11/70–74

11/70 – 74 CPU # 2

CEKBBD0

AH-7968D-MC

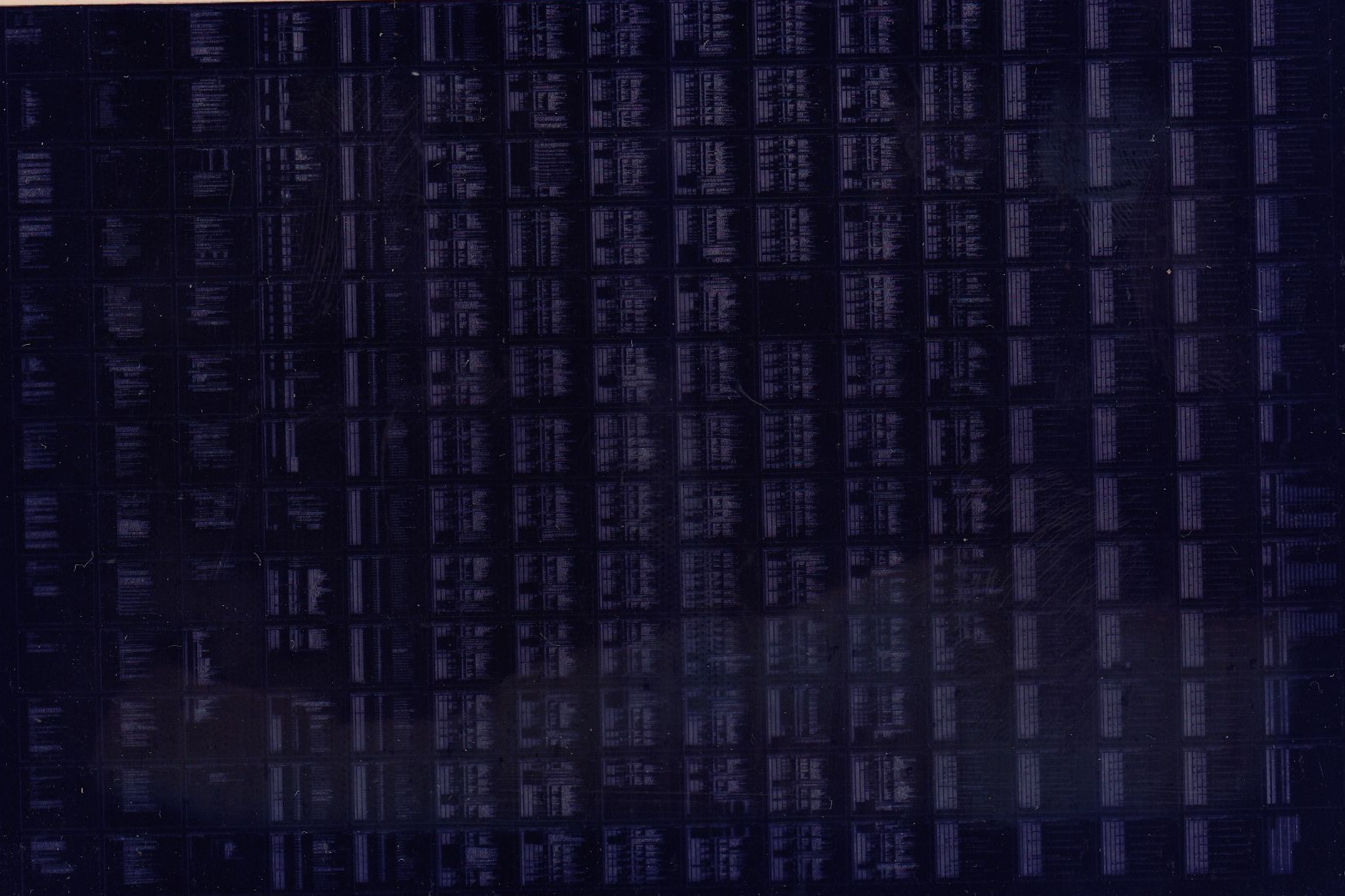
COPYRIGHT 75–80

FICHE 1 OF 2

JAN 1980

digital

MADE IN USA



11/70–74

11/70 – 74 CPU # 2

CEKBBD0

AH-7968D-MC

FICHE 2 OF 2

JAN 1980

digital

MADE IN USA

IDENTIFICATION

PRODUCT CODE:	AC-7966D-MC
PRODUCT NAME:	CEKBBD0 PDP-11/70-74MP CPU DIAGNOSTIC PART 2
DATE :	MAY, 1979
MAINTAINER:	DIAGNOSTIC ENGINEERING
AUTHORS:	DON MONROE
MODIFIED BY:	ERNEST PREISIG 11/3/78

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	DECX/11

CONTENTS

1. ABSTRACT

CEKBA/B ARE PROGRAMS DESIGNED TO DETECT AND REPORT LOGIC FAULTS IN THE PDP 11/70-74MP CENTRAL PROCESSING UNIT. THEY CONSISTS OF 210(8) INDIVIDUAL TESTS CAREFULLY DESIGNED AND SEQUENCED TO DETECT AND ATTEMPT TO IDENTIFY LOGIC FAULTS AT A MINIMUM HARDWARE/SOFTWARE LEVEL. THESE TESTS ARE PARTITIONED INTO TWO STAND-ALONE PROGRAMS AS DESCRIBED BELOW.

A. BASIC INSTRUCTION TESTS

CEKBA CONSISTS OF A LOGICALLY SEQUENCED SET OF INSTRUCTION TESTS DESIGNED TO VERIFY THE INTEGRITY OF THOSE INSTRUCTIONS AND LOGIC OPERATIONS USED BY THE UTILITY ROUTINES THAT PROVIDE ERROR REPORTING AND SCOPE LOOPING FACILITIES FOR CEKBB.

ANY FAULT DETECTED IN THIS PROGRAM CAUSES THE PROGRAM TO ''HALT'' WITH THE CONSOLE ADDRESS LIGHTS INDICATING THE ERROR PROGRAM COUNTER AND THE CONSOLE DATA LIGHTS SHOWING THE TEST NUMBER (FOR TESTS 24 AND ABOVE). ADDITIONAL FAULT IDENTIFICATION INFORMATION IS AVAILABLE IN THE PROGRAM ANNOTATION FOR THE FAILING TEST.

IF THE PROGRAM HALTS AT LOCATION 6 OR 12 (ADDRESS LIGHTS OF 10 OR 14) THE PROGRAM ANNOTATION FOR THE INDICATED TEST NUMBER, SHOULD GIVE A CLUE TO THE PROBLEM. TO LOOP ON THE ERROR THE HALT MUST BE REPLACED BY THE OCTAL CODE SHOWN IN THE COMMENT FIELD OF THE HALT AND THE PROGRAM RESTARTED AT 200, OR THE START ADDRESS OF THAT PARTICULAR TEST.

DURING THE FIRST PASS THE PROGRAM WILL TYPE ''AA'' AND THE PROGRAM TITLE.

B. ADVANCED INSTRUCTION AND MISCELLANEOUS LOGIC TESTS

CEKBB CONSISTS OF A LOGICALLY SEQUENCED SET OF INSTRUCTION TESTS FOLLOWED BY A SET OF MISCELLANEOUS LOGIC TESTS. THE INSTRUCTION TESTS COMPLETE THE TEST OF THE PDP 11/70-74MP INSTRUCTION REPERTOIRE. THE LOGIC TESTS VERIFY SUCH THINGS AS: 1) THE INTERNAL REGISTERS; 2) REGISTERS SET 1; 3) INTERNAL INTERRUPTS; 4) BUS REQUEST LEVELS 4, 5, AND 6; 5) INTERNAL TRAPS, AND ABORTS; 6) OUTER MODE SELECTION; AND 7) EXTERNAL TRAPS AND ABORTS. EACH TEST IN THIS PROGRAM CALLS A ''SCOPE LOOP'' UTILITY THAT FACILITATES USER CONTROL OF TEST SELECTION AND EXECUTION VIA THE CONSOLE SWITCH REGISTER.

UPON DETECTION OF A LOGIC FAULT EACH TEST IN THIS SECTION CALLS AN ''ERROR SERVICE'' THAT REPORTS IT AS HARD COPY ON THE CONSOLE TERMINAL DEVICE. THE ERROR SERVICE ROUTINE ALSO FACILITATES USER CONTROL OF THE PROGRAM SEQUENCE VIA

CONSOLE SWITCH REGISTER OPTIONS. AFTER REPORTING THE ERROR THE PROGRAM CONTINUES ON ITS NORMAL SEQUENCE UNLESS MODIFIED BY THE USER ACTIVATING THE ''HALT ON ERROR'' SWITCH OPTION.

C. IMPORTANT NOTE

THE PROGRAM ANNOTATION IN CEKBA AND THE TYPED ERROR REPORTS IN CEKBB ARE BASED UPON THE KNOWLEDGE THAT ALL PREVIOUS TESTS WERE FAULTLESS AND THAT THERE IS ONLY ONE SINGLE POINT FAILURE IN THE PROCESSOR. THIS MEANS THAT IF EITHER PROGRAM, OR THE PROGRAMS THEMSELVES, ARE NOT RUN IN SEQUENCE, THE ERROR MESSAGE MAY NOT BE VALID.

ALTHOUGH EACH ERROR ANNOTATION AND TYPED MESSAGE CONCLUSION HAS BEEN PROVEN BY PHYSICAL FAULT INSERTION (ONE SIGNAL STUCK LOW), IT IS HUMANLY IMPOSSIBLE TO GUARANTEE THAT THE ERROR REPORT IS 100% CORRECT. THE SOLE FUNCTION OF THE ERROR REPORT IS TO DIRECT THE USER TO THE MOST PROBABLE AREA OF FAILURE.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP 11/70-74MP CPU WITH OPERATORS CONSOLE
LA30 OR EQUIVALENT TERMINAL

2.2 STORAGE

CEKBA REQUIRES 16K TO LOAD AND RUN
CEKBB REQUIRES 16K TO LOAD AND 32K TO RUN

2.3 PRELIMINARY PROGRAMS

CEKBA REQUIRES THAT TWO INSTRUCTIONS WORK:
''BR'' AND ''HALT''

CEKBB REQUIRES THAT CEKBA RUN

3. LOADING PROCEDURE

3.1 METHOD

BOTH CEKBA AND CEKBB ARE LOADED FROM THE XXDP MEDIA.
REFER TO THE XXDP MANUAL FOR FURTHER INFORMATION.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE 5.1

4.2 STARTING ADDRESS

200

4.3 PROGRAM AND OPERATOR ACTION

A. CEKBA

1. LOAD PROGRAM INTO MEMORY (SEE SECTION 3)
2. LOAD ADDRESS 200
3. PRESS START
4. THE PROGRAM WILL PRINT ''AA'' AND THE TITLE THE FIRST TIME THROUGH.
5. THE PROGRAM WILL LOOP AND END OF PASS WILL BE TYPED AFTER THE REQUIRED NUMBER OF PASSES.

B. CEKBB

1. LOAD PROGRAM INTO MEMORY (SEE SECTION 3)
2. ENSURE RH CONTROLLER IS ENABLED, IF SW<5>=0
3. IF AN RK05 IS AVAILABLE (AND THERE WAS NO RH) ENSURE AT LEAST ONE DRIVE IS ENABLED; IF SW<5>=0
4. LOAD ADDRESS 200
5. SET SWITCHES (SEE SECTION 5.1)
6. PRESS START
7. THE PROGRAM WILL LOOP AND AN END OF PASS MESSAGE WILL BE TYPED EVERY PASS.

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

A. CEKBA

NONE

B. CEKBB

WITH SW<15:0>=0 THE PROGRAM WILL PRINT OUT ON ERRORS AND CONTINUE. ''END OF PASS'' WILL BE TYPED AT THE COMPLETION OF EACH PASS.

THE SWITCH SETTINGS ARE:

```
SW<15>=1 ... HALT ON ERROR
SW<14>=1 ... LOOP ON TEST
SW<13>=1 ... INHIBIT ERROR TYPEOUTS
SW<12>=1 ... INHIBIT T BIT TRAPPING
SW<11>=1 ... INHIBIT ITERATIONS
SW<10>=1 ... RING BELL ON ERROR
SW<9> =1 ... LOOP ON ERROR
SW<8> =1 ... LOOP ON TEST IN SW<7:0>
SW<7> =1 ... NO ACTION
SW<6> =1 ... SKIP BUS REQUEST 6 TESTING
SW<5> =1 ... SKIP BUS REQUEST 5 TESTING
SW<4> =1 ... SKIP BUS REQUEST 4 TESTING
SW<0> =1 ... SKIP OPERATOR INTERVENTION TESTING
```

5.2 SUBROUTINE ABSTRACTS

A. CEKBA

SEE 5.2.4 AND 5.2.5

B. CEKBB

5.2.1 SPURIOUS ERROR HANDLER

THIS ROUTINE IS CALLED BY AN UNEXPECTED TRAP TO LOCATION 4 OR 114. IT PRINTS A SHORT MESSAGE FOLLOWED BY THE PROGRAM COUNTER AT THE TIME OF THE TRAP AND THE APPROPRIATE ERROR REGISTER (I.E. THE CPU ERROR REGISTER IN CASES OF A TRAP TO 4 AND THE MEMORY ERROR REGISTER IN CASES OF A TRAP TO 114).

5.2.2 SCOPE

THIS SUBROUTINE CALL (VIA AN IOT INSTRUCTION) IS PLACED BETWEEN EACH TEST. IT RECORDS THE STARTING ADDRESS OF EACH TEST IN LOCATION '$LPADR'' AND '$LPERR'' AS IT IS BEING ENTERED. IT ALSO CONTROLS TEST ITTERATION, LOOPING, AND SEQUENTIAL FLOW CHECKS (SEE 5.2.8).

5.2.3 ERROR

THIS SUBROUTINE CALL (VIA A EMT INSTRUCTION) IS USED TO REPORT ALL ERRORS. (REFER TO 6)

5.2.4 TRAP CATCHER

A ''.+2'' - ''HALT'' SEQUENCE IS REPEATED FROM LOCATION 0 TO

LOCATION 776 TO CATCH ANY UNEXPECTED TRAPS (EXCEPT 4 AND 114). THUS, ANY UNEXPECTED TRAPS WILL HALT AT THE DEVICE TRAP VECTOR +2.

5.2.5 TRAP

A NUMBER OF SUBROUTINES ARE CALLED BY THE TRAP INSTRUCTION. FOLLOWING ARE THE CALLS USED AND THE LABEL OF THE STARTING ADDRESS OF THE SUBROUTINES.

5.2.5.1 TYPE ($TYPE)

ROUTINE TO TYPE AN ASCII STRING ON THE TTY.

THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.

5.2.5.2 TYPEOC ($TYPOC)

ROUTINE TO CONVERT A 16-BIT BINARY NUMBER TO A 6-DIGIT OCTAL NUMBER AND TYPE IT.

5.2.5.3 TYPEDS ($TYPDS)

ROUTINE TO CONVERT A 16-BIT BINARY NUMBER TO A 5-DIGIT SIGNED DECIMAL NUMBER AND TYPE IT.

5.2.6 POWER DOWN AND UP

THIS SUBROUTINE CALL (VIA A POWER DOWN) SAVES THE RETURN PC UPON A POWER DOWN. WHEN POWER IS RESTORED A MESSAGE IS TYPED AND THE TEST WILL RESTART.

5.2.7 MONITOR RESTORE (QUIT)

THIS SUBROUTINE IS ENTERED BY TYPING A "CONTROL C" OR WHEN THE END OF PASS IS REACHED AND THE PROGRAM IS RUNNING UNDER A MONITOR. SEE 7.2 FOR DETAILS.

5.2.8 CHECK TEST SEQUENCE (SEQENC)

THIS ROUTINE IS CALLED IN THE SCOPE ROUTINE. IT COMPARES THE ADDRESS OF THE SCOPE CALL WITH THE ADDRESS POINTED TO BY $TSTNM IN THE "TEST ADDRESS TABLE". IF THEY DON'T COMPARE, A MESSAGE IS TYPED, INDICATIONG THAT A TEST WAS SKIPPED.

5.2.9 PERIPHERAL DETERMINATOR AND INTERRUPT ENABLE ROUTINES

5.2.9.1 PERIPHERAL DETERMINATOR

THIS ROUTINE IS EXECUTED, IN LINE, BETWEEN TESTS 32 AND 33 OF CEKBB. IT CHECKS THE SYSTEM TO DETERMINE IF ONE OF FOUR PERIPHERALS IS AVAILABLE (RS04, RP04, TM, OR RK05) FOR BUS REQUEST LEVEL 5 TESTING AND IF A LINE CLOCK IS AVAILABLE FOR LEVEL 6 TESTING.
IF A DEVICE IS FOUND, THE ADDRESS OF ''INT5SU'' (INTERRUPT 5 SUBROUTINE) AND $KW11L/P (LINE CLOCK INTERRUPT SUBROUTINE) IS PLACED IN LOCATIONS ''INTER5'' AND ''INTER6'' RESPECTIVELY.

5.2.9.2 INTERRUPT ENABLE ROUTINES

THESE ROUTINES ARE CALLED VIA A ''JSR PC,@INTERX'' (X=5 OR 6). THE ROUTINE SETS UP AND RESPONDS TO A BUS REQUEST. IF THE BR DOES NOT WORK THE RETURN PC IS INCREMENTED BY 2 AND THE RETURN IS MADE.

5.3 OPERATOR ACTION

THE LAST TEST OF CEKBB REQUIRES OPERATOR INTERVENTION. THIS TEST IS ONLY EXECUTED ON PASS 1, IF SW<0>=0. QUESTIONS ARE TYPED ON THE TELETYPE AND THE OPERATOR MUST RESPOND EITHER ON THE CONSOLE OR ON THE TELETYPE.

IF LOCATION 42 IS NON-ZERO, INDICATING THAT THE PROGRAM WAS LOADED BY A MONITOR, THIS TEST IS SKIPPED ON ALL PASSES.

6. ERRORS

6.1 ERROR HALTS AND DESCRIPTION

A. CEKBA

EVERY ERROR IN CEKBA HALTS THE PROCESSOR. THE COMMENT FIELD OF THE HALT INSTRUCTION CONTAINS THE NAME OF THE SIGNAL THAT WAS MOST LIKELY TO HAVE CAUSED THE ERROR. ALSO, IN THE COMMENT FIELD, IS THE OCTAL CODE THAT SHOULD REPLACE THE HALT IF LOOP ON ERROR IS DESIRED. IF THE PROGRAM HALTS AT LOCATION 6 OR 12 THE USER SHOULD LOOK IN THE TEST DESCRIPTION, OF THE TEST THAT FAILED, TO FIND THE MOST LIKELY CAUSE OF THE ERROR.

B. CEKBB

NONE OF THE ERRORS IN CEKBB HALT THE PROCESSOR IF SW<15>=0.

THERE ARE OVER 450(8) UNIQUE ERRORS THAT CAN OCCUR IN THIS PROGRAM. WHEN AN ERROR IS ENCOUNTERED THE CALL TO THE ERROR ROUTINE IS MADE AND IF SW<13> IS NOT SET, AN ERROR MESSAGE PERTAINING TO THE ERROR WILL BE TYPED. EACH ERROR TYPE OUT WILL CONTAIN THE FOLLOWING:

1. AN ERROR MESSAGE
2. A DATA HEADER
3. A DATA STRING

THE DATA STRING WILL CONTAIN, AT A MINIMUM, THE ERROR PC AND THE TEST NUMBER. IN SOME CASES THE EXPECTED AND ACTUAL VALUES OF A REGISTER ARE ALSO INCLUDED.

REFER TO THE LISTING UNDER $ERRTB FOR THE TYPES OF ERRORS THAT CAN OCCUR.

SEE SECTION 7.1 FOR NON-STANDARD CONFIGURATION.

6.2 ERROR RECOVERY

A. CEKBA

ERROR RECOVERY IS STRICTLY BY USER INTERVENTION.

