          BCS     CMPB2
 4575  015520  102001                          BVC     CMPB2
 4576  015522  100001                          BPL     .+4
 4577  015524  104000                CMPB2:    HLT
 4578
 4579  015526  000261                          SEC
 4580  015530  134344                          BITB    -(R3),-(R4)
 4581  015532  103002                          BCC     BITB2
 4582  015534  102401                          BVS     BITB2
 4583  015536  001401                          BEQ     .+4
 4584  015540  104000                BITB2:    HLT
 4585
 4586  015542  000244                          CLZ
 4587  015544  144344                          BICB    -(R3),-(R4)     ;(R3)=100377,(R4)=077400
 4588  015546  001401                          BEQ     .+4
 4589  015550  104000                          HLT
 4590                                  ;;*****************************************************************
 4591                                  ;*TEST 24       CHECK BINARY WORD OPS USING ADDRESS MODES 3 & 5
 4592                                  ;;*****************************************************************
 4593  015552  112737  000024  001202 TST24:   MOVB    #24,@#$TSTNM            ;LOAD TEST NUMBER
 4594  015560  000004                          SCOPE
 4595  015562  000404                          BR      2$              ;RESERVE SPACE FOR DATA AND ADDRESSES
 4596  015564  000000                          .WORD   0               ;CONTAINS ADDRESS OF SOURCE DATA
 4597  015566  000000                          .WORD   0               ;CONTAINS ADDRESS OF DEST DATA
 4598  015570  000000                          .WORD   0               ;CONTAINS SOURCE DATA
 4599  015572  000000                          .WORD   0               ;CONTAINS DEST DATA
 4600  015574  010701                2$:       MOV     PC,R1
 4601  015576  010100                          MOV     R1,R0           ;SET SCOPE PTR
 4602  015600  024040                          CMP     -(R0),-(R0)     ;ADJUST R0
 4603  015602  010005                          MOV     R0,R5           ;R5 POINTS TO DEST DATA
 4604  015604  024545                          CMP     -(R5),-(R5)     ;SUB 4 FROM R5
 4605  015606  010015                          MOV     R0,(R5)         ;R5 POINTS TO ADDRESS OF DEST DATA
 4606  015610  010502                          MOV     R5,R2
 4607  015612  010004                          MOV     R0,R4           ;R4 POINTS TO DEST DATA
 4608  015614  005740                          TST     -(R0)
 4609  015616  010003                          MOV     R0,R3           ;R3 POINTS TO SOURCE DATA
 4610  015620  010042                          MOV     R0,-(R2)        ;R2 POINTS TO ADDRESS OF SOURCE DATA
 4611  015622  005013                          CLR     (R3)            ;PRESET SOURCE DATA
 4612  015624  005014                          CLR     (R4)            ;PRESET DEST DATA
 4613
 4614  015626  000277                          SCC
 4615  015630  000244                          CLZ
 4616  015632  163235                          SUB     @(R2)+,@(R5)+   ;(R3)=000000,(R4)=000000, CC=0100
 4617  015634  103402                          BCS     SUB3
 4618  015636  102401                          BVS     SUB3
 4619  015640  001401                          BEQ     .+4
 4620  015642  104000                SUB3:     HLT
 4621
 4622  015644  052752  100000                  BIS     #100000,@-(R2)  ;(R3)=100000
 4623  015650  062755  000001                  ADD     #1,@-(R5)       ;(R4)=000001
 4624  015654  163235                          SUB     @(R2)+,@(R5)+   ;(R3)=100000,(R4)=100001, CC=1011
 4625  015656  103002                          BCC     SUB3A
```

```
4626  015660  102001                        BVC     SUB3A
4627  015662  100401                        BMI     .+4
4628  015664  104000              SUB3A:    HLT
4629
4630  015666  005414                        NEG     (R4)                ;(R4)=077777
4631  015670  035255                        BIT     @-(R2),@-(R5)       ;(R3)=100000,(R4)=077777
4632  015672  001401                        BEQ     .+4
4633  015674  104000                        HLT
4634  015676  023235                        CMP     @(R2)+,@(R5)+
4635  015700  102401                        BVS     .+4
4636  015702  104000                        HLT
4637  015704  005152                        COM     @-(R2)
4638  015706  000257                        CCC
4639  015710  063255                        ADD     @(R2)+,@-(R5)
4640  015712  102001                        BVC     ADD3
4641  015714  100401                        BMI     .+4
4642  015716  104000              ADD3:     HLT
4643  015720  000261                        SEC
4644  015722  045235                        BIC     @-(R2),@(R5)+       ;(R3)=077777,(R4)=100000
4645  015724  103001                        BCC     BIC3
4646  015726  100401                        BMI     .+4
4647  015730  104000              BIC3:     HLT
4648
4649  015732  005155                        COM     @-(R5)              ;(R4)=077777
4650  015734  023235                        CMP     @(R2)+,@(R5)+       ;(R3)=077777,(R4)=077777
4651  015736  001401                        BEQ     .+4
4652  015740  104000                        HLT
4653                              ;;=================================================================
4654                              ;#TEST 25       CHECK BINARY BYTE OPS USING ADDRESS MODES 3 & 5
4655                              ;;=================================================================
4656  015742  112737  000025  001202  TST25:  MOVB    #25,@#STSTNM            ;LOAD TEST NUMBER
4657  015750  000004                        SCOPE
4658  015752  000406                        BR      1$                  ;RESERVE SPACE FOR ADDRESS AND DATA
4659  015754  000000                        .WORD   0                   ;CONTAINS ADDRESS OF SOURCE DATA (EVEN BYTE)
4660  015756  000000                        .WORD   0                   ;CONTAINS ADDRESS OF SOURCE DATA (ODD BYTE)
4661  015760  000000                        .WORD   0                   ;CONTAINS ADDRESS OF DEST DATA (EVEN BYTE)
4662  015762  000000                        .WORD   0                   ;CONTAINS ADDRESS OF DEST DATA (ODD BYTE)
4663  015764  000000                        .WORD   0                   ;CONTAINS SOURCE DATA
4664  015766  000000                        .WORD   0                   ;CONTAINS DEST DATA
4665
4666  015770  010700              1$:       MOV     PC,R0
4667  015772  024040                        CMP     -(R0),-(R0)         ;R0=ADDRESS OF DEST DATA
4668  015774  010003                        MOV     R0,R3               ;R3     "       "
4669  015776  010305                        MOV     R3,R5               ;R5     "       "
4670  016000  005743                        TST     -(R3)               ;SUB 2 FROM R3
4671  016002  010043                        MOV     R0,-(R3)            ;R3 POINTS TO ADDRESS OF DEST DATA
4672  016004  005213                        INC     (R3)                ;ODD BYTE
4673  016006  010043                        MOV     R0,-(R3)            ;EVEN BYTE
4674  016010  010304                        MOV     R3,R4
4675  016012  005740                        TST     -(R0)               ;R0=ADDRESS OF SOURCE DATA
4676  016014  010044                        MOV     R0,-(R4)            ;R4 POINTS TO ADDRESS OF SOURCE DATA
4677  016016  005214                        INC     (R4)                ;ODD BYTE
4678  016020  010044                        MOV     R0,-(R4)            ;EVEN BYTE
4679
4680  016022  000261                        SEC                         ;SET CARRY
4681  016024  012734  177001                MOV     #177001,@(R4)+
```

```
4682  016030  112734  000200                    MOVB    #200,@(R4)+     ;SOURCE DATA=100001
4683  016034  115433                            MOVB    @-(R4),@(R3)+
4684  016036  115433                            MOVB    @-(R4),@(R3)+   ;DEST DATA=000600
4685  016040  103401                            BCS     .+4
4686  016042  104000                            HLT                     ;ERROR! MOV DOES AFFECT C BIT IN PSW
4687  016044  022715  000600                    CMP     #600,(R5)       ;CHECK DEST DATA
4688  016050  001401                            BEQ     .+4
4689  016052  104000                            HLT                     ;ERROR! INCORRECT RESULT
4690  016054  024343                            CMP     -(R3),-(R3)     ;POINT R4 BACK TO EVEN BYTE
4691  016056  153433                            BISB    @(R4)+,@(R3)+
4692  016060  153433                            BISB    @(R4)+,@(R3)+   ;DEST DATA=100601
4693  016062  022715  100601                    CMP     #100601,(R5)    ;CHECK RESULT
4694  016066  001401                            BEQ     .+4
4695  016070  104000                            HLT                     ;ERROR! INCORRECT DEST DATA AFTER BISB
4696  016072  145453                            BICB    @-(R4),@-(R3)
4697  016074  145453                            BICB    @-(R4),@-(R3)
4698  016076  133433                            BITB    @(R4)+,@(R3)+
4699  016100  001002                            BNE     BITB3
4700  016102  135433                            BITB    @-(R4),@(R3)+
4701  016104  001001                            BNE     .+4
4702  016106  104000                    BITB3:  HLT
4703
4704  016110  123453                            CMPB    @(R4)+,@-(R3)
4705  016112  001002                            BNE     CMPB3
4706  016114  123453                            CMPB    @(R4)+,@-(R3)
4707  016116  001401                            BEQ     .+4
4708  016120  104000                    CMPB3:  HLT
4709                                    ;;;*****************************************************************
4710                                    ;*TEST 26        CHECK BINARY OPS USING ADDRESS MODE 6
4711                                    ;;*****************************************************************
4712  016122  112737  000026  001202    TST26:  MOVB    #26,@#$TSTNM            ;LOAD TEST NUMBER
4713  016130  000004                            SCOPE
4714  016132  000402                            BR      .+6             ;RESERVE TWO LOCATIONS
4715  016134  000000                    SDATA:  .WORD   0               ;RESERVED FOR SOURCE DATA
4716  016136  000000                    DDATA:  .WORD   0               ;RESERVED FOR DESTINATION DATA
4717
4718  016140  013702  001506                    MOV     @#FACTOR,R2     ;GET RELOCATION FACTOR AND USE AS AN
4719  016144  010205                            MOV     R2,R5           ;INDEX VALUE TO POINT TO DATA
4720  016146  005065  016136                    CLR     DDATA(R5)       ;PRESET DESTINATION DATA
4721  016152  012762  000001  016134            MOV     #1,SDATA(2)     ;THIS ROUTSINE PUT A 1 BIT INTO EVERY
4722  016160  056265  016134  016136    1$:     BIS     SDATA(2),DDATA(5)       ;OTHER BIT POSITION IN THE DEST-
4723  016166  006362  016134                    ASL     SDATA(2)                ;INATION ADDRESS (52525)
4724  016172  006362  016134                    ASL     SDATA(2)
4725  016176  103370                            BCC     1$
4726  016200  022765  052525  016136            CMP     #52525,DDATA(5) ;CHECK RESULT
4727  016206  001401                            BEQ     .+4
4728  016210  104000                            HLT                     ;ERROR! INCORRECT RESULT
4729  016212  012762  177777  016134            MOV     #-1,SDATA(2)
4730  016220  046562  016136  016134            BIC     DDATA(5),SDATA(2)       ;SOURCE DATA=125252
4731  016226  036265  016134  016136            BIT     SDATA(2),DDATA(5)
4732  016234  001401                            BEQ     .+4
4733  016236  104000                            HLT                     ;ERROR! BIT INST FAILED
4734  016240  006365  016136                    ASL     DDATA(5)        ;DDATA=125252
4735  016244  026265  016134  016136            CMP     SDATA(2),DDATA(5)
4736  016252  001401                            BEQ     .+4
4737
```

```
 4738  016254  104000                          HLT                     ;ERROR! CMP INST FAILED
 4739  016256  000257                          CCC
 4740  016260  066265  016134  016136          ADD     SDATA(2),DDATA(5)
 4741  016266  103002                          BCC     ADD6
 4742  016270  102001                          BVC     ADD6
 4743  016272  100001                          BPL     .+4
 4744  016274  104000                ADD6:     HLT
 4745
 4746  016276  006362  016134                  ASL     SDATA(2)        ;SDATA=52524
 4747  016302  166265  016134  016136          SUB     SDATA(2),DDATA(5)
 4748  016310  103401                          BCS     SUB6
 4749  016312  001401                          BEQ     .+4
 4750  016314  104000                SUB6:     HLT
 4751
 4752  016316  112700  000377                  MOVB    #377,R0         ;R0=177777 (MOVB %R EXTENDS SIGN)
 4753  016322  010062  016134                  MOV     R0,SDATA(2)
 4754  016326  012765  177777  016136          MOV     #-1,DDATA(5)
 4755  016334  166500  016136                  SUB     DDATA(5),R0
 4756  016340  001401                          BEQ     .+4
 4757  016342  104000                          HLT
 4758  016344  066265  016134  016136  1$:     ADD     SDATA(2),DDATA(5)
 4759  016352  006362  016134                  ASL     SDATA(2)
 4760  016356  005162  016134                  COM     SDATA(2)
 4761  016362  036265  016134  016136          BIT     SDATA(2),DDATA(5)
 4762  016370  001401                          BEQ     .+4
 4763  016372  104000                          HLT
 4764  016374  005162  016134                  COM     SDATA(2)
 4765  016400  026265  016134  016136          CMP     SDATA(2),DDATA(5)
 4766  016406  001401                          BEQ     .+4
 4767  016410  104000                          HLT
 4768  016412  026200  016134                  CMP     SDATA(2),R0
 4769  016416  001352                          BNE     1$
 4770                                ;;*********************************************************************
 4771                                ;*TEST 27       CHECK BINARY BYTE OPS USING ADDRESS MODE 6
 4772                                ;;*********************************************************************
 4773  016420  112737  000027  001202  TST27:  MOVB    #27,@#$TSTNM            ;LOAD TEST NUMBER
 4774  016426  000004                          SCOPE
 4775                                ;NOTE: SDATAB(2), AND DDATAB(4) REFERENCE EVEN BYTE OF SOURCE & DEST DATA
 4776                                ;AND SDATAB(3), AND DDATAB(5) FEFERENCE ODD BYTE OF SOURCE & DEST DATA
 4777
 4778  016430  013702  001506                  MOV     @#FACTOR,R2     ;GET INDEX VALUE
 4779  016434  010204                          MOV     R2,R4           ;R2 FOR SOURCE EVEN BYTE INDEX, R4 FOR
 4780  016436  010403                          MOV     R4,R3           ;DEST ODD BYTE, R3 FOR SOURCE EVEN
 4781  016440  005203                          INC     R3              ;AND R5 FOR DEST ODD BYTE
 4782  016442  010305                          MOV     R3,R5
 4783  016444  000261                          SEC                             ;SET CARRY
 4784  016446  012762  125252  016570          MOV     #125252,SDATAB(2)
 4785  016454  112763  177125  016570          MOVB    #177125,SDATAB(3)       ;SOURCE DATA = 052652
 4786  016462  016264  016570  016572          MOV     SDATAB(2),DDATAB(4)
 4787  016470  052764  125125  016572          BIS     #125125,DDATAB(4)       ;DEST DATA = 177777
 4788  016476  136263  016570  016570          BITB    SDATAB(2),SDATAB(3)
 4789  016504  001401                          BEQ     .+4
 4790  016506  104000                BITB6:    HLT
 4791
 4792  016510  146264  016570  016572          BICB    SDATAB(2),DDATAB(4)
 4793  016516  103401                          BCS     .+4
```

```
4794  016520  104000                        HLT                             ;ERROR MOV,BIS,BIT;BIC DO NOT AFFECT 'C'
4795  016522  126364  016570  016572        CMPB    SDATAB(3),DDATAB(4)
4796  016530  001401                        BEQ     .+4
4797  016532  104000                        HLT
4798
4799  016534  146365  016570  016572        BICB    SDATAB(3),DDATAB(5)
4800  016542  126265  016570  016572        CMPB    SDATAB(2),DDATAB(5)
4801  016550  001401                        BEQ     .+4
4802  016552  104000                        HLT
4803
4804  016554  136564  016572  016572        BITB    DDATAB(5),DDATAB(4)
4805  016562  001401                        BEQ     .+4
4806  016564  104000                        HLT
4807  016566  000412                        BR      UB7             ;RESERVE TWO WORDS
4808  016570  000000                SDATAB: .WORD   0               ;RESERVED FOR SOURCE DATA
4809  016572  000000                DDATAB: .WORD   0               ;RESERVED FOR DEST DATA
4810
4811                                ;;*****************************************************************
4812                                ;*TEST 30       CHECK BINARY WORD OPS USING ADDRESS MODE 7
4813                                ;*      R2=ADDRESS OF SOURCE DATA, AND R3= ADDRESS OF DEST DATA
4814                                ;;*****************************************************************
4815  016574  112737  000030  001202  TST30:  MOVB    #30,@#$TSTNM            ;LOAD TEST NUMBER
4816  016602  000004                        SCOPE
4817  016604  000000                SBIN7:  .WORD   0               ;CONTAINS ADDRESS OF SOURCE DATA
4818  016606  000000                DBIN7:  .WORD   0               ;CONTAINS ADDRESS OF DEST DATA
4819  016610  000000                        .WORD   0               ;CONTAINS SOURCE DATA
4820  016612  000000                        .WORD   0               ;CONTAINS DEST DATA
4821
4822  016614  010700                UB7:    MOV     PC,R0
4823  016616  024040                        CMP     -(R0),-(R0)
4824  016620  010002                        MOV     R0,R2
4825  016622  024242                        CMP     -(R2),-(R2)
4826  016624  010012                        MOV     R0,(R2)
4827  016626  010203                        MOV     R2,R3
4828  016630  024043                        CMP     -(R0),-(R3)
4829  016632  010013                        MOV     R0,(R3)
4830
4831  016634  000261                        SEC
4832  016636  012777  100000  177740        MOV     #100000,@SBIN7  ;SOURCE DATA = 100000
4833  016644  017777  177734  177734        MOV     @SBIN7,@DBIN7   ;DEST DATA = 100000
4834  016652  103001                        BCC     MOV7
4835  016654  100401                        BMI     .+4
4836  016656  104000                MOV7:   HLT
4837  016660  006377  177722                ASL     @DBIN7          ;DEST DATA = 000000
4838  016664  102001                        BVC     .+4
4839  016666  001401                        BEQ     .+4
4840  016670  104000                        HLT
4841
4842  016672  027777  177706  177706        CMP     @SBIN7,@DBIN7   ;(R2)=100000,(R3)=000000
4843  016700  103402                        BCS     CMP7
4844  016702  102401                        BVS     CMP7
4845  016704  100401                        BMI     .+4
4846  016706  104000                CMP7:   HLT
4847
4848  016710  167777  177670  177670        SUB     @SBIN7,@DBIN7   ;(R2)=100000,(R3)=100000
4849  016716  103003                        BCC     SUB7
```

```
4850  016720  102002                          BVC     SUB7
4851  016722  001401                          BEQ     SUB7
4852  016724  100401                          BMI     .+4
4853  016726  104000                  SUB7:   HLT
4854
4855  016730  006277  177650                  ASR     @SBIN7                  ;(R2)=140000
4856  016734  067777  177644  177644          ADD     @SBIN7,@DBIN7           ;(R2)=140000,(R3)=040000
4857  016742  103003                          BCC     ADD7
4858  016744  102002                          BVC     ADD7
4859  016746  001401                          BEQ     ADD7
4860  016750  100001                          BPL     .+4
4861  016752  104000                  ADD7:   HLT
4862
4863  016754  047777  177624  177624          BIC     @SBIN7,@DBIN7           ;(R2)=140000,(R3)=000000
4864  016762  001401                          BEQ     .+4
4865  016764  104000                          HLT
4866
4867  016766  057777  177612  177612          BIS     @SBIN7,@DBIN7           ;(R2)=140000,(R3)=140000
4868  016774  100401                          BMI     .+4
4869  016776  104000                          HLT
4870
4871  017000  027777  177600  177600          CMP     @SBIN7,@DBIN7
4872  017006  001401                          BEQ     .+4
4873  017010  104000                          HLT
4874                          ;;*****************************************************************
4875                          ;*TEST 31       SOME MISCELLANEOUS OPERATIONS INVOLVING THE PC
4876                          ;*      NOTE: NONE OF THESE OPERATIONS SHOULD AFFECT THE PC
4877                          ;;*****************************************************************
4878  017012  112737  000031  001202  TST31:  MOVB    #31,@#$TSTNM            ;LOAD TEST NUMBER
4879  017020  000004                          SCOPE
4880  017022  005000                          CLR     R0
4881  017024  005067  000072                  CLR     1$
4882  017030  010707                          MOV     PC,PC
4883  017032  120707                          CMPB    PC,PC
4884  017034  030707                          BIT     PC,PC
4885  017036  060007                          ADD     R0,PC
4886  017040  105707                          TSTB    PC
4887  017042  005507                          ADC     PC
4888  017044  021007                          CMP     (R0),PC
4889  017046  131007                          BITB    (R0),PC
4890  017050  062707  000000                  ADD     #0,PC
4891  017054  023707  001506                  CMP     @#FACTOR,PC
4892  017060  133707  001506                  BITB    @#FACTOR,PC
4893  017064  000240                          NOP
4894                          ;THE NEXT TWO INSTRUCTION CAUSE THE PROGRAM TO JUMP TO THE UNRELOCATED
4895                          ;CODE AND TO RETURN ON THE FOLLOWING INST (IF THE CODE IS RELOCATED)
4896  017066  163707  001506                  SUB     @#FACTOR,PC             ;JUMPS TO UNRELOCATED CODE
4897  017072  063707  001506                  ADD     @#FACTOR,PC             ;RETURNS
4898  017076  000240                          NOP
4899  017100  024607                          CMP     -(SP),PC
4900  017102  132607                          BITB    (SP)+,PC
4901  017104  026707  000012                  CMP     1$,PC
4902  017110  166707  000006                  SUB     1$,PC
4903  017114  046707  000002                  BIC     1$,PC
4904  017120  000401                          BR      .+4                     ;BRANCH OVER 1$
4905  017122  000000                  1$:     0
```

```
 4906  017124  000004                    RELE2:  SCOPE
 4907  017126  010702                            MOV     PC,R2
 4908  017130  062702  000012                    ADD     #12,R2
 4909  017134  012707  034240                    MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
 4910  017140  000000                    REL22:  .WORD   0
 4911                                    ;2222222222222 LAST ADDRESS OF CODE TO BE RELOCATED 22222222222
 4912
 4913                                    ;;*****************************************************************
 4914                                    ;*TEST 32        CHECK BINARY BYTE OPS USING ADDRESS MODE 0
 4915                                    ;;*****************************************************************
 4916  017142  112737  000032  001202    TST32:  MOVB    #32,@#$TSTNM            ;LOAD TEST NUMBER
 4917  017150  012767  000001  162140            MOV     #1,$TIMES       ;;DO 1 ITERATION
 4918  017156  000004                            SCOPE
 4919
 4920                                            .SBTTL  START OF SECTION 3
 4921                                    ;3333333333333 FIRST ADDRESS TO BE RELOCATED 333333333
 4922  017160  010700                    REL3:   MOV     PC,R0           ;GET PC
 4923  017162  005740                            TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL3
 4924  017164  010037  001512                    MOV     R0,@#FRSTAD     ;SAVE
 4925  017170  010700                            MOV     PC,R0           ;GET CURRENT PC
 4926  017172  162700  017172                    SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
 4927  017176  010037  001506                    MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
 4928  017202  010737  001212                    MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
 4929  017206  062737  000030  001212            ADD     #30,@#$LPERR    ;ADJUST
 4930  017214  013737  001212  001210            MOV     @#$LPERR,@#$LPADR
 4931  017222  105737  001502                    TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
 4932  017226  001402                            BEQ     .+6
 4933  017230  000167  002250                    JMP     RELE3
 4934  017234  012703  125252                    MOV     #125252,R3
 4935  017240  010304                            MOV     R3,R4           ;R3=R4=125252
 4936  017242  140304                            BICB    R3,R4           ;R3=125252,R4=125000
 4937  017244  022704  125000                    CMP     #125000,R4      ;CHECK RESULT
 4938  017250  001401                            BEQ     .+4
 4939  017252  104000                            HLT
 4940
 4941  017254  005004                            CLR     R4              ;R3=125252,R4=0
 4942  017256  150304                            BISB    R3,R4           ;R3=125252,R4=000252
 4943  017260  022704  000252                    CMP     #252,R4         ;CHECK RESULT
 4944  017264  001401                            BEQ     .+4
 4945  017266  104000                            HLT
 4946
 4947  017270  110404                            MOVB    R4,R4           ;R4=177652
 4948  017272  022704  177652                    CMP     #177652,R4      ;CHECK RESULT
 4949  017276  001401                            BEQ     .+4
 4950  017300  104000                            HLT
 4951
 4952  017302  132704  177525                    BITB    #177525,R4
 4953  017306  001401                            BEQ     .+4
 4954  017310  104000                            HLT
 4955
 4956  017312  105104                            COMB    R4              ;R4=177525
 4957  017314  110404                            MOVB    R4,R4           ;R4=000125
 4958  017316  022704  000125                    CMP     #125,R4         ;CHECK RESULT
 4959  017322  001401                            BEQ     .+4
 4960  017324  104000                            HLT
 4961
```

```
4962  017326  150304                          BISB    R3,R4           ;R3=125252,R4=000377
4963  017330  105204                          INCB    R4
4964  017332  001401                          BEQ     .+4
4965  017334  104000                          HLT
4966                                ;;*****************************************************************
4967                                ;*TEST 33       CHECK BINARY BYTE OPS USING ADDRESS MODE 7
4968                                ;;*****************************************************************
4969  017336  112737  000033  001202  TST33:  MOVB    #33,@#$TSTNM            ;LOAD TEST NUMBER
4970  017344  000004                          SCOPE
4971  017346  000406                          BR      BINB7           ;RESERVE SPACE FOR ADDRESSES & DATA
4972  017350  000000                  SBINB7: .WORD   0               ;CONTAINS ADDRESS OF SOURCE EVEN BYTE
4973  017352  000000                          .WORD   0               ;CONTAINS ADDRESS OF SOURCE ODD BYTE
4974  017354  000000                          .WORD   0               ;CONTAINS ADDRESS OF DEST EVEN BYTE
4975  017356  000000                          .WORD   0               ;CONTAINS ADDRESS OF DEST ODD BYTE
4976  017360  000000                  DBINB7: .WORD   0               ;CONTAINS SOURCE DATA
4977  017362  000000                          .WORD   0               ;CONTAINS DEST DATA
4978
4979  017364  010700                  BINB7:  MOV     PC,R0
4980  017366  024040                          CMP     -(R0),-(R0)     ;R0 = ADDRESS OF DEST DATA
4981  017370  010060  177772                  MOV     R0,-6(R0)       ;LOAD ADDRES OF DEST EVEN BYTE DATA
4982  017374  010060  177774                  MOV     R0,-4(R0)
4983  017400  005260  177774                  INC     -4(R0)          ;LOAD ADDRESS OF DEST ODD BYTE DATA
4984  017404  005740                          TST     -(R0)           ;R0=ADDRESS OF SOURCE DATA
4985  017406  010060  177770                  MOV     R0,-10(R0)      ;LOAD ADDRESS OF SOURCE EVEN BYTE DATA
4986  017412  010060  177772                  MOV     R0,-6(R0)
4987  017416  005260  177772                  INC     -6(R0)          ;LOAD ADDRESS OF SOURCE ODD BYTE DATA
4988
4989  017422  005002                          CLR     R2              ;SET INDEX REGISTERS
4990  017424  012703  000002                  MOV     #2,R3           ;@SBINB7(2);@SBINB7(3) REFERENCE EVEN &
4991  017430  012704  177774                  MOV     #-4,R4          ;ODD BYTE SOURCE DATA; @DBINB7(4);@DBINB7(5)
4992  017434  012705  177776                  MOV     #-2,R5          ;REFERENCE DEST EVEN& ODD BYTE DATA
4993
4994
4995  017440  005020                          CLR     (R0)+           ;PRESET SOURCE DATA
4996  017442  005010                          CLR     (R0)            ;PRESET DEST DATA
4997  017444  013746  001506                  MOV     @#FACTOR,-(SP)  ;GET RELOCATION FACTOR
4998  017450  061602                          ADD     (SP),R2         ;AND ADD TO INDEX VALUES
4999  017452  061603                          ADD     (SP),R3
5000  017454  061604                          ADD     (SP),R4
5001  017456  062605                          ADD     (SP)+,R5
5002
5003  017460  112773  177777  017350          MOVB    #-1,@SBINB7(3)  ;SRC DATA = 177400
5004  017466  132772  000377  017350          BITB    #377,@SBINB7(2) ;CHECK THAT EVEN BYTE WAS NOT AFFECTED
5005  017474  001401                          BEQ     .+4             ;BY MOVB INSTRUCTION
5006  017476  104000                          HLT
5007
5008  017500  157374  017350  017360          BISB    @SBINB7(3),@DBINB7(4)
5009  017506  105274  017360                  INCB    @DBINB7(4)      ;CHECK THAT BIS SET ALL BITS
5010  017512  001401                          BEQ     .+4
5011  017514  104000                          HLT
5012
5013  017516  105375  017360                  DECB    @DBINB7(5)      ;DEST DATA = 177400
5014  017522  005274  017360                  INC     @DBINB7(4)      ;DEST DATA = 177401
5015  017526  127375  017350  017360          CMPB    @SBINB7(3),@DBINB7(5)
5016  017534  001401                          BEQ     .+4
5017  017536  104000                          HLT
```

```
5018
5019  017540  147375  017350  017360          BICB    @SBINB7(3),@DBINB7(5)
5020  017546  001401                          BEQ     .+4
5021  017550  104000                          HLT
5022
5023  017552  105073  017350                  CLRB    @SBINB7(3)      ;SRC DATA = 000000
5024                                ;THIS ROUTINE SETS ALL BITS IN THE SOURCE ODD BYTE BY BISING A BIT FROM
5025                                ;THE DEST EVEN BYTE INTO THE SOURCE ODD BYTE
5026  017556  157473  017360  017350  BIS7:   BISB    @DBINB7(4),@SBINB7(3)
5027  017564  106174  017360                  ROLB    @DBINB7(4)
5028  017570  103372                          BCC     BIS7
5029  017572  022772  177400  017350          CMP     #177400,@SBINB7(2)      ;CHECK RESULT
5030  017600  001401                          BEQ     .+4
5031  017602  104000                          HLT
5032
5033  017604  000372  017350                  SWAB    @SBINB7(2)      ;SRC DATA = 000377
5034  017610  112775  000200  017360          MOVB    #200,@DBINB7(5) ;DEST DATA = 100000
5035
5036  017616  147572  017360  017350  BIC7:   BICB    @DBINB7(5),@SBINB7(2)
5037  017624  106075  017360                  RORB    @DBINB7(5)
5038  017630  103372                          BCC     BIC7
5039  017632  005772  017350                  TST     @SBINB7(2)
5040  017636  001401                          BEQ     .+4
5041  017640  104000                          HLT
5042
5043  017642  012702  000001          OAERR:  MOV     #1,R2           ;LOAD R2 WITH ODD #
5044  017646  010703                          MOV     PC,R3
5045  017650  000401                          BR      .+4             ;RESERVE SPACE FOR A WORD
5046  017652  000000                          .WORD   0               ;WILL CONTAIN AN ODD ADDRESS
5047  017654  005723                          TST     (R3)+           ;STEP R3 TO POINT TO WORD ABOVE
5048  017656  010313                          MOV     R3,(R3)
5049  017660  005213                          INC     (R3)            ;AND MAKE ODD
5050  017662  012737  020010  000004          MOV     #1$,@#ERRVEC    ;SET ODD ADDRESS & RESERVED INSTRUCTION
5051  017670  063737  001506  000004          ADD     @#FACTOR,@#ERRVEC
5052  017676  013737  000004  000010          MOV     @#ERRVEC,@#RESVEC       ;TO TRAP TO 1$ BELOW
5053
5054  017704  000277                          SCC                     ;SET ALL CC'S
5055  017706  160212                          SUB     R2,(R2)
5056  017710  104000                          HLT
5057  017712  060222                          ADD     R2,(R2)+
5058  017714  104000                          HLT
5059  017716  006342                          ASL     -(R2)
5060  017720  104000                          HLT
5061  017722  106512                          MFPD    (R2)            ;NOTE: MAY BE RESERVED
5062  017724  104000                          HLT
5063  017726  170412                          CLRF    (R2)
5064  017730  104000                          HLT
5065  017732  042202                          BIC     (R2)+,R2
5066  017734  104000                          HLT
5067  017736  164202                          SUB     -(R2),R2
5068  017740  104000                          HLT
5069  017742  155202                          BISB    @-(R2),R2
5070  017744  104000                          HLT
5071  017746  105532                          ADCB    @(R2)+
5072  017750  104000                          HLT
5073  017752  163302                          SUB     @(R3)+,R2
```

```
5074  017754  104000                          HLT
5075  017756  005733                          TST     @(R3)+
5076  017760  104000                          HLT
5077  017762  106533                          MFPD    @(R3)+
5078  017764  104000                          HLT
5079  017766  170453                          CLRD    @-(R3)
5080  017770  104000                          HLT
5081  017772  137702  177775                  BITB    @.+1,R2
5082  017776  104000                          HLT
5083  020000  105477  177773                  NEGB    @.-1
5084  020004  104000                          HLT
5085  020006  000406                          BR      2$
5086
5087  020010  062716  000002          1$:     ADD     #2,(SP)         ;ADJUST RETURN PC
5088  020014  052766  000017  000002          BIS     #17,2(SP)       ;SET CONDITION CODES ON RETURN
5089  020022  000002                          RTI
5090
5091  020024  012706  000700          2$:     MOV     #SUPSTK,SP      ;RESET STACK PTR
5092  020030  012737  053440  000004          MOV     #ERPRT,@#ERRVEC ;RESET TIME OUT VECTOR
5093  020036  012737  053366  000010          MOV     #RESERR,@#RESVEC
5094                                  ;;*****************************************************************
5095                                  ;*TEST 34        CHECK JUMP INSTRUCTIONS
5096                                  ;;*****************************************************************
5097  020044  112737  000034  001202  TST34:  MOVB    #34,@#$TSTNM            ;LOAD TEST NUMBER
5098  020052  000004                          SCOPE
5099  020054  010700                          MOV     PC,R0
5100  020056  062700  000012                  ADD     #12,R0          ;SET ADDRESS FOR JMP INST
5101  020062  000277                          SCC                     ;SET CC'S
5102  020064  000110                          JMP     (R0)
5103  020066  000402                          BR      .+6
5104  020070  000250                          CLN                     ;JMP INST JUMPS HERE
5105  020072  000775                          BR      .-4
5106
5107  020074  103003                          BCC     JMP1
5108  020076  102002                          BVC     JMP1
5109  020100  001001                          BNE     JMP1
5110  020102  100001                          BPL     .+4
5111  020104  104000                  JMP1:   HLT                     ;ERROR! INCORRECT CC'S AFTER JMP
5112
5113  020106  005002                          CLR     R2              ;SET INDICATOR
5114  020110  010703                          MOV     PC,R3
5115  020112  000401                          BR      .+4             ;RESERVE WORD FOR JMP ADDRESS
5116  020114  000000                          .WORD   0               ;CONTAINS ADDRESS FOR JMP INST
5117  020116  005723                          TST     (R3)+
5118  020120  010313                          MOV     R3,(R3)
5119  020122  010300                          MOV     R3,R0
5120  020124  062713  000022                  ADD     #22,(R3)        ;(R3) IS JMP ADDRESS
5121  020130  010300                          MOV     R3,R0
5122  020132  000133                          JMP     @(R3)+          ;JUMP TO ADDRESS CONTAINED IN R3
5123  020134  000402                          BR      .+6
5124  020136  005102                          COM     R2              ;COMPLEMENT INDICATOR
5125  020140  000775                          BR      .-4
5126  020142  005202                          INC     R2              ;CHECK INDICATOR
5127  020144  001003                          BNE     JMP3
5128  020146  005720                          TST     (R0)+
5129  020150  020003                          CMP     R0,R3           ;CHECK AUTO-INC R3
```

```
5130  020152  001401                    BEQ     .+4
5131  020154  104000            JMP3:   HLT
5132
5133  020156  005002                    CLR     R2              ;SET INDICATOR
5134  020160  010704                    MOV     PC,R4           ;SET UP JMP REGISTER
5135  020162  010400                    MOV     R4,R0           ;SET UP CHECK REGISTER
5136  020164  000402                    BR      1$
5137  020166  005102                    COM     R2              ;COMPLEMENT INDICATOR
5138  020170  000403                    BR      2$
5139  020172  022424            1$:     CMP     (R4)+,(R4)+
5140  020174  005724                    TST     (R4)+           ;R4=JMP ADDRESS
5141  020176  000144                    JMP     -(R4)           ;USE R4 AS ADDRESS
5142  020200  005202            2$:     INC     R2              ;CHECK INDICATOR
5143  020202  001003                    BNE     JMP4
5144  020204  022020                    CMP     (R0)+,(R0)+
5145  020206  020004                    CMP     R0,R4           ;CHECK AUTO-DEC R4
5146  020210  001401                    BEQ     .+4
5147  020212  104000            JMP4:   HLT
5148
5149  020214  010703                    MOV     PC,R3
5150  020216  000401                    BR      .+4             ;RESERVE WORD FOR JMP ADDRESS
5151  020220  000000            1$:     .WORD   0               ;CONTAINS JUMP ADDRESS
5152  020222  005723                    TST     (R3)+
5153  020224  010313                    MOV     R3,(R3)
5154  020226  062723  000016            ADD     #16,(R3)+
5155  020232  010300                    MOV     R3,R0           ;LOAD CHECK REGISTER
5156  020234  000402                    BR      3$
5157  020236  005102            2$:     COM     R2
5158  020240  000401                    BR      4$
5159  020242  000153            3$:     JMP     @-(R3)          ;JUMP TO 2$ VIA 1$ ABOVE
5160  020244  005202            4$:     INC     R2              ;CHECK INDICATOR
5161  020246  001003                    BNE     JMP5
5162  020250  005740                    TST     -(R0)
5163  020252  020003                    CMP     R0,R3           ;CHECK AUTO-DEC R3
5164  020254  001401                    BEQ     .+4
5165  020256  104000            JMP5:   HLT
5166
5167  020260  000402                    BR      2$
5168  020262  005102            1$:     COM     R2              ;COMPLEMENT INDICATOR
5169  020264  000402                    BR      3$
5170  020266  000167  177770    2$:     JMP     1$
5171  020272  005202            3$:     INC     R2
5172  020274  001401                    BEQ     .+4
5173  020276  104000            JMP6:   HLT
5174
5175  020300  012767  020316  000020    MOV     #1$,7$          ;SET UP JMP ADDRESS
5176  020306  063767  001506  000012    ADD     @#FACTOR,7$     ;ADD RELOCATION FACTOR
5177  020314  000402                    BR      2$              ;GO TO JMP @7$ INST
5178  020316  005102            1$:     COM     R2              ;COMPLEMENT INDICATOR
5179  020320  000403                    BR      3$              ;GO TO CHECK ROUTINE
5180  020322  000177  000000    2$:     JMP     @7$             ;JMP TO 1$ ABOVE VIA 7$
5181  020326  000000            7$:     .WORD   0               ;CONTAINS JMP ADDRESS
5182  020330  005202            3$:     INC     R2              ;CHECK INDICATOR
5183  020332  001401                    BEQ     .+4
5184  020334  104000            JMP7:   HLT
5185                            ;;*****************************************************************
```

```
                                        ;*TEST 35       CHECK JSR INSTRUCTIONS
                                        ;;*****************************************************************
020336  112737  000035  001202  TST35:  MOVB    #35,@#$TSTNM            ;LOAD TEST NUMBER
020344  000004                    SCOPE
020346  013705  001506  JSR1:    MOV     @#FACTOR,R5     ;GET RELOCATION FACTOR
020352  012702  020404            MOV     #3$,R2          ;FORM DEST ADRS
020356  060502                    ADD     R5,R2           ;ADD RELOCATION FACTOR
020360  000277                    SCC                     ;PRESET CC'S
020362  000242                    CLV
020364  004512                    JSR     R5,(R2)         ;GO TO 3$ VIA R2
020366  005702          1$:       TST     R2              ;CHECK INDICATOR
020370  001017                    BNE     4$              ;R2 SHOULD=0
020372  023705  001506            CMP     @#FACTOR,R5     ;CHECK THAT RTS  R5 RESTORED R5
020376  001014                    BNE     4$
020400  000414                    BR      JSR3            ;GO TO NEXT TEST
020402  000205          2$:       RTS     R5              ;RETURN FROM SUBROUTINE
020404  103011          3$:       BCC     4$              ;CHECK THAT JSR DID NOT
020406  102410                    BVS     4$
020410  001007                    BNE     4$              ;AFFECT CC'S
020412  100006                    BPL     4$
020414  005002                    CLR     R2              ;CLEAR INDICATOR
020416  012704  020366            MOV     #1$,R4          ;GET UNRELOCATED RETURN ADDRESS
020422  061604                    ADD     (SP),R4         ;ADD RELOCATION FACTOR (OLD R5)
020424  020405                    CMP     R4,R5           ;CHECK THAT OLD R5 WAS PLACED ON THE
020426  001765                    BEQ     2$              ;STACK,& THAT NEW R5 CONTAINS RETURN PC
020430  104000          4$:       HLT                     ;ERROR! ABOVE

                                        ;CHECK JSR INSTRUCTION ADDRESS MODE 3
020432  013704  001506  JSR3:    MOV     @#FACTOR,R4     ;GET RELOCATION FACTOR
020436  005000                    CLR     R0              ;SET INDICATOR
020440  012705  020460            MOV     #1$,R5
020444  060405                    ADD     R4,R5           ;SET UP JSR DEFERRED ADRS
020446  010502                    MOV     R5,R2
020450  012715  020476            MOV     #5$,(R5)
020454  060415                    ADD     R4,(R5)         ;(R5)=DEST ADRS
020456  000401                    BR      2$              ;RESERVE WORD FOR ADDRESS
020460  000000          1$:       .WORD   0               ;CONTAINS DEST ADRS FOR JSR
020462  004435          2$:       JSR     R4,@(R5)+       ;JSR TO 5$ VIA 1$ ABOVE
020464  005200          3$:       INC     R0              ;CHECK INDICATOR
020466  001013                    BNE     6$
020470  000413                    BR      JSR4
020472  005100          4$:       COM     R0              ;COMPLEMENT INDICATOR
020474  000204                    RTS     4               ;RETURN FROM SUBROUTINE
020476  012703  020464  5$:       MOV     #3$,R3          ;GET UNRELOCATED RETURN ADDRESS
020502  061603                    ADD     (SP),R3         ;ADD RELOCATION FACTOR (OLD R4)
020504  020403                    CMP     R4,R3
020506  001003                    BNE     6$
020510  005722                    TST     (R2)+
020512  020205                    CMP     R2,R5           ;CHECK AUTO-INC R5
020514  001766                    BEQ     4$              ;GO TO RTS
020516  104000          6$:       HLT                     ;ERROR ABOVE

                                        ;CHECK JSR INST ADDRESS MODE 4
020520  013704  001506  JSR4:    MOV     @#FACTOR,R4
020524  010405                    MOV     R4,R5
020526  010703                    MOV     PC,R3
```

```
5242  020530  000401                    BR      2$
5243  020532  000405              1$:   BR      4$
5244  020534  022323              2$:   CMP     (R3)+,(R3)+
5245  020536  000277                    SCC
5246  020540  004443                    JSR     R4,-(R3)        ;GO TO 2$
5247  020542  104000              3$:   HLT
5248  020544  000414                    BR      JSR6            ;GO TO NEXT TEST
5249  020546  103012              4$:   BCC     5$
5250  020550  102011                    BVC     5$
5251  020552  001010                    BNE     5$
5252  020554  100007                    BPL     5$
5253  020556  012702  020542            MOV     #3$,R2          ;GET UNRELOCATED RETURN ADDRESS
5254  020562  061602                    ADD     (SP),R2         ;ADD RELOCATION FACTOR (OLD R4)
5255  020564  020204                    CMP     R2,R4           ;CHECK THAT CALCULATED RETURN
5256  020566  001002                    BNE     5$              ;PC = NEW R4
5257  020570  005724                    TST     (R4)+
5258  020572  000204                    RTS     R4
5259  020574  104000              5$:   HLT
5260
5261                              ;TEST JSR INST ADDRESS MODE 6
5262  020576  000401              JSR6: BR      2$
5263  020600  000405              1$:   BR      3$
5264  020602  010700              2$:   MOV     PC,R0
5265  020604  004767  177770            JSR     PC,1$
5266  020610  100407                    BMI     JSR7            ;GO TO NEXT TEST
5267  020612  104000                    HLT                     ;ERROR ON CC'S
5268  020614  022020              3$:   CMP     (R0)+,(R0)+
5269  020616  020016                    CMP     R0,(SP)         ;CHECK THAT RETURN ADDRESS IS ON THE
5270  020620  001401                    BEQ     .+4             ;STACK
5271  020622  104000                    HLT
5272  020624  000270                    SEN                     ;SET N
5273  020626  000207                    RTS     PC
5274
5275                              ;TEST JSR INST ADDRESS MODE 7
5276  020630  013746  001506      JSR7: MOV     @#FACTOR,-(SP)  ;GET RELOCATION FACTOR
5277  020634  062716  020654            ADD     #1$,(SP)        ;FORM ADDRESS OF 1$ BELOW
5278  020640  000277                    SCC                     ;SET ALL CC'S
5279  020642  004076  000000            JSR     R0,@(SP)        ;JSR TO 1$
5280  020646  003003                    BGT     3$
5281  020650  102002                    BVC     3$
5282  020652  000402                    BR      4$
5283
5284  020654  000200              1$:   RTS     R0              ;RETURN
5285  020656  104000              3$:   HLT                     ;ERROR!! INCORRECT CC'S
5286  020660                      4$:
5287                              ;;*****************************************************************
5288                              ;*TEST 36       CHECK IOT TRAP (AND ROLB/ASLB)
5289                              ;*      THIS TEST CHECKS THAT THE PSW IS CORRECT AFTER THE IOT AND THAT THE
5290                              ;*      'NEW'PSW (FROM IOTVEC+2) IS CORRECT.
5291                              ;;*****************************************************************
5292  020660  112737  000036  001202  TST36: MOVB    #36,@#$TSTNM            ;LOAD TEST NUMBER
5293  020666  000004                    SCOPE
5294  020670  012705  000022      IOTTST: MOV     #IOTVEC+2,R5
5295  020674  005000                    CLR     R0
5296  020676  052740  000200            BIS     #PR4,-(R0)              ;SET PRIORITY LEVEL 4 IN PSW
5297  020702  011015                    MOV     (R0),(R5)               ;SET IOTVEC+2 = PSW
```

```
5298  020704  011504                          MOV     (R5),R4                 ;SAVE IN R4
5299  020706  010746                          MOV     PC,-(SP)
5300  020710  062716  000036                  ADD     #1$-.,(SP)
5301  020714  012645                          MOV     (SP)+,-(R5)             ;LOAD IOT TRAP VECTOR
5302  020716  042710  000357                  BIC     #PR7+17,(R0)
5303  020722  052710  000244                  BIS     #PR5+4,(R0)             ;PSW=X XXX X00 101 1X1 000
5304  020726  012003                          MOV     (R0)+,R3                ;R3 = PSW ABOVE
5305  020730  010340                          MOV     R3,-(R0)                ;RESTORE PSW (MOV CHANGED IT)
5306  020732  000004                          IOT
5307  020734  012737  044532  000020  10$:    MOV     #$SCOPE,@#IOTVEC        ;RESTORE IOT VECTOR
5308  020742  104000                          HLT                             ;ERROR! IOT FAILED TO TRAP
5309  020744  000457                          BR      TST37           ;;GO TO NEXT TEST
5310
5311  020746  012002                  1$:     MOV     (R0)+,R2                ;GET PSW AFTER IOT TRAP
5312                                                                          ;NOTE: R0=0
5313  020750  012725  044532                  MOV     #$SCOPE ,(R5)+          ;RESTORE IOTVEC
5314  020754  012715  000200                  MOV     #PR4,(R5)               ;AND IOTVEC+2
5315  020760  010746                          MOV     PC,-(SP)                ;FORM PC OF 10$ ABOVE
5316  020762  062716  177752                  ADD     #10$-.,(SP)
5317  020766  022626                          CMP     (SP)+,(SP)+             ;CHECK RETURN PC ON STACK
5318  020770  001036                          BNE     99$
5319  020772  022603                          CMP     (SP)+,R3                ;CHECK SAVED PSW
5320  020774  001034                          BNE     99$
5321  020776  032703  140000                  BIT     #UM,R3                  ;BRANCH TO 3$ IF IN USER MODE
5322  021002  100413                          BMI     3$
5323  021004  001003                          BNE     2$                      ;BRANCH TO 2$ IF IN SUPER MODE
5324  021006  020204                          CMP     R2,R4                   ;CHECK PSW AFTER IOT
5325  021010  001026                          BNE     99$
5326  021012  000413                          BR      4$
5327
5328  021014  042704  030000          2$:     BIC     #PUM,R4                 ;CLEAR PREV MODE BITS
5329  021020  052704  010000                  BIS     #PSM,R4                 ;SET PREV SUPER MODE
5330  021024  020204                          CMP     R2,R4                   ;CHECK PSW AFTER IOT
5331  021026  001017                          BNE     99$
5332  021030  000404                          BR      4$
5333
5334  021032  052704  030000          3$:     BIS     #PUM,R4                 ;SET PREV USER MODE
5335  021036  020204                          CMP     R2,R4                   ;CHECK PSW AFTER IOT
5336  021040  001012                          BNE     99$
5337
5338  021042  005002                  4$:     CLR     R2
5339  021044  000261                          SEC
5340  021046  106100                          ROLB    R0                      ;ROTATE R0
5341  021050  102376                          BVC     .-2                     ;UNTIL V SETS (R0=200)
5342
5343  021052  106300                          ASLB    R0                      ;SHIFT SHOULD SET CARRY
5344  021054  103004                          BCC     99$
5345  021056  102003                          BVC     99$
5346  021060  001002                          BNE     99$
5347  021062  005700                          TST     R0
5348  021064  001401                          BEQ     .+4
5349  021066  104000                  99$:    HLT                             ;ERROR! ROL/ASL FAILED TO SET
5350                                                                          ;CC'S PROPERLY (IF R2=0) OR IN-
5351                                                                          ;CORRECT PSW AFTER IOT (IF R2 NOT 0)
5352  021070  042704  000340                  BIC     #PR7,R4
5353  021074  010437  177776                  MOV     R4,@#PSW                ;RESTORE PSW
```

```
5354  021100  012706  000700                  MOV     #SUPSTK,SP              ;RESTORE STACK PTR
5355                                  ;;*****************************************************************
5356                                  ;*TEST 37       CHECK EMT TRAP SEQUENCE
5357                                  ;;*****************************************************************
5358  021104  112737  000037  001202  TST37:  MOVB    #37,@#$TSTNM            ;LOAD TEST NUMBER
5359  021112  000004                  SCOPE
5360                                  .EQUIV  IOT,HLT                 ;REDEFINE HLT CALL
5361  021114  012737  044772  000020          MOV     #$ERROR,@#IOTVEC        ;SETUP VECTOR
5362  021122  012737  000340  000022          MOV     #PR7,@#IOTVEC+2
5363  021130  005000                          CLR     R0
5364  021132  010746                          MOV     PC,-(SP)
5365  021134  062716  000030                  ADD     #EMT1-.,(SP)
5366  021140  012637  000030                  MOV     (SP)+,@#EMTVEC
5367  021144  000262                          SEV                             ;SET V
5368  021146  013737  177776  000032          MOV     @#PSW,@#EMTVEC+2        ;RETAIN CURRENT PSW ON TRAP
5369  021154  000265                          +SEZ!SEC
5370  021156  104000                          EMT                     ;TRAP TO EMT1
5371  021160  001433                          BEQ     EMT1C           ;GO TO EMT1C
5372  021162  000004                          HLT                     ;ERROR! INCORRECT CC'S WERE SET ON RETURN
5373  021164  102027                  EMT1:   BVC     EMT1B           ;'V' SHOULD'VE SET ON EMT TRAP
5374  021166  105100                          COMB    R0              ;R0=000377,CC'S=1001
5375  021170  105500                          ADCB    R0              ;R0=000000,CC'S=0101
5376  021172  106000                          RORB    R0              ;R0=000200,CC'S=1010
5377  021174  102023                          BVC     EMT1B
5378  021176  100022                          BPL     EMT1B
5379  021200  000257                          CCC
5380  021202  105400                          NEGB    R0              ;R0=000200,CC'S=1010
5381  021204  102017                          BVC     EMT1B
5382  021206  100016                          BPL     EMT1B
5383  021210  000242                          CLV                     ;CLEAR 'V'
5384  021212  000261                          SEC                     ;AND SET 'C'
5385  021214  105300                          DECB    R0              ;R0=000177,CC'S=0011
5386  021216  102012                          BVC     EMT1B
5387  021220  100411                          BMI     EMT1B
5388  021222  000242                          CLV                     ;CLEAR 'V'
5389  021224  105200                          INCB    R0              ;R0=000200,CC'S=1011
5390  021226  103006                          BCC     EMT1B
5391  021230  102005                          BVC     EMT1B
5392  021232  100004                          BPL     EMT1B
5393  021234  000242                          CLV                     ;CLEAR 'V'
5394  021236  106200                          ASRB    R0              ;SHIFT R0 UNTIL 'V' CLEARS
5395  021240  102776                          BVS     .-2
5396  021242  000401                          BR      .+4
5397  021244  000004          EMT1B:  HLT                     ;ERROR!
5398  021246  000002                          RTI                     ;EXIT WITH R0=000377
5399  021250  105500          EMT1C:  ADCB    R0              ;R0=000000
5400  021252  103003                          BCC     EMT1D
5401  021254  001002                          BNE     EMT1D
5402  021256  005700                          TST     R0
5403  021260  001401                          BEQ     .+4
5404  021262  000004          EMT1D:  HLT
5405  021264  012737  044772  000030          MOV     #$ERROR,@#EMTVEC        ;RESTORE EMT TO ERROR
5406  021272  012737  000340  000032          MOV     #PR7,@#EMTVEC+2         ;SET PRIORITY 7 ON ERROR
5407  021300  012737  044532  000020          MOV     #$SCOPE,@#IOTVEC        ;RESTORE IOT VECTOR
5408  021306  005037  000022                  CLR     @#IOTVEC+2
5409                                  .EQUIV  ERROR,HLT               ;REDEFINE HLT CALL
```

```
                                ;;*****************************************************************
                                ;*TEST 40        CHECK TRAP INSTRUCTION TRAP SEQUENCE
                                ;;*****************************************************************
021312  112737  000040  001202  TST40:  MOVB    #40,@#$TSTNM            ;LOAD TEST NUMBER
021320  000004                         SCOPE
021322  052737  000340  177776         BIS     #PR7,@#PSW              ;LOCK OUT LINE CLOCK
021330  052737  000340  000016         BIS     #PR7,@#TBITVEC+2
021336  010746                         MOV     PC,-(SP)
021340  062716  000056                 ADD     #TRAP1-.,(SP)
021344  012637  000034                 MOV     (SP)+,@#TRAPVEC
021350  000270                         SEN                             ;SET N
021352  013737  177776  000036         MOV     @#PSW,@#TRAPVEC+2       ;RETAIN CURRENT PSW ON TRAP
021360  000261                         SEC                             ;SET CARRY
021362  010700                         MOV     PC,R0
021364  000264                         SEZ                     ;SET Z BIT
021366  104400                         TRAP                    ;TRAP TO TRAP1
021370  103404                         BCS     .+12
021372  012737  051372  000034         MOV     #$TRAP,@#TRAPVEC        ;RESTORE TRAP VECTOR
021400  104000                         HLT
021402  001404                         BEQ     .+12
021404  012737  051372  000034         MOV     #$TRAP,@#TRAPVEC        ;RESTORE TRAP VECTOR
021412  104000                         HLT
021414  000420                         BR      TRAP1C
021416  100404           TRAP1:        BMI     .+12            ;N BIT GOT SET ON TRAP
021420  012737  051372  000034         MOV     #$TRAP,@#TRAPVEC        ;RESTORE TRAP VECTOR
021426  104000                         HLT
021430  062700  000004                 ADD     #4,R0
021434  020016                         CMP     R0,(SP)         ;CHECK LOW BYTE OF RETURN PC ON
021436  001404                         BEQ     .+12            ;STACK
021440  012737  051372  000034         MOV     #$TRAP,@#TRAPVEC        ;RESTORE TRAP VECTOR
021446  104000                         HLT
021450  124646                         CMPB    -(SP),-(SP)
021452  032626                         BIT     (SP)+,(SP)+
021454  000002                         RTI                     ;RETURN TO INST FOLLOWING TRAP (1$)

021456  012702  000036  TRAP1C: MOV     #TRAPVEC+2,R2           ;RESTORE VECTORS
021462  012712  000340         MOV     #PR7,(R2)
021466  012742  051372         MOV     #$TRAP,-(R2)
021472  042737  000340  000016 BIC     #PR7,@#TBITVEC+2
021500  105037  177776         CLRB    @#PSW                   ;GO BACK TO PRIORITY 0

021504  000004  RELE3:  SCOPE
021506  010702         MOV     PC,R2
021510  062702  000012 ADD     #12,R2
021514  012707  034240 MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
021520  000000  REL33:  .WORD   0
                        ;3333333333333 LAST ADDRESS OF CODE TO BE RELOCATED 33333333333

                        ;;*****************************************************************
                        ;*TEST 41        CHECK STACK OVERFLOW
                        ;;*****************************************************************
021522  112737  000041  001202  TST41:  MOVB    #41,@#$TSTNM            ;LOAD TEST NUMBER
021530  012767  000001  157560         MOV     #1,$TIMES       ;;DO 1 ITERATION
021536  000004                         SCOPE

                                .SBTTL  START OF SECTION 4
```

```
5466                                        ;4444444444444 FIRST ADDRESS TO BE RELOCATED 444444444
5467  021540  010700                REL4:   MOV     PC,R0           ;GET PC
5468  021542  005740                        TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL4
5469  021544  010037  001512                MOV     R0,@#FRSTAD     ;SAVE
5470  021550  010700                        MOV     PC,R0           ;GET CURRENT PC
5471  021552  162700  021552                SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
5472  021556  010037  001506                MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
5473  021562  010737  001212                MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
5474  021566  062737  000030  001212        ADD     #30,@#$LPERR    ;ADJUST
5475  021574  013737  001212  001210        MOV     @#$LPERR,@#$LPADR
5476  021602  105737  001502                TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
5477  021606  001402                        BEQ     .+6
5478  021610  000167  001324                JMP     RELE4
5479
5480  021614  013767  177776  000334  OVFLW: MOV     @#PSW,7$        ;SAVE STATUS IN 7$ BELOW
5481  021622  005037  177776                CLR     @#PSW           ;SET KERNEL MODE
5482  021626  004737  052336                JSR     PC,@#CLRTBIT    ;GO CLEAR 'T' BIT IF SET
5483  021632  052737  000340  177776        BIS     #PR7,@#PSW      ;SET PRIORITY LEVEL 7 TO BLOCK CLOCK
5484  021640  010746                        MOV     PC,-(SP)        ;PUSH CURRENT PC ONTO STACK
5485  021642  062716  000152                ADD     #2$-.,(SP)      ;FORM ADDRESS OF 2$ BELOW
5486  021646  011637  000004                MOV     (SP),@#ERRVEC   ;SET ERROR VECTOR
5487  021652  012737  000340  000006        MOV     #340,@#ERRVEC+2 ;SET PRIORITY LEVEL 7 ON TRAP
5488  021660  013727  000014                MOV     @#BPTVEC,(PC)+  ;SAVE BPT VECTOR ADRS
5489  021664  000000                43$:    .WORD   0
5490  021666  062716  000100                ADD     #41$-2$,(SP)    ;FORM ADDRESS OF 41$ BELOW
5491  021672  012637  000014                MOV     (SP)+,@#BPTVEC  ;SET BPT TRAP VECTOR TO 41$
5492  021676  012737  000340  000016        MOV     #340,@#BPTVEC+2
5493
5494  021704  012703  000376                MOV     #376,R3
5495  021710  010313                        MOV     R3,(R3)         ;LOAD 376 INTO ADDRESS 376
5496  021712  010306                        MOV     R3,SP           ;SET STACK PTR AT BOUNDARY
5497  021714  032767  140000  000234        BIT     #UM,7$          ;CHECK IF ENTERED TEST IN KERNEL
5498  021722  001015                        BNE     1$              ;MODE. BRANCH IF NOT IN KERNEL
5499
5500                                ;THE BELOW INSTRUCTIONS SHOULD NOT CAUSE AN OVERFLOW TRAP
5501  021724  005716                        TST     (SP)            ;BECAUSE TST IS A NON MODIFYING INST
5502  021726  021666  177776                CMP     (SP),-2(SP)     ;SO IS COMPARE
5503  021732  012656                        MOV     (SP)+,@-(SP)    ;BECAUSE OF ADDRESS MODE 5
5504  021734  057636  000000                BIS     @(SP),@(SP)+    ;BECAUSE OF ADDRESS MODE 3
5505  021740  054676  000000                BIS     -(SP),@(SP)     ;BECAUSE OF ADDRESS MODE 7
5506  021744  005006                        CLR     SP
5507  021746  013766  020000  020000        MOV     @#20000,20000(SP)
5508  021754  000425                        BR      3$              ;BRANCH OVER NON KERNEL MODE TESTS
5509
5510                                ;NOTE:  NO OVEFLOW TRAP WILL OCCUR IF NOT IN KERNEL MODE!!!
5511  021756  156737  000175  177777  1$:   BISB    7$+1,@#PSW+1    ;RESTORE MODE BITS IN PSW
5512  021764  012706  000376                MOV     #376,SP         ;SET STACK PTR
5513  021770  016646  177776                MOV     -2(SP),-(SP)    ;SHOULD NOT TRAP
5514  021774  051616                        BIS     (SP),(SP)
5515  021776  061666  177776                ADD     (SP),-2(SP)
5516  022002  105037  177777                CLRB    @#PSW+1         ;SET KERNEL MODE
5517  022006  012706  000700                MOV     #SUPSTK,SP      ;RESTORE THE STACK
5518  022012  000451                        BR      6$              ;EXIT TEST
5519
5520                                ;ERROR SERVICE ROUTINE
5521  022014  012600                2$:     MOV     (SP)+,R0        ;SAVE PC OF INSTRUCTION THAT TRAPPED
```

```
5522  022016  012602                        MOV     (SP)+,R2        ;SAVE PSW
5523  022020  012706  000700                MOV     #SUPSTK,SP      ;SET STACK PTR
5524  022024  104000                        HLT                     ;ERROR! AN INSTRUCTION THAT WAS NOT
5525                                                                  ;SUPPOESED TO TRAP TRAPPED
5526                                                                  ;R0 CONTAINS PC, R2 CONTAINS PSW
5527  022026  000443                        BR      6$              ;EXIT TEST
5528                                  ;THE BELOW INSTRUCTIONS WILL CAUSE A STACK OVERFLOW
5529                                  ;STACK PTR IS AT 376
5530  022030  062737  000066  000004  3$:   ADD     #4$-2$,@#ERRVEC ;SET ERROR VECTOR TO 4$
5531  022036  010306                        MOV     R3,SP           ;SET STACK PTR AT 376
5532  022040  112702  000001                MOVB    #1,R2
5533  022044  005000                        CLR     R0
5534  022046  005016                        CLR     (SP)            ;SETS BIT 0 IN R0
5535  022050  006302                        ASL     R2              ;SHIFT INDICATOR BIT
5536  022052  105226                        INCB    (SP)+           ;SETS BIT 1 IN R0
5537  022054  006302                        ASL     R2
5538  022056  060746                        ADD     PC,-(SP)        ;SETS BIT 2 IN R0
5539  022060  006302                        ASL     R2
5540  022062  000003                        BPT                     ;SETS BIT 3 IN R0
5541  022064  006302                        ASL     R2
5542  022066  004767  000014                JSR     PC,40$          ;SETS BIT 4 IN R0
5543  022072  006302                        ASL     R2
5544  022074  050666  177776                BIS     SP,-2(SP)       ;SETS BIT 5 IN R0
5545  022100  000410                        BR      5$
5546
5547                                  ;PROGRAM WILL TRAP HERE ON OVERFLOW TRAP
5548  022102  050200                  4$:   BIS     R2,R0           ;SET APPROPRIATE BIT IN R0
5549  022104  000002                        RTI                     ;RETURN FROM TRAP
5550
5551  022106  052700  001000          40$:  BIS     #1000,R0                ;SET IND THAT JSR WAS EXECUTED
5552  022112  000207                        RTS     PC
5553
5554  022114  052700  000400          41$:  BIS     #400,R0                 ;SET IND THAT BPT WAS EXECUTED
5555  022120  000002                        RTI
5556
5557                                  ;CHECK THAT ABOVE INSRUCTIONS DID TRAP
5558  022122  012706  000700          5$:   MOV     #SUPSTK,SP      ;SET STACK PTR
5559  022126  022700  001477                CMP     #1477,R0        ;EACH INSTRUCTION SET A BIT IN R0
5560  022132  001401                        BEQ     .+4             ;R0= 1477
5561  022134  104000                        HLT
5562
5563                                  ;EXIT ROUTINE
5564  022136  012706  001200          6$:   MOV     #KERSTK,SP      ;SET KERNEL STACK PTR
5565  022142  016737  177516  000014        MOV     43$,@#BPTVEC    ;RESTORE BPT VECTOR
5566  022150  005037  000016                CLR     @#BPTVEC+2
5567  022154  012746                        MOV     (PC)+,-(SP)     ;PUSH OLD PSW ONTO STACK
5568  022156  000000                  7$:   .WORD   0               ;CONTAINS SAVED PSW
5569  022160  010746                        MOV     PC,-(SP)        ;PUSH CURRENT PC ONTO STACK
5570  022162  062716  000006                ADD     #6,(SP)         ;ADD OFFSET
5571  022166  000002                        RTI
5572  022170  012706  000700                MOV     #SUPSTK,SP      ;SET STACK PTR
5573  022174  012737  053440  000004        MOV     #ERPRT,@#ERRVEC ;RESET TIME OUT VECTOR
5574  022202  013737  177776  000006        MOV     @#PSW,@#ERRVEC+2
5575  022210  052737  000340  000006        BIS     #PR7,@#ERRVEC+2
5576  022216  042737  000020  000006        BIC     #BIT4,@#ERRVEC+2
5577  022224  005037  177766                CLR     @#CPUERR
```

```
5578                                    ;;*****************************************************************
5579                                    ;*TEST 42       CHECK THAT ALL RESERVED INSTRUCTIONS TRAP
5580                                    ;;*****************************************************************
5581  022230  112737  000042  001202  TST42:  MOVB    #42,@#STSTNM            ;LOAD TEST NUMBER
5582  022236  000004                    SCOPE
5583  022240  012702  022344          RESTRP: MOV     #5$,R2          ;GET ADDRESS OF RESERVED INSTRUCTION TABLE
5584  022244  063702  001506                  ADD     @#FACTOR,R2
5585  022250  132737  000040  001471        BITB    #40,@#OPT.CP+1  ;CHECK IF 11/45 FLOATING POINT IS AVAIL.
5586  022256  001402                    BEQ     .+6             ;BRANCH IF NOT AVAILABLE
5587  022260  005067  000110            CLR     50$             ;SET TABLE TERMINATOR AT GROUP 7
5588  022264  012737  022322  000010    MOV     #4$,@#RESVEC    ;SET RESERVED INSTRUCTION TRAP
5589  022272  063737  001506  000010    ADD     @#FACTOR,@#RESVEC
5590  022300  012203                  1$:     MOV     (R2)+,R3        ;GET FIRST RESERVED INSTRUCTION
5591  022302  001437                    BEQ     7$              ;0 TERMINATES THE TABLE
5592  022304  012204                    MOV     (R2)+,R4        ;GET LAST RESERVED INSTRUCTION IN GROUP
5593  022306  010317                  2$:     MOV     R3,(PC)         ;EXECUTE RESERVED INSTRUCTION
5594  022310  000000                  3$:     .WORD   0               ;CONTAINS RESERVED INSTRUCTION
5595  022312  104000                    HLT                     ;ERROR! INSTRUCTION IN R3
5596  022314  104000                    HLT                     ;(2$) ABOVE FAILED TO CAUSE A
5597  022316  104000                    HLT                     ;RESERVED INSTRUCTION TRAP
5598  022320  000405                    BR      41$
5599  022322  012716  022334          4$:     MOV     #41$,(SP)       ;ADJUST RETURN PC
5600  022326  063716  001506                  ADD     @#FACTOR,(SP)   ;TO RETURN TO 41$
5601  022332  000002                    RTI                     ;RETURN TO 41$
5602  022334  020304                  41$:    CMP     R3,R4           ;HAS GROUP OF RESERVED INSTRUCTIONS
5603  022336  001760                    BEQ     1$              ;BEEN EXECUTED
5604  022340  005203                    INC     R3              ;INCREMENT THIS RESERVED INSTRUCTION
5605  022342  000761                    BR      2$              ;TO NEXT ONE AND EXECUTE
5606                                  ;TABLE OF 11/40,11/45 RESERVED INSTRUCTIONS (0 TERMINATES THE TABLE)
5607  022344  000007                  5$:     7               ;GROUP 1
5608  022346  000077                          77              ;  "
5609  022350  000210                          210             ;GROUP 2
5610  022352  000227                          227             ;  "
5611  022354  007000                          7000            ;GROUP 3
5612  022356  007777                          7777            ;  "
5613  022360  075040                          75040           ;GROUP 4
5614  022362  076777                          76777           ;  "
5615  022364  106400                          106400          ;GROUP 5
5616  022366  106477                          106477          ;  "
5617  022370  106700                          106700          ;GROUP 6
5618  022372  107777                          107777          ;  "
5619  022374  170000                  50$:    170000          ;GROUP 7        FLOATING POINT
5620  022376  177777                          177777          ;  "            INSTRUCTIONS
5621  022400  000000                          0               ;0 TERMINATES THE TABLE
5622
5623  022402  012737  053366  000010  7$:     MOV     #RESERR,@#RESVEC        ;RESTORE RESERVED TRAP
5624                                  ;;*****************************************************************
5625                                  ;*TEST 43       CHECK THAT ALL BITS IN THE PSW CAN BE SET AND CLEARED
5626                                  ;;*****************************************************************
5627  022410  112737  000043  001202  TST43:  MOVB    #43,@#STSTNM            ;LOAD TEST NUMBER
5628  022416  000004                    SCOPE
5629  022420  105737  001503          PSWCHK: TSTB    @#MMON          ;IF MEM MGMT IS ON SKIP THIS TEST
5630  022424  001065                    BNE     4$
5631  022426  013767  177776  000132        MOV     @#PSW,3$        ;SAVE STATUS
5632  022434  005037  177776            CLR     @#PSW           ;CLEAR MODE BITS IN PSW
5633  022440  004737  052336            JSR     PC,@#CLRTBIT    ;GO CLEAR 'T' BIT IF SET
```

```
5634  022444  013746  000016                    MOV     @#TBITVEC+2,-(SP)
5635  022450  012704  177776                    MOV     #PSW,R4         ;LOAD ADDRESS OF PSW INTO R4
5636  022454  000250                            CLN
5637  022456  005714                            TST     (R4)            ;CHECK THAT PSW WAS CLEARED
5638  022460  001401                            BEQ     .+4
5639  022462  104000                            HLT                     ;ERROR! PSW FAILED TO CLEAR
5640  022464  012700  170357                    MOV     #170357,R0
5641  022470  052700  170000                    BIS     #170000,R0      ;SET BITS 15-12 IF MEM MGMT
5642  022474  012702  000001          10$:      MOV     #1,R2           ;R2 = TEST BIT
5643  022500  030200                    1$:     BIT     R2,R0           ;CHECK IF BIT CAN BE SET/CLEARED
5644  022502  001423                            BEQ     2$
5645  022504  005037  000016                    CLR     @#TBITVEC+2
5646  022510  030227  000020                    BIT     R2,#20          ;CHECK IF TEST WILL SET 'T' BIT
5647  022514  001403                            BEQ     20$
5648  022516  012737  000002  000016            MOV     #RTI,@#TBITVEC+2;SET RTI INTO RETURN
5649  022524  005014                    20$:    CLR     (R4)            ;CLEAR PSW
5650  022526  050214                            BIS     R2,(R4)         ;SET R2 INTO PSW
5651  022530  011403                            MOV     (R4),R3         ;GET BIT
5652  022532  020203                            CMP     R2,R3           ;CHECK THAT BIT WAS SET IN PSW
5653  022534  001401                            BEQ     .+4
5654  022536  104000                            HLT                     ;ERROR! BIT IN R2 FAILED TO SET IN PSW
5655  022540  000244                            CLZ                     ;CLEAR Z BIT
5656  022542  040214                            BIC     R2,(R4)         ;CLEAR BIT IN PSW
5657  022544  011403                            MOV     (R4),R3         ;GET PSW RESULT
5658  022546  001401                            BEQ     2$              ;BRANCH IF BIC ABOVE CLEARED BIT IN PSW
5659  022550  104000                            HLT                     ;ERROR! BIT IN R2 FAILED TO CLEAR IN PSW
5660  022552  006302                    2$:     ASL     R2              ;SHIFT TEST BIT
5661  022554  103351                            BCC     1$              ;BRANCH IF ALL BITS NOT TESTED
5662  022556  005014                            CLR     (R4)            ;CLEAR STATUS
5663  022560  012637  000016                    MOV     (SP)+,@#TBITVEC+2       ;RESTORE T BIT RETURN
5664  022564  012746                            MOV     (PC)+,-(SP)     ;PUSH ORIGINAL STATUS ON STACK
5665  022566  000000                    3$:     .WORD   0               ;CONTAINS ORIGINAL PSW
5666  022570  010746                            MOV     PC,-(SP)        ;SET RETURN PC
5667  022572  062716  000006                    ADD     #6,(SP)
5668  022576  000002                            RTI                     ;RETURN
5669  022600  013704  177776            4$:     MOV     @#PSW,R4        ;SAVE PSW IN R4
5670  022604  112737  000340  177776            MOVB    #340,@#PSW      ;SET PRIORITY LEVEL 7
5671  022612  004737  052336                    JSR     PC,@#CLRTBIT    ;GO CLEAR 'T' BIT IF SET
5672                                    ;;======================================================================
5673                                    ;*TEST 44       CHECK THAT ALL BITS IN THE CURRENT STACK PTR CAN BE SET CLEARED
5674                                    ;;======================================================================
5675  022616  112737  000044  001202  TST44:  MOVB    #44,@#$TSTNM            ;LOAD TEST NUMBER
5676  022624  000004                            SCOPE
5677  022626  010603                    CHKSP:  MOV     SP,R3           ;SAVE STACK PTR
5678  022630  000257                            CCC
5679  022632  112706  000377                    MOVB    #377,SP         ;SET STACK PTR = -1
5680  022636  006006                    1$:     ROR     SP              ;ROTATE 0 BIT THROUGH ALL BIT
5681  022640  103776                            BCS     1$              ;BIT POSITIONS
5682  022642  005206                            INC     SP              ;SHOULD INCREMENT SP TO 0
5683  022644  001403                            BEQ     2$
5684  022646  010602                            MOV     SP,R2           ;SAVE ERROR STACK PTR
5685  022650  010306                            MOV     R3,SP           ;SET STACK PTR FOR TRAP
5686  022652  104000                            HLT                     ;ERROR!
5687
5688  022654  010306                    2$:     MOV     R3,SP           ;RESTORE ORIGINAL STACK PTR
5689
```

```
                                ;CHECK BYTE OPERATIONS USING THE STACK
022656  010600          SPCHK:  MOV     SP,R0           ;SAVE STACK PTR
022660  010003                  MOV     R0,R3

022662  005043                  CLR     -(R3)
022664  112746  177777          MOVB    #-1,-(SP)       ;(SP) = 377
022670  022713  000377          CMP     #377,(R3)       ;CHECK THAT ONLY EVEN BYTE WAS AFFECTED
022674  001002                  BNE     1$
022676  020306                  CMP     R3,SP           ;CHECK AUTO-DEC
022700  001401                  BEQ     .+4
022702  104000          1$:     HLT

022704  105226                  INCB    (SP)+
022706  005723                  TST     (R3)+           ;CHECK RESULT
022710  001002                  BNE     2$
022712  020006                  CMP     R0,SP           ;CHECK AUTO-INC
022714  001401                  BEQ     .+4
022716  104000          2$:     HLT

022720  005143                  COM     -(R3)           ;(R3)=177777
022722  144613                  BICB    -(SP),(R3)
022724  022713  177400          CMP     #177400,(R3)    ;CHECK RESULT
022730  001002                  BNE     3$
022732  020603                  CMP     SP,R3
022734  001401                  BEQ     .+4
022736  104000          3$:     HLT

022740  132627  000377          BITB    (SP)+,#377
022744  001002                  BNE     4$
022746  020600                  CMP     SP,R0
022750  001401                  BEQ     .+4
022752  104000          4$:     HLT

022754  012746  000001          MOV     #1,-(SP)
022760  062706  000002          ADD     #2,SP
022764  012702  177401          MOV     #177401,R2
022770  120246                  CMPB    R2,-(SP)
022772  001004                  BNE     5$
022774  122602                  CMPB    (SP)+,R2
022776  001002                  BNE     5$
023000  020006                  CMP     R0,SP
023002  001401                  BEQ     .+4
023004  104000          5$:     HLT
023006  105037  177776          CLRB    @#PSW
023012  010446                  MOV     R4,-(SP)        ;RESTORE ORIGINAL PSW TO STACK
023014  010746                  MOV     PC,-(SP)
023016  062716  000006          ADD     #6,(SP)
023022  000002                  RTI
                        ;;**************************************************************
                        ;*TEST 45        CHECK THAT 'C' BIT SETS/CLEARS PROPERLY
                        ;;**************************************************************
023024  112737  000045  001202  TST45:  MOVB    #45,@#$TSTNM            ;LOAD TEST NUMBER
023032  000004                  SCOPE
023034  012727  177776          CBIT:   MOV     #177776,(PC)+   ;LOAD CONSTANT
023040  000000                  1$:     .WORD   0
023042  010700                  MOV     PC,R0           ;GET CURRENT PC
```

```
5746  023044  162700  000004              2$:   SUB     #4,R0           ;POINT R0 TO 1$ ABOVE
5747  023050  005520                            ADC     (R0)+           ;ADD 'C' BIT TO 1$ ABOVE
5748  023052  006340                            ASL     -(R0)           ;SHIFT 1$
5749  023054  102375                            BVC     2$              ;UNTIL 'V' BIT SETS
5750  023056  022767  077776  177754            CMP     #077776,1$      ;CHECK RESULT
5751  023064  001401                            BEQ     .+4
5752  023066  104000                            HLT                     ;ERROR! INCORRECT RESULT IN 1$ ABOVE
5753                                                                    ;R0=ADDRESS OF DATA
5754
5755                                    ;CHECK THAT CONDITION CODES ARE SET PROPERLY WHEN A NUMBER (CURRENT PC)
5756                                    ;AND THAT NUMBER +1 ARE COMPARED, AND VICE VERSA.
5757  023070  010700                    CMPN:   MOV     PC,R0           ;GET CURRENT PC
5758  023072  010002                            MOV     R0,R2           ;SAVE IN R2
5759  023074  005202                            INC     R2              ;MAKE R2 = R0+1
5760  023076  000277                            SCC
5761  023100  000251                            +CLC!CLN                ;CLEAR C & N BITS
5762  023102  020002                            CMP     R0,R2           ;COMPARE # WITH #+1
5763  023104  103003                            BCC     1$              ;CARRY BIT SHOULD SET
5764  023106  102402                            BVS     1$              ;V BIT SHOULD CLEAR
5765  023110  001401                            BEQ     1$              ;Z BIT SHOULD CLEAR
5766  023112  100401                            BMI     .+4             ;N BIT SHOULD SET
5767  023114  104000                    1$:     HLT                     ;ERROR! COMPARE # WITH #+1 FAILED TO
5768                                                                    ;SET CONDITION CODES IN PSW CORRECTLY
5769
5770  023116  000277                            SCC                     ;SET CONDITION CODES IN PSW
5771  023120  120200                            CMPB    R2,R0           ;COMPARE #+1 WITH #
5772  023122  103403                            BCS     2$              ;C BIT SHOULD CLEAR
5773  023124  102402                            BVS     2$              ;V BIT SHOULD CLEAR
5774  023126  001401                            BEQ     2$              ;Z BIT SHOULD CLEAR
5775  023130  100001                            BPL     .+4             ;N BIT SHOLD CLEAR
5776  023132  104000                    2$:     HLT                     ;ERROR! COMPARE #+1 WITH # FAILED TO SET
5777                                                                    ;CONDITION CODES IN PSW CORRECTLY
5778  023134  105037  177776                    CLRB    @#PSW           ;ENSURE PRIORITY 0
5779  023140  000004                    RELE4:  SCOPE
5780  023142  010702                            MOV     PC,R2
5781  023144  062702  000012                    ADD     #12,R2
5782  023150  012707  034240                    MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
5783  023154  000000                    REL44:  .WORD   0
5784                                    ;4444444444444 LAST ADDRESS OF CODE TO BE RELOCATED 444444444444
5785
5786                                    ;;*****************************************************************
5787                                    ;*TEST 46        CHECK EXTENDED INSTRUCTION SET
5788                                    ;;*****************************************************************
5789  023156  112737  000046  001202    TST46:  MOVB    #46,@#$TSTNM            ;LOAD TEST NUMBER
5790  023164  012767  000001  156124            MOV     #1,$TIMES       ;;DO 1 ITERATION
5791  023172  000004                            SCOPE
5792
5793                                            .SBTTL  START OF SECTION 5
5794                                    ;5555555555555 FIRST ADDRESS TO BE RELOCATED 555555555
5795  023174  010700                    REL5:   MOV     PC,R0           ;GET PC
5796  023176  005740                            TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL5
5797  023200  010037  001512                    MOV     R0,@#FRSTAD     ;SAVE
5798  023204  010700                            MOV     PC,R0           ;GET CURRENT PC
5799  023206  162700  023206                    SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
5800  023212  010037  001506                    MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
5801  023216  010737  001212                    MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
```

```
5802  023222  062737  000030  001212          ADD     #30,@#$LPERR    ;ADJUST
5803  023230  013737  001212  001210          MOV     @#$LPERR,@#$LPADR
5804  023236  105737  001502                  TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
5805  023242  001402                          BEQ     .+6
5806  023244  000167  001454                  JMP     RELES
5807  023250  005000                EXTINST:  CLR     R0
5808  023252  000277                          SCC                     ;PRESET CC'S
5809  023254  006700                          SXT     R0              ;EXTEND SIGN (1) INTO R0
5810  023256  103005                          BCC     SXT0            ;CHECK RESULT CC'S
5811  023260  102404                          BVS     SXT0
5812  023262  001403                          BEQ     SXT0
5813  023264  100002                          BPL     SXT0
5814  023266  005200                          INC     R0              ;CHECK RESULT
5815  023270  001401                          BEQ     .+4
5816  023272  104000                SXT0:     HLT
5817
5818  023274  010700                          MOV     PC,R0
5819  023276  010002                          MOV     R0,R2
5820  023300  012703  177777                  MOV     #-1,R3
5821  023304  005102                          COM     R2
5822  023306  000243                          +CLV!CLC                ;CLEAR C AND V BITS
5823  023310  074003                          XOR     R0,R3           ;R3 SHOULD CONTAIN COMPLEMENT OF R0
5824  023312  103404                          BCS     XOR0            ;CHECK THAT C WAS NOT AFFECTED
5825  023314  102403                          BVS     XOR0            ;AND THAT V WAS CLEARED
5826  023316  001402                          BEQ     XOR0
5827  023320  020203                          CMP     R2,R3           ;CHECK RESULT
5828  023322  001401                          BEQ     .+4
5829  023324  104000                XOR0:     HLT                     ;ERROR!   XOR FAILED
5830
5831  023326  010700                          MOV     PC,R0
5832  023330  022020                          CMP     (R0)+,(R0)+     ;SET ADDRESS REGISTER
5833  023332  000401                          BR      1$              ;RESERVE WORD FOR TEST DATA
5834  023334  000000                          .WORD   0               ;CONTAINS TEST DATA
5835  023336  005700                1$:       TST     R0              ;EXTEND SIGN OF ADDRESS INTO
5836  023340  006710                          SXT     (R0)            ;ADDRESS (R0)=-1 IF MSB R0=1
5837  023342  005002                          CLR     R2              ;OTHERWISE, (R0)=0
5838  023344  005700                          TST     R0              ;CHECK SIGN OF ADDRESS
5839  023346  100001                          BPL     .+4
5840  023350  005102                          COM     R2              ;COMPLEMENT CHECK REG IF NEG
5841  023352  021002                          CMP     (R0),R2         ;CHECK RESULT OF SXT
5842  023354  001401                          BEQ     .+4
5843  023356  104000                SXT1:     HLT                     ;ERROR!   SXT FAILED TO EXTEND SIGN PROPERLY
5844
5845  023360  012710  100000                  MOV     #100000,(R0)    ;PRESET DATA
5846  023364  011002                          MOV     (R0),R2
5847  023366  000277                          SCC                     ;PRESET CC'S
5848  023370  074210                          XOR     R2,(R0)         ;XOR 100000 WITH 100000 RESULT = 0
5849  023372  103007                          BCC     XOR1            ;CHECK CC'S AFTER XOR
5850  023374  102406                          BVS     XOR1
5851  023376  001005                          BNE     XOR1
5852  023400  100404                          BMI     XOR1
5853  023402  005710                          TST     (R0)            ;CHECK RESULT (0)
5854  023404  001002                          BNE     XOR1
5855  023406  005402                          NEG     R2              ;CHECK THAT REG WAS NOT AFFECTED
5856  023410  102401                          BVS     .+4
5857  023412  104000                XOR1:     HLT
```

```
5858
5859  023414  010702                          MOV     PC,R2
5860  023416  022222                          CMP     (R2)+,(R2)+
5861  023420  000401                          BR      SXT4            ;PRESERVE WORD FOR DATA
5862  023422  000000                          .WORD   0               ;RESERVED FOR DATA
5863  023424  012722  125252          SXT4:   MOV     #125252,(R2)+   ;PRESET DATA
5864  023430  006742                          SXT     -(R2)           ;EXTEND SIGN
5865  023432  074722                          XOR     PC,(R2)+
5866  023434  010700                          MOV     PC,R0           ;GET PC
5867  023436  005740                          TST     -(R0)           ;SUBTRACT 2 FROM PC
5868  023440  005100                          COM     R0              ;R0=RESULT OF XOR PC-1 ABOVE
5869  023442  074042                          XOR     R0,-(R2)        ;CHECK RESULT OF SXT AND XOR ABOVE
5870  023444  001401                          BEQ     .+4
5871  023446  104000          XOR24:  HLT                             ;ERROR!   SXT & XOR ABOVE INCORRECT
5872
5873  023450  012704  000001                  MOV     #1,R4           ;SET R4
5874  023454  006767  000060                  SXT     XOR6A           ;PRESET DATA=0
5875  023460  074467  000054          2$:     XOR     R4,XOR6A
5876  023464  100423                          BMI     XOR6
5877  023466  006304                          ASL     R4              ;SHIFT R4
5878  023470  102373                          BVC     2$              ;UNTIL V SETS (R4=100000)
5879  023472  100020                          BPL     XOR6            ;BRANCH IF 'N' IS CLEAR
5880  023474  074467  000040                  XOR     R4,XOR6A        ;XOR6A=177777
5881  023500  100015                          BPL     XOR6
5882  023502  074767  000032                  XOR     PC,XOR6A        ;XOR PC WITH XOR6A (177777)
5883  023506  010767  000030                  MOV     PC,XOR6B        ;FORM PC AS USED IN XOR ABOVE
5884  023512  162767  000004  000022          SUB     #4,XOR6B
5885  023520  005167  000016                  COM     XOR6B
5886  023524  026767  000012  000006          CMP     XOR6B,XOR6A     ;XOR6A SHOULD = COMPLEMENT OF PC
5887  023532  001401                          BEQ     .+4
5888  023534  104000          XOR6:   HLT                             ;ERROR!  XOR TESTS ABOVE FAILED
5889
5890  023536  000402                          BR      .+6
5891
5892  023540  000000          XOR6A:  .WORD   0               ;CONTAINS DATA USED BY TEST ABOVE
5893  023542  000000          XOR6B:  .WORD   0
5894
5895
5896  023544  012700  077777                  MOV     #077777,R0      ;SET SOURCE OPERAND FOR ADD
5897  023550  006767  177764                  SXT     XOR6A           ;CLEAR XOR6A
5898  023554  001004                          BNE     SXT6            ;CHECK CC'S AFTER EXTENDING ZERO'S
5899  023556  100403                          BMI     SXT6
5900  023560  103402                          BCS     SXT6
5901  023562  102401                          BVS     SXT6
5902  023564  000401                          BR      .+4
5903  023566  104000          SXT6:   HLT                             ;ERROR!  SXT FAILED
5904
5905  023570  012702  000001                  MOV     #1,R2           ;SET DEST OPERAND FOR ADD
5906  023574  013703  001506                  MOV     @#FACTOR,R3     ;LOAD INDEX REGISTER
5907  023600  060002                          ADD     R0,R2           ;RESULT OF ADD=100000
5908  023602  006763  023540                  SXT     XOR6A(3)        ;EXTEND SIGN OF ADD ABOVE
5909  023606  001403                          BEQ     SXT6A
5910  023610  005267  177724                  INC     XOR6A           ;CHECK RESULT OF SXT
5911  023614  001401                          BEQ     .+4
5912  023616  104000          SXT6A:  HLT                             ;ERROR!  SXT ABOVE FAILED TO EXTEND
5913                                                                  ;SIGN
```

```
5914  023620  010703                          MOV     PC,R3
5915  023622  000402                          BR      .+6             ;PRESERVE 2 WORDS FOR DATA
5916  023624  000000                  SXRA:   .WORD   0               ;RESERVED WORD FOR DATA
5917  023626  000000                  SXRB:   .WORD   0               ;RESERVED WORD FOR DATA
5918  023630  005723                          TST     (R3)+
5919  023632  010304                          MOV     R3,R4           ;R3 = ADDRESS OF SXRA
5920  023634  000250                          CLN                     ;CLEAR N BIT
5921  023636  006724                          SXT     (R4)+           ;EXTEND ZEROS INTO SXRA
5922  023640  001401                          BEQ     .+4
5923  023642  104000                  SXT2:   HLT                     ;ERROR!  SXT FAILED
5924
5925  023644  010467  177754                  MOV     R4,SXRA         ;SXRA = ADDRESS OF SXRB
5926  023650  000257                          CCC                     ;CLEAR CONDITION CODES
5927  023652  006733                          SXT     @(R3)+          ;EXTEND ZEROS INTO SXRB
5928  023654  001401                          BEQ     .+4
5929  023656  104000                  SXT3:   HLT                     ;ERROR!
5930
5931  023660  000270                          SEN                     ;SET N BIT
5932  023662  006753                          SXT     @-(R3)          ;EXTEND ONES INTO SXRB
5933  023664  100401                          BMI     .+4
5934  023666  104000                  SXT5:   HLT                     ;ERROR!
5935
5936  023670  012704  025252                  MOV     #025252,R4      ;R4 = 025252
5937  023674  074433                          XOR     R4,@(R3)+       ;SXRB = 152525 (COMPLEMENT OF R4)
5938  023676  005002                          CLR     R2
5939  023700  074253                          XOR     R2,@-(R3)       ;SXRB REMAINS UNCHANGED
5940  023702  001405                          BEQ     XOR35           ;CHECK CONDITION CODES
5941  023704  100004                          BPL     XOR35
5942  023706  005104                          COM     R4              ;R4 = 152525
5943  023710  020467  177712                  CMP     R4,SXRB         ;CHECK XOR
5944  023714  001401                          BEQ     .+4
5945  023716  104000                  XOR35:  HLT                     ;ERROR!  XOR FAILED
5946
5947  023720  005743                          TST     -(R3)           ;R3 = ADDRESS OF SXRA-2
5948  023722  000250                          CLN                     ;CLEAR N BIT
5949  023724  006773  000002                  SXT     @2(R3)          ;SXRB = 0
5950  023730  001401                          BEQ     .+4
5951  023732  104000                  SXT7:   HLT                     ;ERROR!  SXT FAILED
5952
5953  023734  074473  000002                  XOR     R4,@2(R3)       ;SXRB = R4
5954  023740  020473  000002                  CMP     R4,@2(R3)       ;CHECK XOR
5955  023744  001401                          BEQ     .+4
5956  023746  104000                  XOR7:   HLT                     ;ERROR!  XOR FAILED
5957                                  ;;**************************************************************
5958                                  ;*TEST 47        SOB TEST
5959                                  ;*      NOTE:   DO NOT INSERT ANY CODE IN FOLLOWING SOB TESTS
5960                                  ;               SINCE IT TESTS THE MAXIMUM BRANCH WIDTH OF THE INSTRUCTION.
5961                                  ;;**************************************************************
5962  023750  112737  000047  001202  TST47:  MOVB    #47,@#$TSTNM            ;LOAD TEST NUMBER
5963  023756  000004                          SCOPE
5964
5965  023760  005005                          CLR     R5              ;CLEAR ERROR INDICATOR
5966  023762  000407                          BR      SOB0            ;BRANCH TO SOB TEST
5967
5968  023764  005004                  SOB10:  CLR     R4              ;R4 = 0
5969  023766  005705                          TST     R5              ;CHECK ERROR INDICATOR
```

```
5970  023770  001401                        BEQ     .+4             ;SOB BRANCHED CORRECTLY
5971  023772  104000                        HLT                     ;ERROR!
5972
5973  023774  005005            SOB9:       CLR     R5              ;CLEAR INDICATOR (R5)
5974  023776  006004                        ROR     R4              ;ROTATE RIGHT R4
5975  024000  000467                        BR      SOB8
5976
5977  024002  012700  000010    SOB0:       MOV     #10,R0          ;R0=10
5978  024006  000277                        SCC                     ;SET CONDITION CODES
5979  024010  001012            SOB1:       BNE     SOB2            ;CHECK CONDITION CODES AFTER SOB
5980  024012  100011                        BPL     SOB2            ;SOB SHOULD NOT EFFECT THE
5981  024014  102010                        BVC     SOB2            ;CONDITION CODES.
5982  024016  103007                        BCC     SOB2
5983  024020  077005                        SOB     R0,SOB1
5984  024022  001005                        BNE     SOB2            ;CHECK CONDITION CODES AFTER
5985  024024  100004                        BPL     SOB2            ;SOB FALLS THROUGH,
5986  024026  102003                        BVC     SOB2            ;SOB SHOULD NOT EFFECT
5987  024030  103002                        BCC     SOB2            ;CONDITION CODES.
5988  024032  005700                        TST     R0              ;CHECK IF R0=0
5989  024034  001401                        BEQ     .+4
5990  024036  104000            SOB2:       HLT                     ;ERROR!
5991
5992  024040  012702  000100                MOV     #100,R2         ;R2=100
5993  024044  012700  000101                MOV     #101,R0         ;SET CHECK REGISTER, R0=101
5994  024050  001414            SOB3:       BEQ     SOB4            ;CHECK CONDITION CODES AFTER
5995  024052  100413                        BMI     SOB4            ;SOB BRANCH,
5996  024054  102412                        BVS     SOB4            ;SOB SHOULD NOT EFFECT
5997  024056  103411                        BCS     SOB4            ;CONDITION CODES.
5998  024060  005300                        DEC     R0              ;DECREMENT CHECK REGISTER
5999  024062  020002                        CMP     R0,R2           ;CHECK THAT SOB DECREMENTS
6000  024064  001006                        BNE     SOB4
6001  024066  000257                        CCC                     ;SET CONDITION CODES BEFORE SOB
6002  024070  077211                        SOB     R2,SOB3         ;BRANCH TO SOB3 UNTIL R2=0
6003  024072  001403                        BEQ     SOB4            ;CHECK CONDITION CODES AFTER
6004  024074  100402                        BMI     SOB4            ;SOB FALLS THROUGH
6005  024076  005702                        TST     R2              ;CHECK IF R2=0
6006  024100  001401                        BEQ     .+4
6007  024102  104000            SOB4:       HLT                     ;ERROR!
6008
6009  024104  012700  000001    SOB5:       MOV     #1,R0           ;R0=1
6010  024110  000401                        BR      .+4
6011  024112  104000                        HLT                     ;ERROR!
6012  024114  077002                        SOB     R0,.-2          ;SOB SHOULD NOT BRANCH
6013
6014  024116  005700                        TST     R0              ;CHECK IF R0=0 AFTER SOB
6015  024120  001401                        BEQ     .+4
6016  024122  104000                        HLT                     ;ERROR!
6017
6018  024124  012704  100000    SOB5A:      MOV     #100000,R4      ;R4=100000
6019  024130  000403                        BR      1$
6020  024132  005204            3$:         INC     R4              ;R4=100000
6021  024134  100403                        BMI     2$              ;N BIT SHOULD BE SET
6022  024136  104000                        HLT                     ;ERROR!   SOB DID NOT
6023                                                                ;INCREMENT PROPERLY
6024
6025  024140  077404            1$:         SOB     R4,3$           ;SOB SHOULD BRANCH
```

```
6026  024142  104000                          HLT                     ;ERROR!  SOB DID NOT BRANCH
6027
6028  024144  012703  000100          2$:     MOV     #100,R3         ;R3=100
6029  024150  077301                  SOB6:   SOB     R3,SOB6         ;USE SOB TO BRANCH TO ITSELF
6030  024152  005703                          TST     R3              ;CHECK IF R3=0
6031  024154  001703                          BEQ     SOB10
6032  024156  104000                  SOB7:   HLT                     ;ERROR!
6033
6034  024160  005705                  SOB8:   TST     R5              ;CHECK INDICATOR (R5)
6035                                                                  ;IF SOB BRANCHES INCORRECTLY
6036                                                                  ;WHEN CHECKING MAX. BRANCH,
6037                                                                  ;R5 WILL NOT BE CLEARED AT
6038                                                                  ;THIS POINT INDICATING AN ERROR.
6039
6040  024162  001401                          BEQ     .+4             ;BRANCH IF SOB BRANCHES CORRECTLY
6041  024164  104000                          HLT                     ;ERROR!
6042
6043  024166  005205                          INC     R5              ;SET INDICATOR (R5)
6044  024170  077477                          SOB     R4,SOB9         ;TEST MAX. BRANCH OF SOB
6045  024172  005704                          TST     R4              ;CHECK IF R4=0
6046  024174  001401                          BEQ     .+4
6047  024176  104000                          HLT                     ;ERROR!
6048                                ;;*****************************************************************
6049                                ;*TEST 50        CHECK THE MARK INSTRUCTION
6050                                ;;*****************************************************************
6051  024200  112737  000050  001202  TST50:  MOVB    #50,@#$TSTNM                    ;LOAD TEST NUMBER
6052  024206  000004                          SCOPE
6053  024210  010602                  MRKTST: MOV     SP,R2
6054  024212  010705                          MOV     PC,R5           ;THE STACK LOOKS LIKE THIS AFTER
6055  024214  010500                          MOV     R5,R0           ;THE JSR INSTRUCTION
6056  024216  010546                          MOV     R5,-(SP)        ; -2(SP)= R0    THIS IS A
6057  024220  010746                          MOV     PC,-(SP)        ; -4(SP)= PC    STRING
6058  024222  010746                          MOV     PC,-(SP)        ; -6(SP)= PC+2  OF
6059  024224  010746                          MOV     PC,-(SP)        ;-10(SP)= PC+4  FIVE
6060  024226  010746                          MOV     PC,-(SP)        ;-12(SP)= PC+6  DUMMY
6061  024230  010746                          MOV     PC,-(SP)        ;-14(SP)= PC+10 ARGUMENTS
6062  024232  012746  006405                  MOV     #MARK+5,-(SP)   ;-16(SP)= MARK 5
6063  024236  010605                          MOV     SP,R5           ;-20(SP)= PC PUSHED BY JSR
6064  024240  004767  000002                  JSR     PC,MARK1
6065  024244  000403                          BR      .+10
6066  024246  000205                  MARK1:  RTS     R5
6067  024250  104000                          HLT                     ;ERROR! SHOULD BE DOING MARK 5 INST.
6068  024252  000407                          BR      MARKEX
6069  024254  020602                          CMP     SP,R2
6070  024256  001402                          BEQ     .+6
6071  024260  104000                          HLT                     ;ERROR! SP NOT RETURNED TO PROPER
6072  024262  000403                          BR      MARKEX          ;VALUE BY MARK INSTRUCTION
6073  024264  020005                          CMP     R0,R5
6074  024266  001401                          BEQ     .+4
6075  024270  104000                          HLT                     ;ERROR! DID NOT RESTORE R5 FROM STACK
6076  024272  010206                  MARKEX: MOV     R2,SP           ;RESTORE SP
6077                                ;;*****************************************************************
6078                                ;*TEST 51        RTT/RTI TEST
6079                                ;*      RTT/RTI TEST INSURES THAT CP DOES THE INSTRUCTION FOLLOWING
6080                                ;*      AN RTT IF THE "T"BIT IS SET IN THE PSW,BUT DOES HONOR
6081                                ;*      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
                                        ;*****************************************************************
024274  112737  000051  001202  TST51:  MOVB    #51,@#$TSTNM            ;LOAD TEST NUMBER
024302  000004                  SCOPE
024304  013767  177776  000202  RTT1:   MOV     @#PSW,SAVPSW            ;SAVE PSW
024312  032767  000020  000174          BIT     #20,SAVPSW              ;CHECK IF "T"BIT SET
024320  001176                          BNE     RTT2EX                  ;BRANCH TO EXIT
024322  010746                  1$:     MOV     PC,-(SP)                ;GET CURRENT PC
024324  062716  000116                  ADD     #5$-.,(SP)              ;FORM RELOCATED PC
024330  012637  000014                  MOV     (SP)+,@#TBITVEC         ;LOAD INTO TRAP VECTOR
024334  016746  000154                  MOV     SAVPSW,-(SP)            ;GET CURRENT PSW
024340  011637  000016                  MOV     (SP),@#TBITVEC+2
024344  052737  000340  177776          BIS     #PR7,@#PSW              ;SET PRIORITY LEVEL 7
024352  005000                          CLR     R0
024354  052716  000360                  BIS     #PR7+20,(SP)            ;SET "T"BIT IN PSW ON STACK
024360  010746                          MOV     PC,-(SP)                ;PUT THE PC ON THE STACK
024362  062716  000006                  ADD     #6,(SP)                 ;ADJUST PC FOR NEXT INSTRUCTION
024366  000006                  2$:     RTT
024370  005200                          INC     R0                      ;DONE TO SEE IF INSTR. FOLLOWING
                                                                        ;RTT IS EXECUTED IF T-BIT SET
024372  042737  000340  177776          BIC     #PR7,@#PSW              ;SET PRIORITY LEVEL 0
024400  022767  000006  177760          CMP     #RTT,2$
024406  001005                          BNE     3$                      ;CHECK IF INC WAS EXECUTED
024410  022700  177776                  CMP     #177776,R0              ;CHECK IF COM-R0 EXECUTED
024414  001406                          BEQ     4$
024416  104000                          HLT                             ;ERROR!R0 NOT COMPLIMENTED
024420  000415                          BR      6$                      ;EXIT TEST
024422  005700                  3$:     TST     R0                      ;TEST IF TRAPED BEFORE INC INST.
                                                                        ;WAS EXECUTED
024424  001413                          BEQ     6$
024426  104000                          HLT                             ;ERROR!
024430  000411                          BR      6$                      ;EXIT TEST
024432  012767  000002  177726  4$:     MOV     #RTI,2$
024440  000730                          BR      1$
024442  005100                  5$:     COM     R0                      ;RTT CHECK
024444  016766  000044  000002          MOV     SAVPSW,2(SP)
024452  000002                          RTI
024454  012767  000006  177704  6$:     MOV     #RTT,2$
024462  012737  001472  000014          MOV     #$RTRN,@#TBITVEC                ;RESTORE 'T' TRAP VECTOR
```

```
6138  024470  005037  000016                  CLR     @#TBITVEC+2
6139  024474  042737  000360  000016          BIC     #PR7+BIT4,@#TBITVEC+2
6140  024502                          RTT1EX:
6141                                  ;;*****************************************************************
6142                                  ;*TEST 52        SECOND RTT TEST
6143                                  ;;*****************************************************************
6144  024502  112737  000052  001202  TST52:  MOVB    #52,@#$TSTNM            ;LOAD TEST NUMBER
6145  024510  000004                  SCOPE
6146  024512  000401                  BR      RTT2A
6147  024514  000000          SAVPSW: .WORD   0
6148  024516  016700  177772  RTT2A:  MOV     SAVPSW,R0               ;GET SAVED PSW
6149  024522  105000                  CLRB    R0                      ;CLEAR PRIOITY LEVEL,T, AND COND CODES
6150  024524  012702  144000          MOV     #UM+REG,R2
6151  024530  074002                  XOR     R0,R2
6152  024532  001435                  BEQ     2$                      ;USER MODE REG. SET #1 ON
6153  024534  012702  044000          MOV     #SM+REG,R2
6154  024540  074002                  XOR     R0,R2
6155  024542  001447                  BEQ     3$                      ;SUPER MODE REG. SET #1 ON
6156  024544  032700  140000          BIT     #UM,R0
6157  024550  001062                  BNE     RTT2EX
6158
6159                                  ;TEST THAT RTT CLEARS BITS 11,12,13 & PRIORITY LEVEL BITS IN KERNEL MODE
6160  024552  012702  177777          MOV     #-1,R2                  ;KERNEL MODE REG. SET 0 ON
6161  024556  012737  034240  177776  MOV     #PUM+REG+PR5,@#PSW      ;SELECT REG. SET #1
6162  024564  005002                  CLR     R12                     ;SHOULD CLEAR REG #12
6163  024566  012746  000100          MOV     #PR2,-(SP)
6164  024572  010746                  MOV     PC,-(SP)
6165  024574  062716  000006          ADD     #1$-.,(SP)              ;FORM NEW PC
6166  024600  000006                  RTT
6167  024602  013700  177776  1$:     MOV     @#PSW,R0                ;NOW USING REG SET 0
6168  024606  005702                  TST     R2                      ;SHOULD TEST R2 NOT R12
6169  024610  001001                  BNE     4$
6170  024612  104000                  HLT                             ;ERROR!DID NOT CLEAR BIT #11 OF PSW
6171  024614  022700  000100  4$:     CMP     #PR2,R0         ;TESTS THE PSW AFTER THE RTT
6172  024620  001436                  BEQ     RTT2EX
6173  024622  104000                  HLT                             ;ERROR! INCORRECT PSW AFTER THE RTT
6174  024624  000434                  BR      RTT2EX
6175
6176                                  ;TEST TO INSURE THAT RTI DOES NOT CLEAR BITS 11-15 IN USER MODE
6177  024626  052737  030340  177776  2$:     BIS     #PUM+PR7,@#PSW  ;PSW<15-5>=144X
6178  024634  005046                  CLR     -(SP)
6179  024636  010746                  MOV     PC,-(SP)
6180  024640  062716  000006          ADD     #5$-.,(SP)
6181  024644  000002                  RTI                             ;ATTEMPS TO INSERT A PSW OF 0
6182  024646  022737  174340  177776  5$:     CMP     #UM+PUM+REG+PR7,@#PSW   ;SHOULD CHECK AGAINST REG #0
6183  024654  001420                  BEQ     RTT2EX
6184  024656  104000                  HLT                             ;ERROR! RTI CLEARED BITS IN PSW
6185  024660  000416                  BR      RTT2EX
6186
6187                                  ;TEST THAT BITS 11-15 AND PRIORTY BITS ARE NOT ALTERED IN SUPER MODE
6188  024662  052737  030200  177776  3$:     BIS     #PUM+PR4,@#PSW  ;PSW<15-5>=044X
6189  024670  012746  000340          MOV     #PR7,-(SP)
6190  024674  010746                  MOV     PC,-(SP)
6191  024676  062716  000006          ADD     #6$-.,(SP)
6192  024702  000006                  RTT                             ;ATTEMPTS TO CLEAR 11-15 AND ALTER PR
6193
```

```
6194  024704  022737  074200  177776  6$:     CMP     #SM+PUM+REG+PR4,@#PSW
6195  024712  001401                          BEQ     RTT2EX
6196  024714  104000                          HLT                             ;ERROR! RTT ALTERED PR IN
6197                                                                          ;SUPER MODE OR BITS 11-15.
6198  024716  016737  177572  177776  RTT2EX: MOV     SAVPSW,@#PSW
6199  024724  000004                  RELE5:  SCOPE
6200  024726  010702                          MOV     PC,R2
6201  024730  062702  000012                  ADD     #12,R2
6202  024734  012707  034240                  MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
6203  024740  000000                  REL55:  .WORD   0
6204                                  ;55555555555555 LAST ADDRESS OF CODE TO BE RELOCATED 55555555555
6205
6206                                  ;;**********************************************************************
6207                                  ;*TEST 53        CHECK ASH, ASHC, MUL, AND DIV INSTRUCTIONS
6208                                  ;;**********************************************************************
6209  024742  112737  000053  001202  TST53:  MOVB    #53,@#$TSTNM            ;LOAD TEST NUMBER
6210  024750  012767  000001  154340          MOV     #1,$TIMES       ;;DO 1 ITERATION
6211  024756  000004                          SCOPE
6212
6213                                          .SBTTL  START OF SECTION 6
6214                                  ;666666666666 FIRST ADDRESS TO BE RELOCATED 666666666
6215  024760  010700                  REL6:   MOV     PC,R0           ;GET PC
6216  024762  005740                          TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL6
6217  024764  010037  001512                  MOV     R0,@#FRSTAD     ;SAVE
6218  024770  010700                          MOV     PC,R0           ;GET CURRENT PC
6219  024772  162700  024772                  SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
6220  024776  010037  001506                  MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
6221  025002  010737  001212                  MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
6222  025006  062737  000030  001212          ADD     #30,@#$LPERR    ;ADJUST
6223  025014  013737  001212  001210          MOV     @#$LPERR,@#$LPADR
6224  025022  105737  001502                  TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
6225  025026  001402                          BEQ     .+6
6226  025030  000167  002016                  JMP     RELE6
6227  025034  012700  000001          ASHLO:  MOV     #1,R0           ;R0 WILL BE THE SHIFT COUNT
6228  025040  012703  000021                  MOV     #17.,R3         ;MAX SHIFT COUNT
6229  025044  005067  000014          1$:     CLR     2$              ;PRESET SAVED CC'S LOCATION=0
6230  025050  010002                          MOV     R0,R2           ;GET SHIFT COUNT FOR PASS
6231  025052  010705                          MOV     PC,R5           ;R5 & R4 WILL BE DATA SHIFTED BY
6232  025054  010504                          MOV     R5,R4           ;ASH & ASL INSTRUCTIONS
6233  025056  072502                          ASH     R2,R5           ;SHIFT R5
6234  025060  113727  177776                  MOVB    @#PSW,(PC)+     ;SAVE CC'S
6235  025064  000000                  2$:     .WORD   0               ;CONTAINS ASH CC'S IN EVEN BYTE
6236                                                                  ;ASL CC'S IN ODD BYTE
6237  025066  006304                  3$:     ASL     R4              ;SHIFT R4
6238  025070  113746  177776                  MOVB    @#PSW,-(SP)     ;SAVE PSW ON STACK
6239  025074  132716  000002                  BITB    #2,(SP)         ;CHECK IF ASL SET V BIT
6240  025100  001403                          BEQ     30$
6241  025102  152767  000002  177755          BISB    #2,2$+1         ;IF ASL SET V THEN SET V IN 2$+1
6242  025110  112637  177776          30$:    MOVB    (SP)+,@#PSW     ;RESTORE ORIGINAL PSW
6243  025114  077214                          SOB     R2,3$           ;SHIFT R4 R2 TIMES
6244  025116  153767  177776  177741          BISB    @#PSW,2$+1      ;SAVE CC'S AFTER ASL
6245  025124  020504                          CMP     R5,R4           ;CHECK ASH & ASL RESULTS
6246  025126  001004                          BNE     4$
6247  025130  126767  177730  177727          CMPB    2$,2$+1         ;CHECK ASH & ASL CC'S
6248  025136  001401                          BEQ     .+4
6249  025140  104000                  4$:     HLT                     ;ERROR! INCORRECT RESULT OR CC'S
```

```
6250  025142  005200                        INC     R0              ;INCREMENT PASS SHIFT COUNT
6251  025144  020003                        CMP     R0,R3
6252  025146  001336                        BNE     1$
6253
6254  025150  012700  177777       ASHR0:   MOV     #-1,R0          ;R0 = RIGHT SHIFT COUNT FOR PASS
6255  025154  012703  177757               MOV     #-17.,R3        ;MAX SHIFT COUNT
6256  025160  010002               1$:      MOV     R0,R2           ;GET SHIFT COUNT FOR PASS
6257  025162  010705                        MOV     PC,R5           ;R5 & R4 = DATA TO BE SHIFTED
6258  025164  010504                        MOV     R5,R4           ;BY ASH & ASR INSTRUCTIONS
6259  025166  072502                        ASH     R2,R5           ;SHIFT R5 R2 TIMES
6260  025170  113727  177776               MOVB    @#PSW,(PC)+     ;SAVE CC'S IN EVEN BYTE
6261  025174  000000               2$:      .WORD   0               ;CONTAINS ASH CC'S IN EVEN BYTE
6262                                                                ;ASR CC'S IN ODD BYTE
6263  025176  005402                        NEG     R2
6264  025200  006204               3$:      ASR     R4              ;SHIFT R4
6265  025202  077202                        SOB     R2,3$           ;SHIFT R4 R2 TIMES
6266  025204  113767  177776  177763       MOVB    @#PSW,2$+1      ;SAVE CC'S AFTER ASR
6267  025212  142767  000002  177755       BICB    #2,2$+1         ;ASH RIGHT WILL NOT SET V ASR MAY SET V
6268  025220  020504                        CMP     R5,R4           ;CHECK ASH & ASR RESULTS
6269  025222  001004                        BNE     4$
6270  025224  126767  177744  177743       CMPB    2$,2$+1         ;CHECK ASH & ASR CC'S
6271  025232  001401                        BEQ     .+4
6272  025234  104000               4$:      HLT
6273  025236  005300                        DEC     R0              ;DECREMENT PASS SHIFT COUNT
6274  025240  020003                        CMP     R0,R3
6275  025242  001346                        BNE     1$
6276
6277  025244  012746  000037       ASHCLO:  MOV     #31.,-(SP)      ;PUT MAX SHIFT COUNT ON STACK
6278  025250  012746  000001               MOV     #1,-(SP)        ;PUT LEFT SHIFT COUNT ON STACK
6279  025254  011600               1$:      MOV     (SP),R0         ;GET PASS SHIFT COUNT
6280  025256  010705                        MOV     PC,R5           ;CURRENT PC IS DATA TO BE SHIFTED
6281  025260  010503                        MOV     R5,R3           ;ASHC SHIFTS R4,R5;ASL,ROL SHIFTS R2,R3
6282  025262  005004                        CLR     R4
6283  025264  005002                        CLR     R2
6284  025266  073400                        ASHC    R0,R4           ;SHIFT R4 LEFT AS SPECIFIED BY R0
6285  025270  006303               2$:      ASL     R3              ;SHIFT R2,R3 LEFT
6286  025272  006102                        ROL     R2              ;AS SPECIFIED BY R0
6287  025274  077003                        SOB     R0,2$
6288  025276  020402                        CMP     R4,R2           ;CHECK RESULTS
6289  025300  001002                        BNE     3$
6290  025302  020503                        CMP     R5,R3
6291  025304  001401                        BEQ     .+4
6292  025306  104000               3$:      HLT
6293  025310  005216                        INC     (SP)            ;INCREMENT NEXT PASS SHIFT COUNT
6294  025312  021666  000002               CMP     (SP),2(SP)      ;REACHED MAX COUNT (31.)
6295  025316  001356                        BNE     1$
6296  025320  022626                        CMP     (SP)+,(SP)+     ;RESTORE STACK PTR
6297
6298  025322  012746  177740       ASHCR0:  MOV     #-32.,-(SP)     ;PUT MAX RIGHT SHIFT COUNT ON STACK
6299  025326  012746  177777               MOV     #-1,-(SP)       ;PUT PASS SHIFT COUNT ON STACK
6300  025332  011600               1$:      MOV     (SP),R0         ;GET PASS SHIFT COUNT
6301  025334  010702                        MOV     PC,R2           ;R2,R3 & R4,R5 ARE THE DATA REGISTERS
6302  025336  010204                        MOV     R2,R4           ;TO BE SHIFTED BY TEST
6303  025340  005003                        CLR     R3
6304  025342  005005                        CLR     R5
6305  025344  000262                        SEV                     ;SET V BIT IN PSW
```

```
6306  025346  073200                        ASHC    R0,R2           ;SHIFT R2,R3 RIGHT R0 TIMES
6307  025350  102410                        BVS     3$              ;SHIFT RIGHT CLEARS V
6308  025352  005400                        NEG     R0              ;NEGATE SHIFT COUNT FOR SOB
6309  025354  006204              2$:       ASR     R4              ;SHIFT R4,R5 RIGHT R0 TIMES
6310  025356  006005                        ROR     R5
6311  025360  077003                        SOB     R0,2$
6312  025362  020204                        CMP     R2,R4           ;CHECK RESULT
6313  025364  001002                        BNE     3$
6314  025366  020305                        CMP     R3,R5
6315  025370  001401                        BEQ     .+4
6316  025372  104000              3$:       HLT
6317  025374  005316                        DEC     (SP)            ;SET SHIFT COUNT FOR NEXT PASS
6318  025376  021666  000002                CMP     (SP),2(SP)      ;CHECK IF MAX SHIFT COUNT
6319  025402  001353                        BNE     1$
6320  025404  022626                        CMP     (SP)+,(SP)+     ;RESTORE STACK PTR
6321                                  ;;**********************************************************************
6322                                  ;*TEST 54        CHECK MUL
6323                                  ;*      THE BELOW TEST OF THE MUL INSTRUCTION MULTIPLIES THE CURRENT PC
6324                                  ;*      BY 1,2,4,8 ETC AND SHIFTS THE SAME PC VALUE USING AN ASHC LEFT BY
6325                                  ;*      0,1,2,3,ETC AND COMPARES THE RESULTS. CONDITION CODE RESULTS ARE NOT CHECKED.
6326                                  ;;**********************************************************************
6327  025406  112737  000054  001202  TST54:  MOVB    #54,@#$TSTNM            ;LOAD TEST NUMBER
6328  025414  000004                        SCOPE
6329  025416  012700  000001          MUL0:   MOV     #1,R0           ;R0 CONTAINS MULTIPLIER FOR MUL
6330  025422  012706  000700                  MOV     #SUPSTK,SP      ;SETUP THE STACK
6331  025426  005016                        CLR     (SP)            ;(SP) CONTAINS SHIFT VALUE FOR ASHC
6332  025430  010702              1$:       MOV     PC,R2           ;R3,R2 & R5,R4 ARE DATA REGISTERS
6333  025432  010227                        MOV     R2,(PC)+        ;SAVE MULTIPICAND
6334  025434  000000                        .WORD   0               ;CONTAINS ORIGINAL MULTIPICAND
6335  025436  005003                        CLR     R3
6336  025440  005004                        CLR     R4
6337  025442  010205                        MOV     R2,R5           ;FOR MUL AND ASHC
6338  025444  100001                        BPL     .+4             ;IF MULTPICAND IS NEG THEN SET R4 = -1
6339  025446  005104                        COM     R4              ;FOR ASHC
6340  025450  000277                        SCC                     ;PRESET CC'S
6341  025452  070200                        MUL     R0,R2           ;MULTIPLY R2 BY R0 LEAVE PRODUCT
6342                                                                ;IN R2,R3 MSH IN R2,LSH IN R3
6343  025454  102406                        BVS     2$
6344  025456  001405                        BEQ     2$              ;PRODUCT WILL NEVER BE = 0
6345  025460  073416                        ASHC    (SP),R4         ;'MULTIPLY' R4,R5 BY (SP) LEAVE PRODUCT
6346                                                                ;IN R4,R5 MSH IN R4,LSH IN R5
6347  025462  020204                        CMP     R2,R4           ;CHECK MSH RESULT
6348  025464  001002                        BNE     2$
6349  025466  020305                        CMP     R3,R5           ;CHECK LSH RESULT
6350  025470  001401                        BEQ     .+4
6351  025472  104000              2$:       HLT
6352  025474  005216                        INC     (SP)            ;INCREMENT ASHC SHIFT COUNT
6353  025476  006300                        ASL     R0              ;SHIFT MUL MULTIPLIER
6354  025500  102353                        BVC     1$
6355                                  ;CHECK MUL INST WITH MULTIPIER (R0) = 100000
6356  025502  010702                        MOV     PC,R2           ;R2 = MULTIPICAND
6357  025504  005202                        INC     R2
6358  025506  010227                        MOV     R2,(PC)+        ;SAVE MULTIPICAND
6359  025510  000000                        .WORD   0               ;CONTAINS ORIGINAL MULTIPICAND
6360  025512  005103                        COM     R3
6361  025514  010204                        MOV     R2,R4           ;R4 WILL BE MSH 'PRODUCT'
```

```
6362  025516  006204                          ASR     R4              ;FORM 'PRODUCT'
6363  025520  005104                          COM     R4              ;COMPLEMENT MSH 'PRODUCT'
6364  025522  070200                          MUL     R0,R2           ;MULTIPLY R2 BY 100000 LEAVING
6365                                                                  ;R2 = MSH, R3 = LSH PRODUCT
6366  025524  020204                          CMP     R2,R4           ;COMPARE MSH PRODUCTS
6367  025526  001002                          BNE     3$
6368  025530  020003                          CMP     R0,R3           ;CHECK LSH PRODUCT
6369  025532  001401                          BEQ     .+4
6370  025534  104000                  3$:     HLT
6371                                  ;;**********************************************************************
6372                                  ;*TEST 55        CHECK THE DIV INSTRUCTION
6373                                  ;*      THE BELOW TEST OF THE DIV INSTRUCTION DIVIDES THE CURRENT PC BY
6374                                  ;*      1,2,4,8,ETC LEAVING THE QUOTIENT/REMAINDER IN R2/R3. NEXT THE QUOTIENT
6375                                  ;*      IS MULTPLIED BY 1,2,4,8,ETC AND THE REMAINDER ADDED. THE RESULT IS
6376                                  ;*      THEN COMPARED WITH THE ORIGINAL CURRENT PC.
6377                                  ;;**********************************************************************
6378  025536  112737  000055  001202  TST55:  MOVB    #55,@#$TSTNM            ;LOAD TEST NUMBER
6379  025544  000004                          SCOPE
6380  025546  012700  000001          DIV0:   MOV     #1,R0           ;R0=DIVISOR
6381  025552  010716                          MOV     PC,(SP)         ;SAVE DATA ON STACK
6382  025554  011603                  1$:     MOV     (SP),R3         ;GET DATA
6383  025556  005002                          CLR     R2              ;CLEAR MSH DIVIDEND
6384  025560  000277                          SCC
6385  025562  071200                          DIV     R0,R2           ;DIVIDE R2 BY R0 LEAVING QUOTIENT IN R2
6386                                                                  ;AND REMAINDER IN R3
6387  025564  103417                          BCS     2$
6388  025566  100416                          BMI     2$
6389  025570  102007                          BVC     20$             ;BRANCH IF DIVIDE WORKED
6390  025572  022700  000001                  CMP     #1,R0           ;V BIT SHOULD ONLY SET IF DIVIDING BY 1
6391  025576  001012                          BNE     2$              ;AND THE LSH OF DIVIDEND
6392  025600  032716  100000                  BIT     #100000,(SP)    ;IS NEGATIVE
6393  025604  001407                          BEQ     2$
6394  025606  000407                          BR      3$
6395  025610  010204                  20$:    MOV     R2,R4           ;GET QUOTIENT
6396  025612  070400                          MUL     R0,R4           ;MULTIPLY QUOTIENT BY DIVISOR
6397  025614  060305                          ADD     R3,R5           ;ADD REMAINDER TO LSH PRODUCT
6398  025616  103402                          BCS     2$              ;SHOULD BE NO CARRY
6399  025620  021605                          CMP     (SP),R5         ;CHECK RESULT
6400  025622  001401                          BEQ     .+4
6401  025624  104000                  2$:     HLT                     ;ERROR! DIVIDE FAILED
6402                                                                  ;QUOTIENT IS IN R2,REMAINDER IN R3
6403                                                                  ;ORIGINAL PC IS ON STACK AND FINAL
6404                                                                  ;PRODUCT IN R4,R5 [MSH][LSH]
6405  025626  006300                  3$:     ASL     R0              ;GET NEXT DIVISOR
6406  025630  102351                          BVC     1$
6407
6408                                  ;CHECK ASH,ASHC,MUL, AND DIV INSTRUCTIONS USING ADDRESS MODE 1
6409  025632  005016                  ASHL1:  CLR     (SP)            ;(SP) = SHIFT COUNT
6410  025634  005000                          CLR     R0              ;R0 = SHIFT COUNT FOR CHECK ASH
6411  025636  012702  000020                  MOV     #16.,R2         ;R2 = MAX LEFT SHIFT COUNT
6412  025642  005067  000012          1$:     CLR     2$              ;CLEAR CC'S HOLDING ADDRESS
6413  025646  010703                          MOV     PC,R3           ;R3,R4 = DATA TO BE SHIFTED
6414  025650  010304                          MOV     R3,R4
6415  025652  072316                          ASH     (SP),R3         ;SHIFT R3 LEFT (SP) TIMES
6416  025654  013727  177776                  MOV     @#PSW,(PC)+     ;SAVE CC'S
6417  025660  000000                  2$:     .WORD   0               ;CONTAINS ASH (SP),R3 CC'S IN EVEN BYTE
```

```
6418
6419  025662  072400                        ASH     R0,R4           ;AND ASH R0,R4 CC'S IN ODD BYTE
6420  025664  113767  177776  177767        MOVB    @#PSW,2$+1      ;SHIFT R4 LEFT R0 TIMES
6421  025672  020304                        CMP     R3,R4           ;SAVE CC'S IN ODD BYTE OF 2$
6422  025674  001004                        BNE     3$              ;COMPARE RESULTS
6423  025676  126767  177756  177755        CMPB    2$,2$+1         ;BRANCH IF THEY DO NOT COMPARE
6424  025704  001401                        BEQ     .+4             ;CHECK CC'S AFTER ASH INSTRUCTIONS
6425  025706  104000                3$:     HLT                     ;ERROR! EITHER RESULTS OF SHIFT OR
6426                                                                ;RESULT CC'S ARE INCORRECT
6427  025710  005200                        INC     R0              ;INCREMENT SHIFT COUNT FOR ASH R0,R4
6428  025712  005216                        INC     (SP)            ;INCRENENT SHIFT COUNT FOR ASH (SP),R3
6429  025714  020200                        CMP     R2,R0           ;CHECK FOR MAX SHIFT COUNT
6430  025716  001351                        BNE     1$
6431
6432  025720  005016                ASHR1:  CLR     (SP)            ;(SP) = SHIFT COUNT FOR ASH (SP),R4
6433  025722  005000                        CLR     R0              ;R0 = SHIFT COUNT FOR ASH R0,R5
6434  025724  005402                        NEG     R2              ;R2 = MAX RIGHT SHIFT COUNT (SET BY
6435                                                                ;ABOVE TEST TO 16. NOW = -16.
6436  025726  005067  000012        1$:     CLR     2$              ;CLEAR CC'S HOLDING ADDRESS
6437  025732  010704                        MOV     PC,R4           ;R4,R5 = DATA TO BE SHIFTED RIGHT
6438  025734  010405                        MOV     R4,R5
6439  025736  072416                        ASH     (SP),R4         ;SHIFT R4 RIGHT (SP) TIMES
6440  025740  013727  177776                MOV     @#PSW,(PC)+     ;SAVE CC'S
6441  025744  000000                2$:     .WORD   0               ;CONTAINS ASH (SP),R4 CC'S IN EVEN BYTE
6442                                                                ;AND ASH R0,R5 CC'S IN ODD BYTE
6443  025746  072500                        ASH     R0,R5           ;SHIFT R5 RIGHT R0 TIMES
6444  025750  113767  177776  177767        MOVB    @#PSW,2$+1      ;SAVE CC'S IN ODD BYTE 2$
6445  025756  020405                        CMP     R4,R5           ;CHECK RESULTS
6446  025760  001004                        BNE     3$
6447  025762  126767  177756  177755        CMPB    2$,2$+1         ;CHECK RESULT CC'S
6448  025770  001401                        BEQ     .+4
6449  025772  104000                3$:     HLT                     ;ERROR! EITHER RESULTS OR RESULT CC'S
6450                                                                ;DID NOT COMPARE
6451  025774  005300                        DEC     R0              ;DECREMENT SHIFT COUNT
6452  025776  005316                        DEC     (SP)            ;DECREMENT SHIFT COUNT FOR ASH (SP),R4
6453  026000  020002                        CMP     R0,R2           ;CHECK FOR MAX RIGHT SHIFT
6454  026002  001351                        BNE     1$
6455                                ;;**************************************************************
6456                                ;*TEST 56        DIVIDE AGAIN
6457                                ;*      THE BELOW TEST CHECKS THE DIVIDE INSTRUCTION BY DIVIDING
6458                                ;*      THE CURRENT PC BY ITSELF+1. THE QUOTIENT (IN R2) ALWAYS = 0,
6459                                ;*      AND THE REMAINDER (IN R3) ALWAYS = THE CURRENT PC.
6460                                ;;**************************************************************
6461  026004  112737  000056  001202 TST56:  MOVB    #56,@#$TSTNM            ;LOAD TEST NUMBER
6462  026012  000004                        SCOPE
6463  026014  010703                DIV1:   MOV     PC,R3           ;CURRENT PC IS LSH DIVIDEND
6464  026016  006702                        SXT     R2              ;EXTEND SIGN TO R2 (MSH DIVIDEND)
6465  026020  010304                        MOV     R3,R4           ;SAVE ORIGINAL DIVIDEND
6466  026022  010316                        MOV     R3,(SP)         ;PUT ON STACK
6467  026024  005216                        INC     (SP)            ;ADD 1 (WILL BE DIVISOR)
6468  026026  100002                        BPL     1$              ;BRANCH IF POSITIVE
6469  026030  162716  000002                SUB     #2,(SP)         ;MAKE DIVISOR 1 LESS THAN DIVIDEND
6470  026034  071216                1$:     DIV     (SP),R2         ;DIVIDE R2 BY (SP)
6471  026036  103410                        BCS     2$              ;CHECK CONDITION CODES
6472  026040  102407                        BVS     2$
6473  026042  001006                        BNE     2$
```

```
6474  026044  100405                          BMI     2$
6475  026046  005702                          TST     R2              ;CHECK QUOTIENT (R2 = 0)
6476  026050  001361                          BNE     DIV1
6477  026052  010416                          MOV     R4,(SP)         ;GET ORIGINAL DIVISOR
6478  026054  020316                          CMP     R3,(SP)         ;CHECK REMAINDER
6479  026056  001401                          BEQ     .+4
6480  026060  104000                  2$:     HLT                     ;REPORT ERROR
6481                                  ;;**********************************************************************
6482                                  ;*TEST 57        CHECK SPL INSTRUCTION
6483                                  ;;**********************************************************************
6484  026062  112737  000057  001202  TST57:  MOVB    #57,@#$TSTNM            ;LOAD TEST NUMBER
6485  026070  000004                          SCOPE
6486  026072  012702                  SPLTST: MOV     (PC)+,R2        ;R2 CONTAINS OP CODE FOR SPL  7
6487  026074  000237                          SPL     7
6488  026076  005004                          CLR     R4
6489  026100  042744  000340                 BIC     #PR7,-(R4)      ;CLEAR PRIORITY LEVEL BITS IN PSW
6490  026104  011403                          MOV     (R4),R3         ;GET CURRENT PSW
6491  026106  042703  177757                 BIC     #177757,R3      ;R3 CONTAINS CORRECT PSW AFTER SPL
6492
6493  026112  012767  000230  000010          MOV     #SPL+0,2$       ;INITIALIZE SPL INSTRUCTIONS
6494  026120  012767  000237  000050          MOV     #SPL+7,5$
6495  026126  000257                  1$:     CCC                     ;CLEAR CONDITION CODES
6496  026130  000230                  2$:     SPL     0               ;SET PRIORITY LEVEL (NOTE: SPL=NOP IF USER/SUPER MODE)
6497  026132  121403                          CMPB    (R4),R3         ;CHECK RESULT OF SPL ABOVE
6498  026134  001401                          BEQ     .+4
6499  026136  104000                          HLT                     ;ERROR! SPL ABOVE FAILED
6500  026140  032714  140000                 BIT     #UM,(R4)        ;IF NOT IN KERNEL MODE THEN SPL
6501  026144  001002                          BNE     3$              ;ACTS AS A NOP
6502  026146  062703  000040                 ADD     #40,R3          ;SET NEXT CORRECT PSW RESULT
6503  026152  005267  177752          3$:     INC     2$              ;SET NEXT SPL INSTRUCTION
6504  026156  026702  177746                 CMP     2$,R2           ;CHECK IF DONE
6505  026162  002761                          BLT     1$              ;LOOP UNTIL DONE CHANGING SPL EACH PASS
6506  026164  012702                          MOV     (PC)+,R2        ;R2 CONTAINS SPL INSTRUCTION BELOW
6507  026166  000230                          SPL     0
6508  026170  052703  000017                 BIS     #17,R3          ;SET CONDITION CODE RESULT INTO R3
6509  026174  000277                  4$:     SCC                     ;SET CONDITION CODES
6510  026176  000237                  5$:     SPL     7               ;SET PRIORITY LEVEL
6511  026200  121403                          CMPB    (R4),R3         ;CHECK RESULT OF SPL ABOVE
6512  026202  001401                          BEQ     .+4
6513  026204  104000                          HLT                     ;ERROR! SPL ABOVE FAILED
6514  026206  032714  140000                 BIT     #UM,(R4)        ;CHECK IF IN KERNEL MODE
6515  026212  001002                          BNE     6$
6516  026214  162703  000040                 SUB     #40,R3          ;SET NEXT CORRECT PSW RESULT
6517  026220  005367  177752          6$:     DEC     5$              ;SET NEXT SPL
6518  026224  026702  177746                 CMP     5$,R2           ;CHECK IF DONE ALL SPL'S
6519  026230  002361                          BGE     4$
6520                                  ;;**********************************************************************
6521                                  ;*TEST 60        CHECK PIRQ LOGIC
6522                                  ;*      THIS TEST CHECKS THAT WHEN A REQUEST IS MADE AT A LEVEL = TO THE
6523                                  ;*      CURRENT PROCESSER PRIORITY LEVEL THAT NO INTERRUPT TAKES PLACE, AND
6524                                  ;*      THAT WHEN A REQUEST IS MADE AT A LEVEL 1 GREATER THAN THE CURRENT PRO-
6525                                  ;*      CESSER LEVEL THAT AN INTERRUPT OCCURS
6526                                  ;;**********************************************************************
6527  026232  112737  000060  001202  TST60:  MOVB    #60,@#$TSTNM            ;LOAD TEST NUMBER
6528  026240  000004                          SCOPE
6529  026242  012700  026406          PIRQ0:  MOV     #4$,R0          ;R0 POINTS TO A TABLE OF CORRECT PIRQ
```

```
6530
6531  026246  012702  000400                    MOV     #400,R2         ;CONTENTS AFTER AN INTERRUPT
6532  026252  005003                            CLR     R3              ;R2 CONTAINS INTERRUPT REQUEST LEVEL
6533  026254  012704  177772                    MOV     #PIRQ,R4        ;R3 CONTAINS PROCESSER PRIORITY LEVEL
6534  026260  005014                            CLR     (R4)            ;R4 CONTAINS ADDRESS OF PIRQ REGISTER
6535  026262  013737  177776  000242            MOV     @#PSW,@#PIRQVEC+2       ;INITIALZE REQUEST LEVEL TO 0
6536  026270  112737  000340  000242            MOVB    #PR7,@#PIRQVEC+2        ;RETAIN MODE & REG SET ON TRAP
6537  026276  112737  000340  000016            MOVB    #PR7,@#TBITVEC+2        ;ASSUME LEVEL 7 ON INTERRUPT
6538  026304  012737  026344  000240  1$:       MOV     #2$,@#PIRQVEC           ;PRIORITY LEVEL 7 ON TRAP
6539  026312  063737  001506  000240            ADD     @#FACTOR,@#PIRQVEC      ;SET PIRQ ERROR INTERRUPT VECTOR
6540  026320  110337  177776                    MOVB    R3,@#PSW                ;ADD RELOCATION FACTOR
6541  026324  050214                            BIS     R2,(R4)                 ;SET CP PRIORITY LEVEL
6542  026326  100436                            BMI     5$                      ;MAKE REQUEST AT LEVEL = TO CP LEVEL
6543  026330  062737  000002  000240            ADD     #3$-2$,@#PIRQVEC        ;BRANCH WHEN DONE
6544  026336  006302                            ASL     R2                              ;SET PIRQ INTERRUPT VECTOR TO 3$
6545  026340  050214                            BIS     R2,(R4)         ;MAKE REQUEST AT LEVEL 1 HIGHER
6546  026342  000240                            NOP
6547  026344  104000                  2$:       HLT                             ;ERROR! EITHER AN INTERRUPT OCCURED
6548                                                                            ;WHEN RQST LEVEL = CP LEVEL (PIRQVEC)=2$
6549                                                                            ;OR INTERRUPT FAILED (PIRQVEC)=3$
6550  026346  022014                  3$:       CMP     (R0)+,(R4)              ;CHECK CONTENTS OF PIRQ REGISTER
6551  026350  001406                            BEQ     6$
6552  026352  013737  177772  001276            MOV     @#PIRQ,@#STMP0          ;SAVE PIRQ
6553  026360  005037  177772                    CLR     @#PIRQ
6554  026364  104000                            HLT                             ;ERROR! INCORRECT PIRQ CONTENTS
6555  026366  062703  000040            6$:     ADD     #40,R3                  ;SET NEXT CP PRIORITY LEVEL
6556  026372  040214                            BIC     R2,(R4)                 ;LOWER LEVEL BY 1
6557  026374  012716  026304                    MOV     #1$,(SP)                ;ADJUST RETURN ADDRESS
6558  026400  063716  001506                    ADD     @#FACTOR,(SP)           ;TO RETURN TO 1$
6559  026404  000006                  30$:      RTT
6560
6561                                            ;TABLE OF CORRECT PIRQ REGISTER CONTENTS ON INTERRUPT
6562  026406  001042                  4$:       1042                            ;PIR1+PIA1
6563  026410  003104                            3104                            ;PIR2+PIR1+PIA2
6564  026412  007146                            7146                            ;PIR3+PIR2+PIR1+PIA3
6565  026414  017210                            17210                           ;PIR4+PIR3+PIR2+PIR1+PIA4
6566  026416  037252                            37252                           ;PIR5+PIR4+PIR3+PIR2+PIR1+PIA5
6567  026420  077314                            77314                           ;PIR6+PIR5+PIR4+PIR3+PIR2+PIR1+PIA6
6568  026422  177356                            177356                          ;PIR7+PIR6+PIR5+PIR4+PIR3+PIR2+PIR1+PIA7
6569
6570  026424  005014                  5$:       CLR     (R4)                    ;CLEAR PIRQ REGISTER
6571  026426  012737  000242  000240            MOV     #PIRQVEC+2,@#PIRQVEC    ;RESET PIRQVEC TO HALT AT PIRQVEC+2
6572  026434  005037  000242                    CLR     @#PIRQVEC+2
6573  026440  105037  177776                    CLRB    @#PSW
6574  026444  042737  000340  000016            BIC     #PR7,@#TBITVEC+2
6575                                            ;;*****************************************************************
6576                                            ;*TEST 61        CHECK MICRO-BREAK REGISTER
6577                                            ;*      THIS TEST SHIFTS A '0' BIT THRU ALL BIT POSITIONS.
6578                                            ;;*****************************************************************
6579  026452  112737  000061  001202  TST61:   MOVB    #61,@#STSTNM            ;LOAD TEST NUMBER
6580  026460  000004                            SCOPE
6581  026462  012702  177770          MBRK:     MOV     #UBREAK,R2      ;SET ADDRESS OF MICRO BREAK REGISTER
6582  026466  011246                            MOV     (R2),-(SP)      ;SAVE ORIG CONTENTS
6583  026470  012700  000376                    MOV     #376,R0         ;SET DATA PATTERN
6584  026474  010012                  1$:       MOV     R0,(R2)         ;LOAD REGISTER WITH PATTERN
6585  026476  021200                            CMP     (R2),R0         ;AND CHECK
```

```
6586  026500  001004                        BNE     3$              ;BRANCH IF INCORRECT
6587  026502  000261              2$:       SEC                     ;SET 'C'
6588  026504  006100                        ROL     R0              ;SHIFT DATA
6589  026506  103772                        BCS     1$
6590  026510  000402                        BR      4$
6591  026512  104000              3$:       HLT                     ;ERROR DATA IN R0 NOT IN UBREAK REG
6592  026514  000772                        BR      2$              ;CONTINUE TEST
6593  026516  012612              4$:       MOV     (SP)+,(R2)      ;RESTORE ORIG UBREAK CONTENTS
6594                                        ;;**************************************************************
6595                                        ;*TEST 62        CHECK MFPI/MTPI INSTRUCTIONS
6596                                        ;;**************************************************************
6597  026520  112737  000062  001202  TST62:  MOVB    #62,@#STSTNM            ;LOAD TEST NUMBER
6598  026526  000004                        SCOPE
6599  026530  032737  140000  177776  MPI:   BIT     #UM,@#PSW       ;KERNEL MODE?
6600  026536  001545                        BEQ     ENDCP           ;YES EXIT TEST
6601  026540  010746                        MOV     PC,-(SP)
6602  026542  062716  000134                ADD     #5$-.,(SP)
6603  026546  012637  000250                MOV     (SP)+,@#MMVEC   ;SET MEM MGMT ABORT VECTOR
6604  026552  005046                        CLR     -(SP)           ;CLEAR CHECK WORD
6605  026554  010603                        MOV     SP,R3
6606  026556  010346                        MOV     R3,-(SP)        ;PUT ADDRESS OF CHECK WORD ON THE STACK
6607  026560  105737  001503                TSTB    @#MMON          ;CHECK IF MEM MGMT IS ENABLED
6608  026564  001417                        BEQ     1$              ;BRANCH IF OFF
6609  026566  013737  177640  177654        MOV     @#UIPAR0,@#UIPAR6       ;SET UP USER PAGE ADDR. REG.
6610  026574  012737  006006  177614        MOV     #6006,@#UIPDR6          ;SET USER PAGE DESC REG R/W UP 6 PAGES
6611  026602  013737  172240  172254        MOV     @#SIPAR0,@#SIPAR6
6612  026610  012737  006006  172214        MOV     #6006,@#SIPDR6  ;SET SUPER PAGE DESC. REG.
6613  026616  062706  140000          10$:  ADD     #140000,SP      ;SET CURRENT MODE'S STACK POINTER
6614  026622  000240                        NOP
6615  026624  010746                1$:     MOV     PC,-(SP)
6616  026626  062716  000024                ADD     #3$-.,(SP)
6617  026632  012637  000020                MOV     (SP)+,@#IOTVEC  ;SET IOT TRAP VECTOR
6618  026636  000004                        IOT                     ;TRAP TO 3$ BELOW
6619  026640  005266  000002                INC     2(SP)           ;INCREMENT CHECK WORD
6620  026644  001417                        BEQ     6$
6621  026646  104000                4$:     HLT                     ;ERROR! MFPI,MTPI FAILURE-FOR BETTER
6622  026650  000415                        BR      6$              ;ISOLATION SUGGEST RUNNING MFPI DIAG. DCKTD/E
6623  026652  000240                3$:     NOP                     ;PSW=KERNEL MODE,PREV USER OR SUPER MODE
6624  026654  006506                        MFPI    SP              ;GET PREV. MODES STACK POINTER
6625  026656  006536                        MFPI    @(SP)+          ;GET DATA (AN ADDRESS) ON PREV MODE'S STACK
6626  026660  006576  000000                MFPI    @(SP)           ;GET DATA (=0) FROM PREV MODES ADDRESS
6627  026664  000240                        NOP                     ;SPACE AND PUSH ONTO KERNEL STACK
6628  026666  001367                        BNE     4$              ;ERROR IF BRANCH TAKEN! SHOULD HAVE A ZERO ON THE STACK
6629  026670  005116                        COM     (SP)            ;COMPLEMENT OPERAND
6630  026672  006636                        MTPI    @(SP)+          ;POP OPERAND OFF KERNEL STACK AND MOVE
6631                                                                ;IT TO PREV MODE'S SPACE
6632  026674  000002                        RTI                     ;RETURN TO INST FOLLOWING IOT ABOVE
6633  026676  104000                5$:     HLT                     ;ERROR! MEMORY MANG. ABORT
6634  026700  105037  177776                CLRB    @#PSW           ;SET PRIORITY LEVEL BACK TO 0
6635  026704  012737  053272  000250  6$:   MOV     #KTABRT,@#MMVEC ;RESTORE VECTOR
6636  026712  012737  044532  000020        MOV     #$SCOPE,@#IOTVEC
6637  026720  012706  000700                MOV     #SUPSTK,SP      ;RESTORE STACK POINTER
6638                                        ;;**************************************************************
6639                                        ;*TEST 63        CHECK ILLEGAL HALT
6640                                        ;;**************************************************************
6641  026724  112737  000063  001202  TST63:  MOVB    #63,@#STSTNM            ;LOAD TEST NUMBER
```

```
                                        SCOPE
                                HALT1:  MOV     PC,-(SP)        ;GET CURRENT PC
                                        ADD     #2$-.,(SP)
                                        MOV     (SP),@#ERRVEC   ;SET ERROR TRAP VECTOR TO 2$ BELOW
                                        MOV     (SP)+,@#RESVEC  ;LOAD RESERVED INST TRAP VECTOR (11/40)
                                        HALT                    ;SHOULD TRAP TO 4 IN USER/SUPER MODE
                                1$:     HLT                     ;ERROR! HALT ABOVE FAILED IN USER/SUPER MODE
                                        BR      3$
                                2$:     MOV     PC,(SP)         ;REPLACE RETURN PC WITH
                                        ADD     #3$-.,(SP)      ;ADDRESS OF 3$ BELOW
                                        RTI                     ;RETURN (TO 3$)

                                3$:     MOV     #ERPRT,@#ERRVEC ;RESTORE ERROR TRAP VECTOR
                                        MOV     #RESERR,@#RESVEC
                                        CLRB    @#PSW
                                        CLR     @#CPUERR
                        ;;**************************************************************
                        ;*TEST 64       CHECK RESET IN SUPER/USER MODE
                        ;;**************************************************************
                        TST64:  MOVB    #64,@#STSTNM            ;LOAD TEST NUMBER
                                SCOPE
                        RESET1: SCC
                                MOV     @#PSW,R0        ;GET CURRENT PSW
                                SCC
                                RESET
                                CMP     @#PSW,R0        ;CHECK THAT PSW UNCHANGED BY RESET ABOVE
                                BEQ     .+4
                                HLT                     ;ERROR! RESET CLEARED MODE BITS IN PSW
                                MOV     R0,@#PSW        ;RESTORE PSW (FOR ERROR)
                        ENDCP:
                        RELE6:  SCOPE
                                MOV     PC,R2
                                ADD     #12,R2
                                MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
                        REL66:  .WORD   0
                        ;6666666666666 LAST ADDRESS OF CODE TO BE RELOCATED 66666666666


                        ;;**************************************************************
                        ;*TEST 65       TEST STACK LIMIT REGISTER
                        ;;**************************************************************
                        TST65:  MOVB    #65,@#STSTNM            ;LOAD TEST NUMBER
                                MOV     #1,$TIMES       ;;DO 1 ITERATION
                                SCOPE

                                .SBTTL  START OF SECTION 7
                        ;777777777777 FIRST ADDRESS TO BE RELOCATED 777777777
                        REL7:   MOV     PC,R0           ;GET PC
                                TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL7
                                MOV     R0,@#FRSTAD     ;SAVE
                                MOV     PC,R0           ;GET CURRENT PC
                                SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
                                MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
                                MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
                                ADD     #30,@#$LPERR    ;ADJUST
                                MOV     @#$LPERR,@#$LPADR
                                TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
```

```
6698  027154  001402                          BEQ     .+6
6699  027156  000167  001206                  JMP     RELE7
6700                          ;THIS TEST SHIFTS A '1' BIT THROUGH ALL BIT POSITIONS
6701  027162  012702  177774                  MOV     #STKLMT,R2              ;GET ADDRESS OF STACK LIM REG
6702  027166  005022                          CLR     (R2)+                   ;CLEAR STACK LIMIT REG
6703  027170  032712  000020                  BIT     #20,(R2)                ;EXIT TEST IF 'T' BIT IS SET
6704  027174  001116                          BNE     101$
6705  027176  052712  000340                  BIS     #340,(R2)               ;SET PRIORITY LEVEL 7 TO PREVENT
6706                                                                          ;ANY INTERRUPTS FROM OCCURRING
6707  027202  012700  000400                  MOV     #400,R0                 ;SET CHECK DATA
6708  027206  010042                  1$:     MOV     R0,-(R2)                ;MOVE TO STACK LIMIT REG
6709  027210  022200                          CMP     (R2)+,R0                ;AND CHECK RESULT
6710  027212  001401                          BEQ     2$
6711  027214  104000                          HLT                             ;ERROR! STACK LIMIT DID NOT
6712                                                                          ;LOAD CORRECTLY. CORRECT RESULT
6713                                                                          ;IS IN R0
6714  027216  006300                  2$:     ASL     R0                      ;SHIFT '1' BIT LEFT
6715  027220  103372                          BCC     1$                      ;LOOP UNTIL 1 BIT SHIFTS OUT
6716  027222  005042                          CLR     -(R2)                   ;CLEAR STACK LIMIT REG
6717
6718                          ;THIS TEST CHECKS THAT A PROPER 'RED' ZONE VIOLATION OCCURS, NOTE THAT
6719                          ;NO 'RED ZONE' VIOLATION WILL OCCUR IF IN USER/SUPER MODES.
6720                          ;A RED ZONE VIOLATION PUSHES THE CURRENT PSW,PC ON A STACK AT 2 AND 0
6721                          ;AND TAKES THE NEXT INSTRUCTION FROM THE PC IN LOCATION4. THE INST-
6722                          ;RUCTION CAUSING THE RED ZONE VIOLATION IS 'ABORTED'.
6723  027224  010746                          MOV     PC,-(SP)                ;GET CURRENT PC
6724  027226  062716  000060                  ADD     #4$-.,(SP)              ;FORM ADDRESS OF 4$ BELOW
6725  027232  012637  000004                  MOV     (SP)+,@#ERRVEC          ;SET ERROR TRAP VECTOR TO 4$ BELOW
6726  027236  013737  177776  000006          MOV     @#PSW,@#ERRVEC+2        ;RETAIN CURRENT STATUS ON TRAP
6727  027244  010712                          MOV     PC,(R2)                 ;SET STACK LIMIT TO CURRENT PC
6728                                                                          ;+400
6729  027246  011206                          MOV     (R2),SP                 ;AND STACK PTR = STACK LIMIT REG
6730  027250  010603                          MOV     SP,R3                   ;SAVE STACK PTR
6731  027252  016304  000336                  MOV     336(R3),R4              ;SAVE MEMORY LOC CONTENTS
6732                                                                          ;AT 'RED ZONE' BOUNDARY
6733  027256  032737  140000  177776          BIT     #UM,@#PSW               ;BRANCH IF IN KERNEL MODE
6734  027264  001403                          BEQ     20$
6735  027266  010466  000336                  MOV     R4,336(SP)              ;SHOULD NOT CAUSE TRAP
6736  027272  000432                          BR      100$
6737
6738  027274  005066  000336          20$:    CLR     336(SP)                 ;SHOULD CAUSE 'RED ZONE' TRAP
6739  027300  012706  000700          3$:     MOV     #SUPSTK,SP              ;RESTORE THE STACK
6740  027304  104000                          HLT                             ;ERROR! FAILED TO TRAP
6741
6742  027306  032737  140000  000002  4$:     BIT     #UM,@#2                 ;CHECK IF TRAPPED WHEN IN USER
6743                                                                          ;/SUPER MODES (2 CONTAINS OLD PSW)
6744  027314  001013                          BNE     99$                     ;GO TO ERROR CALL
6745  027316  010600                          MOV     SP,R0                   ;STACK PTR SHOULD = 0
6746  027320  001011                          BNE     99$                     ;GO TO ERROR CALL IF NOT 0
6747  027322  026304  000336                  CMP     336(R3),R4              ;CHECK THAT INST WAS ABORTED
6748  027326  001006                          BNE     99$                     ;GO REPORT ERRPR
6749  027330  005012                  5$:     CLR     (R2)                    ;CLEAR STACK LIMIT REG
6750  027332  010705                          MOV     PC,R5                   ;GET CURRENT PC
6751  027334  062705  177744                  ADD     #3$-.,R5                ;FORM ADDRESS OF  3$ ABOVE
6752  027340  020516                          CMP     R5,(SP)                 ;CHECK THAT RETURN PC IS ON
6753                                                                          ;THE STACK (AT 0)
```

```
6754  027342  001406                          BEQ     100$                    ;EXIT TEST
6755
6756                                  ;ERROR
6757  027344  005012                  99$:    CLR     (R2)                    ;CLEAR STACK LIMIT REG
6758  027346  010463  000336                  MOV     R4,336(R3)              ;RESTORE MEM LOCATION
6759  027352  012706  000700                  MOV     #SUPSTK,SP              ;SET STACK PTR
6760  027356  104000                          HLT                             ;ERROR!
6761  027360  010463  000336          100$:   MOV     R4,336(R3)              ;RESTORE MEM LOCATION
6762  027364  005022                          CLR     (R2)+                   ;CLEAR STACK LIM REG
6763  027366  012706  000700                  MOV     #SUPSTK,SP              ;SET STACK PTR
6764  027372  042712  000340                  BIC     #340,(R2)               ;SET PRIORITY LEVEL BACK TO 0
6765  027376  012737  053440  000004          MOV     #ERPRT,@#ERRVEC         ;RESTORE ERROR TRAP VECTOR
6766  027404  013737  177776  000006          MOV     @#PSW,@#ERRVEC+2
6767  027412  112737  000340  000006          MOVB    #PR7,@#ERRVEC+2
6768  027420  042737  000020  000006          BIC     #BIT4,@#ERRVEC+2
6769  027426  005037  177766                  CLR     @#CPUERR                        ;CLEAR ERROR REG
6770  027432                          101$:
6771                                  ;;*****************************************************************
6772                                  ;*TEST 66        MEMORY MANAGEMENT REGISTER TESTS
6773                                  ;*      PDR TEST - THIS TEST WRITES 64. RANDOM #'S INTO EACH PDR REGISTER
6774                                  ;*      NOTE:  IF MEM MGMT IS ENABLED ONLY PDR/PAR PAIRS 3-5 ARE TESTED.
6775                                  ;;*****************************************************************
6776  027432  112737  000066  001202  TST66:  MOVB    #66,@#$TSTNM                    ;LOAD TEST NUMBER
6777  027440  000004                          SCOPE
6778
6779  027442  012702  027666          KTPDR:  MOV     #PDRTBL,R2              ;SET TABLE ADDRESS OF PDR'S
6780  027446  012705  100360                  MOV     #100360,R5              ;SET BIT MASK (11/45)
6781  027452  012200                  1$:     MOV     (R2)+,R0                ;GET PDR ADDRESS
6782  027454  001435                          BEQ     100$                    ;EXIT ON '0' TERMINATOR
6783  027456  012716  000010          2$:     MOV     #8.,(SP)                ;SET LOOP COUNT (FOR 8 REGS)
6784  027462  105737  001503                  TSTB    @#MMON                  ;BRANCH IF MEM MGMT DISABLED
6785  027466  001404                          BEQ     3$
6786  027470  062700  000006                  ADD     #6,R0                   ;SET R0 TO PDR3
6787  027474  012716  000003                  MOV     #3,(SP)                 ;AND LIMIT TO TEST 3 PDRS
6788  027500  012703  000040          3$:     MOV     #32.,R3                 ;SET DATA COUNT
6789  027504  005004                          CLR     R4                      ;INITIALIZE DATA TO BE WRITTEN
6790  027506  040504                  4$:     BIC     R5,R4                   ;CLEAR NON-SETTABLE BITS
6791  027510  010410                          MOV     R4,(R0)                 ;WRITE INTO PDR
6792  027512  021004                          CMP     (R0),R4                 ;AND CHECK DATA READ BACK
6793  027514  001013                          BNE     99$                     ;GO TO ERROR CALL
6794  027516  005104                          COM     R4                      ;COMPLEMENT DATA
6795  027520  040504                          BIC     R5,R4                   ;CLEAR NON-SETTABLE BITS
6796  027522  010410                          MOV     R4,(R0)                 ;WRITE COMPLEMENT DATA INTO PDR
6797  027524  021004                          CMP     (R0),R4                 ;AND CHECK
6798  027526  001006                          BNE     99$                     ;GO TO ERROR CALL
6799  027530  060104                          ADD     R1,R4                   ;STEP DATA
6800  027532  077313                          SOB     R3,4$
6801  027534  005020                  5$:     CLR     (R0)+                   ;STEP TO NEXT REGISTER
6802  027536  005316                          DEC     (SP)                    ;DECREMENT REGISTER COUNT
6803  027540  001357                          BNE     3$
6804  027542  000743                          BR      1$                      ;GET NEXT SET OF 8 REGISTERS
6805
6806  027544  104000                  99$:    HLT                             ;ERROR! INCORRECT DATA READ
6807                                                                          ;BACK FROM PDR. ADDRESS OF
6808                                                                          ;PDR IS IN R0, DATA IS IN R4
6809  027546  000772                          BR      5$                      ;STEP TO NEXT REGISTER
```