B. CEKBB

```
SW<15:9>=0 - MOST ERRORS WILL CAUSE EXECUTION TO GO TO
             THE START OF THE NEXT TEST AFTER THE MESSAGE
             IS TYPED.  A FEW TESTS ARE DIVIDED INTO
             SECTIONS.  IN THESE TESTS AN ERROR WILL CAUSE
             EXECUTION TO GO TO THE NEXT SECTION.

SW<15>=1   - PRESSING THE CONSOLE CONTINUE WILL CAUSE THE
             PROGRAM TO TYPE AN ERROR MESSAGE AND HALT.
             PRESSING THE CONSOLE CONTINUE AGAIN WILL CAUSE
             THE PROGRAM TO CONTINUE AS IF SW<15>=0.
```

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS

A. CEKBA

NONE

B. CEKBB

IF THE USER WANTS TO RUN THE BUS REQUEST 5 TEST HE MUST ENSURE THAT EITHER AN RH CONTROLLER IS ACTIVE OR THAT A UNIBUS DEVICE (RK, RS, RP, TM) IS ACTIVE.

IF A RP11-E IS SHIPPED IN PLACE OF A RP04, THIS REPRESENTS A NON-STANDARD CONFIGURATION AND LOCATION 1244 SHOULD BE CHANGED FROM 176700 TO 176714.

7.2 OPERATING RESTRICTIONS

A. CEKBA

NONE

B. CEKBB

SINCE THE PROGRAM COULD POSSIBLY DESTROY A MONITOR, IN PAGE 6, ALL LOCATIONS BETWEEN 152000 AND 157776 ARE SAVED AT THE BOTTOM OF THE PROGRAM. TO RESTORE THESE LOCTIONS A "CONTROL C" SHOULD BE TYPED ON THE TERMINAL. THE LOCATIONS WILL BE RESTORED, A MESSAGE TYPED, AND THE PROCESSOR WILL HALT.

IF THE PROGRAM IS RUNNING UNDER A MONITOR THE LOCATIONS ARE RESTORED AND CONTROL IS RETURNED TO THE MONITOR THRU THE END OF PASS LINKAGE.

8. MISCELLANEOUS

8.1 EXECUTION TIME

A. CEKBA

FIVE(5) SECONDS PER END OF PASS MESSAGE IF RUNNING UNDER A MONITOR.
2 MINUTES IF THE PROGRAM WAS DUMPED.

B. CEKBB

THE FIRST PASS TAKES APPROXIMATELY 8 SECONDS. ALL SUB-SEQUENT PASSES TAKE APPROXIMATELY 3 MINUTES.

8.2 STACK POINTER

STACK IS INITIALLY SET TO 1100.

8.3 PASS COUNT

A PROGRAM PASS THRU COUNT IS KEPT IN "$PASS".

A. CEKBA

IF THE PROGRAM IS RUNNING UNDER A MONITOR OR WAS LOADED BY ACT 11 THE PROGRAM MAKES 144(8) PASSES FOR EACH END OF PASS MESSAGE. IF THE PROGRAM WAS DUMPED, 4000(8) PASSES ARE MADE FOR EACH END OF PASS MESSAGE.
THE PASS COUNT IS DISPLAYED IN THE DATA LIGHTS WHEN DISPLAY IS SELECTED BY THE ROTARY SWITCH.

B. CEKBB

THE PROGRAM MAKES 1 PASS FOR EACH END OF PASS MESSAGE.

8.4 ITERATIONS

A. CEKBA

NONE

B. CEKBB

THE FIRST PASS OF THE PROGRAM WILL AUTOMATICALLY INHIBIT ITERATIONS. ALL SUBSEQUENT PASSES WILL PERFORM FULL, (2000 DECIMAL) ITERATIONS.

8.5 SPECIAL REGISTERS

A. CEKBA

RO IS RESERVED FOR THE TEST NUMBER.

B. CEKBB

NONE

8.6 T BIT TRAPPING

A. CEKBA

NONE

B. CEKBB

EVERY OTHER PASS, STARTING WITH PASS 2, RUNS WITH THE T BIT ON. THIS CAUSES EVERY INSTRUCTION TO T BIT TRAP THEREFORE, IT IS NOT POSSIBLE TO ''SINGLE INSTRUCTION'' THE TEST WITHOUT TURNING THE T BIT OFF.

CERTAIN TESTS AUTOMATTICALLY TURN IT OFF IF IT WAS ON. THESE TESTS WILL ALSO TURN IT BACK ON UNLESS THE FOLLOWING TEST REQUIRES THAT IT ALSO BE OFF.

8.7 OSCILLOSCOPE SYNC POINTS

A. CEKBA

BEGINNING WITH TEST 24 EACH TEST HAS AN OSCILLOSCOPE SYNC INSTRUCTION. THE ADDRESS OF THE CONDITION CODE ROM STATE (44) IS IN THE PROCESSOR MICROBREAK REGISTER (ADDRESS 17777770). THIS WILL CAUSE PIN AE1 (SLOT 10) ON THE BACKPLANE TO GO HIGH EACH TIME A CONDITION CODE (OR NOP) INSTRUCTION IS EXECUTED. THEREFORE, IF THE OSCILLOSCOPE EXTERNAL SYNC IS CONNECTED TO THIS PIN AND THE SYNC SELECT PUT ON EXTERNAL THE OSCILLOSCOPE WILL BE SYNCHRONIZED WITH THE INSTRUCTION IMMEDIATELY PRECEEDING THE INSTRUCTION UNDER TEST (IUT).

B. CEKBB

ONLY TESTS 1 THRU 20 CONTAIN SYNC INSTRUCTIONS.

8.8 CACHE CONTROL

THE FIRST PASS OF BOTH PROGRAMS RUN WITH THE CACHE DISABLED (FORCING MISSES IN BOTH GROUPS). ALL SUBSEQUENT PASSES RUN WITH THE CACH ENABLED.

9. PROGRAM DESCRIPTION AND HISTORY

CEKBBB -	PROGRAM ENHANCED TO LOOP ON SUBTEST WITH SWITCH 8 SET
CEKBBC -	DOCUMETATION OF NON-STANDARD RP11 WAS ADDED TO SECTION 6.1
CEKBBD -	ERROR PRINTOUTS OF CALLS > AN EMT 377 WHICH DECODES NEXT WORD FOR THE ERROR POINTER WAS INCORRECT. MEMORY SIZING ROUTINE ENHANCED TO HANDLE UNEXPECTED MAIN MEMORY TIMEOUT TRAPS TO 114. PROGRAM ENHANCED TO RUN ON AN 11/74. ALSO,THIS SECTION OF REV HISTORY WAS ADDED TO THE DOC. (9.0) .

```
              DOCUMENT
           ***************
    PDP 11/70-74MP CPU DIAGNOSTIC PART 2
           ***************
```

TABLE OF CONTENTS
****

TABLE OF CONTENTS
****

```
          17      COPYRIGHT (C) 1975,1978
                  DIGITAL EQUIPMENT CORP.
                  MAYNARD, MASS. 01754

                  PROGRAM BY DONALD W. MONROE

                  THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                  PACKAGE (MAINDEC-11-DZQAC-A5).
```

```
           249     SUPERVISOR ''I'' PAGE DESCRIPTOR REGISTERS

           260     SUPERVISOR 'D'' PAGE DESCRIPTOR REGISTERS

           271     SUPERVISOR ''I'' PAGE ADDRESS REGISTERS

           282     SUPERVISOR 'D'' PAGE ADDRESS REGISTERS

           293     KERNEL ''I'' PAGE DESCRIPTOR REGISTERS

           304     KERNEL 'D'' PAGE DESCRIPTOR REGISTERS

           315     KERNEL ''I'' PAGE ADDRESS REGISTERS

           326     KERNEL 'D'' PAGE ADDRESS REGISTERS

           *******************************************************************
     340   UNIBUS MAP REGISTER DEFINITIONS
           *******************************************************************

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

           *******************************************************************
     434   TRAP CATCHER
           *******************************************************************

           437     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

           *******************************************************************
     441   STARTING ADDRESS(ES)
           *******************************************************************

           *******************************************************************
     447           ACT11 HOOKS
           *******************************************************************

           449     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
```

```
        *****************************************************************
473     COMMON TAGS
        *****************************************************************

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

        *****************************************************************
547     ERROR POINTER TABLE
        *****************************************************************

        549     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:

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

        1560    TEST 1  SPL

                        IF FORK A FAILS EXECUTION WILL GO TO ONE OF 3 STATES:
                        RSD.02,SVC.70, OR D30.00.
                        RSD.02 WILL CAUSE A TRAP TO LOCATION 10.
                        SVC.70 WILL HANG THE PROCESSOR IN THE PAUSE STATE.  THIS
                        WILL ONLY OCCUR IF RACF E17(AFIR04(1)*AFIR05(0)*(RTS:CCOP))
                        IS BAD.
                        D30.00 WILL CAUSE A TRAP TO LOCATION 10 AFTER STATE D10.60.
                        THIS FAILURE CAN BE DIFFERENTIATED FROM THE FIRST BY TESTING
                        THE REGISTER ASSOCIATED WITH BITS <2:0> OF THE OP CODE TO

        1571            SEE IF IT WAS INCREMENTED.

                        IF BOTH LEVELS 2 AND 5 COME UP AS LEVEL 4, PDRD E31(1)
                        COULD BE BAD.

                        ONCE IT IS DETERMINED THAT THE INSTRUCTION WORKS A BIT TEST IS
                        MADE ON THE PSW <7:5>.

                        ROM FLOW-43,361

        1614    TEST 2  RESET

                        IF FORK A FAILS EXECUTION WILL EITHER GO TO WAT.00 OR TRP.02.
                        THE LA30 PRINTER WILL BE STARTED SUCH THAT A FAILURE INTO
                        WAT.00 WILL RECOVER.

                        IF TRP.01 IS ENTERED A TRAP SEQUENCE WILL EXECUTE
                        WITH A TRAP VECTOR OF 4.

                        ROM FLOW-15,255,374
```

```
1672    TEST 3  MARK

                FORK A CAN FAIL INTO ONE OF THE FOLLOWING:
                RSD.00, MFP.80, MTP.00, SVC.70, OR D67.01.
                STATE RSD.00 WILL CAUSE A TRAP TO LOCATION 10.
                MFP.80 WILL EXECUTE AN MFP INSTRUCTION.
                MTP.00 WILL EXECUTE AN MTP INSTRUCTION EXCEPT FOR THE ALU.

1679            SVC.70 WILL HANG THE PROCESSOR IN THE PAUSE CONDITION.
                THIS WILL ONLY HAPPEN IF RACF E8 IS BAD.
                D67.01 WILL STEP THE PCB AND TRAP TO LOCATION 10 AFTER STATE D10.60.

                ROM FLOW-47,252,235,234

1733    TEST 4  ASH*DM0

                IF FORK A FAILS EXECUTION WILL GO TO RSD.00.

                IF GRAJ SC05 L DOES NOT GO LOW OR DOES NOT GET THRU TO
                RACK BRCAB04 L A SHIFT RIGHT WILL OCCUR.
                IF THE SHIFT COUNTER DOES NOT SHIFT OR GRAJ SC=0 DOES NOT GO
                LOW THE PROCESSOR WILL HANG UP IN STATE ASH.41.

                ROM FLOW-52,305,257,166 LEFT SHIFT
                         52,305,277     RIGHT SHIFT

1866    TEST 5  ASH*DM1

                IF FORK A FAILS EXECUTION WILL GO TO RSD.00.

                IF FORK B FAILS EXECUTION WILL EITHER GO TO RSD.00 OR
                MUL.00.
                MUL.00 WILL ONLY BE ENTERED IF EITHER B FORK MUX INPUT B2
                DOES NOT GO HIGH OR THE MUX IS BAD.

                ROM FLOW-1,175,62,52,305,257

1902    TEST 6  ASH*DM2

                IF FORK A FAILS EXECUTION WILL GO TO RSD.00.

                ALL OTHER LOGIC HAS BEEN TESTED

                ROM FLOW-2,175,62,52,305

1923    TEST 7  ASH*DM4

                IF FORK A FAILS EXECUTION WILL GO TO RSD.00.

                ALL OTHER LOGIC HAS BEEN TESTED.

                ROM FLOW-4,122,177,62,52,305
```

```
1945    TEST 10 ASHC*DMO

                NEITHER FORK A NOR BEN03 SHOULD FAIL.

1949            IF THE INSTRUCTION FAILS, ONE OF THE ASC STATES IS BAD.

                ONCE IT IS DETERMINED THAT THE INSTRUCTION WORKS,
                A BIT STUCK TEST IS PERFORMED ON THE SHIFT COUNTER.

                ROM FLOW-53,306,267,227      RIGHT SHIFT
                         53,306,247,176,136 LEFT SHIFT
                         53,306,207         NO SHIFT
2078    TEST 11 ASHC*DM1

                THE ONLY POSSIBLE FAILURES ARE FORK B OR STATE ASC.00.

                IF FORK B FAILS EXECUTION WILL EITHER GO TO RSD.00 OR
                ASH.00.
                ASH.00 WOULD PERFORM AN ASH INSTRUCTION INSTEAD OF AN ASHC.

                ROM FLOW-1,175,63,53,306,267,227

2115    TEST 12 MUL*DMO

                FORK A SHOULD NOT FAIL.
                THE FOLLOWING WOULD BE BEN11 FAILURES:
                IF EITHER GRAD DR00 IS STUCK HIGH OR NOT GETTING THRU TO
                RACK E64(C1) OR RACK E64 IS BAD (INPUT C1 FAILED HIGH)
                THE MULITPLICAND WILL BE MULITPLIED BY 177777.
                IF EITHER GRAD DR00 IS STUCK LOW OR NOT GETTING THRU TO
                RACK E64(C1) OR RACK E64 IS BAD (INPUT C1 FAILED LOW)
                THE MULTIPLICAND WILL BE MULTIPLIED BY ZERO.
                IF GRAJ SC=0 IS NOT GETTING TO RACK E50(C1) AND RACK E50
                 IS BAD (INPUT C1 FAILED LOW) THE MULTIPLICAND WILL ONLY
                BE MULTIPLIED BY BIT 0 OF THE MULTIPLIER.
                IF RACK E50(C1) FAILS HIGH THE PROCESSOR WILL HANG UP IN
                STATE MUL.20(266).
                IF THE INSTRUCTION FAILS TO MULTIPLY CORRECTLY AND ONE
                OF THE ABOVE CONDITIONS CANNOT BE DETERMINED THEN THE
                FAILURE COULD BE IN THE INSTRUCTION DECODE ROM.

                IF THE CC'S COME UP BAD THEN THE FAILURE COULD BE IN
                STATES MUL.40,50, OR 60, OR IN THE CONDITION CODE ROM.

                ROM FLOW-50,102,266/246,226/206,310

2222    TEST 13 MUL*DM1

                FORK A SHOULD NOT FAIL.

                FORK B WILL FAIL TO RSD.00 IF THE R(MUL:ASHC+MFP) FIELD OF
                THE INSTRUCTION DECODE ROM IS BAD.
                IF STATE MUL.00 FAILS THE RESULT WILL BE BAD.

                ROM FLOW-1,175,60,102,266/246,226/206,310

2254      TEST 14 DIV*DMO

                FORK A SHOULD NOT FAIL.
           SECTION 1
                THE FIRST SECTION HAS A ZERO DIVISOR. IF STATE DIV.00
                FAILS OR IRCF Z2(1) DOES NOT GET TO RACK E49 OR RACK
                E49(B1) IS STUCK LOW EXECUTION WILL GO FROM DIV.10 TO
                DIV.20. THIS WILL CAUSE THE DIVIDE TO ABORT (GO TO DVE.20)
                AFTER STATE DIV.60 AND THE C BIT WILL BE CLEAR.

           SECTION 2
                THE NEXT SECTION DIVIDES 4 BY 2. IF RACK E49(B1) IS STUCK
                HIGH THE ALGORITHM WILL ABORT THINKING THAT THE DIVISOR
                IS ZERO. IF BEN04 FAILS THE DIVIDE WILL ABORT THINKING
                THAT THE DIVIDEND IS THE MOST NEGATIVE NUMBER.
                IF BEN16 FAILS THE DIVIDE WILL COMPLETE BUT R1 WILL CONTAIN
                155776. IF BEN05 FAILS THE ALGORITHM WILL ABORT THRU STATE
                DVE.20. IF BEN04 (AFTER DIV.70) FAILS R0 WILL END UP WITH 177777
                AND R1 WILL HAVE 177774. IF BEN03 FAILS R0 WILL END UP WITH

2273            20 AND R1 WILL HAVE 177774. IF BEN16 FAILS AFTER DVC.00 R0
                WILL HAVE 2 BUT R1 WILL HAVE 177776.

           SECTION 3
                THE NEXT SECTION DIVIDES 6 BY 2 TO TEST BEN16*DR0(1).
                A FAILURE WILL LEAVE THE REMAINDER (R1)=2 INSTEAD OF ZERO.

           SECTION 4
                THE NEXT SECTION DIVIDES 4 BY -2 TO TEST BEN15*SR15(1).
                IF THIS FAILS R0 WILL CONTAIN 27777 AND R1 WILL CONTAIN 4.

           SECTION 5
                THE NEXT SECTION DIVIDES 1177777 BY 1 TO TEST BEN05*DIV QUIT.
                IF THIS FAILS R0 WILL CONTAIN 177777.

           SECTION 6
                THE NEXT SECTION DIVIDES 1000000 B2 -2 TO TEST BEN05*DIV QUIT.
                THIS SECTION WILL ONLY FAIL IF GRAJ E5 (Z2(0)*LEFT SAVE (1)) IS BAD.

           SECTION 7
                THE NEXT SECTION DIVIDES 100000000000 BY 2 TO TEST
                BEN04*NEGATIVE DIVIDEND.

           SECTION 8
                THE NEXT SECTION DIVIDED 177776 177777 BY -1 TO TEST BEN05*DIV QUIT.
                THIS TEST WILL ONLY FAIL IF GRAJ E5(N(1)*SR15(1)) IS BAD.

           SECTION 9
                THE NEXT SECTION DIVIDES -5 BY 2 TO ENSURE THAT THE REMAINDER IS
                STORED AS A NEGATIVE NUMBER.
                SECTION 10
                     THE NEXT SECTION DIVIDES -5 BY -2 TO TEST STATES DVC.20,DVC.40, & DVC.60

                SECTION 11
                     THE NEXT SECTION DIVIDES -2**16 BY 2**14 TO TEST STATE DVN.20

                SECTION 12
                     THE NEXT SECTION DIVIDES 100 000200 BY -177 TO TEST STATES
                     DVD.00 AND DVD.10.

2584    TEST 15 MTP*DM0

                     IF FORK A FAILS EXECUTION WILL GO TO RSD.00.

                     THE ONLY OTHER POSSIBLE FAILURES WOULD BE IN ROM STATES
                     MTP.00 OR MTP.10.

                     NOTE: THIS TEST ONLY TESTS THE CPU FUNCTIONS OF THIS INSTRUCTION.
                     THE MEMORY MANAGEMENT TEST VERIFIES THE INTERMODE TRANSFER.
                     AS FAR AS THE CPU IS CONCERNED THERE IS NO DIFFERENCE
                     BETWEEN MTPI AND MTPD.

                     ROM FLOW-45,151,146,205

2627.   TEST 16 MTP*DM1

                     IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
                     THIS WILL ONLY HAPPEN IF RACF E20(4) IS STUCK HIGH.

                     THIS TEST ENSURES STATE MTP.10 RELOADS THE
                     DR IF THE DESTINATION IS R6 AND THAT IT PUTS THE PC IN THE
                     DR IF THE DESTINATION FIELD IS R7.

                     ROM FLOW-45,151,146,111,155,312

2670    TEST 17 MFP*DM0

                     IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
                     IF ANYTHING ELSE FAILS, THEN A ROM STATE IS BAD.

                     ROM FLOW-46,304,250,222,300

2710    TEST 20 MFP*DM2

                     IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
                     IF FORK B FAILS EXECUTION WILL ALSO GO TO RSD.00 BUT THE DR
                     WILL HAVE BEEN INCREMENTED. IF ANYTHING ELSE FAILS THEN
                     STATE MFP.00 IS BAD.

                     ROM FLOW-2,175,66,250,222,300
```

```
2747    TEST 21 BPT

                FORK A SHOULD NOT FAIL.
                IF THE TRAP VECTOR LOGIC FAILS THE TRAP VECTOR WOULD
                COME OUT TO BE 4.
                 THE ONLY OTHER FAILURE WOULD BE TRP.00.
                 IF THIS STATE FAILS TO LOAD THE DR THE TRAP VECTOR WILL
                BE WHATEVER IS IN R3.  IF IT FAILS TO LOAD THE BR THE OLD
                PS WILL FAIL TO BE STACKED.

2777    ALL THE LOGIC FOR (JMP+JSR)*DMO HAS BEEN TESTED.

2781    TEST 22 BIT TEST OF PIRQ REGISTER

                IF ONE OF THE BLOCK LEVELS IS STUCK HIGH OR TMCB
                PS07(0)'S STUCK HIGH OR PDRD PRIORITY=0 IS STUCK HIGH,
                A PIRQ TRAP LOOP WILL OCCUR WHEN THAT PIR LEVEL IS ENABLED.

                A COUNT PATTERN IS THEN RUN THRU THE REGISTER TO ENSURE THAT
                THE ENCODER FUNCTIONS PROPERLY.

2824    TEST 23 PIR LEVEL 1 INTERRUPT

                IF BEN13 FAILS EXECUTION WOULD GO TO ONE OF THE FOLLOWING:
                PUP.00, BRK.20, OR SER.00.
                PUP.00 WOULD START THE POWER UP ROUTINE.
                BRK.20 WOULD CAUSE A TRAP TO ZERO.
                SER.00 WOULD PUT 4 IN THE SP AND PERFORM A RED ZONE TRAP.
                IF TMCB PIRQ DOES NOT GET TO DAPE OR IF DAPE TV05*07 DOES NOT
                GO HIGH OR DOES NOT GET THRU TO THE ALU A TRAP TO 4 WILL OCCUR.
                IF THE INTERRUPT DOESN'T OCCUR AN ATTEMPT IS MADE TO ISOLATE THE
                FAILURE.

2900    TEST 24 PIR LEVEL 2 INTERRUPT

                IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
                THIS WILL ONLY HAPPEN IF TMCB E63(2) IS BAD.

                IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(2)
                IS BAD OR TMCB HONOR PIR2 IS BEING HELD HIGH.

2933    TEST 25 PIR LEVEL 3 INTERRUPT

                IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
                THIS WILL ONLY HAPPEN IF TMCB E63(3) IS BAD.

                IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(3)
                IS BAD OR TMCB HONOR PIR3 IS BEING HELD HIGH.
```

```
2966    TEST 26 PIR LEVEL 4 INTERRUPT

                IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
                THIS WILL ONLY HAPPEN IF TMCB E63(5) IS BAD.

                IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(5)
                IS BAD OR TMCA HONOR PIR 4 IS BEING HELD HIGH.

3000    TEST 27 PIR LEVEL 5 INTERRUPT

                IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
                THIS WILL ONLY HAPPEN IF TMCB E63(11) IS BAD.

                IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(11)
                IS BAD OR TMCA HONOR PIR 5 IS BEING HELD HIGH.

3035    TEST 30 PIR LEVEL 6 INTERRUPT

                IF BEN13 FAILS EXECUTION WILL GO TO BRK.20.

                THIS WILL ONLY HAPPEN IF EITHER TMCB E63(12)
                IS BAD, OR E61(1) IS BAD.

                IF THE INTERRUPT DOES NOT OCCUR LEVEL 7 IS TRYED, TO TRY
                AND ISOLATE THE FAILURE BEFORE TMCB E55(9-8).

3077    TEST 31 PIR LEVEL 7 INTERRUPT

                IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
                THIS WILL ONLY HAPPEN IF TMCB E63(6) IS BAD.

                IF THE INTERRUPT DOES NOT OCCUR THEN EITHER TMCB E70(6)

3083            IS BAD OR TMCA HONOR PIR7 IS BEING HELD HIGH.

3112    TEST 32 UNIBUS TIMEOUT

                IF TMCC ABORT DOES NOT GO HIGH OR DOES NOT GET TO RACA
                OR IF RACA ZAP DOES NOT GO LOW THE PROCESSOR WILL NOT TRAP TO 4.

                IF BEN06 FAILS THE STACKED PC WILL BE 160000 INSTEAD OF 1$-2.

                IF BEN 13 FAILS EITHER TMCC AERF(1) L IS NOT GOING LOW
                OR TMCB E53(11) IS BAD.
                A TEST IS THEN MADE TO ENSURE THAT TMCC PRIORITY CLEAR GOES LOW.
```

```
        ****************************************************************
3188    PERIPHERAL DETERMINATOR & INTERRUPT ENABLE ROUTINES
        ****************************************************************

        3189            THIS SECTION OF CODE TRYS TO FIND A DEVICE ON BR5 AND BR6.
                        WHEN IT FINDS A DEVICE IT PUTS THE ADDRESS OF THAT DEVICES

        3191            SUBROUTINE IN A LOCATION.

                        THE CODE TO INITIATE AN INTERRUPT SEQUENCE ON CERTAIN
                        DEVICES IS ALSO HERE.  WHEN A TEST REQUIRES AN INTERRUPT ON
                        A CERTAIN LEVEL IT LOOKS AT INTERX TO DETERMINE IF A
                        DEVICE IS AVAILABLE AND IF SO DOES A JSR TO THAT DEVICES
                        INTERRUPT ENABLE ROUTINE.

        3306    TEST 33 BR LEVEL 4 INTERRUPT

                        BEN 13 SHOULD NOT FAIL.
                        IF THE INTERRUPT DOESN'T OCCUR AN ATTEMPT IS MADE TO
                        ISOLATE THE FAILURE.

        3370    TEST 34 BR LEVEL 5 INTERRUPT

                        THE ONLY POSSIBLE FAILURE IS THAT TMCA HONOR BR 5
                        DOES NOT GO LOW OR TMCB E62(6) IS BAD.

        3403    TEST 35 BR LEVEL 6 INTERRUPT

                        THE ONLY POSSIBLE FAILURE IS THAT TMCA HONOR BR 6 DOES NOT GO LOW
                        OR TMCB E62(12) IS BAD.

        3436    TEST 36 YELLOW ZONE TRAP

                        A YELLOW ZONE IS FIRST ATTEMPTED WITH THE SP AT 376.
                        IF BEN 13 FAILS THE TRAP WILL NOT OCCUR.

                        IF THE PROCESSOR FAILS TO TRAP EITHER TMCD SL YEL IS
                        NOT GOING HIGH OR TMCA HONOR SLY IS NOT GOING LOW
                        OR TMCB E70(3) IS BAD OR BEN13 FAILED.

                        IF TMCC PRIORITY CLEAR DOES NOT GO LOW THE PROCESSOR WILL HANG
                        UP IN A RED ZONE TRAP LOOP.

                        A JSR WITH A BAD SP IS THEN EXECUTED TO ENSURE TMCC
                        KERNAL R6 GOES HIGH WHEN ENABLED BY ''STACK REFERENCE * KERNAL MODE''.

                        IF THE TRAP WORKS TESTS WILL BE PERFORMED TO ENSURE ALL THE
                        APPROPRIATE CONDITIONS DISABLE THE TRAP EXCEPT
                        THE PRIORITY ARBITRATOR.
```

```
3527     TEST 37 ROM FIELD CHECK OF PC MANIPULATOR STATES

                 THIS TEST EXECUTES THE MACHINE STATES THAT MANIPULATE
                 THE PC TO ENSURE THAT THE PCB ROM FIELD OR DRX ROM FIELD
                 OR SRX ROM FIELD OF THESE STATES IS FUNCTIONAL.
                 THESE STATES ARE S13.00, S45.00, MTP.10, D45.80, D45.90,
                 D45.00, AND D45.01.

3618     TEST 40 RED ZONE TRAP

                 A RED ZONE TRAP IS FIRST ATTEMPTED WITH THE SP AT 336.
                 IF BEN13 FAILS EXECUTION WILL GO TO EITHER BRK.80 OR BRK.20
                 OR PUP.00.

3623             BRK.80 WILL CAUSE THE OLD RSW AND PC TO BE STACKED ON THE
                 OLD STACK INSTEAD OF LOCATIONS 2 AND 0.
                 BRK.20  WILL MAKE IT LOOK LIKE THE RED ZONE FAILED.
                 PUP.00 WILL CAUSE A TRAP TO LOCATION 24.

                 IF THE PROCESSOR FAILS TO TRAP EITHER TMCD SL RED IS
                 NOT GOING LOW OR TMCC ABORT IS NOT GOING LOW.

                 IF UBCB ABORT RESTART FAILS TO GO LOW OR E10(13)
                 IS BAD THE PROCESSOR WILL HANG IN THE PAUSE STATE.

3726     TEST 41 BIT TEST OF STACK LIMIT REGISTER

                 FIRST A 125252 AND 52525 PATTERN IS PUT IN THE REGISTER TO ENSURE
                 THAT THE REGISTER DOESN'T HAVE ANY STUCK BITS AND THAT
                 THE DMUX SELECT AND INPUT LINES WORK.
3731
                 IF SCCE SL ADRS DOES NOT GET TO TMCD OR IF TMCD E28 OR E14
                 IS BAD THE BR WILL BE SELECTED.  THE PB REGISTER IS LOADED
                 WITH 200 SO IF TMCD LO BYTE EN DOES NOT GO LOW AN
                 ERROR WILL BE DETECTED.

3782     TEST 42 SL REGISTER COMPARATOR TEST 1

                 THIS TEST RUNS A HIGH BYTE COUNT PATTERN THRU THE BUS
3785             ADDRESS MUX FOR EACH PATTERN OF THE STACK LIMIT REGISTER.
                 FOR EACH PATTERN OF ADDRESSES THERE WILL BE ONE YEL TRAP
                 AT THE ADDRESS CORRESPONDING TO THE SL REG+340 AND A RED
                 TRAP AT EVERY ADDRESS BELOW THIS.
                 THIS TEST ONLY TESTS ADDRESSES UP TO THE I/O PAGE.
                 THE I/O PAGE ADDRESSES WILL BE TESTED SEPARATELY WITH
                 MEMORY MANAGEMENT ENABLED AND THE I/O PAGE MAPPED INTO RESIDENT
                 MEMORY

                 THE FOLLOWING ARE THE TYPES OF ERRORS THAT CAN OCCUR IN THIS TEST:
                         TYPE              DESCRIPTION
                           0         RED ZONE TRAP ON YELLOW ZONE ADDRESS
                           2         RED ZONE TRAP ON LEGAL ADDRESS
                           4         YELLOW ZONE TRAP ON RED ZONE ADDRESS
                           6         YELLOW ZONE TRAP ON LEGAL ADDRESS
```

```
                          10       NO TRAP ON RED ZONE ADDRESS
                          12       NO TRAP ON YELLOW ZONE ADDRESS

                     THE LOW BYTE ADDRESS IN THE STACK POINTER IS ALWAYS
                     340 AND WILL NOT BE TYPED ON AN ERROR.

        3808    NOTE:   IF THE LOOP ON ERROR SWITCH IS UP (SWITCH 9) THE TEST WILL
                     LOOP ON THE FIRST ERROR WITH NO ERROR TYPEOUT.  OTHERWISE ALL
                     ERRORS WILL BE RECORDED IN A TABLE AND TYPED OUT AT THE
                     END OF THE TEST.
                     IF SWITCH 3 (DISABLE MEMORY MANAGEMENT TESTS) IS NOT ON,
                     THIS TEST IS SKIPPED AND TEST 70 WILL EXECUTE.

        3904    TEST 43 ODD ADDRESS ERROR

                     BEN 13 SHOULD NOT FAIL.
                     IF THE PROCESSOR FAILS TO TRAP IN ALL SECTIONS EITHER TMCC ODD
                     ADRS ERR IS NOT GOING LOW OR TMCC BUS ERROR IS NOT GOING LOW.

                     EACH TYPE OF ODD ADDRESS ERROR IS TESTED INDIVIDUALLY TO
                     ALLOW MAXIMUM ISOLATION.
                NOTE:AN ODD ADDRESS ON ''KERNEL DATI'' CANNOT BE TESTED.
                        THIS SIGNAL COMES UP WHEN A TRAP VECTOR IS READ IN FROM THE BUS.

        4002    TEST 44 ILLEGAL INSTRUCTIONS
                     THIS TEST ENSURES THAT ILLEGAL OP CODES TRAP TO LOCATION 10.
                     ONLY THOSE OP CODES THAT HAVE A SINGLE BIT THAT
                     DISTINGUISHES THEM FROM A LEGAL INSTRUCTION WILL BE TESTED.
        4039    TEST 45  T BIT TRAP

                     IF BEN 13 FAILS EXECUTION WOULD GO TO BRK.20.
                     THIS WOULD LOOK LIKE THE TRAP DIDN'T OCCUR.

                     IF THE TRAP DOESN'T OCCUR THEN EITHER PDRD PS04(1) DOES NOT GET
                     TO TMCB AS A HIGH OR IT DOES NOT GET TO TMCB E51(10)
                     AS A LOW OR E51 IS BAD OR IRCD RTT DOES NOT GET TO TMCB AS A HIGH.

                     THIS TEST ALSO CHECKS THAT PS<08> SET WILL INHIBIT A T BIT TRAP IF
                     THIS IS A KB11-E.

        4058    TEST 46 T BIT TRAP AND RTT

                     IF THE INSTRUCTION AFTER THE RTT DOES NOT GET EXECUTED THEN
                     EITHER IRCD RTT DOES NOT GO LOW OR IT DOES NOT GET TO
                     TMCB E74(11) OR TMCB E74 IS BAD.


        4174    TEST 47 PRIORITY ARBITRATION

                     THIS TEST ASSURES THAT EACH NECESSARY INPUT TO AN HONOR FLAG
                     CAN DISABLE THAT FLAG.

                     EACH SECTION WILL PERFORM A SETUP SO THAT A TIGHT ERROR LOOP
                     CAN BE OBTAINED.
```