```
6810  027550                          100$:
6811                                  ;;*****************************************************************
6812                                  ;*TEST 67       PAR TEST
6813                                  ;*      PAR TEST - THIS TEST WRITES 64. COMPLEMENTING RANDOM #'S INTO EACH PAR.
6814                                  ;;*****************************************************************
6815  027550  112737  000067  001202  TST67:  MOVB    #67,@#$TSTNM            ;LOAD TEST NUMBER
6816  027556  000004                  SCOPE
6817  027560  012702  027704          KTPAR:  MOV     #PARTBL,R2      ;GET TABLE ADDRESS OF PAR'S
6818  027564  005005                  CLR     R5
6819  027566  012200                  1$:     MOV     (R2)+,R0        ;GET PAR ADDRESS
6820  027570  001435                  BEQ     100$            ;EXIT ON '0' TERMINATOR
6821  027572  012716  000010          2$:     MOV     #8.,(SP)        ;SET LOOP COUNT (FOR 8 REGS.)
6822  027576  105737  001503          TSTB    @#MMON          ;BRANCH IF MEM MGMT DISABLED
6823  027602  001404                  BEQ     3$
6824  027604  062700  000006          ADD     #6,R0           ;SET R0 TO PAR3
6825  027610  012716  000003          MOV     #3,(SP)         ;AND LIMIT TEST TO 3 PARS
6826  027614  012703  000040          3$:     MOV     #32.,R3         ;SET DATA COUNT
6827  027620  005004                  CLR     R4              ;INITIALIZE DATA
6828  027622  040504                  4$:     BIC     R5,R4           ;CLEAR NON-SETTABLE BITS
6829  027624  010410                  MOV     R4,(R0)         ;WRITE INTO PAR
6830  027626  021004                  CMP     (R0),R4         ;AND CHECK
6831  027630  001013                  BNE     99$             ;TAKE ERROR EXIT
6832  027632  005104                  COM     R4              ;COMPLEMENT DATA
6833  027634  040504                  BIC     R5,R4           ;CLEAR NON-SETTABLE BITS
6834  027636  010410                  MOV     R4,(R0)         ;WRITE COMPLEMENT DATA
6835  027640  021004                  CMP     (R0),R4         ;AND CHECK
6836  027642  001006                  BNE     99$             ;TAKE ERROR EXIT
6837  027644  060104                  ADD     R1,R4           ;STEP DATA
6838  027646  077313                  SOB     R3,4$           ;LOOP UNTIL FINISHED
6839
6840  027650  005020                  5$:     CLR     (R0)+
6841  027652  005316                  DEC     (SP)            ;DECREMENT REGISTER COUNT
6842  027654  001357                  BNE     3$              ;BRANCH IF 8 REGS NOT DONE
6843  027656  000743                  BR      1$
6844
6845  027660  104000                  99$:    HLT                     ;ERROR! INCORRECT DATA READ BACK
6846                                                                  ;FROM PAR. ADDRESS OF PAR IS IN
6847                                                                  ;R0, DATA IS IN R4
6848  027662  000772                  BR      5$              ;DO NEXT REGISTER
6849  027664                          100$:
6850  027664  000416                  BR      TST70           ;;GO TO NEXT TEST
6851                                  ;TABLES FOR PDR & PAR TESTS ABOVE
6852  027666  172300                  PDRTBL: .WORD   KIPDR0
6853  027670  177600                          .WORD   UIPDR0
6854  027672  172200                          .WORD   SIPDR0          ;CHANGED TO '0' IF 11/40
6855  027674  172320                          .WORD   KDPDR0
6856  027676  177620                          .WORD   UDPDR0
6857  027700  172220                          .WORD   SDPDR0
6858  027702  000000                          .WORD   0               ;TERMINATOR
6859
6860  027704  172340                  PARTBL: .WORD   KIPAR0
6861  027706  177640                          .WORD   UIPAR0
6862  027710  172240                          .WORD   SIPAR0          ;CHANGED TO '0' IF 11/40
6863  027712  172360                          .WORD   KDPAR0
6864  027714  177660                          .WORD   UDPAR0
6865  027716  172260                          .WORD   SDPAR0
```

```
6866  027720  000000                          .WORD   0               ;TERMINATOR
6867
6868                                  ;;*****************************************************************
6869                                  ;*TEST 70       CHECK KT ABORT LOGIC
6870                                  ;*      THIS TEST CHECKS KT ABORT LOGIC. TEST CREATES AN ABORT CONDITION
6871                                  ;*      AND INSURES THAT ABORT IS TAKEN PROPERLY. NOTE: TEST IS EXECUTED ONLY
6872                                  ;*      IF TEST IS ENTERED WITH MEM MGMT ENABLED.
6873                                  ;*****************************************************************
6874  027722  112737  000070  001202  TST70:  MOVB    #70,@#$TSTNM            ;LOAD TEST NUMBER
6875  027730  000004                  SCOPE
6876  027732  105737  001503          KTABT:  TSTB    @#MMON          ;BRANCH IF MEM MGMT DISABLED
6877  027736  001515                  BEQ     KTEX
6878  027740  005037  172350                  CLR     @#KIPAR4        ;SET UP MEM MGMT REGISTERS
6879  027744  005037  172310                  CLR     @#KIPDR4        ;TO ABORT IF A MEMORY
6880  027750  005037  177650                  CLR     @#UIPAR4        ;REFERENCE IS MADE TO
6881  027754  005037  177610                  CLR     @#UIPDR4        ;ADDRESSES (VIRTUAL) BETWEEN
6882  027760  005037  172250                  CLR     @#SIPAR4
6883  027764  005037  172210                  CLR     @#SIPDR4
6884  027770  013746  000250          1$:     MOV     @#MMVEC,-(SP)   ;SAVE MEM MGMT VECTOR
6885  027774  013746  000252                  MOV     @#MMVEC+2,-(SP) ;AND PRIORITY
6886  030000  010746                  MOV     PC,-(SP)        ;SET MEM MGMT
6887  030002  062716  000040                  ADD     #4$-.,(SP)      ;VECTOR TO 4$ BELOW
6888  030006  012637  000250                  MOV     (SP)+,@#MMVEC
6889  030012  013737  177776  000252          MOV     @#PSW,@#MMVEC+2
6890  030020  005000                  CLR     R0              ;CLEAR ABORT INDICATOR
6891  030022  010702                  MOV     PC,R2           ;SET R2 AND R3 NOTE:
6892  030024  012703  100000                  MOV     #100000,R3      ;THE REF VIA R3 CAUSES THE
6893  030030  014223                  2$:     MOV     -(R2),(R3)+     ;ABORT
6894  030032  005700                  3$:     TST     R0              ;BRANCH IF THE ABORT OCCURRED
6895  030034  001001                  BNE     .+4
6896  030036  104000                  HLT                     ;REPORT ERROR
6897  030040  000445                  BR      100$
6898                                  ;ABORT HERE
6899  030042  013700  177776          4$:     MOV     @#PSW,R0        ;SR0 SHOULD CONTAIN
6900  030046  000300                  SWAB    R0              ;CAUSE FOR ABORT AND
6901  030050  006200                  ASR     R0              ;ALSO WHICH SEGMENT
6902  030052  042700  177637                  BIC     #177637,R0      ;WAS IN USE WHEN ABORT
6903  030056  062700  100011                  ADD     #100011,R0      ;OCCURRED.
6904  030062  020037  177572                  CMP     R0,@#SR0
6905  030066  001025                  BNE     99$
6906  030070  012700  030030                  MOV     #2$,R0          ;GET ADDRESS OF INST THAT ABORTED
6907  030074  020037  177576                  CMP     R0,@#SR2        ;THAT ABORTED
6908  030100  001020                  BNE     99$
6909  030102  012700  000362                  MOV     #362,R0
6910  030106  120037  177574                  CMPB    R0,@#SR1        ;SR1 (11/45) CONTAINS REGISTER
6911  030112  001013                  BNE     99$             ;MODIFICATIONS MADE
6912  030114  012700  000023                  MOV     #23,R0
6913  030120  120037  177575                  CMPB    R0,@#SR1+1
6914  030124  001006                  BNE     99$
6915  030126  012700  030030                  MOV     #2$,R0
6916  030132  005720                  5$:     TST     (R0)+           ;R0=ADDRESS OF INST FOLLOWING ABORT
6917  030134  020016                  CMP     R0,(SP)         ;(3$)
6918  030136  001001                  BNE     99$
6919  030140  000002                  RTI                     ;RETURN
6920                                  ;ENTER HERE ON ERROR
6921  030142  104000                  99$:    HLT                     ;REPORT ERROR
```

```
6922  030144  010716                          MOV     PC,(SP)
6923  030146  062716  177664                  ADD     #3$-.,(SP)
6924  030152  000002                          RTI                     ;RETURN
6925  030154  012637  000252          100$:   MOV     (SP)+,@#MMVEC+2 ;RESTORE ABORT VECTOR
6926  030160  012637  000250                  MOV     (SP)+,@#MMVEC   ;& PRIORITY.
6927  030164  012737  000001  177572          MOV     #1,@#SR0        ;CLEAR ERROR CONDITIONS
6928  030172                          KTEX:
6929                                  ;;******************************************************************
6930                                  ;*TEST 71       MAPPING REGISTER TESTS
6931                                  ;*      THIS TEST LOADS RANDOM #'S INTO EACH MAPPING REGISTER
6932                                  ;;******************************************************************
6933  030172  112737  000071  001202  TST71:  MOVB    #71,@#$TSTNM            ;LOAD TEST NUMBER
6934  030200  000004                  SCOPE
6935  030202  032737  000040  172516          BIT     #BIT5,@#MMR3    ;IS MAP ON?
6936  030210  001053                          BNE     MAPTWO          ;BRANCH IF YES
6937  030212  012700  170200          MAPTST: MOV     #MAPLO,R0       ;SET ADRS OF FIRST MAP REGISTER
6938  030216  012706  000700                  MOV     #SUPSTK,SP      ;SETUP THE SP
6939  030222  012716  000001                  MOV     #1,(SP)         ;SET BIT MASK FOR MAPLO <15-01>
6940  030226  012702  177700                  MOV     #177700,R2      ;AND ALSO FOR MAPHO <21-16>
6941  030232  012703  000040          1$:     MOV     #32.,R3         ;SET DATA COUNT
6942  030236  005005                          CLR     R5              ;SET INITIAL DATA
6943  030240  010504                  2$:     MOV     R5,R4           ;GET DATA
6944  030242  041604                          BIC     (SP),R4         ;CLEAR UNUSED BITS
6945  030244  010410                          MOV     R4,(R0)         ;LOAD DATA INTO MAPLO <15-01>
6946  030246  021004                          CMP     (R0),R4         ;CHECK DATA
6947  030250  001032                          BNE     99$             ;BRANCH IF INCORRECT
6948  030252  005105                          COM     R5              ;COMPLEMENT TEST DATA
6949  030254  010504                          MOV     R5,R4           ;GET TEST DATA
6950  030256  041604                          BIC     (SP),R4         ;CLEAR UNUSED BITS
6951  030260  010410                          MOV     R4,(R0)         ;LOAD COMPLEMENT DATA
6952  030262  021004                          CMP     (R0),R4         ;AND CHECK
6953  030264  001024                          BNE     99$
6954  030266  005720                          TST     (R0)+           ;STEP TO NEXT REGISTER
6955  030270  010504                          MOV     R5,R4           ;GET COMPLEMENT TEST DATA
6956  030272  040204                          BIC     R2,R4           ;CLEAR UNUSED BITS
6957  030274  010410                          MOV     R4,(R0)         ;LOAD TEST DATA INTO MAPHO <21-16>
6958  030276  021004                          CMP     (R0),R4         ;AND CHECK
6959  030300  001016                          BNE     99$
6960  030302  005105                          COM     R5              ;COMPLEMENT TEST DATA
6961  030304  010504                          MOV     R5,R4           ;GET TEST DATA
6962  030306  040204                          BIC     R2,R4           ;CLEAR UNUSED BITS
6963  030310  010410                          MOV     R4,(R0)         ;LOAD TEST DATA
6964  030312  021004                          CMP     (R0),R4         ;AND CHECK
6965  030314  001010                          BNE     99$
6966  030316  060705                          ADD     PC,R5           ;FORM NEXT TEST DATA
6967  030320  005740                          TST     -(R0)           ;RESET PTR TO REGISTER <15-01>
6968  030322  077332                          SOB     R3,2$           ;AND TEST UNTIL ALL #'S USED
6969  030324  022020                          CMP     (R0)+,(R0)+     ;STEP TO NEXT REGISTER PAIR
6970  030326  022700  170400                  CMP     #MAPLO+128.,R0  ;BRANCH IF NOT LAST PAIR
6971  030332  001337                          BNE     1$
6972  030334  000401                          BR      MAPTWO
6973
6974  030336  104000          99$:    HLT                             ;ERROR! INCORRECT DATA READ BACK
6975                                                                  ;FROM MAP REG. ADRS OF REGISTER 15
6976                                                                  ;IN R0, GOOD DATA IS IN R4
6977  030340  005737  001504  MAPTWO: TST     @#QV                    ;QV OR AUTO-ACCEPT?
```

```
6978  030344  001411                          BEQ     RELE7           ;BRANCH IF NO
6979  030346  012737  020000  170204          MOV     #20000,@#MAPL1  ;SET MAP 1 INCASE ACT11
6980  030354  005037  170206                  CLR     @#MAPH1
6981  030360  005037  170200                  CLR     @#MAPL0
6982  030364  005037  170202                  CLR     @#MAPH0
6983  030370  000004                  RELE7:  SCOPE
6984  030372  010702                          MOV     PC,R2
6985  030374  062702  000012                  ADD     #12,R2
6986  030400  012707  034240                  MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
6987  030404  000000                  REL77:  .WORD   0
6988                                  ;7777777777777 LAST ADDRESS OF CODE TO BE RELOCATED 77777777777
6989
6990                                  ;;*****************************************************************
6991                                  ;*TEST 72        FLOATING POINT TEST 1
6992                                  ;*
6993                                  ;*      THIS TEST TAKES TWO RANDOM NUMBERS (A AND B) AND
6994                                  ;*      COMPARES THE RESULTS OF TWO EQUAL CALCULATIONS.
6995                                  ;*      EACH SECTION EVALUATES A DIFFERENT EQUATION AS DESCRIBED BELOW:
6996                                  ;*      SECT1   (A+B)**2=A**2+2*A*B+B**2
6997                                  ;*      SECT2   (A+B)*(A-B)=A**2-B**2
6998                                  ;*      SECT3   A/B*B=A
6999                                  ;;*****************************************************************
7000  030406  112737  000072  001202  TST72:  MOVB    #72,@#$TSTNM            ;LOAD TEST NUMBER
7001  030414  000004                          SCOPE
7002  030416  012737  000001  001316          MOV     #1,@#$TIMES     ;SET ITTERATIONS TO 1
7003
7004                                          .SBTTL  START OF SECTION 8
7005                                  ;8888888888888 FIRST ADDRESS TO BE RELOCATED 888888888
7006  030424  010700                  REL8:   MOV     PC,R0           ;GET PC
7007  030426  005740                          TST     -(R0)           ;R0 CONTAINS THE ADDRESS OF REL8
7008  030430  010037  001512                  MOV     R0,@#FRSTAD     ;SAVE
7009  030434  010700                          MOV     PC,R0           ;GET CURRENT PC
7010  030436  162700  030436                  SUB     #.,R0           ;SUBTRACT RELOCATION FACTOR
7011  030442  010037  001506                  MOV     R0,@#FACTOR     ;SAVE RELOCATION FACTOR
7012  030446  010737  001212                  MOV     PC,@#$LPERR     ;SET LOOP ADDRESS
7013  030452  062737  000030  001212          ADD     #30,@#$LPERR    ;ADJUST
7014  030460  013737  001212  001210          MOV     @#$LPERR,@#$LPADR
7015  030466  105737  001502                  TSTB    @#NEXEC         ;BR IF TEST CODE TO BE EXECUTED
7016  030472  001402                          BEQ     .+6
7017  030474  000167  002414                  JMP     RELE8
7018  030500  032737  020000  001470          BIT     #FPOPT,@#OPT.CP ;FLOATING POINT AVAILABLE?
7019  030506  001002                          BNE     100$            ;BRANCH IF YES
7020  030510  000167  002416                  JMP     REL88+2
7021  030514  004737  050706          100$:   JSR     PC,@#FLTSGL     ;GET RANDOM OPERANDS
7022  030520  170127  000000                  LDFPS   #0              ;INIT FPS
7023  030524  172537  001276                  LDF     @#$TMP0,AC1     ;LOAD A OPERAND
7024  030530  172437  001302                  LDF     @#$TMP2,AC0     ;LOAD B OPERAND
7025  030534  013737  001256  001404          MOV     @#$REG0,@#$AC1  ;SETUP EXTENDED
7026  030542  013737  001260  001402          MOV     @#$REG1,@#$AC0  ;EXPONENTS
7027  030550  004767  002202                  JSR     PC,FLTADD       ;PERFORM THE ADD
7028  030554  174100                          STF     AC1,AC0         ;SETUP AC0 TO
7029  030556  013737  001404  001402          MOV     @#$AC1,@#$AC0   ;PERFORM THE SQUARE
7030  030564  004767  002116                  JSR     PC,FLTMPY       ;DO THE MULTIPLY
7031  030570  174137  001306                  STF     AC1,@#$TMP4     ;SAVE RESULT
7032  030574  013737  001404  001262          MOV     @#$AC1,@#$REG2  ;AND SOFTWARE EXP
7033                                  ;
```

```
7034                                        ;NOW DO THE RIGHT HAND SIDE OF THE EQUATION
7035                                        ;DO THE A*A FIRST
7036  030602  013737  001256  001402            MOV     @#$REG0,@#$AC0  ;GET EXT EXPONENT
7037  030610  172437  001276                    LDF     @#$TMP0,AC0     ;LOAD OPERAND A
7038  030614  013737  001402  001404            MOV     @#$AC0,@#$AC1   ;SET OPERAND B EXT EXPONENT
7039  030622  172500                            LDF     AC0,AC1         ;LOAD B OPERAND
7040  030624  004767  002056                    JSR     PC,FLTMPY       ;EXECUTE THE MULTIPLY
7041  030630  174102                            STF     AC1,AC2         ;SAVE RESULT
7042  030632  013737  001404  001406            MOV     @#$AC1,@#$AC2
7043                                        ;
7044                                        ;NOW DO THE B*B
7045  030640  172437  001302                    LDF     @#$TMP2,AC0     ;LOAD B OPERAND
7046  030644  172500                            LDF     AC0,AC1
7047  030646  013737  001260  001402            MOV     @#$REG1,@#$AC0  ;AND EXT EXPONENT
7048  030654  013737  001402  001404            MOV     @#$AC0,@#$AC1
7049  030662  004767  002020                    JSR     PC,FLTMPY       ;DO THE MULTIPLY
7050  030666  174103                            STF     AC1,AC3         ;SAVE THE RESULT
7051  030670  013737  001404  001410            MOV     @#$AC1,@#$AC3
7052                                        ;
7053                                        ;NOW DO THE 2*B*A
7054  030676  012701  001302                    MOV     #$TMP2,R1
7055  030702  172411                            LDF     (R1),AC0        ;LOAD THE B OPERAND
7056  030704  172541                            LDF     -(R1),AC1       ;LOAD THE A OPERAND
7057  030706  013737  001260  001402            MOV     @#$REG1,@#$AC0  ;AND THE EXT EXPONENTS
7058  030714  013737  001256  001404            MOV     @#$REG0,@#$AC1
7059  030722  004767  001760                    JSR     PC,FLTMPY       ;DO THE MULTIPLY
7060  030726  172427  040000                    LDF     #^O40000,AC0    ;SETUP TO MULTIPLY BY TWO
7061  030732  012737  000002  001402            MOV     #2,@#$AC0
7062  030740  004767  001742                    JSR     PC,FLTMPY       ;DO THE MULTIPLY
7063                                        ;
7064                                        ;NOW SUM THE RESULTS
7065  030744  013737  001410  001402            MOV     @#$AC3,@#$AC0
7066  030752  172403                            LDF     AC3,AC0         ;GET RESULT OF B*B
7067  030754  004767  001776                    JSR     PC,FLTADD       ;ADD THE RESULT
7068  030760  172402                            LDF     AC2,AC0         ;GET RESULT OF A*A
7069  030762  013737  001406  001402            MOV     @#$AC2,@#$AC0
7070  030770  004767  001762                    JSR     PC,FLTADD       ;ADD THIS RESULT
7071  030774  174137  001312                    STF     AC1,@#$TMP6     ;SAVE FINAL RESULT
7072  031000  013737  001404  001264            MOV     @#$AC1,@#$REG3
7073                                        ;
7074                                        ;NOW CHECK BOTH SIDES OF THE EQUATION
7075                                        ;CALCULATE THE NUMBER OF CORRECT BITS
7076                                        ;PUT LARGEST EXPONENT OF A**2 OR B**2 IN $AC2
7077  031006  023737  001406  001410            CMP     @#$AC2,@#$AC3
7078  031014  002003                            BGE     1$              ;BRANCH IF $AC2 ALREADY HAS LARGEST
7079  031016  013737  001410  001406            MOV     @#$AC3,@#$AC2   ;$AC3 WAS LARGER
7080  031024  163737  001404  001406  1$:       SUB     @#$AC1,@#$AC2   ;NOW CALCULATE NUMBER
7081  031032  162737  000024  001406            SUB     #20.,@#$AC2     ;OF CORRECT BITS WITHIN 2
7082  031040  005437  001406                    NEG     @#$AC2          ;MAKE RESULT POSITIVE
7083  031044  172437  001306                    LDF     @#$TMP4,AC0     ;LOAD RESULT OF LEFT HAND SIDE
7084  031050  013737  001262  001402            MOV     @#$REG2,@#$AC0  ;AND EXTENDED EXPONENT
7085  031056  004767  001670                    JSR     PC,FLTSUB       ;SUBTRACT TO SEE HOW CLOSE THEY ARE
7086  031062  163737  001264  001404            SUB     @#$REG3,@#$AC1  ;GET DIFFERENCE IN EXT EXPONENTS
7087                                                                    ;ACTUAL EXP'S ARE EQUAL TO 200
7088  031070  100002                            BPL     3$              ;ENSURE RESULT IS POSITIVE
7089  031072  005437  001404                    NEG     @#$AC1
```

```
      031076  023737  001406  001404  3$:     CMP     @#$AC2,@#$AC1     ;ANSWERS WITHIN ALLOWABLE NUMBER?
      031104  003401                          BLE     SECT2             ;BRANCH IF YES
      031106  104014                  4$:     ERROR   14                ;RESULTS ARE WRONG
                                      ;;****************************************************************
      031110  170127  000000          SECT2:  LDFPS   #0
                                      ;DO A+B
      031114  172537  001276                  LDF     @#STMP0,AC1       ;LOAD A OPERAND
      031120  172437  001302                  LDF     @#STMP2,AC0       ;LOAD B OPERAND
      031124  013737  001256  001404          MOV     @#SREG0,@#$AC1
      031132  013737  001260  001402          MOV     @#SREG1,@#$AC0
      031140  004767  001612                  JSR     PC,FLTADD         ;ADD THEM
      031144  174102                          STF     AC1,AC2           ;SAVE IN AC2
      031146  013737  001404  001406          MOV     @#$AC1,@#$AC2     ;AND EXT EXPONENT
                                      ;NOW DO THE A-B
      031154  172537  001276                  LDF     @#STMP0,AC1       ;LOAD OPERAND A
      031160  013737  001256  001404          MOV     @#SREG0,@#$AC1    ;AND EXT EXPONENT
      031166  172437  001302                  LDF     @#STMP2,AC0       ;LOAD OPERAND B
      031172  013737  001260  001402          MOV     @#SREG1,@#$AC0
      031200  004767  001546                  JSR     PC,FLTSUB         ;SUBTRACT THEM
                                      ;NOW DO (A+B)*(A-B)
      031204  172402                          LDF     AC2,AC0           ;GET RESULT OF (A+B)
      031206  013737  001406  001402          MOV     @#$AC2,@#$AC0
      031214  004767  001466                  JSR     PC,FLTMPY         ;FORM THE PRODUCT
      031220  174137  001306                  STF     AC1,@#STMP4       ;SAVE RESULT
      031224  013737  001404  001262          MOV     @#$AC1,@#SREG2    ;AND EXT EXPONENT
                                      ;NOW DO THE B*B
      031232  172437  001302                  LDF     @#STMP2,AC0       ;LOAD OPERAND B
      031236  013737  001260  001402          MOV     @#SREG1,@#$AC0
      031244  172500                          LDF     AC0,AC1           ;B OPERAND IS IN AC0
      031246  013737  001402  001404          MOV     @#$AC0,@#$AC1     ;AND EXT EXPONENT
      031254  004767  001426                  JSR     PC,FLTMPY         ;
      031260  174102                          STF     AC1,AC2           ;SAVE RESULT IN AC2
      031262  013737  001404  001406          MOV     @#$AC1,@#$AC2
                                      ;NOW DO THE A*A
      031270  172437  001276                  LDF     @#STMP0,AC0       ;LOAD OPERAND A
      031274  013737  001256  001402          MOV     @#SREG0,@#$AC0
      031302  172500                          LDF     AC0,AC1
      031304  013737  001402  001404          MOV     @#$AC0,@#$AC1
      031312  004767  001370                  JSR     PC,FLTMPY         ;EXECUTE THE MULTIPLY
      031316  013737  001404  001410          MOV     @#$AC1,@#$AC3     ;SAVE EXT EXPO OF A*A
                                      ;NOW DO A**2-B**2
      031324  172402                          LDF     AC2,AC0           ;GET B*B
      031326  013737  001406  001402          MOV     @#$AC2,@#$AC0     ;A*A IN AC1
      031334  004767  001412                  JSR     PC,FLTSUB
      031340  174137  001312                  STF     AC1,@#STMP6       ;SAVE IN MEMORY
      031344  013737  001404  001264          MOV     @#$AC1,@#SREG3
                                      ;NOW COMPUTE THE RESULTS
                                      ;CALCULATE THE NUMBER OF CORRECT BITS
      031352  023737  001406  001410          CMP     @#$AC2,@#$AC3     ;DETERMINE WHICH EXP IS LARGER
      031360  002003                          BGE     2$                ;BRANCH IF AC2 LARGER
      031362  013737  001410  001406          MOV     @#$AC3,@#$AC2     ;PUT LARGEST IN AC2
      031370  163737  001404  001406  2$:     SUB     @#$AC1,@#$AC2
      031376  162737  000026  001406          SUB     #22.,@#$AC2
      031404  005437  001406                  NEG     @#$AC2
      031410  172437  001306                  LDF     @#STMP4,AC0       ;GET LEFT HAND SIDE
      031414  013737  001262  001402          MOV     @#SREG2,@#$AC0
```

```
7146  031422  004767  001324                    JSR     PC,FLTSUB       ;SUBTRACT TO SEE HOW CLOSE THEY ARE
7147  031426  163737  001264  001404            SUB     @#$REG3,@#$AC1  ;SUB EXT EXPONENTS
7148                                                                      ;ACTUAL EXPONENTS ARE EQUAL
7149  031434  100002                            BPL     1$              ;MAKE SURE RESULT IS POSITIVE
7150  031436  005437  001404                    NEG     @#$AC1
7151  031442  023737  001406  001404  1$:       CMP     @#$AC2,@#$AC1   ;RESULTS WITHIN RANGE ALLOWED?
7152  031450  003401                            BLE     SECT3           ;BRANCH IF YES
7153  031452  104014                            ERROR   14              ;RESULTS WRONG
7154
7155                                            ;;*****************************************************************
7156  031454  172537  001276                SECT3:  LDF     @#$TMP0,AC1     ;LOAD OPERAND A
7157  031460  172437  001302                        LDF     @#$TMP2,AC0     ;AND OPERAND B
7158  031464  013737  001256  001404                MOV     @#$REG0,@#$AC1
7159  031472  013737  001260  001402                MOV     @#$REG1,@#$AC0
7160  031500  004767  001224                        JSR     PC,FLTDIV       ;GO DIVIDE THEM
7161  031504  004767  001176                        JSR     PC,FLTMPY       ;MULTIPLY RESULT BY B
7162  031510  174137  001306                        STF     AC1,@#$TMP4     ;SAVE RESULT
7163  031514  013737  001404  001262                MOV     @#$AC1,@#$REG2
7164  031522  172437  001276                        LDF     @#$TMP0,AC0     ;LOAD OPERAND A
7165  031526  174037  001312                        STF     AC0,@#$TMP6     ;SAVE INCASE TYPE OUT
7166  031532  013737  001256  001402                MOV     @#$REG0,@#$AC0
7167  031540  013737  001256  001264                MOV     @#$REG0,@#$REG3
7168  031546  004767  001200                        JSR     PC,FLTSUB       ;SUBTRACT RIGHT AND LEFT HAND SIDES
7169  031552  163737  001256  001404                SUB     @#$REG0,@#$AC1  ;SEE IF RESULT OK
7170  031560  100002                            BPL     1$              ;ENSURE DIFFERANCE IS POSITIVE
7171  031562  005437  001404                    NEG     @#$AC1
7172  031566  022737  000027  001404  1$:       CMP     #23.,@#$AC1     ;RESULTS WITHIN 2 BITS?
7173  031574  003001                            BGT     2$              ;BRANCH IF NO
7174  031576  000401                            BR      TST73           ;;GO TO NEXT TEST
7175  031600  104014                        2$:     ERROR   14              ;RESULTS WRONG
7176
7177                                            ;;*****************************************************************
7178                                            ;*TEST 73       FLOATING POINT TEST 2
7179                                            ;*
7180                                            ;*      THIS TEST TAKES TWO RANDOM NUMBERS (A AND B) AND
7181                                            ;*      COMPARES THE RESULTS OF TWO EQUAL CALCULATIONS.
7182                                            ;*      EACH SECTION EVALUATES A DIFFERENT EQUATION AS DESCRIBED BELOW:
7183                                            ;*      SECT1   (A+B)**2=A**2+2*A*B+B**2
7184                                            ;*      SECT2   (A+B)*(A-B)=A**2-B**2
7185                                            ;*      SECT3   A/B*B=A
7186                                            ;*****************************************************************
7187  031602  112737  000073  001202  TST73:   MOVB    #73,@#$TSTNM            ;LOAD TEST NUMBER
7188  031610  000004                            SCOPE
7189  031612  012737  000001  001316            MOV     #1,@#$TIMES
7190  031620  004737  050700                100$:   JSR     PC,@#FLTDBL     ;GET RANDOM OPERANDS
7191  031624  170127  000200                        LDFPS   #200            ;INIT FPS
7192  031630  172537  001276                        LDF     @#$TMP0,AC1     ;LOAD A OPERAND
7193  031634  172437  001306                        LDF     @#$TMP4,AC0     ;LOAD B OPERAND
7194  031640  013737  001256  001404                MOV     @#$REG0,@#$AC1  ;SETUP EXTENDED
7195  031646  013737  001260  001402                MOV     @#$REG1,@#$AC0  ;EXPONENTS
7196  031654  004767  001076                        JSR     PC,FLTADD       ;PERFORM THE ADD
7197  031660  174100                            STF     AC1,AC0         ;SETUP AC0 TO
7198  031662  013737  001404  001402                MOV     @#$AC1,@#$AC0   ;PERFORM THE SQUARE
7199  031670  004767  001012                        JSR     PC,FLTMPY       ;DO THE MULTIPLY
7200  031674  174137  001422                        STF     AC1,@#FLTMP0    ;SAVE RESULT
7201  031700  013737  001404  001262                MOV     @#$AC1,@#$REG2  ;AND SOFTWARE EXP
```

```
7202                                        ;
7203                                        ;NOW DO THE RIGHT HAND SIDE OF THE EQUATION
7204                                        ;DO THE A*A FIRST
7205  031706  013737  001256  001402        MOV     @#$REG0,@#$AC0  ;GET EXT EXPONENT
7206  031714  172437  001276                LDF     @#$TMP0,AC0     ;LOAD OPERAND A
7207  031720  013737  001402  001404        MOV     @#$AC0,@#$AC1   ;SET OPERAND B EXT EXPONENT
7208  031726  172500                        LDF     AC0,AC1         ;LOAD B OPERAND
7209  031730  004767  000752                JSR     PC,FLTMPY       ;EXECUTE THE MULTIPLY
7210  031734  174102                        STF     AC1,AC2         ;SAVE RESULT
7211  031736  013737  001404  001406        MOV     @#$AC1,@#$AC2
7212                                        ;
7213                                        ;NOW DO THE B*B
7214  031744  172437  001306                LDF     @#$TMP4,AC0     ;LOAD B OPERAND
7215  031750  172500                        LDF     AC0,AC1
7216  031752  013737  001260  001402        MOV     @#$REG1,@#$AC0  ;AND EXT EXPONENT
7217  031760  013737  001402  001404        MOV     @#$AC0,@#$AC1
7218  031766  004767  000714                JSR     PC,FLTMPY       ;DO THE MULTIPLY
7219  031772  174103                        STF     AC1,AC3         ;SAVE THE RESULT
7220  031774  013737  001404  001410        MOV     @#$AC1,@#$AC3
7221                                        ;
7222                                        ;NOW DO THE 2*B*A
7223  032002  012701  001306                MOV     #$TMP4,R1
7224  032006  172411                        LDF     (R1),AC0        ;LOAD THE B OPERAND
7225  032010  172541                        LDF     -(R1),AC1       ;LOAD THE A OPERAND
7226  032012  013737  001260  001402        MOV     @#$REG1,@#$AC0  ;AND THE EXT EXPONENTS
7227  032020  013737  001256  001404        MOV     @#$REG0,@#$AC1
7228  032026  004767  000654                JSR     PC,FLTMPY       ;DO THE MULTIPLY
7229  032032  172427  040000                LDF     #^O040000,AC0   ;SETUP TO MULTIPLY BY TWO
7230  032036  012737  000002  001402        MOV     #2,@#$AC0
7231  032044  004767  000636                JSR     PC,FLTMPY       ;DO THE MULTIPLY
7232                                        ;
7233                                        ;NOW SUM THE RESULTS
7234  032050  013737  001410  001402        MOV     @#$AC3,@#$AC0
7235  032056  172403                        LDF     AC3,AC0         ;GET RESULT OF B*B
7236  032060  004767  000672                JSR     PC,FLTADD       ;ADD THE RESULT
7237  032064  172402                        LDF     AC2,AC0         ;GET RESULT OF A*A
7238  032066  013737  001406  001402        MOV     @#$AC2,@#$AC0
7239  032074  004767  000656                JSR     PC,FLTADD       ;ADD THIS RESULT
7240  032100  174137  001432                STF     AC1,@#FLTMP1    ;SAVE FINAL RESULT
7241  032104  013737  001404  001264        MOV     @#$AC1,@#$REG3
7242                                        ;
7243                                        ;NOW CHECK BOTH SIDES OF THE EQUATION
7244                                        ;CALCULATE THE NUMBER OF CORRECT BITS
7245                                        ;PUT LARGEST EXPONENT OF A**2 OR B**2 IN $AC2
7246  032112  023737  001406  001410        CMP     @#$AC2,@#$AC3
7247  032120  002003                        BGE     1$              ;BRANCH IF $AC2 ALREADY HAS LARGEST
7248  032122  013737  001410  001406        MOV     @#$AC3,@#$AC2   ;$AC3 WAS LARGER
7249  032130  163737  001404  001406  1$:   SUB     @#$AC1,@#$AC2   ;NOW CALCULATE NUMBER
7250  032136  162737  000065  001406        SUB     #53.,@#$AC2     ;OF CORRECT BITS WITHIN 2
7251  032144  005437  001406                NEG     @#$AC2          ;MAKE RESULT POSITIVE
7252  032150  172437  001422                LDF     @#FLTMP0,AC0    ;LOAD RESULT OF LEFT HAND SIDE
7253  032154  013737  001262  001402        MOV     @#$REG2,@#$AC0  ;AND EXTENDED EXPONENT
7254  032162  004767  000564                JSR     PC,FLTSUB       ;SUBTRACT TO SEE HOW CLOSE THEY ARE
7255  032166  163737  001264  001404        SUB     @#$REG3,@#$AC1  ;GET DIFFERENCE IN EXT EXPONENTS
7256                                                                ;ACTUAL EXP'S ARE EQUAL TO 200
7257  032174  100002                        BPL     3$              ;ENSURE RESULT IS POSITIVE
```

```
7258  032176  005437  001404                    NEG     @#$AC1
7259  032202  023737  001406  001404  3$:       CMP     @#$AC2,@#$AC1   ;ANSWERS WITHIN ALLOWABLE NUMBER?
7260  032210  003401                            BLE     SECT2D          ;BRANCH IF YES
7261  032212  104016                    4$:     ERROR   16              ;RESULTS ARE WRONG
7262                                    ;;**************************************************************
7263  032214  170127  000200            SECT2D: LDFPS   #200
7264                                    ;DO A+B
7265  032220  172537  001276                    LDF     @#$TMP0,AC1     ;LOAD A OPERAND
7266  032224  172437  001306                    LDF     @#$TMP4,AC0     ;LOAD B OPERAND
7267  032230  013737  001256  001404            MOV     @#$REG0,@#$AC1
7268  032236  013737  001260  001402            MOV     @#$REG1,@#$AC0
7269  032244  004767  000506                    JSR     PC,FLTADD       ;ADD THEM
7270  032250  174102                            STF     AC1,AC2         ;SAVE IN AC2
7271  032252  013737  001404  001406            MOV     @#$AC1,@#$AC2   ;AND EXT EXPONENT
7272                                    ;NOW DO THE A-B
7273  032260  172537  001276                    LDF     @#$TMP0,AC1     ;LOAD OPERAND A
7274  032264  013737  001256  001404            MOV     @#$REG0,@#$AC1  ;AND EXT EXPONENT
7275  032272  172437  001306                    LDF     @#$TMP4,AC0     ;LOAD OPERAND B
7276  032276  013737  001260  001402            MOV     @#$REG1,@#$AC0
7277  032304  004767  000442                    JSR     PC,FLTSUB       ;SUBTRACT THEM
7278                                    ;NOW DO (A+B)*(A-B)
7279  032310  172402                            LDF     AC2,AC0         ;GET RESULT OF (A+B)
7280  032312  013737  001406  001402            MOV     @#$AC2,@#$AC0
7281  032320  004767  000362                    JSR     PC,FLTMPY       ;FORM THE PRODUCT
7282  032324  174137  001422                    STF     AC1,@#FLTMP0    ;SAVE RESULT
7283  032330  013737  001404  001262            MOV     @#$AC1,@#$REG2  ;AND EXT EXPONENT
7284                                    ;NOW DO THE B*B
7285  032336  172437  001306                    LDF     @#$TMP4,AC0     ;LOAD OPERAND B
7286  032342  013737  001260  001402            MOV     @#$REG1,@#$AC0
7287  032350  172500                            LDF     AC0,AC1         ;B OPERAND IS IN AC0
7288  032352  013737  001402  001404            MOV     @#$AC0,@#$AC1   ;AND EXT EXPONENT
7289  032360  004767  000322                    JSR     PC,FLTMPY       ;
7290  032364  174102                            STF     AC1,AC2         ;SAVE RESULT IN AC2
7291  032366  013737  001404  001406            MOV     @#$AC1,@#$AC2
7292                                    ;NOW DO THE A*A
7293  032374  172437  001276                    LDF     @#$TMP0,AC0     ;LOAD OPERAND A
7294  032400  013737  001256  001402            MOV     @#$REG0,@#$AC0
7295  032406  172500                            LDF     AC0,AC1
7296  032410  013737  001402  001404            MOV     @#$AC0,@#$AC1
7297  032416  004767  000264                    JSR     PC,FLTMPY       ;EXECUTE THE MULTIPLY
7298  032422  013737  001404  001410            MOV     @#$AC1,@#$AC3   ;SAVE EXT EXPO OF A*A
7299                                    ;NOW DO A**2-B**2
7300  032430  172402                            LDF     AC2,AC0         ;GET B*B
7301  032432  013737  001406  001402            MOV     @#$AC2,@#$AC0   ;A*A IN AC1
7302  032440  004767  000306                    JSR     PC,FLTSUB
7303  032444  174137  001432                    STF     AC1,@#FLTMP1    ;SAVE IN MEMORY
7304  032450  013737  001404  001264            MOV     @#$AC1,@#$REG3
7305                                    ;NOW COMPUTE THE RESULTS
7306                                    ;CALCULATE THE NUMBER OF CORRECT BITS
7307  032456  023737  001406  001410            CMP     @#$AC2,@#$AC3   ;DETERMINE WHICH EXP IS LARGER
7308  032464  002003                            BGE     2$              ;BRANCH IF AC2 LARGER
7309  032466  013737  001410  001406            MOV     @#$AC3,@#$AC2   ;PUT LARGEST IN AC2
7310  032474  163737  001404  001406  2$:       SUB     @#$AC1,@#$AC2
7311  032502  162737  000066  001406            SUB     #54.,@#$AC2
7312  032510  005437  001406                    NEG     @#$AC2
7313  032514  172437  001422                    LDF     @#FLTMP0,AC0    ;GET LEFT HAND SIDE
```

```
7314  032520  013737  001262  001402          MOV     @#$REG2,@#$AC0
7315  032526  004767  000220                  JSR     PC,FLTSUB       ;SUBTRACT TO SEE HOW CLOSE THEY ARE
7316  032532  163737  001264  001404          SUB     @#$REG3,@#$AC1  ;SUB EXT EXPONENTS
7317                                                                  ;ACTUAL EXPONENTS ARE EQUAL
7318  032540  100002                          BPL     1$              ;MAKE SURE RESULT IS POSITIVE
7319  032542  005437  001404                  NEG     @#$AC1
7320  032546  023737  001406  001404  1$:     CMP     @#$AC2,@#$AC1   ;RESULTS WITHIN RANGE ALLOWED?
7321  032554  003401                          BLE     SECT3D          ;BRANCH IF YES
7322  032556  104016                          ERROR   16              ;RESULTS WRONG
7323
7324                                  ;;**********************************************************************
7325  032560  172537  001276          SECT3D: LDF     @#$TMP0,AC1     ;LOAD OPERAND A
7326  032564  172437  001306                  LDF     @#$TMP4,AC0     ;AND OPERAND B
7327  032570  013737  001256  001404          MOV     @#$REG0,@#$AC1
7328  032576  013737  001260  001402          MOV     @#$REG1,@#$AC0
7329  032604  004767  000120                  JSR     PC,FLTDIV       ;GO DIVIDE THEM
7330  032610  004767  000072                  JSR     PC,FLTMPY       ;MULTIPLY RESULT BY B
7331  032614  174137  001422                  STF     AC1,@#FLTMP0    ;SAVE RESULT
7332  032620  013737  001404  001262          MOV     @#$AC1,@#$REG2
7333  032626  172437  001276                  LDF     @#$TMP0,AC0     ;LOAD OPERAND A
7334  032632  174037  001432                  STF     AC0,@#FLTMP1    ;SAVE INCASE TYPE OUT
7335  032636  013737  001256  001402          MOV     @#$REG0,@#$AC0
7336  032644  013737  001256  001264          MOV     @#$REG0,@#$REG3
7337  032652  004767  000074                  JSR     PC,FLTSUB       ;SUBTRACT RIGHT AND LEFT HAND SIDES
7338  032656  163737  001256  001404          SUB     @#$REG0,@#$AC1  ;SEE IF RESULT OK
7339  032664  100002                          BPL     1$              ;ENSURE DIFFERANCE IS POSITIVE
7340  032666  005437  001404                  NEG     @#$AC1
7341  032672  022737  000067  001404  1$:     CMP     #55.,@#$AC1     ;RESULTS WITHIN 2 BITS?
7342  032700  003505                          BLE     RELE8           ;BRANCH IF YES
7343  032702  104016                          ERROR   16              ;RESULTS WRONG
7344  032704  000503                          BR      RELE8
7345
7346                                  ;;**********************************************************************
7347                                  .SBTTL  FLOATING POINT MULTIPLY ROUTINE
7348                                  ;*      THIS ROUTINE MULTIPLIES THE CONTENTS OF AC0 AND AC1
7349                                  ;*      AND LEAVES THE RESULT IN AC1. IT ALSO TAKES CARE OF
7350                                  ;*      THE SOFTWARE EXPONENTS THAT ARE KEPT IN $AC0 AND $AC1.
7351                                  ;;**********************************************************************
7352  032706  063737  001402  001404  FLTMPY: ADD     @#$AC0,@#$AC1   ;ADD SOFTWARE EXPONENTS
7353  032714  171100                          MULF    AC0,AC1         ;DO THE MULTIPLY
7354  032716  012746  100400                  MOV     #100400,-(SP)   ;PUT CONTROL WORD ON STACK
7355  032722  004737  051040                  JSR     PC,@#EXPEXT     ;CALCULATE EXT EXPONENT
7356  032726  000207                  1$:     RTS     PC              ;RETURN
7357
7358                                  ;;**********************************************************************
7359                                  .SBTTL  FLOATING POINT DIVIDE ROUTINE
7360                                  ;*      THIS ROUTINE DIVIDES THE CONTENTS OF AC1 BY AC0
7361                                  ;*      AND LEAVES THE RESULT IN AC1.
7362                                  ;;**********************************************************************
7363  032730  163737  001402  001404  FLTDIV: SUB     @#$AC0,@#$AC1   ;ADJUST SOFTWARE EXPONENTS
7364  032736  174500                          DIVF    AC0,AC1         ;EXECUTE THE DIVIDE
7365  032740  012746  100400                  MOV     #100400,-(SP)   ;PUT CONTROL WORD ON STACK
7366  032744  004737  051040                  JSR     PC,@#EXPEXT     ;CALCULATE EXT EXPONENT
7367  032750  000207                  1$:     RTS     PC              ;RETURN
7368
7369                                  ;;**********************************************************************
```

```
7370                                    .SBTTL  FLOATING POINT ADD ROUTINE
7371                                    ;*      THIS ROUTINE ADDS THE CONTENTS OF AC0 TO AC1.
7372                                    ;*      THIS CAN ONLY BE DONE IF THE SOFTWARE EXPONENTS
7373                                    ;*      ARE CLOSE ENOUGH TOGETHER SUCH THAT AN ADJUSTMENT
7374                                    ;*      OF THE REAL EXPONENT LEAVES A NON-ZERO NUMBER.
7375                                    ;;*****************************************************************
7376  032752  010667  000134           FLTSUB: MOV     SP,SUBFLG       ;SET SUBTRACT FLAG
7377  032756  023737  001402  001404   FLTADD: CMP     @#$AC0,@#$AC1   ;CHECK SOFTWARE EXPONENTS
7378  032764  003016                           BGT     1$
7379  032766  001434                           BEQ     2$
7380                                    ;ACCUMULATOR 1 IS LARGER THAN ACCUMULATOR 0
7381  032770  013702  001404                   MOV     @#$AC1,R2       ;GET OPERAND B SOFTWARE EXP
7382  032774  163702  001402                   SUB     @#$AC0,R2       ;GET DIFFERENCE IN SOFTWARE EXP'S
7383  033000  020227  000071                   CMP     R2,#57.         ;EXP WITHIN DBL PREC RANGE?
7384  033004  002003                           BGE     7$              ;BRANCH IF ADD NOT REQUIRED
7385                                                                   ;RESULT IS OPERAND B
7386  033006  005402                           NEG     R2
7387  033010  176402                           LDEXP   R2,AC0          ;RELOAD THE EXPONENT
7388  033012  000422                           BR      2$
7389  033014  176427  177703           7$:     LDEXP   #-75,AC0        ;FAKE EXPONENT SO HARDWARE
7390  033020  000417                           BR      2$              ;WILL DETECT OUT OF RANGE
7391                                    ;
7392                                    ;ACCUMULATOR 0 IS LARGER THAN ACCUMULATOR 1
7393  033022  013702  001402           1$:     MOV     @#$AC0,R2       ;GET SOFTWARE EXP OF OPERAND A
7394  033026  163702  001404                   SUB     @#$AC1,R2       ;GET DIFFERENCE IN EXP'S
7395  033032  013737  001402  001404           MOV     @#$AC0,@#$AC1   ;MAKE SOFTWARE EXP'S EQUAL
7396  033040  020227  000071                   CMP     R2,#57. ;EXP WITHIN DBL PREC RANGE?
7397  033044  002003                           BGE     4$              ;BRANCH IF NO
7398  033046  005402                           NEG     R2
7399  033050  176502                           LDEXP   R2,AC1          ;RELOAD THE EXPONENT
7400  033052  000402                           BR      2$
7401                                    ;
7402                                    ;ACCUMULATOR 0 IS MUCH LARGER THAN ACCUMULATOR 1 SO RESULT IS 0
7403  033054  176527  177703           4$:     LDEXP   #-75,AC1        ;FAKE EXPONENT SO HARDWARE
7404                                                                   ;WILL DETECT OUT OF RANGE
7405  033060  005767  000026           2$:     TST     SUBFLG          ;ADD OR SUBTRACT?
7406  033064  001402                           BEQ     5$              ;BRANCH IF ADD
7407  033066  173100                           SUBF    AC0,AC1
7408  033070  000401                           BR      6$
7409  033072  172100                   5$:     ADDF    AC0,AC1         ;EXECUTE THE ADD
7410  033074  012746  100400           6$:     MOV     #100400,-(SP)   ;PUT CONTROL WORD ON STACK
7411  033100  004737  051040                   JSR     PC,@#EXPEXT     ;CALCULATE EXT EXPONENT
7412  033104  005067  000002           3$:     CLR     SUBFLG          ;INIT SUBTRACT FLAG
7413  033110  000207                           RTS     PC              ;RETURN
7414  033112  000000                   SUBFLG: .WORD
7415  033114  000004                   RELE8:  SCOPE
7416  033116  010702                           MOV     PC,R2
7417  033120  062702  000012                   ADD     #12,R2
7418  033124  012707  034240                   MOV     #RELOC,PC       ;GO RELOCATE PROGRAM CODE
7419  033130  000000                   REL88:  .WORD   0
7420                                    ;8888888888888 LAST ADDRESS OF CODE TO BE RELOCATED 88888888888
7421
7422                                    ;;*****************************************************************
7423                                    ;*TEST 74        TELETYPE AND CLOCK TESTS
7424                                    ;;*****************************************************************
7425  033132  112737  000074  001202   TST74:  MOVB    #74,@#STSTNM            ;LOAD TEST NUMBER
```

```
7426  033140  000240                          NOP
7427  033142  113737  001202  177570          MOVB    @#$TSTNM,@#SWR
7428  033150  005037  001506          TTYCHK: CLR     @#FACTOR
7429  033154  012704  000100                  MOV     #100,R4         ;SET R4 = CONSTANT 100
7430  033160  032737  000400  001470          BIT     #TTOPT,@#OPT.CP ;BRANCH IF TTY
7431  033166  001002                          BNE     1$              ;ON SYSTEM
7432  033170  000167  000204                  JMP     ARBFIN          ;JUMP IF NOT
7433  033174  122777  000200  146042  1$:     CMPB    #200,@$TPS      ;CHECK IF TTY IS READY
7434  033202  001374                          BNE     1$
7435  033204  012737  001525  001270          MOV     #NULLS-1,@#$REGS;SET ADDRESS OF ASCII STRING TO TYPE
7436  033212  106277  146026                  ASRB    @$TPS           ;SET IE BIT. SEE TPISR FOR INT SERVICE.
7437  033216  000001                          WAIT                    ;WAIT FOR INTERRUPT
7438
7439
7440  033220                          DUMMY:
7441                                  ;ROUTINE TO CHECK PRIORITY ARBITRATION LOGIC
7442                                  ;THE BELOW TEST WILL INHIBIT INTERRUPTS ON LEVEL 6 AND ABOVE (LOCKING
7443                                  ;OUT THE LINE CLOCK) AND THEN SET UP THE TTY TO INTERRUPT. NEXT THE
7444                                  ;PRIORITY LEVEL WILL BE SET TO 0 ALLOWING INTERRUPTS IN WHICH CASE
7445                                  ;THE LINE CLOCK (AT LEVEL 6) SHOULD INTERRUPT BEFORE THE TTY (AT LEVEL 4).
7446  033220  132[illegible]  000020  177776  1$:     BITB    #20,@#PSW
7447  033226  00010[illegible]              BNE     ARBEX           ;EXIT TEST IF 'T' BIT SET
7448  033230  030477  146010          2$:     BIT     R4,@$TPS        ;WAIT FOR TTY TO BE NOT
7449  033234  001375                          BNE     2$              ;BUSY
7450  033236  112737  000300  177776          MOVB    #300,@#PSW      ;SET PRIORITY LEVEL 6
7451  033244  150477  145774          3$:     BISB    R4,@$TPS        ;SET IE BIT
7452  033250  100375                          BPL     3$              ;AND WAIT FOR READY
7453  033252  032737  001000  001470          BIT     #LKOPT,@#OPT.CP ;LINE CLOCK AVAILABLE?
7454  033260  001447                          BEQ     ARBFIN          ;BRANCH IF NO
7455  033262  012737  033352  000064          MOV     #7$,@#TPVEC     ;SET TTY VECTOR
7456  033270  012737  033364  000100          MOV     #8$,@#LKVEC     ;SET CLOCK VECTORS
7457  033276  012737  000340  000102          MOV     #PR7,@#LKVEC+2
7458  033304  005027                          CLR     (PC)+           ;CLEAR CHECK WORD
7459  033306  000000                  4$:     .WORD   0
7460  033310  000240                          NOP
7461  033312  000240                          NOP
7462  033314  000240                          NOP
7463  033316  010437  177546                  MOV     R4,@#LKS
7464  033322  113700                  5$:     MOVB    @(PC)+,R0       ;GET CLOCK STATUS & BRANCH IF READY
7465  033324  177546                  6$:     .WORD   LKS             ;CONTAINS ADDRESS OF L CLOCK STAT
7466  033326  100375                          BPL     5$
7467  033330  000240                          NOP                     ;AT THIS TIME BOTH THE CLOCK
7468                                                                  ;ARE READY TO INTERRUPT
7469  033332  105037  177776                  CLRB    @#PSW           ;SET PRIORITY LEVEL 0
7470                                  ;A CLOCK INTERRUPT WILL OCCUR (8$) AND LOC 4$ WILL BE INCREMENTED
7471                                  ;AFTER THE CLOCK SERVICE A TTY INTERRUPT WILL OCCUR. THE TTY INT SERV
7472                                  ;ICE WILL SHIFT LEFT 4$.
7473
7474  033336  022767  000002  177742          CMP     #2,4$           ;CHECK THAT THE CLOCK
7475  033344  001415                          BEQ     ARBFIN          ;& TTY INTERRUPTED IN
7476  033346  104000                          HLT                     ;THE PROPER SEQUENCE
7477  033350  000413                          BR      ARBFIN
7478
7479  033352  005077  145666          7$:     CLR     @$TPS           ;CLEAR IE BIT
7480  033356  006367  177724                  ASL     4$              ;SHIFT INDICATOR
7481  033362  000002                          RTI                     ;RETURN
```

```
7482
7483  033364  005267  177716          8$:     INC     4$
7484  033370  012737  044442  000100          MOV     #LKSRV,@#LKVEC          ;SET CLOCK VECTORS
7485  033376  000002                          RTI
7486
7487
7488  033400  012737  052610  000064  ARBFIN: MOV     #TPISR,@#TPVEC          ;RESTORE TTY VECTOR
7489  033406  005077  145632                  CLR     @$TPS                   ;CLEAR IE BIT
7490  033412                          ARBEX:
7491                                  ;;**************************************************************
7492                                  ;*TEST 75       TURN ON UBE AND MBT
7493                                  ;*      TURN ON THE MASS BUS TESTER AND UNIBUS EXERCISER IF PRESENT
7494                                  ;;**************************************************************
7495  033412  112737  000075  001202  TST75:  MOVB    #75,@#$TSTNM            ;LOAD TEST NUMBER
7496  033420  000240                          NOP
7497  033422  113737  001202  177570          MOVB    @#$TSTNM,@#SWR
7498  033430  032737  001000  001470          BIT     #LKOPT,@#OPT.CP         ;BRANCH IF NOT AVAIL
7499  033436  001411                          BEQ     UBESET
7500  033440  012737  044442  000100          MOV     #LKSRV,@#LKVEC
7501  033446  012737  000340  000102          MOV     #PR7,@#LKVEC+2
7502  033454  052737  000100  177546          BIS     #100,@#LKS              ;SET IE BIT
7503
7504                                  ;;**************************************************************
7505                                  ;TURN ON THE UNIBUS EXERCISER IF PRESENT
7506  033462  105737  001470          UBESET: TSTB    @#OPT.CP                ;IS UBE OPTION AVAILABLE?
7507  033466  100015                          BPL     MBTSET                  ;BRANCH IF NO
7508  033470  032737  010000  177570          BIT     #SW12,@#SWR             ;INHIBIT UBE?
7509  033476  001011                          BNE     MBTSET                  ;BRANCH IF YES
7510  033500  032737  000040  172516          BIT     #BIT5,@#MMR3            ;IS MAP ON?
7511  033506  001050                          BNE     STMM                    ;BRANCH IF YES
7512  033510  004737  051434                  JSR     PC,@#UBEINIT            ;INITIALIZE UBE
7513  033514  012772  064545  000000          MOV     #64545,@(R2)            ;START UBE
7514
7515                                  ;;**************************************************************
7516                                  ;TURN ON THE MASS BUS TESTER IF PRESENT
7517  033522  032737  002000  001470  MBTSET: BIT     #MBTOPT,@#OPT.CP        ;IS MBT AVAILABLE?
7518  033530  001437                          BEQ     STMM                    ;BRANCH IF NO
7519  033532  032737  000010  177570          BIT     #SW3,@#SWR              ;INHIBIT MBT?
7520  033540  001033                          BNE     STMM                    ;BRANCH IF YES
7521  033542  122737  000060  001532          CMPB    #60,@#SUBPASS           ;FIRST SUB-PASS?
7522  033550  001027                          BNE     STMM                    ;BRANCH IF NO
7523  033552  105737  001200                  TSTB    @#$PASS                 ;FIRST PASS?
7524  033556  001024                          BNE     STMM                    ;BRANCH IF NO
7525  033560  105737  001503                  TSTB    @#MMON                  ;MEM MGMT ON?
7526  033564  001021                          BNE     STMM                    ;BRANCH IF YES
7527  033566  052777  000047  146464  MBT1:   BIS     #47,@MBTTBL+12          ;CLEAR THE MBT
7528  033574  012777  000007  146456          MOV     #7,@MBTTBL+12           ;SELECT UNIT 7
7529  033602  005077  146442                  CLR     @MBTTBL+2               ;CLEAR THE WORD COUNT
7530  033606  012777  044130  146454          MOV     #MBTSRV,@MBTTBL+22      ;SETUP INTERRUPT VECTOR
7531  033614  012777  000240  146450          MOV     #PR5,@MBTTBL+24         ;SET VECTOR PSW
7532  033622  112777  000161  146416          MOVB    #161,@MBTTBL            ;START MBT
7533
7534                                  ;;**************************************************************
7535                                  .SBTTL  STMM ROUTINE
7536                                  ;ROUTINE TO SET UP MEMORY MANAGEMENT TO RELOCATE PROGRAM CODE ABOVE 16K
7537                                  ;CHECK IF PROGRAM IS TO BE RELOCATED.
```

```
                                        ;SW6=1=NO RELOCATION
                                        ;;*****************************************************************
033630  032737  000100  177570  STMM:   BIT     #SW6,@#SWR      ;RELOCATION DISABLED?
033636  001402                          BEQ     3$              ;BRANCH IF NO
033640  000167  002450                  JMP     ENDM

                                        ;THE PROGRAM IS GOING TO RELOCATE.
                                        ;RELOCATION WILL BE PERFORMED IN KERNEL MODE WITH PSW SET AT PRIORITY
                                        ;LEVEL 4 (TO PREVENT TTY INTERRUPT-WHICH CHANGES DATA IN PROGRAM)
                                        ;THE 'T' BIT IS CLEARED (IF SET). AFTER THE DATA HAS BEEN WRITTEN IT IS
                                        ;VERIFIED BEFORE EXECUTION.
033644  013727  177776          3$:     MOV     @#PSW,(PC)+     ;SAVE CURRENT PSW
033650  000000          OLDPSW: .WORD   0
033652  012737  000200  177776          MOV     #PR4,@#PSW
033660  004767  016452                  JSR     PC,CLRTBIT      ;CO CLEAR 'T' BIT IF SET

                                        ;NOW SETUP MEMORY MANAGEMENT REGISTERS.
033664  012700  077406                  MOV     #77406,R0       ;SET CONSTANT=R/W UP 4K WORDS
033670  010037  172300                  MOV     R0,@#KIPDR0     ;SET KIPDR0,1,2,3,& 7 R/W UP 4K WORDS
033674  010037  172302                  MOV     R0,@#KIPDR1
033700  010037  172304                  MOV     R0,@#KIPDR2
033704  010037  172306                  MOV     R0,@#KIPDR3
033710  010037  172310                  MOV     R0,@#KIPDR4
033714  010037  172312                  MOV     R0,@#KIPDR5
033720  010037  172316                  MOV     R0,@#KIPDR7

033724  005037  172340                  CLR     @#KIPAR0        ;NOTE: THESE 2 INSTRUCIONS EFFECTIVELY
033730  012737  000200  172342          MOV     #200,@#KIPAR1   ;RELOCATE PROGRAM EXECUTION
033736  012737  000400  172344          MOV     #400,@#KIPAR2
033744  013737  001520  172346          MOV     @#NEXPAR,@#KIPAR3       ;SET UP KIPAR3 & KIPAR4 & 5
033752  013737  172346  172350          MOV     @#KIPAR3,@#KIPAR4
033760  062737  000200  172350          ADD     #200,@#KIPAR4
033766  013737  172346  172352          MOV     @#KIPAR3,@#KIPAR5
033774  062737  000400  172352          ADD     #400,@#KIPAR5
034002  012737  177600  172356          MOV     #177600,@#KIPAR7;AND OF COUSE THE I/O PAGE
                                ;NOW SETUP USER MEM MGMT REGISTERS
034010  010037  177600          1$:     MOV     R0,@#UIPDR0     ;SET UP USER MEM MGMT REGS
034014  010037  177602                  MOV     R0,@#UIPDR1
034020  010037  177604                  MOV     R0,@#UIPDR2
034024  010037  177616                  MOV     R0,@#UIPDR7
034030  016737  145464  177640          MOV     NEXPAR,@#UIPAR0
034036  013737  177640  177642          MOV     @#UIPAR0,@#UIPAR1
034044  062737  000200  177642          ADD     #200,@#UIPAR1
034052  013737  177640  177644          MOV     @#UIPAR0,@#UIPAR2
034060  062737  000400  177644          ADD     #400,@#UIPAR2
034066  013737  172356  177656          MOV     @#KIPAR7,@#UIPAR7

034074  010037  172200                  MOV     R0,@#SIPDR0     ;SET UP SUPERVISOR MEM MGMT REGS
034100  010037  172202                  MOV     R0,@#SIPDR1
034104  010037  172204                  MOV     R0,@#SIPDR2
034110  010037  172216                  MOV     R0,@#SIPDR7
034114  016737  145400  172240          MOV     NEXPAR,@#SIPAR0
034122  013737  172240  172242          MOV     @#SIPAR0,@#SIPAR1
034130  062737  000200  172242          ADD     #200,@#SIPAR1
034136  013737  172240  172244          MOV     @#SIPAR0,@#SIPAR2
034144  062737  000400  172244          ADD     #400,@#SIPAR2
```

```
7594  034152  013737  172356  172256          MOV     @#KIPAR7,@#SIPAR7
7595  034160  012737  000001  177572          MOV     #1,@#SR0        ;ENABLE MEM MGMT
7596  034166  012737  000060  172516          MOV     #60,@#SR3       ;SETUP SR3
7597  034174  110637  001503                  MOVB    SP,@#MMON       ;SET MEM MGMT ON IND = ON
7598  034200  005037  000006          RETRY:  CLR     @#ERRVEC+2
7599  034204  012737  036312  000004          MOV     #ENDMEM,@#ERRVEC;SET TIME OUT TRAP VECTOR
7600  034212  012702  060000                  MOV     #60000,R2       ;SETUP GENERAL REGISTERS
7601  034216  005000                          CLR     R0              ;DATA WILL BE RELOCATED FROM
7602                                                                  ;ADDRESS IN R0 TO ADDRESS IN R2
7603  034220  012703  137776                  MOV     #137776,R3      ;GET 12K WORDS TO RELOCATE
7604  034224  010013                          MOV     R0,(R3)         ;TRAP TO ENDMEM IF INSUFFICIENT MEMORY
7605  034226  012737  053440  000004          MOV     #ERPRT,@#ERRVEC ;RESTORE ERROR TRAP VECTOR
7606  034234  000137  034434                  JMP     @#IOMON
7607                                  ;;*****************************************************************
7608                                  .SBTTL  RELOCATION ROUTINE
7609                                  ;*      THIS ROUTINE IS USED TO RELOCATE THE 9 SUBTESTS UP TO 28K.
7610                                  ;*      IF RELOCATION BY AN I/O DEVICE IS SELECTED, CONTROL IS PASSED
7611                                  ;*      TO THE I/O MONITOR.
7612                                  ;*      ENTER WITH:
7613                                  ;*              FRSTAD=PHYSICAL ADDRESS OF FIRST CODE
7614                                  ;*              FACTOR=NUMBER OF BYTES ABOVE BASE CODE
7615                                  ;*              R2     =LAST PHYSICAL ADDRESS OF THE SECTION
7616                                  ;*      EXIT TO I/O MONITOR WITH:
7617                                  ;*              OLDBASE=FIRST PHYSICAL ADDRESS TO BE RELOCATED
7618                                  ;*              NWBASL =FIRST PHYSICAL ADDRESS TO RELOCATE TO
7619                                  ;*              IOWC   =TWO'S COMPLIMENT WORD COUNT
7620                                  ;;*****************************************************************
7621  034240  032737  000100  177570  RELOC:  BIT     #SW6,@#SWR              ;IS RELOCATION DISABLED?
7622  034246  001067                          BNE     EXITRE                  ;BRANCH IF YES
7623  034250  105737  001503                  TSTB    @#MMON                  ;IS MEMORY MGMT ON?
7624  034254  001064                          BNE     EXITRE                  ;BRANCH IF YES
7625  034256  013700  001512                  MOV     @#FRSTAD,R0             ;GET FIRST ADDRESS TO BE RELOCATED
7626  034262  010005                          MOV     R0,R5
7627                                  ;LAST ADDRESS IS IN R2
7628  034264  010203                          MOV     R2,R3                   ;SAVE LAST ADDRESS
7629  034266  010204                          MOV     R2,R4
7630  034270  160004                          SUB     R0,R4                   ;R4 NOW HAS BYTE COUNT
7631  034272  010437  001510                  MOV     R4,@#$FACTOR            ;SAVE BYTE COUNT
7632  034276  005737  001506                  TST     @#FACTOR                ;FIRST RELOC IS TO ENDTAG+2
7633  034302  001004                          BNE     1$                      ;BRANCH IF NOT EXECUTING BASE CODE
7634  034304  010237  034432                  MOV     R2,@#RETPC              ;SAVE RETURN PC TO NEXT SECTION
7635  034310  013702  001514                  MOV     @#FRSTMEM,R2            ;GET FIRST ADDRESS TO RELOCATE TO
7636  034314  060204                  1$:     ADD     R2,R4                   ;R4 NOW CONTAINS LAST MEM ADDRESS
7637  034316  020437  001516                  CMP     R4,@#LSTMEM             ;ENOUGH MEMORY?
7638  034322  101042                          BHI     NOMEM                   ;BRANCH IF NO
7639  034324  160204                          SUB     R2,R4                   ;R4 NOW HAS BYTE COUNT
7640  034326  005037  001506                  CLR     @#FACTOR
7641  034332  032737  000400  177570          BIT     #SW8,@#SWR              ;INHIBIT RELOC BY I/O DEVICE?
7642  034340  001014                          BNE     RELNIO                  ;BRANCH IF YES
7643  034342  010037  001540                  MOV     R0,@#OLDBASE            ;SAVE START ADDRESS
7644  034346  010237  001542                  MOV     R2,@#NWBASL             ;SAVE NEW BASE ADDRESS
7645  034352  005037  001544                  CLR     @#NWBASH                ;
7646  034356  006204                          ASR     R4                      ;MAKE IT A WORD COUNT
7647  034360  005404                          NEG     R4                      ;GET TWO'S COMPLIMENT
7648  034362  010437  001546                  MOV     R4,@#IOWC               ;SAVE R4 AS WORDCOUNT
7649  034366  000167  000120                  JMP     ENTER2                  ;GO TO I/O MONITOR
```

```
                                        ;RELOCATE BY CPU-MEMORY MANAGEMENT OFF
034372  012022                  RELNIO: MOV     (R0)+,(R2)+             ;RELOCATE CODE
034374  020003                          CMP     R0,R3                   ;DONE YET?
034376  001375                          BNE     RELNIO                  ;BRANCH IF NO
034400  004737  051744                  JSR     PC,@#CHKDAT             ;GO CHECK DATA
034404  102010                          BVC     EXITRE
034406  010037  001276                  MOV     R0,@#$TMP0              ;SAVE R0 FOR TYPEOUT
034412  010237  001474                  MOV     R2,@#VADR               ;SAVE R2
034416  004737  001642                  JSR     PC,@#CNVADR             ;CONVERT R2 TO A PHYSICAL ADR
034422  104006                          ERROR   6
034424  000401                          BR      NOMEM
034426  010207                  EXITRE: MOV     R2,PC                   ;GO EXECUTE RELOCATED CODE
034430  011707                  NOMEM:  MOV     (PC),PC                 ;GO TO NEXT SECTION
034432  000000                  RETPC:  .WORD   0                       ;CONTAINS PC OF NEXT SECTION
                                        ;;**************************************************************
                                        .SBTTL  I/O RELOCATION MONITOR
                                        ;*      THIS ROUTINE IS USED TO SCHEDULE I/O DEVICES FOR SUBTEST
                                        ;*      RELOCATION AND PROGRAM RELOCATION. THE I/O DEVICE UNIT
                                        ;*      NUMBER IS DETERMINED, THE BUS ADDRESS CALCULATED, THE WORD
                                        ;*      COUNT CALCULATED AND PASSED TO THE DEVICE HANDLER.
                                        ;;**************************************************************
034434  012737  034442  001212  IOMON:  MOV     #1$,@#$LPERR    ;SETUP ERROR LOOP
034442  005037  001540          1$:     CLR     @#OLDBASE
034446  013705  172346                  MOV     @#KIPAR3,R5             ;SETUP R2 AND R3
034452  005004                          CLR     R4                      ;TO FORM 22 BIT ADDRESS
034454  073427  000006                  ASHC    #6,R4                   ;FORM 22 BIT ADDRESS
034460  010537  001542                  MOV     R5,@#NWBASL             ;SAVE LOWER 16 BITS
034464  010437  001544                  MOV     R4,@#NWBASH             ;SAVE UPPER 6 BITS
034470  032737  000400  177570          BIT     #SW8,@#SWR              ;RELOCATE VIA I/O?
034476  001402                          BEQ     2$                      ;BRANCH IF YES
034500  000167  001436                  JMP     RELOCP                  ;GO RELOCATE VIA CP
034504  012737  174000  001546  2$:     MOV     #174000,@#IOWC          ;SET WORD COUNT TO 2K
034512  005037  001302          ENTER2: CLR     @#$TMP2
034516  012737  177776  001556          MOV     #-2,@#RNTBINX           ;SETUP RUN TABLE INDEX
034524  005037  001276                  CLR     @#$TMP0
034530  005002                          CLR     R2                      ;CLEAR LEGAL DEV FLAG
034532  032737  000040  177570  41$:     BIT     #SW5,@#SWR              ;INHIBIT ROUND ROBIN?
034540  001416                          BEQ     50$                     ;BRANCH IF NO
034542  005737  001276                  TST     @#$TMP0                 ;FLAG SET?
034546  001027                          BNE     43$                     ;BRANCH IF YES
034550  113737  177570  001552          MOVB    @#SWR,@#DEVINDX         ;GET DEVICE FROM SWITCHES
034556  042737  177770  001552          BIC     #177770,@#DEVINDX       ;MASK LOWER 3 BITS
034564  006337  001552                  ASL     @#DEVINDX               ;ADJUST FOR WORD INDEX
034570  005237  001276                  INC     @#$TMP0                 ;SET FLAG
034574  000414                          BR      43$                     ;CONTINUE
034576  012705  000010          50$:    MOV     #10,R5                  ;SET SOB COUNT
034602  022737  000016  001552  40$:     CMP     #16,@#DEVINDX           ;LAST DEVICE YET?
034610  001003                          BNE     42$                     ;BRANCH IF NO
034612  012737  177776  001552  48$:     MOV     #-2,@#DEVINDX           ;INIT DEVICE INDEX
034620  062737  000002  001552  42$:     ADD     #2,@#DEVINDX            ;INCREMENT INDEX
034626  013703  001552          43$:     MOV     @#DEVINDX,R3            ;GET INDEX
034632  012737  000401  001300          MOV     #401,@#$TMP1            ;INIT UNIT MASK
034640  012704  000010                  MOV     #10,R4          ;SET SOB COUNT
034644  133763  001300  001606  44$:     BITB    @#$TMP1,SYSSIZE(R3)     ;IS THIS UNIT EXISTENT?
034652  001405                          BEQ     52$                     ;BRANCH IF NO
034654  005202                          INC     R2                      ;SET LEGAL DEVICE FLAG
```

```
7706  034656  133763  001301  001607          BITB    @#$TMP1+1,SYSSIZE+1(R3) ;HAS IT BEEN USED?
7707  034664  001520                          BEQ     11$                     ;BRANCH IF NO
7708  034666  006337  001300          52$:    ASL     @#$TMP1                 ;SELECT NEXT UNIT
7709  034672  077414                          SOB     R4,44$                  ;CONTINUE
7710  034674  005737  001276                  TST     @#$TMP0                 ;INHIBIT ROUND ROBIN?
7711  034700  001013                          BNE     45$                     ;BRANCH IF YES
7712  034702  077541                          SOB     R5,40$                  ;CONTINUE
7713  034704  005702                          TST     R2                      ;ANY DEVICES AT ALL?
7714  034706  001442                          BEQ     46$                     ;BRANCH IF NO
7715  034710  012704  000010                  MOV     #10,R4                  ;SET SOB COUNT
7716  034714  012701  001607                  MOV     #SYSSIZE+1,R1           ;GET ADR OF SIZE TABLE
7717  034720  105021          47$:    CLRB    (R1)+                   ;CLEAR ALL USED BITS
7718  034722  005201                          INC     R1                      ;IN ALL DEVICES
7719  034724  077403                          SOB     R4,47$                  ;CONTINUE
7720  034726  000701                          BR      41$
7721  034730  005702          45$:    TST     R2                      ;WAS IT A LEGAL DEVICE?
7722  034732  001403                          BEQ     49$                     ;BRANCH IF NO
7723  034734  105063  001607                  CLRB    SYSSIZE+1(R3)           ;CLEAR ALL USED BITS THIS DEV
7724  034740  000732                          BR      43$
7725  034742  010367  000016          49$:    MOV     R3,60$                  ;
7726  034746  062767  053610  000010          ADD     #MSGINX,60$             ;GEN MESSAGE ADR
7727  034754  017767  000004  000002          MOV     @60$,60$
7728  034762  104400                          TYPE
7729  034764  000000          60$:    .WORD
7730  034766  104400  034774                  TYPE    ,65$            ;;TYPE ASCIZ STRING
7731  034772  000407                          BR      64$             ;;GET OVER THE ASCIZ
7732                          ;;65$:  .ASCIZ  /UNAVAILABLE/<CRLF>
7733  035012          64$:
7734  035012  000637                          BR      ENTER2
7735  035014  105737  001504          46$:    TSTB    @#QV                    ;ACT11?
7736  035020  001016                          BNE     51$                     ;BRANCH IF YES
7737  035022  005227  177777                  INC     #-1
7738  035026  001013                          BNE     51$
7739  035030  104400  035036                  TYPE    ,67$            ;;TYPE ASCIZ STRING
7740  035034  000410                          BR      66$             ;;GET OVER THE ASCIZ
7741                          ;;67$:  .ASCIZ  ?NO I/O DEVICES?<CRLF>
7742  035056          66$:
7743  035056  105737  001503          51$:    TSTB    @#MMON                  ;MGMT ON?
7744  035062  001012                          BNE     61$                     ;BRANCH IF YES
7745  035064  013700  001540                  MOV     @#OLDBASE,R0            ;RESTORE R0
7746  035070  013702  001542                  MOV     @#NWBASL,R2             ;RESTORE R2
7747  035074  013703  001510                  MOV     @#$FACTOR,R3            ;GET RELOCATION FACTOR
7748  035100  060003                          ADD     R0,R3                   ;FORM LAST ADDRESS
7749  035102  010005                          MOV     R0,R5                   ;SETUP R5
7750  035104  000167  177262                  JMP     RELNIO                  ;GO RELOCATE WITH CP
7751  035110  012702  060000          61$:    MOV     #60000,R2               ;SETUP REGISTERS
7752  035114  012703  137776                  MOV     #137776,R3              ;WITH FROM
7753  035120  005000                          CLR     R0                      ;AND TOO ADDRESS
7754  035122  000137  036142                  JMP     @#RELOCP                ;RELOCATE VIA CP
7755  035126  105763  001676          11$:    TSTB    RP3HSTAT(R3)            ;IS HANDLER BUSY?
7756  035132  100405                          BMI     8$                      ;BRANCH IF NO
7757  035134  005737  001276                  TST     @#$TMP0                 ;ROUND ROBIN?
7758  035140  001372                          BNE     11$                     ;BRANCH IF NO
7759  035142  000167  177364                  JMP     41$
7760  035146  005763  001676          8$:     TST     RP3HSTAT(R3)            ;DID HANDLER FAIL?
7761  035152  100005                          BPL     62$                     ;BRANCH IF NO
```

```
7762  035154  005737  001276                        TST     @#$TMPO                 ;ROUND ROBIN
7763  035160  001402                                BEQ     62$                     ;BRANCH IF YES
7764  035162  000137  036122                        JMP     @#15$
7765  035166  153763  001301  001607  62$:          BISB    @#$TMP1+1,SYSSIZE+1(R3) ;SET UNIT USED BIT
7766  035174  005002                                CLR     R2
7767  035176  006037  001300          30$:          ROR     @#$TMP1                 ;ENCODE THE BIT POSITION
7768  035202  005202                                INC     R2                      ;INTO A UNIT NUMBER
7769  035204  103374                                BCC     30$
7770  035206  005302                                DEC     R2
7771  035210  010237  001554                        MOV     R2,@#UNITNO             ;SAVE UNIT NUMBER
7772  035214  013763  001546  001716  10$:          MOV     @#IOWC,RP3HWC(R3)       ;GIVE WORD COUNT TO HANDLER
7773  035222  010304                                MOV     R3,R4                   ;
7774  035224  072427  000003                        ASH     #3,R4                   ;ENCODE DEVICE FOR RUNTABLE
7775  035230  053704  001554                        BIS     @#UNITNO,R4             ;ENCODE UNIT NUMBER
7776  035234  006304                                ASL     R4
7777  035236  062737  000002  001556                ADD     #2,@#RNTBINX            ;INCREMENT RUN TABLE INDEX
7778  035244  013702  001556                        MOV     @#RNTBINX,R2            ;GET RUN TABLE INDEX
7779  035250  110462  001627                        MOVB    R4,RUNTBL+1(R2)         ;ENTER DEV & UNIT IN TABLE
7780  035254  013763  001554  002012                MOV     @#UNITNO,RP3UNIT(R3)    ;GIVE HANDLER UNIT NUMBER
7781  035262  012737  000240  000012                MOV     #PRS,@#RESVEC+2         ;SETUP RESERVED VECTOR PSW
7782  035270  016337  002026  000010                MOV     RP3HANA(R3),@#RESVEC    ;SETUP RESERVED VECTOR
7783  035276  006303                                ASL     R3                      ;ADJUST INDEX
7784  035300  013763  001540  001732                MOV     @#OLDBASE,RP3OLD(R3)    ;GIVE HANDLER OLD BASE ADDRESS
7785  035306  013763  001542  001762                MOV     @#NWBASL,RP3NWL(R3)     ;GIVE HANDLER
7786  035314  013763  001544  001764                MOV     @#NWBASH,RP3NWH(R3)     ;NEW BASE ADDRESS
7787  035322  005063  001734                        CLR     RP3OLD+2(R3)            ;ENSURE OLD BASE HIGH IS CLR
7788  035326  000010                                CALLHANDLER
7789  035330  105737  001503                        TSTB    @#MMON                  ;IS MEMORY MANAGEMENT ON?
7790  035334  001416                                BEQ     13$                     ;BRANCH IF NO
7791  035336  022737  000012  001556                CMP     #12,@#RNTBINX           ;TRANSFERED 12K YET?
7792  035344  001412                                BEQ     13$                     ;BRANCH IF YES
7793  035346  062737  010000  001540                ADD     #10000,@#OLDBASE        ;ADD 2K
7794  035354  062737  010000  001542                ADD     #10000,@#NWBASL         ;TO BASE
7795  035362  005537  001544                        ADC     @#NWBASH                ;ADDRESSES
7796  035366  000137  034532                        JMP     @#41$
7797  035372  113705  001603          13$:          MOVB    @#LTICKS+1,R5           ;GET SECOND COUNT
7798  035376  062705  000002                        ADD     #2,R5                   ;INCREMENT BY TWO
7799  035402  162705  000074                        SUB     #60.,R5                 ;ENSURE RESULT IS 59 OR LESS
7800  035406  100002                                BPL     31$                     ;
7801  035410  062705  000074                        ADD     #60.,R5                 ;COUNT WAS LESS THAN 58-RESTORE
7802  035414  012700  000010          31$:          MOV     #10,R0                  ;SET SOB COUNT
7803  035420  005002                                CLR     R2
7804  035422  005003                                CLR     R3
7805  035424  005004                                CLR     R4
7806  035426  066203  001676          14$:          ADD     RP3HSTAT(R2),R3         ;ADD ALL THE HANDLER
7807  035432  005504                                ADC     R4                      ;STATUS WORDS. WHEN ALL
7808  035434  062702  000002                        ADD     #2,R2                   ;TRANSFERS ARE FINISHED
7809  035440  077006                                SOB     R0,14$                  ;RESULT WILL BE 2000
7810  035442  006103                                ROL     R3                      ;(WITHOUT ROTATE)
7811  035444  005504                                ADC     R4
7812  035446  022703  004000                        CMP     #4000,R3                ;ALL DONE?
7813  035452  001406                                BEQ     32$                     ;BRANCH IF YES
7814  035454  123705  001603                        CMPB    @#LTICKS+1,R5           ;TWO SECONDS ELAPSED YET?
7815  035460  001355                                BNE     31$                     ;BRANCH IF NO
7816  035462  104015                                ERROR   15                      ;DEVICE HUNG
7817  035464  000177  143522                        JMP     @$LPERR                 ;RESTART RELOCATION
```