THE FOLLOWING IS A TABLE OF CONTENTS OF THIS TEST:

SECTION NUMBER	LEVEL UNDER TEST	DISABLING FUNCTION
1	PIR 1	BR 4
2	PIR 1	SL YELLOW
3	PIR 2	SL YELLOW
4	PIR 3	SL YELLOW
5	BR 4	PIR 4
6	BR 4	PIR 5
7	BR 4	BR 5
8	BR 4	PIR 6
9	BR 4	PIR 7
10	PIR 4	BR 5
11	PIR 4	BR 6
12	PIR 4	SL YELLOW
13	BR 5	PIR 5
14	BR 5	PIR 6
15	BR 5	PIR 7
16	PIR 5	BR 6
17	PIR 5	SL YELLOW
18	BR 6	PIR 6
19	BR 6	PIR 7
20	PIR 6	SL YELLOW
21	PIR 7	SL YELLOW

```
4646    TEST 50 GPR SET 1 SELECT TEST

                THIS TEST FIRST ENSURES THAT PSW BIT 11 SETS AND CLEARS.

4649            IT THEN ENSURES THAT GRAC GDREG SET 1 AND GSREG SET 1 GOES
                HIGH FOR THE MUX SELECTS LL,LH,AND HL.
                MUX SELECT HH WILL BE TESTED IN SUPERVISOR MODE.

4748    TEST 51 REGISTER SET 1 STUCK BIT TEST

                THIS TEST ENSURES THAT ALL BITS IN GPR'S R10 THRU R15 WORK

4807    TEST 52 PSW HIGH BYTE BIT TEST

                THIS TEST ENSURES THAT THE PRESENT AND PREVIOUS MODE BITS OF THE PSW
                CAN BE SET AND CLEARED AND THAT THEY ARE NOT STUCK TOGETHER.

4837    TEST 53 SP SELECTION TEST IN SUPER AND USER MODE

                THIS TEST ENSURES THAT THE CORRECT STACK POINTERS ARE
                SELECTED IN SUPERVISOR AND USER MODE

4895    TEST 54 SUPER AND USER SP BIT TEST

                THIS TEST ENSURES THAT THE SUPERVISOR AND USER STACK POINTERS
                DON'T HAVE ANY STUCK BITS.
```

```
4933     TEST 55 MTP*DMO*DF6*PREVIOUS MODE(SUPER*USER)

                 THIS TEST ENSURES THAT THE CORRECT SP'S ARE SELECTED WHEN EXECUTING
                 A MTP WITH DIFFERENT PREVIOUS MODE BITS SELECTED.

4991     TEST 56 MFP*DMO*DF6*PREVIOUS MODE SUPER

                 THE ONLY POSSIBLE WAY THIS TEST CAN FAIL IS THAT
                 IRCC DMO (MFP+MTP) DOES NOT GO HIGH ON MFP.
                  THIS WILL ONLY HAPPEN IF THE IR DECODE ROM HAS
                 A BAD FIELD (R(MFP+MTP)).

5023     TEST 57 UPAD 7 IN USER MODE

                 THIS TEST ENSURES THAT A UPAD 7(OCCURS IN RTI) CAUSES THE USER
                 STACK POINTER TO BE USED TO FETCH THE NEW PS AND PC.

5027
                 IF PDRD PS14(1) DOES NOT GET TO THE GSAM(ON GRAC)
                 THE TEST WILL BLOW UP.

5074     TEST 60 SPL*SUPERVISOR MODE

                 THIS TEST ENSURES THAT SPL DOES NOT LOAD THE PSW IN SUPER+USER MODE.

5090     TEST 61 PSW CLOCKING TEST

                 THIS TEST ENSURES THAT ALL THE BITS IN THE PSW GET LOADED WHEN THE
                 FOLLOWING SIGNALS ARE TRUE:
                 1) LOAD PS*KERNEL MODE, AND 2) LOAD PS*KERNEL DATI.

                 IT ALSO ENSURES THAT THE PRESET LOGIC ON BITS 11, 12, 13,
                 14, AND 15 FUNCTIONS PROPERLY.

                 FOLLOWING IS A TABLE TO DESCRIBE THE PSW FOR EACH SECTION

                 SECTION PSW AT START    PSW ON STK(OR VECTOR)   EXPEC PSW
                    1        000XXX               174XXX          174XXX
                    2        174XXX               000XXX          174XXX
                    3        040XXX               134XXX          174XXX
                    4        144XXX               000XXX          030XXX
                    5        030XXX               000XXX          000XXX

5196     TEST 62 ILLEGAL HALT
                 THIS TEST ENSURES THAT A HALT IN SUPER OR USER MODE WILL TRAP TO
                 LOCATION 4.

                 IF BEN6 FAILS EXECUTION WOULD GO TO FET.04 WHICH WOULD
                 CAUSE THE HALT TO LOOK LIKE A NOP.
                  IF TMCE SET CONF GOES LOW THE PROCESSOR WILL HALT AT LOCALOCATION 1$.

                 THE CPU ERROR REGISTER BIT 7 IS ALSO TESTED HERE.
```

```
          5229    TEST 63 WAIT

                          THIS TEST ENSURES THAT THE WAIT INSTRUCTION WORKS PROPERLY.
                          IT FIRST EXECUTES WITH A LEVEL 4 INTERRUPT.
                          THEN THE T BIT IS ENABLED TO ENSURE THAT THE INTERRUPT
                          OCCURS AND NOT THE T BIT TRAP.

          5250    TEST 64 CHECK MFPT INSTRUCTION (KB-11E/EM ONLY)

          6011    TEST 65 OPERATOR INTERVENTION TEST

                          THIS TEST ENSURES THAT THE RESET AND WAIT FLOWS PUT RO
                          AND THE PC IN THE LIGHTS. THE TEST IS ONLY EXECUTED ON
                          PASS 1 AND CAN BE DISABLED ALTOGETHER WITH SWITCH 0.

                          THE RESET LOOP IS STOPED BY CHANGING THE POSITION OF
                          SWITCH 7.

                          THE WAIT IS EXITED BY TYPING A CHARACTER
                          ON THE TERMINAL.

          *******************************************************************
  6074    END OF PASS ROUTINE
          *******************************************************************

          6076    INCREMENT THE PASS NUMBER ($PASS)
                  INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                  TYPE 'END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY''
                  WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
                  IF SW12=1 INHIBIT TRACE TRAP
                  IF THERES A MONITOR GO TO IT
                  IF THERE ISN'T JUMP TO LOOP

          *******************************************************************
  6146    SPURIOUS ERROR HANDLER
          *******************************************************************

          6147            THIS ROUTINE IS ENTERED BY AN UNEXPECTED TRAP TO 4 OR 114.
                           IF SWITCH 13 IS OFF, AN ERROR MESSAGE, THE ERROR REGISTER,
                          THE ERROR PC, AND THE TEST NUMBER ARE TYPED OUT.
                           IF SWITCH 13 IS ON, ONLY THE ERROR MESSAGE WILL BE TYPED.
```

```
        *****************************************************************
 6211   SCOPE HANDLER ROUTINE
        *****************************************************************

        6213    THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)

        6215    AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                SW14=1  LOOP ON TEST
                SW11=1  INHIBIT ITERATIONS
                SW09=1  LOOP ON ERROR
                SW08=1  LOOP ON TEST IN SWR<7:0>
                CALL
                        SCOPE           ;SCOPE=IOT

        *****************************************************************
 6277   ERROR HANDLER ROUTINE
        *****************************************************************

        6279    THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                AND GO TO ETYPDM ON ERROR
                THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                SW15=1  HALT ON ERROR
                SW13=1  INHIBIT ERROR TYPEOUTS
                SW10=1  BELL ON ERROR
                SW09=1  LOOP ON ERROR
                CALL
                        ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

        *****************************************************************
 6328   ERROR MESSAGE TYPEOUT ROUTINE
        *****************************************************************

        6329    THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
                AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
        ***********************************************************************
 6371   STACK LIMIT TEST TYPE OUT ROUTINE
        ***********************************************************************

        6372            THIS ROUTINE TYPES THE ADDRESS AND STACK LIMIT REGISTER
                VALUES THAT FAILED IN TEST 153 OR 201 IN OCTAL AND BINARY.

        ***********************************************************************
 6489   MONITOR RESTORE ROUTINE
        ***********************************************************************

        6490    THIS ROUTINE IS ENTERED BY TYPING A CHARACTER ON THE KEYBOARD
                IF THE CHARACTER IS NOT A CTRL C EXECUTION IS RETURNED TO THE
                TEST FOLLOWING THE ONE THAT WAS INTERRUPTED.
                IF IT IS A CTRL C THE MONITOR IS RESTORED AND THE
                PROCESSOR HALTS.

        ***********************************************************************
 6527   CHECK TEST SEQUENCE ROUTINE
        ***********************************************************************

        6528    THIS ROUTINE IS CALLED IN THE SCOPE ROUTINE. IT VERIFYS
                THAT A TEST HAS NOT BEEN SKIPPED.

        ***********************************************************************
 6588   TYPE ROUTINE
        ***********************************************************************

        6590    ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                NOTE1:  $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.

        6593    NOTE2:  $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                NOTE3:  $FILLC CONTAINS THE CHARACTER TO FILL AFTER.

                CALL:
                1) USING A TRAP INSTRUCTION
                        TYPE    ,MESADR         ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
                OR
                        TYPE
                        MESADR

                2) USING A JSR INSTRUCTION
                        MOV     PS,-(SP)        ;;PUSH PROCESSOR STATUS WORD ON THE STACK
                        JSR     PC,$TYPE        ;;CALL TYPE ROUTINE
                        MESADDR                 ;;FIRST ADRESS OF MESSAGE
```

```
          *****************************************************************
    6661  BINARY TO OCTAL (ASCII) AND TYPE
          *****************************************************************

          6663    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

          *****************************************************************
    6739  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
          *****************************************************************

          6741    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
```

```
        ****************************************************************
6807    TRAP DECODER
        ****************************************************************

        6809    THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE ''TRAP'' INSTRUCTION
                AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
                OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
                GO TO THAT ROUTINE.

        ****************************************************************
6822    TRAP TABLE
        ****************************************************************

        6824    THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                BY THE ''TRAP'' INSTRUCTION.

        ****************************************************************
6837    POWER DOWN AND UP ROUTINES
        ****************************************************************
```

```
                                  .TITLE  PDP 11/70-74MP CPU DIAGNOSTIC PART 2
                                  ;*COPYRIGHT (C) 1975,1979
                                  ;*DIGITAL EQUIPMENT CORP.
                                  ;*MAYNARD, MASS. 01754
                                  ;*
                                  ;*PROGRAM BY DONALD W. MONROE
                                  ;*
                                  ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                  ;*PACKAGE (MAINDEC-11-DZQAC-A5).
                                  ;*
          000001                  $TN=1
          177400                          $SWR=177400
```

```
.SBTTL  OPERATIONAL SWITCH SETTINGS
;*
;*      SWITCH              USE
;*      ------         ---------------------
;*        15           HALT ON ERROR
;*        14           LOOP ON TEST
;*        13           INHIBIT ERROR TYPEOUTS
;*        12           INHIBIT TRACE TRAP
;*        11           INHIBIT ITERATIONS
;*        10           BELL ON ERROR
;*         9           LOOP ON ERROR
;*         8           LOOP ON TEST IN SWR<7:0>
;*         7           NOT USED
;*         6           SKIP BR6 TEST
;*         5           SKIP BR5 TEST
;*         4           SKIP BR4 TEST
;*         3           NOT USED
;*         2           NOT USED
;*         1           NOT USED
;*         0           DISABLE OPERATOR INTERVENTION TEST
```

```

                                 .SBTTL  BASIC DEFINITIONS

                                 ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
            001100               STACK=  1100                ;;FIRST ADDRESS OF THE STACK
            001100               KERSTK= STACK               ;;KERNEL STACK
            000700               SUPSTK= STACK-200           ;;SUPERVISOR STACK
            000600               USESTK= STACK-300           ;;USER STACK
                                 .EQUIV  EMT,ERROR           ;;BASIC DEFINITION OF ERROR CALL
                                 .EQUIV  IOT,SCOPE           ;;BASIC DEFINITION OF SCOPE CALL
            177776               PS=     177776              ;;PROCESSOR STATUS WORD
                                 .EQUIV  PS,PSW
            177774               STKLMT= 177774              ;;STACK LIMIT REGISTER
            177772               PIRQ=   177772              ;;PROGRAM INTERRUPT REQUEST REGISTER
            177570               SWR=    177570              ;;SWITCH REGISTER
            177570               DISPLAY=SWR

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

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

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

                                 ;*"SWITCH REGISTER" SWITCH DEFINITIONS
            100000               SW15=   100000
```