```
7818  035470  005704                    32$:  TST   R4                ;ANY DEVICE ERRORS?
7819  035472  001402                          BEQ   82$               ;BRANCH IF NO
7820  035474  000167  000422                  JMP   15$               ;ERROR
7821  035500  105737  001503              82$:  TSTB  @#MMON            ;MEM MGMT ON?
7822  035504  001012                          BNE   25$               ;BRANCH IF YES
7823  035506  013705  001540                  MOV   @#OLDBASE,R5      ;SETUP R5 FOR DATA CHECK
7824  035512  010500                          MOV   R5,R0
7825  035514  063700  001510                  ADD   @#$FACTOR,R0      ;GET LAST ADDRESS
7826                                                                  ;OF GOOD DATA
7827  035520  013702  001542                  MOV   @#NWBASL,R2
7828  035524  063702  001510                  ADD   @#$FACTOR,R2      ;GET LAST ADDRESS
7829                                                                  ;OF DATA TO BE CHECKED
7830  035530  000406                          BR    22$               ;CONTINUE
7831  035532  012700  057776              25$:  MOV   #57776,R0         ;GET LAST ADR OF GOOD DATA
7832  035536  012702  137776                  MOV   #137776,R2        ;GET LAST ADR OF DATA TO BE CHECKED
7833  035542  012705  002100                  MOV   #2100,R5          ;DON'T CHECK FIRST 2100 LOCATIONS
7834  035546  004737  051744              22$:  JSR   PC,@#CHKDAT       ;GO CHECK DATA
7835  035552  102413                          BVS   81$               ;BRANCH IF ERROR
7836  035554  105737  001204                  TSTB  @#$ERFLG          ;ANY ERRORS?
7837  035560  001002                          BNE   83$               ;BRANCH IF YES
7838  035562  000167  000412                  JMP   EXIT              ;RETURN
7839  035566  032737  001000  177570    83$:  BIT   #SW9,@#SWR        ;LOOP ON ERROR?
7840  035574  001473                          BEQ   100$+2            ;BRANCH IF NO
7841  035576  000167  000244                  JMP   20$               ;GO DO FUNCTION AGAIN
7842  035602  005001                    81$:  CLR   R1
7843  035604  010037  001276                  MOV   R0,@#$TMP0
7844  035610  010237  001474                  MOV   R2,@#VADR
7845  035614  010203                          MOV   R2,R3             ;SAVE ERROR ADDRESS
7846  035616  005004                          CLR   R4
7847  035620  105737  001503                  TSTB  @#MMON            ;IS MEM MGMT ON?
7848  035624  001406                          BEQ   16$               ;BRANCH IF NO
7849  035626  162703  010000              17$:  SUB   #10000,R3         ;SUBTRACT 2K FROM ERROR ADDRESS
7850  035632  100403                          BMI   16$               ;BRANCH IF BLOCK IS FOUND
7851  035634  062704  000002                  ADD   #2,R4             ;COUNT ONE MORE BLOCK
7852  035640  000772                          BR    17$               ;CONTINUE
7853                          ;R4 NOW CONTAINS INDEX OF ERROR FOR RUN TIME TABLE
7854  035642  116404  001627              16$:  MOVB  RUNTBL+1(R4),R4   ;GET DEVICE THAT FAILED
7855  035646  042704  177400                  BIC   #177400,R4        ;ENSURE HIGH BYTE CLEAR
7856  035652  006004                          ROR   R4                ;THROW AWAY LSB
7857  035654  005005                          CLR   R5                ;ENSURE R5 CLEAR
7858  035656  073427  177775                  ASHC  #-3,R4            ;GET UNIT NUMBER IN R5
7859  035662  010500                          MOV   R5,R0
7860  035664  072027  177764                  ASH   #-14,R0
7861  035670  042700  177770                  BIC   #177770,R0
7862  035674  010037  001304                  MOV   R0,@#$TMP3
7863  035700  010403                          MOV   R4,R3             ;AND DEVICE INDEX IN R4 & R3
7864  035702  010337  001302                  MOV   R3,@#$TMP2
7865  035706  012737  000001  001300          MOV   #1,@#$TMP1                ;ENCODE 3 BIT UNIT NO INTO
7866  035714  162705  020000              19$:  SUB   #20000,R5         ;ONE BIT IN THE LOW BYTE OF $TMP1
7867  035720  103403                          BCS   18$               ;BRANCH IF DONE
7868  035722  006137  001300                  ROL   @#$TMP1           ;SELECT NEXT UNIT
7869  035726  000772                          BR    19$               ;CONTINUE
7870  035730  012737  036046  001212    18$:  MOV   #20$,@#$LPERR     ;SETUP LOOP RETURN
7871  035736  005701                          TST   R1                ;DEVICE ERROR?
7872  035740  001010                          BNE   100$              ;BRANCH IF YES
7873  035742  104010                          ERROR 10                ;DATA CHECK ERROR
```

```
7874  035744  105737  001503                    TSTB    @#MMON                  ;MGMT ON?
7875  035750  001002                            BNE     70$                     ;BRANCH IF YES
7876  035752  000137  034512            71$:    JMP     @#ENTER2
7877  035756  000137  034434            70$:    JMP     @#IOMON
7878  035762  104007                    100$:   ERROR   7
7879  035764  042763  100000  001676            BIC     #BIT15,RP3HSTAT(R3)     ;CLEAR THE ERROR
7880  035772  022703  000002                    CMP     #2,R3                   ;RK05 ERROR?
7881  035776  002405                            BLT     90$                     ;BRANCH IF RH70
7882  036000  003016                            BGT     92$                     ;BRANCH IF RP03
7883  036002  112777  000001  144116            MOVB    #1,@RKCS                ;RK CONTROLLER CLEAR
7884  036010  000412                            BR      92$
7885  036012  022703  000012            90$:    CMP     #12,R3                  ;RS04?
7886  036016  001004                            BNE     91$                     ;BRANCH IF NO
7887  036020  052777  000040  144166            BIS     #BIT5,@RSCS2            ;CLEAR RS CONTROLLER
7888  036026  000403                            BR      92$
7889  036030  052777  000040  144116    91$:    BIS     #BIT5,@RP4CS2           ;CLEAR RP04 CONTROLLER
7890  036036  105737  001503            92$:    TSTB    @#MMON                  ;MGMT ON?
7891  036042  001345                            BNE     70$                     ;BRANCH IF YES
7892  036044  000742                            BR      71$
7893  036046  052763  000400  001676    20$:    BIS     #BIT8,RP3HSTAT(R3)      ;SET REPEAT FLAG IN HANDLER
7894  036054  016337  002026  000010            MOV     RP3HANA(R3),@#RESVEC    ;SETUP RESERVED INSTRUCTION VECTOR
7895  036062  000010                            CALLHANDLER
7896  036064  105763  001676            21$:    TSTB    RP3HSTAT(R3)            ;HANDLER FINISHED?
7897  036070  100375                            BPL     21$                     ;BRANCH IF NO
7898  036072  005763  001676                    TST     RP3HSTAT(R3)            ;ANY ERROR?
7899  036076  100714                            BMI     18$                     ;BRANCH IF YES
7900  036100  005701                            TST     R1                      ;DEVICE ERROR?
7901  036102  001002                            BNE     80$                     ;BRANCH IF YES
7902  036104  000167  177364                    JMP     32$+4                   ;GO CHECK DATA
7903  036110  032737  001000  177570    80$:    BIT     #BIT9,@#SWR             ;STILL LOOPING?
7904  036116  001353                            BNE     20$                     ;BRANCH IF YES
7905  036120  000721                            BR      100$+2                  ;CONTINUE TEST
7906                                    ;THE FOLLOWING CODE HANDLES DEVICE ERROR ON RELOCATION
7907  036122  005004                    15$:    CLR     R4                      ;SET INDEX
7908  036124  010601                            MOV     SP,R1
7909  036126  005764  001642            24$:    TST     RUNTRAK(R4)             ;SEARCH FOR DEVICE ERROR
7910  036132  100643                            BMI     16$                     ;BRANCH IF ERROR
7911  036134  062704  000002                    ADD     #2,R4                   ;INCREMENT INDEX
7912  036140  000772                            BR      24$                     ;CONTINUE SEARCH
7913                                    ;RELOCATE BY CPU-MEMORY MANAGEMENT ON
7914  036142  012022                    RELOCP: MOV     (R0)+,(R2)+             ;RELOCATE CODE
7915  036144  020302                            CMP     R3,R2                   ;DONE YET?
7916  036146  001375                            BNE     RELOCP                  ;BRANCH IF NO
7917  036150  012705  001700                    MOV     #1700,R5
7918  036154  004737  051744                    JSR     PC,@#CHKDAT             ;CHECK DATA
7919  036160  102007                            BVC     EXIT
7920  036162  010037  001276                    MOV     R0,@#STMP0
7921  036166  010237  001474                    MOV     R2,@#VADR
7922  036172  104006                            ERROR   6
7923  036174  000167  176000                    JMP     RETRY
7924  036200  105737  001503            EXIT:   TSTB    @#MMON                  ;MEM MGMT ON?
7925  036204  001002                            BNE     .+6                     ;BRANCH IF YES
7926  036206  000137  034426                    JMP     @#EXITRE
7927  036212  062737  000077  001520            ADD     #77,@#NEXPAR            ;SET VALUE FOR NEXT RELOCATION
7928  036220  013737  172346  172340            MOV     @#KIPAR3,@#KIPAR0
7929  036226  013737  172350  172342            MOV     @#KIPAR4,@#KIPAR1
```

```
7930  036234  013737  172352  172344          MOV     @#KIPAR5,@#KIPAR2
7931                                  ;*************************************************************************
7932                                  ;PROGRAM IS NOW EXECUTING IN KERNEL MODE RELOCATED TO ADDRESS AS SPEC-
7933                                  ;IFIED IN KIPAR0. FOR EX. IF KIPAR0=1600 THEN PROGRAM EXECUTING AT
7934                                  ;ADDRESS 160000+(PC)
7935  036242  013700  172340          MOV     @#KIPAR0,R0     ;GET PAR0
7936  036246  072027  177771          ASH     #-7,R0          ;GET BITS <14:7> IN LOW BYTE
7937  036252  110037  001203          MOVB    R0,@#$TSTNM+1   ;PUT IN DSPLAY REG HIGH BYTE
7938  036256  012706  001200          MOV     #KERSTK,SP      ;SET KERNEL STACK PTR
7939  036262  005037  177776          CLR     @#PSW
7940  036266  016746  175356          MOV     OLDPSW,-(SP)    ;RESTORE OLD PSW
7941  036272  012746  005406          MOV     #LOOP,-(SP)
7942  036276  105737  001502          TSTB    @#NEXEC         ;BRANCH IF TEST CODE TO
7943  036302  001402                  BEQ     1$              ;BE EXECUTED
7944  036304  012716  033630          MOV     #STMM,(SP)
7945  036310  000002          1$:     RTI                     ;RESTART PROGRAM AT LOOP
7946
7947                                  ;WHEN RELOCATION ABOVE 28K IS COMPLETE PROGRAM TRAPS TO ENDMEM.
7948  036312  022626          ENDMEM: CMP     (SP)+,(SP)+     ;POP STACK TWICE
7949  036314  005037  177572  ENDM:   CLR     @#SR0           ;DISABLE MEM MGMT
7950  036320  042737  000020  172516          BIC     #BIT4,@#MMR3    ;CLEAR 22 BIT MODE
7951
7952                                  ;*************************************************************************
7953                                  ;AT THIS TIME A 'SUB-PASS' HAS BEEN COMPLETED.
7954                                  ;PROGRAM NOW EXECUTING IN KERNEL MODE AT PC AS SHOWN (NO RELOCATION)
7955  036326  012737  000600  001520          MOV     #600,@#NEXPAR   ;RESET NEXT VALUE FOR PAR REGISTERS
7956  036334  005737  003244                  TST     @#PROT
7957  036340  001403                  BEQ     2$
7958  036342  012737  001600  001520          MOV     #1600,@#NEXPAR
7959  036350  105037  001503          2$:     CLRB    @#MMON          ;SET MEM MGMT ON IND = OFF
7960                                  ;;*********************************************************************
7961                                  .SBTTL  END OF SUB-PASS ROUTINE
7962                                  ;*      THIS ROUTINE SETSUP THE PSW AND MAINTENANCE REGISTERS
7963                                  ;*      FOR THE NEXT SUB-PASS. IT THEN STARTS THE PRINTER
7964                                  ;*      (IF NOT ON ACT11) FOR TYPING THE END OF SUB-PASS MESSAGE.
7965                                  ;;*********************************************************************
7966  036354                  END:
7967  036354  012737  053440  000004  END1:   MOV     #ERPRT,@#ERRVEC
7968  036362  005037  177776          CLR     @#PSW           ;CLEAR MODE BITS IN PSW
7969  036366  004767  013744          JSR     PC,CLRTBIT      ;GO CLEAR 'T' BIT IF SET
7970  036372  012706  001200          MOV     #KERSTK,SP      ;SET KERNEL STACK PTR
7971  036376  032777  000100  142640  BIT     #100,@$TPS      ;CHECK IF OUTPUT DEVICE IS BUSY
7972  036404  001374                  BNE     .-6             ;IS AVAILABLE
7973  036406  105237  001532  1$:     INCB    @#SUBPASS
7974  036412  113702  001532          MOVB    @#SUBPASS,R2
7975  036416  162702  000060          SUB     #60,R2
7976  036422  022702  000006          CMP     #6,R2                   ;END OF TEST?
7977  036426  001013                  BNE     2$              ;BRANCH IF NOT AT END
7978  036430  012737  000060  001532          MOV     #60,@#SUBPASS           ;INIT SUBPASS COUNT TO ASCII 0
7979  036436  005037  177750          CLR     @#MAINT                 ;CLEAR MAINTENANCE REG
7980  036442  005037  001604          CLR     @#$MAINT                ;CLEAR SOFTWARE VALUE
7981  036446  005046                  CLR     -(SP)
7982  036450  012746  036556          MOV     #$EOP,-(SP)
7983  036454  000002                  RTI
7984  036456  006302          2$:     ASL     R2
7985  036460  012737  001472  000014          MOV     #$RTRN,@#TBITVEC        ;SET 'T' TRAP VECTOR
```

```
 7986  036466  012737  001531  001270          MOV     #SUBPASS-1,@#$REG5
 7987  036474  106277  142544                  ASRB    @$TPS
 7988  036500  016246  053534                  MOV     PSWTAB(2),-(SP) ;PUSH NEXT PASS PSW ON STACK
 7989  036504  012746  005406                  MOV     #LOOP,-(SP)     ;RESART PROGRAM AT LOOP
 7990  036510  016237  053550  001604          MOV     MRGTAB(R2),@#$MAINT
 7991  036516  016237  053550  177750          MOV     MRGTAB(R2),@#MAINT
 7992  036524  105737  001504          3$:     TSTB    @#QV            ;QV PASS?
 7993  036530  001011                  BNE     RTI1            ;BRANCH IF YES
 7994  036532  122777  000200  142504          CMPB    #200,@$TPS      ;IS PRINTER READY?
 7995  036540  001371                  BNE     3$              ;BRANCH IF NO
 7996  036542  012737  054600  001270          MOV     #MSG20-1,@#$REG5
 7997  036550  106277  142470                  ASRB    @$TPS           ;TYPE END SUBPASS MESSAGE
 7998  036554  000002                  RTI1:   RTI                     ;RESTART PROGRAM AT LOOP WITH NEW PSW
 7999                                                                  ;(FROM TABLE BELOW)
 8000
 8001                                  ;;**********************************************************************
 8002
 8003                                  .SBTTL  END OF PASS ROUTINE
 8004
 8005                                  ;*INCREMENT THE PASS NUMBER ($PASS)
 8006                                  ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
 8007                                  ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
 8008                                  ;*IF THERES A MONITOR GO TO IT
 8009                                  ;*IF THERE ISN'T JUMP TO LOOP
 8010
 8011  036556                          $EOP:
 8012  036556  004737  046656                  JSR     PC,@#TYPTIME
 8013  036562  005067  142414                  CLR     $TSTNM          ;;ZERO THE TEST NUMBER
 8014  036566  005067  142524                  CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
 8015  036572  005267  142402                  INC     $PASS           ;;INCREMENT THE PASS NUMBER
 8016  036576  042767  100000  142374          BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
 8017  036604  005327                  DEC     (PC)+           ;;LOOP?
 8018  036606  000001          $EOPCT: .WORD   1
 8019  036610  003063                  BGT     $DOAGN          ;;YES
 8020  036612  012737                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
 8021  036614  000001          $ENDCT: .WORD   1
 8022  036616  036606                  $EOPCT
 8023  036620  104400  036626                  TYPE    ,65$            ;;TYPE ASCIZ STRING
 8024  036624  000407                  BR      64$             ;;GET OVER THE ASCIZ
 8025                                  ;;65$:  .ASCIZ  <12><15>/END PASS #/
 8026  036644                          64$:
 8027  036644  016746  142330                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
 8028                                                                  ;;TYPE PASS NUMBER
 8029  036650  104410                  TYPDS                           ;;GO TYPE--DECIMAL ASCII WITH SIGN
 8030  036652  104400  036660                  TYPE    ,67$            ;;TYPE ASCIZ STRING
 8031  036656  000421                  BR      66$             ;;GET OVER THE ASCIZ
 8032                                  ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
 8033  036722                          66$:
 8034  036722  016746  142266                  MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
 8035                                                                  ;;TOTAL NUMBER OF ERRORS
 8036  036726  104410                  TYPDS                           ;;GO TYPE--DECIMAL ASCII WITH SIGN
 8037  036730  104400  001327                  TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
 8038  036734  005067  142254                  CLR     $ERTTL          ;;CLEAR ERROR TOTAL
 8039  036740  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
 8040  036744  001405                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
 8041  036746  000005                  RESET                           ;;CLEAR THE WORLD
```

```
8042  036750  004710                  $ENDAD: JSR     PC,(R0)          ;;GO TO MONITOR
8043  036752  000240                          NOP                      ;;SAVE ROOM
8044  036754  000240                          NOP                      ;;FOR
8045  036756  000240                          NOP                      ;;ACT11
8046  036760                          $DOAGN:
8047  036760  000137  005406                  JMP     @#LOOP           ;;RETURN
8048  036764     377     377      000 $ENULL: .BYTE   -1,-1,0          ;;NULL CHARACTER STRING
8049           036770                          .EVEN
8050                                  ;;*****************************************************************
8051                                  .SBTTL  RP11/RP03 HANDLER
8052                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
8053                                  ;;*****************************************************************
8054  036770  104412                  RP3DRV: SAVREG
8055  036772  105037  001676                  CLRB    @#RP3HSTA                ;CLEAR DONE FLAG
8056  036776  032737  000400  001676          BIT     #BIT8,@#RP3HSTA          ;REPEAT FLAG SET?
8057  037004  001403                          BEQ     8$                       ;BRANCH IF NO
8058  037006  104414                          RESREG
8059  037010  000137  040642                  JMP     @#RP3RPT
8060  037014  013737  001556  001566  8$:     MOV     @#RNTBINX,@#RP311        ;SAVE RUN TABLE INDEX
8061  037022  032737  000020  177570          BIT     #SW4,@#SWR               ;INHIBIT RND DSK ADR?
8062  037030  001403                          BEQ     1$                       ;BRANCH IF NO
8063  037032  005000                          CLR     R0
8064  037034  005001                          CLR     R1
8065  037036  000410                          BR      4$
8066  037040  004737  050602          1$:     JSR     PC,@#$RAND               ;GO GET RANDOM NUMBER
8067  037044  013700  001524                  MOV     @#$HINUM,R0              ;GET HI NUMBER
8068  037050  013701  001522                  MOV     @#$LONUM,R1              ;GET LO NUMBER
8069  037054  073027  177771                  ASHC    #-7,R0                   ;ADJUST TO FORM CYL ADR
8070  037060  042700  177000          4$:     BIC     #177000,R0               ;GET RID OF UNUSED BITS
8071  037064  022700  000624                  CMP     #624,R0                  ;LEGAL CYL?
8072  037070  100003                          BPL     5$                       ;BRANCH IF YES
8073  037072  062700  000624                  ADD     #624,R0                  ;MAKE IT LEGAL
8074  037076  000770                          BR      4$
8075  037100  013702  001566          5$:     MOV     @#RP311,R2               ;GET RUN TABLE INDEX
8076  037104  016203  001626                  MOV     RUNTBL(R2),R3            ;GET DEVICE ID
8077  037110  042703  000777                  BIC     #777,R3                  ;ID ONLY
8078  037114  050300                          BIS     R3,R0                    ;COMBINE WITH CYL ADR
8079  037116  010062  001626                  MOV     R0,RUNTBL(R2)            ;PUT BACK IN TABLE
8080  037122  072127  177775                  ASH     #-3,R1                   ;GEN TRK-SECT ADR
8081  037126  010103                          MOV     R1,R3                    ;SAVE
8082  037130  042701  160377          6$:     BIC     #160377,R1               ;GET RID OF ALL BUT TRK
8083  037134  022701  011400                  CMP     #11400,R1                ;LEGAL TRAK?
8084  037140  100003                          BPL     2$                       ;BRANCH IF YES
8085  037142  062701  011400                  ADD     #11400,R1                ;MAKE IT LEGAL
8086  037146  000770                          BR      6$
8087  037150  042703  177760          2$:     BIC     #177760,R3               ;GET SECTOR ADR
8088  037154  022703  000011                  CMP     #11,R3                   ;IS IT LEGAL?
8089  037160  100003                          BPL     3$                       ;BRANCH IF YES
8090  037162  062703  000011                  ADD     #11,R3                   ;MAKE IT LEGAL
8091  037166  000770                          BR      2$
8092  037170  050301                  3$:     BIS     R3,R1                    ;COMBINE TRK-SECT
8093  037172  010162  001642                  MOV     R1,RUNTRAK(R2)           ;PUT IN TABLE
8094  037176  010037  002044                  MOV     R0,@#RP3HDC              ;SAVE DESIRED CYL
8095  037202  010137  002112                  MOV     R1,@#RP3DA               ;SAVE DSK ADR
8096  037206  112737  177775  002072          MOVB    #-3,@#RP3TRY             ;INIT TRY COUNT
8097  037214  032737  000040  172516          BIT     #BIT5,@#MMR3             ;MAP ON?
```

```
8098   037222  001405                          BEQ     7$                      ;BRANCH IF NO
8099   037224  005046                          CLR     -(SP)                   ;PUT DEVICE ID ON STACK
8100   037226  013746  001732                 MOV     @#RP3OLD,-(SP)          ;PUT ADR OF BUS ADR ON STK
8101   037232  004737  052052                 JSR     PC,@#GETMAP             ;GET MAP REGISTER
8102   037236  012737  000103  001564  7$:    MOV     #103,@#RP310            ;GET FUNCTION
8103   037244  013700  001734                 MOV     @#RP3OLD+2,R0           ;GET BAE BITS
8104   037250  072027  000004                 ASH     #4,R0                   ;SHIFT TO BITS 4 & 5
8105   037254  050037  001564                 BIS     R0,@#RP310              ;COMBINE WITH FUNCTION
8106   037260  010037  001734                 MOV     R0,@#RP3OLD+2
8107   037264  013700  002012                 MOV     @#RP3UNIT,R0
8108   037270  072027  000010                 ASH     #10,R0                  ;SHIFT UNIT NO TO RIGHT BITS
8109   037274  050037  001564                 BIS     R0,@#RP310              ;COMBINE WITH FUNC & BAE
8110   037300  010037  002012                 MOV     R0,@#RP3UNIT
8111   037304  104414                          RESREG
8112   037306  005777  142566          RP3WTRY:TST     @RP3DS                          ;IS DRIVE READY?
8113   037312  100375                          BPL     RP3WTRY                         ;BRANCH IF NO
8114   037314  053777  002012  142562         BIS     @#RP3UNIT,@RP3CS                ;SET UNIT BITS
8115   037322  004737  037354                 JSR     PC,@#LDRP3              ;LOAD RP3 REGISTERS
8116   037326  012777  040672  142562         MOV     #RP3SRV,@RP3VEC         ;SET VECTOR
8117   037334  005077  142560                 CLR     @RP3PSW
8118   037340  005037  002060                 CLR     @#RP3FUN                ;SET FUNCTION TO WRITE
8119   037344  013777  001564  142532         MOV     @#RP310,@RP3CS          ;LOAD FUNCT AND GO
8120   037352  000002                          RTI                             ;RETURN
8121   037354  013777  002042  142530  LDRP3: MOV     @#RP3HDA,@RP3DA         ;LOAD DSK ADR
8122   037362  013777  002044  142524         MOV     @#RP3HDC,@RP3DC         ;LOAD CYL ADR
8123   037370  013777  001716  142510         MOV     @#RP3HWC,@RP3WC         ;LOAD WORD COUNT
8124   037376  013777  001732  142504         MOV     @#RP3OLD,@RP3BA         ;LOAD BUS ADR
8125   037404  000207                          RTS     PC                      ;RETURN
8126
8127                                   ;;*****************************************************************
8128                                   .SBTTL  RK11/RK05 HANDLER
8129                                   ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
8130                                   ;;*****************************************************************
8131   037406  104412                  RKDRV:  SAVREG
8132   037410  105037  001700                  CLRB    @#RKHSTAT               ;CLEAR DONE FLAG IN HANDLER STAT
8133   037414  032737  000400  001700         BIT     #BIT8,@#RKHSTAT         ;REPEAT FLAG SET?
8134   037422  001403                          BEQ     5$                      ;BRANCH IF NO
8135   037424  104414                          RESREG
8136   037426  000137  041460                 JMP     @#RKRPT
8137   037432  013737  001556  001572  5$:    MOV     @#RNTBINX,@#RK11        ;SAVE RUN TABLE INDEX
8138   037440  105037  001700                  CLRB    @#RKHSTAT               ;CLEAR DONE FLAG IN HANDLER STAT
8139   037444  032737  000020  177570         BIT     #SW4,@#SWR              ;RANDOM DSK ADDRESS?
8140   037452  001403                          BEQ     6$                      ;BRANCH IF YES
8141   037454  005000                          CLR     R0                      ;CLEAR REGISTERS
8142   037456  005001                          CLR     R1
8143   037460  000404                          BR      7$                      ;FOR ADDRESS CHECKING
8144   037462  004737  050602          6$:    JSR     PC,@#$RAND              ;GET RANDOM NUMBER
8145   037466  013700  001524                 MOV     @#$HINUM,R0             ;GET HIGH NUMBER
8146   037472  072027  177775          7$:    ASH     #-3,R0                  ;ADJUST TO FORM
8147                                                                          ;CYLINDER ADDRESS
8148   037476  010001                          MOV     R0,R1                   ;SAVE IN R1
8149   037500  042701  160037          4$:    BIC     #160037,R1              ;GET RID OF SURF-SECT BITS
8150   037504  022701  014300                 CMP     #14300,R1               ;IS IT A LEGAL CYLINDER?
8151   037510  100003                          BPL     3$                      ;BRANCH IF YES
8152   037512  062701  014340                 ADD     #14340,R1               ;ADD MAXIMUM CYLINDER
8153   037516  000770                          BR      4$                      ;TRY AGAIN
```

```
8154  037520  072127  177773          3$:  ASH     #-5,R1                  ;ADJUST CYLINDER ADDRESS
8155  037524  013702  001572               MOV     @#RK11,R2               ;GET RUN TABLE INDEX
8156  037530  016203  001626               MOV     RUNTBL(R2),R3           ;GET RUN TABLE ENTRY
8157  037534  042703  000777               BIC     #777,R3                 ;SAVE ID AND UNIT NO.
8158  037540  050103                       BIS     R1,R3                   ;INSERT CYLINDER ADDR
8159  037542  010362  001626               MOV     R3,RUNTBL(R2)           ;ENTER CYLINDER ADR IN RUN TABLE
8160  037546  072027  177770               ASH     #-10,R0                 ;GENER SECTOR-SURF ADDRESS
8161  037552  042700  177740               BIC     #177740,R0              ;GET RID OF EXTRA BITS
8162  037556  010003                       MOV     R0,R3                   ;SAVE
8163  037560  042700  000020               BIC     #BIT4,R0                ;GET RID OF SURFACE BIT
8164  037564  022700  000012               CMP     #12,R0                  ;IS SECTOR ADDRESS LEGAL?
8165  037570  100004                       BPL     1$                      ;BRANCH IF YES
8166  037572  062700  000012               ADD     #12,R0                  ;MAKE IT LEGAL
8167  037576  042700  000020               BIC     #BIT4,R0                ;GET RID OF CARRY FROM ADD
8168  037602  042703  000017          1$:  BIC     #17,R3                  ;GET SURFACE ADDRESS
8169  037606  050300                       BIS     R3,R0                   ;GENER COMP SECT-SURF ADDRESS
8170  037610  010062  001642               MOV     R0,RUNTRAK(R2)          ;SAVE IN RUN TRAK TABLE
8171  037614  072127  000005               ASH     #5,R1                   ;ADJUST CYLINDER ADDRESS
8172  037620  050100                       BIS     R1,R0                   ;CONCATINATE TRK & SECT ADDR
8173  037622  013701  002014               MOV     @#RKUNIT,R1             ;GET UNIT NUMBER
8174  037626  072127  000015               ASH     #15,R1          ;ADJUST
8175  037632  050100                       BIS     R1,R0                   ;CONCATINATE UNIT,TRK,SURF,SECT
8176  037634  010037  002046               MOV     R0,@#RKHDA              ;SAVE
8177  037640  112737  177775  002073       MOVB    #-3,@#RKTRY             ;SET RETRY COUNT
8178  037646  032737  000040  172516       BIT     #BIT5,@#MMR3            ;MAP ON?
8179  037654  001406                       BEQ     2$                      ;BRANCH IF NO
8180  037656  012746  000001               MOV     #1,-(SP)                ;PUT DEVICE ID ON STACK
8181  037662  012746  001736               MOV     #RKOLD,-(SP)            ;PUT ADDRESS OF ADR ON STACK
8182  037666  004737  052052               JSR     PC,@#GETMAP             ;GET MAP REG
8183  037672  012767  000103  141670  2$:  MOV     #103,RK10               ;SET FUNCTION
8184  037700  013700  001740               MOV     @#RKOLD+2,R0            ;GET BA EXTENDED
8185  037704  072027  000004               ASH     #4,R0                   ;ADJUST
8186  037710  050037  001570               BIS     R0,@#RK10               ;PUT IN WITH FUNCTION
8187  037714  010037  001740               MOV     R0,@#RKOLD+2            ;SAVE IN MEMORY
8188  037720  104414                       RESREG
8189  037722  013777  002046  142204  RKWTRY: MOV  @#RKHDA,@RKDA           ;LOAD DISK ADDRESS
8190  037730  032777  000100  142164       BIT     #BIT6,@RKDS             ;UNIT READY?
8191  037736  001774                       BEQ     .-6                     ;BRANCH IF NO
8192  037740  013777  001720  142162       MOV     @#RKHWC,@RKWC           ;LOAD WORD COUNT
8193  037746  013777  001736  142156       MOV     @#RKOLD,@RKBA           ;LOAD BUS ADDRESS
8194  037754  012777  041510  142154       MOV     #RKSRV,@RKVEC           ;LOAD INTERRUPT VECTOR
8195  037762  005077  142152               CLR     @RKPSW
8196  037766  005037  002062               CLR     @#RKFUN                 ;SET FUNCTION TO WRITE
8197  037772  013777  001570  142126       MOV     @#RK10,@RKCS            ;LOAD FUNCTION AND GO
8198  040000  000006                       RTT                             ;RETURN
8199
8200                                  ;;*****************************************************************
8201                                  .SBTTL  RH70/RP04 HANDLER
8202                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
8203                                  ;;*****************************************************************
8204  040002  104412                 RP4DRV: SAVREG
8205  040004  105037  001706               CLRB    @#RP4HSTA               ;CLEAR DONE FLAG
8206  040010  032737  000400  001706       BIT     #BIT8,@#RP4HST          ;REPEAT FLAG SET?
8207  040016  001403                       BEQ     6$                      ;BRANCH IF NO
8208  040020  104414                       RESREG
8209  040022  000137  042336               JMP     @#RP4RPT
```

```
8210  040026  013737  001556  001574  6$:   MOV   @#RNTBINX,@#RP411      ;SAVE RUN TABLE INDEX
8211  040034  105037  001706              CLRB  @#RP4HSTA                ;CLEAR DONE FLAG
8212  040040  032737  000020  177570      BIT   #SW4,@#SWR              ;RANDOM DSK ADDRESS?
8213  040046  001403                      BEQ   1$                       ;BRANCH IF YES
8214  040050  005000                      CLR   R0
8215  040052  005001                      CLR   R1
8216  040054  000410                      BR    4$
8217  040056  004737  050602          1$: JSR   PC,@#$RAND               ;GET RANDOM NUMBER
8218  040062  013700  001524              MOV   @#$HINUM,R0              ;GET HI NUMBER
8219  040066  013701  001522              MOV   @#$LONUM,R1              ;GET LO NUMBER
8220  040072  073027  177771              ASHC  #-7,R0                   ;ADJUST TO FORM CYL. ADR.
8221  040076  042700  177000          4$: BIC   #177000,R0               ;GET RID OF UNUSED BITS
8222  040102  022700  000631              CMP   #631,R0                  ;LEGAL CYLINDER
8223  040106  100003                      BPL   5$                       ;BRANCH IF YES
8224  040110  062700  000631              ADD   #631,R0                  ;MAKE IT LEGAL
8225  040114  000770                      BR    4$
8226
8227  040116  013702  001574          5$: MOV   @#RP411,R2               ;GET RUN TABLE INDEX
8228  040122  016203  001626              MOV   RUNTBL(R2),R3            ;GET DEVICE ID
8229  040126  042703  000777              BIC   #777,R3                  ;SAVE ID ONLY
8230  040132  050003                      BIS   R0,R3                    ;COMBINE WITH CYL ADR
8231  040134  010362  001626              MOV   R3,RUNTBL(R2)            ;PUT IN RUN TABLE
8232  040140  072127  177775              ASH   #-3,R1                   ;GEN TRAK-SECT ADR
8233  040144  042701  160340              BIC   #160340,R1               ;GET RID OF UNUSED BITS
8234  040150  010103                      MOV   R1,R3                    ;SAVE
8235  040152  042701  000037              BIC   #37,R1                   ;GET RID OF SECT BITS
8236  040156  022701  011000              CMP   #11000,R1                ;LEGAL TRAK?
8237  040162  100004                      BPL   2$                       ;BRANCH IF YES
8238  040164  062701  011000              ADD   #11000,R1                ;MAKE IT LEGAL
8239  040170  042701  020000              BIC   #BIT13,R1                ;GET RID OF ADD CARRY
8240  040174  042703  177740          2$: BIC   #177740,R3               ;GET SECTOR ADR
8241  040200  022703  000025              CMP   #25,R3                   ;LEGAL SECTOR
8242  040204  100004                      BPL   3$                       ;BRANCH IF YES
8243  040206  062703  000025              ADD   #25,R3                   ;MAKE IT LEGAL
8244  040212  042703  000040              BIC   #BIT5,R3                 ;GET RID OF ADD CARRY
8245  040216  050301                  3$: BIS   R3,R1                    ;COMBINE TRAK-SECTOR
8246  040220  010162  001642              MOV   R1,RUNTRAK(R2)           ;PUT TRAK-SECT IN TABLE
8247  040224  010037  002054              MOV   R0,@#RP4HDC              ;SAVE CYLINDER ADR
8248  040230  010137  002052              MOV   R1,@#RP4HDA              ;SAVE TRAK-SECTOR ADR
8249  040234  112737  177775  002075      MOVB  #-3,@#RP4TRY             ;SET TRY COUNT
8250  040242  104414                      RESREG
8251  040244  004767  000026          RP4WTRY:JSR PC,LDRP4               ;LOAD RP4 REGISTERS
8252  040250  012777  042362  141720      MOV   #RP4SRV,@RP4VEC ;LOAD INTERRUPT VECTOR
8253  040256  005077  141716              CLR   @RP4PSW                  ;
8254  040262  005037  002066              CLR   @#RP4FUN                 ;SET FUNCTION TO WRITE
8255  040266  112777  000161  141646      MOVB  #161,@RP4CS1             ;LOAD FUNCTION AND GO
8256  040274  000002                      RTI                            ;RETURN
8257
8258  040276  013777  002022  141650  LDRP4: MOV   @#RP4UNIT,@RP4CS2     ;LOAD UNIT NUMBER
8259  040304  012777  010000  141662      MOV   #BIT12,@RP4OF            ;SET FORMAT TO 16 BIT
8260  040312  013777  002054  141644      MOV   @#RP4HDC,@RP4DC          ;LOAD CYLINDER ADR
8261  040320  013777  002052  141624      MOV   @#RP4HDA,@RP4DA          ;LOAD TRAK-SECTOR
8262  040326  013777  001726  141610      MOV   @#RP4HWC,@RP4WC          ;LOAD WORD COUNT
8263  040334  013777  001754  141606      MOV   @#RP4OLD+2,@RP4BAE       ;LOAD EXTENDED ADR BITS
8264  040342  013777  001752  141576      MOV   @#RP4OLD,@RP4BA          ;LOAD BUS ADR
8265  040350  000207                      RTS   PC                       ;RETURN
```

```
                                        ;;*****************************************************************
                                        .SBTTL  RH70/RS04 HANDLER
                                        ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF HANDLER
                                        ;;*****************************************************************
040352  104412                  RSDRV:  SAVREG
040354  105037  001710                  CLRB    @#RSHSTAT               ;CLEAR DONE FLAG
040360  032737  000400  001710          BIT     #BIT8,@#RSHSTAT         ;REPEAT FLAG SET?
040366  001403                        BEQ     3$                      ;BRANCH IF NO
040370  104414                        RESREG
040372  000137  043046                JMP     @#RSRPT
040376  013737  001556  001576  3$:     MOV     @#RNTBINX,@#RS11        ;SAVE RUN TABLE INDEX
040404  032737  000020  177570          BIT     #SW4,@#SWR              ;RANDOM DSK ADR?
040412  001403                        BEQ     1$                      ;BRANCH IF YES
040414  005000                        CLR     R0
040416  005001                        CLR     R1
040420  000407                        BR      4$
040422  004737  050602          1$:     JSR     PC,@#$RAND              ;GET RANDOM NUMBER
040426  013700  001524                MOV     @#$HINUM,R0
040432  072027  177774                ASH     #-4,R0
040436  010001                        MOV     R0,R1                   ;SAVE RANDOM NUMBER
040440  042700  170077          4$:     BIC     #170077,R0              ;GET TRACK ADR
040444  022700  007600                CMP     #7600,R0                ;IS IT LEGAL?
040450  100003                        BPL     5$                      ;BRANCH IF YES
040452  062700  007600                ADD     #7600,R0                ;MAKE IT LEGAL
040456  000770                        BR      4$
040460  013702  001576          5$:     MOV     @#RS11,R2               ;GET RUN TABLE INDEX
040464  072027  177772                ASH     #-6,R0                  ;ADJUST TRACK ADR
040470  110062  001626                MOVB    R0,RUNTBL(R2)           ;SAVE TRAK ADR IN RUN TBL
040474  042701  177700          6$:     BIC     #177700,R1              ;GET SECTOR ADR
040500  022701  000077                CMP     #77,R1                  ;IS IT LEGAL?
040504  100003                        BPL     2$                      ;BRANCH IF YES
040506  062701  000077                ADD     #77,R1                  ;MAKE IT LEGAL
040512  000770                        BR      6$
040514  010162  001642          2$:     MOV     R1,RUNTRAK(R2)          ;SAVE IN RUN TRAK TABLE
040520  072027  000006                ASH     #6,R0                   ;ADJUST TRACK ADDR
040524  050100                        BIS     R1,R0                   ;COMBINE SECTOR TRAK
040526  010037  002056                MOV     R0,@#RSHDA              ;SAVE AS DSK ADR
040532  112737  177775  002076        MOVB    #-3,@#RSTRY             ;SET TRY COUNT
040540  104414                        RESREG
040542  004737  040602          RSWTRY: JSR     PC,@#LDRS               ;GO LOAD REGISTERS
040546  012777  043072  141450        MOV     #RSSRV,@RSVEC           ;SET INTERRUPT VECTOR
040554  005077  141446                CLR     @RSPSW
040560  005037  002070                CLR     @#RSFUN                 ;SET FUNCTION TO WRITE
040564  105777  141430          1$:     TSTB    @RSDS                   ;IS DRIVE READY?
040570  001775                        BEQ     1$                      ;BRANCH IF NO
040572  112777  000161  141402        MOVB    #161,@RSCS1             ;LOAD FUNCTION AND GO
040600  000002                        RTI

040602  013777  002024  141404  LDRS:   MOV     @#RSUNIT,@RSCS2         ;LOAD UNIT NUMBER
040610  013777  002056  141374          MOV     @#RSHDA,@RSDA           ;LOAD DSK ADR
040616  013777  001730  141360          MOV     @#RSHWC,@RSWC           ;LOAD WORD COUNT
040624  013777  001760  141356          MOV     @#RSOLD+2,@RSBAE        ;LOAD EXTENDED ADDRESS
040632  013777  001756  141346          MOV     @#RSOLD,@RSBA           ;LOAD BUS ADDRESS
040640  000207                        RTS     PC                      ;RETURN
```

```
8322                                    ;;*****************************************************************
8323                                    .SBTTL  RP11/RP03 SERVICE ROUTINE
8324                                    ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
8325                                    ;;*****************************************************************
8326  040642  000005                RP3RPT: RESET
8327  040644  005337  002060               DEC     @#RP3FUN                ;RESTORE FUNCTION
8328  040650  022737  000001  002060       CMP     #1,@#RP3FUN             ;WHAT IS IT?
8329  040656  001472                       BEQ     RP31                    ;BRANCH IF WC
8330  040660  002402                       BLT     1$                      ;BRANCH IF WRITE
8331  040662  000137  037306               JMP     @#RP3WTRY               ;BRANCH TO READ
8332  040666  000167  000414          1$:  JMP     RP33
8333  040672  005237  002060       RP3SRV: INC     @#RP3FUN                ;INCREMENT FUNCTION
8334  040676  022737  000002  002060       CMP     #2,@#RP3FUN             ;WHAT IS IT?
8335  040704  001501                       BEQ     RP3WCK                  ;BRANCH TO WRITE CHECK
8336  040706  100002                       BPL     .+6
8337  040710  000137  041346               JMP     @#RP3READ
8338
8339                                    ;FUNCTION JUST EXECUTED WAS A WRITE
8340  040714  032737  000400  001676       BIT     #BIT8,@#RP3HSTAT        ;REPEAT FLAG SET?
8341  040722  001036                       BNE     RP3LOOP                 ;BRANCH IF YES
8342  040724  005777  141154               TST     @RP3CS                  ;ANY ERRORS?
8343  040730  100045                       BPL     RP31                    ;BRANCH IF NO
8344  040732  105737  002072               TSTB    @#RP3TRY                ;TRIED 3 TIMES?
8345  040736  001415                       BEQ     RP3ERR                  ;BRANCH IF YES
8346  040740  112777  000001  141136       MOVB    #BIT0,@RP3CS            ;CLEAR THE DRIVE
8347  040746  105777  141132               TSTB    @RP3CS                  ;CONTROLLER READY?
8348  040752  100375                       BPL     .-4                     ;BRANCH IF NO
8349  040754  105237  002072               INCB    @#RP3TRY                ;INCREMENT TRY COUNT
8350  040760  013746  177776               MOV     @#PSW,-(SP)             ;MAINTAIN SAME PSW
8351  040764  012746  037306               MOV     #RP3WTRY,-(SP)          ;SET RETRY ADDRESS
8352  040770  000002                       RTI                             ;RETURN
8353  040772  012737  100200  001676 RP3ERR: MOV     #100200,@#RP3HSTA       ;SET ERROR BIT IN HAND. STA
8354  041000  010046                       MOV     R0,-(SP)                ;SAVE R0
8355  041002  013700  001566               MOV     @#RP311,R0              ;GET RUNTABLE INDEX
8356  041006  052760  100000  001642       BIS     #BIT15,RUNTAK(R0)       ;SET ERROR BIT
8357  041014  012600                       MOV     (SP)+,R0                ;RESTORE R0
8358  041016  000002                       RTI                             ;RETURN
8359
8360  041020  012737  100200  001676 RP3LOOP:MOV     #100200,@#RP3HSTAT      ;SET DONE AND ERROR
8361  041026  005777  141052               TST     @RP3CS                  ;ANY ERRORS?
8362  041032  100403                       BMI     1$                      ;BRANCH IF YES
8363  041034  042737  100000  001676       BIC     #BIT15,@#RP3HSTAT       ;CLEAR ERROR BIT
8364  041042  000002                  1$:  RTI                             ;RETURN
8365                                    ;WRITE WAS OK- NOW DO A WRITE CHECK
8366  041044  112737  177775  002072 RP31:  MOVB    #-3,@#RP3TRY            ;INIT TRY COUNT
8367  041052  012737  000107  001564       MOV     #107,@#RP310            ;SET FUNCTION
8368  041060  053737  001734  001564       BIS     @#RP3OLD+2,@#RP310      ;SET BAE BITS
8369  041066  053737  002012  001564       BIS     @#RP3UNIT,@#RP310       ;SET UNIT BITS
8370  041074  004737  037354          RP32: JSR     PC,@#LDRP3              ;LOAD RP3 REGISTERS
8371  041100  013777  001564  140776       MOV     @#RP310,@RP3CS          ;LOAD FUNCTION AND GO
8372  041106  000002                       RTI                             ;RETURN
8373
8374                                    ;FUNCTION JUST EXECUTED WAS A WRITE CHECK
8375  041110  032737  000400  001676 RP3WCK: BIT     #BIT8,@#RP3HSTAT                ;REPEAT FLAG SET?
8376  041116  001340                       BNE     RP3LOOP                 ;BRANCH IF YES
8377  041120  005777  140760               TST     @RP3CS                  ;ANY ERRORS?
```

```
8378  041124  100031                          BPL     1$                      ;BRANCH IF NO
8379  041126  005737  001566                  TST     @#RP311                 ;FIRST 2K?
8380  041132  001422                          BEQ     4$                      ;BRANCH IF YES
8381  041134  105737  002072          5$:     TSTB    @#RP3TRY                ;TRIED 3 TIMES?
8382  041140  001714                          BEQ     RP3ERR                  ;BRANCH IF YES
8383  041142  005337  002060                  DEC     @#RP3FUN                ;RESTORE FUNCTION
8384  041146  112777  000001  140730          MOVB    #BIT0,@RP3CS            ;CLEAR THE DRIVE
8385  041154  105777  140724                  TSTB    @RP3CS                  ;CONTROLLER READY?
8386  041160  100375                          BPL     .-4                     ;BRANCH IF NO
8387  041162  105237  002072                  INCB    @#RP3TRY                ;INCREMENT TRY COUNT
8388  041166  013746  177776                  MOV     @#PSW,-(SP)
8389  041172  012746  041074                  MOV     #RP32,-(SP)
8390  041176  000002                          RTI                             ;GO TRY AGAIN
8391  041200  032777  000010  140674  4$:     BIT     #BIT3,@RP3ER            ;WRITE CHECK ERROR?
8392  041206  001752                          BEQ     5$                      ;BRANCH IF NO
8393
8394                                  ;WRITE CHECK OK- NOW DO A READ
8395  041210  112737  177775  002072  1$:     MOVB    #-3,@#RP3TRY            ;RESTORE TRY COUNT
8396  041216  032737  000040  172516          BIT     #BIT5,@#MMR3            ;MAP ON?
8397  041224  001407                          BEQ     2$                      ;BRANCH IF NO
8398  041226  005046                          CLR     -(SP)                   ;PUT DEVICE ID ON STACK
8399  041230  004737  052316                  JSR     PC,@#GIVEMAP            ;RETURN MAP REGISTER
8400  041234  012746  001762                  MOV     #RP3NWL,-(SP)           ;PUT ADR OF BUS ADR ON STK
8401  041240  004737  052052                  JSR     PC,@#GETMAP             ;GET MAP REGISTERS
8402  041244  010046                  2$:     MOV     R0,-(SP)                ;SAVE R0
8403  041246  013700  001764                  MOV     @#RP3NWH,R0             ;GET BAE BITS
8404  041252  072027  000004                  ASH     #4,R0                   ;ADJUST
8405  041256  010037  001764                  MOV     R0,@#RP3NWH             ;SAVE
8406  041262  012600                          MOV     (SP)+,R0                ;RESTORE R0
8407  041264  012737  000105  001564          MOV     #105,@#RP310            ;SET FUNCTION
8408  041272  053737  001764  001564          BIS     @#RP3NWH,@#RP310        ;SET BAE BITS
8409  041300  053737  002012  001564          BIS     @#RP3UNIT,@#RP310       ;SET UNIT NUMBER
8410  041306  013777  002042  140576  RP33:   MOV     @#RP3HDA,@RP3DA         ;LOAD DSK ADR
8411  041314  013777  002044  140572          MOV     @#RP3HDC,@RP3DC         ;LOAD CYL
8412  041322  013777  001716  140556          MOV     @#RP3HWC,@RP3WC         ;LOAD WORD COUNT
8413  041330  013777  001762  140552          MOV     @#RP3NWL,@RP3BA         ;LOAD BUS ADR
8414  041336  013777  001564  140540          MOV     @#RP310,@RP3CS          ;LOAD FUNCTION AND GO
8415  041344  000002                          RTI                             ;RETURN
8416
8417                                  ;FUNCTION JUST EXECUTED WAS A READ
8418  041346  032737  000400  001676  RP3READ: BIT     #BIT8,@#RP3HSTAT                ;REPEAT FLAG SET?
8419  041354  001221                          BNE     RP3LOOP                 ;BRANCH IF YES
8420  041356  005777  140522                  TST     @RP3CS                  ;ANY ERRORS?
8421  041362  100022                          BPL     1$                      ;BRANCH IF NO
8422  041364  105737  002072                  TSTB    @#RP3TRY                ;TRIED 3 TIMES?
8423  041370  001600                          BEQ     RP3ERR                  ;BRANCH IF YES
8424  041372  005337  002060                  DEC     @#RP3FUN                ;RESTORE FUNCTION
8425  041376  112777  000001  140500          MOVB    #BIT0,@RP3CS            ;CLEAR THE DRIVE
8426  041404  105777  140474                  TSTB    @RP3CS                  ;CONTROLLER READY?
8427  041410  100375                          BPL     .-4                     ;BRANCH OF NO
8428  041412  105237  002072                  INCB    @#RP3TRY                ;INCREMENT TRY COUNT
8429  041416  013746  177776                  MOV     @#PSW,-(SP)
8430  041422  012746  041306                  MOV     #RP33,-(SP)
8431  041426  000002                          RTI                             ;GO TRY AGAIN
8432  041430  032737  000040  172516  1$:     BIT     #BIT5,@#MMR3            ;MAP ON?
8433  041436  001404                          BEQ     2$                      ;BRANCH IF NO
```

```
8434  041440  005046                        CLR     -(SP)                   ;PUT DEVICE ID IN STK
8435  041442  004737  052316                JSR     PC,@#GIVEMAP            ;RETURN MAP REGISTERS
8436  041446  005726                        TST     (SP)+                   ;RESTORE STACK
8437  041450  112737  000200  001676  2$:   MOVB    #200,@#RP3HSTA          ;SET DONE FLAG
8438  041456  000002                        RTI                             ;RETURN
8439                                  ;;*****************************************************************
8440                                  .SBTTL  RK11/RK05 SERVICE ROUTINE
8441                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
8442                                  ;;*****************************************************************
8443  041460  000005                RKRPT:  RESET
8444  041462  005337  002062                DEC     @#RKFUN                 ;RESTORE FUNCTION
8445  041466  022737  000001  002062        CMP     #1,@#RKFUN              ;WHAT IS IT?
8446  041474  001475                        BEQ     RK1                     ;BRANCH IF WC
8447  041476  002402                        BLT     1$                      ;BRANCH IF WRITE
8448  041500  000137  037722                JMP     @#RKWTRY                ;IT WAS A WRITE
8449  041504  000137  042142        1$:     JMP     @#RK3
8450  041510  062737  000001  002062  RKSRV: ADD     #1,@#RKFUN              ;FIND OUT WHAT FUNCTION
8451                                                                        ;WAS EXECUTED
8452  041516  022737  000002  002062        CMP     #2,@#RKFUN              ;WAS IT A WRITE CHECK?
8453  041524  001507                        BEQ     RKWRCK                  ;BRANCH IF YES
8454  041526  100002                        BPL     .+6                     ;BRANCH IF IT WAS A WRITE
8455  041530  000137  042174                JMP     @#RKREAD
8456
8457                                  ;FUNCTION JUST EXECUTED WAS A WRITE. ANY ERRORS?
8458  041534  032737  000400  001700        BIT     #BIT8,@#RKHSTAT         ;REPEAT FLAG SET?
8459  041542  001040                        BNE     RKLOOP                  ;BRANCH IF YES
8460  041544  005777  140356                TST     @RKCS                   ;ANY ERRORS?
8461  041550  100047                        BPL     RK1                     ;BRANCH IF NO
8462  041552  105737  002073                TSTB    @#RKTRY                 ;TRYED 3 TIMES?
8463  041556  001417                        BEQ     RKERR                   ;BRANCH IF YES
8464  041560  012777  000001  140340        MOV     #1,@RKCS                ;CLEAR THE ERROR
8465  041566  004737  042320                JSR     PC,@#TIMER              ;WAIT A LITTLE
8466  041572  105777  140330                TSTB    @RKCS                   ;WAIT FOR CONT CLR TO FINISH
8467  041576  100375                        BPL     .-4
8468  041600  105237  002073                INCB    @#RKTRY                 ;INCREMENT TRY COUNT
8469  041604  013746  177776                MOV     @#PSW,-(SP)
8470  041610  012746  037722                MOV     #RKWTRY,-(SP)
8471  041614  000002                        RTI
8472  041616  012737  100200  001700  RKERR: MOV     #100200,@#RKHSTAT       ;SET ERROR & DONE FLAG
8473  041624  010046                        MOV     R0,-(SP)                ;SAVE R0
8474  041626  013700  001572                MOV     @#RK11,R0               ;GET SAVED RUN TABLE INDEX
8475  041632  052760  100000  001642        BIS     #BIT15,RUNTRAK(R0)      ;SET ERROR BIT IN RUN TABLE
8476  041640  012600                        MOV     (SP)+,R0                ;RESTORE R0
8477  041642  000002                        RTI                             ;RETURN
8478
8479  041644  012737  100200  001700  RKLOOP:MOV     #100200,@#RKHSTAT       ;SET DONE AND ERROR BITS
8480  041652  005777  140250                TST     @RKCS                   ;ANY ERRORS?
8481  041656  100403                        BMI     1$                      ;BRANCH IF YES
8482  041660  042737  100000  001700        BIC     #BIT15,@#RKHSTAT        ;CLEAR ERROR BIT
8483  041666  000002                  1$:   RTI                             ;RETURN
8484                                  ;WRITE WAS OK, NOW DO A WRITE CHECK
8485  041670  112737  177775  002073  RK1:   MOVB    #-3,@#RKTRY             ;RESTORE TRY COUNT
8486  041676  012767  000507  137664        MOV     #507,RK10               ;SET FUNCTION TO WRITE
8487  041704  053767  001740  137656        BIS     @#RKOLD+2,RK10          ;SET BA EXT BITS
8488  041712  013777  002046  140214  RK2:   MOV     @#RKHDA,@RKDA           ;LOAD DISK ADDRESS
8489  041720  013777  001720  140202        MOV     @#RKHWC,@RKWC           ;LOAD WORD COUNT
```

```
 8490  041726  013777  001736  140176          MOV     @#RKOLD,@RKBA           ;LOAD BUS ADDRESS
 8491  041734  016777  137630  140164          MOV     RK10,@RKCS              ;START FUNCTION
 8492  041742  000002                          RTI                             ;RETURN
 8493
 8494                                  ;FUNCTION JUST EXECUTED WAS A WRITE CHECK. ANY ERRORS?
 8495  041744  032737  000400  001700  RKWRCK:BIT     #BIT8,@#RKHSTAT                 ;REPEAT FLAG SET?
 8496  041752  001334                          BNE     RKLOOP                  ;BRANCH IF YES
 8497  041754  005777  140146                  TST     @RKCS                   ;ANY ERRORS?
 8498  041760  100033                          BPL     1$                      ;BRANCH IF NO
 8499  041762  005737  001572                  TST     @#RK11                  ;FIRST 2K?
 8500  041766  001424                          BEQ     4$                      ;BRANCH IF YES
 8501  041770  105737  002073          5$:     TSTB    @#RKTRY                 ;TRYED 3 TIMES?
 8502  041774  001710                          BEQ     RKERR                   ;BRANCH IF YES
 8503  041776  005337  002062                  DEC     @#RKFUN                 ;SET FUNCTION BACK TO WC
 8504  042002  012777  000001  140116          MOV     #1,@RKCS                ;CLEAR THE ERROR
 8505  042010  004737  042320                  JSR     PC,@#TIMER              ;WAIT A LITTLE
 8506  042014  105777  140106                  TSTB    @RKCS                   ;WAIT FOR CLR TO FINISH
 8507  042020  100375                          BPL     .-4
 8508  042022  105237  002073                  INCB    @#RKTRY                 ;INCREMENT TRY COUNT
 8509  042026  013746  177776                  MOV     @#PSW,-(SP)
 8510  042032  012746  041712                  MOV     #RK2,-(SP)
 8511  042036  000002                          RTI
 8512  042040  032777  040000  140060  4$:     BIT     #BIT14,@RKCS            ;HARD ERROR?
 8513  042046  001350                          BNE     5$                      ;BRANCH IF YES
 8514
 8515                                  ;WRITE CHECK WAS OK, NOW DO A READ.
 8516  042050  112737  177775  002073  1$:     MOVB    #-3,@#RKTRY             ;RESTORE TRY COUNT
 8517  042056  032737  000040  172516          BIT     #BIT5,@#MMR3            ;MAP ON?
 8518  042064  001410                          BEQ     2$                      ;BRANCH IF NO
 8519  042066  012746  000001                  MOV     #1,-(SP)                ;PUT DEVICE ID ON STACK
 8520  042072  004767  010220                  JSR     PC,GIVEMAP              ;RELINQUISH MAP REG
 8521  042076  012746  001766                  MOV     #RKNEWL,-(SP)           ;PUT ADR OF BADR ON STACK
 8522  042102  004737  052052                  JSR     PC,@#GETMAP             ;GET MAPREGISTER
 8523  042106  010046                  2$:     MOV     R0,-(SP)                ;SAVE R0
 8524  042110  013700  001770                  MOV     @#RKNEWH,R0             ;GET BA EXT
 8525  042114  072027  000004                  ASH     #4,R0                   ;ADJUST
 8526  042120  010037  001770                  MOV     R0,@#RKNEWH             ;SAVE
 8527  042124  012600                          MOV     (SP)+,R0                ;RESTORE R0
 8528  042126  012767  000105  137434          MOV     #105,RK10               ;SET FUNCTION
 8529  042134  053767  001770  137426          BIS     @#RKNEWH,RK10           ;SET BA EXT BITS IN FUNCTION
 8530  042142  013777  002046  137764  RK3:    MOV     @#RKHDA,@RKDA           ;LOAD DISK ADDRESS
 8531  042150  013777  001720  137752          MOV     @#RKHWC,@RKWC           ;LOAD WORD COUNT
 8532  042156  013777  001766  137746          MOV     @#RKNEWL,@RKBA          ;LOAD BUS ADDRESS
 8533  042164  016777  137400  137734          MOV     RK10,@RKCS              ;LOAD FUNCTION AND GO
 8534  042172  000002                          RTI                             ;RETURN
 8535
 8536                                  ;FUNCTION JUST EXECUTED WAS A READ. ANY ERRORS?
 8537  042174  032737  000400  001700  RKREAD: BIT     #BIT8,@#RKHSTAT                 ;REPEAT FLAG SET?
 8538  042202  001220                          BNE     RKLOOP                  ;BRANCH IF YES
 8539  042204  005777  137716                  TST     @RKCS                   ;ANY ERRORS?
 8540  042210  100026                          BPL     1$                      ;BRANCH IF NO
 8541  042212  105737  002073                  TSTB    @#RKTRY                 ;TRYED 3 TIMES?
 8542  042216  001002                          BNE     3$                      ;BRANCH IF NO
 8543  042220  000167  177372                  JMP     RKERR
 8544  042224  005337  002062          3$:     DEC     @#RKFUN                 ;SET FUNCTION BACK TO READ
 8545  042230  012777  000001  137670          MOV     #1,@RKCS                ;CLEAR THE ERROR
```

```
8546  042236  004737  042320                  JSR     PC,@#TIMER              ;WAIT A LITTLE
8547  042242  105777  137660                  TSTB    @RKCS                   ;WAIT FOR CLR TO FINISH
8548  042246  100375                          BPL     .-4
8549  042250  105237  002073                  INCB    @#RKTRY                 ;INCREMENT TRY COUNT
8550  042254  013746  177776                  MOV     @#PSW,-(SP)
8551  042260  012746  042142                  MOV     #RK3,-(SP)
8552  042264  000002                          RTI
8553  042266  032737  000040  172516  1$:     BIT     #BIT5,@#MMR3            ;MAP ON?
8554  042274  001405                          BEQ     2$                      ;BRANCH IF NO
8555  042276  012746  000001                  MOV     #1,-(SP)                ;PUT RK ID ON STACK
8556  042302  004737  052316                  JSR     PC,@#GIVEMAP            ;RELINQUISH MAP REGSTER
8557  042306  005726                          TST     (SP)+                   ;POP THE STACK
8558  042310  112737  000200  001700  2$:     MOVB    #200,@#RKHSTAT          ;SET DON E FLAG
8559  042316  000002                          RTI                             ;RETURN
8560  042320  005067  000010          TIMER:  CLR     1$
8561  042324  105267  000004          2$:     INCB    1$
8562  042330  001375                          BNE     2$
8563  042332  000207                          RTS     PC
8564  042334  000000                  1$:     .WORD
8565
8566                                  ;;*****************************************************************
8567                                  .SBTTL  RH70/RP04 SERVICE ROUTINE
8568                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
8569                                  ;;*****************************************************************
8570  042336  000005                  RP4RPT: RESET
8571  042340  005337  002066                  DEC     @#RP4FUN                ;RESTORE FUNCTION
8572  042344  022737  000001  002066          CMP     #1,@#RP4FUN             ;WHAT IS IT?
8573  042352  001501                          BEQ     RP41                    ;BRANCH IF WC
8574  042354  002560                          BLT     RP43                    ;BRANCH IF READ
8575  042356  000137  040244                  JMP     @#RP4WTRY               ;GO TO WRITE
8576  042362  005237  002066          RP4SRV: INC     @#RP4FUN                ;FIND OUT WHAT FUNCTION
8577  042366  022737  000002  002066          CMP     #2,@#RP4FUN             ;WAS JUST EXECUTED
8578  042374  001504                          BEQ     RP4WCK
8579  042376  100566                          BMI     RP4READ
8580
8581                                  ;WRITE FUNCTION WAS JUST EXECUTED.
8582  042400  032737  000400  001706          BIT     #BIT8,@#RP4HSTAT        ;REPEAT FLAG SET?
8583  042406  001050                          BNE     RP4LOOP                 ;BRANCH IF YES
8584  042410  032777  040000  137542          BIT     #BIT14,@RP4DS           ;ANY ERRORS
8585  042416  001457                          BEQ     RP41                    ;BRANCH IF NO
8586  042420  105737  002075                  TSTB    @#RP4TRY                ;TRIED 3 TIMES?
8587  042424  001426                          BEQ     RP4ERR                  ;BRANCH IF YES
8588  042426  052777  000040  137520          BIS     #BIT5,@RP4CS2           ;CLEAR ALL ERRORS
8589  042434  004737  040276                  JSR     PC,@#LDRP4              ;RELOAD THE UNIT NO
8590  042440  105237  002075                  INCB    @#RP4TRY                ;INCREMENT TRY COUNT
8591  042444  013746  177776                  MOV     @#PSW,-(SP)             ;SETUP THE STACK TO
8592  042450  012746  040244                  MOV     #RP4WTRY,-(SP)          ;TRY WRITE AGAIN
8593  042454  032737  000400  001706          BIT     #BIT8,@#RP4HSTAT        ;REPEAT FLAG SET?
8594  042462  001006                          BNE     2$                      ;BRANCH IF YES
8595  042464  012777  000007  137450          MOV     #7,@RP4CS1              ;RECALIBRATE
8596  042472  105777  137444          1$:     TSTB    @RP4CS1                 ;DRIVE READY?
8597  042476  100375                          BPL     1$                      ;BRANCH IF NO
8598  042500  000002                  2$:     RTI
8599  042502  012737  100200  001706  RP4ERR: MOV     #100200,@#RP4HSTA       ;SET ERROR & DONE BIT
8600  042510  010046                          MOV     R0,-(SP)                ;SAVE R0
8601  042512  013700  001574                  MOV     @#RP411,R0              ;GET RUN TABLE INDEX
```

```
 8602  042516  052760  100000  001642          BIS     #BIT15,RUNTRAK(R0)      ;SET ERROR BIT
 8603  042524  012600                          MOV     (SP)+,R0                ;RESTORE R0
 8604  042526  000002                          RTI                             ;RETURN
 8605
 8606  042530  012737  100200  001706  RP4LOOP:MOV     #100200,@#RP4HSTAT      ;SET DONE AND ERROR BITS
 8607  042536  032777  040000  137414          BIT     #BIT14,@RP4DS           ;ANY ERRORS?
 8608  042544  001003                          BNE     1$                              ;BRANCH IF YES
 8609  042546  042737  100000  001706          BIC     #BIT15,@#RP4HSTAT       ;CLEAR ERROR BIT
 8610  042554  000002                  1$:     RTI                             ;RETURN
 8611                                  ;WRITE OK...NOW DO A WRITE CHECK.
 8612  042556  112737  177775  002075  RP41:   MOVB    #-3,@#RP4TRY            ;INITIALIZE TRY COUNT
 8613  042564  105777  137370          RP42:   TSTB    @RP4DS                  ;IS DRIVE READY?
 8614  042570  001775                          BEQ     RP42                    ;BRANCH IF NO
 8615  042572  004737  040276                  JSR     PC,@#LDRP4
 8616  042576  112777  000151  137336          MOVB    #151,@RP4CS1            ;LOAD FUNCTION AND GO
 8617  042604  000002                          RTI
 8618
 8619                                  ;FUNCTION JUST EXECUTED WAS A WRITE CHECK
 8620  042606  032737  000400  001706  RP4WCK: BIT     #BIT8,@#RP4HSTAT                ;REPEAT FLAG SET?
 8621  042614  001345                          BNE     RP4LOOP                 ;BRANCH IF YES
 8622  042616  032777  040000  137334          BIT     #BIT14,@RP4DS           ;ANY ERRORS?
 8623  042624  001421                          BEQ     1$                      ;BRANCH IF NO
 8624  042626  105737  002075          3$:     TSTB    @#RP4TRY                ;TRIED 3 TIMES?
 8625  042632  001723                          BEQ     RP4ERR                  ;BRANCH IF YES
 8626  042634  005337  002062                  DEC     @#RKFUN                 ;SET FUNCTION TO WC
 8627  042640  052777  000040  137306          BIS     #BIT5,@RP4CS2           ;CLEAR ALL ERRORS
 8628  042646  004737  040276                  JSR     PC,@#LDRP4              ;RELOAD THE UNIT NO
 8629  042652  105237  002075                  INCB    @#RP4TRY                ;INCREMENT TRY COUNT
 8630  042656  013746  177776                  MOV     @#PSW,-(SP)
 8631  042662  012746  042564                  MOV     #RP42,-(SP)
 8632  042666  000002                          RTI                             ;TRY AGAIN
 8633  042670  032777  040000  137256  1$:     BIT     #BIT14,@RP4CS2          ;WRITE CHECK ERROR?
 8634  042676  001404                          BEQ     2$                      ;BRANCH IF NO
 8635  042700  005737  001574                  TST     @#RP411                 ;FIRST 2K?
 8636  042704  001401                          BEQ     2$
 8637  042706  000747                          BR      3$
 8638
 8639                                  ;WRITE CHECK WAS OK...NOW DO A READ.
 8640  042710  112737  177775  002075  2$:     MOVB    #-3,@#RP4TRY            ;INITIALIZE TRY COUNT
 8641  042716  105777  137236          RP43:   TSTB    @RP4DS                  ;IS DRIVE READY?
 8642  042722  001775                          BEQ     RP43                    ;BRANCH IF NO
 8643  042724  004737  040276                  JSR     PC,@#LDRP4              ;LOAD REGISTERS
 8644  042730  013777  002004  137212          MOV     @#RP4NWH,@RP4BAE        ;LOAD EXTENDED ADR BITS
 8645  042736  013777  002002  137202          MOV     @#RP4NWL,@RP4BA         ;LOAD BUS ADR
 8646  042744  112777  000171  137170          MOVB    #171,@RP4CS1            ;LOAD FUNCTION AND GO
 8647  042752  000002                          RTI                             ;RETURN
 8648
 8649                                  ;FUNCTION JUST EXECUTED WAS A READ.
 8650  042754  032737  000400  001706  RP4READ:BIT     #BIT8,@#RP4HSTAT        ;REPEAT FLAG SET?
 8651  042762  001262                          BNE     RP4LOOP                 ;BRANCH IF YES
 8652  042764  032777  040000  137166          BIT     #BIT14,@RP4DS           ;ANY ERRORS?
 8653  042772  001421                          BEQ     1$                      ;BRANCH IF NO
 8654  042774  105737  002075                  TSTB    @#RP4TRY                ;TRIED 3 TIMES?
 8655  043000  001640                          BEQ     RP4ERR                  ;BRANCH IF YES
 8656  043002  005337  002066                  DEC     @#RP4FUN                ;SET FUNCTION TO A READ
 8657  043006  052777  000040  137140          BIS     #BIT5,@RP4CS2           ;CLEAR ALL ERRORS
```

```
8658  043014  004737  040276                  JSR     PC,@#LDRP4              ;RELOAD THE UNIT NO
8659  043020  105237  002075                  INCB    @#RP4TRY                ;INCREMENT TRY COUNT
8660  043024  013746  177776                  MOV     @#PSW,-(SP)
8661  043030  012746  042716                  MOV     #RP43,-(SP)
8662  043034  000002                          RTI                             ;TRY AGAIN
8663  043036  112737  000200  001706  1$:     MOVB    #200,@#RP4HSTA          ;SET DONE FLAG
8664  043044  000002                          RTI                             ;RETURN
8665                                  ;;*****************************************************************
8666                                  .SBTTL  RH70/RS04 SERVICE ROUTINE
8667                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
8668                                  ;;*****************************************************************
8669  043046  000005                  RSRPT:  RESET
8670  043050  005337  002070                  DEC     @#RSFUN                 ;RESTORE FUNCTION
8671  043054  022737  000001  002070          CMP     #1,@#RSFUN              ;WHAT IS IT?
8672  043062  001465                          BEQ     RS41                    ;BRANCH IF WC
8673  043064  002542                          BLT     RS43                    ;BRANCH IF WRITE
8674  043066  000137  040542                  JMP     @#RSWTRY
8675  043072  005237  002070          RSSRV:  INC     @#RSFUN                 ;FIND OUT WHAT FUNCTION
8676  043076  022737  000002  002070          CMP     #2,@#RSFUN              ;WAS JUST EXECUTED
8677  043104  001470                          BEQ     RSWCK
8678  043106  100550                          BMI     RSREAD
8679
8680                                  ;WRITE FUNCTION WAS JUST EXECUTED
8681  043110  032737  000400  001710          BIT     #BIT8,@#RSHSTAT ;REPEAT FLAG SET?
8682  043116  001034                          BNE     RSLOOP                  ;BRANCH IF YES
8683  043120  032777  040000  137072          BIT     #BIT14,@RSDS            ;ANY ERRORS?
8684  043126  001443                          BEQ     RS41                    ;BRANCH IF NO
8685  043130  105737  002076                  TSTB    @#RSTRY                 ;TRIED 3 TIMES?
8686  043134  001412                          BEQ     RSERR                   ;BRANCH IF YES
8687  043136  052777  000040  137050          BIS     #BIT5,@RSCS2            ;CLEAR ALL ERRORS
8688  043144  105237  002076                  INCB    @#RSTRY                 ;INCREMENT TRY COUNT
8689  043150  013746  177776                  MOV     @#PSW,-(SP)             ;SETUP THE STACK TO
8690  043154  012746  040542                  MOV     #RSWTRY,-(SP)           ;TRY THE WRITE AGAIN
8691  043160  000002                          RTI
8692  043162  012737  100200  001710  RSERR:  MOV     #100200,@#RSHSTAT       ;SET ERROR AND DONE BIT
8693  043170  010046                          MOV     R0,-(SP)                ;SAVE R0
8694  043172  013700  001576                  MOV     @#RS11,R0               ;GET RUN TBL INDEX
8695  043176  052760  100000  001642          BIS     #BIT15,RUNTRAK(R0)      ;SET ERROR BIT
8696  043204  012600                          MOV     (SP)+,R0                ;RESTORE R0
8697  043206  000002                          RTI
8698
8699  043210  012737  100200  001710  RSLOOP: MOV     #100200,@#RSHSTAT       ;SET DONE AND ERROR BITS
8700  043216  032777  040000  136774          BIT     #BIT14,@RSDS            ;ANY ERRORS?
8701  043224  001003                          BNE     1$                      ;BRANCH IF YES
8702  043226  042737  100000  001710          BIC     #BIT15,@#RSHSTAT        ;CLEAR ERROR BIT
8703  043234  000002                  1$:     RTI                             ;RETURN
8704                                  ;WRITE OK...NOW DO A WRITE CHECK
8705  043236  112737  177775  002076  RS41:   MOVB    #-3,@#RSTRY             ;INIT TRY COUNT
8706  043244  105777  136750          RS42:   TSTB    @RSDS                   ;IS DRIVE READY?
8707  043250  001775                          BEQ     RS42                    ;BRANCH IF NO
8708  043252  004737  040602                  JSR     PC,@#LDRS               ;LOAD RS REGISTERS
8709  043256  112777  000151  136716          MOVB    #151,@RSCS1             ;LOAD FUNCTION AND GO
8710  043264  000002                          RTI                             ;RETURN
8711
8712                                  ;FUNCTION JUST EXECUTED WAS A WRITE CHECK
8713  043266  032737  000400  001710  RSWCK:  BIT     #BIT8,@#RSHSTAT         ;REPEAT FLAG SET?
```

```
8714  043274  001345                        BNE     RSLOOP                  ;BRANCH IF YES
8715  043276  032777  040000  136714        BIT     #BIT14,@RSDS            ;ANY ERRORS?
8716  043304  001417                        BEQ     1$                      ;BRANCH IF NO
8717  043306  105737  002076          3$:   TSTB    @#RSTRY                 ;TRIED 3 TIMES?
8718  043312  001723                        BEQ     RSERR                   ;BRANCH IF YES
8719  043314  005337  002070                DEC     @#RSFUN                 ;SET FUNCTION BACK TO WC
8720  043320  052777  000040  136666        BIS     #BIT5,@RSCS2            ;CLEAR THE ERROR
8721  043326  105237  002076                INCB    @#RSTRY                 ;INCREMENT THE TRY COUNT
8722  043332  013746  177776                MOV     @#PSW,-(SP)
8723  043336  016746  177702                MOV     RS42,-(SP)
8724  043342  000002                        RTI                             ;TRY AGAIN
8725
8726  043344  032777  040000  136642  1$:   BIT     #BIT14,@RSCS2           ;WRITE CHECK ERROR?
8727  043352  001404                        BEQ     2$                      ;BRANCH IF NO
8728  043354  005737  001576                TST     @#RS11                  ;FIRST 2K?
8729  043360  001401                        BEQ     2$                      ;BRANCH IF YES
8730  043362  000751                        BR      3$
8731
8732                                  ;WRITE CHECK WAS OK...NOW DO A READ.
8733  043364  112737  177775  002076  2$:   MOVB    #-3,@#RSTRY             ;INIT TRY COUNT
8734  043372  105777  136622          RS43: TSTB    @RSDS                   ;IS DRIVE READY?
8735  043376  001775                        BEQ     RS43                    ;BRANCH IF NO
8736  043400  004737  040602                JSR     PC,@#LDRS               ;LOAD RS REGISTERS
8737  043404  013777  002010  136576        MOV     @#RSNEWH,@RSBAE         ;LOAD BAE
8738  043412  013777  002006  136566        MOV     @#RSNEWL,@RSBA          ;LOAD BUS ADR
8739  043420  112777  000171  136554        MOVB    #171,@RSCS1             ;LOAD FUNCTION AND GO
8740  043426  000002                        RTI                             ;RETURN
8741
8742                                  ;FUNCTION JUST EXECUTED WAS A READ.
8743  043430  032737  000400  001710  RSREAD: BIT     #BIT8,@#RSHSTAT         ;REPEAT FLAG SET?
8744  043436  001264                        BNE     RSLOOP                  ;BRANCH IF YES
8745  043440  032777  040000  136552        BIT     #BIT14,@RSDS            ;ANY ERRORS?
8746  043446  001417                        BEQ     1$                      ;BRANCH IF NO
8747  043450  105737  002076                TSTB    @#RSTRY                 ;TRIED 3 TIMES?
8748  043454  001642                        BEQ     RSERR                   ;BRANCH IF YES
8749  043456  005337  002070                DEC     @#RSFUN                 ;RESTORE FUN TO READ
8750  043462  052777  000040  136524        BIS     #BIT5,@RSCS2            ;CLEAR ALL ERRORS
8751  043470  105237  002076                INCB    @#RSTRY                 ;INCREMENT TRY COUNT
8752  043474  013746  177776                MOV     @#PSW,-(SP)
8753  043500  012746  043372                MOV     #RS43,-(SP)
8754  043504  000002                        RTI                             ;TRY AGAIN
8755  043506  112737  000200  001710  1$:   MOVB    #200,@#RSHSTAT          ;SET DONE FLAG
8756  043514  000002                        RTI                             ;RETURN
8757                                  ;;*****************************************************************
8758                                  .SBTTL  UNIBUS EXERCISER SERVICE ROUTINE
8759                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
8760                                  ;;*****************************************************************
8761  043516  104412                  UBESRV: SAVREG
8762  043520  004737  053126                  JSR     PC,@#LDKT               ;GO TO LOW CORE
8763  043524  012704  002236                  MOV     #UBETBL+6,R4            ;GET ADDRESS OF UBECR1
8764  043530  005774  000000                  TST     @(R4)                   ;WAS THERE AN ERROR?
8765  043534  100440                          BMI     UBE2                    ;BRANCH IF YES
8766  043536  012746  000003                  MOV     #3,-(SP)                ;PUT DEVICE ID IN STACK
8767  043542  004737  052316                  JSR     PC,@#GIVEMAP            ;GIVE UP MAP REG
8768  043546  062737  000776  001662          ADD     #776,@#UBESAV           ;INCREMENT UBE BUS ADR
8769  043554  005537  001664                  ADC     @#UBESAV+2
```