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

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

                ;*BASIC "CPU" TRAP VECTOR ADDRESSES
000004          ERRVEC= 4                       ;;TIME OUT AND OTHER ERRORS
```

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

                        .SBTTL  CACHE   REGISTER DEFINITIONS


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


                        .SBTTL  CPU REGISTER DEFINITIONS


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





                        .SBTTL  MEMORY MANAGEMENT DEFINITIONS


                        ;*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES

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

                        ;*USER "I" PAGE DESCRIPTOR REGISTERS

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

                                ;*USER "D" PAGE DESCRIPTOR REGISTORS

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

                                ;*USER "I" PAGE ADDRESS REGISTERS

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

                                ;*USER "D" PAGE ADDRESS REGISTERS

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

                                ;*SUPERVISOR "I" PAGE DESCRIPTOR REGISTERS

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

                                ;*SUPERVISOR "D" PAGE DESCRIPTOR REGISTERS

        172220                  SDPDR0= 172220
        172222                  SDPDR1= 172222
        172224                  SDPDR2= 172224
        172226                  SDPDR3= 172226
        172230                  SDPDR4= 172230
        172232                  SDPDR5= 172232
        172234                  SDPDR6= 172234
        172236                  SDPDR7= 172236

                                ;*SUPERVISOR "I" PAGE ADDRESS REGISTERS

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

                                ;*SUPERVISOR "D" PAGE ADDRESS REGISTERS

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

                                ;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

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

                                ;*KERNEL "D" PAGE DESCRIPTOR REGISTERS

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

                                ;*KERNEL "I" PAGE ADDRESS REGISTERS

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

                                ;*KERNEL 'D'' PAGE ADDRESS REGISTERS

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




                                .SBTTL  UNIBUS MAP REGISTER DEFINITIONS


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


        170200                  MAPL00 = 170200
        170202                  MAPH00 = 170202
        170204                  MAPL01 = 170204
        170206                  MAPH01 = 170206
        170210                  MAPL02 = 170210
        170212                  MAPH02 = 170212
        170214                  MAPL03 = 170214
        170216                  MAPH03 = 170216
        170220                  MAPL04 = 170220
        170222                  MAPH04 = 170222
        170224                  MAPL05 = 170224
        170226                  MAPH05 = 170226
        170230                  MAPL06 = 170230
        170232                  MAPH06 = 170232
        170234                  MAPL07 = 170234
        170236                  MAPH07 = 170236
        170240                  MAPL10 = 170240
        170242                  MAPH10 = 170242
        170244                  MAPL11 = 170244
        170246                  MAPH11 = 170246
        170250                  MAPL12 = 170250
        170252                  MAPH12 = 170252
        170254                  MAPL13 = 170254
        170256                  MAPH13 = 170256
        170260                  MAPL14 = 170260
        170262                  MAPH14 = 170262
        170264                  MAPL15 = 170264
        170266                  MAPH15 = 170266
        170270                  MAPL16 = 170270
        170272                  MAPH16 = 170272
        170274                  MAPL17 = 170274
        170276                  MAPH17 = 170276
        170300                  MAPL20 = 170300
        170302                  MAPH20 = 170302
        170304                  MAPL21 = 170304
        170306                  MAPH21 = 170306
        170310                  MAPL22 = 170310
        170312                  MAPH22 = 170312
        170314                  MAPL23 = 170314
        170316                  MAPH23 = 170316
        170320                  MAPL24 = 170320
        170320                  MAPH24 = 170320
        170324                  MAPL25 = 170324
        170326                  MAPH25 = 170326
        170330                  MAPL26 = 170330
        170332                  MAPH26 = 170332
        170334                  MAPL27 = 170334
        170336                  MAPH27 = 170336
        170340                  MAPL30 = 170340
        170342                  MAPH30 = 170342
        170344                  MAPL31 = 170344
        170346                  MAPH31 = 170346
        170350                  MAPL32 = 170350
        170352                  MAPH32 = 170352
        170354                  MAPL33 = 170354
        170356                  MAPH33 = 170356
        170360                  MAPL34 = 170360
        170362                  MAPH34 = 170362
        170364                  MAPL35 = 170364
        170366                  MAPH35 = 170366
        170370                  MAPL36 = 170370
        170372                  MAPH36 = 170372
        170374                  MAPL37 = 170374
        170376                  MAPH37 = 170376
                                .EQUIV  MAPL00,MAPL0
                                .EQUIV  MAPH00,MAPH0
                                .EQUIV  MAPL01,MAPL1
                                .EQUIV  MAPH01,MAPH1
                                .EQUIV  MAPL02,MAPL2
                                .EQUIV  MAPH02,MAPH2
                                .EQUIV  MAPL03,MAPL3
                                .EQUIV  MAPH03,MAPH3
                                .EQUIV  MAPL04,MAPL4
                                .EQUIV  MAPH04,MAPH4
                                .EQUIV  MAPL05,MAPL5
                                .EQUIV  MAPH05,MAPH5
                                .EQUIV  MAPL06,MAPL6
                                .EQUIV  MAPH06,MAPH6
                                .EQUIV  MAPL07,MAPL7
                                .EQUIV  MAPH07,MAPH7




        172544                  PLKC=172544
```

```


                                  .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  004742                   JMP     @#START         ;;JUMP TO STARTING ADDRESS OF PROGRAM
                                  ;;*************************************************************

                                  .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

         000204                           $SVPC=.                 ;;SAVE LOCATION COUNTER
         000046                           .=46                    ;;SET LOCATION COUNTER
 000046  032572                           .WORD   $ENDAD          ;;SET LOC.46 TO ADDRESS $ENDAD
         000052                           .=52                    ;;SET LOCATION COUNTER
 000052  000000                           .WORD   0               ;;SET LOC.52 TO ZERO
         000204                           .=$SVPC                 ;; RESTORE LOCATION COUNTER
```

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

                                        .SBTTL  COMMON TAGS

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

        001100                                  .=1100

001100                                  $CMTAG:                                 ;;START OF COMMON TAGS
001100  000000                          $PASS:   .WORD   0                      ;;CONTAINS PASS COUNT
001102     000                          $TSTNM:  .BYTE   0                      ;;CONTAINS THE TEST NUMBER
001103     000                          $ERFLG:  .BYTE   0                      ;;CONTAINS ERROR FLAG
001104  000000                          $ICNT:   .WORD   0                      ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                          $LPADR:  .WORD   0                      ;;CONTAINS SCOPE LOOP
001110  000000                          $LPERR:  .WORD   0                      ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                          $ERTTL:  .WORD   0                      ;;CONTAINS TOTAL ERRORS DETECTED
001114     000                          $ITEMB:  .BYTE   0                      ;;CONTAINS ITEM CONTROL BYTE
001115     001                          $ERMAX:  .BYTE   1                      ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                          $ERRPC:  .WORD   0                      ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                          $GDADR:  .WORD   0                      ;;CONTAINS  OF 'GOOD' DATA
001122  000000                          $BDADR:  .WORD   0                      ;;CONTAINS  OF 'BAD' DATA
001124  000000                          $GDDAT:  .WORD   0                      ;;CONTAINS 'GOOD' DATA
001126  000000                          $BDDAT:  .WORD   0                      ;;CONTAINS 'BAD' DATA
001130  000000  000000  000000                   .WORD   0,0,0                  ;;RESERVED--NOT TO BE USED
001136  177560                          $TKS:    177560                         ;;TTY KBD STATUS
001140  177562                          $TKB:    177562                         ;;TTY KBD BUFFER
001142  177564                          $TPS:    177564                         ;;TTY PRINTER STATUS REG.
001144  177566                          $TPB:    177566                         ;;TTY PRINTER BUFFER REG.
001146     000                          $NULL:   .BYTE   0                      ;;CONTAINS NULL CHARACTER FOR FILLS
001147     002                          $FILLS:  .BYTE   2                      ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001150     012                          $FILLC:  .BYTE   12                     ;;INSERT FILL CHARS. AFTER A "LINE FEED"
001151     000                          $TPFLG:  .BYTE   0                      ;;"TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
001152  000000                          $REGAD:  .WORD   0                      ;;CONTAINS THE  FROM
                                                                                ;;WHICH  ($REG0) WAS OBTAINED
001154  000000                          $REG0:   .WORD   0                      ;;CONTAINS (($REGAD)+0)
001156  000000                          $REG1:   .WORD   0                      ;;CONTAINS (($REGAD)+2)
001160  000000                          $REG2:   .WORD   0                      ;;CONTAINS (($REGAD)+4)
001162  000000                          $TMP0:   .WORD   0                      ;;USER DEFINED
001164  000000                          $TMP1:   .WORD   0                      ;;USER DEFINED
001166  000000                          $TMP2:   .WORD   0                      ;;USER DEFINED
001170  000000                          $TMP3:   .WORD   0                      ;;USER DEFINED
001172  000000                          $TIMES: 0                               ;;MAX. NUMBER OF ITERATIONS
001174  000000                          $ESCAPE:0                               ;;ESCAPE ON ERROR
001176  177607  000377                  $BELL:   .ASCIZ  <207><377><377>        ;;CODE FOR BELL
001202     077                          $QUES:   .ASCII  /?/                    ;;QUESTION MARK
001203     015                          $CRLF:   .ASCII  <15>                   ;;CARRIAGE RETURN
001204  000012                          $LF:     .ASCIZ  <12>                   ;;LINE FEED
001206  000000                          $ERPSW:  .WORD   0                      ;ERROR PSW
001210  000000                          $$TSTNM:.WORD    0                      ;TEST NUMBER STORAGE
001212  000100                          $PR2:    .WORD   PR2                    ;PRIORITY LEVEL 2
001214  000240                          $PR5:    .WORD   PR5                    ;PRIORITY LEVEL 5
001216  000000                          $EPIRQ:  .WORD   0                      ;ERROR PIRQ
001220  000000                          E1STKLM:.WORD    0                      ;STACK LIMIT REGISTER ERROR 1
001222  000000                          E2STKLM:.WORD    0                      ;STACK LIMIT REGISTER ERROR 2
001224  000000                          INTER5:  .WORD   0                      ;ADDRESS OF BR5 INTER SUBROUTINE
```

```
001226  000000      INT5VEC:.WORD   0               ;BR 5 INTERRUPT VECTOR
001230  000000      INT5ST: .WORD   0               ;BR 5 STATUS REG
001232  000000      INTER6: .WORD   0               ;ADDRESS OF BR6 INTERRUPT SUBROUTINE
001234  000000      INT6VEC:.WORD   0               ;BR 6 INTERRUPT VECTOR
001236  000000      INT6ST: .WORD   0               ;BR 6 STATUS REG
001240  172040      RSCS1:  .WORD   172040          ;ADDRESS OF RS STATUS REGISTER
001242  000204      RSVEC:  .WORD   204             ;ADDRESS OF RS VECTOR
001244  176700      RPCS1:  .WORD   176700          ;ADDRESS OF RP STATUS REGISTER
001246  000254      RPVEC:  .WORD   254             ;ADDRESS OF RP VECTOR
001250  177404      RKCS1:  .WORD   177404          ;ADDRESS OF RK STATUS REGISTER
001252  000220      RKVEC:  .WORD   220             ;ADDRESS OF RK VECTOR
001254  172440      TMCS1:  .WORD   172440          ;ADDRESS OF TM STATUS REG
001256  000224      TMVEC:  .WORD   224             ;ADDRESS OF TM VECTOR
001260  177546      LKSTAT: .WORD   177546          ;ADDRESS OF LINE CLOCK STATUS REGISTER
001262  000100      LKVEC:  .WORD   100             ;ADDRESS OF LINE CLOCK VECTOR
001264  172540      PLKSTAT:.WORD   172540          ;ADDRESS OF PROG LINE CLOCK STATUS REG
001266  000104      PLKVEC: .WORD   104             ;ADDRESS OF PROG LINE CLOCK VECTOR
001270  000000      NEXTTST:.WORD   0               ;ADDRESS OF NEXT TEST
001272     000      KB11E:  .BYTE   0               ;KB-11E/EM WITHOUT MP CACHE
001273     000      KB11EM: .BYTE   0               ;KB-11E/EM WITH MP MODS

                    ;OPCODE FOR MFPT INSTRUCTION (AVAILABLE ON KB11-E AND KB11-EM ONLY)
        000007              MFPT=7
```

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

                        .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


 001274                 $ERRTB:

 001274  036546         ITEM1:  EM1                          ;EITHER SPL FAILED OR BITS STUCK
 001276  036614                 DH1                          ;ERRORPC SPL 5 PSW   SPL 2 PSW  TEST NUMBER
                                                             ;         EXPECT ACTUAL  EXPECT ACTUAL
 001300  036734                 DT1                          ;$ERRPC,$PR5,$ERPSW,$PR2,$TMP0
 001302  036752         ITEM2:  EM2                          ;PR5 LOADS OK BUT PR2 DOESN'T
 001304  036614                 DH1
 001306  036734                 DT1
 001310  036773         ITEM3:  EM3                          ;PR2 LOADS OK BUT PR5 DOESN'T
 001312  036614                 DH1
 001314  036734                 DT1
 001316  037020         ITEM4:  EM4                          ;FORK A FAILED TO D30.00
 001320  037063                 DH4                          ;ERRORPC TEST NUMBER
 001322  037110                 DT4                          ;$ERRPC,$$TSTNM
 001324  037116         ITEM5:  EM5                          ;FORK A FAILED TO RSD.02
 001326  037063                 DH4
 001330  037110                 DT4
 001332  037204         ITEM6:  EM6                          ;RESET DID NOT WORK
 001334  037063                 DH4
 001336  037110                 DT4
 001340  037237         ITEM7:  EM7                          ;FORK A FAILED INTO TRP.02
 001342  037063                 DH4
 001344  037110                 DT4
 001346  037253         ITEM10: EM10                         ;FORK A FAILED INTO WAT.00
 001350  037063                 DH4
 001352  037110                 DT4
 001354  037267         ITEM11: EM11                         ;FORK A FAILED TO MTP.00
 001356  037063                 DH4
 001360  037110                 DT4
 001362  037326         ITEM12: EM12                         ;FORK A FAILED TO MFP.80
 001364  037063                 DH4
 001366  037110                 DT4
 001370  037365         ITEM13: EM13                         ;PCB DID NOT LOAD FROM R5
 001372  037063                 DH4
 001374  037110                 DT4
 001376  037415         ITEM14: EM14                         ;MARK DID NOT LOAD SP PROPERLY
 001400  037445                 DH14                         ;ERRORPC         SP          TEST NUMBER
                                                             ;          EXPECT  ACTUAL
 001402  037526                 DT14                         ;$ERRPC,$REG1,$REG0,$$TSTNM
```

```
001404  037540        ITEM15: EM15          ;R5 DID NOT LOAD PROPERLY
001406  037570                DH15          ;ERRORPC        R5          TEST NUMBER
001410  037526                DT14
001412  037651        ITEM16: EM16          ;FORK A FAILED TO RSD.00
001414  037063                DH4
001416  037110                DT4
001420  037705        ITEM17: EM17          ;FORK A FAILED TO D67.01
001422  037063                DH4
001424  037110                DT4
001426  037762        ITEM20: EM20          ;R1 SHIFTED RIGHT INSTEAD OF LEFT
001430  037063                DH4
001432  037110                DT4
001434  040053        ITEM21: EM21          ;R1 DID NOT SHIFT
001436  037063                DH4
001440  037110                DT4
001442  040073        ITEM22: EM22          ;R1 SHIFTED BUT CARRY DID NOT SET
001444  037063                DH4
001446  037110                DT4
001450  040170        ITEM23: EM23          ;R1 SHIFTED LEFT INSTEAD OF RIGHT
001452  037063                DH4
001454  037110                DT4
001456  040260        ITEM24: EM24          ;SHIFT RIGHT DID NOT SIGN FILL
001460  037063                DH4
001462  037110                DT4
001464  040313        ITEM25: EM25          ;R1 SHIFTED BUT DON'T KNOW WHERE
001466  040345                DH25          ;ERRORPC        R5          TEST NUMBER
                                            ;          EXPECT  ACTUAL
001470  040420                DT25          ;$ERRPC,$TMP1,$REG1,$$TSTNM
001472  040432        ITEM26: EM26          ;SHIFT OK BUT CARRY DID NOT LOAD
001474  037063                DH4
001476  037110                DT4
001500  040472        ITEM27: EM27          ;ASH.20 DID NOT LOAD CC'S CORRECTLY
001502  041456                DH42          ;ERRORPC        PSW         TEST NUMBER
                                            ;          EXPECT  ACTUAL
001504  041534                DT42          ;$ERRPC,$TMP0,$ERPSW,$$TSTNM
001506  040534        ITEM30: EM30          ;R1 SHIFTED WITH A SHIFT COUNT OF 0
001510  037063                DH4
001512  037110                DT4
001514  040572        ITEM31: EM31          ;ASH.40 DID NOT LOAD CC'S CORRECTLY
001516  041456                DH42
001520  041534                DT42
001522  040634        ITEM32: EM32          ;FORK A FAILED TO RSD.00
001524  037063                DH4
001526  037110                DT4
001530  040700        ITEM33: EM33          ;STATE ASH.00 FAILED
001532  037063                DH4
001534  037110                DT4
001536  040716        ITEM34: EM34          ;FORK B FAILED INTO MUL.00
001540  037063                DH4
001542  037110                DT4
001544  040751        ITEM35: EM35          ;FORK B FAILED TO RSD.00
001546  037063                DH4
001550  037110                DT4
001552  041067        ITEM36: EM36          ;FORK A FAILED TO RSD.00
001554  037063                DH4
001556  037110                DT4
```

```
001560  041152          ITEM37: EM37                    ;RACE E45 BAD (AFIR04(1)*MUL:ASHC+MFP))
001562  037063                  DH4
001564  037110                  DT4
001566  041222          ITEM40: EM40                    ;RACE E33 BAD (AFIR05(1)*(MUL:ASHC+MFP))
001570  037063                  DH4
001572  037110                  DT4
001574  041272          ITEM41: EM41                    ;R0 DID NOT SIGN FILL ON RIGHT SHIFT
001576  041335                  DH41                    ;ERROR PC        R0          TEST NUMBER
                                                        ;           EXPECT  ACTUAL
001600  041414                  DT41                    ;$ERRPC,$TMP0,$REG0,$$TSTNM
001602  041426          ITEM42: EM42                    ;BAD CC'S ON RIGHT SHIFT
001604  041456                  DH42                    ;ERRORPC     PSW  TEST NUMBER
                                                        ;         EXPECT ACTUAL
001606  041534                  DT42                    ;$ERRPC,$TMP0,$ERPSW,$$TSTNM
001610  041546          ITEM43: EM43                    ;R0<0> DID NOT GO TO R1<15>
001612  041600                  DH43                    ;ERRORPC          R0                R1           TEST NUMBER
                                                        ;           EXPECT  ACTUAL  EXPECT  ACTUAL
001614  041710                  DT43                    ;$ERRPC,$TMP0,$REG0,$TMP1,$REG1,$$TSTNM
001616  041726          ITEM44: EM44                    ;R0 DID NOT SHIFT LEFT PROPERLY
001620  041600                  DH43
001622  041710                  DT43
001624  041772          ITEM45: EM45                    ;BAD CC'S ON LEFT SHIFT
001626  041456                  DH42
001630  041534                  DT42
001632  040313          ITEM46: EM25                    ;R1 DID NOT SHIFT LEFT PROPERLY
001634  041600                  DH43
001636  041710                  DT43
001640  042150          ITEM47: EM47                    ;BAD CC'S ON NO SHIFT
001642  041456                  DH42
001644  041534                  DT42
001646  042175          ITEM50: EM50                    ;R1 DID NOT ROTATE PROPERLY
001650  041600                  DH43
001652  041710                  DT43
001654  042227          ITEM51: EM51                    ;BITS STUCK IN SC (52 PATTERN)
001656  042267                  DH51                    ;ERRORPC          R0                R1               C BIT
                                                        ;           EXPECT  ACTUAL  EXPECT  ACTUAL  EXPECT  ACTUAL
001660  042426                  DT51                    ;$ERRPC,$TMP0,$REG0,$TMP1,$REG1,$TMP2,$ERPSW,$$TSTNM
001662  042450          ITEM52: EM52                    ;BITS STUCK IN SC (25 PATTERN)
001664  042267                  DH51
001666  042426                  DT51
001670  042523          ITEM53: EM53                    ;IRCB B FORK MUX INPUT B3 NOT GOING LOW
001672  042267                  DH51
001674  042426                  DT51
001676  042556          ITEM54: EM54                    ;STATE ASC.00 FAILED
001700  041600                  DH43
001702  041710                  DT43
001704  042574          ITEM55: EM55                    ;FORK A FAILED TO RSD.00
001706  037063                  DH4
001710  037110                  DT4
001712  042652          ITEM56: EM56                    ;EITHER GRAD DR00 STUCK H OR RACK E64 BAD
001714  041600                  DH43
001716  041710                  DT43
001720  042723          ITEM57: EM57                    ;EITHER GRAD DR00 STUCK LOW OR RACK E64 BAD
001722  041600                  DH43
001724  041710                  DT43
001726  042773          ITEM60: EM60                    ;EITHER GRAJ SC=0 NOT GETTING TO RACK E50
```

```
001730  041600                 DH43             ;OR RACK E50 BAD
001732  041710                 DT43
001734  043074       ITEM61:   EM61             ;INSTRUCTION FAILED TO LOAD R0 & R1 CORRECTLY
001736  041600                 DH43             ;ON POSITIVE
001740  041710                 DT43
001742  043152       ITEM62:   EM62             ;BAD CC'S
001744  041456                 DH42
001746  041534                 DT42
001750  043214       ITEM63:   EM63             ;R0 DID NOT LOAD ON NEG.
001752  041600                 DH43
001754  041710                 DT43
001756  043254       ITEM64:   EM64             ;R1 DID NOT LOAD ON NEG.
001760  041600                 DH43
001762  041710                 DT43
001764  043314       ITEM65:   EM65             ;BAD CC'S DUE TO STATE MUL.50
001766  041456                 DH42
001770  041534                 DT42
001772  043351       ITEM66:   EM66             ;C DID NOT SET ON OVERFLOW
001774  037063                 DH4
001776  037110                 DT4
002000  043402       ITEM67:   EM67             ;C DID NOT SET ON UNDERFLOW
002002  037063                 DH4
002004  037110                 DT4
002006  043434       ITEM70:   EM70             ;STATE MUL.00 FAILED
002010  041600                 DH43
002012  041710                 DT43
002014  042574       ITEM71:   EM55             ;BAD FIELD IN IR DECODE ROM
002016  037063                 DH4
002020  037110                 DT4
002022  043452       ITEM72:   EM72             ;STATE ASH.30 DID NOT LOAD CC'S CORRECTLY
002024  041456                 DH42
002026  041534                 DT42
002030  043514       ITEM73:   EM73             ;STATE ASH.41 FAILED
002032  040345                 DH25
002034  040420                 DT25
002036  043532       ITEM74:   EM74             ;BAD CC'S DUE TO STATE ASH.41
002040  041456                 DH42
002042  041534                 DT42
002044  043574       ITEM75:   EM75             ;BEN2 FAILED
002046  037063                 DH4
002050  037110                 DT4
002052  043661       ITEM76:   EM76             ;BAD CC'S DUE TO DVE.00
002054  041456                 DH42
002056  041534                 DT42
002060  043704       ITEM77:   EM77             ;BAD CC'S DUE TO DVC.70
002062  041456                 DH42
002064  041534                 DT42
002066  043727       ITE100:   EM100            ;BEN16 FAILED
002070  037063                 DH4
002072  037110                 DT4
002074  043757       ITE101:   EM101            ;BEN4 FAILED
002076  037063                 DH4
002100  037110                 DT4
002102  044103       ITE102:   EM102            ;QUOTIENT OK REMAINDER BAD
002104  041600                 DH43
002106  041710                 DT43
```

```
002110  044155          ITE103: EM103                   ;BEN3 FAILED
002112  037063                  DH4
002114  037110                  DT4
002116  044204          ITE104: EM104                   ;BEN04 STUCK TO DIV SUB
002120  037063                  DH4
002122  037110                  DT4
002124  044274          ITE105: EM105                   ;CAN'T DETERMINE CAUSE OF FAILURE
002126  041600                  DH43
002130  041710                  DT43
002132  044325          ITE106: EM106                   ;BEN16 FAILED TO DIV.50
002134  037063                  DH4
002136  037110                  DT4
002140  044274          ITE107: EM105                   ;EM107 SAME AS EM105
002142  041600                  DH43
002144  041710                  DT43
002146  044415          ITE110: EM110                   ;BEN04*-N FAILED
002150  037063                  DH4
002152  037110                  DT4
002154  044505          ITE111: EM111                   ;BEN02 FAILED
002156  037063                  DH4
002160  037110                  DT4
002162  044535          ITE112: EM112                   ;BEN05 FAILED
002164  037063                  DH4
002166  037110                  DT4
002170  044655          ITE113: EM113                   ;BEN16 FAILED (RACK E50(B0))
002172  037063                  DH4
002174  037110                  DT4
002176  044704          ITE114: EM114                   ;BEN16 FAILED (RACK E64(B0))
002200  037063                  DH4
002202  037110                  DT4
002204  044733          ITE115: EM115                   ;ROM STATE FAILED
002206  041600                  DH43
002210  041710                  DT43
002212  044103          ITE116: EM102                   ;QUOTIENT OK, REMAINDER BAD
002214  041600                  DH43
002216  041710                  DT43
002220  045001          ITE117: EM117                   ;BAD CC'S IN DVC.90 OR RACK E63 BAD
002222  041456                  DH42
002224  041534                  DT42
002226  045056          ITE120: EM120                   ;BEN05 DIV QUIT (N(0)*SR15(0)) DID NOT
002230  037063                  DH4                     ;GO LOW OR RACK E63(C0) STUCK HIGH
002232  037110                  DT4
002234  045145          ITE121: EM121                   ;CC'S BAD DUE TO DIV.30 OR DVE.20
002236  041456                  DH42
002240  041534                  DT42
002242  045215          ITE122: EM122                   ;GRAJ E5 BAD
002244  037063                  DH4
002246  037110                  DT4
002250  045255          ITE123: EM123                   ;RACK E63(D0) STUCK LOW
002252  037063                  DH4
002254  037110                  DT4
002256  045304          ITE124: EM124                   ;CC'S DID NOT LOAD PROPERLY
002260  041456                  DH42
002262  041534                  DT42
002264  045336          ITE125: EM125                   ;EITHER GRAJ E5(N(1)*SR15(1)) BAD
002266  041600                  DH43                    ;OR ROM STATE BAD
```

```
002270  041710                  DT43
002272  044103         ITE126:  EM102                    ;QUOT. OK REMAINDER BAD
002274  041600                  DH43
002276  041710                  DT43
002300  045411         ITE127:  EM127                    ;QUOT. BAD, REMAINDER OK
002302  041600                  DH43
002304  041710                  DT43
002306  045463         ITE130:  EM130                    ;QUOTIENT BAD
002310  041600                  DH43
002312  041710                  DT43
002314  044103         ITE131:  EM102                    ;QUOT. OK, REMAINDER BAD
002316  041600                  DH43
002320  041710                  DT43
002322  045463         ITE132:  EM130                    ;QUOT. BAD
002324  041600                  DH43
002326  041710                  DT43
002330  044103         ITE133:  EM102                    ;QUOT. OK, REMAIN. BAD
002332  041600                  DH43
002334  041710                  DT43
002336  045524         ITE134:  EM134                    ;BAD CC'S ON DIVISION OVERFLOW
002340  041456                  DH42
002342  041534                  DT42
002344  045573         ITE135:  EM135                    ;R0 DID NOT LOAD CORRECTLY
002346  041335                  DH41
002350  041414                  DT41
002352  045602         ITE136:  EM136                    ;THE SP DID NOT INCREMENT
002354  037063                  DH4
002356  037110                  DT4
002360  045304         ITE137:  EM124                    ;CC'S DID NOT LOAD CORRECTLY
002362  041456                  DH42
002364  041534                  DT42
002366  045635         ITE140:  EM140                    ;FORK A FAILED
002370  037063                  DH4
002372  037110                  DT4
002374  045672         ITE141:  EM141                    ;STATE MTP.10 FAILED
002376  037063                  DH4
002400  037110                  DT4
002402  045731         ITE142:  EM142                    ;SP LOADED INCORRECTLY
002404  037445                  DH14
002406  037526                  DT14
002410  045757         ITE143:  EM143                    ;STATE MTP.00 DID NOT PUT PCB IN DR
002412  037063                  DH4
002414  037110                  DT4
002416  046013         ITE144:  EM144                    ;FORK A FAILED
002420  037063                  DH4
002422  037110                  DT4
002424  046042         ITE145:  EM145                    ;STATE MFP.10 DID NOT DEC. THE SP
002426  037063                  DH4
002430  037110                  DT4
002432  046101         ITE146:  EM146                    ;R0 DID NOT GET PUT ON THE STACK
002434  037063                  DH4
002436  037110                  DT4
002440  046140         ITE147:  EM147                    ;BAD CC'S
002442  041456                  DH42
002444  041534                  DT42
002446  045635         ITE150:  EM140                    ;RACF X/CLASS DOES NOT GO HIGH
```

```
002450  037063                  DH4
002452  037110                  DT4
002454  046171          ITE151: EM151                   ;STATE MFP.00 IS BAD
002456  037063                  DH4
002460  037110                  DT4
002462  046204          ITE152: EM152                   ;BAD CC'S
002464  041456                  DH42
002466  041534                  DT42
002470  037651          ITE153: EM16                    ;RACE X/CLASS DOES NOT GO HIGH
002472  037063                  DH4
002474  037110                  DT4
002476  042574          ITE154: EM55                    ;R(NUL:ASHC+MFP) FIELD OF IR ROM BAD
002500  037063                  DH4
002502  037110                  DT4
002504  046233          ITE155: EM155                   ;STATE TRP.01 FAILED
002506  037063                  DH4
002510  037110                  DT4
002512  046264          ITE156: EM156                   ;TRAP VECTOR DECODE FAILED
002514  037063                  DH4
002516  037110                  DT4
002520  046404          ITE157: EM157                   ;STATE TRP.01 FAILED
002522  037063                  DH4
002524  037110                  DT4
002526  046435          ITE160: EM160                   ;STATE TRP.01 FAILED
002530  037063                  DH4
002532  037110                  DT4
002534  046466          ITE161: EM161                   ;TRAP VECTOR DECODE FAILED
002536  037063                  DH4
002540  037110                  DT4
002542  046563          ITE162: EM162                   ;STATE TRP.01 FAILED
002544  037063                  DH4
002546  037110                  DT4
002550  046614          ITE163: EM163                   ;BIT FAILED IN PIRQ REG
002552  046643                  DH163                   ;ERRORPC      PIRQ        TEST NUMBER
                                                        ;          EXPECT  ACTUAL
002554  046722                  DT163                   ;$ERRPC,$TMP0,$EPIRQ,$$TSTNM
002556  046734          ITE164: EM164                   ;STATE TST.10 HAS BAD BEN FIELD
002560  037063                  DH4
002562  037110                  DT4
002564  046765          ITE165: EM165                   ;HONOR PIR 1 DOES NOT GO LOW
002566  037063                  DH4
002570  037110                  DT4
002572  047047          ITE166: EM166                   ;PIR 6 WORKS BUT 4 & 1 DON'T
002574  037063                  DH4
002576  037110                  DT4
002600  047154          ITE167: EM167                   ;TMCA ABOVE BR7 MIGHT BE STUCK LOW
002602  037063                  DH4
002604  037110                  DT4
002606  047216          ITE170: EM170                   ;TMCE BRQ CLDCK MIGH BE STUCH LOW
002610  037063                  DH4
002612  037110                  DT4
002614  047260          ITE171: EM171                   ;BEN 13 FAILED TO PUP.00
002616  037063                  DH4
002620  037110                  DT4
002622  047401          ITE172: EM172                   ;BEN 13 FAILED TO SER.00
002624  037063                  DH4
```

```
002626  037110                  DT4
002630  042021       ITE173:    EM46            ;MFPT FAILED TO LOAD R0 CORRECTLY
002632  042054                  DH46
002634  042136                  DT46
002636  047654       ITE174:    EM174           ;PIR 2 DID NOT INTERRUPT
002640  037063                  DH4
002642  037110                  DT4
002644  047736       ITE175:    EM175           ;BEN 13 FAILED
002646  037063                  DH4
002650  037110                  DT4
002652  047753       ITE176:    EM176           ;LEVEL 1 INT. WHEN CPU LEVEL 1 ENABLED
002654  050145                  DH201           ;ERRORPC PIRQ     TEST NUMBER
002656  050176                  DT201           ;$ERRPC,$EPIRQ,$$TSTNM
002660  050015       ITE177:    EM177           ;PIR 3 DID NOT INTERRUPT
002662  037063                  DH4
002664  037110                  DT4
002666  047736       ITE200:    EM175           ;BEN 13 FAILED
002670  037063                  DH4
002672  037110                  DT4
002674  050077       ITE201:    EM201           ;LEVEL 2 WHEN CPU LEVEL 2 ENABLED