```
8770  043560  013737  001662  001666          MOV     @#UBESAV,@#UBEADR
8771  043566  013737  001664  001670          MOV     @#UBESAV+2,@#UBEADR+2
8772  043574  012746  001666                  MOV     #UBEADR,-(SP)
8773  043600  004737  052052                  JSR     PC,@#GETMAP
8774  043604  013754  001670                  MOV     @#UBEADR+2,@-(R4)       ;LOAD UBECR2
8775  043610  013754  001666                  MOV     @#UBEADR,@-(R4)         ;LOAD UBEBA
8776  043614  012754  170000                  MOV     #170000,@-(R4)          ;LOAD UBECC
8777  043620  004737  053214                  JSR     PC,@#RESKT              ;GO BACK TO ORIGINAL CORE
8778  043624  104414                          RESREG
8779  043626  012777  064545  136402          MOV     #64545,@UBETBL+6        ;RESTART UBE
8780  043634  000002                          RTI                             ;RETURN
8781
8782                          ;UBE ERROR-IS IT LAST MEMORY?
8783  043636  005037  001276          UBE2:   CLR     @#$TMP0
8784  043642  162704  000004                  SUB     #4,R4                   ;ADJUST R4
8785  043646  017403  000002                  MOV     @2(R4),R3               ;GET BECR2
8786  043652  042703  000003                  BIC     #3,R3                   ;GET RID OF ADDRESS BITS
8787  043656  022703  000400                  CMP     #400,R3                 ;WAS ERROR A TIMEOUT?
8788  043662  001050                          BNE     UBEERR                  ;BRANCH IF NO
8789  043664  017437  000000  001672          MOV     @(R4),@#ERRBA           ;SAVE BUS ADR OF ERROR
8790  043672  017437  000002  001674          MOV     @2(R4),@#ERRBA+2
8791  043700  042737  177774  001674          BIC     #177774,@#ERRBA+2
8792  043706  004737  051534                  JSR     PC,@#PHYMAP             ;GET PHYSICAL ADDRESS THAT TIMED OUT
8793  043712  162737  000004  001476          SUB     #4,@#PA1500             ;ADJUST PHYSICAL ADR THAT FAILED
8794  043720  005637  001500                  SBC     @#PA2116                ;UBE STOPS AT ADR+4
8795  043724  023737  001476  001562          CMP     @#PA1500,@#MXMMLO       ;AT MAXIMUM MEMORY LO?
8796  043732  001022                          BNE     MHOLE                   ;BRANCH IF NO
8797  043734  023737  001500  001560          CMP     @#PA2116,@#MXMMHI       ;AT MAX MEMORY HI?
8798  043742  001016                          BNE     MHOLE                   ;BRANCH IF NO
8799  043744  012746  000003                  MOV     #3,-(SP)                ;PUT DEVICE ID ON STACK
8800  043750  004737  052316                  JSR     PC,@#GIVEMAP
8801  043754  005726                          TST     (SP)+
8802  043756  004737  051434                  JSR     PC,@#UBEINIT
8803  043762  004737  053214                  JSR     PC,@#RESKT
8804  043766  104414                          RESREG
8805  043770  012777  064545  136240          MOV     #64545,@UBETBL+6
8806  043776  000002                          RTI
8807
8808  044000  010637  001276          MHOLE:  MOV     SP,@#$TMP0
8809  044004  013737  001212  001302  UBEERR: MOV     @#$LPERR,@#$TMP2        ;SAVE LOOP ARROR ADR
8810  044012  012737  044054  001212          MOV     #UBE3,@#$LPERR          ;SET LOOP ADR
8811  044020  012703  000022                  MOV     #22,R3
8812  044024  005737  001276                  TST     @#$TMP0
8813  044030  001002                          BNE     1$
8814  044032  104007                          ERROR   7
8815  044034  000407                          BR      UBE3
8816  044036  013737  001476  001226  1$:     MOV     @#PA1500,@#$GDDAT
8817  044044  013737  001500  001230          MOV     @#PA2116,@#$BDDAT
8818  044052  104012                          ERROR   12
8819
8820                          ;RESTART UBE IN SAME MEMORY
8821  044054  013737  001302  001212  UBE3:   MOV     @#$TMP2,@#$LPERR        ;RESTORE ERROR LOOP ADR
8822  044062  010446                          MOV     R4,-(SP)                ;SAVE R4
8823  044064  012704  002230                  MOV     #UBETBL,R4              ;GET ADDRESS OF UBE TABLE
8824  044070  012734  170000                  MOV     #170000,@(R4)+          ;SET UBECC
8825  044074  013734  001666                  MOV     @#UBEADR,@(R4)+         ;SET UBEBA <15:00>
```

```
8826  044100  005074  000004                  CLR     @4(R4)                  ;CLEAR ALL ERRORS
8827  044104  013734  001670                  MOV     @#UBEADR+2,@(R4)+       ;SET EXT ADR BITS
8828  044110  012774  064545  000000          MOV     #64545,@(R4)            ;START UBE
8829  044116  012604                          MOV     (SP)+,R4                ;RESTORE R4
8830  044120  004737  053214                  JSR     PC,@#RESKT
8831  044124  104414                          RESREG
8832  044126  000002                          RTI                             ;RETURN
8833
8834                                  ;;*****************************************************************
8835                                  .SBTTL  MASS BUS TESTER SERVICE ROUTINE
8836                                  ;*      SEE DOCUMENTATION FOR FUNCTIONAL DESCRIPTION OF ROUTINE
8837                                  ;;*****************************************************************
8838  044130  104412                  MBTSRV: SAVREG
8839  044132  004737  053126                  JSR     PC,@#LDKT               ;GO TO LOW CORE
8840  044136  005037  001276                  CLR     @#$TMP0
8841  044142  012704  002246                  MOV     #MBTTBL,R4              ;GET ADDRESS OF ADDRESS OF CS1 REG
8842  044146  032734  040000                  BIT     #BIT14,@(R4)+           ;ANY ERRORS?
8843  044152  001007                          BNE     1$                      ;BRANCH IF YES
8844  044154  004737  053214          2$:     JSR     PC,@#RESKT              ;GO BACK TO ORIGINAL CORE
8845  044160  104414                          RESREG
8846  044162  112777  000161  136056          MOVB    #161,@MBTTBL
8847  044170  000002                          RTI                             ;RESTART MBT AND RETURN
8848  044172  062704  000010          1$:     ADD     #10,R4                  ;ADJUST R4
8849  044176  032774  004000  000000          BIT     #BIT11,@(R4)            ;NON-EXISTANT MEMORY ERROR?
8850  044204  001436                          BEQ     MBTERR                  ;BRANCH IF NO
8851  044206  162704  000006                  SUB     #6,R4                   ;ADJUST R4
8852  044212  013437  001476                  MOV     @(R4)+,@#PA1500         ;GET BUS ADR
8853  044216  013437  001500                  MOV     @(R4)+,@#PA2116         ;GET BUS ADR EXT
8854  044222  162737  000004  001476          SUB     #4,@#PA1500             ;ADJUST BUS ADR
8855  044230  005637  001500                  SBC     @#PA2116
8856  044234  023737  001476  001562          CMP     @#PA1500,@#MXMMLO       ;IS IT LAST MEMORY?
8857  044242  001015                          BNE     MEMHOLE                 ;BRANCH IF NO
8858  044244  023737  001500  001560          CMP     @#PA2116,@#MXMMHI       ;CHECK EXT ADR BITS
8859  044252  001011                          BNE     MEMHOLE
8860  044254  005724                          TST     (R4)+                   ;INCREMENT R4
8861  044256  052774  000047  000000          BIS     #47,@(R4)               ;CLEAR THE ERROR
8862  044264  012734  000007                  MOV     #7,@(R4)+               ;SELECT UNIT 7
8863  044270  005074  177766                  CLR     @-12(R4)                ;CLEAR WORD COUNT
8864  044274  000727                          BR      2$                      ;CONTINUE
8865
8866  044276  010637  001276  MEMHOLE:       MOV     SP,@#$TMP0
8867  044302  013737  001212  001302  MBTERR: MOV     @#$LPERR,@#$TMP2        ;SAVE LOOP ADDRESS
8868  044310  012737  044352  001212          MOV     #1$,@#$LPERR            ;SET NEW LOOP ADR
8869  044316  012703  000020                  MOV     #20,R3                  ;PUT DEVICE ID IN R3
8870  044322  005737  001276                  TST     @#$TMP0
8871  044326  001002                          BNE     2$
8872  044330  104007                          ERROR   7
8873  044332  000407                          BR      1$
8874  044334  013737  001476  001226  2$:     MOV     @#PA1500,@#$GDDAT
8875  044342  013737  001500  001230          MOV     @#PA2116,@#$BDDAT
8876  044350  104013                          ERROR   13
8877  044352  013737  001302  001212  1$:     MOV     @#$TMP2,@#$LPERR        ;RESTORE LOOP ADR
8878  044360  012704  002256                  MOV     #MBTTBL+10,R4           ;GET ADR OF MBTTBL+10
8879  044364  015400                          MOV     @-(R4),R0               ;GET BUS ADR EXTENDED
8880  044366  015401                          MOV     @-(R4),R1               ;GET BUS ADR
8881  044370  015402                          MOV     @-(R4),R2               ;GET WORD COUNT
```

```
8882  044372  006302                          ASL     R2                      ;ADJUST WORD COUNT
8883  044374  160201                          SUB     R2,R1                   ;FORM START ADR OF THIS XFER
8884  044376  005600                          SBC     R0
8885  044400  052774  000047  000010          BIS     #47,@10(R4)             ;CLEAR THE WORLD
8886  044406  012774  000007  000010          MOV     #7,@10(R4)              ;SELECT UNIT 7
8887  044414  005724                          TST     (R4)+                   ;ADJUST R4
8888  044416  010134                          MOV     R1,@(R4)+               ;RESTORE BUS ADR
8889  044420  010074  000000                  MOV     R0,@(R4)
8890  044424  004737  053214                  JSR     PC,@#RESKT              ;GO BACK TO ORIGINAL CORE
8891  044430  104414                          RESREG
8892  044432  112777  000161  135606          MOVB    #161,@MBTTBL            ;START MBT AGAIN
8893  044440  000002                          RTI                             ;RETURN
8894                          ;;*****************************************************************
8895                          .SBTTL  LINE CLOCK SERVICE ROUTINE
8896                          ;*      THIS ROUTINE FIRST REMAPS PROGRAM EXECUTION TO LOW
8897                          ;*      MEMORY. IT THEN INCREMENTS AND KEEPS TRACK OF THE
8898                          ;*      SECOND AND MINUTE COUNTS KEPT IN LOCATIONS "LTICKS"
8899                          ;*      AND "MTICKS" RESPECTIVELY.
8900                          ;;*****************************************************************
8901  044442  104412          LKSRV:  SAVREG
8902  044444  004737  053126          JSR     PC,@#LDKT               ;GO TO LOW CORE
8903  044450  105237  001602          INCB    @#LTICKS                ;INCREMENT TICK COUNT
8904  044454  122737  000074  001602  CMPB    #60.,@#LTICKS           ;ONE SECOND YET?
8905  044462  001014                  BNE     1$                      ;BRANCH IF NO
8906  044464  105237  001603          INCB    @#LTICKS+1              ;INCREMENT SECOND COUNT
8907  044470  105037  001602          CLRB    @#LTICKS                ;CLEAR SECOND COUNT
8908  044474  122737  000074  001603  CMPB    #60.,@#LTICKS+1         ;ONE MINUTE YET?
8909  044502  001004                  BNE     1$                      ;BRANCH IF NO
8910  044504  105037  001603          CLRB    @#LTICKS+1
8911  044510  005237  001600          INC     @#MTICKS                ;INCREMENT MINUTE COUNT
8912  044514  004737  053214  1$:     JSR     PC,@#RESKT              ;RESTORE THE KT
8913  044520  104414                  RESREG
8914  044522  012737  000100  177546  MOV     #BIT6,@#LKS             ;CLEAR READY BIT IN CLOCK
8915  044530  000002                  RTI                             ;RETURN
8916
8917                          ;;*****************************************************************
8918
8919                          .SBTTL  SCOPE HANDLER ROUTINE
8920
8921                          ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
8922                          ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
8923                          ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
8924                          ;*SW14=1         LOOP ON TEST
8925                          ;*SW11=1         INHIBIT ITERATIONS
8926                          ;*SW09=1         LOOP ON ERROR
8927                          ;*CALL
8928                          ;*      SCOPE           ;;SCOPE=IOT
8929
8930  044532                  $SCOPE:
8931  044532  032737  040000  177570          BIT     #SW14,@#SWR     ;;LOOP ON PRESENT TEST?
8932  044540  001077                          BNE     $OVER           ;;YES IF SW14=1
8933                          ;#####START OF CODE FOR THE XOR TESTER#####
8934  044542  000416          $XTSTR: BR      6$                      ;;IF RUNNING ON THE "XOR" TESTER CHANGE
8935                                                                  ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
8936  044544  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
8937  044550  012737  044570  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
```

```
 8938  044556  005737  177060                 TST     @#177060        ;;TIME OUT ON XOR?
 8939  044562  012637  000004                 MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 8940  044566  000453                         BR      $SVLAD          ;;GO TO THE NEXT TEST
 8941  044570  022626                 5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
 8942  044572  012637  000004                 MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
 8943  044576  000413                         BR      7$              ;;LOOP ON THE PRESENT TEST
 8944  044600                         6$:;#####END OF CODE FOR THE XOR TESTER#####
 8945  044600  105767  134400         2$:     TSTB    $ERFLG          ;;HAS AN ERROR OCCURRED?
 8946  044604  001421                         BEQ     3$              ;;BR IF NO
 8947  044606  126767  134405  134370         CMPB    $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
 8948  044614  101015                         BHI     3$              ;;BR IF NO
 8949  044616  032737  001000  177570         BIT     #BIT09,@#SWR    ;;LOOP ON ERROR?
 8950  044624  001404                         BEQ     4$              ;;BR IF NO
 8951  044626  016767  134360  134354 7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
 8952  044634  000441                         BR      $OVER
 8953  044636  105067  134342         4$:     CLRB    $ERFLG          ;;ZERO THE ERROR FLAG
 8954  044642  005067  134450                 CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
 8955  044646  000415                         BR      1$              ;;ESCAPE TO THE NEXT TEST
 8956  044650  032737  004000  177570 3$:     BIT     #BIT11,@#SWR    ;;INHIBIT ITERATIONS?
 8957  044656  001011                         BNE     1$              ;;BR IF YES
 8958  044660  105767  134314                 TSTB    $PASS           ;;IF FIRST PASS OF PROGRAM
 8959  044664  001406                         BEQ     1$              ;;      INHIBIT ITERATIONS
 8960  044666  005267  134314                 INC     $ICNT           ;;INCREMENT ITERATION COUNT
 8961  044672  026767  134420  134306         CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
 8962  044700  002017                         BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
 8963  044702  012767  000001  134276 1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
 8964  044710  016767  000054  134400         MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
 8965  044716  011667  134266         $SVLAD: MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
 8966  044722  011667  134264                 MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
 8967  044726  005067  134366                 CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
 8968  044732  112767  000001  134257         MOVB    #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
 8969  044740  105767  134240         $OVER:  TSTB    $ERFLG.         ;;ANY ERRORS?
 8970  044744  001403                         BEQ     1$              ;;BRANCH IF NO
 8971  044746  116737  134232  001203         MOVB    $ERFLG,@#$TSTNM+1
 8972  044754  016737  134222  177570 1$:     MOV     $TSTNM,@#DISPLAY ;;DISPLAY TEST NUMBER
 8973  044762  016716  134222                 MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
 8974  044766  000002                         RTI                     ;;FIXES PS
 8975  044770  000010                 $MXCNT: 10                      ;;MAX. NUMBER OF ITERATIONS
 8976                                 ;;**************************************************************
 8977
 8978                                 .SBTTL  ERROR HANDLER ROUTINE
 8979
 8980                                 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
 8981                                 ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
 8982                                 ;*AND GO TO $ERRTYP ON ERROR
 8983                                 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
 8984                                 ;*SW15=1         HALT ON ERROR
 8985                                 ;*               HALT CAN OCCUR BEFORE AND AFTER THE ERROR TYPEOUT
 8986                                 ;*SW13=1         INHIBIT ERROR TYPEOUTS
 8987                                 ;*SW10=1         BELL ON ERROR
 8988                                 ;*SW09=1         LOOP ON ERROR
 8989                                 ;*CALL
 8990                                 ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER
 8991
 8992  044772                         $ERROR:
 8993  044772  116737  134206  001203         MOVB    $ERFLG,@#$TSTNM+1
```

```
 8994  045000  105267  134200          7$:   INCB    $ERFLG          ;;SET THE ERROR FLAG
 8995  045004  001775                        BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
 8996  045006  016737  134170  177570        MOV     $TSTNM,@#DISPLAY;;DISPLAY TEST NUMBER AND ERROR FLAG
 8997  045014  005737  177570                TST     @#SWR           ;;HALT ON ERROR = 1?
 8998  045020  100001                        BPL     8$              ;;BRANCH IF NO
 8999  045022  000000                        HALT                    ;;YES--HALT
 9000  045024  032737  002000  177570  8$:   BIT     #BIT10,@#SWR    ;;BELL ON ERROR?
 9001  045032  001402                        BEQ     1$              ;;NO - SKIP
 9002  045034  104400  001322                TYPE    ,$BELL          ;;RING BELL
 9003  045040  005267  134150          1$:   INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
 9004  045044  011667  134150                MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
 9005  045050  162767  000002  134142        SUB     #2,$ERRPC
 9006  045056  117767  134136  134132        MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
 9007  045064  032737  020000  177570        BIT     #BIT13,@#SWR    ;;SKIP TYPEOUT IF SET
 9008  045072  001004                        BNE     2$              ;;SKIP TYPEOUTS
 9009  045074  004767  000056                JSR     PC,$ERRTYP      ;;GO TO USER ERROR ROUTINE
 9010  045100  104400  001327                TYPE    ,$CRLF
 9011  045104  005737  177570          2$:   TST     @#SWR           ;;HALT ON ERROR
 9012  045110  100001                        BPL     9$              ;;SKIP IF CONTINUE
 9013  045112  000000                        HALT                    ;;HALT ON ERROR!
 9014  045114  022767  036750  132720  9$:   CMP     #$ENDAD,42      ;;ACT-11?
 9015  045122  001001                        BNE     3$              ;;BRANCH IF NO
 9016  045124  000000                        HALT                    ;;YES
 9017  045126  032737  001000  177570  3$:   BIT     #BIT09,@#SWR    ;;LOOP ON ERROR SWITCH SET?
 9018  045134  001402                        BEQ     4$              ;;BR IF NO
 9019  045136  016716  134050                MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
 9020  045142  005767  134152          4$:   TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
 9021  045146  001402                        BEQ     5$              ;;BR IF NONE
 9022  045150  016716  134144                MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
 9023  045154                          5$:
 9024  045154  000002                        RTI                     ;;RETURN
 9025                                  ;;**************************************************************
 9026                                  .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
 9027
 9028                                  ;*THIS ROUTINE FIRST TYPES A STANDARD MESSAGE CONSISTING OF THE
 9029                                  ;*VIRTUAL PC, THE PHYSICAL PC, THE PSW AT THE TIME OF THE ERROR CALL,
 9030                                  ;*AND THE  SUB-PASS COUNT. THE SUB-PASS COUNT CONSISTS OF THE SUB PASS COUNT IN THE
 9031                                  ;*HIGH BYTE AND THE PASS COUNT IN THE LOW BYTE.
 9032                                  ;*
 9033                                  ;*IT THEN USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
 9034                                  ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE"
 9035                                  ;*THE ERROR MESSAGE POINTER AND TYPES THE ERROR MESSAGE. THE DATA
 9036                                  ;*HEADER POINTER IS THEN OBTAINED AND A DATA HEADER IS TYPED.
 9037                                  ;*THE DATA POINTER AND DATA FORMAT ARE THEN OBTAINED. THERE ARE
 9038                                  ;*FOUR TYPES OF DATA FORMAT, AS FOLLOWS:
 9039                                  ;*
 9040                                  ;*      0       TYPE THE CONTENTS OF THE DATA TABLE WORD IN
 9041                                  ;*              6 DIGIT OCTAL FORMAT
 9042                                  ;*      1       CONVERT THE CONTENTS OF THE DATA TABLE WORD TO
 9043                                  ;*              22 BITS AND TYPE AN 8 DIGIT OCTAL NUMBER
 9044                                  ;*      2       TYPE THE CONTENTS OF THE DATA TABLE WORD AND
 9045                                  ;*              THE WORD+2 IN 8 DIGIT OCTAL FORMAT
 9046                                  ;*      3       USE THE CONTENTS OF THE DATA TABLE WORD AS A
 9047                                  ;*              DEVICE ID AND TYPE THE DEVICES NAME
 9048                                  ;*      4       CONVERT THE TWO WORDS POINTED TO BY THE DATA
 9049                                  ;*              TABLE TO FLOATING POINT FORMAT AND TYPE.
```

```
                                        ;*      5       CONVERT THE FOUR WORDS POINTED TO BY THE DATA
                                        ;*              TABLE TO FLOATING  DOUBLE FORMAT AND TYPE
                                        ;;*****************************************************************

045156  104412                  $ERRTYP:SAVREG
045160  104400  001327                  TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
045164  004737  046656                  JSR     PC,@#TYPTIME    ;GO TYPE THE TIME
045170  104400  053660                  TYPE    ,MSG3
045174  104400  001327                  TYPE    ,$CRLF
045200  016746  134014                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                ;;TYPE THE VIRTUAL PC
045204  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
045206  104400  046430                  TYPE    ,8$
045212  013700  001474                  MOV     @#VADR,R0               ;SAVE VADR
045216  013737  001220  001474          MOV     @#$ERRPC,@#VADR         ;SAVE THE VIR PC FOR CONVERSION
045224  122737  000014  001216          CMPB    #14,@#$ITEMB
045232  003403                          BLE     51$
045234  105737  001216                  TSTB    @#$ITEMB                ;ERROR ZERO?
045240  001005                          BNE     42$                     ;BRANCH IF NO
045242  004737  051642          51$:    JSR     PC,@#CNVADR             ;CONVERT TO 22 BITS
045246  010037  001474                  MOV     R0,@#VADR
045252  000407                          BR      41$
045254  013737  001474  001476  42$:    MOV     @#VADR,@#PA1500
045262  005037  001500                  CLR     @#PA2116
045266  010037  001474                  MOV     R0,@#VADR
045272  012746  001476          41$:    MOV     #PA1500,-(SP)           ;PUT ADDRESS OFPC ON STACK
045276  004737  047710                  JSR     PC,@#$DB2O              ;CONVERT TO ASCII
045302  062716  000003                  ADD     #3,(SP)                 ;GET RID OF 3 MS DIGITS
045306  012667  000002                  MOV     (SP)+,30$               ;SAVE POINTER TO ASCII
045312  104400                          TYPE                            ;TYPE IT
045314  000000                  30$:    .WORD
045316  104400  046430                  TYPE    ,8$
045322  016646  000030                  MOV     30(SP),-(SP)            ;GET PSW AT TIME OF ERROR
045326  104402                          TYPOC                           ;TYPE IT
045330  104400  046430                  TYPE    ,8$
045334  016746  134244                  MOV     $MAINT,-(SP)    ;;SAVE $MAINT FOR TYPEOUT
                                                                ;;TYPE THE MAINTENANCE REG
045340  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
045342  104400  046430                  TYPE    ,8$
045346  116746  133630                  MOVB    $TSTNM,-(SP)
045352  105066  000001                  CLRB    1(SP)
045356  104402                          TYPOC                   ;TYPE THE TEST NUMBER
045360  104400  046430                  TYPE    ,8$
045364  013746  001532                  MOV     @#SUBPASS,-(SP)
045370  162716  000060                  SUB     #60,(SP)
045374  104402                          TYPOC
045376  104400  046430                  TYPE    ,8$
045402  016746  133572                  MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                                                ;;TYPE THE PASS COUNT
045406  104402                          TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
045410  104400  001327                  TYPE    ,$CRLF
045414  005000                          CLR     R0
045416  153700  001216                  BISB    @#$ITEMB,R0             ;PICK UP THE INDEX
045422  001431                          BEQ     6$                      ;EXIT IF ZERO
045424  022700  000007          1$:     CMP     #7,R0           ;IS THIS ERROR 7?
045430  001551                          BEQ     15$             ;BRANCH IF YES
```

```
9106  045432  005300                        DEC     R0              ;;ADJUST THE INDEX SO THAT IT WILL
9107  045434  006300                        ASL     R0              ;;      WORK FOR THE ERROR TABLE
9108  045436  006300                        ASL     R0
9109  045440  006300                        ASL     R0
9110  045442  062700  002304                ADD     #$ERRTB,R0      ;;FORM TABLE POINTER
9111  045446  012067  000004                MOV     (R0)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
9112  045452  001404                        BEQ     3$              ;;SKIP TYPEOUT IF NO POINTER
9113  045454  104400                        TYPE                    ;;TYPE THE "ERROR MESSAGE"
9114  045456  000000              2$:       .WORD   0               ;;"ERROR MESSAGE" POINTER GOES HERE
9115  045460  104400  001327                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
9116  045464  012067  000004        3$:     MOV     (R0)+,4$        ;;PICKUP "DATA HEADER" POINTER
9117  045470  001404                        BEQ     5$              ;;SKIP TYPEOUT IF 0
9118  045472  104400                        TYPE                    ;;TYPE THE "DATA HEADER"
9119  045474  000000              4$:       .WORD   0               ;;"DATA HEADER" POINTER GOES HERE
9120  045476  104400  001327                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
9121  045502  012001              5$:       MOV     (R0)+,R1                ;;PICKUP "DATA TABLE" POINTER
9122  045504  001004                        BNE     7$              ;;GO TYPE THE DATA
9123  045506  104414              6$:       RESREG
9124  045510  104400  001327                TYPE    ,$CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
9125  045514  000207                        RTS     PC              ;;RETURN
9126  045516  011002              7$:       MOV     (R0),R2         ;GET "DATA FORMAT" POINTER
9127  045520  122712  000001      10$:      CMPB    #1,(R2)         ;DATA FORMAT 1?
9128  045524  001424                        BEQ     9$              ;BRANCH IF YES
9129  045526  122712  000002                CMPB    #2,(R2)         ;DATA FORMAT 2?
9130  045532  001441                        BEQ     11$             ;BRANCH IF YES
9131  045534  122712  000003                CMPB    #3,(R2)         ;DATA FORMAT 3?
9132  045540  001445                        BEQ     24$             ;BRANCH IF YES
9133  045542  122712  000004                CMPB    #4,(R2)         ;DATA FORMAT 4?
9134  045546  001456                        BEQ     40$             ;BRANCH IF YES
9135  045550  122712  000005                CMPB    #5,(R2)         ;DATA FORMAT 5?
9136  045554  001465                        BEQ     60$             ;BRANCH IF YES
9137                          ;;**************************************************************
9138                          ;DATA FORMAT 0
9139  045556  005202                        INC     R2              ;INCREMENT FORMAT POINTER
9140  045560  013146                        MOV     @(R1)+,-(SP)    ;PUSH DATA TO BE TYPED
9141  045562  104402                        TYPOC
9142  045564  005711              13$:      TST     (R1)            ;ANY MORE DATA?
9143  045566  001747                        BEQ     6$              ;BRANCH IF NO
9144  045570  104400  046430                TYPE    ,8$             ;TYPE TWO SPACES
9145  045574  000751                        BR      10$
9146                          ;;**************************************************************
9147                          ;DATA FORMAT 1
9148  045576  005202              9$:       INC     R2              ;INCREMENT FORMAT POINTER
9149  045600  004737  051642                JSR     PC,@#CNVADR     ;GET 22 BIT ADR
9150  045604  012746  001476      14$:      MOV     #PA1500,-(SP)   ;PUSH ADR OF 22 BIT ADR
9151  045610  004737  047710                JSR     PC,@#$DB2O      ;CONVERT TO ASCII
9152  045614  062716  000003                ADD     #3,(SP)         ;DELETE LEADING ZEROS
9153  045620  012667  000002                MOV     (SP)+,12$       ;GET ADR OF ASCII STRING
9154  045624  104400                        TYPE
9155  045626  000000              12$:      .WORD
9156  045630  062701  000002                ADD     #2,R1           ;INCREMENT R1
9157  045634  000753                        BR      13$
9158                          ;;**************************************************************
9159                          ;DATA FORMAT 2
9160  045636  005202              11$:      INC     R2              ;INCREMENT FORMAT POINTER
9161  045640  011100                        MOV     (R1),R0
```

```
9162  045642  012037  001476                  MOV     (R0)+,@#PA1500
9163  045646  011037  001500                  MOV     (R0),@#PA2116
9164  045652  000754                          BR      14$
9165                                  ;;*****************************************************************
9166                                  ;DATA FORMAT 3
9167  045654  005202                  24$:    INC     R2                      ;INCREMENT FORMAT POINTER
9168  045656  013167  000016                  MOV     @(R1)+,25$              ;GET DEVICE ID
9169  045662  062767  053610  000010          ADD     #MSGINX,25$             ;FORM ADR OF ASCIZ ADR
9170  045670  017767  000004  000002          MOV     @25$,25$                ;GET ADR OF ASCIZ
9171  045676  104400                          TYPE
9172  045700  000000                  25$:    .WORD
9173  045702  000730                          BR      13$                     ;CONTINUE
9174                                  ;;*****************************************************************
9175                                  ;DATA FORMAT 4
9176  045704  005202                  40$:    INC     R2
9177  045706  012167  000002                  MOV     (R1)+,44$               ;GET ADDRESS OF DATA
9178  045712  104416                          FL2O                            ;CONVERT TO FLOATING FORMAT
9179  045714  000000                  44$:    .WORD
9180  045716  012667  000002                  MOV     (SP)+,45$               ;GET ADDRESS OF ASCIZ STRING
9181  045722  104400                          TYPE                            ;TYPE THE DATA
9182  045724  000000                  45$:    .WORD
9183  045726  000716                          BR      13$
9184                                  ;;*****************************************************************
9185                                  ;DATA FORMAT 5
9186  045730  005202                  60$:    INC     R2                      ;INCREMENT FORMAT POINTER
9187  045732  012167  000002                  MOV     (R1)+,61$               ;GET ADDRESS OF DATA
9188  045736  104420                          FLD2O                           ;CONVERT TO FLOATING ASCIZ
9189  045740  000000                  61$:    .WORD
9190  045742  012667  000002                  MOV     (SP)+,62$               ;GET ADDRESS OF ASCIZ STRING
9191  045746  104400                          TYPE                            ;TYPE THE DATA
9192  045750  000000                  62$:    .WORD
9193  045752  000704                          BR      13$
9194                                  ;;*****************************************************************
9195                                  ;ERROR 7 DECODE
9196  045754  010300                  15$:    MOV     R3,R0                   ;SAVE R3
9197  045756  062700  053610                  ADD     #MSGINX,R0              ;GEN ADRS OF ASCIZ
9198  045762  011067  000002                  MOV     (R0),16$
9199  045766  104400                          TYPE
9200  045770  000000                  16$:    .WORD
9201  045772  104400  046000                  TYPE    ,65$                    ;;TYPE ASCIZ STRING
9202  045776  000404                          BR      64$                     ;;GET OVER THE ASCIZ
9203                                  ;;65$:   .ASCIZ  /FAILED/<CRLF>
9204  046010                          64$:
9205  046010  010300                          MOV     R3,R0                   ;SAVE DEVICE ID
9206  046012  022700  000010                  CMP     #10,R0                  ;MASS BUS DEVICE?
9207  046016  003403                          BLE     17$                     ;BRANCH IF YES
9208  046020  104400  054114                  TYPE    ,MSG12
9209  046024  000411                          BR      18$
9210                                  ;;*****************************************************************
9211                                  ;MASS BUS ERR
9212  046026  022703  000020          17$:    CMP     #20,R3                  ;MBT ERROR?
9213  046032  001426                          BEQ     26$                     ;BRANCH IF MBT ERROR
9214  046034  002435                          BLT     27$                     ;BRANCH IF UBE ERROR
9215  046036  104400  054227                  TYPE    ,MSG13
9216  046042  022700  000012                  CMP     #12,R0                  ;WAS IT RS?
9217  046046  001140                          BNE     29$                     ;BRANCH IF NO
```

```
                                        ;;*****************************************************************
                                        ;UNIBUS ERROR OR RS04 ERROR
046050  062700  053564        18$:      ADD     #REGINX,R0      ;FORM ADR OF REG TABLE
046054  011000                          MOV     (R0),R0         ;GET ADR OF REG TABLE
046056  022703  000002                   CMP     #2,R3           ;RP3 OR RK?
046062  001404                          BEQ     20$             ;BRANCH IF RK
046064  100406                          BMI     21$             ;BRANCH IF NOT RP03
046066  012704  000007                   MOV     #7,R4           ;SET RP03 SOB COUNT
046072  000423                          BR      22$
046074  012704  000006        20$:      MOV     #6,R4           ;SET RK05 SOB COUNT
046100  000420                          BR      22$
046102  012704  000011        21$:      MOV     #11,R4          ;SET RS04 SOB COUNT
046106  000415                          BR      22$
                                        ;;*****************************************************************
                                        ;MBT ERROR
046110  104400  054416        26$:      TYPE    ,MSG16
046114  012704  000011                   MOV     #11,R4          ;SET MBT SOB COUNT
046120  062700  053564        28$:      ADD     #REGINX,R0
046124  011000                          MOV     (R0),R0         ;GET ADR OF MBT TABLE
046126  000405                          BR      22$             ;GO TYPE REGISTERS
                                        ;UNIBUS  EXERCISER ERROR
046130  104400  054525        27$:      TYPE    ,MSG17
046134  012704  000004                   MOV     #4,R4           ;SET UBE SOB COUNT
046140  000767                          BR      28$             ;GO TYPE UBE REGISTERS
046142  013046                22$:      MOV     @(R0)+,-(SP)    ;GET DATA IN REG
046144  104402                          TYPOC                   ;TYPE IT
046146  104400  046430                   TYPE    ,8$             ;TYPE TWO SPACES
046152  077405                          SOB     R4,22$          ;CONTINUE
                                        ;;*****************************************************************
                                        ;THIS CODE TYPES A PHYSICAL BUS ADDRESS IF THE ERROR WAS AN RP03, RK05, OR UBE
046154  022703  000022                   CMP     #22,R3          ;UBE ERROR?
046160  001454                          BEQ     73$             ;BRANCH IF YES
046162  022703  000002                   CMP     #2,R3           ;RK05?
046166  002445                          BLT     32$             ;BRANCH IF NOT RK OR RP03
046170  001005                          BNE     70$             ;BRANCH IF RP03
                                        ;RK05 ERROR
046172  104400  054721                   TYPE    ,MSG22
046176  012700  002126                   MOV     #RKCS,R0        ;GET ADR OF ADR OF RKCS REG
046202  000404                          BR      71$
                                        ;RP03 ERROR
046204  012700  002104        70$:      MOV     #RP3CS,R0       ;GET ADR OF ADR OF RP3CS REG
046210  104400  054731                   TYPE    ,MSG23
                                        ;GET, CALCULATE, & TYPE PHYSICAL BUS ADDRESS
046214  013001                71$:      MOV     @(R0)+,R1       ;GET BUS ADR EXTENDED BITS
046216  005720                          TST     (R0)+           ;ADJUST R0
046220  013037  001672                   MOV     @(R0)+,@#ERRBA  ;GET BUS ADRESS THAT FAILED
046224  072127  177774                   ASH     #-4,R1          ;GET BITS 4&5 INTO BITS 0&1
046230  042701  177774                   BIC     #177774,R1      ;GET RID OF UNUSED BITS
046234  010137  001674                   MOV     R1,@#ERRBA+2    ;SAVE EXTENDED BITS
046240  162737  000002  001672  74$:      SUB     #2,@#ERRBA      ;DECREMENT BUS ADR
046246  005637  001674                   SBC     @#ERRBA+2
046252  004737  051534                   JSR     PC,@#PHYMAP     ;GO CONVERT TO 22 BIT PHYSICAL
046256  012746  001476                   MOV     #PA1500,-(SP)
046262  004737  047710                   JSR     PC,@#$DB2O      ;CONVERT TO ASCIZ STRING
046266  062716  000003                   ADD     #3,(SP)         ;GET RID OF LEADING ZEROS
046272  012667  000002                   MOV     (SP)+,72$
```

```
9274  046276  104400                    TYPE
9275  046300  000000              72$:  .WORD
9276  046302  104400  001327      32$:  TYPE    ,$CRLF
9277  046306  000167  177174            JMP     6$                  ;EXIT
9278                                    ;GET UBE VIRTUAL ADDRESS
9279  046312  012700  002232      73$:  MOV     #UBETBL+2,R0        ;GET ADR OF UBE TABLE +2
9280  046316  013037  001672            MOV     @(R0)+,@#ERRBA      ;GET BUS ADR THAT FAILED
9281  046322  013037  001674            MOV     @(R0)+,@#ERRBA+2    ;GET BAE BITS
9282  046326  042737  177774  001674    BIC     #177774,@#ERRBA+2   ;MASK OFF ADR BITS
9283  046334  162737  000002  001672    SUB     #2,@#ERRBA
9284  046342  005637  001674            SBC     @#ERRBA+2
9285  046346  000734                    BR      74$                 ;GO CONVERT & TYPE PHYSICAL ADR
9286                                    ;;*****************************************************************
9287                                    ;RP04 ERROR
9288  046350  062700  053564      29$:  ADD     #REGINX,R0
9289  046354  011000                    MOV     (R0),R0             ;FORM ADR OF RP04 TABLE
9290  046356  012704  000011            MOV     #11,R4              ;SET SOB COUNT
9291  046362  013046              31$:  MOV     @(R0)+,-(SP)        ;GET DATA TO BE TYPED
9292  046364  104402                    TYPOC                       ;TYPE DATA
9293  046366  104400  046430            TYPE    ,8$
9294  046372  077405                    SOB     R4,31$              ;CONTINUE
9295  046374  104400  001327            TYPE    ,$CRLF
9296  046400  104400  001327            TYPE    ,$CRLF
9297  046404  012704  000004            MOV     #4,R4               ;SET SOB COUNT
9298  046410  104400  054337            TYPE    ,MSG14
9299  046414  013046              50$:  MOV     @(R0)+,-(SP)        ;GET DTA TO BE TYPED
9300  046416  104402                    TYPOC                       ;TYPE IT
9301  046420  104400  046430            TYPE    ,8$
9302  046424  077405                    SOB     R4,50$              ;CONTINUE
9303  046426  000725                    BR      32$
9304  046430  020040      000     8$:   .ASCIZ  / /                 ;;TWO(2) SPACES
9305          046434                    .EVEN
9306                                    ;;*****************************************************************
9307
9308                                    .SBTTL  TYPE ROUTINE
9309
9310                                    ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
9311                                    ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
9312                                    ;*NOTE1:        $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
9313                                    ;*NOTE2:        $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
9314                                    ;*NOTE3:        $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
9315                                    ;*
9316                                    ;*CALL:
9317                                    ;*1) USING A TRAP INSTRUCTION
9318                                    ;*      TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
9319                                    ;*OR
9320                                    ;*      TYPE
9321                                    ;*      MESADR
9322                                    ;*
9323                                    ;*2) USING A JSR INSTRUCTION
9324                                    ;*      MOV     PS,-(SP)        ;;PUSH PROCESSOR STATUS WORD ON THE STACK
9325                                    ;*      JSR     PC,$TYPE        ;;CALL TYPE ROUTINE
9326                                    ;*      MESADDR                 ;;FIRST ADRESS OF MESSAGE
9327
9328  046434  105767  132613    $TYPE:  TSTB    $TPFLG              ;;IS THERE A TERMINAL?
9329  046440  100002                    BPL     1$                  ;;BR IF YES
```

```
9330  046442  000000                        HALT                    ;;HALT HERE IF NO TERMINAL
9331  046444  000407                        BR      3$              ;;LEAVE
9332  046446  010046              1$:       MOV     R0,-(SP)        ;;SAVE R0
9333  046450  017600  000002                MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
9334  046454  112046              2$:       MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
9335  046456  001005                        BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
9336  046460  005726                        TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
9337  046462  012600                        MOV     (SP)+,R0        ;;RESTORE R0
9338  046464  062716  000002        3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
9339  046470  000002                        RTI                     ;;RETURN
9340  046472  122716  000011        4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
9341  046476  001426                        BEQ     8$
9342  046500  122716  000200                CMPB    #CRLF,(SP)      ;;BRANCH IF NOT
9343  046504  001004                        BNE     5$
9344  046506  005726                        TST     (SP)+           ;;POP  <CR><LF> EQUIV
9345  046510  104400  001327                TYPE    ,$CRLF
9346  046514  000757                        BR      2$              ;;GET NEXT CHARACTER
9347  046516  004767  000056        5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
9348  046522  126726  132524        6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
9349  046526  001352                        BNE     2$              ;;IF NO GO GET NEXT CHAR.
9350  046530  016746  132514                MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
9351                                                                ;;AND THE NULL CHAR.
9352  046534  105366  000001        7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
9353  046540  002770                        BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
9354  046542  004767  000032                JSR     PC,$TYPEC       ;;GO TYPE A NULL
9355  046546  105367  000100                DECB    $CHARCNT        ;;DON'T COUNT THE NULL AS A CHARACTER
9356  046552  000770                        BR      7$              ;;LOOP
9357
9358                                        ;;HORIZONTAL TAB PROCESSOR
9359
9360  046554  112716  000040        8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
9361  046560  004767  000014        9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
9362  046564  132767  000007  000060        BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
9363  046572  001372                        BNE     9$              ;;TAB STOP
9364  046574  005726                        TST     (SP)+           ;;POP SPACE OFF STACK
9365  046576  000726                        BR      2$              ;;GET NEXT CHARACTER
9366  046600  005737  001466        $TYPEC: TST     @#NOTYPE        ;;INHIBIT TYPING?
9367  046604  100423                        BMI     $TYPEX          ;;BRANCH IF YES
9368  046606  105777  132432                TSTB    @$TPS           ;WAIT UNTIL PRINTER IS READY
9369  046612  100372                        BPL     $TYPEC
9370  046614  116677  000002  132424        MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
9371  046622  122766  000015  000002        CMPB    #CR,2(SP)       ;;BRANCH IF
9372  046630  001003                        BNE     1$              ;;NOT <CR>
9373  046632  105067  000014                CLRB    $CHARCNT        ;;
9374  046636  000406                        BR      $TYPEX          ;;EXIT
9375  046640  122766  000012  000002  1$:   CMPB    #LF,2(SP)       ;;BRANCH IF
9376  046646  001402                        BEQ     $TYPEX          ;;<LF>
9377  046650  105227                        INCB    (PC)+           ;;INC SPACE
9378  046652  000000              $CHARCNT:.WORD  0               ;;COUNT
9379  046654  000207              $TYPEX: RTS     PC
9380
9381                                        ;;*****************************************************************
9382                                        .SBTTL  ROUTINE TO TYPE THE ELAPSED RUN TIME OF THE PROGRAM
9383                                        ;*      THIS ROUTINE CONVERTS THE CONTENTS OF LOCATIONS "LTICKS"
9384                                        ;*      AND "MTICKS" TO SECONDS AND MINUTES/HOURS RESPECTIVELY
9385                                        ;*      AND TYPES THEM IN THE FOLLOWING FORMAT:
```

```
                                        ;*              HHH:MM:SS
                                        ;;**************************************************************
                                        ;TYPTIME:SAVREG
046656  104412                                JSR     PC,@#LDKT       ;GO BACK TO LOW CORE
046660  004737  053126                        MOVB    @#LTICKS+1,R1   ;GET SECOND COUNT
046664  113701  001603                        CLR     R0
046670  005000                               DIV     #10.,R0
046672  071027  000012                        ADD     #60,R1
046676  062701  000060                        MOVB    R1,@#TIMEBUF+10
046702  110137  047114                        MOV     R0,R1
046706  010001                               CLR     R0
046710  005000                               DIV     #6,R0
046712  071027  000006                        ADD     #60,R1
046716  062701  000060                        MOVB    R1,@#TIMEBUF+7
046722  110137  047113                        MOV     @#MTICKS,R1     ;GET MINUTE COUNT
046726  013701  001600                        CLR     R0
046732  005000                               DIV     #10.,R0         ;GET HOURS AND MINUTES
046734  071027  000012                        ADD     #60,R1          ;MAKE REMAINDER ASCII
046740  062701  000060                        MOVB    R1,TIMEBUF+5    ;PUT IN BUFFER
046744  110167  000141                        MOV     R0,R1
046750  010001                               CLR     R0
046752  005000                               DIV     #6.,R0
046754  071027  000006                        ADD     #60,R1
046760  062701  000060                        MOVB    R1,TIMEBUF+4
046764  110167  000120                        TST     R0
046770  005700                               BEQ     2$
046772  001434                               MOV     R0,R1
046774  010001                               CLR     R0
046776  005000                               DIV     #10.,R0
047000  071027  000012                        ADD     #60,R1
047004  062701  000060                        MOVB    R1,TIMEBUF+2
047010  110167  000072                        TST     R0
047014  005700                               BEQ     2$
047016  001422                               MOV     R0,R1
047020  010001                               CLR     R0
047022  005000                               DIV     #10,R0
047024  071027  000010                        ADD     #60,R1
047030  062701  000060                        MOVB    R1,TIMEBUF+1
047034  110167  000045                        TST     R0
047040  005700                               BEQ     2$
047042  001410                               MOV     R0,R1
047044  010001                               CLR     R0
047046  005000                               DIV     #10.,R0
047050  071027  000012                        ADD     #60,R1
047054  062701  000060                        MOVB    R1,TIMEBUF
047060  110167  000020                2$:     TYPE    ,TIMEBUF
047064  104400  047104                        TYPE    ,$CRLF
047070  104400  001327                        JSR     PC,@#RESKT      ;GO BACK TO ORIGINAL MEMORY
047074  004737  053214                        RESREG
047100  104414                               RTS     PC
047102  000207
047104    001     001     001   TIMEBUF:.BYTE   1,1,1,72,1,1,72,60,60,0
047107    072     001     001
047112    072     060     060
047115    000
                                        ;;**************************************************************
```

```
                                        .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.
                                        ;;*****************************************************************
047116  104400  056370          TYPSIZ: TYPE    ,SWITCH
047122  104400  053747                  TYPE    ,MSG4
047126  012700  000010                  MOV     #10,R0          ;SET SOB COUNT
047132  005001                          CLR     R1
047134  105761  001606          1$:     TSTB    SYSSIZE(R1)     ;DEVICE AVAILABLE?
047140  001004                          BNE     2$              ;BRANCH IF YES
047142  062701  000002          7$:     ADD     #2,R1           ;INCREMENT INDEX
047146  077006                          SOB     R0,1$           ;CONTINUE
047150  000207                          RTS     PC              ;RETURN
047152  010102                  2$:     MOV     R1,R2           ;GET INDEX
047154  062702  053610                  ADD     #MSGINX,R2      ;GET ADR OF MESSAGE ADR
047160  011267  000002                  MOV     (R2),3$         ;GET ADDRESS OF MESSAGE
047164  104400                          TYPE
047166  000000                  3$:     .WORD
047170  112767  000060  000034          MOVB    #60,4$          ;INIT UNIT NO. BUFFER (ASCII)
047176  116102  001606                  MOVB    SYSSIZE(R1),R2  ;GET WORD WITH AVAILABLE UNITS
047202  012703  000010                  MOV     #10,R3          ;SET SOB COUNT
047206  006002                  6$:     ROR     R2              ;GET UNITS
047210  103002                          BCC     5$              ;BRANCH IF NOT A UNIT
047212  104400  047232                  TYPE    ,4$
047216  005267  000010          5$:     INC     4$
047222  077307                          SOB     R3,6$           ;CONTINUE
047224  104400  001327                  TYPE    ,$CRLF
047230  000744                          BR      7$
047232      000      054      040  4$:     .BYTE   0,54,40,0       ;NUMBER,COMMA,SPACE,TERMINATOR
047235      000
                                        ;;*****************************************************************
                                        .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
                                        ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
                                        ;*OCTAL (ASCII) NUMBER AND TYPE IT.
                                        ;*$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
                                        ;
047236 017646 000000                    $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
047242 116667 000001 000211             MOVB    1(SP),$OFILL            ;;LOAD ZERO FILL SWITCH
047250 112667 000207                    MOVB    (SP)+,$OMODE+1          ;;NUMBER OF DIGITS TO TYPE
047254 062716 000002                    ADD     #2,(SP)                 ;;ADJUST RETURN ADDRESS
047260 000406                           BR      $TYPON
047262 112767 000001 000171             $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
047270 112767 000006 000165             MOVB    #6,$OMODE+1             ;;SET FOR SIX(6) DIGITS
047276 112767 000005 000154             $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
047304 010346                           MOV     R3,-(SP)                ;;SAVE R3
047306 010446                           MOV     R4,-(SP)                ;;SAVE R4
047310 010546                           MOV     R5,-(SP)                ;;SAVE R5
047312 116704 000145                    MOVB    $OMODE+1,R4             ;;GET THE NUMBER OF DIGITS TO TYPE
047316 005404                           NEG     R4
047320 062704 000006                    ADD     #6,R4                   ;;SUBTRACT IT FOR MAX. ALLOWED
047324 110467 000132                    MOVB    R4,$OMODE               ;;SAVE IT FOR USE
047330 116704 000125                    MOVB    $OFILL,R4               ;;GET THE ZERO FILL SWITCH
047334 016605 000012                    MOV     12(SP),R5               ;;PICKUP THE INPUT NUMBER
047340 005003                           CLR     R3                      ;;CLEAR THE OUTPUT WORD
047342 006105                           1$:     ROL     R5              ;;ROTATE MSB INTO "C"
047344 000404                           BR      3$                      ;;GO DO MSB
047346 006105                           2$:     ROL     R5              ;;FORM THIS DIGIT
047350 006105                           ROL     R5
047352 006105                           ROL     R5
047354 010503                           MOV     R5,R3
047356 006103                           3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
047360 105367 000076                    DECB    $OMODE                  ;;TYPE THIS DIGIT?
047364 100016                           BPL     7$                      ;;BR IF NO
047366 042703 177770                    BIC     #177770,R3              ;;GET RID OF JUNK
047372 001002                           BNE     4$                      ;;TEST FOR 0
047374 005704                           TST     R4                      ;;SUPPRESS THIS 0?
047376 001403                           BEQ     5$                      ;;BR IF YES
047400 005204                           4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
047402 052703 000060                    BIS     #'0,R3                  ;;MAKE THIS DIGIT ASCII
047406 052703 000040                    5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
047412 110367 000040                    MOVB    R3,8$                   ;;SAVE FOR TYPING
047416 104400 047456                    TYPE    ,8$                     ;;GO TYPE THIS DIGIT
047422 105367 000032                    7$:     DECB    $OCNT           ;;COUNT BY 1
047426 003347                           BGT     2$                      ;;BR IF MORE TO DO
047430 002402                           BLT     6$                      ;;BR IF DONE
047432 005204                           INC     R4                      ;;INSURE LAST DIGIT ISN'T A BLANK
047434 000744                           BR      2$                      ;;GO DO THE LAST DIGIT
047436 012605                           6$:     MOV     (SP)+,R5        ;;RESTORE R5
047440 012604                           MOV     (SP)+,R4                ;;RESTORE R4
047442 012603                           MOV     (SP)+,R3                ;;RESTORE R3
047444 016666 000002 000004             MOV     2(SP),4(SP)             ;;SET THE STACK FOR RETURNING
047452 012616                           MOV     (SP)+,(SP)
047454 000002                           RTI                             ;;RETURN
047456    000                           8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
047457    000                           .BYTE   0                       ;;TERMINATOR FOR TYPE ROUTINE
047460    000                           $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
047461    000                           $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
047462 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

047464                                  $TYPDS:
047464  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
047466  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
047470  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
047472  010346                          MOV     R3,-(SP)        ;;PUSH R3 ON STACK
047474  010546                          MOV     R5,-(SP)        ;;PUSH R5 ON STACK
047476  012746  020200                  MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
047502  016605  000020                  MOV     20(SP),R5       ;;GET THE INPUT NUMBER
047506  100004                          BPL     1$              ;;BR IF INPUT IS POS.
047510  005405                          NEG     R5              ;;MAKE THE BINARY NUMBER POS.
047512  112766  000055  000001          MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
047520  005000                  1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
047522  012703  047700                  MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
047526  112723  000040                  MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
047532  005002                  2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
047534  016001  047670                  MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
047540  160105                  3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
047542  002402                          BLT     4$              ;;BR IF DONE
047544  005202                          INC     R2              ;;INCREASE THE BCD DIGIT BY 1
047546  000774                          BR      3$
047550  060105                  4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
047552  005702                          TST     R2              ;;CHECK IF BCD DIGIT=0
047554  001002                          BNE     5$              ;;FALL THROUGH IF 0
047556  105716                          TSTB    (SP)            ;;STILL DOING LEADING 0'S?
047560  100407                          BMI     7$              ;;BR IF YES
047562  106316                  5$:     ASLB    (SP)            ;;MSD?
047564  103003                          BCC     6$              ;;BR IF NO
047566  116663  000001  177777          MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
047574  052702  000060          6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
047600  052702  000040          7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
047604  110223                          MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
047606  005720                          TST     (R0)+           ;;JUST INCREMENTING
047610  020027  000010                  CMP     R0,#10          ;;CHECK THE TABLE INDEX
047614  002746                          BLT     2$              ;;GO DO THE NEXT DIGIT
047616  003002                          BGT     8$              ;;GO TO EXIT
047620  010502                          MOV     R5,R2           ;;GET THE LSD
047622  000764                          BR      6$              ;;GO CHANGE TO ASCII
047624  105726                  8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
047626  100003                          BPL     9$              ;;BR IF NO
047630  116663  177777  177776          MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
047636  105013                  9$:     CLRB    (R3)            ;;SET THE TERMINATOR
047640  012605                          MOV     (SP)+,R5        ;;POP STACK INTO R5
047642  012603                          MOV     (SP)+,R3        ;;POP STACK INTO R3
047644  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
047646  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
```

```
 9610  047650  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
 9611  047652  104400  047700                  TYPE    $DBLK           ;;NOW TYPE THE NUMBER
 9612  047656  016666  000002  000004          MOV     2(SP),4(SP)     ;;ADJUST THE STACK
 9613  047664  012616                          MOV     (SP)+,(SP)
 9614  047666  000002                          RTI                     ;;RETURN TO USER
 9615  047670  023420                $DTBL:    10000.
 9616  047672  001750                          1000.
 9617  047674  000144                          100.
 9618  047676  000012                          10.
 9619  047700  000004                $DBLK:    .BLKW   4
 9620                                ;;*****************************************************************
 9621
 9622                                .SBTTL  DOUBLE LENGTH BINARY TO OCTAL ASCII CONVERT ROUTINE
 9623
 9624                                ;*THIS ROUTINE WILL CONVERT A 32-BIT UNSIGNED BINARY NUMBER TO AN
 9625                                ;*UNSIGNED OCTAL ASCIZ NUMBER.
 9626                                ;*CALL
 9627                                ;*      MOV     #PNTR,-(SP)     ;;POINTER TO LOW WORD OF BINARY NUMBER
 9628                                ;*      JSR     PC,@#$DB20      ;;CALL THE ROUTINE
 9629                                ;*      RETURN                  ;;THE ADDRESS OF THE FIRST ASCIZ CHAR. IS ON THE STACK
 9630
 9631
 9632  047710  104412                $DB20:    SAVREG                  ;;SAVE ALL REGISTERS
 9633  047712  016601  000002                  MOV     2(SP),R1        ;;PICKUP THE POINTER TO LOW WORD
 9634  047716  012705  050045                  MOV     #$OCTVL+13.,R5  ;;POINTER TO DATA TABLE
 9635  047722  012704  000014                  MOV     #12.,R4         ;;DO ELEVEN CHARACTERS
 9636  047726  012703  177770                  MOV     #^C7,R3         ;;MASK
 9637  047732  012100                          MOV     (R1)+,R0        ;;LOWER WORD
 9638  047734  012101                          MOV     (R1)+,R1        ;;HIGH WORD
 9639  047736  005002                          CLR     R2              ;;TERMINATOR
 9640  047740  110245                  1$:     MOVB    R2,-(R5)        ;;PUT CHARACTER IN DATA TABLE
 9641  047742  010002                          MOV     R0,R2           ;;GET THIS DIGIT
 9642  047744  005304                          DEC     R4              ;;COUNT THIS CHARACTER
 9643  047746  003016                          BGT     3$              ;;BR IF NOT THE LAST DIGIT
 9644  047750  001414                          BEQ     2$              ;;BR IF IT IS THE LAST DIGIT
 9645  047752  005205                          INC     R5              ;;ALL DIGITS DONE-ADJUST POINTER FOR FIRST
 9646  047754  010566  000002                  MOV     R5,2(SP)        ;;ASCIZ CHAR. & PUT IT ON THE STACK
 9647  047760  122765  000061  000003          CMPB    #61,3(R5)       ;LAST NUBER LEGAL?
 9648  047766  002003                          BGE     4$              ;BRANCH IF YES
 9649  047770  112765  000060  000003          MOVB    #60,3(R5)       ;MAKE IT ZERO
 9650  047776  104414                  4$:     RESREG                  ;;RESTORE ALL REGISTERS
 9651  050000  000207                          RTS     PC              ;;RETURN TO USER
 9652  050002  006203                  2$:     ASR     R3              ;;POSITION THE MASK FOR THE LAST DIGIT
 9653  050004  006001                  3$:     ROR     R1              ;;POSITION THE BINARY NUMBER FOR
 9654  050006  006000                          ROR     R0              ;;      THE NEXT OCTAL DIGIT
 9655  050010  006001                          ROR     R1
 9656  050012  006000                          ROR     R0
 9657  050014  006001                          ROR     R1
 9658  050016  006000                          ROR     R0
 9659  050020  040302                          BIC     R3,R2           ;;MASK OUT ALL JUNK
 9660  050022  062702  000060                  ADD     #'0,R2          ;;MAKE THIS CHAR. ASCII
 9661  050026  000744                          BR      1$              ;;GO PUT IT IN THE DATA TABLE
 9662  050030  000016                $OCTVL:   .BLKB   14.             ;;RESERVE DATA TABLE
 9663                                ;;*****************************************************************
 9664
 9665                                .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

050046                                  $SAVREG:
050046  010046                                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
050050  010146                                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
050052  010246                                  MOV     R2,-(SP)        ;;PUSH R2 ON STACK
050054  010346                                  MOV     R3,-(SP)        ;;PUSH R3 ON STACK
050056  010446                                  MOV     R4,-(SP)        ;;PUSH R4 ON STACK
050060  010546                                  MOV     R5,-(SP)        ;;PUSH R5 ON STACK
050062  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PS OF MAIN FLOW
050066  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PC OF MAIN FLOW
050072  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PS OF CALL
050076  016646  000022                          MOV     22(SP),-(SP)    ;;SAVE PC OF CALL
050102  000002                                  RTI

                                        ;*RESTORE R0-R5
                                        ;*CALL:
                                        ;*      RESREG
050104                                  $RESREG:
050104  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PC OF CALL
050110  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PS OF CALL
050114  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PC OF MAIN FLOW
050120  012666  000022                          MOV     (SP)+,22(SP)    ;;RESTORE PS OF MAIN FLOW
050124  012605                                  MOV     (SP)+,R5        ;;POP STACK INTO R5
050126  012604                                  MOV     (SP)+,R4        ;;POP STACK INTO R4
050130  012603                                  MOV     (SP)+,R3        ;;POP STACK INTO R3
050132  012602                                  MOV     (SP)+,R2        ;;POP STACK INTO R2
050134  012601                                  MOV     (SP)+,R1        ;;POP STACK INTO R1
050136  012600                                  MOV     (SP)+,R0        ;;POP STACK INTO R0
050140  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>
                                        ;*
```

```
 9722                                   ;*IT IS ENTERED BY A TRAP CALL WITH THE ADDRESS OF THE FLOATING
 9723                                   ;*NUMBER IN THE WORD FOLLOWING THE CALL.
 9724                                   ;*IT RETURNS WITH THE ADDRESS OF THE ASCIZ STRING ON THE STACK.
 9725                                   ;*****************************************************************
 9726  050142  104412                  $FL20:  SAVREG
 9727  050144  017600  000000                  MOV     @(SP),R0        ;GET ADDRESS OF DATA
 9728  050150  062716  000002                  ADD     #2,(SP)         ;ADJUST RETURN PC
 9729  050154  016001  000002                  MOV     2(R0),R1        ;PUT SECOND DATA WORD IN R1
 9730  050160  011000                          MOV     (R0),R0         ;PUT FIRST DATA WORD IN R0
 9731  050162  012704  001357                  MOV     #$FLBUFF+23,R4  ;GET ADDRESS OF BUFFER END IN R4
 9732  050166  112744  000000                  MOVB    #0,-(R4)        ;PUT TERMINATOR IN BUFFER
 9733  050172  012705  000005                  MOV     #5,R5           ;SET SOB COUNT FOR FRACTION DIGITS
 9734  050176  010103                  1$:     MOV     R1,R3           ;GET LSB'S OF FRACTION
 9735  050200  042703  177770                  BIC     #^C7,R3         ;SAVE LS 3 BITS
 9736  050204  062703  000060                  ADD     #60,R3          ;MAKE THEM ASCII
 9737  050210  110344                          MOVB    R3,-(R4)        ;STORE IN BUFFER
 9738  050212  073027  177775                  ASHC    #-3,R0          ;SHIFT NUMBER TO NEXT 3 BITS
 9739  050216  077511                          SOB     R5,1$           ;CONTINUE FOR 7 DIGITS
 9740  050220  010103                          MOV     R1,R3           ;GET NEXT DIGITS
 9741  050222  042703  177776                  BIC     #^C1,R3         ;ONLY WANT 1 BIT
 9742  050226  062703  000060                  ADD     #60,R3          ;MAKE THEM ASCII
 9743  050232  110344                          MOVB    R3,-(R4)        ;STORE IN BUFFER
 9744  050234  112744  000040                  MOVB    #40,-(R4)       ;PUT SPACE IN BUFFER
 9745  050240  073027  177777                  ASHC    #-1,R0
 9746  050244  012705  000002                  MOV     #2,R5           ;SET SOB COUNT
 9747  050250  010103                  3$:     MOV     R1,R3           ;GET LOW WORD
 9748  050252  042703  177770                  BIC     #^C7,R3         ;MASK 3 BITS
 9749  050256  062703  000060                  ADD     #60,R3          ;MAKE THEM ASCII
 9750  050262  110344                          MOVB    R3,-(R4)        ;PUT IN BUFFER
 9751  050264  073027  177775                  ASHC    #-3,R0          ;GET NEXT 3 BITS
 9752  050270  077511                          SOB     R5,3$           ;CONVERT THEM
 9753  050272  010103                          MOV     R1,R3           ;
 9754  050274  042703  177776                  BIC     #^C1,R3         ;ONLY WANT 1 BIT
 9755  050300  062703  000060                  ADD     #60,R3          ;MAKE IT ASCII
 9756  050304  110344                          MOVB    R3,-(R4)        ;PUT IN BUFFER
 9757  050306  112744  000040                  MOVB    #40,-(R4)       ;PUT SPACE IN BUFFER
 9758  050312  112744  000040                  MOVB    #40,-(R4)
 9759  050316  072127  177777                  ASH     #-1,R1          ;GET FIRST 3 BITS OF EXPONENT
 9760  050322  012705  000002                  MOV     #2,R5           ;SET SOB COUNT FOR 2 DIGITS
 9761  050326  010103                  2$:     MOV     R1,R3           ;GET LSB'S OF EXPONENT
 9762  050330  042703  177770                  BIC     #^C7,R3         ;SAVE 3 BITS
 9763  050334  062703  000060                  ADD     #60,R3          ;MAKE THEM ASCII
 9764  050340  110344                          MOVB    R3,-(R4)        ;STORE IN BUFFER
 9765  050342  072127  177775                  ASH     #-3,R1          ;GET NEXT 3 BITS
 9766  050346  077511                          SOB     R5,2$           ;CONTINUE
 9767  050350  010103                          MOV     R1,R3           ;GET LAST 2 BITS OF EXPONENT
 9768  050352  042703  177774                  BIC     #^C3,R3         ;MAKE SURE ONLY 2 BITS
 9769  050356  062703  000060                  ADD     #60,R3          ;MAKE THEM ASCII
 9770  050362  110344                          MOVB    R3,-(R4)        ;STORE IN BUFFER
 9771  050364  112744  000040                  MOVB    #40,-(R4)       ;PUT SPACE IN BUFFER
 9772  050370  112744  000040                  MOVB    #40,-(R4)
 9773  050374  042700  177776                  BIC     #^C1,R0         ;GET SIGN BIT (IT WAS EXTENDED)
 9774  050400  062700  000060                  ADD     #60,R0          ;MAKE IT ASCII
 9775  050404  110044                          MOVB    R0,-(R4)        ;PUT IT IN THE BUFFER
 9776  050406  104414                  RESREG
 9777  050410  011646                          MOV     (SP),-(SP)      ;SAVE RETURN PC
```

```
 9778  050412  016666  000004  000002          MOV     4(SP),2(SP)     ;AND RETURN PSW
 9779  050420  012766  001334  000004          MOV     #$FLBUFF,4(SP)  ;PUT BUFFER ADDRESS ON STACK
 9780  050426  000006                          RTT                     ;RETURN
 9781                                          .EVEN
 9782                                  ;;*****************************************************************
 9783                                  .SBTTL  CONVERT FLOATING DOUBLE BINARY TO OCTAL ASCIZ
 9784                                  ;*
 9785                                  ;*THIS ROUTINE CONVERTS A 64 BIT FLOATING NUMBER TO AN OCTAL
 9786                                  ;*ASCIZ STRING IN THE FOLLOWING FORMAT:
 9787                                  ;*
 9788                                  ;*              U  VVV  WWW XXXXXX YYYYYY ZZZZZZ
 9789                                  ;*
 9790                                  ;*      WHERE   U = SIGN BIT
 9791                                  ;*              V = 8-BIT EXPONENT (RIGHT JUSTIFIED)
 9792                                  ;*              W = FRACTION BITS<57:51> (RIGHT JUSTIFIED)
 9793                                  ;*              X = FRACTION BITS <50:35>
 9794                                  ;*              Y = FRACTION BITS <34:19>
 9795                                  ;*              Z = FRACTION BITS <18:03>
 9796                                  ;*
 9797                                  ;*IT IS ENTERED BY A TRAP CALL WITH THE ADDRESS OF THE FLOATING
 9798                                  ;*NUMBER IN THE WORD FOLLOWING THE CALL.
 9799                                  ;*IT RETURNS WITH THE ADDRESS OF THE ASCIZ STRING ON THE STACK.
 9800                                  ;;*****************************************************************
 9801  050430  104412                  $FLD20: SAVREG
 9802  050432  017667  000000  000006          MOV     @(SP),1$        ;GET ADDRESS OF DATA TO CONVERT
 9803  050440  062716  000002                  ADD     #2,(SP)         ;ADJUST RETURN PC
 9804  050444  104416                          FL20                    ;CONVERT MS 32 BITS
 9805  050446  000000                  1$:     .WORD
 9806  050450  012600                          MOV     (SP)+,R0        ;GET ADDRESS OF CONVERTED DATA
 9807  050452  010067  130722                  MOV     R0,$BUFF        ;SAVE IT
 9808  050456  062700  000041                  ADD     #41,R0          ;ADJUST TO END OF BUFFER
 9809  050462  105040                          CLRB    -(R0)           ;PUT TERMINATOR IN BUFFER
 9810  050464  016701  177756                  MOV     1$,R1           ;GET ADDRESS OF DATA TO CONVERT
 9811  050470  062701  000004                  ADD     #4,R1           ;ADJUST TO LOWER 32 BITS
 9812  050474  012102                          MOV     (R1)+,R2        ;SAVE THE DATA
 9813  050476  012103                          MOV     (R1)+,R3
 9814  050500  012701  000002                  MOV     #2,R1           ;SET LOOP COUNT
 9815  050504  012704  000005          3$:     MOV     #5,R4           ;SET LOOP COUNT
 9816  050510  010305          4$:     MOV     R3,R5           ;GET LS 32 BITS OF DATA
 9817  050512  042705  177770                  BIC     #^C7,R5         ;MASK 3 BITS
 9818  050516  062705  000060                  ADD     #60,R5          ;MAKE THEM ASCII
 9819  050522  110540                          MOVB    R5,-(R0)        ;PUT IN BUFFER
 9820  050524  073227  177775                  ASHC    #-3,R2          ;GET NEXT 3 BITS
 9821  050530  077411                          SOB     R4,4$           ;CONTINUE
 9822  050532  010305                          MOV     R3,R5           ;GET LS 32 BITS
 9823  050534  042705  177776                  BIC     #^C1,R5         ;ONLY WANT 1 BIT
 9824  050540  062705  000060                  ADD     #60,R5          ;MAKE IT ASCII
 9825  050544  110540                          MOVB    R5,-(R0)        ;PUT IN TABLE
 9826  050546  112740  000040                  MOVB    #40,-(R0)       ;PUT SPACE IN TABLE
 9827  050552  073227  177777                  ASHC    #-1,R2
 9828  050556  077126                          SOB     R1,3$           ;CONVERT NEXT 16 BITS
 9829  050560  104414                          RESREG
 9830  050562  011646                          MOV     (SP),-(SP)      ;ADJUST STACK
 9831  050564  016666  000004  000002          MOV     4(SP),2(SP)     ;TO RETURN WITH ADDRESS
 9832  050572  016766  130602  000004          MOV     $BUFF,4(SP)     ;OF BUFFER ON STACK
 9833  050600  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

050602                          $RAND:
050602  010046                          MOV     R0,-(SP)        ;;PUSH R0 ON STACK
050604  010146                          MOV     R1,-(SP)        ;;PUSH R1 ON STACK
050606  010246                          MOV     R2,-(SP)        ;;PUSH R2 ON STACK
050610  016700  130706                  MOV     $LONUM,R0       ;;SET R0 WITH LOW
050614  016701  130704                  MOV     $HINUM,R1       ;;SET R1 WITH HIGH
050620  012702  177771                  MOV     #-7,R2          ;;SET SHIFT COUNT
050624  006300                  1$:     ASL     R0              ;;SHIFT R0 LEFT AND
050626  006101                          ROL     R1              ;;ROTATE CARRY INTO R1 AND
050630  005202                          INC     R2              ;;CHECK FOR DONE
050632  001374                          BNE     1$              ;;CONTINUE SHIFT LOOP
050634  066700  130662                  ADD     $LONUM,R0       ;;ADD NUMBER TO MAKE X 129
050640  005501                          ADC     R1              ;;PROPOGATE CARRY
050642  066701  130656                  ADD     $HINUM,R1       ;;ADD NUMBER TO MAKE X 129
050646  062700  001057                  ADD     #1057,R0        ;;ADD LOW CONSTANT
050652  005501                          ADC     R1              ;;PROPOGATE CARRY
050654  062701  047401                  ADD     #47401,R1       ;;ADD HIGH CONSTANT
050660  010067  130636                  MOV     R0,$LONUM       ;;SAVE R0
050664  010167  130634                  MOV     R1,$HINUM       ;;SAVE R1
050670  012602                          MOV     (SP)+,R2        ;;POP STACK INTO R2
050672  012601                          MOV     (SP)+,R1        ;;POP STACK INTO R1
050674  012600                          MOV     (SP)+,R0        ;;POP STACK INTO R0
050676  000207                          RTS     PC              ;;RETURN
                                ;;**********************************************************************
                                .SBTTL  FLOATING POINT NUMBER GENERATOR
                                ;*      THIS ROUTINE GENERATES TWO RANDOM FLOATING POINT NUMBERS
                                ;*      IN EITHER SINGLE OR DOUBLE PRECISION. FOR SINGLE PRECISION
                                ;*      THE NUMBERS ARE STORED IN $TMP0 AND $TMP2. DOUBLE PRECISICN
                                ;*      NUMBERS ARE STORED IN $TMP0 AND $TMP4.
                                ;*      IN EITHER SINGLE OR DOUBLE THE EXTENDED EXPONENT IS STORED
                                ;*      IN $REG0 AND $REG1.
                                ;;**********************************************************************
050700  012767  000002  000130  FLTDBL: MOV     #2,SOBDBL       ;SET LOOP FOR 2, FOUR WORD NUMBERS
050706  016700  000124          FLTSGL: MOV     SOBDBL,R0       ;SET WORD LENGTH LOOP
050712  012702  001276                  MOV     #$TMP0,R2       ;GET ADDRESS TO STORE WORDS IN
050716  012701  000002          2$:     MOV     #2,R1           ;SET NUMBER OF WORDS TO 2
050722  004767  177654          1$:     JSR     PC,$RAND        ;GET RANDOM NUMBER
050726  022701  000002                  CMP     #2,R1           ;FIRST TIME?
050732  001404                          BEQ     3$              ;BRANCH IF YES
050734  022767  000002  000074          CMP     #2,SOBDBL       ;DOUBLE PRECISION?
050742  001407                          BEQ     4$              ;BRANCH IF YES
050744  016703  130554          3$:     MOV     $HINUM,R3       ;GET EXPONENT PART
050750  042703  000177                  BIC     #177,R3         ;CHECK FOR MINUS ZERO
```

```
9890  050754  022703  100000                      CMP     #BIT15,R3
9891  050760  001760                              BEQ     1$              ;BRANCH IF MINUS ZERO
9892  050762  016722  130536          4$:         MOV     $HINUM,(R2)+    ;SAVE HINUM
9893  050766  016722  130530                      MOV     $LONUM,(R2)+    ;SAVE LONUM
9894  050772  077125                              SOB     R1,1$           ;CONTINUE
9895  050774  077030                              SOB     R0,2$           ;CONTINUE FOR DOUBLE PREC
9896  050776  012746  001276                      MOV     #$TMP0,-(SP)    ;PUT ADDRESS OF NUMBER ON STACK
9897  051002  012746  001002                      MOV     #1002,-(SP)     ;PUT CONTROL WORD ON STACK
9898  051006  022767  000002  000022              CMP     #2,SOBDBL       ;DOUBLE PREC?
9899  051014  001002                              BNE     5$              ;BRANCH IF NO
9900  051016  012716  001004                      MOV     #1004,(SP)      ;CHANGE CONTROL WORD
9901  051022  004767  000012          5$:         JSR     PC,EXPEXT       ;CALCULATE EXT EXPONENTS
9902  051026  012767  000001  000002              MOV     #1,SOBDBL       ;INIT SOBDBL FOR SINGLE PREC
9903  051034  000207                              RTS     PC              ;RETURN
9904  051036  000001                  SOBDBL:     .WORD   1
9905                                  ;;**************************************************************
9906                                  .SBTTL  FLOATING POINT EXPONENT EXTENSION
9907                                  ;*      THIS ROUTINE CONVERTS THE ACTUAL EXPONENT OF A FLOATING POINT
9908                                  ;*      NUMBER INTO AN ACTUAL EXPONENT OF 200 AND AN EXTENDED
9909                                  ;*      EXPONENT EQUAL TO THE DIFFERANCE BETWEEN THE ORIGINAL
9910                                  ;*      ACTUAL EXPONENT AND 200.
9911                                  ;*
9912                                  ;*      THE ROUTINE IS ENTERED WITH A CONTROL WORD ON THE STACK.
9913                                  ;*      BIT 15 OF THE CONTROL WORD INDICATES WHETHER THE NUMBER
9914                                  ;*      IS IN MEMORY (<15>=0) OR IN AN ACCUMULATOR (<15>=1).
9915                                  ;*      IF THE NUMBER IS IN AN ACCUMULATOR, BITS <9:8> INDICATE
9916                                  ;*      THE ACCUMULATOR NUMBER.  IF THE NUMBER(S) IS IN MEMORY,
9917                                  ;*      BITS <9:8> INDICATE THE NUMBER OF NUMBERS TO CONVERT AND
9918                                  ;*      BITS <2:0> INDICATE THE WORD LENGTH OF THE NUMBER(S).
9919                                  ;*      IN THE CASE OF A MEMORY CONVERSION, THE ADDRESS OF THE
9920                                  ;*      FIRST WORD TO CONVERT IS ALSO ON THE STACK (PRECEDING
9921                                  ;*      THE CONTROL WORD).
9922                                  ;;**************************************************************
9923  051040  012605                  EXPEXT: MOV     (SP)+,R5        ;SAVE RETURN PC
9924  051042  012600                          MOV     (SP)+,R0        ;GET CONTROL WORD
9925  051044  100437                          BMI     1$              ;BRANCH IF ACC CONVERSION
9926  051046  012601                          MOV     (SP)+,R1        ;GET START ADDRESS
9927  051050  162700  000400                  SUB     #400,R0
9928  051054  012702  001276                  MOV     #$TMP0,R2       ;GET OFFSET FROM $TMP0
9929  051060  160102                          SUB     R1,R2
9930  051062  005402                          NEG     R2
9931  051064  006202                          ASR     R2
9932  051066  062702  001256                  ADD     #$REG0,R2       ;GEN ADDRESS OF EXT WORD
9933  051072  011103                  3$:     MOV     (R1),R3         ;GET DATA
9934  051074  042703  100177                  BIC     #100177,R3      ;GET EXPONENT
9935  051100  072327  177771                  ASH     #-7,R3          ;RIGHT JUSTIFY EXPONENT
9936  051104  162703  000200                  SUB     #200,R3         ;CONVERT TO 2'S COMPLIMENT
9937  051110  010312                          MOV     R3,(R2)         ;ADD TO EXTENDED EXPONENT
9938  051112  042711  077600                  BIC     #77600,(R1)     ;MAKE ACTUAL
9939  051116  052711  040000                  BIS     #BIT14,(R1)     ;EXPONENT 200
9940  051122  162700  000400                  SUB     #400,R0         ;ANY MORE WORDS?
9941  051126  100435                          BMI     2$              ;BRANCH IF NO
9942  051130  110003                          MOVB    R0,R3           ;GET WORD LENGTH
9943  051132  006303                          ASL     R3
9944  051134  060301                          ADD     R3,R1           ;SELECT NEXT NUMBER ADDRESS
9945  051136  062702  000002                  ADD     #2,R2           ;SELECT NEXT EXTENDED ADDRESS
```

```
                        BR      3$                      ;CONTINUE
                ;ACCUMULATOR CONVERSION
                1$:     ASH     #-2,R0                  ;GET ACCUMULATOR NUMBER
                        BIC     #177477,R0              ;
                        MOV     R0,R2                   ;GENERATE
                        ASH     #-5,R2                  ;ADDRESS OF
                        ADD     #$ACO,R2                ;EXTENDED EXPONENT
                        BIC     #300,5$                 ;GENERATE INSTRUCTION
                        BIS     R0,5$                   ;TO GET EXPONENT
                5$:     STEXP   AC0,R3                  ;GET EXPONENT
                        ADD     R3,(R2)                 ;ADD TO EXTENDED EXPONENT
                        CLR     R3
                        BIC     #300,4$                 ;GENERATE INSTRUCTION
                        BIS     R0,4$                   ;TO LOAD EXPONENT BACK TO ACC
                4$:     LDEXP   R3,AC0                  ;LOAD EXPONENT OF 200
                2$:     MOV     R5,-(SP)                ;RESTORE RETURN PC
                        RTS     PC                      ;RETURN
                ;;**********************************************************************