002676  050145                  DH201
002700  050176                  DT201
002702  050206       ITE202:    EM202           ;PIR 4 DID NOT INTERRUPT
002704  037063                  DH4
002706  037110                  DT4
002710  047736       ITE203:    EM175           ;BEN 13 FAILED
002712  037063                  DH4
002714  037110                  DT4
002716  050270       ITE204:    EM204           ;LEVEL 3 WHEN CPU LEVEL 3 ENABLED
002720  050145                  DH201
002722  050176                  DT201
002724  050336       ITE205:    EM205           ;PIR 5 DID NOT INTERRUPT
002726  037063                  DH4
002730  037110                  DT4
002732  047736       ITE206:    EM175           ;BEN 13 FAILED
002734  037063                  DH4
002736  037110                  DT4
002740  050421       ITE207:    EM207           ;LEVEL 4 INTERRUPT WHEN NOT SUPPOSE TO
002742  050145                  DH201
002744  050176                  DT201
002746  050467       ITE210:    EM210           ;FAILURE AFTER TMCB E70
002750  037063                  DH4
002752  037110                  DT4
002754  050535       ITE211:    EM211           ;FAILURE IN TMCB E70 OR BEFORE
002756  037063                  DH4
002760  037110                  DT4
002762  050616       ITE212:    EM212           ;BEN 13 FAILED
002764  037063                  DH4
002766  037110                  DT4
002770  050662       ITE213:    EM213           ;LEVEL 5 INTERRUPT WHEN NOT SUPPOSE TO
002772  050145                  DH201
002774  050176                  DT201
002776  050730       ITE214:    EM214           ;LEVEL 7 DID NOT INTERRUPT
003000  037063                  DH4
003002  037110                  DT4
003004  047736       ITE215:    EM175           ;BEN 13 FAILED
```

```
 999  003006  037063                 DH4
1000  003010  037110                 DT4
1001  003012  051012       ITE216:   EM216                   ;LEVEL 6 INTERRUPT WHEN NOT SUPPOSE TO
1002  003014  050145                 DH201
1003  003016  050176                 DT201
1004  003020  051060       ITE217:   EM217                   ;NO TIMEOUT ON DATI
1005  003022  037063                 DH4
1006  003024  037110                 DT4
1007  003026  051114       ITE220:   EM220                   ;BEN 13 FAILED
1008  003030  037063                 DH4
1009  003032  037110                 DT4
1010  003034  051175       ITE221:   EM221                   ;NO TIMEOUT ON DATO
1011  003036  037063                 DH4
1012  003040  037110                 DT4
1013  003042  051231       ITE222:   EM222                   ;BR 4 INTERRUPTS WHEN CPU AT LEVEL 7
1014  003044  037063                 DH4
1015  003046  037110                 DT4
1016  003050  051311       ITE223:   EM223                   ;BOTH BR 4 & BR 6 FAILED
1017  003052  037063                 DH4
1018  003054  037110                 DT4
1019  003056  051332       ITE224:   EM224                   ;BR 4 FAILED
1020  003060  037063                 DH4
1021  003062  037110                 DT4
1022  003064  051501       ITE225:   EM225                   ;BR 4 FAILED BUT BR 6 OK
1023  003066  037063                 DH4
1024  003070  037110                 DT4
1025  003072  051612       ITE226:   EM226                   ;BR 5 INTERRUPT FAILED
1026  003074  037063                 DH4
1027  003076  037110                 DT4
1028  003100  051672       ITE227:   EM227                   ;BR 6 INTERRUPT FAILED
1029  003102  037063                 DH4
1030  003104  037110                 DT4
1031  003106  051753       ITE230:   EM230                   ;YEL ZONE TRAP FAILED
1032  003110  037063                 DH4
1033  003112  037110                 DT4
1034  003114  000000       ITE231:   0                       ;DELETED
1035  003116  037063                 DH4
1036  003120  037110                 DT4
1037  003122  052240       ITE232:   EM232                   ;JSR WITH BAD STACK FAILED
1038  003124  037063                 DH4
1039  003126  037110                 DT4
1040  003130  052312       ITE233:   EM233                   ;STACK LIMIT REG DID NOT DISABLE YEL ZONE
1041  003132  037063                 DH4
1042  003134  037110                 DT4
1043  003136  052407       ITE234:   EM234                   ;TMCC KERNAL R6 DID NOT DISABLE YEL ZONE
1044  003140  037063                 DH4
1045  003142  037110                 DT4
1046  003144  052514       ITE235:   EM235                   ;RED ZONE REFERENCE FAILED
1047  003146  037063                 DH4
1048  003150  037110                 DT4
1049  003152  000000       ITE236:   0                       ;DELETED
1050  003154  037063                 DH4
1051  003156  037110                 DT4
1052  003160  052661       ITE237:   EM237                   ;BEN 13 FAILED TO PUP.00
1053  003162  037063                 DH4
1054  003164  037110                 DT4
```

```
003166  052754          ITE240: EM240           ;BEN 13 FAILED TO BRK.80
003170  037063                  DH4
003172  037110                  DT4
003174  053056          ITE241: EM241           ;NO RED ZONE ON STACK OVERFLOW
003176  037063                  DH4
003200  037110                  DT4
003202  053167          ITE242: EM242           ;NO RED ZONE WHEN SL REG>BADDR
003204  037063                  DH4
003206  037110                  DT4
003210  053323          ITE243: EM243           ;52400 PATTERN FAILED IN SL REG
003212  053373                  DH243           ;ERRORPC      SL REG      TEST NUMBER
                                                ;           EXPECT  ACTUAL
003214  053456                  DT243           ;$ERRPC,$TMP0,E2STKLMT,$$TSTNM
003216  053470          ITE244: EM244           ;125000 PATTERN FAILED IN SL REG
003220  053373                  DH243
003222  053540                  DT244           ;$ERRPC,$TMP0,E1STKLMT,$$TSTNM
003224  053552          ITE245: EM245           ;BR SELECTED INSTEAD OF SL
003226  037063                  DH4
003230  037110                  DT4
003232  053645          ITE246: EM246           ;SL AND PB BOTH SELECTED
003234  037063                  DH4
003236  037110                  DT4
003240  053766          ITE247: EM247           ;PSW SELECTED INSTEAD OF SL
003242  037063                  DH4
003244  037110                  DT4
003246  054077          ITE250: EM250           ;WHAT HAPPENED?
003250  054124                  DH250           ;BOTH PATTERNS FAILED
003252  054242                  DT250
003254  054260          ITE251: EM251           ;YEL ZONE IN RED REGION (SP=330)
003256  037063                  DH4
003260  037110                  DT4
003262  054425          ITE252: EM252           ;YEL ZONE IN RED REGION (SP=240)
003264  037063                  DH4
003266  037110                  DT4
003270  054475          ITE253: EM253           ;YEL ZONE IN RED REGION (SP=140)
003272  037063                  DH4
003274  037110                  DT4
003276  054545          ITE254: EM254           ;RED ZONE IN YELLOW REGION (SP=376)
003300  037063                  DH4
003302  037110                  DT4
003304  054635          ITE255: EM255           ;CPU ERR REG BIT 4 DOES NOT SET
003306  054714                  DH255
003310  041414                  DT41
003312  054775          ITE256: EM256           ;CPU ERR REG DOES NOT CLEAR
003314  037063                  DH4
003316  037110                  DT4
003320  055032          ITE257: EM257           ;CPU ERROR BIT 3 DOES NOT SET
003322  054714                  DH255
003324  041414                  DT41
003326  055103          ITE260: EM260           ;CPU ERROR BIT 3 DOES NOT CLEAR
003330  037063                  DH4
003332  037110                  DT4
003334  055146          ITE261: EM261           ;CPU ERROR BIT 2 DOES NOT SET
003336  054714                  DH255
003340  041414                  DT41
003342  055220          ITE262: EM262           ;CPU ERROR BIT 2 DOES NOT CLEAR
```

```
003344  037063                    DH4
003346  037110                    DT4
003350  055262          ITE263:  EM263
003352  055451                    DH263
003354  000000                    0
003356  056074          ITE264:  EM264                     ;GOING TO NEXT TEST
003360  000000                    0
003362  000000                    0
003364  056117          ITE265:  EM265                     ;NEITHER - BYIN NOR DATI CAUSED ODD ADDR.
003366  037063                    DH4
003370  037110                    DT4
003372  056435          ITE266:  EM266                     ;DATI TRAPPED BUT - BYIN DIDN'T
003374  037063                    DH4
003376  037110                    DT4
003400  056553          ITE267:  EM267                     ;BOTH DATI & DATO FAILED
003402  037063                    DH4
003404  037110                    DT4
003406  056633          ITE270:  EM270                     ;DATO WORKS BUT DATI FAILED
003410  037063                    DH4
003412  037110                    DT4
003414  056731          ITE271:  EM271                     ;NO TRAP ON DATO
003416  037063                    DH4
003420  037110                    DT4
003422  057061          ITE272:  EM272                     ;NO TRAP ON SM357*SRC1 DATI
003424  037063                    DH4
003426  037110                    DT4
003430  057120          ITE273:  EM273                     ;ODD ADDR BIT IN CPU ERROR FAILED
003432  037063                    DH4
003434  037110                    DT4
003436  057166          ITE274:  EM274                     ;NO TRAP ON DATO
003440  037063                    DH4
003442  037110                    DT4
003444  057206          ITE275:  EM275                     ;T BIT TRAP FAILED
003446  037063                    DH4
003450  037110                    DT4
003452  057556          ITE276:  EM276                     ;T BIT NEVER SET
003454  037063                    DH4
003456  037110                    DT4
003460  057607          ITE277:  EM277                     ;PS<08> DID NOT DISABLE T TRAP (KB-11E/EM ONLY)
003462  037063                    DH4
003464  037110                    DT4
003466  057663          ITE300:  EM300                     ;TRAP VECTOR DECODE FAILED
003470  037063                    DH4
003472  037110                    DT4
003474  057752          ITE301:  EM301                     ;RTT DID NOT DISABLE T BIT
003476  037063                    DH4
003500  037110                    DT4
003502  060037          ITE302:  EM302                     ;PIR 1 DID NOT DISABLE ON BR4
003504  037063                    DH4
003506  037110                    DT4
003510  060104          ITE303:  EM303                     ;PIR 1 CAME THRU ON YELLOW ZONE
003512  037063                    DH4
003514  037110                    DT4
003516  060235          ITE304:  EM304                     ;PIR 2 CAME THRU ON YELLOW ZONE
003520  037063                    DH4
003522  037110                    DT4
```

```
 003524  060322          ITE305: EM305                 ;PIR 3 CAME THRU ON YELLOW ZONE
 003526  037063                  DH4
 003530  037110                  DT4
 003532  060410          ITE306: EM306                 ;BR4 CAME THRU ON PIR 5
 003534  037063                  DH4
 003536  037110                  DT4
 003540  060532          ITE307: EM307                 ;BR4 CAME THRU ON PIR 5
 003542  037063                  DH4
 003544  037110                  DT4
 003546  060605          ITE310: EM310                 ;BR4 CAME THRU ON BR 5
 003550  037063                  DH4
 003552  037110                  DT4
 003554  060657          ITE311: EM311                 ;BR4 CAME IN ON PIR 6
 003556  037063                  DH4
 003560  037110                  DT4
 003562  060730          ITE312: EM312                 ;BR4 CAME IN ON PIR 7
 003564  037063                  DH4
 003566  037110                  DT4
 003570  061054          ITE313: EM313                 ;PIR 4 CAME IN ON BR5
 003572  037063                  DH4
 003574  037110                  DT4
 003576  061124          ITE314: EM314                 ;PIR 4 CAME IN ON BR6
 003600  037063                  DH4
 003602  037110                  DT4
 003604  061235          ITE315: EM315                 ;PIR 4 CAME THRU ON SL YELLOW
 003606  037063                  DH4
 003610  037110                  DT4
 003612  061357          ITE316: EM316                 ;BR5 CAME IN ON PIR 5
 003614  037063                  DH4
 003616  037110                  DT4
 003620  061472          ITE317: EM317                 ;BR5 CAME IN ON PIR 6
 003622  037063                  DH4
 003624  037110                  DT4
 003626  061512          ITE320: EM320                 ;BR5 CAME IN ON PIR 7
 003630  037063                  DH4
 003632  037110                  DT4
 003634  061630          ITE321: EM321                 ;PIR 5 CAME IN ON BR6
 003636  037063                  DH4
 003640  037110                  DT4
 003642  061700          ITE322: EM322                 ;PIR 5 CAME IN ON SL YELLOW
 003644  037063                  DH4
 003646  037110                  DT4
 003650  061755          ITE323: EM323                 ;BR6 CAME IN ON PIR 6
 003652  037063                  DH4
 003654  037110                  DT4
 003656  062065          ITE324: EM324                 ;BR6 CAME IN ON PIR 7
 003660  037063                  DH4
 003662  037110                  DT4
 003664  062146          ITE325: EM325                 ;PIR 6 CAME IN ON SL YELLOW
 003666  037063                  DH4
 003670  037110                  DT4
 003672  062223          ITE326: EM326                 ;PIR 7 CAME IN ON SL YELLOW
 003674  037063                  DH4
 003676  037110                  DT4
 003700  062304          ITE327: EM327                 ;PSW BIT 11 DID NOT SET
 003702  037063                  DH4
```

```
003704  037110          DT4
003706  062447  ITE330: EM330           ;R12 CLEARED R2
003710  037063          DH4
003712  037110          DT4
003714  062560  ITE331: EM331           ;R2 SRC WAS AFFECTED BY CLR R12
003716  037063          DH4
003720  037110          DT4
003722  062673  ITE332: EM332           ;R2 DST WAS AFFECTED BY (R12)+
003724  037063          DH4
003726  037110          DT4
003730  062754  ITE333: EM333           ;R2 SRC WAS AFFECTED BY (R12)+
003732  037063          DH4
003734  037110          DT4
003736  063007  ITE334: EM334           ;R15 DST AFTER UPAD 2
003740  037063          DH4
003742  037110          DT4
003744  063042  ITE335: EM335           ;R15 SRC DOES NOT SELECT ON UPAD 2
003746  037063          DH4
003750  037110          DT4
003752  063077  ITE336: EM336           ;PDRD PS11 DOES NOT GET TO GRAC
003754  037063          DH4
003756  037110          DT4
003760  063146  ITE337: EM337           ;BAD BITS IN GPR SET 1 SRC
003762  063213          DH337           ;ERRORPC PATTERN TESTNUMBER
003764  063250          DT337           ;$ERRPC,$TMP1,$$TSTNM
003766  063260  ITE340: EM340           ;BAD BITS IN GPR SET 1 DST
003770  063213          DH337
003772  063250          DT337
003774  063332  ITE341: EM341           ;GRAB DST SET 1 DOES NOT GO LOW ON R14
003776  037063          DH4
004000  037110          DT4
004002  063377  ITE342: EM342           ;GRAB SRC SET 1 DOES NOT GO LOW ON R14
004004  037063          DH4
004006  037110          DT4
004010  063444  ITE343: EM343           ;50000 PATTERN FAILED IN PSW
004012  041456          DH42
004014  041534          DT42
004016  063500  ITE344: EM344           ;164000 PATTERN FAILED IN PSW
004020  041456          DH42
004022  041534          DT42
004024  063535  IRE345: EM345           ;PSW HIGH BYTE DID NOT CLEAR
004026  041456          DH42
004030  041534          DT42
004032  063570  ITE346: EM346           ;SUPER SP DOES NOT SELECT ON UPAD 5
004034  037063          DH4
004036  037110          DT4
004040  063650  ITE347: EM347           ;SUPER SP DOES NOT SELECT ON UPAD 0
004042  037063          DH4
004044  037110          DT4
004046  063730  ITE350: EM350           ;USER SP DOES NOT SELECT ON UPAD 5
004050  037063          DH4
004052  037110          DT4
004054  064005  ITE351: EM351           ;USER SP DOES NOT SELECT ON UPAD 0
004056  037063          DH4
004060  037110          DT4
004062  064062  ITE352: EM352           ;EITHER USER OR SUPER SP DST FAILED
```

```
004064  063213                  DH337
004066  063250                  DT337
004070  064127       ITE353: EM353                  ;EITHER USER OR SUPER SP SRC FAILED
004072  063213                  DH337
004074  063250                  DT337
004076  064174       ITE354: EM354                  ;KSP SRC CHANGED ON MTP
004100  037063                  DH4
004102  037110                  DT4
004104  064226       ITE355: EM355                  ;KSP SRC & DST CHANGED ON MTP
004106  037063                  DH4
004110  037110                  DT4
004112  064305       ITE356: EM356                  ;KSP DST CHANGED ON MTP
004114  037063                  DH4
004116  037110                  DT4
004120  064340       ITE357: EM357                  ;SSP DID NOT LOAD PROPERLY ON MTPD
004122  064431                  DH357                  ;ERRORPC      SSP          TEST NUMBER
                                                       ;          EXPECT  ACTUAL
004124  037526                  DT14
004126  064503       ITE360: EM360                  ;USER SP DID NOT LOAD ON MTP
004130  037063                  DH4
004132  037110                  DT4
004134  064562       ITE361: EM361                  ;BAD FIELD IN IR DECODE ROM
004136  037063                  DH4
004140  037110                  DT4
004142  064632       ITE362: EM362                  ;SSP WAS READ AND USP WAS WRITTEN
004144  037063                  DH4
004146  037110                  DT4
004150  064667       ITE363: EM363                  ;SSP WAS READ AND WRITTEN
004152  037063                  DH4
004154  037110                  DT4
004156  064737       ITE364: EM364                  ;CAN'T DETERMINE WHAT HAPPENED
004160  037063                  DH4                    ;SSP WAS READ BUT THE WRITE FAILED
004162  037110                  DT4
004164  065021       ITE365: EM365                  ;USP WAS READ BUT SSP WAS WRITTEN
004166  037063                  DH4
004170  037110                  DT4
004172  065056       ITE366: EM366                  ;USP WAS READ BUT REG 7 WAS WRITTEN
004174  037063                  DH4
004176  037110                  DT4
004200  065112       ITE367: EM367                  ;SPL WORKED IN SUPERVISOR MODE
004202  041456                  DH42
004204  041534                  DT42
004206  065226       ITE370: EM370                  ;BIT FAILED TO SET IN PSW
004210  041456                  DH42
004212  041534                  DT42
004214  065353       ITE371: EM371                  ;BIT FAILED TO CLEAR IN PSW
004216  041456                  DH42
004220  041534                  DT42
004222  065460       ITE372: EM372                  ;BITS <13:11] DID NOT PRESET
004224  041456                  DH42
004226  041534                  DT42
004230  065610       ITE373: EM373                  ;IOT DID NOT CHANGE PSW CORRECTLY
004232  041456                  DH42
004234  041534                  DT42
004236  065722       ITE374: EM374                  ;PREVIOUS MODE BITS DID NOT CLEAR
004240  041456                  DH42
```

```
004242  041534                    DT42
004244  066012          ITE375:   EM375           ;NO KT ABORT
004246  037063                    DH4
004250  037110                    DT4
004252  066165          ITE376:   EM376           ;PS<08> FAILED TO SET
004254  037063                    DH4
004256  037110                    DT4
004260  000000          ITE377:   0
004262  000000                    0
004264  000000                    0
004266  066215          ITE400:   EM400           ;KT ABORT TRAPPED TO 4
004270  037063                    DH4
004272  037110                    DT4
004274  066316          ITE401:   EM401           ;KT ABORT TRAPPED TO 10
004276  037063                    DH4
004300  037110                    DT4
004302  066363          ITE402:   EM402           ;KT ABORT TRAPPED TO 240
004304  037063                    DH4
004306  037110                    DT4
004310  066425          ITE403:   EM403           ;KT TRAP DID NOT WORK
004312  037063                    DH4
004314  037110                    DT4
004316  000000          ITE404:   0               ;DELETED
004320  037063                    DH4
004322  037110                    DT4
004324  066531          ITE405:   EM405           ;NO KT TRAP ON SOURCE OPERAND
004326  037063                    DH4
004330  037110                    DT4
004332  066647          ITE406:   EM406           ;NO ABORT ON NEXM
004334  037063                    DH4
004336  037110                    DT4
004340  000000          ITE407:   0               ;DELETED
004342  037063                    DH4
004344  037110                    DT4
004346  066777          ITE410:   EM410           ;NEXM BIT DID NOT SET IN CPUERR REG
004350  054714                    DH255
004352  041414                    DT41
004354  067102          ITE411:   EM411           ;NEXM BIT DID NOT CLEAR IN CPUERR REG
004356  037063                    DH4
004360  037110                    DT4
004362  067151          ITE412:   EM412           ;KT ABORT ON NEXM (KB11-B/C)
004364  037063                    DH4
004366  037110                    DT4
004370  067226          ITE413:   EM413           ;KT ABORT ON SL RED
004372  037063                    DH4
004374  037110                    DT4
004376  067301          ITE414:   EM414           ;KT ABORT ON ODD ADDRESS
004400  037063                    DH4
004402  037110                    DT4
004404  067362          ITE415:   EM415           ;KT ABORT FAILED TO OVER-RIDE NEXM (KB11-E/EM)
004406  037063                    DH4
004410  037110                    DT4
004412  067445          ITE416:   EM416           ;TMCE CACHE BEND DID NOT GO
004414  037063                    DH4             ;HIGH ON KT ABORT
004416  037110                    DT4
004420  067520          ITE417:   EM417           ;ILLEGAL HALT DID NOT TRAP
```

```
 1391  004422  037063                       DH4
 1392  004424  037110                       DT4
 1393  004426  067611          ITE420: EM420                        ;CPU ERROR REG BIT 5 DID NOT SET
 1394  004430  054714                       DH255
 1395  004432  041414                       DT41
 1396  004434  067666          ITE421: EM4_1                        ;BEN 6 FAILED ON PS RESTORE
 1397  004436  037063                       DH4
 1398  004440  037110                       DT4
 1399  004442  067757          ITE422: EM422                        ;NO PE ABORT
 1400  004444  037063                       DH4
 1401  004446  037110                       DT4
 1402  004450  070167          ITE423: EM423                        ;PE ABORTED TO 4
 1403  004452  037063                       DH4
 1404  004454  037110                       DT4
 1405  004456  070261          ITE424: EM424                        ;PE ABORTED TO 14
 1406  004460  037063                       DH4
 1407  004462  037110                       DT4
 1408  004464  070340          ITE425: EM425                        ;PE ABORTED TO 104
 1409  004466  037063                       DH4
 1410  004470  037110                       DT4
 1411  004472  070357          ITE426: EM426                        ;NO PE TRAP
 1412  004474  037063                       DH4
 1413  004476  037110                       DT4
 1414  004500  070443          ITE427: EM427                        ;PE TRAP, TRAPPED TO
 1415  004502  037063                       DH4                      ;WRONG VECTOR
 1416  004504  037110                       DT4
 1417  004506  070547          ITE430: EM430                        ;PIR 6 CAME IN ON MEM MGT TRAP
 1418  004510  037063                       DH4
 1419  004512  037110                       DT4
 1420  004514  070612          ITE431: EM431                        ;PIR 3 CAME IN ON MEM MGT TRAP
 1421  004516  037063                       DH4
 1422  004520  037110                       DT4
 1423  004522  070656          ITE432: EM432                        ;YEL ZONE CAME IN ON PE TRAP
 1424  004524  037063                       DH4
 1425  004526  037110                       DT4
 1426  004530  070742          ITE433: EM433                        ;MEM MGT TRAP CAME IN ON PE
 1427  004532  037063                       DH4
 1428  004534  037110                       DT4
 1429  004536  000000          ITE434: 0                            ;DELETED
 1430  004540  037063                       DH4
 1431  004542  037110                       DT4
 1432  004544  071010          ITE435: EM435                        ;TMCC PRIORITY CLEAR DID NOT WORK
 1433  004546  037063                       DH4
 1434  004550  037110                       DT4
 1435  004552  071130          ITE436: EM436                        ;UNIBUS PE ABORT DID NOT HAPPEN
 1436  004554  037063                       DH4
 1437  004556  037110                       DT4
 1438  004560  071173          ITE437: EM437                        ;UNIBUS PE TRAPPED TO 0
 1439  004562  037063                       DH4
 1440  004564  037110                       DT4
```

```
004566  071243                          ITE440: EM440                   ;WAIT INSTRUCTION FAILED
004570  037063                                  DH4
004572  037110                                  DT4
004574  071316                          ITE441: EM441                   ;T BIT INTERRUPTED WAIT
004576  037063                                  DH4
004600  037110                                  DT4
004602  071367                          ITE442: EM442                   ;PIRQ DID NOT INTERRUPT WAIT
004604  037063                                  DH4
004606  037110                                  DT4
004610  071444                          ITE443: EM443                   ;STATE S13.00 FAILED
004612  037063                                  DH4
004614  037110                                  DT4
004616  071462                          ITE444: EM444                   ;STATE S45.00 FAILED
004620  037063                                  DH4
004622  037110                                  DT4
004624  071500                          ITE445: EM445                   ;STATE MTP.10 FAILED
004626  037063                                  DH4
004630  037110                                  DT4
004632  071516                          ITE446: EM446                   ;STATE NEG.00 FAILED
004634  037063                                  DH4
004636  037110                                  DT4
004640  071534                          ITE447: EM447                   ;STATE D45.00 FAILED
004642  037063                                  DH4
004644  037110                                  DT4
004646  071552                          ITE450: EM450                   ;STATE D45.90 FAILED
004650  037063                                  DH4
004652  037110                                  DT4
004654  071570                          ITE451: EM451                   ;STATE D45.00 FAILED
004656  037063                                  DH4
004660  037110                                  DT4
004662  071606                          ITE452: EM452                   ;STATE D45.01 FAILED
004664  037063                                  DH4
004666  037110                                  DT4
004670  071624                          ITE453: EM453                   ;RESERVED INSTRUCTION TRAP FAILED
004672  037063                                  DH4
004674  037110                                  DT4
004676  071672                          ITE454: EM454                   ;TMCB PIRQ DOES NOT GO LOW
004700  037063                                  DH4
004702  037110                                  DT4
004704  071757                          ITE455: EM455                   ;BEN06 FAILED
004706  037063                                  DH4
004710  037110                                  DT4
004712  072075                          ITE456: EM456                   ;ODD ADDRESS FAILED ON DATI & DATO
004714  037063                                  DH4
004716  037110                                  DT4
004720  072151                          ITE457: EM457                   ;PSW BIT 11 DOES NOT CLEAR
004722  037063                                  DH4
004724  037110                                  DT4
004726  072202                          ITE460: EM460                   ;PSW CHANGED ON RESET
004730  041456                                  DH42
004732  041534                                  DT42
004734  072246                          ITE461: EM461                   ;PSW CHANGES ON RESET IN SUPER MODE
004736  041456                                  DH42
004740  041534                                  DT42
004742  012737  000014  177746          START:  MOV     #14,@#CONTRL    ;FORCE MISSES IN CACHE
004750  005037  170200                          CLR     @#MAPL0         ;SETUP MAP 0 AND 1 TO PHISICAL CORE
```

```
004754  005037  170202                      CLR     @#MAPH0
004760  012737  020000  170204              MOV     #20000,@#MAPL1
004766  005037  170206                      CLR     @#MAPH1
004772  012737  000340  177776              MOV     #340,@#PS          ;;LOCK OUT ALL INTERRUPTS
005000  012706  001100                      MOV     #$CMTAG,R6         ;;FIRST LOCATION TO BE CLEARED
005004  005026                              CLR     (R6)+              ;;CLEAR MEMORY LOCATION
005006  022706  001136                      CMP     #$TKS,R6           ;;DONE?
005012  001374                              BNE     .-6                ;;LOOP BACK IF NO
005014  012706  001100                      MOV     #STACK,SP          ;;SETUP THE STACK POINTER
005020  012737  033270  000020              MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
005026  012737  000340  000022              MOV     #340,@#IOTVEC+2 ;;LEVEL 7
005034  012737  033536  000030              MOV     #$ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
005042  012737  000340  000032              MOV     #340,@#EMTVEC+2 ;;LEVEL 7
005050  012737  036234  000034              MOV     #$TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
005056  012737  000340  000036              MOV2 ;;LEVEL 7
005140  012767  000002  025472              MOV     #RTI,$RTRN         ;;SET $RTRN TO A RTI
005146  012737  005174  000010              MOV     #65$,@#RESVEC      ;;TRY TO DO A RTT
005154  005046                              CLR     -(SP)              ;;DUMMY PS
005156  012746  005164                      MOV     #64$,-(SP)         ;;AND PC
005162  000006                              RTT                        ;;TRY THE RTT
005164  012767  000006  025446      64$:    MOV     #RTT,$RTRN         ;;RTT IS LEGAL--SET $RTRN TO A RTT
005172  000402                              BR      66$
005174  062706  000010              65$:    ADD     #10,SP             ;;RTT ILLEGAL--CLEAN OFF THE STACK
005200  012737  000012  000010      66$:    MOV     #RESVEC+2,@#RESVEC ;;RESTORE TRAP CATCHER
005206  005067  025434                      CLR     $TBIT              ;;CLEAR ''T'' BIT SWITCH
005212  012767  005212  173666              MOV     #.,$LPADR          ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
005220  012767  005220  173662              MOV     #.,$LPERR          ;;SETUP THE ERROR LOOP ADDRESS


                                    ;;*******************************************************************
005226  005227  177777                      INC     #-1                ;;FIRST TIME?
005232  001037                              BNE     67$                ;;BRANCH IF NO
005234  022737  032572  000042              CMP     #$ENDAD,@#42       ;;ACT-11?
005242  001433                              BEQ     67$                ;;BRANCH IF YES
005244  104400  005252                      TYPE    ,68$               ;;TYPE ASCIZ STRING
005250  000430                              BR      67$                ;;GET OVER THE ASCIZ
                                    ;;68$:  .ASCIZ  <CRLF>''CEKBB-D  PDP11/70-74MP CPU DIAGNOSTIC PART 2''<CRLF>
005332                              67$:
005332  005227  177777                      INC     #-1                ;FIRST TIME?
005336  001000                              BNE     100$               ;BR IF NO
005340                              100$:
                                    ;;*******************************************************************
                                    ;SAVE THE MONITOR IF INITIAL LOAD
005340  005737  000042              LOOP:   TST     @#42               ;RUNNING UNDER  XXDP CHAIN?
005344  001003                              BNE     3$                 ;BRANCH IF YES
005346  005227  177777                      INC     #-1                ;INITIAL LOAD?
005352  001010                              BNE     2$                 ;BRANCH IF NO
005354  012700  002734              3$:     MOV     #^D1500,R0         ;SETUP SOB COUNT
005360  012701  073300                      MOV     #SUBTAB+2,R1       ;GET END ADDRESS OF PROGRAM
```

```
005364  012702  160000                   MOV     #160000,R2       ;GET END ADDRESS OF MONITOR
005370  014221                   1$:     MOV     -(R2),(R1)+      ;SAVE 1500 DECIMAL WORDS
005372  077002                           SOB     R0,1$
005374  012737  034644  000060   2$:     MOV     #QUIT,@#TKVEC    ;SETUP KEYBOARD VECTOR
005402  012737  000340  000062           MOV     #PR7,@#TKVEC+2   ;SETUP KEYBOARD PSW
005410  005077  173524                   CLR     @$TKB            ;ENSURE BUFFER CLEAR
005414  152777  000100  173514           BISB    #BIT6,@$TKS      ;SET INTERRUPT ENABLE BIT
005422  012737  032654  000004           MOV     #CPUSPUR,@#ERRVEC
005430  012737  000340  000006           MOV     #PR7,@#ERRVEC+2
005436  012737  033060  000114           MOV     #CACHSPU,@#CACHVEC
005444  012737  000340  000116           MOV     #PR7,@#CACHVEC+2
005452  005037  177766                   CLR     @#CPUERR         ;ENSURE ERROR REGISTER CLEAR
005456  012737  177777  177744           MOV     #-1,@#MEMERR     ;ENSURE MEMORY ERROR REG CLEAR
005464  005767  173410                   TST     $PASS            ;FIRST PASS?
005470  001402                           BEQ     TST1             ;;BRANCH IF YES
005472  005037  177746                   CLR     @#CONTRL         ;ENABLE CACHE

                                 ;;*****************************************************************
                                 ;*TEST 1          SPL
                                 ;*
                                 ;*      IF FORK A FAILS EXECUTION WILL GO TO ONE OF 3 STATES:
                                 ;*      RSD.02,SVC.70, OR D30.00.
                                 ;*      RSD.02 WILL CAUSE A TRAP TO LOCATION 10.
                                 ;*      SVC.70 WILL HANG THE PROCESSOR IN THE PAUSE STATE.  THIS
                                 ;*      WILL ONLY OCCUR IF RACF E17(AFIR04(1)*AFIR05(0)*(RTS:CCOP))
                                 ;*      IS BAD.
                                 ;*      D30.00 WILL CAUSE A TRAP TO LOCATION 10 AFTER STATE D10.60.
                                 ;*      THIS FAILURE CAN BE DIFFERENTIATED FROM THE FIRST BY TESTING
                                 ;*      THE REGISTER ASSOCIATED WITH BITS <2:0> OF THE OP CODE TO
                                 ;*      SEE IF IT WAS INCREMENTED.
                                 ;*
                                 ;*      IF BOTH LEVELS 2 AND 5 COME UP AS LEVEL 4, PDRD E31(1)
                                 ;*      COULD BE BAD.
                                 ;*
                                 ;*      ONCE IT IS DETERMINED THAT THE INSTRUCTION WORKS A BIT TEST IS
                                 ;*      MADE ON THE PSW <7:5>.
                                 ;*
                                 ;*      ROM FLOW-43,361
                                 ;;*****************************************************************
005476  000004                   TST1:   SCOPE
005500  012767  005660  173466           MOV     #TST2,$ESCAPE    ;SAVE START ADDRESS OF NEXT TEST
005506  012767  005660  173554           MOV     #TST2,NEXTTST    ;SAVE START ADDRESS OF NEXT TEST
005514  012706  001100                   MOV     #STACK,SP        ;INITIALIZE THE SP
005520  005005                           CLR     R5               ;INITIALIZE ERROR RECORD
005522  012737  005642  000010           MOV     #1$,@#RESVEC     ;SETUP RESERVED VECTOR TO TRAP TO THIS ROUTINE
005530  012737  000113  177770           MOV     #113,@#177770    ;SETUP MICROPROCESSOR BREAK REG
005536  000235                           SPL     5                ;EXECUTE INSTRUCTION UNDER TEST
005540  013767  177776  173440           MOV     @#PSW,$ERPSW     ;SAVE PSW
005546  042767  177437  173432           BIC     #177437,$ERPSW   ;MASK OFF THE PRIORITY
005554  022767  000240  173424           CMP     #PR5,$ERPSW      ;DID PRIORITY LEVEL 5 GET SET?
005562  001401                           BEQ     2$               ;BRANCH IF YES
005564  005205                           INC     R5               ;SET ERROR RECORD
005566  012737  000044  177770   2$:     MOV     #44,@#177770     ;SETUP MICROPROCESSOR BREAK REG
005574  000240                           NOP                      ;SYNC INSTRUCTION
005576  000232                           SPL     2                ;TEST TO SEE IF BITS 5&7 CLEAR & BIT 6 SETS
005600  013767  177776  173354           MOV     @#PSW,$TMP0      ;SAVE PSW
```

```
005606  042767  177437  173346          BIC     #177437,$TMPO       ;MASK OFF THE PRIORITY
005614  022767  000100  173340          CMP     #PR2,$TMPO          ;DID BIT 6 SET & 5&7 CLEAR?
005622  001404                          BEQ     3$                  ;BRANCH IF YES
005624  005705                          TST     R5                  ;DID SPL 5 FAIL?
005626  001401                          BEQ     4$                  ;BRANCH IF NO
005630  104001                          ERROR   1                   ;EITHER SPL DOES NOT LOAD PSW OR BITS STUCK
005632  104002                  4$:     ERROR   2                   ;PR5 OK BUT PR2 BAD
005634  005705                  3$:     TST     R5                  ;DID SPL 5 FAIL?
005636  001410                          BEQ     TST2                ;;BRANCH IF NO
005640  104003                          ERROR   3                   ;PR2 OK BUT PR5 FAILED
005642  012737  000012  000010  1$:     MOV     #12,@#RESVEC        ;RESTORE RESERVEC VECTOR ADDRESS
005650  005705                          TST     R5                  ;DID R5 GET INCREMENTED?
005652  001401                          BEQ     5$                  ;BRANCH OF NO
005654  104004                          ERROR   4                   ;FORK A FAILED TO D30.00
005656  104005                  5$:     ERROR   5                   ;FORK A FAILED TO RSD.02
                                ;;*******************************************************************
                                ;*TEST 2                RESET
                                ;*
                                ;*      IF FORK A FAILS EXECUTION WILL EITHER GO TO WAT.00 OR TRP.02.
                                ;*      THE LA30 PRINTER WILL BE STARTED SUCH THAT A FAILURE INTO
                                ;*      WAT.00 WILL RECOVER.
                                ;*
                                ;*      IF TRP.01 IS ENTERED A TRAP SEQUENCE WILL EXECUTE
                                ;*      WITH A TRAP VECTOR OF 4.
                                ;*
                                ;*      ROM FLOW-15,255,374
                                ;;*******************************************************************
005660  000004                  TST2:   SCOPE
005662  012767  003674  173214          MOV     #^D1980,$ICNT       ;SET ITTERATION COUNT
005670  012767  006154  173276          MOV     #TST3,$ESCAPE       ;SAVE START ADDRESS OF NEXT TEST
005676  012767  006154  173364          MOV     #TST3,NEXTTST       ;SAVE START ADDRESS OF NEXT TEST
005704  013767  177776  173256          MOV     @#PSW,$TMP3         ;SAVE PSW
005712  012767  005742  173166          MOV     #4$,$LPADR          ;SETUP LOOP ADR
005720  012767  005742  173162          MOV     #4$,$LPERR          ;SETUP ERROR LOOP
005726  012737  006144  000064          MOV     #2$,@#TPVEC         ;PUT ADDRESS OF 2$ IN PRINTER INTERRUPT VEC
005734  012737  006134  000004          MOV     #1$,@#ERRVEC        ;PUT ADDRESS OF 1$ IN LOCATION 4
005742  012706  001100          4$:     MOV     #STACK,SP           ;INITIALIZE THE SP
                                                                    ;TO CATCH A FAILURE TO THE TRP.02 STATE
005746  000230                          SPL     0                   ;SET CPU AT 0
005750  012777  000015  173166          MOV     #15,@$TPB           ;SEND A CR TO THE PRINTER
005756  105777  173160          7$:     TSTB    @$TPS               ;WAIT FOR CR TO FINISH
005762  100375                          BPL     7$                  ;INCASE TP DOUBLE BUFFERED
005764  012777  000101  173152          MOV     #101,@$TPB          ;SEND AN "A"
005772  052777  000100  173142          BIS     #BIT6,@$TPS         ;SET THE INTERRUPT FLAG
006000  012737  034157  177776          MOV     #34157,@#PSW        ;SET AS MANY BITS AS POSSIBLE IN PSW
006006  000240                          NOP                         ;SYNC POINT
006010  000005                          RESET                       ;EXECUTE INSTRUCTION UNDER TEST
006012  013767  177776  173166          MOV     @#PSW,$ERPSW        ;SAVE PSW
006020  017700  173116                  MOV     @$TPS,R0            ;GET THE PRINTER STATUS
006024  042777  000100  173112          BIC     #BIT6,@$TPB         ;CLEAR INTERRUPT FLAG INCASE RESET FAILED
006032  032700  000100                  BIT     #BIT6,R0            ;DID INTERRUPT FLAG CLEAR?
006036  001401                          BEQ     3$                  ;BRANCH IF YES
006040  104006                          ERROR   6                   ;RESET DID NOT WORK
006042  105777  173074          3$:     TSTB    @$TPS               ;WAIT FOR CHARACTER TO FINISH
006046  100375                          BPL     3$
006050  012737  032654  000004          MOV     #CPUSPUR,@#ERRVEC           ;RESTORE ERRVEC
```

```
006056  012767  034157  173076          MOV     #34157,$TMPO     ;SAVE EXPECTED VALUE
006064  032737  000020  177776          BIT     #BIT4,@#PSW      ;IS T BIT ON?
006072  001407                          BEQ     5$               ;BRANCH IF NO
006074  012767  034177  173060          MOV     #34177,$TMPO     ;SAVE EXPECTED VALUE
006102  022767  034177  173076          CMP     #34177,$ERPSW    ;DID PSW COME OUT OK?
006110  000403                          BR      6$
006112  022767  034157  173066  5$:     CMP     #34157,$ERPSW    ;DID PSW CHANGE?
006120                          6$:
006120  001415                          BEQ     TST3             ;;BRANCH IF NO
006122  012767  034157  173032          MOV     #34157,$TMPO     ;SAVE EXPECTED VALUE
006130  104377                          ERROR   377              ;PSW CHANGED ON RESET
006132  000460                                  460
006134  042777  000100  173000  1$:     BIC     #BIT6,@$TPS      ;CLEAR PRINTER INTERRUPT FLAG
006142  104007                          ERROR   7                ;FORK A FAILED INTO TRP.02
006144  042777  000100  172770  2$:     BIC     #BIT6,@$TPS      ;CLEAR PRINTER INTERRUPT FLAG
006152  104010                          ERROR   10               ;FORK A FAILED TO WAT.00

                                ;;*****************************************************************
                                ;*TEST 3         MARK
                                ;*
                                ;*       FORK A CAN FAIL INTO ONE OF THE FOLLOWING:
                                ;*       RSD.00, MFP.80, MTP.00, SVC.70, OR D67.01.
                                ;*       STATE RSD.00 WILL CAUSE A TRAP TO LOCATION 10.