                .SBTTL  POWER DOWN AND UP ROUTINES

                ;POWER DOWN ROUTINE
                $PWRDN: MOV     #$ILLUP,@#PWRVEC        ;;SET FOR FAST UP
                        MOV     #340,@#PWRVEC+2         ;;PRIO:7
                        MOV     R0,-(SP)                ;;PUSH R0 ON STACK
                        MOV     R1,-(SP)                ;;PUSH R1 ON STACK
                        MOV     R2,-(SP)                ;;PUSH R2 ON STACK
                        MOV     R3,-(SP)                ;;PUSH R3 ON STACK
                        MOV     R4,-(SP)                ;;PUSH R4 ON STACK
                        MOV     R5,-(SP)                ;;PUSH R5 ON STACK
                        MOV     SP,$SAVR6               ;;SAVE SP
                        MOV     #$PWRUP,@#PWRVEC        ;;SET UP VECTOR
                        HALT
                        BR      .-2                     ;;HANG UP

                ;POWER UP ROUTINE
                $PWRUP: MOV     $SAVR6,SP               ;;GET SP
                        CLR     $SAVR6                  ;;WAIT LOOP FOR THE TTY
                1$:     INC     $SAVR6                  ;;WAIT FOR THE INC
                        BNE     1$                      ;;OF  WORD
                        MOV     (SP)+,R5                ;;POP STACK INTO R5
                        MOV     (SP)+,R4                ;;POP STACK INTO R4
                        MOV     (SP)+,R3                ;;POP STACK INTO R3
                        MOV     (SP)+,R2                ;;POP STACK INTO R2
                        MOV     (SP)+,R1                ;;POP STACK INTO R1
                        MOV     (SP)+,R0                ;;POP STACK INTO R0
                        MOV     #$PWRDN,@#PWRVEC        ;;SET UP THE POWER DOWN VECTOR
                        MOV     #340,@#PWRVEC+2         ;;PRIO:7
                        TYPE                            ;REPORT THE POWER FAILURE
                $PWRMG: .WORD   $POWER                  ;;POWER FAIL MESSAGE POINTER
                        MOV     (PC)+,(SP)              ;;RESTART AT START
                $PWRAD: .WORD   START                   ;;RESTART ADDRESS
                        RTI
                $ILLUP: HALT                            ;;THE POWER UP SEQUENCE WAS STARTED
                        BR      .-2                     ;; BEFORE THE POWER DOWN WAS COMPLETE
                $SAVR6: 0                               ;;PUT THE SP HERE
```

```
10002  051362  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
10003  051370  000122
10004                                          .EVEN
10005                                  ;;**************************************************************
10006
10007                                  .SBTTL  TRAP DECODER
10008
10009                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
10010                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
10011                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
10012                                  ;*GO TO THAT ROUTINE.
10013
10014  051372  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
10015  051374  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
10016  051400  005740                          TST     -(R0)           ;;BACKUP BY 2
10017  051402  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
10018  051404  016000  051412                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
10019  051410  000200                          RTS     R0              ;;GO TO ROUTINE
10020
10021
10022                                  .SBTTL  TRAP TABLE
10023
10024                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
10025                                  ;*BY THE "TRAP" INSTRUCTION.
10026
10027                                  ;       ROUTINE
10028                                  ;       -------
10029  051412                          $TRPAD:
10030  051412  046434                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
10031  051414  047262                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
10032  051416  047236                          $TYPOS  ;;CALL=TYPOS    TRAP+4(104404)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
10033  051420  047276                          $TYPON  ;;CALL=TYPON    TRAP+6(104406)  TYPE OCTAL NUMBER (AS PER LAST CALL)
10034  051422  047464                          $TYPDS  ;;CALL=TYPDS    TRAP+10(104410) TYPE DECIMAL NUMBER (WITH SIGN)
10035  051424  050046                          $SAVREG ;;CALL=SAVREG   TRAP+12(104412) SAVE R0-R5 ROUTINE
10036  051426  050104                          $RESREG ;;CALL=RESREG   TRAP+14(104414) RESTORE R0-R5 ROUTINE
10037  051430  050142                          $FL20   ;;CALL=FL20     TRAP+16(104416)
10038  051432  050430                          $FLD20  ;;CALL=FLD20    TRAP+20(104420)
10039                                  ;;**************************************************************
10040                                  .SBTTL  UNIBUS EXERCISER INITIALIZATION ROUTINE
10041                                  ;*THIS ROUTINE INITIALIZES THE BASE ADDRESS FOR THE
10042                                  ;*UNIBUS EXERCISER AND LOADS UP THE EXERCISER REGISTERS.
10043                                  ;;**************************************************************
10044  051434  012703  001662          UBEINIT:MOV     #UBESAV,R3      ;GET ADDRESS OF NEXT UBE ADRES
10045  051440  005023                          CLR     (R3)+           ;INITIALIZE
10046  051442  005023                          CLR     (R3)+
10047  051444  005023                          CLR     (R3)+
10048  051446  005013                          CLR     (R3)
10049
10050
10051                                  ;SET UP THE UBE AND START IT
10052  051450  012702  002230                  MOV     #UBETBL,R2      ;GET ADDRESS OF UBE TABLE
10053  051454  005072  000010                  CLR     @10(R2)         ;CLEAR ALL ERRORS
10054  051460  012772  043516  000012          MOV     #UBESRV,@12(R2) ;SET UP UBE VECTOR
10055  051466  012772  000340  000014          MOV     #PR7,@14(R2)    ;SET UP UBE VECTOR PSW
10056  051474  012732  170000                  MOV     #170000,@(R2)+  ;SET CC FOR 2K WORD TRANSFER
10057                                                                  ;UBE IS DOING BYTE TRANSFERS
```

```
10058  051500  012746  000003                  MOV     #3,-(SP)        ;PUT DEVICE ID IN STACK
10059  051504  012746  001666                  MOV     #UBEADR,-(SP)   ;PUT ADDRESS OF PHYSICAL BA ON STACK
10060  051510  004737  052052                  JSR     PC,@#GETMAP     ;GO GET MAP REGISTER
10061  051514  013732  001666                  MOV     @#UBEADR,@(R2)+ ;LOAD UBE BUS ADDRESS
10062  051520  013732  001670                  MOV     @#UBEADR+2,@(R2)+;LOAD ADR BITS 16 & 17
10063  051524  052737  000040  172516          BIS     #40,@#SR3       ;ENABLE MAP
10064  051532  000207                          RTS     PC              ;RETURN
10065                                  ;;*****************************************************************
10066                                  .SBTTL  CONVERT UNIBUS VIRTUAL ADDRESS TO PHYSICAL ADDRESS
10067                                  ;*      THIS ROUTINE CONVERTS THE CONTENTS OF LOCATIONS
10068                                  ;*      "ERRBA" AND "ERRBA+2" FROM A VIRTUAL 18-BIT ADDRESS
10069                                  ;*      TO A PHYSICAL 22-BIT ADDRESS AS MAPPED BY THE APPROPRIATE
10070                                  ;*      MAP REGISTER. THE 22-BIT ADDRESS IS STORED IN LOCATIONS
10071                                  ;*      "PA2116" AND "PA1500".
10072                                  ;;*****************************************************************
10073  051534  104412                  PHYMAP: SAVREG
10074  051536  013703  001672                  MOV     @#ERRBA,R3      ;GET BUS ADDRESS <15:00>
10075  051542  013702  001674                  MOV     @#ERRBA+2,R2    ;GET BUS ADDRESS <17:16>
10076  051546  042702  177774                  BIC     #177774,R2      ;
10077  051552  032737  000040  172516          BIT     #BIT5,@#MMR3    ;MAP ON?
10078  051560  001005                          BNE     1$              ;BRANCH IF YES
10079  051562  010337  001476                  MOV     R3,@#PA1500     ;PHY ADR=BUS ADR
10080  051566  010237  001500                  MOV     R2,@#PA2116
10081  051572  000421                          BR      MAPEND
10082  051574  010305                  1$:     MOV     R3,R5           ;SAVE ADR BITS <15:00>
10083  051576  073227  000005                  ASHC    #5,R2           ;GET MAP REG SELECT BITS
10084  051602  042702  000003                  BIC     #3,R2           ;
10085  051606  062702  170200                  ADD     #MAPLO,R2       ;FORM ADDRESS OF MAP REG
10086  051612  012237  001476                  MOV     (R2)+,@#PA1500  ;GET CONTENTS OF MAP REG LO
10087  051616  011237  001500                  MOV     (R2),@#PA2116   ;GET CONTENTS OF MAP REG HI
10088  051622  042705  160000                  BIC     #160000,R5      ;FORM PHYSICAL ADDRESS
10089  051626  060537  001476                  ADD     R5,@#PA1500     ;THAT TIMED OUT
10090  051632  005537  001500                  ADC     @#PA2116        ;
10091  051636  104414                  MAPEND: RESREG
10092  051640  000207                          RTS     PC
10093
10094                                  ;;*****************************************************************
10095                                  .SBTTL  CONVERT A VIRTUAL ADDRESS TO A PHYSICAL ADDRESS
10096                                  ;*      THIS ROUTINE CONVERTS A 16-BIT VIRTUAL ADDRESS TO A
10097                                  ;*      22-BIT PHYSICAL ADDRESS. THE VIRTUAL ADDRESS IS
10098                                  ;*      ASSUMED TO BE IN LOCATION "VADR" AND THE PHYSICAL
10099                                  ;*      ADDRESS IS PLACED IN LOCATIONS "PA2116" AND "PA1500".
10100                                  ;*
10101                                  ;*      IF MEMORY MANAGEMENT IS OFF THE PHYSICAL ADDRESS IS
10102                                  ;*      GENERATED BY SUBTRACTING THE CONTENTS OF LOCATION
10103                                  ;*      "FACTOR" FROM THE VIRTUAL ADDRESS. THIS LOCATION
10104                                  ;*      CONTAINS THE BYTE OFFSET BETWEEN THE RELOCATED CODE
10105                                  ;*      AND THE NON-RELOCATED CODE.
10106                                  ;*
10107                                  ;*      IF MEMORY MANAGEMENT IS ON, THE CONTENTS OF THE
10108                                  ;*      APPROPRIATE PAR REGISTER IS ADDED(AFTER ADJUSTMENT)
10109                                  ;*      TO THE LEAST SIGNIFICANT 13 BITS OF THE VIRTUAL ADDRESS.
10110                                  ;;*****************************************************************
10111  051642  104412                  CNVADR: SAVREG
10112  051644  013703  001474                  MOV     @#VADR,R3               ;GET VIRTUAL ADDRESS TO CONVERT
10113  051650  105737  001503                  TSTB    @#MMON                  ;IS MEMORY MGMT ON?
```

```
10114  051654  001426                          BEQ     1$                      ;BRANCH IF NO
10115  051656  005002                          CLR     R2
10116  051660  073227  000003                  ASHC    #3,R2                   ;GET PAR SELECT BITS
10117  051664  072327  177775                  ASH     #-3,R3                  ;RETURN VIR ADDR TO ORIGINAL
10118  051670  042703  160000                  BIC     #160000,R3              ;MAKE SURE SIGN DIDN'T EXTEND
10119  051674  006102                          ROL     R2                      ;MAKE R2 EVEN FOR WORD ADDRESSING
10120  051676  062702  172340                  ADD     #KIPAR0,R2              ;GET ADDRESS OF PAR
10121  051702  011205                          MOV     (R2),R5                 ;GET PAR DATA
10122  051704  005004                          CLR     R4                      ;SETUP R4
10123  051706  073427  000006                  ASHC    #6,R4                   ;SHIFT PAR DATA
10124  051712  060305                          ADD     R3,R5                   ;FORM PHYSICAL ADDRESS
10125  051714  005504                          ADC     R4
10126  051716  010437  001500          2$:     MOV     R4,@#PA2116             ;SAVE PHYSICAL
10127  051722  010537  001476                  MOV     R5,@#PA1500             ;ADDRESS
10128  051726  104414                          RESREG
10129  051730  000207                          RTS     PC                      ;RETURN
10130  051732  163703  001506          1$:     SUB     @#FACTOR,R3             ;FORM PHYSICAL ADDRESS
10131  051736  005004                          CLR     R4
10132  051740  010305                          MOV     R3,R5
10133  051742  000765                          BR      2$                      ;RETURN
10134
10135                                  ;;*****************************************************************
10136                                  .SBTTL  ROUTINE TO CHECK RELOCATED DATA
10137                                  ;*ROUTINE TO CHECK DATA RELOCATED
10138                                  ;*CALL: R0=     HIGHEST ADDRESS +2 OF SOURCE DATA
10139                                  ;*      R2=     HIGHEST ADDRESS +2 OF DEST DATA
10140                                  ;*      R5=     LOWEST ADDRESS OF THE SOURCE DATA
10141                                  ;*
10142                                  ;*      THIS ROUTINE USES A COMPARE INSTRUCTION TO CHECK
10143                                  ;*      THE DATA THAT WAS RELOCATED. IF A PARITY ERROR OCCURS
10144                                  ;*      DURING THIS CHECK A SPECIAL ERROR MESSAGE IS TYPED
10145                                  ;*      INSTED OF THE UNEXPECTED TRAP MESSAGE.
10146                                  ;;*****************************************************************
10147  051744  012737  052006  000114  CHKDAT: MOV     #2$,@#CACHVEC           ;SETUP PARITY VECTOR
10148  051752  024042                          CMP     -(R0),-(R2)             ;CHECK DATA
10149  051754  001013                          BNE     99$+2
10150  051756  005112                          COM     (R2)                    ;COMPLEMENT DEST DATA
10151  051760  005112                          COM     (R2)                    ;TWICE
10152  051762  021210                          CMP     (R2),(R0)               ;CHECK DATA
10153  051764  001006                          BNE     99$
10154  051766  020005                  1$:     CMP     R0,R5                   ;BRANCH IF ALL DATA NOT CHECKED
10155  051770  001365                          BNE     CHKDAT
10156  051772  012737  052640  000114          MOV     #.PARSRV,@#CACHVEC              ;RESTORE CACHVEC
10157  052000  000207                          RTS     PC                      ;RETURN
10158  052002  000262                  99$:    SEV
10159  052004  000207                          RTS     PC
10160  052006  013737  177744  001302  2$:     MOV     @#MEMERR,@#$TMP2                ;SAVE ERROR REG
10161  052014  013737  177740  001304          MOV     @#LOADRS,@#$TMP3                ;SAVE ERROR ADR
10162  052022  013737  177742  001306          MOV     @#HIADRS,@#$TMP4
10163  052030  010237  001474                  MOV     R2,@#VADR
10164  052034  010037  001276                  MOV     R0,@#$TMP0
10165  052040  104005                          ERROR   5
10166  052042  012737  177777  177744          MOV     #-1,@#MEMERR            ;CLEAR ERROR REG
10167  052050  000754                          BR      99$                     ;RETURN
10168
10169                                  ;;*****************************************************************
```

```
10170                                   .SBTTL  ROUTINE TO GET A MAP REGISTER
10171                                   ;*THIS ROUTINE TAKES AN 18 BIT RANDOM NUMBER, FINDS TWO
10172                                   ;*CONSECUTIVE MAP REGISTERS THAT ARE NOT IN USE, LOADS THE
10173                                   ;*REGISTERS WITH THE PHYSICAL ADDRESS MINUS THE RANDOM NUMBER
10174                                   ;*AND THE NUMBER + 4K, AND RETURNS A NEW BUS ADDRESS, BASED
10175                                   ;*ON THE RANDOM NUMBER.
10176                                   ;*
10177                                   ;*      MAP REGISTERS 0 AND 1 ARE NOT USED IF THE PROGRAM IS
10178                                   ;*      RUNNING ON ACT11. THIS ALLOWS "MOTHER" TO ACCESS THE
10179                                   ;*      END OF PASS HOOKS.
10180                                   ;*
1C181                                   ;*      THE MAP TABLE (MAPTBL) CONTAINS 4 BYTES, ONE FOR EACH
10182                                   ;*      UNIBUS DEVICE. IF THE UBE IS PRESENT IT USES THE
10183                                   ;*      4TH BYTE. WHEN A REGISTER IS ASSIGNED TO A DEVICE,
10184                                   ;*      THE LOWER 4 ADDRESS BITS OF THAT REGISTER ARE PLACED
10185                                   ;*      IN THE TABLE. WHEN A DEVICE REQUESTS A REGISTER PAIR
10186                                   ;*      THIS TABLE IS THEN SEARCHED TO SEE IF THE REGISTER
10187                                   ;*      PAIR IS IN USE.
10188                                   ;*      ENTER WITH:
10189                                   ;*              4(SP)=DEVICE ID
10190                                   ;*              2(SP)=ADDRESS OF THE PHYSICAL ADDRESS
10191                                   ;**************************************************************
10192  052052  016600  000004  GETMAP: MOV     4(SP),R0                ;GET DIVICE ID
10193  052056  016601  000002          MOV     2(SP),R1                ;GET ADR OF PHY ADR
10194  052062  013746  177776          MOV     @#PSW,-(SP)             ;SAVE CURRENT PRIORITY
10195  052066  005116                  COM     (SP)                    ;MAKE IT READY FOR RESTORE
10196  052070  042716  177437          BIC     #^CPR7,(SP)             ;
10197  052074  000237                  SPL     7                       ;IF THIS IS RK CALL, LOCK OUT UBE
10198  052076  104412                  SAVREG
10199  052100  012137  001226          MOV     (R1)+,@#$GDDAT          ;SAVE PHYSICAL
10200  052104  012137  001230          MOV     (R1)+,@#$BDDAT          ;ADDRESS
10201  052110  004737  050602  2$:     JSR     PC,@#$RAND              ;GET RANDOM NUMBER
10202  052114  013702  001524          MOV     @#$HINUM,R2             ;GET HIGH RANDOM NUMBER
10203  052120  013703  001522          MOV     @#$LONUM,R3             ;GET LOW RANDOM NUMBER
10204  052124  073227  177764          ASHC    #-14,R2                 ;CONVERT TO 20 BIT NUMBER
10205  052130  042702  177760          BIC     #177760,R2              ;GET RID OF 11 BITS OF SIGN EXT
10206  052134  022702  000016          CMP     #16,R2                  ;LEGAL MAP REG SELECT?
10207  052140  100001                  BPL     3$                      ;BRANCH IF YES
10208  052142  000762                  BR      2$                      ;TRY AGAIN
10209  052144  005737  001504  3$:     TST     @#QV                    ;ACT11 (QV OR AUTO)?
10210  052150  001403                  BEQ     4$                      ;BRANCH IF NO
10211  052152  122702  000000          CMPB    #0,R2                   ;MAP SELECT 0?
10212  052156  001754                  BEQ     2$                      ;BRANCH IF YES.(ACT MUST
10213                                                                  ;USE THIS MAP REG)
10214  052160  010204          4$:     MOV     R2,R4                   ;SAVE MAP SELECT BITS
10215  052162  042703  100000          BIC     #BIT15,R3               ;CLEAR SELECT BIT 0
10216  052166  073227  177776          ASHC    #-2,R2                  ;FORM 18 BIT ADDRESS
10217  052172  042703  000001          BIC     #BIT0,R3                ;MAKE SURE ITS EVEN
10218  052176  010241                  MOV     R2,-(R1)                ;RETURN NEW BUS ADDRESS
10219  052200  010341                  MOV     R3,-(R1)                ;TO THE APPROPRIATE HANDLER
10220  052202  012705  000004          MOV     #4,R5                   ;SET SOB COUNT
10221  052206  120465  001655  1$:     CMPB    R4,MAPTBL-1(R5)         ;IS THIS MAP IN USE?
10222  052212  001435                  BEQ     5$                      ;BRANCH IF YES
10223  052214  077504                  SOB     R5,1$                   ;CONTINUE
10224  052216  110460  001656          MOVB    R4,MAPTBL(R0)           ;PUT MAP SELECT BITS IN TABLE
10225  052222  072427  000003          ASH     #3,R4                   ;FORM INDEX TO GET MAP REG ADDR
```

```
10226  052226  062704  170200                ADD     #MAPLO,R4               ;GENERATE MAP ADDRESS
10227  052232  042703  160000                BIC     #160000,R3              ;GET LS 13 BITS OF RAND NO.
10228  052236  013701  001226                MOV     @#$GDDAT,R1             ;GET PHYSICAL
10229  052242  013702  001230                MOV     @#$BDDAT,R2             ;ADDRESS
10230  052246  160301                        SUB     R3,R1                   ;GENERATE MAP
10231  052250  005602                        SBC     R2                      ;REGISTER DATA
10232  052252  010124                        MOV     R1,(R4)+                ;LOAD THE
10233  052254  010224                        MOV     R2,(R4)+                ;FIRST MAP REGISTER
10234  052256  062701  020000                ADD     #20000,R1               ;ADD 4K
10235  052262  005502                        ADC     R2                      ;TO MAP DATA
10236  052264  010124                        MOV     R1,(R4)+                ;LOAD THE
10237  052266  010224                        MOV     R2,(R4)+                ;SECOND MAP REGISTER
10238  052270  104414                        RESREG
10239  052272  042637  177776                BIC     (SP)+,@#PSW             ;RETURN PRIORITY TO ORIGINAL VALUE
10240  052276  011666  000004                MOV     (SP),4(SP)              ;SETUP RETURN PC
10241  052302  022626                        CMP     (SP)+,(SP)+             ;CLEAN UP THE STACK
10242  052304  000207                        RTS     PC                      ;RETURN
10243                          ;REGISTER PAIR IS IN USE, TRY ANOTHER RANDOM NUMBER
10244  052306  062701  000004        5$:     ADD     #4,R1                   ;RESTORE R1
10245  052312  000137  052110                JMP     @#2$                    ;GET ANOTHER RANDOM NUMBER
10246                          ;;*****************************************************************
10247                          .SBTTL  GIVE MAP SUBROUTINE
10248                          ;*      THIS ROUTINE TAKES THE MAP ADDRESS OUT OF THE MAP TABLE
10249                          ;*      FOR THE REQUESTING DEVICE AND REPLACES IT WITH 377.
10250                          ;;*****************************************************************
10251  052316  010046        GIVEMAP:MOV     R0,-(SP)                ;SAVE R0
10252  052320  016600  000004                MOV     4(SP),R0                ;GET DEVICE ID
10253  052324  112760  000377  001656        MOVB    #377,MAPTBL(R0)         ;TAKE IT OUT OF THE TABLE
10254  052332  012600                        MOV     (SP)+,R0                ;RESTORE R0
10255  052334  000207                        RTS     PC                      ;RETURN
10256
10257                          ;;*****************************************************************
10258                          .SBTTL  ROUTINE TO CLEAR 'T' BIT
10259                          ;;*****************************************************************
10260  052336  013746  177776        CLRTBIT:MOV     @#PSW,-(SP)             ;PUSH PSW ONTO STACK
10261  052342  011627                        MOV     (SP),(PC)+              ;SAVE IN RETPSW BELOW
10262  052344  000000                RETPSW: .WORD   0
10263  052346  042716  000020                BIC     #20,(SP)                ;CLEAR T BIT IN PSW ON STACK
10264                          ;;*****************************************************************
10265                          .SBTTL  ROUTINE TO RESTORE THE T BIT
10266                          ;;*****************************************************************
10267  052352  012746  052360        RESPSW: MOV     #1$,-(SP)               ;SET RETURN PC FOR RTI
10268  052356  000002                        RTI                             ;CLEAR 'T' BIT IN PSW
10269  052360  000207                1$:     RTS     PC                      ;RETURN
10270
10271  052362  042737  177400  177776  RESTPS: BIC     #177400,@#PSW   ;SET KERNEL MODE
10272  052370  016746  177750                MOV     RETPSW,-(SP)    ;PUSH ORIG PSW ONTO STACK
10273  052374  000766                        BR      RESPSW
10274
10275                          ;;*****************************************************************
10276                          .SBTTL  KEYBOARD INT SERV ROUTINE
10277                          ;*THIS ROUTINE HANDLES INTERRUPTS FROM THE KEYBOARD
10278                          ;*
10279                          ;*TYPING A CONTROL 'C' WILL CAUSE THE PROCESSOR TO  HALT
10280                          ;*
10281                          ;*TYPING A CARRAGE RETURN WILL CAUSE A CARRIAGE RETURN-LINE FEED
```

```
10282                                   ;*TO BE TYPED.
10283                                   ;*
10284                                   ;*TYPING A CONTROL 'O' WILL INHIBIT ANY FURTHER TYPEOUT. THE SECOND CONTROL 'O'
10285                                   ;*WILL ENABLE TYPEOUT AGAIN AND ECHO A CR-LF.
10286                                   ;*
10287                                   ;*ANY OTHER CHARACTER WILL JUST BE ECHOED.
10288                                   ;;*****************************************************************
10289
10290           000003                          CNTRLC=3
10291           000017                          CNTRLO=17
10292
10293  052376   017746  126640         TKISR:  MOV     @$TKB,-(SP)             ;GET CHARACTER
10294  052402   042716  177600                 BIC     #177600,(SP)            ;STRIP UNUSED BITS
10295  052406   022716  000003                 CMP     #CNTRLC,(SP)            ;BRANCH IF NOT CONTROL C (↑C)
10296  052412   001010                        BNE     1$
10297  052414   012737  001326  001270        MOV     #$CRLF-1,@#$REGS        ;ECHO CR LF
10298  052422   106277  126616                ASRB    @$TPS
10299  052426   005726                        TST     (SP)+                   ;POP CHARACTER OFF THE STACK
10300  052430   000000                        HALT
10301  052432   000002                        RTI                             ;RETURN
10302
10303  052434   122716  000015         1$:     CMPB    #15,(SP)                ;BRANCH IF NOT <CR>
10304  052440   001007                        BNE     2$
10305  052442   012737  001326  001270        MOV     #$CRLF-1,@#$REGS        ;ECHO CR LF
10306  052450   106277  126570                ASRB    @$TPS
10307  052454   005726                        TST     (SP)+                   ;POP CHARACTEROFF STACK
10308  052456   000002                        RTI                             ;RETURN
10309
10310  052460   122716  000017         2$:     CMPB    #CNTRLO,(SP)            ;BRANCH IF NOT CONTROL O (↑O)
10311  052464   001012                        BNE     3$
10312  052466   005726                        TST     (SP)+
10313  052470   005167  126772                COM     NOTYPE
10314  052474   100405                        BMI     7$
10315  052476   012737  001326  001270        MOV     #$CRLF-1,@#$REGS        ;ECHO CR LF
10316  052504   106277  126534                ASRB    @$TPS
10317  052510   000002                 7$:     RTI
10318
10319  052512   104412                 3$:     SAVREG
10320  052514   011605                        MOV     (SP),R5         ;RETRIEVE CHARACTER
10321  052516   004737  053126                JSR     PC,@#LDKT               ;GO TO LOW CORE
10322  052522   013700  001442                MOV     @#TKBFRP,R0             ;GET BUFFER PTR
10323  052526   110520                 4$:     MOVB    R5,(R0)+                ;LOAD CHAR INTO BFR
10324  052530   105010                        CLRB    (R0)                    ;CLEAR NEXT LOC
10325  052532   022700  001464         5$:     CMP     #TKBFR+20,R0            ;BRANCH IF NOT END OF BFR
10326  052536   001002                        BNE     6$
10327  052540   012700  001444                MOV     #TKBFR,R0               ;RESET BUFFER PTR
10328  052544   010037  001442         6$:     MOV     R0,@#TKBFRP             ;RESTORE BFR PTR
10329  052550   004737  053214                JSR     PC,@#RESKT              ;GO BACK TO ORIGINAL MEMORY
10330  052554   104414                        RESREG
10331  052556   005737  001466         ECHO:   TST     @#NOTYPE                ;TYPEOUT DISABLED?
10332  052562   100004                        BPL     1$                      ;BRANCH IF NO
10333  052564   005726                        TST     (SP)+                   ;FIX UP STACK
10334  052566   105077  126452                CLRB    @$TPS                   ;CLEAR IE BIT
10335  052572   000002                        RTI                             ;RETURN
10336  052574   105777  126444         1$:     TSTB    @$TPS           ;PRINTER READY?
10337  052600   100375                        BPL     .-4                     ;BRANCH IF NO
```

```
10338  052602  112677  126440                  MOVB    (SP)+,@$TPB             ;MOVE CHAR TO PRINTER
10339  052606  000002                          RTI                             ;RETURN
10340
10341
10342                                  ;;*****************************************************************
10343                                  .SBTTL  TELETYPE INTERRUPT SERVICE ROUTINE
10344                                  ;*THIS ROUTINE TYPES A MESSAGE POINTED TO BY THE ADR STORED
10345                                  ;*IN LOCATION $REG5. THIS ROUTINE IS INTERRUPT DRIVEN.
10346                                  ;;*****************************************************************
10347  052610  005237  001270          TPISR:  INC     @#$REG5         ;STEP MESSAGE ADDRESS PTR
10348  052614  117746  126450                  MOVB    @$REG5,-(SP)            ;GET CHAR TO BE TYPED
10349  052620  001356                          BNE     ECHO                    ;GO TYPE CHAR IF NOT '0'
10350  052622  005726                          TST     (SP)+                   ;POP STACK
10351  052624  005077  126414                  CLR     @$TPS                   ;CLEAR IE BIT
10352  052630  012737  001526  001270          MOV     #NULLS,@#$REG5
10353  052636  000002                          RTI                             ;RETURN
10354
10355                                  ;;*****************************************************************
10356                                  .SBTTL  PARITY ERROR SERVICE
10357                                  ;*      THIS ROUTINE FIELDS UNEXPECTED TRAPS TO 114. IT IS ASSUMED
10358                                  ;*      THAT THE ERROR WAS IN CACHE AND WAS CAUSED BY THE "OTHER
10359                                  ;*      WORD" RATHER THAN THE "WANTED WORD" WHICH MEANS THAT THE
10360                                  ;*      BAD DATA IS STILL IN THE CACHE. SO, TO CLEAR THE BAD DATA
10361                                  ;*      THE ERROR ADDRESS IS REFERENCED CAUSING THE CACHE TO GO
10362                                  ;*      TO MAIN MEMORY TO GET THE DATA. THIS PREVENTS AN
10363                                  ;*      ARBITRARY REFERENCE TO THE BAD WORD FROM TRAPPING.
10364                                  ;*
10365                                  ;*      AFTER THE ERROR IS REPORTED, BITS 2 AND 3 OF THE MEMORY
10366                                  ;*      ERROR REGISTER ARE TESTED TO SEE IF THE BAD DATA IS IN
10367                                  ;*      MAIN MEMORY. IF IT IS, THE PROGRAM RESTARTS SINCE THE
10368                                  ;*      GOOD DATA IS NOW LOST FOREVER. OTHERWISE THE PROGRAM
10369                                  ;*      RETURNS TO THE ADDRESS POINTED TO BY "$LPERR".
10370                                  ;;*****************************************************************
10371  052640  012737  053120  000114  .PARSRV:MOV     #RT1,@#CACHVEC  ;PUT NEW ADDRESS IN PARITY VECTOR
10372  052646  016637  000002  001276          MOV     2(SP),@#$TMP0   ;SAVER ERROR PSW
10373  052654  011637  001474                  MOV     (SP),@#VADR     ;SAVE PC
10374  052660  162737  000002  001474          SUB     #2,@#VADR       ;ADJUST ERROR PC
10375  052666  013702  177744                  MOV     @#MEMERR,R2     ;GET ERROR REGISTER
10376  052672  013703  177740                  MOV     @#LOADRS,R3     ;GET LO ADDRESS ERROR REG
10377  052676  010337  001304                  MOV     R3,@#$TMP3      ;PUT LOW ADR IN MEMORY
10378  052702  013737  177742  001306          MOV     @#HIADRS,@#$TMP4        ;GET HI ADDRESS ERROR REG
10379  052710  042703  176000                  BIC     #176000,R3      ;MASK OFF LOWER TEN BITS
10380  052714  013704  172354                  MOV     @#KIPAR6,R4     ;SAVE PAR6
10381  052720  105737  001503                  TSTB    @#MMON          ;IS MEMORY MGMT ON?
10382  052724  001407                          BEQ     1$              ;BRANCH IF NO
10383  052726  005037  172354                  CLR     @#KIPAR6        ;CLEAR PAR6
10384  052732  012737  077406  172314          MOV     #77406,@#KIPDR6 ;ENSURE PDR 6 RESIDENT
10385  052740  052703  140000                  BIS     #140000,R3      ;SETUP R3 TO REFERENCE THRU PAR6
10386  052744  105713                  1$:     TSTB    (R3)            ;REFERENCE ADDRESS THAT TRAPPED
10387                                                                  ;SHOULD CAUSE ABORT
10388  052746  005102                  2$:     COM     R2              ;GET ORIGINAL MEMORY
10389  052750  010237  177744                  MOV     R2,@#MEMERR     ;ERROR REG DATA
10390  052754  013737  177744  001302  PERET:  MOV     @#MEMERR,@#$TMP2        ;SAVE ERROR REG FOR TYPEOUT
10391  052762  013737  001212  001266          MOV     @#$LPERR,@#$REG4        ;SAVE LOOP ADDRESS
10392  052770  012737  053000  001212          MOV     #2$,@#$LPERR            ;SET RETURN ADDRESS IF LOOPING
10393  052776  104004                          ERROR   4
```

```
10394  053000  013737  001266  001212  2$:     MOV     @#$REG4,@#$LPERR        ;RESTORE LOOP ADDRESS
10395  053006  010437  172354                  MOV     R4,@#KIPAR6     ;RESTORE PAR6
10396  053012  013704  177744                  MOV     @#MEMERR,R4     ;GET MEM ERR REG
10397  053016  012737  177777  177744          MOV     #-1,@#MEMERR            ;CLEAR ERR REG
10398  053024  012737  052640  000114          MOV     #.PARSRV,@#CACHVEC      ;RESTORE PARITY VECTOR
10399  053032  042704  177763                  BIC     #177763,R4      ;CLEAR ALL BUT BITS 2 & 3
10400  053036  001426                          BEQ     1$              ;BRANCH IF NOT MAIN MEMORY ERROR
10401  053040  104400  053046                  TYPE    ,65$            ;;TYPE ASCIZ STRING
10402  053044  000420                          BR      64$             ;;GET OVER THE ASCIZ
10403                          ;;65$:  .ASCIZ  /FATAL PARITY ERROR-RESTARTING/<CRLF>
10404  053106                  64$:
10405  053106  000005                          RESET                   ;CLEAR THE WORLD
10406  053110  000137  003212                  JMP     @#START
10407  053114  012716  053122          1$:     MOV     #X,(SP)                 ;PUT ADDRESS ON STACK TO GET ORIGINAL
10408                                                                  ;PSW BACK
10409  053120  000002                  RTI:    RTI                     ;GET OLD PSW
10410  053122  000177  126064          X:      JMP     @$LPERR                 ;JUMP TO START OF TEST THAT HAD THE PE
10411                          ;;**************************************************************
10412                          .SBTTL  CONTEXT SWITCH DOWN SUBROUTINE
10413                          ;*      SUBROUTINE TO SAVE & LOAD KIPAR'S 0,1,AND 2 (IF MEM MGMT ENABLED)
10414                          ;*      THIS ROUTINE IS CALLED BY THE KEYBOARD INTERRUPT, LINE CLOCK
10415                          ;*      INTERRUPT, UBE SERVICE ROUTINE, MBT SERVICE ROUTINE, AND TYPE TIME ROUTINE.
10416                          ;;**************************************************************
10417  053126  105737  001503          LDKT:   TSTB    @#MMON                  ;BRANCH IF MEM MGMT DISABLED
10418  053132  001427                          BEQ     1$
10419  053134  012604                          MOV     (SP)+,R4        ;SAVE RETURN PC
10420  053136  013737  177776  053270          MOV     @#PSW,@#$SAVPSW ;SAVE THE CURRENT PSW
10421  053144  042737  140000  177776          BIC     #140000,@#PSW   ;GO TO KERNEL MODE
10422  053152  012700  172340                  MOV     #KIPAR0,R0      ;GET ADDRESS OF PAR0
10423  053156  012001                          MOV     (R0)+,R1        ;GET PAR0
10424  053160  012002                          MOV     (R0)+,R2        ;GET PAR1
10425  053162  012003                          MOV     (R0)+,R3        ;GET PAR2
10426  053164  012740  000400                  MOV     #400,-(R0)      ;RELOC BACK TO LOW CORE
10427  053170  012740  000200                  MOV     #200,-(R0)
10428  053174  005040                          CLR     -(R0)
10429  053176  012700  053262                  MOV     #$SAVPAR,R0     ;GET ADDRESS OF SAVE BUFFER
10430  053202  010120                          MOV     R1,(R0)+        ;PUT PAR DATA IN MEMORY
10431  053204  010220                          MOV     R2,(R0)+
10432  053206  010320                          MOV     R3,(R0)+
10433  053210  010446                          MOV     R4,-(SP)        ;PUT RETURN PC ON STACK
10434  053212  000207                  1$:     RTS     PC
10435
10436                          ;;**************************************************************
10437                          .SBTTL  CONTEXT SWITCH UP SUBROUTINE
10438                          ;SUBROUTINE TO RESTORE KIPAR0, 1, AND 2 (IF MGMT ENABLED)
10439                          ;;**************************************************************
10440  053214  105737  001503          RESKT:  TSTB    @#MMON          ;BRANCH IF MEM MGMT DISABLED
10441  053220  001417                          BEQ     1$
10442  053222  012604                          MOV     (SP)+,R4        ;GET RETURN PC
10443  053224  012700  053262                  MOV     #$SAVPAR,R0     ;GET ADDRESS OF SAVE BUFF
10444  053230  012001                          MOV     (R0)+,R1        ;GET OLD PAR DATA
10445  053232  012002                          MOV     (R0)+,R2
10446  053234  012003                          MOV     (R0)+,R3
10447  053236  012700  172340                  MOV     #KIPAR0,R0      ;GET ADDRESS OF PAR0
10448  053242  010120                          MOV     R1,(R0)+        ;RELOCATE BACK
10449  053244  010220                          MOV     R2,(R0)+
```

```
10450  053246  010310                        MOV     R3,(R0)
10451  053250  013737  053270  177776        MOV     @#$SAVPSW,@#PSW
10452  053256  010446                        MOV     R4,-(SP)
10453  053260  000207                1$:     RTS     PC
10454  053262  000003                $SAVPAR:.BLKW   3
10455  053270  000000                $SAVPSW:.WORD
10456                                 ;;*****************************************************************
10457                                 .SBTTL  KT ABORT SUBROUTINE
10458                                 ;;*****************************************************************
10459  053272  016637  000002  001276  KTABRT: MOV     2(SP),@#$TMP0           ;SAVE ERROR PSW
10460  053300  011637  001474                MOV     (SP),@#VADR             ;SAVE ERROR PC
10461  053304  162737  000002  001474        SUB     #2,@#VADR
10462  053312  013737  177572  001302        MOV     @#MMR0,@#$TMP2          ;SAVE MMR0
10463  053320  013737  177576  001304        MOV     @#MMR2,@#$TMP3          ;SAVE MMR2
10464  053326  013737  001212  001266        MOV     @#$LPERR,@#$REG4        ;SAVE LOOP ADDRESS
10465  053334  012737  053344  001212        MOV     #1$,@#$LPERR            ;SET RETURN ADR IF LOOPING
10466  053342  104003                        ERROR   3
10467  053344  013737  001266  001212  1$:   MOV     @#$REG4,@#$LPERR        ;RESTORE LOOP ADR
10468  053352  042737  170000  177572        BIC     #170000,@#MMR0          ;CLEAR ERRORS
10469  053360  013716  001212                MOV     @#$LPERR,(SP)           ;GET LOOP ADDRESS
10470  053364  000002                        RTI                             ;RETURN
10471
10472                                 ;;*****************************************************************
10473                                 .SBTTL  RESERVED INSTRUCTION ROUTINE
10474                                 ;;*****************************************************************
10475  053366  016637  000002  001276  RESERR: MOV     2(SP),@#$TMP0           ;SAVE PSW
10476  053374  011637  001474                MOV     (SP),@#VADR             ;SAVE ERROR PC
10477  053400  162737  000002  001474        SUB     #2,@#VADR
10478  053406  013737  001212  001266        MOV     @#$LPERR,@#$REG4        ;SAVE LOOP ADR
10479  053414  012737  053424  001212        MOV     #1$,@#$LPERR            ;SET RETURN ADR IF LOOPING
10480  053422  104002                        ERROR   2
10481  053424  013737  001266  001212  1$:   MOV     @#$REG4,@#$LPERR        ;RESTORE LOOP ADR
10482  053432  013716  001212                MOV     @#$LPERR,(SP)           ;GET LOOP ADDRESS
10483  053436  000002                        RTI                             ;RETURN
10484
10485                                 ;;*****************************************************************
10486                                 .SBTTL  TRAP TO 4 SERVICE ROUTINE
10487                                 ;;*****************************************************************
10488  053440  016637  000002  001276  ERPRT:  MOV     2(SP),@#$TMP0           ;SAVE ERROR PSW
10489  053446  011637  001474                MOV     (SP),@#VADR             ;SAVE ERROR PC
10490  053452  162737  000002  001474        SUB     #2,@#VADR
10491  053460  012706  000700                MOV     #SUPSTK,SP              ;RESTORE SP
10492  053464  013737  177766  001302        MOV     @#CPUERR,@#$TMP2        ;GET ERROR REG
10493  053472  013737  001212  001266        MOV     @#$LPERR,@#$REG4        ;SAVE LOOP ADR
10494  053500  012737  053510  001212        MOV     #1$,@#$LPERR            ;SET RETURN ADR IF LOOPING
10495  053506  104001                        ERROR   1
10496  053510  013737  001266  001212  1$:   MOV     @#$REG4,@#$LPERR        ;SET LOOP ADR
10497  053516  005037  177766                CLR     @#CPUERR
10498  053522  013746  001276                MOV     @#$TMP0,-(SP)   ;SETUP STACK TO RETURN
10499  053526  013746  001212                MOV     @#$LPERR,-(SP)
10500  053532  000002                        RTI                             ;RETURN
10501
```

```
10502                                   ;THE BELOW TABLE REPRESENTS THE 'NEW' PSW SET BY THE PROGRAM ON
10503                                   ;SUCCESSIVE SUB-PASSES.
10504                                   ;NOTE THE BELOW TABLE MAY BE MODIFIED TO CAUSE THE PROGRAM TO RUN
10505                                   ;UNDER USER DEFINED PARAMETERS BY PATCHING IN THE DESIRED PASS PARAMETER
10506                                   ;FOR EXAMPLE TO CAUSE THE PROGRAM TO RUN WITHOUT SETTING THE 'T' BIT
10507                                   ;IN ALL PASSES PATCH OUT THE 'T' BIT IN THE TABLE.
10508   053534  000000            PSWTAB: 000000                  ;
10509   053536  000020                    000020                  ;T-BIT TRAPPING
10510   053540  140000                    140000                  ;USER MODE
10511   053542  144020                    144020                  ;USER MODE, REG SET #1, T-BIT TRAPPING
10512   053544  040000                    040000                  ;SUPERVISOR MODE
10513   053546  044020                    044020                  ;SUPERVISOR MODE, REG SET #1, T-BIT TRAPPING
10514
10515                                   ;THE BELOW TABLE IS USED TO SET MEMORY MARGINS
10516   053550  000000            MRGTAB: .WORD   0                       ;NO MARGINS
10517   053552  000004                    .WORD   4                       ;EARLY STROBE
10518   053554  000006                    .WORD   6                       ;LATE STROBE
10519   053556  000010                    .WORD   10                      ;LOW DRIVE CURRENT
10520   053560  000000                    .WORD   0                       ;NO MARGINS
10521   053562  000012                    .WORD   12                      ;HIGH DRIVE CURRENT
10522                                   ;MESSAGES
10523                                           .EVEN
10524   053564  002100            REGINX: RP3DS
10525   053566  002122                    RKDS
10526   053570  000000                    .WORD
10527   053572  000000                    .WORD
10528   053574  002142                    RP4CS1
10529   053576  002202                    RSCS1
10530   053600  000000                    .WORD
10531   053602  000000                    .WORD
10532   053604  002246                    MBTTBL
10533   053606  002230                    UBETBL
10534   053610  054064            MSGINX: .WORD   MSG5
10535   053612  054072                    .WORD   MSG6
10536   053614  054701                    .WORD   MSG21
10537   053616  054701                    .WORD   MSG21
10538   053620  054100                    .WORD   MSG10
10539   053622  054106                    .WORD   MSG11
10540   053624  054701                    .WORD   MSG21
10541   053626  054701                    .WORD   MSG21
10542   053630  054375                    .WORD   MSG15
10543   053632  054740                    .WORD   MSG24
10544   053634  046200  053517  046040 MSG1:    .ASCIZ  <CRLF>'LOW LIM?'
10545   053642  046511  000077
10546   053646  044510  044107  046040 MSG2:    .ASCIZ  'HIGH LIM?'
10547   053654  046511  000077
10548   053660  051105  047522  050122 MSG3:    .ASCIZ  /ERRORPC PHYSC PC    PSW   MAINT   TEST NO SUB-PASS CNT/
10549   053666  020103  044120  051531
10550   053674  020103  041520  020040
10551   053702  020040  051520  020127
10552   053710  020040  040515  047111
10553   053716  020124  020040  042524
10554   053724  052123  047040  020117
10555   053732  052523  026502  040520
10556   053740  051523  041440  052116
10557   053746     000
```

```
10558  053747     124  042510  043040  MSG4:   .ASCII  /THE FOLLOWING DEVICES AND DRIVES WILL BE USED FOR RELOCATION:/<CRLF>
10559  053754  046117  047514  044527
10560  053762  043516  042040  053105
10561  053770  041511  051505  040440
10562  053776  042116  042040  044522
10563  054004  042526  020123  044527
10564  054012  046114  041040  020105
10565  054020  051525  042105  043040
10566  054026  051117  051040  046105
10567  054034  041517  052101  047511
10568  054042  035116     200
10569  054045     104  053105  041511          .ASCIZ  /DEVICE DRIVES/<CRLF>
10570  054052  004505  051104  053111
10571  054060  051505  000200
10572  054064  050122  031460  000011  MSG5:   .ASCIZ  ?RP03   ?
10573  054072  045522  032460  000011  MSG6:   .ASCIZ  ?RK05   ?
10574  054100  050122  032060  000011  MSG10:  .ASCIZ  ?RP04   ?
10575  054106  051522  032060  000011  MSG11:  .ASCIZ  ?RS04   ?
10576  054114  051104  051526  040524  MSG12:  .ASCIZ  /DRVSTA  ERRREG  CSREG   WRDCNT  BUSADR  DSKADR  CYLADR(RP03)  PHYS BUSA
10577  054122  020040  051105  051122
10578  054130  043505  020040  051503
10579  054136  042522  020107  020040
10580  054144  051127  041504  052116
10581  054152  020040  052502  040523
10582  054160  051104  020040  051504
10583  054166  040513  051104  020040
10584  054174  054503  040514  051104
10585  054202  051050  030120  024463
10586  054210  020040  044120  051531
10587  054216  041040  051525  042101
10588  054224  100122     000
10589  054227     040  051503  020061  MSG13:  .ASCIZ  / CS1    WRDCNT  BUSADR  BADREX  DSKADR   CS2     CS3    DRVSTA  ERRREG/
10590  054234  020040  053440  042122
10591  054242  047103  020124  041040
10592  054250  051525  042101  020122
10593  054256  041040  042101  042522
10594  054264  020130  042040  045523
10595  054272  042101  020122  020040
10596  054300  051503  020062  020040
10597  054306  020040  051503  020063
10598  054314  020040  042040  053122
10599  054322  052123  020101  042440
10600  054330  051122  042522  100107
10601  054336     000
10602  054337     104  051505  054503  MSG14:  .ASCIZ  /DESCYL   ER2     ER3    RPCC/<CRLF>
10603  054344  020114  020040  051105
10604  054352  020062  020040  020040
10605  054360  051105  020063  020040
10606  054366  051040  041520  100103
10607  054374     000
10608  054375     115  051501  020123  MSG15:  .ASCIZ  /MASS BUS TESTER /
10609  054402  052502  020123  042524
10610  054410  052123  051105  000040
10611  054416  041440  030523  020040  MSG16:  .ASCIZ  / CS1    WRDCNT  BUSADR  BADREX    MR2     CS2      ST     ER      CS3/<
10612  054424  020040  051127  041504
10613  054432  052116  020040  052502
```

```
10614   054440  040523  051104  020040
10615   054446  040502  051104  054105
10616   054454  020040  020040  051115
10617   054462  020062  020040  020040
10618   054470  051503  020062  020040
10619   054476  020040  051440  020124
10620   054504  020040  020040  051105
10621   054512  020040  020040  020040
10622   054520  051503  100063     000
10623   054525     040  041440  020103  MSG17:  .ASCIZ  /  CC    BUSADR    CR2     CR1  PHYS BUSADR/<CRLF>
10624   054532  020040  041040  051525
10625   054540  042101  020122  020040
10626   054546  041440  031122  020040
10627   054554  020040  041440  030522
10628   054562  020040  044120  051531
10629   054570  041040  051525  042101
10630   054576  100122     000
10631   054601     124  042510  050440  MSG20:  .ASCIZ  /THE QUICK BROWN FOX JUMPED OVER THE LAZY DOGS BACK 0123456789/<15><12>
10632   054606  044525  045503  041040
10633   054614  047522  047127  043040
10634   054622  054117  045040  046525
10635   054630  042520  020104  053117
10636   054636  051105  052040  042510
10637   054644  046040  055101  020131
10638   054652  047504  051507  041040
10639   054660  041501  020113  030460
10640   054666  031462  032464  033466
10641   054674  034470  005015     000
10642   054701     111  046114  043505  MSG21:  .ASCIZ  /ILLEGAL DEVICE/<CRLF>
10643   054706  046101  042040  053105
10644   054714  041511  100105     000
10645   054721     040  020040  020040  MSG22:  .ASCII  /       /
10646   054726  020040     040
10647   054731     040  020040  020040  MSG23:  .ASCIZ  /     /
10648   054736  000040
10649   054740  047125  041111  051525  MSG24:  .ASCIZ  /UNIBUS EXERCISER /
10650   054746  042440  042530  041522
10651   054754  051511  051105  000040
10652   054762  050117  027124  050103  MSG25:  .ASCIZ  /OPT.CP=/
10653   054770  000075
10654   054772  047125  054105  042520  EM1:    .ASCIZ  /UNEXPECTED TRAP TO 4/
10655   055000  052103  042105  052040
10656   055006  040522  020120  047524
10657   055014  032040     000
10658   055017     120  047503  052106  DH1:    .ASCIZ  /PCOFTP  PHYSPC    PSW   CPUERR/
10659   055024  020120  050040  054510
10660   055032  050123  020103  020040
10661   055040  050040  053523  020040
10662   055046  041440  052520  051105
10663   055054  000122
10664   055056     000     001     000  DF1:    .BYTE   0,1,0,0,0
10665   055061     000     000
10666           055064                  .EVEN
10667   055064  001474  001474  001276  DT1:    .WORD   VADR,VADR,$TMP0,$TMP2,0
10668   055072  001302  000000
10669   055076  047125  054105  042520  EM2:    .ASCIZ  /UNEXPECTED TRAP TO 10/
```

```
10670  055104  052103  042105  052040
10671  055112  040522  020120  047524
10672  055120  030440  000060
10673  055124  041520  043117  050124  DH2:    .ASCIZ  /PCOFTP  PHYSPC    PSW/
10674  055132  020040  044120  051531
10675  055140  041520  020040  020040
10676  055146  051520  000127
10677                                  .EVEN
10678  055152  001474  001474  001276  DT2:    .WORD   VADR,VADR,$TMP0,0
10679  055160  000000
10680  055162  047125  054105  042520  EM3:    .ASCIZ  /UNEXPECTED TRAP TO 250(MGMT)/
10681  055170  052103  042105  052040
10682  055176  040522  020120  047524
10683  055204  031040  030065  046450
10684  055212  046507  024524     000
10685  055217     120  047503  052106  DH3:    .ASCIZ  /PCOFTP  PHYSPC    PSW    MMR0    MMR2/
10686  055224  020120  050040  054510
10687  055232  050123  020103  020040
10688  055240  050040  053523  020040
10689  055246  020040  046515  030122
10690  055254  020040  020040  046515
10691  055262  031122     000
10692          055266                  .EVEN
10693  055266  001474  001474  001276  DT3:    .WORD   VADR,VADR,$TMP0,$TMP2,$TMP3,0
10694  055274  001302  001304  000000
10695  055302  047125  054105  042520  EM4:    .ASCIZ  /UNEXPECTED TRAP TO 114/
10696  055310  052103  042105  052040
10697  055316  040522  020120  047524
10698  055324  030440  032061     000
10699  055331     120  047503  052106  DH4:    .ASCIZ  /PCOFTP  PHYSC PC    PSW  ERRREG  ERR ADR REG/
10700  055336  020120  050040  054510
10701  055344  041523  050040  020103
10702  055352  020040  050040  053523
10703  055360  020040  042440  051122
10704  055366  042522  020107  042440
10705  055374  051122  040440  051104
10706  055402  051040  043505     000
10707  055407     000     001     000  DF4:    .BYTE   0,1,0,0,2
10708  055412     000     002
10709  055414  040520  044522  054524  EM5:    .ASCIZ  /PARITY ERROR DURING DATA CHECK/
10710  055422  042440  051122  051117
10711  055430  042040  051125  047111
10712  055436  020107  040504  040524
10713  055444  044440  042510  045503
10714  055452     000
10715  055453     123  041522  042101  DH5:    .ASCIZ  /SRCADR  DSTADR  EADRREG  MEM ERR REG/
10716  055460  020122  042040  052123
10717  055466  042101  020122  042440
10718  055474  042101  051122  043505
10719  055502  020040  042515  020115
10720  055510  051105  020122  042522
10721  055516  000107
10722  055520     000     001     002  DF5:    .BYTE   0,1,2,0
10723  055523     000
10724                                  .EVEN
10725  055524  001276  001474  001304  DT5:    .WORD   $TMP0,VADR,$TMP3,$TMP2,0
```

```
10726  055532  001302  000000
10727  055536  051105  047522  020122  EM6:    .ASCIZ  /ERROR DURING DATA CHECK-RELOC WAS BY CP/
10728  055544  052504  044522  043516
10729  055552  042040  052101  020101
10730  055560  044103  041505  026513
10731  055566  042522  047514  020103
10732  055574  040527  020123  054502
10733  055602  041440  000120
10734  055606  051123  040503  051104  DH6:    .ASCIZ  /SRCADR  DSTADR/
10735  055614  020040  051504  040524
10736  055622  051104     000
10737          055626                  .EVEN
10738  055626  001276  001474  000000  DT6:    .WORD   $TMP0,VADR,0
10739  055634  051105  047522  020122  EM10:   .ASCIZ  ?ERROR DURING DATA CHECK-RELOC WAS BY I/O?
10740  055642  052504  044522  043516
10741  055650  042040  052101  020101
10742  055656  044103  041505  026513
10743  055664  042522  047514  020103
10744  055672  040527  020123  054502
10745  055700  044440  047457     000
10746  055705     123  041522  042101  DH10:   .ASCIZ  /SRCADR   DSTADR   DEVICE THAT DID XFER/
10747  055712  020122  020040  051504
10748  055720  040524  051104  020040
10749  055726  042040  053105  041511
10750  055734  020105  044124  052101
10751  055742  042040  042111  054040
10752  055750  042506  000122
10753  055754     000     001     003  DF10:   .BYTE   0,1,3,0
10754  055757     000
10755                                  .EVEN
10756  055760  001276  001474  001302  DT10:   .WORD   $TMP0,VADR,$TMP2,$TMP3,0
10757  055766  001304  000000
10758  055772  044502  024124  024523  EM11:   .ASCIZ  /BIT(S) STUCK IN MICRO-BREAK REGISTER/
10759  056000  051440  052524  045503
10760  056006  044440  020116  044515
10761  056014  051103  026517  051102
10762  056022  040505  020113  042522
10763  056030  044507  052123  051105
10764  056036     000
10765  056037     107  047517  042104  DH11:   .ASCIZ  /GOODDAT BAD DATA/
10766  056044  052101  041040  042101
10767  056052  042040  052101  000101
10768  056060     000     000          DF11:   .BYTE   0,0
10769                                  .EVEN
10770  056062  001276  001300  000000  DT11:   .WORD   $TMP0,$TMP1,0
10771  056070  041125  020105  047516  EM12:   .ASCIZ  /UBE NON-EXISTANT MEMORY ERROR/
10772  056076  026516  054105  051511
10773  056104  040524  052116  046440
10774  056112  046505  051117  020131
10775  056120  051105  047522  000122
10776  056126  044120  051531  041040  DH12:   .ASCIZ  /PHYS BUSADR/
10777  056134  051525  042101  000122
10778  056142     002          DF12:   .BYTE   2
10779          056144                  .EVEN
10780  056144  001226  000000  DT12:   .WORD   $GDDAT,0
10781  056150  041115  020124  047516  EM13:   .ASCIZ  /MBT NON-EXISTANT MEMORY ERROR/
```

```
10782  056156  026516  054105  051511
10783  056164  040524  052116  046440
10784  056172  046505  051117  020131
10785  056200  051105  047522  000122
10786  056206  044120  051531  040440  DH13:   .ASCIZ  /PHYS ADDRESS/
10787  056214  042104  042522  051523
10788  056222     000
10789  056223     106  047514  052101  EM14:   .ASCIZ  /FLOATING POINT ERROR/
10790  056230  047111  020107  047520
10791  056236  047111  020124  051105
10792  056244  047522  000122
10793  056250  042011  040524  030524  DH14:   .ASCIZ  /      DTAT1                              DATA2/
10794  056256  004411  004411  040504
10795  056264  040524  000062
10796                                  .EVEN
10797  056270  001306  001262  001312  DT14:   .WORD   $TMP4,$REG2,$TMP6,$REG3,0
10798  056276  001264  000000
10799  056302     004     000     004  DF14:   .BYTE   4,0,4,0
10800  056305     000
10801  056306  042504  044526  042503  EM15:   .ASCIZ  /DEVICE HUNG/
10802  056314  044040  047125  000107
10803  056322  004411  040504  040524  DH16:   .ASCIZ  /                DATA1                                DATA2/
10804  056330  004461  004411  020011
10805  056336  020040  042040  052101
10806  056344  031101     000
10807  056347     005     000     005  DF16:   .BYTE   5,0,5,0
10808  056352     000
10809          056354                  .EVEN
10810  056354  001422  001262  001432  DT16:   .WORD   FLTMP0,$REG2,FLTMP1,$REG3,0
10811  056362  001264  000000
10812  056366  000000                  ENDTAG: .WORD   0
10813                                  ;;*****************************************************************
10814                                  ;THE FOLLOWING ASCII GETS OVERLAYED WHEN THE PROGRAM RUNS.
10815  056370  050117  051105  052101  SWITCH: .ASCII  /OPERATIONAL SWITCH SETTINGS/<CRLF>
10816  056376  047511  040516  020114
10817  056404  053523  052111  044103
10818  056412  051440  052105  044524
10819  056420  043516  100123
10820  056424  053523  052111  044103          .ASCII  /SWITCH                  USE/<CRLF>
10821  056432  004411  052411  042523
10822  056440     200
10823  056441     040  030440  004465          .ASCII  /  15           HALT ON ERROR/<CRLF>
10824  056446  044011  046101  020124
10825  056454  047117  042440  051122
10826  056462  051117     200
10827  056465     040  030440  004464          .ASCII  /  14           LOOP ON TEST/<CRLF>
10828  056472  046011  047517  020120
10829  056500  047117  052040  051505
10830  056506  100124
10831  056510  020040  031461  004411          .ASCII  /  13           INHIBIT ERROR TYPEOUTS/<CRLF>
10832  056516  047111  044510  044502
10833  056524  020124  051105  047522
10834  056532  020122  054524  042520
10835  056540  052517  051524     200
10836  056545     040  030440  004462          .ASCII  /  12           INHIBIT UBE/<CRLF>
10837  056552  044411  044116  041111
```

```
10838  056560  052111  052440  042502
10839  056566     200
10840  056567     040  030440  004461          .ASCII  /  11          INHIBIT ITTERATIONS/<CRLF>
10841  056574  044411  044116  041111
10842  056602  052111  044440  052124
10843  056610  051105  052101  047511
10844  056616  051516     200
10845  056621     040  030440  004460          .ASCII  /  10          BELL ON ERROR/<CRLF>
10846  056626  041011  046105  020114
10847  056634  047117  042440  051122
10848  056642  051117     200
10849  056645     040  020040  004471          .ASCII  /   9          LOOP ON ERROR/<CRLF>
10850  056652  046011  047517  020120
10851  056660  047117  042440  051122
10852  056666  051117     200
10853  056671     040  020040  004470          .ASCII  ?   8          INHIBIT RELOCATION VIA I/O DEVICE?<CRLF>
10854  056676  044411  044116  041111
10855  056704  052111  051040  046105
10856  056712  041517  052101  047511
10857  056720  020116  044526  020101
10858  056726  027511  020117  042504
10859  056734  044526  042503     200
10860  056741     040  020040  004467          .ASCII  /   7          INHIBIT TYPEOUT OF THIS TEXT AND SYS SIZE/<CRLF>
10861  056746  044411  044116  041111
10862  056754  052111  052040  050131
10863  056762  047505  052125  047440
10864  056770  020106  044124  051511
10865  056776  052040  054105  020124
10866  057004  047101  020104  054523
10867  057012  020123  044523  042532
10868  057020     200
10869  057021     040  020040  004466          .ASCII  /   6          INHIBIT RELOCATION/<CRLF>
10870  057026  044411  044116  041111
10871  057034  052111  051040  046105
10872  057042  041517  052101  047511
10873  057050  100116
10874  057052  020040  032440  004411          .ASCII  /   5          INHIBIT ROUND ROBIN RELOCATION/<CRLF>
10875  057060  047111  044510  044502
10876  057066  020124  047522  047125
10877  057074  020104  047522  044502
10878  057102  020116  042522  047514
10879  057110  040503  044524  047117
10880  057116     200
10881  057117     040  020040  004464          .ASCII  /   4          INHIBIT RANDOM DISK ADDRESS/<CRLF>
10882  057124  044411  044116  041111
10883  057132  052111  051040  047101
10884  057140  047504  020115  044504
10885  057146  045523  040440  042104
10886  057154  042522  051523     200
10887  057161     040  020040  004463          .ASCII  /   3          INHIBIT MBT/<CRLF>
10888  057166  044411  044116  041111
10889  057174  052111  046440  052102
10890  057202     200
10891  057203     040  020040  004462          .ASCII  /   2          THESE THREE SWITCHES/<CRLF>
10892  057210  052011  042510  042523
10893  057216  052040  051110  042505
```

```
10894  057224  051440  044527  041524
10895  057232  042510  100123
10896  057236  020040  030440  004411          .ASCII  /  1          ARE ENCODED TO SELECT RELOCATION/<CRLF>
10897  057244  051101  020105  047105
10898  057252  047503  042504  020104
10899  057260  047524  051440  046105
10900  057266  041505  020124  042522
10901  057274  047514  040503  044524
10902  057302  047117     200
10903  057305     040  020040  004460          .ASCII  /  0          ON THE FOLLOWING DEVICES:/<CRLF>
10904  057312  047411  020116  044124
10905  057320  020105  047506  046114
10906  057326  053517  047111  020107
10907  057334  042504  044526  042503
10908  057342  035123     200
10909  057345     011  027060  027056          .ASCII  ?      0...RP11/RP03?<CRLF>
10910  057352  050122  030461  051057
10911  057360  030120  100063
10912  057364  030411  027056  051056          .ASCII  ?      1...RK11/RK05?<CRLF>
10913  057372  030513  027461  045522
10914  057400  032460     200
10915  057403     011  027062  027056          .ASCII  ?      2...NOT USED?<CRLF>
10916  057410  047516  020124  051525
10917  057416  042105     200
10918  057421     011  027063  027056          .ASCII  ?      3...NOT USED?<CRLF>
10919  057426  047516  020124  051525
10920  057434  042105     200
10921  057437     011  027064  027056          .ASCII  ?      4...RH70/RP04?<CRLF>
10922  057444  044122  030067  051057
10923  057452  030120  100064
10924  057456  032411  027056  051056          .ASCII  ?      5...RH70/RS04 OR RS03?<CRLF>
10925  057464  033510  027460  051522
10926  057472  032060  047440  020122
10927  057500  051522  031460     200
10928  057505     011  027066  027056          .ASCII  ?      6...NOT USED?<CRLF>
10929  057512  047516  020124  051525
10930  057520  042105     200
10931  057523     011  027067  027056          .ASCIZ  ?      7...NOT USED?<CRLF>
10932  057530  047516  020124  051525
10933  057536  042105  000200
10934          000001                          .END
```

```
AA      001505          1777#   2236#
ACO     =%000000        1595#   7024*   7028*   7037*   7039    7045*   7046    7055*   7060*   7066*   7068*   7083*   7097*
                        7106*   7110*   7116*   7118    7124*   7126    7131*   7144*   7157*   7164*   7165    7193*   7197*
                        7206*   7208    7214*   7215    7224*   7229*   7235*   7237*   7252*   7266*   7275*   7279*   7285*
                        7287    7293*   7295    7300*   7313*   7326*   7333*   7334    7353    7364    7387*   7389*   7407
                        7409    9955    9960*
AC1     =%000001        1596#   7023*   7028    7031    7039*   7041    7046*   7050    7056*   7071    7096*   7101    7104*
                        7113    7118*   7121    7126*   7134    7156*   7162    7192*   7197    7200    7208*   7210    7215*
                        7219    7225*   7240    7265*   7270    7273*   7282    7287*   7290    7295*   7303    7325*   7331
                        7353*   7364*   7399*   7403*   7407*   7409*
AC2     =%000002        1597#   7041*   7068    7101*   7110    7121*   7131    7210*   7237    7270*   7279    7290*   7300
AC3     =%000003        1598#   7050*   7066    7219*   7235
AC4     =%000004        1599#
AC5     =%000005        1600#
ADCB2     011204        3398    3400#
ADCB5     012030        3628    3630#
ADCB6     012532        3784    3785    3787#
ADCB7     013454        4007    4008    4009    4011#
ADCO      007076        2797    2798    2799    2801#
ADC1      007776        3026    3027    3028    3030#
ADC2      011006        3328    3330#
ADC5      011630        3556    3557    3559#
ADC6      012334        3727    3728    3730#
ADC7      013322        3965    3966    3968#
ADDO      014150        4174    4175    4176    4178#
ADD1      014254        4211    4212    4214#
ADD1A     014500        4294    4295    4296    4298#
ADD1B     014516        4303    4304    4306#
ADD2      015140        4448    4449    4451#
ADD3      015716        4640    4642#
ADD6      016274        4741    4742    4744#
ADD7      016752        4857    4858    4859    4861#
ARBEX     033412        7447    7490#
ARBFIN    033400        7432    7454    7475    7477    7488#
ASHCLO    025244        6277#
ASHCRO    025322        6298#
ASHLO     025034        6227#
ASHL1     025632        6409#
ASHRO     025150        6254#
ASHR1     025720        6432#
ASLB1     010346        3160    3161    3163#
ASLB1A    010572        3247    3248    3250#
ASLB3     012020        3622    3623    3625#
ASL34     011310        3435    3436    3437    3439#
ASLB6     012514        3776    3777    3778    3780#
ASLB7     013552        4039    4040    4042#
ASLO      007220        2841    2842    2843    2844    2846#
ASL1      010152        3089    3090    3091    3093#
ASL3      011544        3525    3526    3528#
ASL4      011100        3360    3361    3362    3364#
ASL6      012304        3715    3716    3718#
ASL7      013150        3912    3913    3915#
ASRB1     010442        3196    3198#
ASRB1A    010456        3202    3203    3205#
ASRB2     011254        3420    3421    3423#
ASRB2A    011272        3428    3429    3431#
```

DEQKCB.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
ASRB5   011760          3603    3604    3606#
ASRB6   012632          3816    3817    3819#
ASRB7   013570          4046    4047    4049#
ASRO    007246          2855    2856    2857    2859#
ASR1    010040          3046    3047    3048    3050#
ASR2    011022          3334    3335    3337#
ASR3    011530          3519    3521#
ASR6    012166          3677    3678    3680#
ASR7    013204          3926    3927    3929#
BICB1   014700          4366    4367    4369#
BICB1A  014722          4377    4380#   4438    4439
BICO    014062          4145    4146    4147    4149#
BIC1    014376          4257    4258    4260#
BIC2    015230          4477    4478    4479    4481#
BIC3    015730          4645    4647#
BIC7    017616          5036#   5038
BINB    015454          4541    4543    4546    4548    4551    4553    4556    4559#
BINB7   017364          4971    4979#
BIN1    015060          4409    4412    4415    4418    4421    4424    4427    4432#
BISB1   014666          4361    4363#
BISO    014040          4136    4137    4139#
BISOA   014116          4163    4165#
BIS1    014364          4251    4252    4254#
BIS2    015166          4461    4463#
BIS2A   015270          4495    4496    4498#
BIS7    017556          5026#   5028
BITB1   014656          4355    4356    4358#
BITB2   015540          4581    4582    4584#
BITB3   016106          4699    4702#
BITB6   016506          4790#
BITT1   014312          4228    4229    4231#
BITT2   015254          4488    4489    4491#
BITO  = 000001          1312#   2519    2527    2529    8346    8384    8425    10217
BITOO = 000001          1302#   1312
BITO1 = 000002          1301#   1311
BITO2 = 000004          1300#   1310
BITO3 = 000010          1299#   1309
BITO4 = 000020          1298#   1308
BITO5 = 000040          1297#   1307
BITO6 = 000100          1296#   1306
BITO7 = 000200          1295#   1305
BITO8 = 000400          1294#   1304
BITO9 = 001000          1293#   1303    8949    9017
BIT1  = 000002          1311#
BIT10 = 002000          1292#   9000
BIT11 = 004000          1291#   1646    8849    8956
BIT12 = 010000          1290#   1645    2483    2503    8259
BIT13 = 020000          1289#   8239    9007
BIT14 = 040000          1288#   1644    2183    8512    8584    8607    8622    8633    8652    8683    8700    8715    8726
                        8745    8842    9939
BIT15 = 100000          1287#   7879    8356    8363    8475    8482    8602    8609    8695    8702    9890    10215
BIT2  = 000004          1310#
BIT3  = 000010          1309#   8391
BIT4  = 000020          1308#   2560    5576    6139    6768    7950    8163    8167
BIT5  = 000040          1307#   2491    2511    6935    7510    7887    7889    8097    8178    8244    8396    8432    8517
                        8553    8588    8627    8657    8687    8720    8750    10077
```

```
BIT6   = 000100          1306#   8190    8914
BIT7   = 000200          1305#   2429
BIT8   = 000400          1304#   2435    2472    2489    2509    7893    8056    8133    8206    8273    8340    8375    8418
                         8458    8495    8537    8582    8593    8620    8650    8681    8713    8743
BIT9   = 001000          1303#   7903
BPTVEC= 000014           1319#   5488    5491*   5492*   5565*   5566*
CACHVE= 000114           1326#   2573*   2575*  10147*  10156*  10371*  10398*
CALLHA= 000010           1647#   7788    7895
CBIT    023034           5743#
CC0     006704           2719    2720    2721    2722    2723    2724    2725    2727#
CC1     006720           2731    2732    2733    2735#
CC2     006734           2739    2740    2741    2743#
CC3     006746           2747    2748    2750#
CC4     006762           2754    2755    2756    2758#
CHKDAT  051744           7654    7834    7918   10147#  10155
CHKSP   022626           5677#
CLRTBI  052336           5482    5633    5671    7552    7969   10260#
CLR0    007014           2768    2769    2770    2771    2773#
CMPB1   014632           4345    4347#
CMPB2   015524           4574    4575    4577#
CMPB3   016120           4705    4708#
CMPN    023070           5757#
CMP0    013742           4099    4100    4101    4103#
CMP0A   014204           4189    4192#
CMP1    014274           4219    4220    4221    4223#
CMP1A   014414           4264    4265    4266    4268#
CMP2    015156           4455    4456    4458#
CMP7    016706           4843    4844    4846#
CNTRLC= 000003          10290#  10295
CNTRLO= 000017          10291#  10310
CNVADR  051642           7658    9069    9149   10111#
COMB1   010424           3187    3188    3190#
COMB1A  010604           3253    3254    3256#
COMB2   011166           3389    3390    3392#
COMB5   011736           3595    3597#
COMB6   012564           3797    3798    3800#
COMB7   013470           4014    4015    4017#
COM0    007060           2788    2789    2790    2791    2793#
COM1    010164           3096    3097    3099#
COM3    011606           3546    3548#
COM4    010710           3291    3293#
COM6    012152           3672    3673    3675#
COM7    013274           3954    3955    3957#
CONTRL= 177746           1336#
CPCHK   003766           2321#
CPUERR= 177766           1349#   5577*   6657*   6769*  10492   10497*
CR     = 000015          1224#   9371    9381
CRLF   = 000200          1225#   2162    2166    2208    2412    2548    7733    7742    9204    9342    9381   10404   10544
                        10558   10569   10576   10589   10602   10611   10623   10642   10815   10820   10823   10827   10831
                        10836   10840   10845   10849   10853   10860   10869   10874   10881   10887   10891   10896   10903
                        10909   10912   10915   10918   10921   10924   10928   10931
DBINB7  017360           4976#   5008*   5009*   5013*   5014*   5015    5019*   5026    5027*   5034*   5036    5037*
DBIN7   016606           4818#   4833*   4837*   4842    4848*   4856*   4863*   4867*   4871
DDATA   016136           4716#   4720*   4722*   4726    4730    4731    4734*   4735    4740*   4747*   4754*   4755    4758*
                         4761    4765
DDATAB  016572           4786*   4787*   4792*   4795    4799*   4800    4804    4809#
```

```
DECB1   010374          3173    3174    3176#
DECB1A  010524          3225    3226    3228#
DECB2   011322          3442    3443    3445#
DECB5   012076          3651    3653#
DECB6A  012664          3828    3829    3830    3832#
DECB7   013536          4033    4034    4036#
DECO    007140          2814    2815    2816    2817    2819#
DEC1    007756          3019    3021#
DEC1A   010226          3117    3119#
DEC2    011056          3351    3352    3354#
DEC5    011562          3532    3533    3534    3536#
DEC6    012320          3721    3722    3724#
DEC7    013166          3919    3920    3922#
DEVICE  001550          1795#
DEVIND  001552          1796#   7690*   7691*   7692*   7696    7698*   7699*   7700
DF1     055056          2049    2054    2059   10664#
DF10    055754          2084   10753#
DF11    056060          2089   10768#
DF12    056142          2094    2099   10778#
DF14    056302          2104   10799#
DF16    056347          2114   10807#
DF4     055407          2064    2074   10707#
DF5     055520          2069   10722#
DH1     055017          2047   10658#
DH10    055705          2082   10746#
DH11    056037          2087   10765#
DH12    056126          2092   10776#
DH13    056206          2097   10786#
DH14    056250          2102   10793#
DH16    056322          2112   10803#
DH2     055124          2052   10673#
DH3     055217          2057   10685#
DH4     055331          2062   10699#
DH5     055453          2067   10715#
DH6     055606          2072   10734#
DISPLA= 177570          1219#   8972*   8996*
DIVO    025546          6380#
DIV1    026014          6463#   6476
DT1     055064          2048   10667#
DT10    055760          2083   10756#
DT11    056062          2088   10770#
DT12    056144          2093    2098   10780#
DT14    056270          2103   10797#
DT16    056354          2113   10810#
DT2     055152          2053   10678#
DT3     055266          2058    2063   10693#
DT5     055524          2068   10725#
DT6     055626          2073   10738#
DUMMY   033220          7440#
ECHO    052556         10331#  10349
EISOPT= 040000          1637#
EMTVEC= 000030          1322#   2136*   2298*   2299*   5366*   5368*   5405*   5406*
EMT1    021164          5365    5373#
EMT1B   021244          5373    5377    5378    5381    5382    5386    5387    5390    5391    5392    5397#
EMT1C   021250          5371    5399#
EMT1D   021262          5400    5401    5404#
```

```
EM1     054772          2046    10654#
EM10    055634          2081    10739#
EM11    055772          2086    10758#
EM12    056070          2091    10771#
EM13    056150          2096    10781#
EM14    056223          2101     2111   10789#
EM15    056306          2106    10801#
EM2     055076          2051    10669#
EM3     055162          2056    10680#
EM4     055302          2061    10695#
EM5     055414          2066    10709#
EM6     055536          2071    10727#
END     036354          7966#
ENDCP   027052          6600     6671#
ENDM    036314          7542     7949#
ENDMEM  036312          7599     7948#
ENDTAG  056366          2251    10812#
END1    036354          7967#
ENTER2  034512          7649     7682#   7734    7876
ERPRT   053440          2402     2558    5092    5573    6654    6765    7605    7967   10488#
ERRBA   001672          1818#    8789*   8790*   8791*   9263*   9266*   9267*   9268*   9280*   9281*   9282*   9283*   9284*
                       10074    10075
ERRRTN  001332          1753#
ERRVEC= 000004          1315#    2181*   2182*   2321*   2322*   2402*   2423*   2448*   2451*   2457*   2477*   2497*   2556
                        5050*    5051*   5052    5092*   5486*   5487*   5530*   5573*   5574*   5575*   5576*   6645*   6654*
                        6725*    6726*   6765*   6766*   6767*   6768*   7598*   7599*   7605*   7967*   8936    8937*   8939*
                        8342*
EXIT    036200          7838     7919    7924#
EXITFL  001536          1790#
EXITRE  034426          7622     7624    7655    7661#   7926
EXPEXT  051040          7355     7366    7411    9901    9923#
EXTINS  023250          5807#
FACTOR  001506          1778#    2593*   2710*   3744    3874*   3987    4440    4441    4718    4778    4891    4892    4896
                        4897     4927*   4997    5051    5176    5190    5198    5214    5239    5276    5472*   5584    5589
                        5600     5800*   5906    6220*   6539    6558    6693*   7011*   7428*   7632    7640*   10130
FLD20 = 104420          9188    10038#
FLTADD  032756          7027     7067    7070    7100    7196    7236    7239    7269   7377#
FLTDBL  050700          7190     9879#
FLTDIV  032730          7160     7329    7363#
FLTMPY  032706          7030     7040    7049    7059    7062    7112    7120    7128    7161    7199    7209    7218    7228
                        7231     7281    7289    7297    7330    7352#
FLTMP0  001422          1764#    7200*   7252    7282*   7313    7331*  10810
FLTMP1  001432          1765#    7240*   7303*   7334*  10810
FLTSGL  050706          7021     9880#
FLTSUB  032752          7085     7108    7133    7146    7168    7254    7277    7302    7315    7337    7376#
FL20  = 104416          9178     9804   10037#
FPOPT = 020000          1638#    2330    7018
FRSTAD  001512          1780#    2590*   2707*   3871*   4518    4924*   5469*   5797*   6217*   6690*   7008*   7625
FRSTME  001514          1781#    2251*   7635
GETMAP  052052          8101     8182    8401    8522    8773   10060  10192#
GIVEMA  052316          8399     8435    8520    8556    8767    8800  10251#
GNS   = ****** U         1660     2161    2165    2207    2411    2547    7732    7741    8025    8032    9203   10030   10031
                       10032    10033   10034   10035   10036   10037   10038   10403
GSTST   007546          2961#
HALT1   026734          6643#
HIADRS= 177742          1334#   10162   10378
```

```
HITMIS= 177752          1338#
HT    = 000011          1222#   9340    9381
INCB1   010310          3142    3143    3145#
INCB2   011416          3477    3478    3479    3481#
INCB3   011770          3609    3611#
INCB6   012476          3771    3773#
INCB6A  012646          3822    3823    3825#
INCB7   013522          4027    4028    4030#
INCO    007160          2823    2824    2825    2826    2828#
INC1    010072          3061    3062    3063    3065#
INC3    011616          3551    3553#
INC4    010740          3304    3305    3307#
INC6    012262          3706    3707    3708    3710#
INC7    013334          3971    3973#
INSTBL  003172          2188    2195    2215#
IOMON   034434          7606    7671#   7877
IOTTST  020670          5294#
IOTVEC= 000020          1320#   2296*   2297*   5294    5307*   5361*   5362*   5407*   5408*   6617*   6636*
IOWC    001546          1794#   7648*   7681*   7772
JMP1    020104          5107    5108    5109    5111#
JMP3    020154          5127    5131#
JMP4    020212          5143    5147#
JMP5    020256          5161    5165#
JMP6    020276          5173#
JMP7    020334          5184#
JSR1    020346          5190#
JSR3    020432          5200    5214#
JSR4    020520          5226    5239#
JSR6    020576          5248    5262#
JSR7    020630          5266    5276#
KDPAR0= 172360          1492#   6863
KDPAR1= 172362          1493#
KDPAR2= 172364          1494#
KDPAR3= 172366          1495#
KDPAR4= 172370          1496#
KDPAR5= 172372          1497#
KDPAR6= 172374          1498#
KDPAR7= 172376          1499#
KDPDR0= 172320          1470#   6855
KDPDR1= 172322          1471#
KDPDR2= 172324          1472#
KDPDR3= 172326          1473#
KDPDR4= 172330          1474#
KDPDR5= 172332          1475#
KDPDR6= 172334          1476#
KDPDR7= 172336          1477#
KERSTK= 001200          1209#   2218    2263    5564    7938    7970
KIPAR0= 172340          1481#   6860    7564*   7928*   7935    10120   10422   10447
KIPAR1= 172342          1482#   7565*   7929*
KIPAR2= 172344          1483#   7566*   7930*
KIPAR3= 172346          1484#   7567*   7568    7570    7673    7928
KIPAR4= 172350          1485#   6878*   7568*   7569*   7929
KIPAR5= 172352          1486#   7570*   7571*   7930
KIPAR6= 172354          1487#  10380   10383*  10395*
KIPAR7= 172356          1488#   7572*   7583    7594
KIPDR0= 172300          1459#   6852    7556*
```

```
KIPDR1= 172302          1460#   7557*
KIPDR2= 172304          1461#   7558*
KIPDR3= 172306          1462#   7559*
KIPDR4= 172310          1463#   6879*   7560*
KIPDR5= 172312          1464#   7561*
KIPDR6= 172314          1465#  10384*
KIPDR7= 172316          1466#   7562*
KM    = 000000          1648#
KSP   =%000006          1243#
KTABRT  053272          2576    6635   10459#
KTABT   027732          6876#
KTEX    030172          6877    6928#
KTOPT = 100000          1636#
KTPAR   027560          6817#
KTPDR   027442          6779#
LDKT    053126          8762    8839    8902    9389   10321   10417#
LDRP3   037354          8115    8121#   8370
LDRP4   040276          8251    8258#   8589    8615    8628    8643    8658
LDRS    040602          8306    8315#   8708    8736
LF    = 000012          1223#   9375    9381
LKOPT = 001000          1640#   2334    7453    7498
LKS   = 177546          1605#   2332    7463*   7465    7502*   8914*
LKSRV   044442          7484    7500    8901#
LKVEC = 000100          1606#   7456*   7457*   7484*   7500*   7501*
LOADRS= 177740          1333#  10161   10376
LOOP    005406          2531    2537    2539    2541    2543    2554#   7941    7989    8047
LSTMEM  001516          1782#   2249*   2250*   7637
LTICKS  001602          1808#   7797    7814    8903*   8904    8906*   8907*   8908    8910*   9390
MAINT = 177750          1337#   7979*   7991*
MAPEND  051636         10081   10091#
MAPH0 = 170202          1576#   6982*
MAPH00= 170202          1512#   1576
MAPH01= 170206          1514#   1578
MAPH02= 170212          1516#   1580
MAPH03= 170216          1518#   1582
MAPH04= 170222          1520#   1584
MAPH05= 170226          1522#   1586
MAPH06= 170232          1524#   1588
MAPH07= 170236          1526#   1590
MAPH1 = 170206          1578#   6980*
MAPH10= 170242          1528#
MAPH11= 170246          1530#
MAPH12= 170252          1532#
MAPH13= 170256          1534#
MAPH14= 170262          1536#
MAPH15= 170266          1538#
MAPH16= 170272          1540#
MAPH17= 170276          1542#
MAPH2 = 170212          1580#
MAPH20= 170302          1544#
MAPH21= 170306          1546#
MAPH22= 170312          1548#
MAPH23= 170316          1550#
MAPH24= 170320          1552#
MAPH25= 170326          1554#
MAPH26= 170332          1556#
```

```
MAPH27= 170336           1558#
MAPH3 = 170216           1582#
MAPH30= 170342           1560#
MAPH31= 170346           1562#
MAPH32= 170352           1564#
MAPH33= 170356           1566#
MAPH34= 170362           1568#
MAPH35= 170366           1570#
MAPH36= 170372           1572#
MAPH37= 170376           1574#
MAPH4 = 170222           1584#
MAPH5 = 170226           1586#
MAPH6 = 170232           1588#
MAPH7 = 170236           1590#
MAPL0 = 170200           1575#   6937    6970    6981*  10085   10226
MAPL00= 170200           1511#   1575
MAPL01= 170204           1513#   1577
MAPL02= 170210           1515#   1579
MAPL03= 170214           1517#   1581
MAPL04= 170220           1519#   1583
MAPL05= 170224           1521#   1585
MAPL06= 170230           1523#   1587
MAPL07= 170234           1525#   1589
MAPL1 = 170204           1577#   6979*
MAPL10= 170240           1527#
MAPL11= 170244           1529#
MAPL12= 170250           1531#
MAPL13= 170254           1533#
MAPL14= 170260           1535#
MAPL15= 170264           1537#
MAPL16= 170270           1539#
MAPL17= 170274           1541#
MAPL2 = 170210           1579#
MAPL20= 170300           1543#
MAPL21= 170304           1545#
MAPL22= 170310           1547#
MAPL23= 170314           1549#
MAPL24= 170320           1551#
MAPL25= 170324           1553#
MAPL26= 170330           1555#
MAPL27= 170334           1557#
MAPL3 = 170214           1581#
MAPL30= 170340           1559#
MAPL31= 170344           1561#
MAPL32= 170350           1563#
MAPL33= 170354           1565#
MAPL34= 170360           1567#
MAPL35= 170364           1569#
MAPL36= 170370           1571#
MAPL37= 170374           1573#
MAPL4 = 170220           1583#
MAPL5 = 170224           1585#
MAPL6 = 170230           1587#
MAPL7 = 170234           1589#
MAPTBL  001656           1814#   2276*   2277*  10221   10224*  10253*
MAPTST  030212           6937#
```

```
MAPTWO  030340          6936    6972    6977#
MARKEX  024272          6068    6072    6076#
MARK1   024246          6064    6066#
MBRK    026462          6581#
MBTAS = 160116          1626#
MBTBA = 160104          1621#   2014
MBTBAE= 160174          1630#   2015
MBTCS1= 160100          1619#   2012
MBTCS2= 160110          1623#   2017
MBTCS3= 160176          1631#   2020
MBTDB = 160120          1627#
MBTDT = 160126          1629#   2023
MBTER = 160114          1625#   2019
MBTERR  044302          8850    8867#
MBTMR1= 160124          1628#
MBTMR2= 160106          1622#   2016
MBTN2   002276          2024#   2379
MBTN3   002300          2025#   2383
MBTN4   002302          2026#   2387
MBTOPT= 002000          1639#   2377    2401    7517
MBTPSW= 000776          1633#   2022
MBTSET  033522          7507    7509    7517#
MBTSRV  044130          7530    8838#
MBTST = 160112          1624#   2018
MBTTBL  002246          2012#   2375    2393    2397*   2398*   2399    7527*   7528*   7529*   7530*   7531*   7532*   8841
                        8846*   8878    8892*   10532
MBTVEC= 000774          1632#   2021
MBTWC = 160102          1620#   2013
MBT1    033566          7527#
MEMERR= 177744          1335#   10160   10166*  10375   10389*  10390   10396   10397*
MEMHOL  044276          8857    8859    8866#
MEMSIZ  003326          2245    2247    2249#
MHOLE   044000          8796    8798    8808#
MMON    001503          1775#   2265*   5629    6607    6784    6822    6876    7525    7597*   7623    7743    7789    7821
                        7847    7874    7890    7924    7959*   10113   10381   10417   10440
MMR0  = 177572          1360#   1364    10462   10468*
MMR1  = 177574          1361#   1365
MMR2  = 177576          1362#   1366    10463
MMR3  = 172516          1363#   1367    6935    7510    7950*   8097    8178    8396    8432    8517    8553    10077
MMVEC = 000250          1328#   2576*   2577*   6603*   6635*   6884    6885    6888*   6889*   6925*   6926*
MOVB1   014606          4330    4331    4332    4333    4336#
MOVO    013672          4077    4078    4080#
MOVOA   013722          4092    4094#
MOV1    014464          4289    4291#
MOV7    016656          4834    4836#
MPI     026530          6599#
MRGTAB  053550          7990    7991    10516#
MRKTST  024210          6053#
MSGINX  053610          7726    9169    9197    9458    10534#
MSG1    053634          10544#
MSG10   054100          10538   10574#
MSG11   054106          10539   10575#
MSG12   054114          9208    10576#
MSG13   054227          9215    10589#
MSG14   054337          9298    10602#
MSG15   054375          10542   10608#
```

```
MSG16   054416          9233   10611#
MSG17   054525          9239   10623#
MSG2    053646         10546#
MSG20   054601          7996   10631#
MSG21   054701         10536   10537   10540   10541   10642#
MSG22   054721          9254   10645#
MSG23   054731          9259   10647#
MSG24   054740         10543   10649#
MSG25   054762          2532   10652#
MSG3    053660          9057   10548#
MSG4    053747          9449   10558#
MSG5    054064         10534   10572#
MSG6    054072         10535   10573#
MTICKS  001600          1807#   2273#   8911*   9400
MULO    025416          6329#
MXMMHI  001560          1799#   2260*   8797    8858
MXMMLO  001562          1800#   2261*   8795    8856
NEGB1   010360          3166    3167    3169#
NEGB4   011346          3454    3455    3457#
NEGB6   012602          3804    3805    3807#
NEGB7   013604          4052    4053    4055#
NEG0    007200          2833    2834    2835    2837#
NEG1    010216          3111    3112    3114#
NEG2    010762          3316    3318#
NEG5    011574          3539    3540    3542#
NEG6    012206          3684    3685    3686    3688#
NEG7    013242          3941    3942    3944#
NEXEC   001502          1774#   2597    2714    3878    4931    5476    5804    6224    6697    7015    7942
NEXPAR  001520          1783#   2266*   2269*   7567    7578    7589    7927*   7955*   7958*
NOMEM   034430          7638    7660    7662#
NOTYPE  001466          1768#   2317*   9366   10313*  10331
NULLS   001526          1786#   7435   10352
NWBASH  001544          1793#   7645*   7677*   7786    7795*
NWBASL  001542          1792#   7644*   7676*   7746    7785    7794*   7827
OAERR   017642          5043#
OLDBAS  001540          1791#   7643*   7672*   7745    7784    7793*   7823
OLDPSW  033650          7550#   7940
OPT.CP  001470          1769#   2404*   2533    5585    7018    7430    7453    7498    7506    7517
OVFLW   021614          5480#
PARTBL  027704          6817    6860#
PA1500  001476          1772#   8793*   8795    8816    8852*   8854*   8856    8874    9072*   9075    9150    9162*   9270
                       10079*  10086*  10089*  10127*
PA2116  001500          1773#   8794*   8797    8817    8853*   8855*   8858    8875    9073*   9163*  10080*  10087*  10090*
                       10126*
PC      =%000007        1246#   2189*   2199*   2225*   2228*   2544*   2588    2591    2594    2690    2692*   2705    2708
                        2711   2963*   2965*   2977*   3005    3127    3277    3379    3497    3582    3659*   3661    3854
                        3856*  3869    3872    3875    3886    4076    4179    4181    4185    4188    4201    4320    4385
                        4513   4528    4600    4666    4822    4882*   4883    4884    4885*   4886    4887*   4888    4889
                        4890*  4891    4892    4896*   4897*   4899    4900    4901    4902*   4903*   4907    4909*   4922
                        4925   4928    4979    5044    5099    5114    5134    5149    5241    5264    5265*   5273*   5299
                        5315   5364    5417    5423    5452    5454*   5467    5470    5473    5482*   5484    5488*   5538
                        5542*  5552*   5567    5569    5593*   5633*   5664    5666    5671*   5735    5743*   5745    5757
                        5780   5782*   5795    5798    5801    5818    5831    5859    5865    5866    5882    5883    5914
                        6054   6057    6058    6059    6060    6061    6064*   6106    6114    6164    6179    6190    6200
                        6202*  6215    6218    6221    6231    6234*   6257    6260*   6280    6301    6332    6333*   6356
                        6358*  6381    6413    6416*   6437    6440*   6463    6486    6506    6601    6615    6643    6650
```

```
                    6673   6675*  6688   6691   6694   6723   6727   6750   6886   6891   6922   6966   6984
                    6986*  7006   7009   7012   7021*  7027*  7030*  7040*  7049*  7059*  7062*  7067*  7070*
                    7085*  7100*  7108*  7112*  7120*  7128*  7133*  7146*  7160*  7161*  7168*  7190*  7196*
                    7199*  7209*  7218*  7228*  7231*  7236*  7239*  7254*  7269*  7277*  7281*  7289*  7297*
                    7302*  7315*  7329*  7330*  7337*  7355*  7356*  7366*  7367*  7411*  7413*  7416   7418*
                    7458*  7464   7512*  7549*  7552*  7654*  7658*  7661*  7662*  7634*  7918*  7969*  8012*
                    8017*  8020*  8042*  8066*  8101*  8115*  8125*  8144*  8182*  8217*  8251*  8265*  8283*
                    8306*  8320*  8370*  8399*  8401*  8435*  8465*  8505*  8520*  8522*  8546*  8556*  8563*
                    8589*  8615*  8628*  8643*  8658*  8708*  8736*  8762*  8767*  8773*  8777*  8792*  8800*
                    8802*  8803*  8830*  8839*  8844*  8890*  8902*  8912*  9009*  9056*  9069*  9076*  9125*
                    9149*  9151*  9269*  9271*  9347*  9354*  9361*  9377*  9379*  9389*  9433*  9435*  9456*
                    9651*  9869*  9883*  9901*  9903*  9962*  9996   10060*  10064*  10092*  10129*  10157*  10159*
                   10201*  10242*  10255*  10261*  10269*  10321*  10329*  10434*  10453*
PDRTBL  027666      6779   6852#
PERET   052754     10390#
PHYMAP  051534      8792   9269   10073#
PIRQ    = 177772    1217#  6533   6552   6553*
PIRQVE= 000240      1327#  6535*  6536*  6538*  6539*  6543*  6571*  6572*
PIRQO   026242      6529#
PKM     = 000000    1650#
PLKCSB= 172542      1603#
PLKCSR= 172540      1602#
PLKVEC= 000104      1604#
PROT    003244      2229#  2248*  2250   2267   7956
PRO     = 000000    1249#
PR1     = 000040    1250#
PR2     = 000100    1251#  6163   6171
PR3     = 000140    1252#
PR4     = 000200    1253#  2314   2316   5296   5314   6188   6194   7551
PR5     = 000240    1254#  5303   6161   7531   7781
PR6     = 000300    1255#
PR7     = 000340    1256#  2559   2565   2570   2572   2574   2578   5302   5352   5362   5406   5415   5416
                    5446   5448   5483   5575   6111   6113   6119   6139   6177   6182   6189   6489   6536
                    6537   6574   6767   7457   7501   10055  10196
PS      = 177776    1214#  1215   2290*
PSM     = 010000    1645#  5329
PSW     = 177776    1215#  2183*  2204*  2549*  2557   2578*  2961   2975*  2980*  5353*  5368   5415*  5421
                    5449*  5480   5481*  5483*  5511*  5516*  5574   5631   5632*  5635   5669   5670*  5733*
                    5778*  6103   6111*  6119*  6161*  6167   6177*  6182   6188*  6194   6198*  6234   6238
                    6242*  6244   6260   6266   6416   6420   6440   6444   6535   6540*  6573*  6599   6634*
                    6656*  6664   6667   6670*  6726   6733   6766   6889   6899   7446   7450*  7469*  7549
                    7551*  7939*  7968*  8350   8388   8429   8469   8509   8550   8591   8630   8660   8689
                    8722   8752   10194  10239*  10260  10271*  10420  10421*  10451*
PSWCHK  022420      5629#
PSWTAB  053534      7988   10508#
PUM     = 030000    1651#  5328   5334   6161   6177   6182   6188   6194
PWRVEC= 000024      1321#  2302*  2303*  9968*  9969*  9977*  9992*  9993*
QV      001504      1776#  2224*  2232*  2244   2536   6977   7735   7992   10209
REG     = 004000    1646#  6150   6153   6161   6182   6194
REGINX  053564      9220   9235   9288   10524#
RELE0   006552      2599   2689#
RELE1   012734      2716   3853#
RELE2   017124      3880   4906#
RELE3   021504      4933   5451#
RELE4   023140      5478   5779#
RELE5   024724      5806   6199#
```

```
RELE6   027052          6226    6672#
RELE7   030370          6699    6978    6983#
RELE8   033114          7017    7342    7344    7415#
RELNIO  034372          7642    7651#   7653    7750
RELOC   034240          2692    3856    4909    5454    5782    6202    6675    6986    7418    7621#
RELOCP  036142          7680    7754    7914#   7916
RELO    005556          2588#
RELOO   006566          2693#
REL1    006606          2705#
REL11   012750          3857#
REL2    012770          3869#
REL22   017140          4910#
REL3    017160          4922#
REL33   021520          5455#
REL4    021540          5467#
REL44   023154          5783#
REL5    023174          5795#
REL55   024740          6203#
REL6    024760          6215#
REL66   027066          6676#
REL7    027106          6688#
REL77   030404          6987#
REL8    030424          7006#
REL88   033130          7020    7419#
RESERR  053366          2403    2562    5093    5623    6655    10475#
RESET1  027024          6663#
RESKT   053214          8777    8803    8830    8844    8890    8912    9433    10329   10440#
RESPSW  052352          10267#  10273
RESREG= 104414          8058    8111    8135    8188    8208    8250    8275    8305    8778    8804    8831    8845    8891
                        8913    9123    9434    9650    9776    9829    10036#  10091   10128   10238   10330
RESTPS  052362          10271#
RESTRP  022240          5583#
RESVEC= 000010          1316#   2168*   2169*   2323*   2324*   2403*   5052*   5093*   5588*   5589*   5623*   6646*   6655*
                        7781*   7782*   7894*
RETPC   034432          7634*   7663#
RETPSW  052344          10262#  10272
RETRY   034200          7598#   7923
RKBA    002132          1958#   8193*   8490*   8532*
RKCS    002126          1956#   2428*   2437*   7883*   8197*   8460    8464*   8466    8480    8491*   8497    8504*   8506
                        8512    8533*   8539    8545*   8547    9255
RKDA    002134          1959#   2427*   8189*   8488*   8530*
RKDRV   037406          1894    8131#
RKDS    002122          1954#   2431    8190    10525
RKER    002124          1955#   2429
RKERR   041616          8463    8472#   8502    8543
RKFUN   002062          1917#   8196*   8444*   8445    8450*   8452    8503*   8544*   8626*
RKHANA  002030          1894#
RKHDA   002046          1906#   8176*   8189    8488    8530
RKHSTA  001700          1825#   8132*   8133    8138*   8458    8472*   8479*   8482*   8495    8537    8558*
RKHWC   001720          1838#   8192    8489    8531
RKLOOP  041644          8459    8479#   8496    8538
RKNEWH  001770          1868#   8524    8526*   8529
RKNEWL  001766          1867#   8521    8532
RKOLD   001736          1850#   8181    8184    8187*   8193    8487    8490
RKPSW   002140          1961#   8195*
RKREAD  042174          8455    8537#
```

```
RKRPT   041460          8136    8443#
RKSRV   041510          8194    8450#
RKTRY   002073          1929#   8177*   8462    8468*   8485*   8501    8508*   8516*   8541    8549*
RKUNIT  002014          1883#   8173
RKVEC   002136          1960#   8194*
RKWC    002130          1957#   8192*   8489*   8531*
RKWRCK  041744          8453    8495#
RKWTRY  037722          8189#   8448    8470
RK1     041670          8446    8461    8485#
RK10    001570          1803#   8183*   8186*   8197    8486*   8487*   8491    8528*   8529*   8533
RK11    001572          1804#   8137*   8155    8474    8499
RK2     041712          8488#   8510
RK3     042142          8449    8530#   8551
RNTBIN  001556          1798#   7683*   7777*   7778    7791    8060    8137    8210    8277
ROLB1   010322          3148    3149    3151#
ROLB2   011240          3412    3413    3414    3416#
ROLB3   012044          3634    3635    3637#
ROLB6   012616          3810    3811    3813#
ROLB6A  012712          3840    3841    3843#
ROLB7   013622          4059    4060    4062#
ROLO    007232          2849    2850    2852#
ROL1    010012          3033    3034    3035    3037#
ROL1A   010024          3040    3041    3043#
ROL3    011644          3563    3564    3566#
ROL4    011036          3341    3342    3344#
ROL6    012136          3665    3666    3667    3669#
ROL7    013352          3977    3978    3980#
RORB1   010410          3180    3181    3183#
RORB1A  010500          3213    3214    3216#
RORB4   011222          3404    3405    3406    3408#
RORB5   012006          3615    3616    3617    3619#
RORB6   012550          3791    3792    3794#
RORB7   013506          4021    4022    4024#
RORO    007116          2805    2806    2807    2808    2810#
ROR1    007742          3011    3012    3014#
ROR1A   010054          3053    3054    3055    3057#
ROR2    010724          3297    3298    3300#
ROR5    011516          3512    3513    3515#
ROR6    012226          3692    3693    3694    3696#
ROR7    013224          3934    3935    3937#
RPCC    002172          1976#
RP3BA   002110          1947#   8124*   8413#
RP3CS   002104          1945#   2461*   2462    8114*   8119*   8342    8346*   8347    8361    8371*   8377    8384*   8385
                        8414*   8420    8425*   8426    9258
RP3DA   002112          1948#   8095*   8121*   8410*
RP3DC   002114          1949#   8122*   8411*
RP3DRV  036770          1893    8054#
RP3DS   002100          1943#   2468    8112    10524
RP3ER   002102          1944#   8391
RP3ERR  040772          8345    8353#   8382    8423
RP3FUN  002060          1916#   8118*   8327*   8328    8333*   8334    8383*   8424*
RP3HAN  002026          1893#   7782    7894
RP3HDA  002042          1904#   8121    8410
RP3HDC  002044          1905#   8094*   8122    8411
RP3HST  001676          1824#   2279    7755    7760    7806    7879*   7893*   7896    7898    8055*   8056    8340    8353*
                        8360*   8363*   8375    8418    8437*
```

```
RP3HWC  001716          1837#   7772*   8123    8412
RP3LOO  041020          8341    8360#   8376    8419
RP3NWH  001764          1866#   7786*   8403    8405*   8408
RP3NWL  001762          1865#   7785*   8400    8413
RP3OLD  001732          1848#   7784*   7787*   8100    8103    8106*   8124    8368
RP3PSW  002120          1951#   8117*
RP3REA  041346          8337    8418#
RP3RPT  040642          8059    8326#
RP3SRV  040672          8116    8333#
RP3TRY  002072          1928#   8096*   8344    8349*   8366*   8381    8387*   8395*   8422    8428*
RP3UNI  002012          1882#   7780*   8107    8110*   8114    8369    8409
RP3VEC  002116          1950#   8116*
RP3WC   002106          1946#   8123*   8412*
RP3WCK  041110          8335    8375#
RP3WTR  037306          8112#   8113    8331    8351
RP31    041044          8329    8343    8366#
RP310   001564          1801#   8102*   8105*   8109*   8119    8367*   8368*   8369*   8371    8407*   8408*   8409*   8414
RP311   001566          1802#   8060*   8075    8355    8379
RP32    041074          8370#   8389
RP33    041306          8332    8410#   8430
RP4BA   002146          1966#   8264*   8645*
RP4BAE  002150          1967#   8263*   8644*
RP4CS1  002142          1964#   2452    2482*   8255*   8595*   8596    8616*   8646*   10528
RP4CS2  002154          1969#   2481*   2483    2491*   7889*   8258*   8588*   8627*   8633    8657*
RP4CS3  002156          1970#
RP4DA   002152          1968#   8261*
RP4DC   002164          1973#   8260*
RP4DRV  040002          1897    8204#
RP4DS   002160          1971#   2485    8584    8607    8613    8622    8641    8652
RP4ERR  042502          8587    8599#   8625    8655
RP4ER1  002162          1972#
RP4ER2  002166          1974#
RP4ER3  002170          1975#
RP4FUN  002066          1919#   8254*   8571*   8572    8576*   8577    8656*
RP4HAN  002036          1897#
RP4HDA  002052          1908#   8248*   8261
RP4HDC  002054          1909#   8247*   8260
RP4HST  001706          1828#   8205*   8206    8211*   8582    8593    8599*   8606*   8609*   8620    8650    8663*
RP4HWC  001726          1841#   8262
RP4LOO  042530          8583    8606#   8621    8651
RP4NWH  002004          1874#   8644
RP4NWL  002002          1873#   8645
RP4OF   002174          1977#   8259*
RP4OLD  001752          1856#   8263    8264
RP4PSW  002200          1979#   8253*
RP4REA  042754          8579    8650#
RP4RPT  042336          8209    8570#
RP4SRV  042362          8252    8576#
RP4TRY  002075          1931#   8249*   8586    8590*   8612*   8624    8629*   8640*   8654    8659*
RP4UNI  002022          1886#   8258
RP4VEC  002176          1978#   8252*
RP4WC   002144          1965#   8262*
RP4WCK  042606          8578    8620#
RP4WTR  040244          8251#   8575    8592
RP41    042556          8573    8585    8612#
RP411   001574          1805#   8210*   8227    8601    8635
```

```
RP42    042564          8613#   8614    8631
RP43    042716          8574    8641#   8642    8661
RSBA    002206          1984#   8319*   8738*
RSBAE   002210          1985#   8318*   8737*
RSCS1   002202          1982#   2502*   8312*   8709*   8739*   10529
RSCS2   002214          1987#   2501*   2503    2511*   7887*   8315*   8687*   8720*   8726    8750*
RSCS3   002216          1988#
RSDA    002212          1986#   8316*
RSDRV   040352          1898    8271#
RSDS    002220          1989#   2505    8310    8683    8700    8706    8715    8734    8745
RSER    002222          1990#
RSERR   043162          8686    8692#   8718    8748
RSFUN   002070          1920#   8309*   8670*   8671    8675*   8676    8719*   8749*
RSHANA  002040          1898#
RSHDA   002056          1910#   8303*   8316
RSHSTA  001710          1829#   8272*   8273    8681    8692*   8699*   8702*   8713    8743    8755*
RSHWC   001730          1842#   8317
RSLOOP  043210          8682    8699#   8714    8744
RSNEWH  002010          1876#   8737
RSNEWL  002006          1875#   8738
RSOLD   001756          1858#   8318    8319
RSPSW   002226          1992#   8308*
RSREAD  043430          8678    8743#
RSRPT   043046          8276    8669#
RSSRV   043072          8307    8675#
RSTRY   002076          1932#   8304*   8685    8688*   8705*   8717    8721*   8733*   8747    8751*
RSUNIT  002024          1887#   8315
RSVEC   002224          1991#   8307*
RSWC    002204          1983#   8317*
RSWCK   043266          8677    8713#
RSWTRY  040542          8306#   8674    8690
RS11    001576          1806#   8277*   8292    8694    8728
RS41    043236          8672    8684    8705#
RS42    043244          8706#   8707    8723
RS43    043372          8673    8734#   8735    8753
RTI1    036554          7993    7998#
RTT1    024304          6103#
RTT1EX  024502          6140#
RTT2A   024516          6146    6148#
RTT2EX  024716          6105    6157    6172    6174    6183    6185    6195    6198#
RT1     053120          10371   10409#
RUNTBL  001626          1812#   7779*   7854    8076    8079*   8156    8159*   8228    8231*   8294*
RUNTRA  001642          1813#   7909    8093*   8170*   8246*   8300*   8356*   8475*   8602*   8695*
R0      =%000000        1228#   1236    2137*   2138    2139    2141*   2142    2143    2145*   2146    2147    2149*   2150
                        2151    2153    2186*   2193*   2271*   2275*   2278*   2281*   2283*   2285*   2287*   2327    2425*
                        2435*   2436*   2440    2459*   2464*   2472*   2474    2479*   2489*   2490*   2494    2499*   2509*
                        2510*   2514    2521*   2522*   2524*   2525    2529*   2556*   2558*   2561*   2562*   2563*   2564*
                        2566*   2567    2568*   2569*   2570*   2571    2572*   2588*   2589    2590    2591*   2592*   2593
                        2705*   2706    2707    2708*   2709*   2710    2767*   2777    2787*   2796*   2804*   2813*   2822*
                        2832*   2840*   2848*   2854*   2862*   2868*   2869*   2881*   2883*   2884    2896    2904*   2905*
                        2907*   2911*   2915*   2969*   2970    2981    2986    3381*   3382    3384*   3500*   3502    3585*
                        3586*   3589    3590*   3591    3592*   3661*   3662    3735    3743*   3744*   3869*   3870    3871
                        3872*   3873*   3874    3886*   3887    3888    3889*   3890    3891*   3892    3893    3894*   3895*
                        3986*   3987*   3988    3989    3990    3991*   3992    3993*   3994    4076*   4082    4084    4105
                        4135    4141    4144    4151    4153    4155    4159    4162    4179*   4180    4181    4185*   4186*
                        4187    4188    4190    4532*   4533    4539    4542    4544    4547    4549*   4552    4554*   4557
```

```
                4560    4601*   4602    4603    4605    4607    4608    4609    4610    4666*   4667    4668    4671
                4673    4675    4676    4678    4752*   4753    4755*   4768    4822*   4823    4824    4826    4828
                4829    4880*   4885    4888    4889    4922*   4923    4924    4925*   4926*   4927    4979*   4980
                4981*   4982*   4983*   4984    4985*   4986*   4987*   4995*   4996*   5099*   5100*   5102    5119*
                5121*   5128    5129    5135*   5144    5145    5155*   5162    5163    5215*   5224*   5227*   5264*
                5268    5269    5279*   5284*   5295*   5296*   5297    5302*   5303*   5304    5305*   5311    5340*
                5343*   5347    5363*   5374*   5375*   5376*   5380*   5385*   5389*   5394*   5399*   5402    5423*
                5436*   5437    5467*   5468    5469    5470*   5471*   5472    5521*   5533*   5548*   5551*   5554*
                5559    5640*   5641*   5643    5691*   5692    5705    5719    5730    5745*   5746*   5747*   5748*
                5757*   5758    5762    5771    5795*   5796    5797    5798*   5799*   5800    5807*   5809*   5814*
                5818*   5819    5823    5831*   5832    5835    5836*   5838    5841    5845*   5846    5848*   5853
                5866*   5867    5868*   5869    5896*   5907    5977*   5983*   5988    5993*   5998*   5999    6009*
                6012*   6014    6055*   6073    6112*   6117*   6122    6126    6133*   6148*   6149*   6151    6154
                6156    6167*   6171    6215*   6216    6217    6218*   6219*   6220    6227*   6230    6250*   6251
                6254*   6256    6273*   6274    6279*   6284    6287*   6300*   6306    6308*   6311*   6329*   6341
                6353*   6364    6368    6380*   6385    6390    6396    6405*   6410*   6419    6427*   6429    6433*
                6443    6451*   6453    6529*   6550    6583*   6584    6585    6588*   6664*   6667    6670    6688*
                6689    6690    6691*   6692*   6693    6707*   6708    6709    6714*   6745*   6781*   6786*   6791*
                6792    6796*   6797    6801*   6819*   6824*   6829*   6830    6834*   6835    6840*   6890*   6894
                6899*   6900*   6901*   6902*   6903*   6904    6906*   6907    6909*   6910    6912*   6913    6915*
                6916    6917    6937*   6945*   6946    6951*   6952    6954    6957*   6958    6963*   6964    6967
                6969    6970    7006*   7007    7008    7009*   7010*   7011    7464*   7555*   7556    7557    7558
                7559    7560    7561    7562    7574    7575    7576    7577    7585    7586    7587    7588    7601*
                7604    7625*   7626    7630    7643    7651    7652    7656    7745*   7748    7749    7753*   7802*
                7809*   7824*   7825*   7831*   7843    7859*   7860*   7861*   7862    7914    7920    7935*   7936*
                7937    8039*   8042    8063*   8067*   8069*   8070*   8071    8073*   8078*   8079    8094    8103*
                8104*   8105    8106    8107*   8108*   8109    8110    8141*   8145*   8146*   8148    8160*   8161*
                8162    8163*   8164    8166*   8167*   8169*   8170    8172*   8175*   8176    8184*   8185*   8186
                8187    8214*   8218*   8220*   8221*   8222    8224*   8230    8247    8280*   8284*   8285*   8286
                8287*   8288    8290*   8293*   8294    8301*   8302*   8303    8354    8355*   8356*   8357*   8402
                8403*   8404*   8405    8406*   8473    8474*   8475*   8476*   8523    8524*   8525*   8526    8527*
                8600    8601*   8602*   8603*   8693    8694*   8695*   8696*   8879*   8884*   8889    9063*   9070
                9074    9101*   9102*   9104    9106*   9107*   9108*   9109*   9110*   9111    9116    9121    9126
                9161*   9162    9163    9196*   9197*   9198    9205*   9206    9216    9220*   9221*   9235*   9236*
                9242    9255*   9258*   9261    9262    9263    9279*   9280    9281    9288*   9289*   9291    9299
                9332    9333*   9334    9337*   9391*   9392*   9395    9396*   9397*   9401*   9402*   9405    9406*
                9407*   9410    9412    9413*   9414*   9417    9419    9420*   9421*   9424    9426    9427*   9428*
                9450*   9455*   9566    9576*   9580    9596    9597    9610*   9637*   9641    9654*   9656*   9658*
                9682    9707*   9727*   9729    9730*   9738*   9745*   9751*   9773*   9774*   9775    9806*   9807
                9808*   9809*   9819*   9825*   9826*   9848    9851*   9854*   9858*   9861*   9864    9868*   9880*
                9895*   9924*   9927*   9940*   9942    9948*   9949*   9950    9954    9959    9970    9991*   10014
                10015*  10016   10017*  10018*  10019*  10148   10152   10154   10164   10192*  10224*  10251   10252*
                10253*  10254*  10322*  10323*  10324*  10325   10327*  10328   10422*  10423   10424   10425   10426*
                10427*  10428*  10429*  10430*  10431*  10432*  10443*  10444   10445   10446   10447*  10448*  10449*
                10450*
R1      =%000001
                1229*   1237    2171*   2176*   2187*   2191*   2196*   2202*   2272*   2273*   2274*   2279*   2280*
                2284*   2286*   2426*   2439*   2460*   2466*   2480*   2493*   2500*   2513*   2557*   2559*   2560*
                2561    2563    2565*   2574*   2575    2577    2970*   2971    2987    4600*   4601    6799    6837
                7054*   7055    7056    7223*   7224    7225    7716*   7717*   7718*   7842*   7871    7900    7908*
                8064*   8068*   8080*   8081    8082*   8083    8085*   8092*   8093    8095    8142*   8148*   8149*
                8150    8152*   8154*   8158    8171*   8172    8173*   8174*   8175    8215*   8219*   8232*   8233*
                8234    8235*   8236    8238*   8239*   8245*   8246    8248    8281*   8286*   8295*   8296    8298*
                8300    8302    8880*   8883*   8888    9121*   9140    9142    9156*   9161    9168    9177    9187
                9261*   9264*   9265*   9266    9390*   9393*   9394    9395*   9398*   9399    9400*   9403*   9404
                9405*   9408*   9409    9412*   9415*   9416    9419*   9422*   9423    9426*   9429*   9430    9451*
                9452    9454*   9457    9463    9567    9580*   9581    9585    9609*   9633*   9637    9638*   9653*
```

```
                    9655*  9657*  9683   9706*  9729*  9734   9740   9747   9753   9759*  9761   9765*  9767
                    9810*  9811*  9812   9813   9814*  9828*  9849   9852*  9855*  9859*  9860*  9862*  9863*
                    9865   9867*  9882*  9884   9894*  9926*  9929   9933   9938*  9939*  9944*  9971   9990*
                   10193*  10199  10200  10218*  10219*  10228*  10230*  10232  10234*  10236  10244*  10423*  10430
                   10444*  10448
R10    =%000000     1236#
R11    =%000001     1237#
R12    =%000002     1238#  6162*
R13    =%000003     1239#
R14    =%000004     1240#
R15    =%000005     1241#
R2     =%000002     1230#  1238   2172*  2173   2188*  2189   2194*  2198   2203   2254*  2256*  2259*  2260
                    2325*  2330*  2334*  2338*  2346*  2377*  2401*  2404   2431*  2432*  2433   2468*  2469*
                    2470   2485*  2486*  2487   2505*  2506*  2507   2690*  2691*  2884*  2885*  2886   2895
                    2912*  2914*  2917*  2945*  2946*  2947   2949*  2950*  2952*  2971*  2972   2988   3005*
                    3006*  3007*  3010*  3018*  3025*  3032*  3039*  3045*  3052*  3060*  3067*  3075*  3083*
                    3088*  3095*  3102   3110*  3116*  3382*  3383*  3394*  3411*  3425*  3427*  3441*  3453*
                    3464   3470*  3483   3486   3582*  3583   3584   3585   3587   3588   3589*  3591*  3592
                    3593   3594*  3598   3608*  3621*  3627*  3639*  3649*  3650*  3854*  3855*  3890*  3891
                    3894   3895   3988*  4082*  4084*  4091   4098   4105*  4106   4107   4108*  4109   4115*
                    4116*  4118*  4120   4122   4127   4129   4133*  4135*  4187*  4190   4323*  4324*  4325*
                    4326*  4329   4338*  4340*  4344   4354   4360   4365   4371*  4372   4376*  4378   4392*
                    4393*  4399*  4400   4401*  4402   4403*  4405*  4407   4408   4419*  4420   4426   4439*
                    4440*  4444*  4447   4460   4466   4472*  4476   4483*  4484   4487   4494   4502   4506*
                    4507*  4508*  4513*  4514   4515   4518   4533*  4534*  4542   4552   4606*  4610*  4616
                    4622*  4624   4631   4634   4637*  4639   4644   4650   4718*  4719   4778*  4779   4824*
                    4825   4826*  4827   4907*  4908*  4989*  4998*  5043*  5055*  5057*  5059*  5061   5063*
                    5065*  5067*  5069*  5071*  5073*  5081   5113*  5124*  5126*  5133*  5137*  5142*  5157*
                    5160*  5168*  5171*  5178*  5182*  5191*  5192*  5195   5196   5206*  5218*  5233   5234
                    5253*  5254*  5255   5311*  5324   5330   5335   5338*  5445*  5446*  5447*  5452*  5453*
                    5522*  5532*  5535*  5537*  5539*  5541*  5543*  5548   5583*  5584*  5590   5592   5642*
                    5643   5646   5650   5652   5656   5660*  5684*  5725*  5726   5728   5758*  5759*  5762
                    5771   5780*  5781*  5819*  5821*  5827   5837*  5840*  5841   5846*  5848   5855*  5859*
                    5860   5863*  5864*  5865*  5869*  5905*  5907*  5938*  5939   5992*  5999   6002*  6005
                    6053*  6069   6076   6150*  6151*  6153*  6154*  6160*  6168   6200*  6201*  6230*  6233
                    6243*  6256*  6259   6263*  6265*  6283*  6286*  6288   6301*  6302   6306*  6312   6332*
                    6333   6337   6341*  6347   6356*  6357*  6358   6361   6364*  6366   6383*  6385*  6395
                    6411*  6429   6434*  6453   6464*  6470*  6475   6486*  6504   6506*  6518   6531*  6541
                    6544*  6545   6556   6581*  6582   6584*  6585   6593*  6673*  6674*  6701*  6702*  6703
                    6705*  6708*  6709   6716*  6727*  6729   6749*  6757*  6762*  6764*  6779*  6781   6817*
                    6819   6891*  6893   6940*  6956   6962   6984*  6985*  7381*  7382*  7383   7386*  7387
                    7393*  7394*  7396   7398*  7399   7416*  7417*  7513*  7600*  7628   7629   7634   7635*
                    7636   7639   7644   7651*  7657   7661   7685*  7705*  7713   7721   7746*  7751*  7766*
                    7768*  7770*  7771   7778*  7779*  7803*  7806   7808*  7827*  7828*  7832*  7844   7845
                    7914*  7915   7921   7974*  7975*  7976   7984*  7990   7991   8075*  8076   8079*  8093*
                    8155*  8156   8159*  8170*  8227*  8228   8231*  8246*  8292*  8294*  8300*  8881*  8882*
                    8883   9126*  9127   9129   9131   9133   9135   9139*  9148*  9160*  9167*  9176*  9186*
                    9457*  9458*  9459   9463*  9465*  9568   9579*  9583*  9586   9593*  9594*  9595   9600*
                    9608*  9639*  9640   9641*  9659*  9660*  9684   9705*  9812*  9820*  9827*  9850   9853*
                    9856*  9866*  9881*  9892*  9893*  9928*  9929*  9930*  9931*  9932*  9937*  9945*  9950*
                    9951*  9952*  9956*  9972   9989*  10052*  10053*  10054*  10055*  10056*  10061*  10062*  10075*
                   10076*  10080  10083*  10084*  10085*  10086  10087  10115*  10116*  10119*  10120*  10121  10148
                   10150*  10151*  10152  10163  10202*  10204*  10205*  10206  10211  10214  10216*  10218  10229*
                   10231*  10233  10235*  10237  10375*  10388*  10389  10424*  10431  10445*  10449
R3     =%000003     1231#  1239   2195*  2199   2255*  2257*  2258*  2261   2339*  2351*  2356*  2361*  2367
                    2369*  2370*  2371   2372   2373*  2381*  2385*  2389*  2395   2397   2886*  2887*  2888
```

```
                2894    2921*   2922*   2924*   2947*   2951    2954*   2972*   2973    2989    3127*   3128*   3129
3131*   3134*   3159*   3165*   3172*   3179*   3192*   3224*   3231*   3246*   3252*   3258*   3262*
3264*   3267    3497*   3498*   3499*   3500    3501    3502*   3503    3506    3511*   3524*   3538*
3550*   3568*   4091*   4098    4106*   4107*   4109    4141*   4144*   4161*   4162*   4166*   4169
4170*   4173    4204*   4205*   4207*   4210    4218    4227    4236    4242*   4243*   4245    4250
4256    4263    4270*   4272*   4273    4275    4282    4288    4293    4300*   4301    4302    4349*
4351*   4352    4390*   4391*   4392    4402*   4404    4406*   4410    4411    4413*   4414    4416*
4417    4428    4429*   4430    4528*   4529    4532    4547    4557    4560*   4561*   4562*   4563
4569    4573    4580    4587    4609*   4611*   4668*   4669    4670    4671*   4672*   4673*   4674
4683*   4684*   4690    4691*   4692*   4696*   4697*   4698    4700    4704    4706    4780*   4781*
4782    4827*   4828    4829*   4934*   4935    4936    4942    4962    4990*   4999*   5044*   5047
5048*   5049*   5073    5075    5077    5079*   5114*   5117    5118*   5119    5120*   5121    5122
5129    5149*   5152    5153*   5154*   5155    5159    5163    5229*   5230*   5231    5241*   5244
5246    5304*   5305    5319    5321    5494*   5495*   5496    5531    5590*   5593    5602    5604*
5651*   5652    5657*   5677*   5685    5688    5692*   5694*   5696    5698    5703    5709*   5710*
5711    5713    5820*   5823*   5827    5906*   5914*   5918    5919    5927*   5932*   5937*   5939*
5947    5949*   5953*   5954    6028*   6029*   6030    6228*   6251    6255*   6274    6281*   6285*
6290    6303*   6314    6335*   6349    6360*   6368    6382*   6397    6413*   6414    6415*   6421
6463*   6465    6466    6478    6490*   6491*   6497    6502*   6508*   6511    6516*   6532*   6540
6555*   6605*   6606    6730*   6731    6747    6758*   6761*   6788*   6800*   6826*   6838*   6892*
6893*   6941*   6968*   7603*   7604*   7628*   7652    7700*   7703    7706    7723*   7725    7747*
7748*   7752*   7755    7760    7765*   7772*   7773    7780*   7782    7783*   7784*   7785*   7786*
7787*   7804*   7806*   7810*   7812    7845*   7849*   7863*   7864    7879*   7880    7885    7893*
7894    7896    7898    7915    8076*   8077*   8078    8081*   8087*   8088    8090*   8092    8156*
8157*   8158*   8159    8162*   8168*   8169    8228*   8229*   8230*   8231    8234*   8240*   8241
8243*   8244*   8245    8785*   8786*   8787    8811*   8869*   9196    9205    9212    9222    9248
9250    9464*   9469*   9508    9517*   9523*   9524*   9527*   9532*   9533*   9534    9543*   9569
9577*   9578*   9592*   9595*   9604*   9605*   9607*   9636*   9652*   9659    9685    9704*   9734*
9735*   9736*   9737    9740*   9741*   9742*   9743    9747*   9748*   9749*   9750    9753*   9754*
9755*   9756    9761*   9762*   9763*   9764    9767*   9768*   9769*   9770    9813*   9816    9822
9888*   9889*   9890    9933*   9934*   9935*   9936*   9937    9942*   9943*   9944    9955*   9956
9957*   9960    9973    9988*   10044*  10045*  10046*  10047*  10048*  10074*  10079   10082   10112*
10117*  10118*  10124   10130*  10132   10203*  10215*  10217*  10219   10227*  10230   10376*  10377
10379*  10385*  10386   10425*  10432   10446*  10450
R4      =%000004
                1232#   1240    2365*   2367*   2371*   2372*   2393*   2395*   2888*   2889*   2890    2893    2928*
2929*   2931    2951*   2956*   2973*   2974    2990    3129*   3130*   3136*   3141*   3147*   3153*
3186*   3194*   3200*   3201*   3207*   3212*   3219*   3237    3243*   3277*   3278*   3279    3296*
3303*   3310*   3320*   3333*   3340*   3356*   3359*   3366    3369    3503*   3518*   3531*   3545*
3555*   3562*   3569*   4151*   4152*   4153*   4154*   4155    4159*   4160*   4161    4169*   4172*
4173*   4201*   4202    4203*   4204    4208*   4210*   4218    4227    4235*   4236*   4245*   4248*
4250*   4256*   4263    4273*   4274*   4275*   4282*   4288*   4293*   4301*   4302*   4308*   4309
4388*   4389*   4390    4400*   4410*   4411    4414    4420    4423    4425*   4426*   4428*   4438*
4441*   4442    4535*   4545    4555    4563*   4564*   4565*   4569*   4573    4580    4587*   4607*
4612*   4630*   4674*   4676*   4677*   4678*   4681*   4682*   4683    4684    4691    4692    4696
4697    4698    4700    4704    4706    4779*   4780    4935*   4936*   4937    4941*   4942*   4943
4947*   4948    4952    4956*   4957*   4958    4962*   4963*   4991*   5000*   5134*   5135    5139
5140    5141    5145    5207*   5208*   5209    5214*   5217    5220    5223*   5231    5239*   5240
5246*   5255    5257    5258*   5298*   5324    5328*   5329*   5330    5334*   5335    5352*   5353
5592*   5602    5635*   5637    5649*   5650*   5651    5656*   5657    5662*   5669*   5734    5873*
5875    5877*   5880    5919*   5921*   5925    5936*   5937    5942*   5943    5953    5954    5968*
5974*   6018*   6020*   6025*   6044*   6045    6232*   6237*   6245    6258*   6264*   6268    6282*
6284*   6288    6302*   6309*   6312    6336*   6339*   6345*   6347    6361*   6362*   6363*   6366
6395*   6396*   6414*   6419*   6421    6437*   6438    6439*   6445    6465*   6477    6488*   6489*
6490    6497    6500    6511    6514    6533*   6534*   6541*   6545*   6550    6556*   6570*   6731*
6735    6747    6758    6761    6789*   6790*   6791    6792    6794*   6795*   6796    6797    6799*
6827*   6828*   6829    6830    6832*   6833*   6834    6835    6837*   6943*   6944*   6945    6946
```

```
                    6949*  6950*  6951   6952   6955*  6956*  6957   6958   6961*  6962*  6963   6964   7429*
                    7448   7451   7463   7629*  7630*  7631   7636*  7637   7639*  7646*  7647*  7648   7674*
                    7675*  7677   7702*  7709*  7715*  7719*  7773*  7774*  7775*  7776*  7779   7805*  7807*
                    7811*  7818   7846*  7851*  7854*  7855*  7856*  7858*  7863   7907*  7909   7911*  8763*
                    8764   8774*  8775*  8776*  8784*  8785   8789   8790   8822   8823*  8824*  8825*  8826*
                    8827*  8828*  8829*  8841*  8842   8848*  8849   8851*  8852   8653   8860   8861*  8862*
                    8863*  8878*  8879   8880   8881   8885*  8886*  8887   8888*  8889*  9225*  9227*  9229*
                    9234*  9240*  9245*  9290*  9294*  9297*  9302*  9509   9511*  9512*  9513*  9514   9515*
                    9529   9531*  9539*  9542*  9635*  9642*  9686   9703*  9731*  9732*  9737*  9743*  9744*
                    9750*  9756*  9757*  9758*  9764*  9770*  9771*  9772*  9775*  9815*  9821*  9974   9987*
                   10122*  10123*  10125*  10126  10131*  10214*  10221  10224  10225*  10226*  10232*  10233*  10236*
                   10237*  10380*  10395  10396*  10399*  10419*  10433  10442*  10452
R5      =%000005    1233#  1241   2170*  2366*  2368*  2394*  2396*  2890*  2891*  2892   2935*  2936*  2938*
                    2940   2974*  2976   2991   3279*  3280*  3284   3290*  3315*  3325*  3327*  3346*  3350*
                    3369   3379*  3380*  3381   3388*  3397*  3403*  3419*  3434*  3447*  3459   3476*  3486
                    3593*  3602*  3614*  3633*  3644*  4120*  4122   4127*  4129   4320*  4321   4322*  4323
                    4329*  4334*  4341*  4342*  4344   4352*  4354   4360*  4365*  4372   4376   4385*  4386
                    4387*  4388   4404*  4407*  4408   4417   4422*  4423   4430*  4442*  4443*  4447*  4454
                    4460*  4466*  4473*  4476*  4484*  4485*  4487   4494*  4500*  4501*  4502   4514*  4515
                    4536*  4537   4540   4550   4603*  4604   4605*  4606   4616*  4623*  4624*  4631   4634
                    4639*  4644*  4649*  4650   4669*  4687   4693   4719*  4782*  4992*  5001*  5190*  5192
                    5195*  5198   5201*  5209   5216*  5217*  5218   5219*  5220*  5223   5234   5240*  5294*
                    5297*  5298   5301*  5313*  5314*  5965*  5969   5973*  6034   6043*  6054*  6055   6056
                    6063*  6066*  6073   6231*  6232   6233*  6245   6257*  6258   6259*  6268   6280*  6281
                    6290   6304*  6310*  6314   6337*  6349   6397*  6399   6438*  6443*  6445   6750*  6751*
                    6752   6780*  6790   6795   6818*  6828   6833   6942*  6943   6948*  6949   6955   6960*
                    6961   6966*  7626*  7673*  7676   7695*  7712*  7749*  7797*  7798*  7799*  7801*  7814
                    7823*  7824   7833*  7857*  7859   7866*  7917*  9510   9516*  9518*  9520*  9521*  9522*
                    9523   9541*  9570   9572*  9574*  9581*  9585*  9600   9606*  9634*  9640*  9645*  9646
                    9647   9649*  9687   9702*  9733*  9739*  9746*  9752*  9760*  9766*  9816*  9817*  9818*
                    9819   9822*  9823*  9824*  9825   9923*  9961   9975   9986*  10082*  10088*  10089  10121*
                   10124*  10127  10132*  10154  10220*  10221  10223*  10320*  10323
R6      =%000006    1234#  1242   2291*  2292*  2293
R7      =%000007    1235#  1246
SAVPSW  024514      6103*  6104   6109   6134   6147#  6148   6198
SAVREG= 104412      8054   8131   8204   8271   8761   8838   8901   9054   9388   9632   9726   9801   10035#
                   10073  10111  10198  10319
SBCB1   010334      3154   3155   3157#
SBCB3   012064      3645   3647#
SBCB4   011334      3448   3449   3451#
SBCB6   012452      3761   3762   3764#
SBCB7   013434      3993   4000   4001   4003#
SBCO    007262      2863   2864   2866#
SBC1    010106      3068   3069   3070   3072#
SBC1A   010124      3076   3077   3078   3080#
SBC5    011656      3570   3572#
SBC6    012242      3699   3700   3702#
SBC7    013130      3904   3905   3907#
SBINB7  017350      4972#  5003*  5004   5008   5015   5019   5023*  5026*  5029   5033*  5036*  5039
SBIN7   016604      4817#  4832*  4833   4842   4848   4855*  4856   4863   4867   4871
SDATA   016134      4715#  4721*  4722   4723*  4724*  4729*  4730*  4731   4735   4740   4746*  4747   4753*
                    4758   4759*  4760*  4761   4764*  4765   4768
SDATAB  016570      4784*  4785*  4786   4788   4792   4795   4799   4800   4808#
SDPAR0= 172260      1448#  6865
SDPAR1= 172262      1449#
SDPAR2= 172264      1450#
```

```
SDPAR3= 172266          1451#
SDPAR4= 172270          1452#
SDPAR5= 172272          1453#
SDPAR6= 172274          1454#
SDPAR7= 172276          1455#
SDPDR0= 172220          1426#   6857
SDPDR1= 172222          1427#
SDPDR2= 172224          1428#
SDPDR3= 172226          1429#
SDPDR4= 172230          1430#
SDPDR5= 172232          1431#
SDPDR6= 172234          1432#
SDPDR7= 172236          1433#
SECT2   031110          7091    7094#
SECT2D  032214          7260    7263#
SECT3   031454          7152    7156#
SECT3D  032560          7321    7325#
SIPAR0= 172240          1437#   6611    6862    7589*   7590    7592
SIPAR1= 172242          1438#   7590*   7591*
SIPAR2= 172244          1439#   7592*   7593*
SIPAR3= 172246          1440#
SIPAR4= 172250          1441#   6882*
SIPAR5= 172252          1442#
SIPAR6= 172254          1443#   6611*
SIPAR7= 172256          1444#   7594*
SIPDR0= 172200          1415#   6854    7585*
SIPDR1= 172202          1416#   7586*
SIPDR2= 172204          1417#   7587*
SIPDR3= 172206          1418#
SIPDR4= 172210          1419#   6883*
SIPDR5= 172212          1420#
SIPDR6= 172214          1421#   6612*
SIPDR7= 172216          1422#   7588*
SIZE    004504          2423#   2551
SIZEHI= 177762          1346#
SIZELO= 177760          1344#   2255
SM    = 040000          1644#   6153    6194
SOBDBL  051036          9879*   9880    9886    9898    9902*   9904#
SOB0    024002          5966    5977#
SOB1    024010          5979#   5983
SOB10   023764          5968#   6031
SOB2    024036          5979    5980    5981    5982    5984    5985    5986    5987    5990#
SOB3    024050          5994#   6002
SOB4    024102          5994    5995    5996    5997    6000    6003    6004    6007#
SOB5    024104          6009#
SOB5A   024124          6018#
SOB6    024150          6029#
SOB7    024156          6032#
SOB8    024160          5975    6034#
SOB9    023774          5973#   6044
SP    =%000006          1242#   1243    1244    1245    2134*   2180*   2184*   2185*   2197*   2218*   2232    2236    2241
                        2263*   2295*   2533*   2554*   2892*   2893*   2894*   2895*   2896*   2897    2963    2976*   2977
                        2979*   2986*   2987*   2988*   2989*   2990*   2991*   2994*   4535    4536    4539*   4540    4544
                        4545    4549    4550    4554    4555    4899    4900    4997*   4998    4999    5000    5001    5087*
                        5088*   5091*   5208    5230    5254    5269    5276*   5277*   5279    5299*   5300*   5301    5315*
                        5316*   5317    5319    5354*   5364*   5365*   5366    5417*   5418*   5419    5437    5441    5442
```

DEQKCB.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
SUB2    015204          4467    4468    4470#
SUB2A   015324          4509    4510    4512#
SUB3    015642          4617    4618    4620#
SUB3A   015664          4625    4626    4628#
SUB6    016314          4748    4750#
SUB7    016726          4849    4850    4851    4853#
SUPSTK= 000700          1210#   2554    5091    5354    5517    5523    5558    5572    6330    6637    6739    6759    6763
                        6938   10491
SWA80   007300          2870    2871    2872    2874#
SWA81   010540          3232    3233    3235#
SWA82   010752          3311    3313#
SWA84   011400          3471    3473#
SWA86   012676          3835    3837#
SWA87   013256          3948    3950#
SWITCH  056370          9448   10815#
SWR     = 177570        1218#   1219    2115*   2542    7427*   7497*   7508    7519    7540    7621    7641    7678    7686
                        7690    7839    7903    8061    8139    8212    8278    8931    8949    8956    8997    9000    9007
                        9011    9017
SW0     = 000001        1284#
SW00    = 000001        1274#   1284
SW01    = 000002        1273#   1283
SW02    = 000004        1272#   1282
SW03    = 000010        1271#   1281
SW04    = 000020        1270#   1280
SW05    = 000040        1269#   1279
SW06    = 000100        1268#   1278
SW07    = 000200        1267#   1277
SW08    = 000400        1266#   1276
SW09    = 001000        1265#   1275
SW1     = 000002        1283#
SW10    = 002000        1264#
SW11    = 004000        1263#
SW12    = 010000        1262#   7508
SW13    = 020000        1261#
SW14    = 040000        1260#   8931
SW15    = 100000        1259#
SW2     = 000004        1282#
SW3     = 000010        1281#   7519
SW4     = 000020        1280#   8061    8139    8212    8278
SW5     = 000040        1279#   7686
SW6     = 000100        1278#   7540    7621
SW7     = 000200        1277#   2542
SW8     = 000400        1276#   7641    7678
SW9     = 001000        1275#   7839
SXRA    023624          5916#   5925*
SXRB    023626          5917#   5943
SXT0    023272          5810    5811    5812    5813    5816#
SXT1    023356          5843#
SXT2    023642          5923#
SXT3    023656          5929#
SXT4    023424          5861    5863#
SXT5    023666          5934#
SXT6    023566          5898    5899    5900    5901    5903#
SXT6A   023616          5909    5912#
SXT7    023732          5951#
SYSSIZ  001606          1811#   2440*   2474*   2494*   2514*   2519*   2527*   2529*   7703    7706    7716    7723*   7765*
```

```
                    9452    9463
SYSTID= 177764      1348#
TBITVE= 000014      1317#   5416*   5448*   5634    5645*   5648*   5663*   6108*   6110*   6137*   6138*   6139*   6537*
                    6574*   7985*
TEST1   010202      3103    3104    3105    3107#
TEST2   010700      3285    3286    3288#
TEST6   012726      3846    3847    3849#
TIMEBU  047104      2283    2288*   2289*   9394*   9399*   9404*   9409*   9416*   9423*   9430*   9431    9436#
TIMER   042320      8465    8505    8546    8560#
TKBFR   001444      1766    1767#  10325   10327
TKBFRP  001442      1766#  10322   10328*
TKISR   052376      2313   10293#
TKVEC =  000060     1324#   2313*   2314*
TPISR   052610      2315    7488   10347#
TPVEC =  000064     1325#   2315*   2316*   7455*   7488#
TRAPVE= 000034      1323#   2135*   2300*   2301*   5419*   5421*   5427*   5430*   5434*   5439*   5445
TRAP1   021416      5418    5433#
TRAP1C  021456      5432    5445#
TRTVEC= 000014      1318#
TSTB1   010552      3238    3239    3241#
TSTB2   011356      3460    3462#
TSTB2A  011366      3465    3467#
TSTB6   012424      3749    3750    3751    3753#
TSTO    007036      2778    2779    2780    2781    2783#
TST1    005540      2582#
TST10   011120      3375#
TST11   011436      3492#
TST12   011660      3576#
TST13   012100      3657#
TST14   012360      3741#
TST15   012752      3863#
TST16   013354      3984#
TST17   013650      4073#
TST2    006570      2699#
TST20   014206      4196#
TST21   014532      4315#
TST22   015062      4436#
TST23   015346      4523#
TST24   015552      4593#
TST25   015742      4656#
TST26   016122      4712#
TST27   016420      4773#
TST3    006764      2762#
TST30   016574      4815#
TST31   017012      4878#
TST32   017142      4916#
TST33   017336      4969#
TST34   020044      5097#
TST35   020336      5188#
TST36   020660      5292#
TST37   021104      5309    5358#
TST4    007302      2878#
TST40   021312      5413#
TST41   021522      5461#
TST42   022230      5581#
TST43   022410      5627#
```

```
TST44   022616          5675#
TST45   023024          5741#
TST46   023156          5789#
TST47   023750          5962#
TST5    007676          3000#
TST50   024200          6051#
TST51   024274          6101#
TST52   024502          6144#
TST53   024742          6209#
TST54   025406          6327#
TST55   025536          6378#
TST56   026004          6461#
TST57   026062          6484#
TST6    010230          3123#
TST60   026232          6527#
TST61   026452          6579#
TST62   026520          6597#
TST63   026724          6641#
TST64   027014          6661#
TST65   027070          6682#
TST66   027432          6776#
TST67   027550          6815#
TST7    010636          3273#
TST70   027722          6850    6874#
TST71   030172          6933#
TST72   030406          7000#
TST73   031602          7174    7187#
TST74   033132          7425#
TST75   033412          7495#
TTOPT = 000400          1641#   2338    7430
TTYCHK  033150          7428#
TYPDS = 104410          8029    8036    10034#
TYPE  = 104400          2159    2163    2205    2409    2532    2535    2545    7728    7730    7739    8023    8030    8037
                        9002    9010    9055    9057    9058    9062    9079    9081    9084    9088    9092    9096    9100
                        9113    9115    9118    9120    9124    9144    9154    9171    9181    9191    9199    9201    9208
                        9215    9233    9239    9244    9254    9259    9274    9276    9293    9295    9296    9298    9301
                        9345    9431    9432    9448    9449    9460    9467    9470    9535    9611    9994    10030#  10401
TYPOC = 104402          2534    9061    9083    9087    9091    9095    9099    9141    9243    9292    9300    10031#
TYPON = 104406          10033#
TYPOS = 104404          10032#
TYPSIZ  047116          2544    9448#
TYPTIM  046656          8012    9056    9388#
UBEADR  001666          1817#   8770*   8771*   8772    8774    8775    8825    8827    10059   10061   10062
UBEBA = 170004          1611#   2000
UBECC = 170002          1610#   1999
UBECLR= 170010          1613#   2003
UBECR1= 170006          1612#   2002    2343*   2344
UBECR2= 170016          1615#   2001
UBEDB = 170000          1609#   2341    2349    2354    2359
UBEERR  044004          8788    8809#
UBEGO = 170014          1614#
UBEINI  051434          7512    8802    10044#
UBEOPT= 000200          1642#   2346
UBESAV  001662          1816#   8768*   8769*   8770    8771    10044
UBESET  033462          7499    7506#
UBESRV  043616          8761#   10054
```

DEQKCB.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
UBETBL  002230          1999#   2365    8763    8779*   8805*   8823    9279    10052   10533
UBEVEC= 000510          1616#   2004    2005
UBE2    043636          8765    8783#
UBE3    044054          8810    8815    8821#
UBM6    012732          3743    3745*   3748    3756    3760*   3767*   3769*   3775*   3783*   3790*   3796*   3803*   3809*
                        3815*   3821*   3827*   3834*   3839*   3845    3852#
UBREAK= 177770          1652#   2138*   2139    2142*   2143    2146*   2147    2150*   2151    2154    2158*   2173*   2175*
                        2177*   2179*   2192*   2198*   2203*   6581
UBRERR  002602          2140    2144    2148    2153#
UBRK2   002624          2152    2158#
UBRTBL  003160          2172    2194    2211#
UB7     016614          4807    4822#
UDPAR0= 177660          1404#   6864
UDPAR1= 177662          1405#
UDPAR2= 177664          1406#
UDPAR3= 177666          1407#
UDPAR4= 177670          1408#
UDPAR5= 177672          1409#
UDPAR6= 177674          1410#
UDPAR7= 177676          1411#
UDPDR0= 177620          1382#   6856
UDPDR1= 177622          1383#
UDPDR2= 177624          1384#
UDPDR3= 177626          1385#
UDPDR4= 177630          1386#
UDPDR5= 177632          1387#
UDPDR6= 177634          1388#
UDPDR7= 177636          1389#
UIPAR0= 177640          1393#   6609    6861    7578*   7579    7581
UIPAR1= 177642          1394#   7579*   7580*
UIPAR2= 177644          1395#   7581*   7582*
UIPAR3= 177646          1396#
UIPAR4= 177650          1397#   6880*
UIPAR5= 177652          1398#
UIPAR6= 177654          1399#   6609*
UIPAR7= 177656          1400#   7583*
UIPDR0= 177600          1371#   6853    7574*
UIPDR1= 177602          1372#   7575*
UIPDR2= 177604          1373#   7576*
UIPDR3= 177606          1374#
UIPDR4= 177610          1375#   6881*
UIPDR5= 177612          1376#
UIPDR6= 177614          1377#   6610*
UIPDR7= 177616          1378#   7577*
UM      = 140000        1649#   5321    5497    6150    6156    6182    6500    6514    6599    6733    6742
UNITNO  001554          1797#   7771*   7775    7780
USESTK= 000600          1211#
USP     =%000006        1245#
UWM6    012112          3660#   3664*   3671*   3676*   3683*   3691*   3698*   3705*   3712*   3714*   3720*   3726*   3732*
UWM7    013050          3883#   3986    3989*   3994*
UW7     013054          3881    3886#
VADR    001474          1771#   7657*   7844*   7921*   9063    9064*   9070*   9072    9074*   10112   10163*  10373*  10374*
                        10460*  10461*  10476*  10477*  10489*  10490*  10667   10678   10693   10725   10738   10756
X       053122          10407   10410#
XOR0    023324          5824    5825    5826    5829#
XOR1    023412          5849    5850    5851    5852    5854    5857#
```

```
XOR24   023446          5871#
XOR35   023716          5940    5941    5945#
XOR6    023534          5876    5879    5881    5888#
XOR6A   023540          5874*   5875*   5880*   5882*   5886    5892#   5897*   5908*   5910*
XOR6B   023542          5883*   5884*   5885*   5886    5893#
XOR7    023746          5956#
XXDP    003240          2226#   2246
XXDPC   003241          2227#   2241*   2538
$AC0    001402          1756#   7026*   7029*   7036*   7038    7047*   7048    7057*   7061*   7065*   7069*   7084*   7099*
                        7107*   7111*   7117*   7119    7125*   7127    7132*   7145*   7159*   7166*   7195*   7198*   7205*
                        7207    7216*   7217    7226*   7230*   7234*   7238*   7253*   7268*   7276*   7280*   7286*   7288
                        7294*   7296    7301*   7314*   7328*   7335*   7352    7363    7377    7382    7393    7395    9952
$AC1    001404          1757#   7025*   7029    7032    7038*   7042    7048*   7051    7058*   7072    7080    7086*   7089*
                        7090    7098*   7102    7105*   7114    7119*   7122    7127*   7129    7135    7141    7147*   7150*
                        7151    7158*   7163    7169*   7171*   7172    7194*   7198    7201    7207*   7211    7217*   7220
                        7227*   7241    7249    7255*   7258*   7259    7267*   7271    7274*   7283    7288*   7291    7296*
                        7298    7304    7310    7316*   7319*   7320    7327*   7332    7338*   7340*   7341    7352*   7363*
                        7377    7381    7394    7395*
$AC2    001406          1758#   7042*   7069    7077    7079*   7080*   7081*   7082*   7090    7102*   7111    7122*   7132
                        7138    7140*   7141*   7142*   7143*   7151    7211*   7238    7246    7248*   7249*   7250*   7251*
                        7259    7271*   7280    7291*   7301    7307    7309*   7310*   7311*   7312*   7320
$AC3    001410          1759#   7051*   7065    7077    7079    7129*   7138    7140    7220*   7234    7246    7248    7298*
                        7307    7309
$AC4    001412          1760#
$AC5    001414          1761#
$BDAOR  001224          1717#
$BDOAT  001230          1719#   8817*   8875*   10200*  10229
$BELL   001322          1749#   9002    9025
$BUFF   001400          1755#   9807*   9832
$CHARC  046652          9355*   9362    9373*   9378#
$CMTAG  001200          1704#   2290    2291    2298    2304    2305    2306
$CM1    =  000010       1731#   1732#   1733#   1734#   1735#   1736#   1737#   1738#   1739#
$CM2    =  000020       1731#   1732#   1733#   1734#   1735#   1736#   1737#   1738#   1739#
$CM3    =  000010       1729#   1731
$CM4    =  000010       1739#   1740#   1741#   1742#   1743#   1744#   1745#   1746#   1747#
$CRLF   001327          1751#   2535    8037    9010    9025    9055    9058    9100    9115    9120    9124    9276    9295
                        9296    9345    9381    9432    9470    10297   10305   10315
$DBLK   047700          9577    9611    9619#
$DB20   047710          9076    9151    9271    9632#
$DOAGN  036760          8019    8040    8046#
$DTBL   047670          9580    9615#
$ENDAD  036750          1690    2230    2407    8042#   9014
$ENDCT  036614          2304    8021#
$ENULL  036764          8048#
$EOP    036556          7982    8011#
$EOPCT  036606          2304*   8018#   8022
$ERFLG  001204          1707#   7836    8923    8945    8947    8953*   8969    8971    8976    8993    8994*   9025
$ERMAX  001217          1714#   2307*   8947    8968*   8976
$ERPSW  001534          1789#
$ERROR  044772          2136    2298    5361    5405    8992#
$ERRPC  001220          1715#   9004*   9005*   9006    9025    9059    9064
$ERRTB  002304          2044#   9110
$ERRTY  045156          9009    9054#
$ERTTL  001214          1712#   8034    8038*   9003*   9025
$ESCAP  001320          1748#   2306*   8967*   9020    9022    9025
$FACTO  001510          1779#   7631*   7747    7825    7828
```

```
$FILLC  001252          1727#   9348    9381
$FILLS  001251          1726#   9381
$FLBUF  001334          1754#   9731    9779
$FLD20  050430          9801#  10038
$FL20   050142          9726#  10037
$GDADR  001222          1716#
$GDDAT  001226          1718#   8816*   8874*  10199*  10228   10780
$GET42  036740          8039#
$HD   = 000000          1165
$HINUM  001524          1785#   2219*   8067    8145    8218    8284    9852    9860    9865*   9888    9892    10202
$ICNT   001206          1709#   8960*   8961    8963*   8975
$ILLUP  051354          9968    9999#
$ITEMB  001216          1713#   9006*   9025    9065    9067    9102
$LF     001330          1752#   9025    9381
$LONUM  001522          1784#   2220*   8068    8219    9851    9858    9864*   9893   10203
$LPADR  001210          1710#   2308*   2596*   2713*   3877*   4930*   5475*   5803*   6223*   6696*   7014*   8951*   8965*
                        8973    8975
$LPERR  001212          1711#   2155*   2309*   2594*   2595*   2596    2711*   2712*   2713    3875*   3876*   3877    4928*
                        4929*   4930    5473*   5474*   5475    5801*   5802*   5803    6221*   6222*   6223    6694*   6695*
                        6696    7012*   7013*   7014    7671*   7817    7870*   8809    8810*   8821*   8867    8868*   8877*
                        8951    8966*   8975    9019   10391   10392*  10394*  10410   10464   10465*  10467*  10469   10478
                       10479*  10481*  10482   10493   10494*  10496*  10499
$MAINT  001604          1810#   7980*   7990*   9085
$MXCNT  044770          8964    8975#
$NULL   001250          1725#   9350    9381
$NWTST= 000001          2579#   2696#   2759#   2875#   2997#   3120#   3270#   3372#   3489#   3573#   3654#   3738#   3860#
                        3981#   4070#   4193#   4312#   4433#   4520#   4590#   4653#   4709#   4770#   4811#   4813    4874#
                        4876    4913#   4966#   5094#   5185#   5287#   5289    5355#   5410#   5458#   5578#   5624#   5672#
                        5738#   5786#   5957#   5959    6048#   6077#   6079    6141#   6206#   6321#   6323    6371#   6373
                        6455#   6457    6481#   6520#   6522    6575#   6577    6594#   6638#   6658#   6679#   6771#   6773
                        6811#   6813    6868#   6870    6929#   6931    6990#   6992    7177#   7179    7422#   7491#   7493
$OCNT   047460          9507*   9536*   9549#
$OCTVL  050030          9634    9662#
$OMODE  047462          9502*   9506*   9511    9514*   9525*   9551#
$OVER   044740          8932    8952    8962    8969#
$PASS   001200          1705#   2264*   2540    7523    8015*   8016*   8027    8048    8958    8976    9097
$POWER  051362          9995   10002#
$PWRAD  051350          9997#
$PWRDN  051226          2302    9968#   9992
$PWRMG  051344          9995#
$PWRUP  051274          9977    9982#
$QUES   001326          1750#   9025
$RAND   050602          8066    8144    8217    8283    9847#   9883   10201
$RDCHR= ****** U       10035
$RDDEC= ****** U       10035
$RDLIN= ****** U       10035
$RDOCT= ****** U       10035
$REGAD  001254          1729#
$REG0   001256          1731#   7025    7036    7058    7098    7105    7125    7158    7166    7167    7169    7194    7205
                        7227    7267    7274    7294    7327    7335    7336    7338    9932
$REG1   001260          1732#   7026    7047    7057    7099    7107    7117    7159    7195    7216    7226    7268    7276
                        7286    7328
$REG2   001262          1733#   7032*   7084    7114*   7145    7163*   7201*   7253    7283*   7314    7332*  10797   10810
$REG3   001264          1734#   7072*   7086    7135*   7147    7167*   7241*   7255    7304*   7316    7336*  10797   10810
$REG4   001266          1735#  10391*  10394   10464*  10467   10478*  10481   10493*  10496
$REG5   001270          1736#   7435*   7986*   7996*  10297*  10305*  10315*  10347*  10348   10352*
```

```
$REG6   001272          1737#
$REG7   001274          1738#
$RESRE  050104          9697#  10036
$RTRN   001472          1770#   2564   6137   7985
$SAVPA  053262         10429   10443  10454#
$SAVPS  053270         10420*  10451  10455#
$SAVRE  050046          9681#  10035
$SAVR6  051360          9976*   9982   9983*  9984*  10001#
$SCOPE  044532          2296    5307   5313   5407   6636   8930#
$SETUP= 000037          2271#   2296   2298   2300   2302   2304   2305   2306   2308   2407   8013   9011
$STUP = 177777          2271#
$SVLAD  044716          8940    8965#
$SVPC = 000220          1688#   1693
$SWR  = 167377          1134#   1165   1180   1181   1182   1183   1184   1185   1186   1187   1747   1748   1749
                        2305    2306   2308   2309   2585   2702   2764   2880   3002   3125   3275   3377   3494
                        3578    3659   3743   3866   3986   4075   4198   4317   4438   4525   4595   4658   4714
                        4775    4817   4880   4919   4971   5099   5190   5294   5360   5415   5464   5583   5629
                        5677    5743   5792   5964   6053   6103   6146   6212   6329   6380   6463   6486   6529
                        6581    6599   6643   6663   6685   6778   6817   6876   6935   7002   7189   7427   7497
                        8008    8014   8041   8047   8048   8923   8924   8925   8926   8927   8931   8943   8945
                        8946    8947   8954   8955   8956   8966   8969   8975   8983   8984   8985   8986   8987
                        8988    8997   9000   9007   9011   9017   9025
$SWRMK= 000000          8927
$TIMES  001316          1747#   2305*  2583*  2700*  2880*  3002*  3864*  4917*  5462*  5790*  6210*  6683*  7002*
                        7189*   8014*  8954*  8961   8964*  8975
$TKB    001242          1722#  10293
$TKS    001240          1721#   2293   2312*
$TMPO   001276          1739#   2153*  2424*  2427   2438*  2458*  2461   2465*  2478*  2481   2492*  2498*  2501
                        2512*   6552*  7023   7037   7096   7104   7124   7156   7164   7192   7206   7265   7273
                        7293    7325   7333   7656*  7684*  7688   7693*  7710   7757   7762   7843*  7920*  8783*
                        8808*   8812   8840*  8866*  8870   9881   9896   9928  10164*  10372*  10459*  10475*  10488*
                       10498   10667  10678  10693  10725  10738  10756  10770
$TMP1   001300          1740#   2154*  7701*  7703   7706   7708*  7765   7767*  7865*  7868*  10770
$TMP2   001302          1741#   7024   7045   7054   7097   7106   7116   7157   7682*  7864*   8809*  8821   8867*
                        8877   10160*  10390*  10462*  10492*  10667  10693  10725  10756
$TMP3   001304          1742#   7862*  10161*  10377*  10463*  10693  10725  10756
$TMP4   001306          1743#   7031*  7083   7113*  7144   7162*  7193   7214   7223   7266   7275   7285   7326
                       10162*  10378*  10797
$TMP5   001310          1744#
$TMP6   001312          1745#   7071*  7134*  7165*  10797
$TMP7   001314          1746#
$TN   = 000076          1134#   1165   2579   2585#  2696   2702#  2759   2764#  2875   2880#  2997   3002#  3120
                        3125#   3270   3275#  3372   3377#  3489   3494#  3573   3578#  3654   3659#  3738   3743#
                        3860    3866#  3981   3986#  4070   4075#  4193   4198#  4312   4317#  4433   4438#  4520
                        4525#   4590   4595#  4653   4658#  4709   4714#  4770   4775#  4811   4817#  4874   4880#
                        4913    4919#  4966   4971#  5094   5099#  5185   5190#  5287   5294#  5309   5355   5360#
                        5410    5415#  5458   5464#  5578   5583#  5624   5629#  5672   5677#  5738   5743#  5786
                        5792#   5957   5964#  6048   6053#  6077   6103#  6141   6146#  6206   6212#  6321   6329#
                        6371    6380#  6455   6463#  6481   6486#  6520   6529#  6575   6581#  6594   6599#  6638
                        6643#   6658   6663#  6679   6685#  6771   6778#  6811   6817#  6850   6868   6876#  6929
                        6935#   6990   7002#  7174   7177   7189#  7422   7427#  7491   7497#
$TPB    001246          1724#   9370*  9381   10338*
$TPFLG  001253          1728#   9328   9381
$TPS    001244          1723#   2336   7433   7436*  7448   7451*  7479*  7489*  7971   7987*  7994   7997*  9368
                        9381   10298*  10306*  10316*  10334*  10336  10351*
$TRAP   051372          2135    2300   5427   5430   5434   5439   5447  10014#
```

DEQKCB.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

```
$TRP   = 000022          10021#  10031#  10032#  10033#  10034#  10035#  10036#  10037#  10038#  10039#
$TRPAD   051412          10018   10029#
$TSTNM   001202           1706#   2582#   2699#   2762#   2878#   3000#   3123#   3273#   3375#   3492#   3576#   3657#   3741#
                          3863#   3984#   4073#   4196#   4315#   4436#   4523#   4593#   4656#   4712#   4773#   4815#   4878#
                          4916#   4969#   5097#   5188#   5292#   5358#   5413#   5461#   5581#   5627#   5675#   5741#   5789#
                          5962#   6051#   6101#   6144#   6209#   6327#   6378#   6461#   6484#   6527#   6579#   6597#   6641#
                          6661#   6682#   6776#   6815#   6874#   6933#   7000#   7187#   7425#   7427    7495#   7497    7937#
                          8013#   8923    8971#   8972    8976    8993#   8996    9025    9089
$TYPBN=  ****** U        10035
$TYPDS   047464           9565#  10034
$TYPE    046434           9328#  10021   10030
$TYPEC   046600           9347    9354    9361    9366#   9369
$TYPEX   046654           9367    9374    9376    9379#
$TYPOC   047262           9505#  10031
$TYPON   047276           9504    9507#  10033
$TYPOS   047236           9500#  10032
$XTSTR   044542           8934#
$$GET4=  000000           8041#
$$TMP4   001416           1762#
$$TMP6   001420           1763#
$$TRP  = 000002          10020#  10031   10032   10033   10034   10035   10036   10037   10038   10039
$OFILL   047461           9501#   9505#   9515    9550#
.      = 057542           1656#   1660    1662#   1665#   1688    1689#   1691#   1693#   1702#   1708#   1753    1754#   1764#
                          1765#   1767#   1811#   1812#   1813#   1816#   1817#   1818#   1933#   2214#   2294    2308    2309
                          2412#   2592    2598    2709    2715    2726    2734    2742    2749    2757    2772    2782    2792
                          2800    2809    2818    2827    2836    2845    2851    2858    2865    2873    2898    2908    2918
                          2925    2932    2941    2957    3003    3013    3020    3029    3036    3042    3049    3056    3064
                          3071    3079    3084    3092    3098    3106    3113    3118    3125    3138    3144    3150    3156
                          3162    3168    3175    3182    3189    3197    3204    3208    3215    3220    3227    3234    3240
                          3244    3249    3255    3259    3265    3268    3275    3287    3292    3299    3306    3312    3317
                          3321    3329    3336    3343    3347    3353    3357    3363    3367    3370    3377    3391    3395
                          3399    3407    3415    3422    3430    3438    3444    3450    3456    3461    3466    3472    3480
                          3484    3487    3494    3507    3514    3520    3527    3535    3541    3547    3552    3558    3565
                          3571    3578    3596    3599    3605    3610    3618    3624    3629    3636    3640    3646    3652
                          3668    3674    3679    3687    3695    3701    3709    3717    3723    3729    3733    3736    3752
                          3757    3763    3772    3779    3786    3793    3799    3806    3812    3818    3824    3831    3836
                          3842    3848    3851    3873    3879    3899    3906    3914    3921    3928    3936    3943    3949
                          3956    3960    3967    3972    3979    4002    4010    4016    4023    4029    4035    4041    4048
                          4054    4061    4068    4079    4087    4093    4102    4110    4130    4138    4148    4156    4164
                          4167    4177    4182    4191    4198    4213    4222    4230    4239    4246    4253    4259    4267
                          4278    4285    4290    4297    4305    4310    4317    4335    4339    4346    4357    4362    4368
                          4373    4379    4382    4431    4450    4457    4462    4469    4480    4490    4497    4503    4511
                          4516    4525    4558    4566    4570    4576    4583    4588    4619    4627    4632    4635    4641
                          4646    4651    4685    4688    4694    4701    4707    4714    4727    4732    4736    4743    4749
                          4756    4762    4766    4789    4793    4796    4801    4805    4835    4838    4839    4845    4852
                          4860    4864    4868    4872    4904    4926    4932    4938    4944    4949    4953    4959    4964
                          5005    5010    5016    5020    5030    5040    5045    5081    5083*   5103    5105    5110    5115
                          5123    5125    5130    5146    5150    5164    5172    5183    5270    5300    5316    5341    5348
                          5365    5395    5396    5403    5418    5426    5429    5433    5438    5471    5477    5485    5560
                          5586    5638    5653    5699    5706    5714    5720    5731    5751    5766    5775    5799    5805
                          5815    5828    5839    5842    5856    5870    5887    5890    5902    5911    5915    5922    5928
                          5933    5944    5950    5955    5970    5989    6006    6010    6012    6015    6040    6046    6065
                          6070    6074    6107    6165    6180    6191    6219    6225    6248    6271    6291    6315    6338
                          6350    6369    6400    6424    6448    6479    6498    6512    6602    6616    6644    6651    6668
                          6692    6698    6724    6751    6887    6895    6923    7010    7016    7733#   7925    7972    8026#
                          8048    8049#   8191    8336    8348    8386    8427    8454    8467    8507    8548    8975    8976
```

```
                    9025    9305#   9381    9619#   9662#   9979    10000   10337   10404#  10454#  10666#  10692#  10737#
                   10779#  10809#
.PARSR  052640      2573    10156   10371#  10398
```

DEQKCB.P11 CROSS REFERENCE TABLE -- MACRO NAMES

```
COMMEN      1#   1594#
ENDCOM      1#   1594#
ERROR    1212#   1643   2156   5409   7092   7153   7175   7261   7322   7343   7659   7816   7873   7878   7922
         8814    8818   8872   8876  10165  10393  10466  10480  10495
ESCAPE      1#   1594#
FLTST1   6990#   6992   7179
HLT      1643#   2727   2735   2743   2750   2758   2773   2783   2793   2801   2810   2819   2828   2837   2846
         2852    2859   2866   2874   2899   2909   2919   2926   2933   2942   2958   2992   3014   3021   3030
         3037    3043   3050   3057   3065   3072   3080   3085   3093   3099   3107   3114   3119   3139   3145
         3151    3157   3163   3169   3176   3183   3190   3198   3205   3209   3216   3221   3228   3235   3241
         3245    3250   3256   3260   3266   3269   3288   3293   3300   3307   3313   3318   3322   3330   3337
         3344    3348   3354   3358   3364   3368   3371   3392   3396   3400   3408   3416   3423   3431   3439
         3445    3451   3457   3462   3467   3473   3481   3485   3488   3508   3515   3521   3528   3536   3542
         3548    3553   3559   3566   3572   3597   3600   3606   3611   3619   3625   3630   3637   3641   3647
         3653    3669   3675   3680   3688   3696   3702   3710   3718   3724   3730   3734   3737   3753   3758
         3764    3773   3780   3787   3794   3800   3807   3813   3819   3825   3832   3837   3843   3849   3900
         3907    3915   3922   3929   3937   3944   3950   3957   3961   3968   3973   3980   4003   4011   4017
         4024    4030   4036   4042   4049   4055   4062   4069   4080   4088   4094   4103   4111   4126   4131
         4139    4149   4157   4165   4168   4178   4183   4192   4214   4223   4231   4240   4247   4254   4260
         4268    4279   4286   4291   4298   4306   4311   4336   4347   4358   4363   4369   4374   4380   4432
         4451    4458   4463   4470   4481   4491   4498   4504   4512   4517   4559   4567   4571   4577   4584
         4589    4620   4628   4633   4636   4642   4647   4652   4686   4689   4695   4702   4708   4728   4733
         4738    4744   4750   4757   4763   4767   4790   4794   4797   4802   4806   4836   4840   4846   4853
         4861    4865   4869   4873   4939   4945   4950   4954   4960   4965   5006   5011   5017   5021   5031
         5041    5056   5058   5060   5062   5064   5066   5068   5070   5072   5074   5076   5078   5080   5082
         5084    5111   5131   5147   5165   5173   5184   5211   5236   5247   5259   5267   5271   5285   5308
         5349    5360#  5372   5397   5404   5409#  5428   5431   5435   5440   5524   5561   5595   5596   5597
         5639    5654   5659   5686   5700   5707   5715   5721   5732   5752   5767   5776   5816   5829   5843
         5857    5871   5888   5903   5912   5923   5929   5934   5945   5951   5956   5971   5990   6007   6011
         6016    6022   6026   6032   6041   6047   6067   6071   6075   6124   6129   6170   6173   6184   6196
         6249    6272   6292   6316   6351   6370   6401   6425   6449   6480   6499   6513   6547   6554   6591
         6621    6633   6648   6669   6711   6740   6760   6806   6845   6896   6921   6974   7476
MSGA     4811#   4813
MSGB     4874#   4876
MSGC     5286#   5289
MSGD     5957#   5959
MSGE     6077#   6079
MSGF     6321#   6323
MSGG     6371#   6373
MSGH     6455#   6457
MSGI     6520#   6522
MSGJ     6575#   6577
MSGK     6770#   6773
MSGL     6810#   6813
MSGM     6868#   6870
MSGN     6928#   6931
MSGO     7490#   7493
MULT        1#   1594#
MYTAGS   1695#   1753
NEWTST      1#   1135#  1594#  2579   2696   2759   2875   2997   3120   3270   3372   3489   3573   3654   3738
         3860    3981   4070   4193   4312   4433   4520   4590   4653   4709   4770   4811   4874   4913   4966
         5094    5185   5286   5355   5410   5458   5578   5624   5672   5738   5786   5957   6048   6077   6140
         6206    6321   6371   6455   6481   6520   6575   6594   6638   6658   6679   6770   6810   6868   6928
         6990    7177   7422   7490
POP         1#   1594#  9606   9702   9866   9986
PUSH        1#   1594#  9565   9682   9847   9970
```

DEQKCB.P11 CROSS REFERENCE TABLE -- MACRO NAMES

```
RELOCA   2115#  2585  2702  3866  4919  5464  5792  6212  6685  7003
RELOCB   2115#  2689  3853  4906  5451  5779  6199  6671  6983  7415
SCOPE    1213#  2584  2689  2701  2763  2879  3001  3124  3274  3376  3493  3577  3658  3742  3853
         3865   3985  4074  4197  4316  4437  4524  4594  4657  4713  4774  4816  4879  4906  4918
         4970   5098  5189  5293  5359  5414  5451  5463  5582  5628  5676  5742  5779  5791  5963
         6052   6102  6145  6199  6211  6328  6379  6462  6485  6528  6580  6598  6642  6662  6672
         6684   6777  6816  6875  6934  6983  7001  7188  7415
SETTRA  10021#  10031  10032  10033  10034  10035  10036  10037  10038
SETUP       1#  1594#  2290
SKIP        1#  1594#  2531  2539  2541  2543  5309  6849  7174
SLASH       1#  1594#
SPACE    1594#
STARS       1#  1594#  1668  1695  2028  2119  2133  2413  2422  2442  2454  2456  2476  2496  2579
         2581   2696  2698  2759  2761  2875  2877  2997  2999  3120  3122  3270  3272  3372  3374
         3489   3491  3573  3575  3654  3656  3738  3740  3860  3862  3981  3983  4070  4072  4193
         4195   4312  4314  4433  4435  4520  4522  4590  4592  4653  4655  4709  4711  4770  4772
         4811   4814  4874  4877  4913  4915  4966  4968  5094  5096  5185  5187  5287  5291  5355
         5357   5410  5412  5458  5460  5578  5580  5624  5626  5672  5674  5738  5740  5786  5788
         5957   5961  6049  6050  6077  6100  6141  6143  6206  6208  6321  6326  6371  6377  6455
         6460   6481  6483  6520  6526  6575  6578  6594  6596  6638  6640  6658  6660  6679  6681
         6771   6775  6811  6814  6868  6873  6929  6932  6990  6999  7093  7155  7177  7186  7262
         7324   7346  7351  7358  7362  7369  7375  7422  7424  7491  7494  7504  7515  7534  7539
         7607   7620  7664  7670  7960  7965  8001  8050  8053  8127  8130  8200  8203  8267  8270
         8322   8325  8439  8442  8566  8569  8665  8668  8757  8760  8834  8837  8894  8900  8917
         8976   9025  9052  9137  9146  9158  9165  9174  9184  9194  9210  9218  9231  9246  9286
         9306   9381  9387  9441  9447  9474  9552  9620  9663  9709  9725  9782  9800  9835  9870
         9878   9905  9922  9963 10005 10039 10043 10065 10072 10094 10110 10135 10146 10169 10191
        10246  10250 10257 10259 10264 10266 10275 10288 10342 10346 10355 10370 10411 10416 10436
        10439  10456 10458 10472 10474 10485 10487 10813
TRMTRP  10021#
TYPBIN      1#  1594#
TYPDEC      1#  1594#  8027  8034
TYPNAM      1#  1134#  1594#  2405
TYPNUM      1#  1594#
TYPOCS      1#  1594#
TYPOCT      1#  1594#  9059  9085  9097
TYPTXT      1#  1594#  2159  2163  2205  2545  7730  7739  8023  8030  9201  10401
$$CMRE   1695#  1731  1732  1733  1734  1735  1736  1737  1738
$$CMTM   1695#  1739  1740  1741  1742  1743  1744  1745  1746
$$ESCA      1#  1594#
$$NEWT      1#  1137#  1594#  2579  2696  2759  2875  2997  3120  3270  3372  3489  3573  3654  3738
         3860   3981  4070  4193  4312  4433  4520  4590  4653  4709  4770  4811  4874  4913  4966
         5094   5185  5287  5355  5410  5458  5578  5624  5672  5738  5786  5957  6048  6077  6141
         6206   6321  6371  6455  6481  6520  6575  6594  6638  6658  6679  6771  6811  6868  6929
         6990   7177  7422  7491
$$SET   10021#  10031  10032  10033  10034  10035  10036  10037  10038
$$SKIP      1#  1594#  5309  6850  7174
.EQUAT      1#  1134#
.HEADE      1#  1134#  1155
.KT11       1#
.SETUP      1#  1134#  2270
.SWRHI      1#  1134#  1175
.SWRLO   1134#  1188#  1191  1194
.$ACT1      1#  1134#  1668
.$CATC      1#  1134#  1653
.$CMTA      1#  1134#  1135#  1695
```

DEQKCB.P11 CROSS REFERENCE TABLE -- MACRO NAMES

ADC	2259	2796	3025	3083	3327	3555	3726	3964	4887	5747	7795	7807	7811	8769	9859
	9862	10090	10125	10235											
ADCB	3134	3194	3394	3397	3627	3769	3783	4006	4066	5071	5375	5399			
ADD	2258	2274	2367	2371	2372	2395	2397	2438	2465	2569	2595	2691	2712	2929	3744
	3855	3876	3987	4107	4173	4186	4210	4293	4302	4308	4440	4441	4447	4623	4639
	4740	4758	4856	4885	4890	4897	4908	4929	4998	4999	5000	5001	5051	5057	5087
	5100	5120	5154	5176	5192	5208	5217	5220	5230	5254	5277	5300	5316	5365	5418
	5436	5453	5474	5485	5490	5515	5530	5538	5570	5584	5589	5600	5667	5724	5736
	5781	5802	5907	6107	6115	6165	6180	6191	6201	6222	6397	6502	6539	6543	6555
	6558	6602	6613	6616	6644	6651	6674	6695	6724	6751	6786	6799	6824	6837	6887
	6903	6923	6966	6985	7013	7352	7417	7569	7571	7580	7582	7591	7593	7636	7699
	7726	7748	7777	7797	7794	7798	7801	7806	7808	7825	7828	7851	7911	7927	8073
	8085	8090	8152	8166	8224	8238	8243	8290	8298	8450	8768	8848	9077	9110	9152
	9156	9169	9197	9220	9235	9272	9288	9338	9393	9398	9403	9408	9415	9422	9429
	9454	9458	9503	9513	9585	9660	9728	9736	9742	9749	9755	9763	9769	9774	9803
	9808	9811	9818	9824	9858	9860	9861	9863	9932	9944	9945	9952	9956	10085	10089
	10120	10124	10226	10234	10244										
ADDF	7409														
ASH	6233	6259	6415	6419	6439	6443	7774	7860	7936	8080	8104	8108	8146	8154	8160
	8171	8174	8185	8232	8285	8293	8301	8404	8525	9264	9759	9765	9935	9948	9951
	10117	10225													
ASHC	2256	6284	6306	6345	7675	7858	8069	8220	9738	9745	9751	9820	9827	10083	10116
	10123	10204	10216												
ASL	2840	2885	2887	2889	2891	2936	2938	2983	3088	3356	3359	3524	3714	3911	4108
	4118	4723	4724	4734	4746	4759	4837	5059	5535	5537	5539	5541	5543	5660	5748
	5877	6237	6285	6353	6405	6544	6714	7480	7692	7708	7776	7783	7984	8882	9107
	9108	9109	9854	9943											
ASLB	3159	3246	3434	3621	3775	4038	4338	5343	9590						
ASR	2854	2922	2950	3045	3333	3518	3676	3712	3925	4855	6264	6309	6362	6901	7646
	9652	9931													
ASRB	3192	3200	3201	3419	3427	3602	3815	4045	4419	4429	5394	7436	7987	7997	10298
	10306	10316													
BCC	2747	2788	2797	2814	2833	2841	2855	2863	2923	2930	2939	3019	3033	3046	3061
	3089	3096	3111	3117	3148	3154	3160	3173	3180	3187	3202	3225	3253	3291	3297
	3316	3334	3341	3347	3360	3389	3404	3435	3442	3454	3477	3525	3532	3539	3546
	3551	3563	3595	3603	3609	3615	3622	3651	3672	3684	3692	3715	3721	3771	3776
	3797	3804	3816	3822	3828	3840	3904	3912	3926	3941	3954	3977	3999	4014	4021
	4027	4033	4039	4046	4059	4077	4123	4136	4145	4163	4174	4211	4228	4251	4257
	4264	4276	4294	4330	4355	4361	4366	4448	4455	4461	4477	4488	4495	4581	4625
	4645	4725	4741	4834	4849	4857	5028	5038	5107	5202	5249	5344	5390	5400	5661
	5763	5810	5849	5982	5987	6715	7769	9466	9591						
BCS	2329	2333	2337	2342	2350	2355	2360	2376	2380	2384	2388	2719	2768	2778	2805
	2823	2849	2870	2906	3026	3053	3068	3076	3103	3142	3166	3196	3213	3232	3238
	3247	3285	3311	3351	3398	3412	3420	3428	3448	3460	3471	3512	3556	3570	3628
	3645	3665	3677	3699	3706	3727	3749	3761	3784	3791	3810	3835	3846	3919	3948
	3965	4007	4052	4085	4092	4099	4219	4237	4303	4467	4509	4574	4617	4685	4748
	4793	4843	5426	5681	5772	5824	5900	5997	6387	6398	6471	6589	7867		
BEQ	2152	2240	2247	2268	2400	2408	2598	2715	2721	2790	2807	2816	2825	2835	2898
	2908	2918	2925	2932	2941	2957	3055	3063	3070	3079	3118	3150	3162	3168	3189
	3197	3208	3244	3259	3265	3268	3287	3305	3321	3362	3367	3370	3395	3406	3414
	3437	3450	3461	3465	3479	3484	3487	3507	3534	3565	3571	3599	3646	3652	3667
	3686	3694	3708	3736	3757	3763	3824	3830	3842	3848	3879	3899	3979	4001	4009
	4041	4054	4068	4087	4093	4110	4156	4167	4176	4182	4191	4213	4221	4230	4239
	4266	4289	4310	4332	4335	4357	4367	4373	4379	4409	4412	4421	4431	4450	4469
	4479	4490	4503	4516	4558	4566	4583	4588	4619	4632	4651	4688	4694	4707	4727
	4732	4736	4749	4756	4762	4766	4789	4796	4801	4805	4839	4851	4859	4864	4872

```
        4932    4938    4944    4949    4953    4959    4964    5005    5010    5016    5020    5030    5040    5130    5146
        5164    5172    5183    5210    5235    5270    5348    5371    5403    5429    5438    5477    5560    5586    5591
        5603    5638    5644    5647    5653    5658    5683    5699    5706    5714    5720    5731    5751    5765    5774
        5805    5812    5815    5826    5828    5842    5870    5887    5909    5911    5922    5928    5940    5944    5950
        5955    5970    5989    5994    6003    6006    6015    6031    6040    6046    6070    6074    6123    6128    6152
        6155    6172    6183    6195    6225    6240    6248    6271    6291    6315    6344    6350    6369    6393    6400
        6424    6448    6479    6498    6512    6551    6600    6608    6620    6668    6698    6710    6734    6754    6782
        6785    6820    6823    6877    6978    7016    7379    7406    7454    7475    7499    7518    7541    7679    7687
        7704    7707    7714    7722    7763    7790    7792    7813    7819    7840    7848    7943    7957    8040    8057
        8062    8098    8134    8140    8179    8191    8207    8213    8274    8279    8311    8329    8335    8345    8380
        8382    8392    8397    8423    8433    8446    8453    8463    8500    8502    8518    8554    8573    8578    8585
        8587    8614    8623    8625    8634    8636    8642    8653    8655    8672    8677    8684    8686    8707    8716
        8718    8727    8729    8735    8746    8748    8850    8946    8950    8959    8970    8995    9001    9018    9021
        9103    9105    9112    9117    9128    9130    9132    9134    9136    9143    9213    9223    9249    9341    9376
        9411    9418    9425    9530    9644    9885    9887    9891   10114   10210   10212   10222   10382   10400   10418
       10441
BGE     2732    2742    2800    2836    2851    2873    3013    6519    7078    7139    7247    7308    7384    7397    8962
        9648
BGT     2733    2749    2755    2809    2827    5280    7173    7378    7882    8019    9537    9599    9643
BHI     2726    2748    2756    7638    8948
BIC     2401    2432    2469    2486    2506    2519    2522    2527    2529    2560    2565    2578    4144    4153    4256
        4476    4500    4644    4730    4863    4903    5065    5302    5328    5352    5448    5576    5656    6119    6139
        6489    6491    6556    6574    6764    6768    6790    6795    6828    6833    6902    6944    6950    6956    6962
        7691    7855    7861    7879    7950    8016    8070    8077    8082    8087    8149    8157    8161    8163    8167
        8168    8221    8229    8233    8235    8239    8240    8244    8287    8295    8363    8482    8609    8702    8786
        8791    9265    9282    9527    9659    9735    9741    9748    9754    9762    9768    9773    9817    9823    9889
        9934    9938    9949    9953    9958   10076   10084   10088   10118   10196   10205   10215   10217   10227   10239
       10263   10271   10294   10379   10399   10421   10468
BICB    2980    4365    4376    4426    4430    4554    4587    4696    4697    4792    4799    4936    5019    5036    5710
        6267
BIS     2183    2257    2312    2330    2334    2338    2346    2377    2435    2472    2489    2491    2509    2511    2559
        2574    2893    2894    2895    2896    4135    4162    4250    4352    4460    4494    4622    4722    4787    4867
        5088    5296    5303    5329    5334    5415    5416    5483    5504    5505    5514    5544    5548    5551    5554
        5575    5641    5650    6111    6113    6177    6188    6508    6541    6545    6705    7502    7527    7775    7887
        7889    7893    8078    8092    8105    8109    8114    8158    8169    8172    8175    8186    8230    8245    8302
        8356    8368    8369    8408    8409    8475    8487    8529    8588    8602    8627    8657    8687    8695    8720
        8750    8861    8885    9532    9533    9593    9594    9939    9954    9959   10063   10385
BISB    2975    4360    4407    4410    4549    4569    4691    4692    4942    4962    5008    5026    5069    5511    6241
        6244    7451    7765    9102
BIT     2429    2483    2503    2542    2961    4122    4129    4227    4487    4631    4731    4761    4884    5321    5442
        5497    5643    5646    6104    6156    6392    6500    6514    6599    6703    6733    6742    6935    7018    7430
        7448    7453    7498    7508    7510    7517    7519    7540    7621    7641    7678    7686    7839    7903    7971
        8056    8061    8097    8133    8139    8178    8190    8206    8212    8273    8278    8340    8375    8391    8396
        8418    8432    8458    8495    8512    8517    8537    8553    8582    8584    8593    8607    8620    8622    8633
        8650    8652    8681    8683    8700    8713    8715    8726    8743    8745    8842    8849    8931    8949    8956
        9000    9007    9017   10077
BITB    4354    4408    4411    4420    4580    4698    4700    4788    4804    4889    4892    4900    4952    5004    5081
        5585    5717    6239    7446    7703    7706    9362
BLE     2724    2741    2757    2772    2818    2850    2865    3020    7091    7152    7260    7321    7342    9066    9207
BLOS    2725    2782    2845    3011    3040    3634    3934    4283
BLT     2526    2723    2734    2740    2792    2858    6505    7881    8330    8447    8574    8673    9214    9251    9353
        9538    9582    9598
BMI     2722    2771    2781    2817    2844    3056    3064    3071    3084    3098    3135    3144    3156    3215    3227
        3240    3249    3292    3306    3312    3317    3329    3336    3343    3363    3391    3407    3422    3438    3456
        3466    3472    3514    3520    3527    3541    3558    3596    3605    3624    3629    3640    3674    3679    3695
        3709    3717    3729    3733    3768    3786    3793    3799    3812    3818    3831    3836    3906    3914    3921
```

```
                3928    3943    3949    3956    3960    3972    4002    4016    4029    4035    4048    4061    4119    4130    4164
                4222    4253    4259    4278    4290    4305    4346    4362    4457    4462    4480    4497    4511    4570    4627
                4641    4646    4835    4845    4852    4868    5266    5322    5387    5433    5766    5852    5876    5899    5933
                5995    6004    6021    6388    6474    6542    7756    7850    7899    7910    8362    8481    8579    8678    8765
                9224    9367    9589    9925    9941   10314
BNE             2140    2144    2148    2235    2245    2294    2364    2392    2406    2430    2434    2471    2484    2488    2504
                2508    2518    2531    2537    2539    2541    2543    2754    2770    2780    2799    2843    2857    2872    2916
                2953    2955    2962    2982    2984    3029    3035    3048    3091    3106    3113    3175    3204    3399    3617
                3751    3778    4079    4101    4125    4138    4147    4189    4296    4377    4415    4418    4424    4427    4519
                4541    4543    4546    4548    4551    4553    4556    4699    4701    4705    4769    5109    5127    5143    5161
                5197    5199    5204    5225    5232    5251    5256    5318    5320    5323    5325    5331    5336    5346    5401
                5498    5630    5697    5704    5712    5718    5727    5729    5851    5854    5898    5979    5984    6000    6105
                6121    6157    6169    6246    6252    6269    6275    6289    6295    6313    6319    6348    6367    6391    6422
                6430    6446    6454    6473    6476    6501    6515    6586    6628    6704    6744    6746    6748    6793    6798
                6803    6831    6836    6842    6895    6905    6908    6911    6914    6918    6936    6947    6953    6959    6965
                6971    7019    7431    7434    7447    7449    7509    7511    7520    7522    7524    7526    7622    7624    7633
                7642    7653    7689    7697    7711    7736    7738    7744    7758    7815    7822    7837    7872    7875    7886
                7891    7901    7904    7916    7925    7972    7977    7993    7995    8341    8376    8419    8459    8496    8513
                8538    8542    8562    8583    8594    8608    8621    8651    8682    8701    8714    8744    8788    8796    8798
                8813    8843    8857    8859    8871    8905    8909    8932    8957    9008    9015    9068    9122    9217    9252
                9335    9343    9349    9363    9372    9453    9528    9587    9857    9899    9985   10078   10149   10153   10155
               10296   10304   10311   10326   10349
BPL             2231    2345    2463    2731    2791    2808    2826    3012    3028    3036    3042    3049    3079    3092    3105
                3182    3234    3255    3293    3353    3415    3430    3444    3480    3535    3552    3610    3618    3636    3668
                3687    3701    3723    3752    3779    3806    3936    3967    4010    4023    4102    4148    4177    4246    4267
                4285    4297    4333    4368    4576    4743    4860    5110    5205    5252    5378    5382    5392    5775    5813
                5839    5879    5881    5941    5980    5985    6338    6468    7088    7149    7170    7257    7318    7339    7452
                7466    7507    7761    7800    7897    8072    8084    8089    8113    8151    8165    8223    8237    8242    8289
                8297    8336    8343    8348    8378    8386    8421    8427    8454    8461    8467    8498    8507    8540    8548
                8597    8998    9012    9329    9369    9526    9573    9603   10207   10332   10337
BPT             5540
BR              2116    2160    2164    2206    2210    2233    2237    2347    2352    2357    2362    2378    2382    2386    2390
                2410    2453    2467    2473    2520    2528    2546    2985    3003    3125    3137    3195    3275    3377    3494
                3578    3770    3851    3881    4117    4198    4317    4382    4525    4595    4658    4714    4807    4904    4971
                5045    5085    5103    5105    5115    5123    5125    5136    5138    5150    5156    5158    5167    5169    5177
                5179    5200    5221    5226    5242    5243    5248    5262    5263    5282    5309    5326    5332    5396    5432
                5508    5518    5527    5545    5598    5605    5833    5861    5890    5902    5915    5966    5975    6010    6019
                6065    6068    6072    6125    6130    6132    6146    6174    6185    6394    6590    6592    6622    6649    6736
                6804    6809    6843    6848    6850    6897    6972    7174    7344    7388    7390    7400    7408    7477    7660
                7694    7720    7724    7731    7734    7740    7830    7852    7869    7884    7888    7892    7905    7912    8024
                8031    8065    8074    8086    8091    8143    8153    8216    8225    8282    8291    8299    8637    8730    8815
                8864    8873    8934    8940    8943    8952    8955    9071    9145    9157    9164    9173    9183    9193    9202
                9209    9226    9228    9230    9237    9241    9256    9285    9303    9331    9346    9356    9365    9374    9471
                9504    9519    9540    9584    9601    9661    9946    9979   10000   10081   10133   10167   10208   10273   10402
BVC             2739    2806    2815    2824    2834    2842    2856    2937    3027    3034    3047    3054    3077    3090    3143
                3149    3161    3181    3193    3203    3214    3248    3298    3304    3352    3421    3443    3513    3519    3533
                3547    3557    3564    3616    3678    3700    3707    3722    3728    3777    3785    3792    3811    3841    3971
                3978    4022    4028    4040    4175    4277    4295    4304    4339    4345    4575    4626    4640    4742    4838
                4850    4858    5108    5250    5281    5341    5345    5373    5377    5381    5386    5391    5749    5878    5981
                5986    6354    6389    6406    7655    7919
BVS             2720    2769    2779    2789    2798    2864    2871    3041    3062    3069    3097    3104    3112    3138    3155
                3167    3174    3188    3220    3226    3233    3239    3254    3286    3328    3335    3342    3357    3361    3390
                3405    3413    3429    3436    3449    3455    3478    3526    3540    3604    3623    3635    3666    3673    3685
                3693    3716    3750    3762    3772    3798    3805    3817    3823    3829    3847    3905    3913    3920    3927
                3935    3942    3955    3966    4000    4008    4015    4034    4047    4053    4060    4078    4086    4100    4124
                4137    4146    4212    4220    4229    4238    4252    4258    4265    4284    4331    4356    4449    4456    4468
```

```
        4478  4489  4496  4510  4582  4618  4635  4844  5203  5395  5764  5773  5811  5825  5850
        5856  5901  5996  6307  6343  6472  7835
CCC     2718  2785  2821  2903  2948  3016  3023  3302  3418  3505  3517  3561  3755  3917  3975
        4095  4128  4343  4445  4638  4739  5379  5678  5926  6001  6495
CFCC    2328
CLC     2523  3178  3185  3295  3387  3613  3910  3931  4019  4044  4234  5761  5822
CLN     3101  3523  3946  4217  4475  5104  5636  5761  5920  5948
CLR     2141  2175  2179  2186  2193  2204  2224  2225  2228  2254  2264  2272  2273  2292  2305
        2306  2317  2339  2373  2424  2425  2459  2478  2479  2498  2499  2549  2566  2568  2767
        2881  2912  2945  3007  3131  3280  3320  3384  3499  3586  3659  3745  3889  3993  4115
        4133  4203  4205  4270  4322  4324  4349  4506  4611  4612  4720  4880  4881  4941  4989
        4995  4996  5113  5133  5206  5215  5295  5338  5363  5408  5481  5506  5533  5534  5566
        5577  5587  5632  5645  5649  5662  5694  5807  5837  5938  5965  5968  5973  6112  6138
        6162  6178  6229  6282  6283  6303  6304  6331  6335  6336  6383  6409  6410  6412  6432
        6433  6436  6488  6532  6534  6553  6570  6572  6604  6657  6702  6716  6738  6749  6757
        6762  6769  6789  6801  6818  6827  6840  6878  6879  6880  6881  6882  6883  6890  6942
        6980  6981  6982  7412  7428  7458  7479  7489  7529  7564  7598  7601  7640  7645  7672
        7674  7682  7684  7685  7753  7766  7787  7803  7804  7805  7842  7846  7857  7907  7939
        7949  7968  7979  7980  7981  8013  8014  8038  8063  8064  8099  8117  8118  8131  8142
        8195  8196  8214  8215  8253  8254  8280  8281  8308  8309  8398  8434  8560  8783  8826
        8840  8863  8954  8967  9073  9101  9391  9396  9401  9406  9413  9420  9427  9451  9517
        9576  9579  9639  9957  9983 10045 10046 10047 10048 10053 10115 10122 10131 10351 10383
       10428 10497
CLRB    2265  2287  2343  3243  4242  4387  4389  4391  4393  5023  5449  5516  5733  5778  6149
        6573  6634  6656  7469  7717  7723  7959  8055  8132  8138  8205  8211  8272  8907  8910
        8953  9090  9373  9605  9809 10324 10334
CLRD    5079
CLRF    5063
CLV     2812  2831  3140  3218  3704  3731  3782  5194  5383  5388  5393  5822
CLZ     2766  2776  3283  3747  3897  4090  4143  4226  4234  4465  4586  4615  5655
CMP     2139  2143  2147  2151  2293  2407  2433  2470  2487  2507  2897  2981  3267  3366  3369
        3483  3486  3662  3735  4098  4109  4155  4180  4181  4188  4190  4218  4263  4454  4502
        4518  4540  4542  4545  4547  4550  4552  4555  4557  4602  4604  4634  4650  4667  4687
        4690  4693  4726  4735  4765  4768  4823  4825  4828  4842  4871  4888  4891  4899  4901
        4937  4943  4948  4958  4980  5029  5123  5139  5144  5145  5163  5198  5209  5231  5234
        5244  5255  5268  5269  5317  5319  5324  5330  5335  5437  5502  5559  5602  5652  5696
        5698  5705  5711  5713  5719  5730  5750  5762  5827  5832  5841  5860  5886  5943  5954
        5999  6069  6073  6120  6122  6171  6182  6194  6245  6251  6268  6274  6288  6290  6294
        6296  6312  6314  6318  6320  6347  6349  6366  6368  6390  6399  6421  6429  6445  6453
        6478  6504  6518  6550  6585  6667  6709  6747  6752  6792  6797  6830  6835  6904  6907
        6917  6946  6952  6958  6964  6969  6970  7077  7090  7138  7151  7172  7246  7259  7307
        7320  7341  7377  7383  7396  7474  7637  7652  7696  7791  7812  7880  7885  7915  7948
        7976  8071  8083  8088  8150  8164  8222  8236  8241  8288  8296  8328  8334  8445  8452
        8572  8577  8671  8676  8787  8795  8797  8856  8858  8941  8961  9014  9104  9206  9212
        9216  9222  9248  9250  9597  9884  9886  9890  9898 10148 10152 10154 10206 10241 10295
       10325
CMPB    2399  2517  2525  4344  4372  4414  4417  4423  4515  4544  4573  4704  4706  4795  4800
        4883  5015  5441  5726  5728  5771  6247  6270  6423  6447  6497  6511  6910  6913  7433
        7521  7814  7994  8904  8908  8947  9065  9127  9129  9131  9133  9135  9340  9342  9348
        9371  9375  9647 10211 10221 10303 10310
COM     2787  2917  2946  3095  3290  3545  3671  3953  4152  4154  4160  4170  4207  4235  4274
        4300  4473  4485  4501  4507  4637  4649  4760  4764  5124  5137  5157  5168  5178  5227
        5709  5821  5840  5868  5885  5942  6133  6339  6360  6363  6629  6794  6832  6948  6960
       10150 10151 10195 10313 10388
COMB    3186  3252  3388  3594  3796  4013  4067  4326  4371  4956  5374
DEC     2813  2915  2952  3018  3116  3350  3531  3720  3918  4534  5998  6273  6317  6451  6452
```

```
        6517  6802  6841  7770  8017  8327  8383  8424  8444  8503  8544  8571  8626  8656  8670
        8719  8749  9106  9642
DECB    3172  3207  3219  3224  3441  3650  3827  4032  4341  4425  5013  5385  9352  9355  9525
        9536
DIV     6325  6470  9392  9397  9402  9407  9414  9421  9428
DIVF    7364
EMT     1212  5370
HALT    1660  2167  2209  6647  8999  9013  9016  9330  9978  9999 10300
INC     2363  2391  2405  2492  2512  2530  2822  2907  2924  2954  2956  2968  3060  3130  3303
        3383  3550  3568  3590  3705  3970  3991  4116  4166  4208  4325  4672  4677  4781  4983
        4987  5014  5049  5126  5142  5160  5171  5182  5224  5604  5682  5759  5814  5910  6020
        6043  6117  6250  6293  6352  6357  6427  6428  6467  6503  6619  7483  7693  7705  7718
        7737  7768  8015  8333  8576  8675  8911  8960  9003  9139  9148  9160  9167  9176  9186
        9468  9531  9539  9583  9645  9856  9984 10347
INCB    3136  3141  3262  3425  3476  3608  3767  3821  4026  4064  4065  4334  4413  4963  5009
        5389  5536  5702  7973  8349  8387  8428  8468  8508  8549  8561  8590  8629  8659  8688
        8721  8751  8903  8906  8994  9377
IOT     1213  5306  5360  6618
JMP     1664  1666  1667  2551  2599  2601  2716  3880  4933  5102  5122  5141  5159  5170  5180
        5478  5806  6226  6699  7017  7020  7432  7542  7606  7649  7680  7750  7754  7759  7764
        7796  7817  7820  7838  7841  7876  7877  7902  7923  7926  8047  8059  8136  8209  8276
        8331  8332  8337  8448  8449  8455  8543  8575  8674  9277 10245 10406 10410
JSR     2544  5195  5223  5246  5265  5279  5482  5542  5633  5671  6064  7021  7027  7030  7040
        7049  7059  7062  7067  7070  7085  7100  7108  7112  7120  7128  7133  7146  7160  7161
        7168  7190  7196  7199  7209  7218  7228  7231  7236  7239  7254  7269  7277  7281  7289
        7297  7302  7315  7329  7330  7337  7355  7366  7411  7512  7552  7654  7658  7834  7918
        7969  8012  8042  8066  8101  8115  8144  8182  8217  8251  8283  8306  8370  8399  8401
        8435  8465  8505  8520  8522  8546  8556  8589  8615  8628  8643  8658  8708  8736  8762
        8767  8773  8777  8792  8800  8802  8803  8830  8839  8844  8890  8902  8912  9009  9056
        9069  9076  9149  9151  9269  9271  9347  9354  9361  9389  9433  9883  9901 10060 10201
       10321 10329
LDEXP   7387  7389  7399  7403  9960
LDF     7023  7024  7037  7039  7045  7046  7055  7056  7060  7066  7068  7083  7096  7097  7104
        7106  7110  7116  7118  7124  7126  7131  7144  7156  7157  7164  7192  7193  7206  7208
        7214  7215  7224  7225  7229  7235  7237  7252  7265  7266  7273  7275  7279  7285  7287
        7293  7295  7300  7313  7325  7326  7333
LDFPS   7022  7094  7191  7263
MARK    2178  6062
MFPD    5061  5077
MFPI    6624  6625  6626
MOV     2115  2134  2135  2136  2137  2138  2142  2145  2146  2149  2150  2153  2154  2155  2158
        2168  2169  2170  2171  2172  2177  2180  2181  2182  2184  2185  2187  2188  2189  2192
        2194  2195  2196  2197  2199  2218  2219  2220  2248  2249  2251  2255  2260  2261  2263
        2266  2269  2271  2276  2277  2278  2279  2280  2282  2283  2284  2290  2291  2295  2296
        2297  2298  2299  2300  2301  2302  2303  2304  2308  2309  2313  2314  2315  2316  2321
        2322  2323  2324  2325  2351  2356  2361  2365  2366  2381  2385  2389  2393  2394  2402
        2403  2404  2423  2426  2427  2428  2431  2437  2448  2451  2457  2458  2460  2461  2468
        2477  2480  2482  2485  2497  2500  2502  2505  2533  2554  2556  2557  2558  2561  2562
        2563  2564  2570  2572  2573  2575  2576  2577  2583  2588  2590  2591  2593  2594  2596
        2690  2692  2700  2705  2707  2708  2710  2711  2713  2880  2884  2886  2888  2890  2892
        2911  2921  2935  2947  2951  2963  2965  2969  2970  2971  2972  2973  2974  2976  2977
        2979  2986  2987  2988  2989  2990  2991  2994  3002  3005  3127  3129  3277  3279  3379
        3381  3382  3497  3500  3502  3503  3582  3585  3589  3591  3592  3593  3661  3743  3854
        3856  3864  3869  3871  3872  3874  3875  3877  3886  3890  3891  3894  3895  3986  3988
        3989  3994  4076  4082  4091  4105  4106  4120  4127  4141  4151  4159  4161  4169  4179
        4185  4187  4201  4204  4245  4273  4288  4301  4320  4323  4385  4388  4390  4392  4438
```

4439 4442 4443 4444 4483 4484 4513 4514 4528 4532 4533 4535 4536 4560 4563
4600 4601 4603 4605 4606 4607 4609 4610 4666 4668 4669 4671 4673 4674 4676
4678 4681 4718 4719 4721 4729 4753 4754 4778 4779 4780 4782 4784 4786 4822
4824 4826 4827 4829 4832 4833 4882 4907 4909 4917 4922 4924 4925 4927 4928
4930 4934 4935 4979 4981 4982 4985 4986 4990 4991 4992 4997 5043 5044 5048
5050 5052 5091 5092 5093 5099 5114 5118 5119 5121 5134 5135 5149 5153 5155
5175 5190 5191 5207 5214 5216 5218 5219 5229 5239 5241 5253 5264 5276
5294 5297 5298 5299 5301 5304 5305 5307 5311 5313 5314 5315 5353 5354 5361
5362 5364 5366 5368 5405 5406 5407 5417 5419 5421 5423 5427 5430 5434 5439
5445 5446 5447 5452 5454 5462 5467 5469 5470 5472 5473 5475 5480 5484 5486
5487 5488 5491 5492 5494 5495 5496 5503 5507 5512 5513 5517 5521 5522 5523
5531 5558 5564 5565 5567 5569 5572 5573 5574 5583 5588 5590 5592 5593 5599
5623 5631 5634 5635 5640 5642 5648 5651 5657 5663 5664 5666 5669 5677 5684
5685 5688 5691 5692 5723 5725 5734 5735 5743 5745 5757 5758 5780 5782 5790
5795 5797 5798 5800 5801 5803 5818 5819 5820 5831 5845 5846 5859 5863 5866
5873 5883 5896 5905 5906 5914 5919 5925 5936 5977 5992 5993 6009 6018 6028
6053 6054 6055 6056 6057 6058 6059 6060 6061 6062 6063 6076 6103 6106 6108
6109 6110 6114 6131 6134 6136 6137 6148 6150 6153 6160 6161 6163 6164 6167
6179 6189 6190 6198 6200 6202 6210 6215 6217 6218 6220 6221 6223 6227 6228
6230 6231 6232 6254 6255 6256 6257 6258 6277 6278 6279 6280 6281 6298 6299
6300 6301 6302 6329 6330 6332 6333 6337 6356 6358 6361 6380 6381 6382 6395
6411 6413 6414 6416 6437 6438 6440 6463 6465 6466 6477 6486 6490 6493 6494
6506 6529 6531 6533 6535 6538 6552 6557 6571 6581 6582 6583 6584 6593 6601
6603 6605 6606 6609 6610 6611 6612 6615 6617 6635 6636 6637 6643 6645 6646
6650 6654 6655 6664 6670 6673 6675 6683 6688 6690 6691 6693 6694 6696 6701
6707 6708 6723 6725 6726 6727 6729 6730 6731 6735 6739 6745 6750 6758 6759
6761 6763 6765 6766 6779 6780 6781 6783 6787 6788 6791 6796 6817 6819 6821
6825 6826 6829 6834 6884 6885 6886 6888 6889 6891 6892 6893 6899 6906 6909
6912 6915 6922 6925 6926 6927 6937 6938 6939 6940 6941 6943 6945 6949 6951
6955 6957 6961 6963 6979 6984 6986 7002 7006 7008 7009 7011 7012 7014 7025
7026 7029 7032 7036 7038 7042 7047 7048 7051 7054 7057 7058 7061 7065 7069
7072 7079 7084 7098 7099 7102 7105 7107 7111 7114 7117 7119 7122 7125 7127
7129 7132 7135 7140 7145 7158 7159 7163 7166 7167 7189 7194 7195 7198 7201
7205 7207 7211 7216 7217 7220 7223 7226 7227 7230 7234 7238 7241 7248 7253
7267 7268 7271 7274 7276 7280 7283 7286 7288 7291 7294 7296 7298 7301 7304
7309 7314 7327 7328 7332 7335 7336 7354 7365 7376 7381 7393 7395 7410 7416
7418 7429 7435 7455 7456 7457 7463 7484 7488 7500 7501 7513 7528 7530 7531
7549 7551 7555 7556 7557 7558 7559 7560 7561 7562 7565 7566 7567 7568 7570
7572 7574 7575 7576 7577 7578 7579 7581 7583 7585 7586 7587 7588 7589 7590
7592 7594 7595 7596 7599 7600 7603 7604 7605 7625 7626 7628 7629 7631 7634
7635 7643 7644 7648 7651 7656 7657 7661 7662 7671 7673 7676 7677 7681 7683
7695 7698 7700 7701 7702 7715 7716 7725 7727 7745 7746 7747 7749 7751 7752
7771 7772 7773 7778 7780 7781 7782 7784 7785 7786 7802 7823 7824 7827 7831
7832 7833 7843 7844 7845 7859 7862 7863 7864 7865 7870 7894 7908 7914 7917
7920 7921 7928 7929 7930 7935 7938 7940 7941 7944 7955 7958 7967 7970 7978
7982 7985 7986 7988 7989 7990 7991 7996 8020 8027 8034 8039 8060 8067 8068
8075 8076 8079 8081 8093 8094 8095 8100 8102 8103 8106 8107 8110 8116 8119
8121 8122 8123 8124 8137 8145 8148 8155 8156 8159 8162 8170 8173 8176 8180
8181 8183 8184 8187 8189 8192 8193 8194 8197 8210 8218 8219 8227 8228 8231
8234 8246 8247 8248 8252 8258 8259 8260 8261 8262 8263 8264 8277 8284 8286
8292 8300 8303 8307 8315 8316 8317 8318 8319 8350 8351 8353 8354 8355 8357
8360 8367 8371 8388 8389 8400 8402 8403 8405 8406 8407 8410 8411 8412 8413
8414 8429 8430 8464 8469 8470 8472 8473 8474 8476 8479 J486 8488 8489 8490
8491 8504 8509 8510 8519 8521 8523 8524 8526 8527 8528 8530 8531 8532 8533
8545 8550 8551 8555 8591 8592 8595 8599 8600 8601 8603 8606 8630 8631 8644
8645 8660 8661 8689 8690 8692 8693 8694 8696 8699 8722 8723 8737 8738 8752

```
        8753  8763  8766  8770  8771  8772  8774  8775  8776  8779  8785  8789  8790  8799  8805
        8808  8809  8810  8811  8816  8817  8821  8822  8823  8824  8825  8827  8828  8829  8841
        8852  8853  8862  8866  8867  8868  8869  8874  8875  8877  8878  8879  8880  8881  8886
        8888  8889  8914  8936  8937  8939  8942  8951  8963  8964  8965  8966  8972  8973  8996
        9004  9019  9022  9059  9063  9064  9070  9072  9074  9075  9078  9082  9085  9093  9097
        9111  9116  9121  9126  9140  9150  9153  9161  9162  9163  9168  9170  9177  9180  9187
        9190  9196  9198  9205  9221  9225  9227  9229  9234  9236  9240  9242  9255  9258  9261
        9263  9266  9270  9273  9279  9280  9281  9289  9290  9291  9297  9299  9332  9333  9337
        9350  9395  9400  9405  9412  9419  9426  9450  9457  9459  9464  9500  9508  9509  9510
        9516  9523  9541  9542  9543  9544  9545  9566  9567  9568  9569  9570  9571  9572  9577
        9580  9600  9606  9607  9608  9609  9610  9612  9613  9633  9634  9635  9636  9637  9638
        9641  9646  9682  9683  9684  9685  9686  9687  9688  9689  9690  9691  9698  9699  9700
        9701  9702  9703  9704  9705  9706  9707  9727  9729  9730  9731  9733  9734  9740  9746
        9747  9753  9760  9761  9767  9777  9778  9779  9802  9806  9807  9810  9812  9813  9814
        9815  9816  9822  9830  9831  9832  9848  9849  9850  9851  9852  9853  9864  9865  9866
        9867  9868  9879  9880  9881  9882  9888  9892  9893  9896  9897  9900  9902  9923  9924
        9926  9928  9933  9937  9950  9961  9968  9969  9970  9971  9972  9973  9974  9975  9976
        9977  9982  9986  9987  9988  9989  9990  9991  9992  9993  9996  10014 10015 10018 10044
       10052 10054 10055 10056 10058 10059 10061 10062 10074 10075 10079 10080 10082 10086 10087
       10112 10121 10126 10127 10132 10147 10156 10160 10161 10162 10163 10164 10166 10192 10193
       10194 10199 10200 10202 10203 10214 10218 10219 10220 10228 10229 10232 10233 10236 10237
       10240 10251 10252 10254 10260 10261 10267 10272 10293 10297 10305 10315 10320 10322 10327
       10328 10352 10371 10372 10373 10375 10376 10377 10378 10380 10384 10389 10390 10391 10392
       10394 10395 10396 10397 10398 10407 10419 10420 10422 10423 10424 10425 10426 10427 10429
       10430 10431 10432 10433 10442 10443 10444 10445 10446 10447 10448 10449 10450 10451 10452
       10459 10460 10462 10463 10464 10465 10467 10469 10475 10476 10478 10479 10481 10482 10488
       10489 10491 10492 10493 10494 10496 10498 10499
MOVB    2173  2198  2203  2232  2236  2241  2285  2288  2289  2307  2398  2440  2474  2481  2494
        2501  2514  2521  2582  2699  2762  2878  2904  2928  3000  3123  3273  3375  3492  3576
        3657  3741  3863  3984  4073  4196  4315  4329  4400  4402  4404  4428  4436  4523  4539
        4561  4562  4564  4565  4593  4656  4682  4683  4684  4712  4752  4773  4785  4815  4878
        4916  4947  4957  4969  5003  5034  5097  5188  5292  5358  5413  5461  5532  5581  5627
        5670  5675  5679  5695  5741  5789  5962  6051  6101  6144  6209  6234  6238  6242  6260
        6266  6327  6378  6420  6444  6461  6484  6527  6536  6537  6540  6579  6597  6641  6661
        6682  6767  6776  6815  6874  6933  7000  7187  7425  7427  7450  7464  7495  7497  7532
        7597  7690  7779  7797  7854  7883  7937  7974  8096  8177  8249  8255  8294  8304  8312
        8346  8366  8384  8395  8425  8437  8485  8516  8558  8612  8616  8640  8646  8663  8705
        8709  8733  8739  8755  8846  8892  8968  8971  8993  9006  9089  9334  9360  9370  9390
        9394  9399  9404  9409  9416  9423  9430  9462  9463  9501  9502  9505  9506  9507  9511
        9514  9515  9534  9575  9578  9592  9595  9604  9640  9649  9732  9737  9743  9744  9750
        9756  9757  9758  9764  9770  9771  9772  9775  9819  9825  9826  9942  10017 10224 10253
       10323 10338 10348
MTPI    6630
MUL     6341  6364  6396
MULF    7353
NEG     2832  2868  3110  3315  3538  3683  3940  4472  4630  5855  6263  6308  6434  7082  7089
        7143  7150  7171  7251  7258  7312  7319  7340  7386  7398  7647  9512  9574  9930
NEGB    3165  3453  3649  3803  4051  5083  5380
NOP     4893  4898  6546  6614  6623  6627  7426  7460  7461  7462  7467  7496  8043  8044  8045
RESET   6666  8041  8326  8443  8570  8669 10405
ROL     2524  2848  2883  2905  3032  3039  3340  3562  3664  3976  6286  6588  7810  7868  9518
        9520  9521  9522  9524  9855 10119
ROLB    3147  3411  3633  3809  3839  4058  4399  4401  4403  4405  4406  4416  5027  5340
ROR     2369  2370  2436  2464  2490  2510  2804  2914  2949  3010  3052  3296  3325  3511  3691
        3933  4172  4272  4351  5680  5974  6310  7767  7856  9465  9653  9654  9655  9656  9657
        9658
```

```
RORB    3179  3212  3403  3614  3790  4020  4340  4342  4422  5037  5376
RTI     5089  5398  5443  5549  5555  5571  5601  5648  5668  5737  6131  6135  6181  6632  6652
        6919  6924  7481  7485  7945  7983  7998  8120  8256  8313  8352  8358  8364  8372  8390
        8415  8431  8438  8471  8477  8483  8492  8511  8534  8552  8559  8598  8604  8610  8617
        8632  8647  8662  8664  8691  8697  8703  8710  8724  8740  8754  8756  8780  8806  8832
        8847  8893  8915  8974  9024  9339  9546  9614  9692  9708  9998  10268 10301 10308 10317
       10335 10339 10353 10409 10470 10483 10500
RTS     5201  5228  5258  5273  5284  5552  6066  7356  7367  7413  8125  8265  8320  8563  9125
        9379  9435  9456  9651  9869  9903  9962  10019 10064 10092 10129 10157 10159 10242 10255
       10269 10434 10453
RTT     1770  6116  6120  6136  6166  6192  6559  8198  9780  9833
SBC     2862  3067  3075  3346  3569  3698  3903  8794  8855  8884  9268  9284  10231
SBCB    3153  3264  3447  3644  3760  3998
SCC     2765  2775  2811  2830  2861  2882  3171  3230  3282  3386  3410  3426  3475  3530  3632
        3663  3682  3690  3746  3896  3902  3909  3963  4005  4075  4121  4134  4142  4216  4225
        4233  4281  4328  4464  4474  4614  5054  5101  5193  5245  5278  5760  5770  5808  5847
        5978  6340  6384  6509  6663  6665
SEC     2326  2331  2335  2340  2348  2353  2358  2374  2746  2795  2803  2839  2913  3009  3017
        3024  3059  3074  3082  3087  3133  3211  3223  3263  3309  3324  3326  3402  3469  3510
        3643  3713  3766  3789  3997  4171  4249  4271  4350  4353  4397  4579  4643  4680  4783
        4831  5339  5369  5384  5422  6587
SEN     2730  4097  4244  4353  5272  5420  5931
SEV     2738  2786  3109  3332  3339  3402  3433  3544  3601  3802  3924  3932  3939  3952  3997
        4057  4083  4097  4209  4249  4262  4353  4446  4453  4486  4493  5367  6305  10158
SEZ     2753  2786  3433  5369  5424
SOB     2176  2191  2202  2275  2281  2286  2368  2396  2439  2466  2493  2513  5983  6002  6012
        6025  6029  6044  6243  6265  6287  6311  6800  6838  6968  7709  7712  7719  7809  9245
        9294  9302  9455  9469  9739  9752  9766  9821  9828  9894  9895  10223
SPL     6487  6493  6494  6496  6507  6510  10197
STEXP   9955
STF     7028  7031  7041  7050  7071  7101  7113  7121  7134  7162  7165  7197  7200  7210  7219
        7240  7270  7282  7290  7303  7331  7334
SUB     2250  2592  2709  3006  3128  3278  3380  3498  3873  4084  4236  4275  4282  4466  4508
        4616  4624  4747  4755  4848  4896  4902  4926  5055  5067  5073  5471  5746  5799  5884
        6219  6469  6516  6692  7010  7080  7081  7086  7141  7142  7147  7169  7249  7250  7255
        7310  7311  7316  7338  7363  7382  7394  7630  7639  7799  7849  7866  7975  8784  8793
        8851  8854  8883  9005  9094  9267  9283  9581  9927  9929  9936  9940  10130 10230 10374
       10461 10477 10490
SUBF    7407
SWAB    2869  3231  3258  3310  3470  3639  3732  3834  3947  3959  4243  4248  5033  6900
SXT     5809  5836  5864  5874  5897  5908  5921  5927  5932  5949  6464
TRAP    5425  10021 10031 10032 10033 10034 10035 10036 10037 10038
TST     2230  2239  2244  2246  2267  2332  2336  2341  2349  2354  2359  2375  2379  2383  2387
        2449  2452  2462  2536  2567  2571  2589  2706  2777  2931  2940  3102  3284  3501  3506
        3583  3584  3587  3588  3845  3870  3887  3888  3892  3893  3898  3990  3992  4202  4309
        4321  4386  4529  4537  4608  4670  4675  4923  4984  5039  5047  5075  5117  5128  5140
        5152  5162  5196  5233  5257  5347  5402  5468  5501  5637  5703  5796  5835  5838  5853
        5867  5918  5947  5969  5988  6005  6014  6030  6034  6045  6126  6168  6216  6475  6689
        6894  6916  6954  6967  6977  7007  7405  7632  7688  7710  7713  7721  7757  7760  7762
        7818  7871  7898  7900  7909  7956  8112  8342  8361  8377  8379  8420  8436  8460  8480
        8497  8499  8539  8557  8635  8728  8764  8801  8812  8860  8870  8887  8938  8997  9011
        9020  9142  9262  9336  9344  9364  9366  9410  9417  9424  9529  9586  9596  10016 10209
       10299 10307 10312 10331 10333 10350
TSTB    2344  2538  2540  2597  2714  3237  3459  3464  3598  3748  3756  3878  4378  4886  4931
        5476  5629  5804  6224  6607  6697  6784  6822  6876  7015  7506  7523  7525  7623  7735
        7743  7755  7789  7821  7836  7847  7874  7890  7896  7924  7942  7992  8310  8344  8347
```

```
            8381    8385    8422    8426    8462    8466    8501    8506    8541    8547    8586    8596    8613    8624    8641
            8654    8685    8706    8717    8734    8747    8945    8958    8969    9067    9328    9368    9452    9588    9602
           10113   10336   10381   10386   10417   10440
TSTF        2327
WAIT        2550    7437
XOR         5823    5848    5865    5869    5875    5880    5882    5937    5939    5953    6151    6154
.ABS        1134
.ASCII      1750    1751   10558   10645   10815   10820   10823   10827   10831   10836   10840   10845   10849   10853   10860
           10869   10874   10881   10887   10891   10896   10903   10909   10912   10915   10918   10921   10924   10928
.ASCIZ      1749    1752    2162    2166    2208    2412    2548    7733    7742    8026    8033    9204    9304   10002   10404
           10544   10546   10548   10569   10572   10573   10574   10575   10576   10589   10602   10608   10611   10623   10631
           10642   10647   10649   10652   10654   10658   10669   10673   10680   10685   10695   10699   10709   10715   10727
           10734   10739   10746   10758   10765   10771   10776   10781   10786   10789   10793   10801   10803   10931
.BLKB       1754    9662
.BLKW       1764    1765    1767    1811    1812    1813    1816    1817    1818    9619   10454
.BYTE       1707    1713    1714    1725    1726    1727    1728    1774    1775    1776    1777    1786    1928    1929    1930
            1931    1932    2211    2226    2227    8048    9436    9472    9547    9548    9549    9550   10664   10707   10722
           10753   10768   10778   10799   10807
.ENABL         1
.END       10934
.ENDC       1160    1184    1186    1187    1188    1204    1593    1665    1669    1691    1693    1696    1703    1729    1739
            1747    1748    1749    1750    2028    2029    2120    2134    2162    2166    2208    2271    2295    2296    2298
            2300    2302    2304    2305    2306    2308    2310    2407    2409    2412    2414    2423    2443    2455    2457
            2477    2497    2532    2540    2542    2544    2548    2580    2581    2582    2584    2585    2697    2698    2699
            2701    2702    2760    2761    2762    2763    2764    2876    2877    2878    2879    2880    2998    2999    3000
            3001    3002    3121    3122    3123    3124    3125    3271    3272    3273    3274    3275    3373    3374    3375
            3376    3377    3490    3491    3492    3493    3494    3574    3575    3576    3577    3578    3655    3656    3657
            3658    3659    3739    3740    3741    3742    3743    3861    3862    3863    3865    3866    3982    3983    3984
            3985    3986    4071    4072    4073    4074    4075    4194    4195    4196    4197    4198    4313    4314    4315
            4316    4317    4434    4435    4436    4437    4438    4521    4522    4523    4524    4525    4591    4592    4593
            4594    4595    4654    4655    4656    4657    4658    4710    4711    4712    4713    4714    4771    4772    4773
            4774    4775    4812    4813    4814    4815    4816    4817    4875    4876    4877    4878    4879    4880    4914
            4915    4916    4918    4919    4967    4968    4969    4970    4971    5095    5096    5097    5098    5099    5186
            5187    5188    5189    5190    5288    5289    5291    5292    5293    5294    5310    5356    5357    5358    5359
            5360    5411    5412    5413    5414    5415    5459    5460    5461    5463    5464    5579    5580    5581    5582
            5583    5625    5626    5627    5628    5629    5673    5674    5675    5676    5677    5739    5740    5741    5742
            5743    5787    5788    5789    5791    5792    5958    5959    5961    5962    5963    5964    6049    6050    6051
            6052    6053    6078    6079    6100    6101    6102    6103    6142    6143    6144    6145    6146    6207    6208
            6209    6211    6212    6322    6323    6326    6327    6328    6329    6372    6373    6377    6378    6379    6380
            6456    6457    6460    6461    6462    6463    6482    6483    6484    6485    6486    6521    6522    6526    6527
            6528    6529    6576    6577    6578    6579    6580    6581    6595    6596    6597    6598    6599    6639    6640
            6641    6642    6643    6659    6660    6661    6662    6663    6680    6681    6682    6684    6685    6772    6773
            6775    6776    6777    6778    6812    6813    6814    6815    6816    6817    6851    6869    6870    6873    6874
            6875    6876    6930    6931    6932    6933    6934    6935    6991    6992    6999    7000    7001    7002    7094
            7156    7175    7178    7179    7186    7187    7188    7189    7263    7325    7347    7352    7359    7363    7370
            7376    7423    7424    7425    7426    7427    7492    7493    7494    7495    7496    7497    7505    7516    7535
            7540    7608    7621    7665    7671    7733    7742    7961    7966    8002    8005    8006    8008    8010    8013
            8019    8022    8023    8026    8033    8039    8041    8047    8048    8049    8051    8054    8128    8131    8201
            8204    8268    8271    8323    8326    8440    8443    8567    8570    8666    8669    8758    8761    8835    8838
            8895    8901    8918    8923    8927    8931    8933    8944    8945    8947    8949    8956    8960    8965    8969
            8975    8976    8977    8983    8994    9000    9004    9010    9011    9017    9024    9025    9026    9053    9138
            9147    9159    9166    9175    9185    9195    9204    9211    9219    9232    9247    9287    9307    9382    9388
            9442    9448    9475    9553    9621    9664    9710    9726    9783    9801    9836    9871    9879    9906    9923
            9964    9976    9986    9996    9998   10005   10006   10015   10018   10020   10030   10031   10032   10033   10034
           10035   10036   10037   10038   10040   10044   10066   10073   10095   10111   10136   10147   10170   10192   10247
           10251   10258   10260   10265   10267   10276   10289   10343   10347   10356   10371   10404   10412   10417   10437
```

```
          10440   10457   10459   10473   10475   10486   10488   10814
.EQUIV     1212    1213    1215    1236    1237    1238    1239    1240    1241    1242    1243    1244    1245    1246    1275
           1276    1277    1278    1279    1280    1281    1282    1283    1284    1303    1304    1305    1306    1307    1308
           1309    1310    1311    1312    1364    1365    1366    1367    1575    1576    1577    1578    1579    1580    1581
           1582    1583    1584    1585    1586    1587    1588    1589    1590    1643    1644    1645    1646    5360    5409
.EVEN      1708    1933    2162    2166    2208    2214    2412    2548    7733    7742    8026    8033    8049    9204    9305
           9440    9781   10004   10404   10523   10666   10677   10692   10724   10737   10755   10769   10779   10796   10809
.IF        1156    1183    1185    1186    1187    1204    1593    1660    1668    1690    1692    1695    1702    1729    1739
           1747    1748    1749    1753    2028    2119    2133    2161    2165    2207    2271    2290    2295    2296    2298
           2300    2302    2304    2305    2306    2308    2406    2407    2408    2411    2413    2422    2442    2454    2456
           2476    2496    2531    2539    2541    2543    2547    2579    2581    2583    2585    2696    2698    2700    2702
           2759    2761    2763    2764    2875    2877    2879    2880    2997    2999    3001    3002    3120    3122    3124
           3125    3270    3272    3274    3275    3372    3374    3376    3377    3489    3491    3493    3494    3573    3575
           3577    3578    3654    3656    3658    3659    3738    3740    3742    3743    3860    3862    3864    3866    3981
           3983    3985    3986    4070    4072    4074    4075    4193    4195    4197    4198    4312    4314    4316    4317
           4433    4435    4437    4438    4520    4522    4524    4525    4590    4592    4594    4595    4653    4655    4657
           4658    4709    4711    4713    4714    4770    4772    4774    4775    4811    4813    4814    4816    4817    4874
           4876    4877    4879    4880    4913    4915    4917    4919    4966    4968    4970    4971    5094    5096    5098
           5099    5185    5187    5189    5190    5287    5289    5291    5293    5294    5309    5355    5357    5359    5360
           5410    5412    5414    5415    5458    5460    5462    5464    5578    5580    5582    5583    5624    5626    5628
           5629    5672    5674    5676    5677    5738    5740    5742    5743    5786    5788    5790    5792    5957    5959
           5961    5963    5964    6048    6050    6052    6053    6077    6079    6100    6102    6103    6141    6143    6145
           6146    6206    6208    6210    6212    6321    6323    6326    6328    6329    6371    6373    6377    6379    6380
           6455    6457    6460    6462    6463    6481    6483    6485    6486    6520    6522    6526    6528    6529    6575
           6577    6578    6580    6581    6594    6596    6598    6599    6638    6640    6642    6643    6658    6660    6662
           6663    6679    6681    6683    6685    6771    6773    6775    6777    6778    6811    6813    6814    6816    6817
           6850    6868    6870    6873    6875    6876    6929    6931    6932    6934    6935    6990    6992    6999    7001
           7002    7093    7155    7174    7177    7179    7186    7188    7189    7262    7324    7346    7351    7358    7362
           7369    7375    7422    7424    7426    7427    7491    7493    7494    7496    7497    7504    7515    7534    7539
           7607    7620    7664    7670    7732    7741    7960    7965    8001    8005    8006    8008    8009    8010    8012
           8018    8021    8023    8025    8032    8041    8047    8048    8050    8053    8127    8130    8200    8203    8267
           8270    8322    8325    8439    8442    8566    8569    8665    8668    8757    8760    8834    8837    8894    8900
           8917    8923    8927    8931    8943    8945    8946    8947    8956    8958    8966    8974    8975    8976    8982
           8993    8997    9000    9007    9009    9010    9011    9017    9024    9025    9052    9137    9146    9158    9165
           9174    9184    9194    9203    9210    9218    9231    9246    9286    9306    9381    9387    9441    9447    9474
           9552    9620    9663    9709    9725    9782    9800    9835    9870    9878    9905    9922    9963    9976    9986
           9994    9996   10002   10005   10014   10018   10020   10021   10031   10032   10033   10034   10035   10036   10037
          10038   10039   10043   10065   10072   10094   10110   10135   10146   10169   10191   10246   10250   10257   10259
          10264   10266   10275   10288   10342   10346   10355   10370   10403   10411   10416   10436   10439   10456   10458
          10472   10474   10485   10487   10813
.IFF       1183    1186    1187    1669    1691    1692    1696    1703    1729    2029    2120    2134    2295    2406    2408
           2414    2423    2443    2455    2457    2477    2497    2531    2539    2541    2543    2580    2581    2582    2584
           2585    2697    2698    2699    2701    2702    2760    2761    2762    2763    2764    2876    2877    2878    2879
           2880    2998    2999    3000    3001    3002    3121    3122    3123    3124    3125    3271    3272    3273    3274
           3275    3373    3374    3375    3376    3377    3490    3491    3492    3493    3494    3574    3575    3576    3577
           3578    3655    3656    3657    3658    3659    3739    3740    3741    3742    3743    3861    3862    3863    3865
           3866    3982    3983    3984    3985    3986    4071    4072    4073    4074    4075    4194    4195    4196    4197
           4198    4313    4314    4315    4316    4317    4434    4435    4436    4437    4438    4521    4522    4523    4524
           4525    4591    4592    4593    4594    4595    4654    4655    4656    4657    4658    4710    4711    4712    4713
           4714    4771    4772    4773    4774    4775    4812    4813    4815    4816    4817    4875    4876    4878    4879
           4880    4914    4915    4916    4918    4919    4967    4968    4969    4970    4971    5095    5096    5097    5098
           5099    5186    5187    5188    5189    5190    5288    5289    5292    5293    5294    5310    5356    5357    5358
           5359    5360    5411    5412    5413    5414    5415    5459    5460    5461    5463    5464    5579    5580    5581
           5582    5583    5625    5626    5627    5628    5629    5673    5674    5675    5676    5677    5739    5740    5741
           5742    5743    5787    5788    5789    5791    5792    5958    5959    5962    5963    5964    6049    6050    6051
           6052    6053    6078    6079    6101    6102    6103    6142    6143    6144    6145    6146    6207    6208    6209
```

DEQKCB.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

```
        6211  6212  6322  6323  6327  6328  6329  6372  6373  6378  6379  6380  6456  6457  6461
        6462  6463  6482  6483  6484  6485  6486  6521  6522  6527  6528  6529  6576  6577  6579
        6580  6581  6595  6596  6597  6598  6599  6639  6640  6641  6642  6643  6659  6660  6661
        6662  6663  6680  6681  6682  6684  6685  6772  6773  6776  6777  6778  6812  6813  6815
        6816  6817  6851  6869  6870  6874  6875  6876  6930  6931  6933  6934  6935  6991  6992
        7000  7001  7002  7094  7156  7175  7178  7179  7187  7188  7189  7263  7325  7347  7352
        7359  7363  7370  7376  7422  7423  7424  7425  7426  7491  7492  7493  7495  7496  7505
        7516  7535  7540  7608  7621  7665  7671  7961  7966  8002  8009  8012  8019  8022  8048
        8051  8054  8128  8131  8201  8204  8268  8271  8323  8326  8440  8443  8567  8570  8666
        8669  8758  8761  8835  8838  8895  8901  8918  8944  8945  8947  8975  8977  8982  9000
        9011  9024  9025  9026  9053  9138  9147  9159  9166  9175  9185  9195  9211  9219  9232
        9247  9287  9307  9382  9388  9442  9448  9475  9553  9621  9664  9710  9726  9783  9801
        9836  9871  9879  9906  9923  9964  9996 10006 10015 10040 10044 10066 10073 10095 10111
       10136 10147 10170 10192 10247 10251 10258 10260 10265 10267 10276 10289 10343 10347 10356
       10371 10412 10417 10437 10440 10457 10459 10473 10475 10486 10488 10814
.IFT    2162  2166  2208  2412  2548  7733  7742  8026  8033  8955  9010  9204 10404
.IFTF   2162  2166  2208  2412  2548  7733  7742  8026  8033  8953  9009  9204 10404
.IIF    1155  1160  1165  1180  1181  1182  1184  1188  1189  1190  1191  1192  1193  1194  1195
        1660  1753  2296  2298  2304  2305  2306  2308  2309  2407  8006  8013  8014  8028  8035
        8048  8049  8923  8924  8925  8926  8927  8954  8955  8969  8975  8976  8983  8984  8985
        8986  8987  8988  9011  9025  9060  9086  9098  9381 10030 10031 10032 10033 10034 10035
       10036 10037 10038
.IRP    1753  2271  2579  2696  2759  2875  2997  3120  3270  3372  3489  3573  3654  3738  3860
        3981  4070  4193  4312  4433  4520  4590  4653  4709  4770  4811  4874  4913  4966  5094
        5185  5287  5355  5410  5458  5578  5624  5672  5738  5786  5957  6048  6077  6141  6206
        6321  6371  6455  6481  6520  6575  6594  6638  6658  6679  6771  6811  6868  6929  6990
        7177  7422  7491  8012  8993  9566  9606  9682  9702  9848  9866  9970  9986
.LIST      1  1134  1135  1136  1137  1139  1594  1660  1694  1729  1731  1732  1733  1734  1735
        1736  1737  1738  1739  1740  1741  1742  1743  1744  1745  1746  1747  2162  2166  2208
        2271  2407  2412  2548  2579  2585  2696  2702  2759  2764  2875  2880  2997  3002  3120
        3125  3270  3275  3372  3377  3489  3494  3573  3578  3654  3659  3738  3743  3860  3866
        3981  3986  4070  4075  4193  4198  4312  4317  4433  4438  4520  4525  4590  4595  4653
        4658  4709  4714  4770  4775  4811  4817  4874  4880  4913  4919  4966  4971  5094  5099
        5185  5190  5287  5294  5355  5360  5410  5415  5458  5464  5578  5583  5624  5629  5672
        5677  5738  5743  5786  5792  5957  5964  6048  6053  6077  6103  6141  6146  6206  6212
        6321  6329  6371  6380  6455  6463  6481  6486  6520  6529  6575  6581  6594  6599  6638
        6643  6658  6663  6679  6685  6771  6778  6811  6817  6868  6876  6929  6935  6990  7002
        7177  7189  7422  7427  7491  7497  7733  7742  8013  8026  8033  8041  8927  9011  9204
       10020 10021 10030 10031 10032 10033 10034 10035 10036 10037 10038 10039 10404
.MACRO     1  1134  1135  1137  1139  1140  1188  1695  2115  4811  4874  5286  5957  6077  6321
        6371  6455  6520  6575  6770  6810  6868  6928  6990  7490 10021
.MCALL  1134  1594
.NLIST     1  1134  1135  1136  1137  1139  1594  1660  1694  1729  1731  1732  1733  1734  1735
        1736  1737  1738  1739  1740  1741  1742  1743  1744  1745  1746  1747  2162  2166  2208
        2271  2407  2412  2548  2579  2585  2696  2702  2759  2764  2875  2880  2997  3002  3120
        3125  3270  3275  3372  3377  3489  3494  3573  3578  3654  3659  3738  3743  3860  3866
        3981  3986  4070  4075  4193  4198  4312  4317  4433  4438  4520  4525  4590  4595  4653
        4658  4709  4714  4770  4775  4811  4817  4874  4880  4913  4919  4966  4971  5094  5099
        5185  5190  5287  5294  5355  5360  5410  5415  5458  5464  5578  5583  5624  5629  5672
        5677  5738  5743  5786  5792  5957  5964  6048  6053  6077  6103  6141  6146  6206  6212
        6321  6329  6371  6380  6455  6463  6481  6486  6520  6529  6575  6581  6594  6599  6638
        6643  6658  6663  6679  6685  6771  6778  6811  6817  6868  6876  6929  6935  6990  7002
        7177  7189  7422  7427  7491  7497  7733  7742  8013  8026  8033  8041  8927  9011  9204
       10020 10021 10030 10031 10032 10033 10034 10035 10036 10037 10038 10039 10404
.PAGE   1165  1204  1695  2028  2115  2218 10502
.REM       1
```

```
.REPT   1660  1731  1739  2602
.SBTTL  1176  1205  1330  1341  1355  1504  1654  1661  1670  1697  1819  1833  1844  1861  1878
        1889  1900  1912  1922  1935  1942  1953  1963  1981  1994  2007  2030  2117  2120  2414
        2579  2586  2696  2703  2759  2875  2997  3120  3270  3372  3489  3573  3654  3738  3860
        3867  3981  4070  4193  4312  4433  4520  4590  4653  4709  4770  4811  4874  4913  4920
        4966  5094  5185  5287  5355  5410  5458  5465  5578  5624  5672  5738  5786  5793  5957
        6048  6077  6141  6206  6213  6321  6371  6455  6481  6520  6575  6594  6638  6658  6679
        6686  6771  6811  6868  6929  6990  7004  7177  7347  7359  7370  7422  7491  7535  7608
        7665  7961  8003  8051  8128  8201  8268  8323  8440  8567  8666  8758  8835  8895  8919
        8978  9026  9308  9382  9442  9476  9554  9622  9665  9710  9783  9837  9871  9906  9965
       10007 10022 10040 10066 10095 10136 10170 10247 10258 10265 10276 10343 10356 10412 10437
       10457 10473 10486
.TITLE  1155
.WORD   1660  1690  1692  1705  1706  1709  1710  1711  1712  1715  1716  1717  1718  1719  1720
        1729  1731  1732  1733  1734  1735  1736  1737  1738  1739  1740  1741  1742  1743  1744
        1745  1746  1753  1755  1756  1757  1758  1759  1760  1761  1762  1763  1766  1768  1769
        1771  1772  1773  1778  1779  1780  1781  1782  1783  1784  1785  1788  1789  1790  1791
        1792  1793  1794  1795  1796  1797  1798  1799  1800  1801  1802  1803  1804  1805  1806
        1807  1808  1810  1814  1815  1824  1825  1826  1827  1828  1829  1830  1831  1837  1838
        1839  1840  1841  1842  1848  1849  1850  1851  1852  1853  1854  1855  1856  1857  1858
        1859  1865  1866  1867  1868  1869  1870  1871  1872  1873  1874  1675  1876  1882  1883
        1884  1885  1886  1887  1893  1894  1895  1896  1897  1898  1904  1905  1906  1907  1908
        1909  1910  1916  1917  1918  1919  1920  1943  1944  1945  1946  1947  1948  1949  1950
        1951  1954  1955  1956  1957  1958  1959  1960  1961  1964  1965  1966  1967  1968  1969
        1970  1971  1972  1973  1974  1975  1976  1977  1978  1979  1982  1983  1984  1985  1986
        1987  1988  1989  1990  1991  1992  1999  2000  2001  2002  2003  2004  2005  2012  2013
        2014  2015  2016  2017  2018  2019  2020  2021  2022  2023  2024  2025  2026  2190  2200
        2215  2229  2602  2605  2608  2611  2614  2617  2620  2623  2626  2629  2632  2635  2638
        2641  2644  2647  2650  2653  2656  2659  2662  2665  2668  2671  2674  2677  2680  2683
        2686  2693  2964  2966  2978  3004  3126  3276  3378  3495  3496  3579  3580  3581  3660
        3852  3857  3882  3883  3884  4199  4200  4318  4319  4383  4384  4526  4527  4596  4597
        4598  4599  4659  4660  4661  4662  4663  4664  4715  4716  4808  4809  4817  4818  4819
        4820  4910  4972  4973  4974  4975  4976  4977  5046  5116  5151  5181  5222  5455  5489
        5568  5594  5665  5744  5783  5834  5862  5892  5893  5916  5917  6147  6203  6235  6261
        6334  6359  6417  6441  6676  6852  6853  6854  6855  6856  6857  6858  6860  6861  6862
        6863  6864  6865  6866  6987  7414  7419  7459  7465  7550  7663  7729  8018  8021  8564
        9080  9114  9119  9155  9172  9179  9182  9189  9192  9200  9275  9378  9461  9551  9805
        9904  9995  9997 10262 10455 10516 10517 10518 10519 10520 10521 10526 10527 10530 10531
       10534 10535 10536 10537 10538 10539 10540 10541 10542 10543 10667 10678 10693 10725 10738
       10756 10770 10780 10797 10810 10812
```

```
% ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 *,DEQKCB.SEQ/SOL/CRF/NL:TOC/PAGNUM/DS:ERFZ=DEQKCB.SML,DEQKCB.P11
 RUN-TIME: 70 107 21 SECONDS
 RUN-TIME RATIO: 376/199=1.8
 CORE USED:  33K  (65 PAGES)
```

N04

Spooler runtime 39 Seconds, 163 KCS, 1151 disk reads, 0 disk writes, 258 pages
[redacted] Date [redacted] Monitor IPC-0 [redacted]
[redacted]
[redacted] 4567890123456789012 **
00000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000111111111111111111111111111111111110
9000000011111111112222222222333333333344444444445555555555666666666677777777778888888888999999999900000000001111111111222222222233312