                                ;*       MFP.80 WILL EXECUTE AN MFP INSTRUCTION.
                                ;*       MTP.00 WILL EXECUTE AN MTP INSTRUCTION EXCEPT FOR THE ALU.
                                ;*       SVC.70 WILL HANG THE PROCESSOR IN THE PAUSE CONDITION.
                                ;*       THIS WILL ONLY HAPPEN IF RACF E8 IS BAD.
                                ;*       D67.01 WILL STEP THE PCB AND TRAP TO LOCATION 10 AFTER STATE D10.60.
                                ;*
                                ;*       ROM FLOW-47,252,235,234
                                ;;*****************************************************************
006154  000004                  TST3:   SCOPE
006156  152777  000100  172752          BISB    #BIT6,@$TKS      ;RESTORE INTER FLAG AFTER RESET
006164  012767  006252  172714          MOV     #9$,$LPADR       ;SETUP LOOP ADDRESS
006172  012767  006252  172710          MOV     #9$,$LPERR       ;SETUP ERROR LOOP
006200  013767  177776  172762          MOV     @#PSW,$TMP3      ;SAVE PSW
006206  032737  000020  177776          BIT     #BIT4,@#PSW      ;IS T BIT ON?
006214  001416                          BEQ     9$               ;BRANCH IF NO
006216  012746  000340                  MOV     #PR7,-(SP)       ;PUT NEW PSW ON STACK
006222  012746  006252                  MOV     #9$,-(SP)        ;PUT RETURN ADDR ON STACK
006226  000006                          RTT                      ;TURN T BIT OFF
006230  012767  006436  172736          MOV     #TST4,$ESCAPE    ;SAVE START ADDRESS OF NEXT TEST
006236  012767  006436  173024          MOV     #TST4,NEXTTST    ;SAVE START ADDRESS OF NEXT TEST
006244  012737  006406  000010          MOV     #2$,@#RESVEC     ;SETUP LOC 10 TO TRAP TO THIS ROUTINE
006252  012706  001100          9$:     MOV     #STACK,SP        ;INITIALIZE STACK
006256  012746  100000                  MOV     #BIT15,-(SP)     ;SET SIGN BIT ON STACK
006262  012766  100000  177776          MOV     #BIT15,-2(SP)    ;SET SIGN BIT AT LOCATION 1074
006270  012705  006326                  MOV     #1$,R5           ;SETUP R5 FOR MARK INSTRUCTION
006274  000240                          NOP                      ;SYNC POINT
006276  006401                          MARK    1                ;EXECUTE INSTRUCTION UNDER TEST
006300  022701  006300          8$:     CMP     #8$,R1           ;DID MTP OCCUR?
006304  001401                          BEQ     3$               ;BRANCH IF NO
006306  104011                  5$:     ERROR   11               ;FORK A FAILED TO MTP
006310  013701  001074          3$:     MOV     @#1074,R1        ;DID MFP OCCUR?
006314  100401                          BMI     4$               ;BRANCH IF NO
006316  104012                          ERROR   12               ;FORK A FAILED TO MFP
```

```
006320  012706  001076          4$:     MOV     #1076,SP        ;INITIALIZE SP INCASE MARK CHANGED IT
006324  104013                          ERROR   13              ;PCB DID NOT LOAD FROM R5
006326  020627  006304          1$:     CMP     SP,#5$-2        ;DID SP GET LOADED PROPERLY?
006332  001410                          BEQ     6$              ;BRANCH IF YES
006334  010667  172614                  MOV     SP,$REG0        ;SAVE SP FOR TYPEOUT
006340  012767  006306  172610          MOV     #5$,$REG1       ;STORE ADDRESS OF 5$ FOR TYPEOUT
006346  012706  001100                  MOV     #STACK,SP       ;REINITIALIZE SP
006352  104014                          ERROR   14              ;MARK DID NOT LOAD SP PROPERLY
006354  012706  001100          6$:     MOV     #STACK,SP       ;RESTORE THE SP
006360  020527  006300                  CMP     R5,#8$          ;DID R5 GET LOADED PROPERLY?
006364  001424                          BEQ     TST4            ;;BRANCH IF YES
006366  010567  172562                  MOV     R5,$REG0        ;SAVE R5 FOR TYPEOUT
006372  013767  006304  172556          MOV     @#5$-2,$REG1    ;STORE ADDRESS OF 5$-2 FOR TYPEOUT
006400  012706  001076                  MOV     #1076,SP        ;RESTORE THE SP
006404  104015                          ERROR   15              ;R5 DID NOT LOAD PROPERLY
006406  012737  000012  000010  2$:     MOV     #12,@#RESVEC    ;RESTORE RESERVED INSTR VECTOR
006414  021627  006300                  CMP     (SP),#8$        ;DID PCB GET INCREMENTED BY D67.01?
006420  001003                          BNE     7$              ;BRANCH IF NO
006422  012706  001100                  MOV     #STACK,SP       ;RESTORE THE SP
006426  104016                          ERROR   16              ;FORK A FAILED TO RSD.00
006430  012706  001100          7$:     MOV     #STACK,SP       ;RESTORE THE SP
006434  104017                          ERROR   17              ;FORK A FAILED TO D67.01
                                ;;*************************************************************
                                ;*TEST 4        ASH*DMO
                                ;*
                                ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
                                ;*
                                ;*      IF GRAJ SC05 L DOES NOT GO LOW OR DOES NOT GET THRU TO
                                ;*      RACK BRCAB04 L A SHIFT RIGHT WILL OCCUR.
                                ;*      IF THE SHIFT COUNTER DOES NOT SHIFT OR GRAJ SC=0 DOES NOT GO
                                ;*      LOW THE PROCESSOR WILL HANG UP IN STATE ASH.41.
                                ;*
                                ;*      ROM FLOW-52,305,257,166 LEFT SHIFT
                                ;*               52,305,277      RIGHT SHIFT
                                ;;*************************************************************
006436  000004                  TST4:   SCOPE
006440  012767  007200  172622          MOV     #TST5,NEXTTST   ;SAVE ADDRESS OF NEXT TEST
006446  012767  006512  172432          MOV     #19$,$LPADR     ;SETUP LOOP ADR
006454  012767  006512  172426          MOV     #19$,$LPERR     ;SETUP ERROR LOOP
006462  032767  000020  172500          BIT     #BIT4,$TMP3     ;WAS T BIT ON?
006470  001405                          BEQ     18$             ;BRANCH IF NO
006472  012746  000360                  MOV     #360,-(SP)      ;PUT NEW PSW ON STACK
006476  012746  006504                  MOV     #18$,-(SP)      ;PUT RETURN ADR ON STACK
006502  000006                          RTT                     ;TURN T BIT ON

                                ;
                                ;ARITHMETIC LEFT SHIFT
006504  012737  007066  000010  18$:    MOV     #1$,@#RESVEC    ;SET UP FOR FORK A FAILURE
006512  012706  001100          19$:    MOV     #STACK,SP       ;INITIALIZE THE SP
006516  012700  177701                  MOV     #177701,R0      ;PUT SHIFT COUNT IN R0 (+1)
006522  012701  100000                  MOV     #BIT15,R1       ;SET BIT 15 IN REGISTER TO BE SHIFTED
006526  000240                          NOP                     ;SYNC POINT
006530  072100                          ASH     R0,R1           ;EXECUTE INSTRUCTION UNDER TEST
006532  103414                          BCS     2$              ;BRANCH IF SHIFT WORKED
006534  020127  140000                  CMP     R1,#140000      ;DID SHIFT GO IN WRONG DIRECTION?
006540  001002                          BNE     3$              ;BRANCH IF NO
006542  104020                          ERROR   20              ;SHIFTED RIGHT INSTEAD OF LEFT
```

```
006544  000425                           BR      9$
006546  020127  100000          3$:      CMP     R1,#BIT15          ;DID R1 SHIFT AT ALL?
006552  001002                           BNE     4$                 ;BRANCH IF YES
006554  104021                           ERROR   21                 ;R1 DID NOT SHIFT
006556  000420                           BR      9$
006560  104022                  4$:      ERROR   22                 ;R1 SHIFTED BUT CARRY DID NOT SET
                                                                    ;SHIFT COUNTER COULD BE STUCK
006562  000416                           BR      9$
006564  102003                  2$:      BVC     10$                ;BRANCH IF V DID NOT SET
006566  100402                           BMI     10$                ;BRANCH IF N DID NOT CLEAR
006570  001001                           BNE     10$                ;BRANCH IF Z DID NOT SET
006572  000412                           BR      9$                 ;CC'S OK
006574  013767  177776  172404  10$:     MOV     @#PSW,$ERPSW       ;SAVE PSW
006602  042767  177760  172376           BIC     #177760,$ERPSW     ;MASK OFF CC'S
006610  012767  000007  172344           MOV     #7,$TMP0           ;PUT EXPECTED CC'S IN STORAGE
006616  104031                           ERROR   31                 ;STATE ASH.40 DID NOT LOAD CC'S CORRECTLY
                                ;;*******************************************************************
                                ;ARITHMETIC RIGHT SHIFT TEST
006620  012767  006626  172262  9$:      MOV     #64$,$LPERR        ;SETUP ERROR LOOP
006626  012701  100001          64$:     MOV     #100001,R1         ;SETUP R1 FOR RIGH SHIFT
006632  012700  000077                   MOV     #77,R0             ;PUT SHIFT COUNT IN R0 (-1)
006636  005002                           CLR     R2                 ;ENSURE R2 CLEAR
006640  000240                           NOP                        ;SYNC POINT
006642  072100                           ASH     R0,R1              ;EXECUTE RIGHT SHIFT
006644  005502                           ADC     R2                 ;SAVE CARRY IN R2
006646  020127  140000                   CMP     R1,#140000         ;DID R1 SHIFT IN RIGHT DIRECTION?
006652  001417                           BEQ     5$                 ;BRANCH IF YES
006654  005701                           TST     R1                 ;DID R1 SHIFT LEFT INSTEAD OF RIGHT?
006656  001002                           BNE     6$                 ;BRANCH IF NO
006660  104023                           ERROR   23                 ;R1 SHIFTED WRONG DIRECTION
006662  000416                           BR      8$
006664  022701  040000          6$:      CMP     #40000,R1          ;DID SHIFT FAIL TO SIGN FILL?
006670  001001                           BNE     7$                 ;BRANCH IF NO
006672  104024                           ERROR   24                 ;SHIFT RIGHT DID NOT SIGN FILL
006674  010167  172256          7$:      MOV     R1,$REG1           ;SAVE R1 FOR TYPE OUT
006700  012767  140000  172256           MOV     #140000,$TMP1      ;PUT EXPECTED VALUE IN STORAGE
006706  104025                           ERROR   25                 ;R1 SHIFTED, BUT DON'T KNOW WHERE
006710  000403                           BR      8$
006712  005702                  5$:      TST     R2                 ;DID CARRY GET SET ON SHIFT?
006714  001001                           BNE     8$                 ;BRANCH IF YES
006716  104026                           ERROR   26                 ;SHIFT OK BUT CARRY DID NOT LOAD
                                ;;*******************************************************************
                                ;CONDITION CODE LOAD TEST(STATE ASH.30)
006720  012767  006726  172162  8$:      MOV     #63$,$LPERR        ;SETUP ERROR LOOP
006726  012701  100000          63$:     MOV     #BIT15,R1          ;SETUP R0 TO TEST CC'S IN STATE ASH.30
                                                                    ;R0 HAS SHIFT COUNT (-1)
006732  000250                           CLN                        ;SET UP CC'S TO
006734  000266                           +SEV!SEZ                   ;COMPLIMENT OF EXPECTED RESULT
                                                                    ;AND OSCILLOSCOPE SYNC POINT
006736  072100                           ASH     R0,R1              ;EXECUTE THE SHIFT
006740  013767  177776  172240           MOV     @#PSW,$ERPSW       ;SAVE PSW
006746  042767  177760  172232           BIC     #177760,$ERPSW     ;MASK OFF THE CC'S
006754  022767  000010  172224           CMP     #10,$ERPSW         ;DID CC'S COME OUT OK?
006762  001404                           BEQ     12$                ;BRANCH IF YES
006764  012767  000010  172170           MOV     #10,$TMP0          ;PUT EXPECTED VALUE IN STORAGE
006772  104072                           ERROR   72                 ;STATE ASH.30 DID NOT LOAD CC'S CORRECTLY
```

```
                                        ;;*****************************************************************
                                        ;SHIFT COUNT=0
006774  012767  007002  172106  12$:    MOV     #62$,$LPERR     ;SETUP ERROR LOOP
007002  012701  100000          62$:    MOV     #BIT15,R1       ;SET SIGN BIT IN REGISTER TO BE SHIFTED
007006  005000                          CLR     R0              ;SET SHIFT COUNT TO ZERO
007010  000263                          +SEC!SEV                ;SETUP CC'S TO COMPLIMENT
007012  000250                          CLN                     ;OF EXPECTED VALUE
                                                                ;AND OSCILLOSCOPE SYNC POINT
007014  072100                          ASH     R0,R1           ;EXECUTE INSTRUCTION
007016  013767  177776  172162          MOV     @#PSW,$ERPSW    ;SAVE PSW
007024  042767  177760  172154          BIC     #177760,$ERPSW  ;MASK OFF THE CC'S
007032  022767  000010  172146          CMP     #10,$ERPSW      ;DID CC'S COME OUT OK?
007040  001416                          BEQ     16$             ;BRANCH IF YES
007042  022701  100000          13$:    CMP     #BIT15,R1       ;DID R1 SHIFT?
007046  001005                          BNE     15$             ;BRANCH IF YES
007050  012767  000010  172104          MOV     #10,$TMPO       ;SAVE EXPECTED VALUE
007056  104027                          ERROR   27              ;STATE ASH.20 DID NOT LOAD CC'S CORRECTLY
007060  000406                          BR      16$
007062  104030                  15$:    ERROR   30              ;R1 SHIFTED WHEN NOT SUPPOSE TO
007064  000404                          BR      16$
007066  012737  000012  000010  1$:     MOV     #12,@#RESVEC    ;RESTORE RESERVED VECTOR
007074  104032                          ERROR   32              ;FORK A FAILED
                                        ;;*****************************************************************
                                        ;CONDITION CODE TEST OF STATE ASH.41
007076  012767  007104  172004  16$:    MOV     #61$,$LPERR     ;SETUP ERROR LOOP
007104  012700  000002          61$:    MOV     #2,R0           ;PUT SHIFT COUNT IN R0
                                                                ;TO TEST STATE ASH.41
007110  012701  060000                  MOV     #60000,R1       ;SETUP R1 FOR SHIFT LEFT
007114  000274                          +SEN!SEZ                ;SETUP CC'S TO COMPLEMENT
007116  000243                          +CLV!CLC                ;OF EXPECTED VALUE
                                                                ;AND OSCILLOSCOPE SYNC POINT
007120  072100                          ASH     R0,R1           ;EXECUTE INSTRUCTION UNDER TEST
007122  013767  177776  172056          MOV     @#PSW,$ERPSW    ;SAVE PSW
007130  020127  100000                  CMP     R1,#BIT15       ;DID R1 SHIFT CORRECTLY?
007134  001406                          BEQ     17$             ;BRANCH IF YES
007136  010167  172014                  MOV     R1,$REG1        ;SAVE R1 FOR TYPEOUT
007142  012767  100000  172014          MOV     #BIT15,$TMP1    ;STORE EXPECTED VALUE
007150  104073                          ERROR   73              ;STATE ASH.41 FAILED
007152  042767  177760  172026  17$:    BIC     #177760,$ERPSW  ;MASK OFF CC'S
007160  022767  000013  172020          CMP     #13,$ERPSW      ;DID CC'S LOAD CORRECTLY?
007166  001404                          BEQ     TST5            ;;BRANCH IF YES
007170  012767  000013  171764          MOV     #13,$TMPO       ;STORE EXPECTED VALUE
007176  104074                          ERROR   74              ;BAD CC'S DUE TO ASH.41
                                        ;;*****************************************************************
                                        ;*TEST 5                ASH*DM1
                                        ;*
                                        ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
                                        ;*
                                        ;*      IF FORK B FAILS EXECUTION WILL EITHER GO TO RSD.00 OR
                                        ;*      MUL.00.
                                        ;*      MUL.00 WILL ONLY BE ENTERED IF EITHER B FORK MUX INPUT B2
                                        ;*      DOES NOT GO HIGH OR THE MUX IS BAD.
                                        ;*
                                        ;*      ROM FLOW-1,175,62,52,305,257
                                        ;;*****************************************************************
007200  000004                  TST5:   SCOPE
```

```
007202  012767  007314  171764          MOV     #TST6,$ESCAPE    ;SAVE START ADDRESS OF NEXT TEST
007210  012767  007314  172052          MOV     #TST6,NEXTTST    ;SAVE START ADDRESS OF NEXT TEST
007216  012706  001100                  MOV     #STACK,SP        ;INITIALIZE THE STACK
007222  012737  007262  000010          MOV     #1$,@#RESVEC     ;SETUP RESERVED VECTOR
007230  012700  001164                  MOV     #$TMP1,R0        ;PUT ADDRESS OF SHIFT COUNT IN R0
007234  012710  000001                  MOV     #1,(R0)          ;SET SHIFT COUNT TO ONE
007240  012701  100000                  MOV     #BIT15,R1        ;SETUP REGISTER TO BE SHIFTED
007244  000240                          NOP                      ;OSCILLOSCOPE SYNC POINT
007246  072110                          ASH     (R0),R1          ;EXECUTE INSTRUCTION UNDER TEST
007250  103421                          BCS     TST6             ;;TEST OK, GO TO NEXT TEST
007252  005701                          TST     R1               ;DID MULTIPLY OCCUR?
007254  100401                          BMI     2$               ;BRANCH IF YES
007256  104033                          ERROR   33               ;STATE ASH.00 FAILED
007260  104034                  2$:     ERROR   34               ;FORK B FAILED INTO MUL.00
007262  012737  007276  000010  1$:     MOV     #3$,@#RESVEC     ;SETUP RESVEC
007270  005000                          CLR     R0               ;PUT EVEN ADDRESS IN R0
007272  000240                          NOP                      ;OSCILLOSCOPE SYNC POINT
007274  072120                          ASH     (R0)+,R1         ;EXECUTE DM2
007276  012737  000012  000010  3$:     MOV     #12,@#RESVEC     ;RESTORE RESERVED VECTOR
007304  005700                          TST     R0               ;DID R0 AUTO INCREMENT?
007306  001401                          BEQ     4$               ;BRANCH IF NO
007310  104035                          ERROR   35               ;FORK B FAILED
007312  104036                  4$:     ERROR   36               ;FORK A FAILED
                                ;;*****************************************************************
                                ;*TEST 6        ASH*DM2
                                ;*
                                ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
                                ;*
                                ;*      ALL OTHER LOGIC HAS BEEN TESTED
                                ;*
                                ;*      ROM FLOW-2,175,62,52,305
                                ;;*****************************************************************
007314  000004                  TST6:   SCOPE
007316  012767  007374  171650          MOV     #TST7,$ESCAPE    ;SAVE START ADDRESS OF NEXT TEST
007324  012767  007374  171736          MOV     #TST7,NEXTTST    ;SAVE START ADDRESS OF NEXT TEST
007332  012706  001100                  MOV     #STACK,SP        ;INITIALIZE THE SP
007336  012737  007364  000010          MOV     #1$,@#RESVEC     ;PUT ADDRESS OF 1$ IN RESERVED VECTOR
007344  012700  001164                  MOV     #$TMP1,R0        ;PUT ADDRESS OF SHIFT COUNT IN R0
007350  005010                          CLR     (R0)             ;PUT SHIFT COUNT OF ZERO IN $TMP1
007352  072120                          ASH     (R0)+,R1         ;EXECUTE INSTRUCTION UNDER TEST
007354  012737  000012  000010          MOV     #12,@#RESVEC     ;RESTORE RESVEC
007362  000404                          BR      TST7             ;;GO TO NEXT TEST
007364  012737  000012  000010  1$:     MOV     #12,@#RESVEC     ;RESTORE RESERVED VECTOR
007372  104037                          ERROR   37               ;FORK A FAILED
                                ;;*****************************************************************
                                ;*TEST 7        ASH*DM4
                                ;*
                                ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
                                ;*
                                ;*      ALL OTHER LOGIC HAS BEEN TESTED.
                                ;*
                                ;*      ROM FLOW-4,122,177,62,52,305
                                ;;*****************************************************************
007374  000004                  TST7:   SCOPE
007376  012767  007460  171570          MOV     #TST10,$ESCAPE   ;SAVE START ADDRESS OF NEXT TEST
007404  012767  007460  171656          MOV     #TST10,NEXTTST   ;SAVE START ADDRESS OF NEXT TEST
```

```
007412  012706  001100                  MOV     #STACK,SP       ;INITIALIZE THE SP
007416  012737  007450  000010          MOV     #1$,@#RESVEC    ;PUT ADDRESS OF 1$ IN RESERVED VECTOR
007424  012700  001166                  MOV     #$TMP2,R0       ;PUT ADDRESS OF SHIFT COUNT+2 IN R0
007430  005060  177776                  CLR     -2(R0)          ;PUT SHIFT COUNT IN $TMP1
007434  000240                          NOP                     ;OSCILLOSCOPE SYNC POINT
007436  072140                          ASH     -(R0),R1        ;EXECUTE INSTRUCTION UNDER TEST
007440  012737  000012  000010          MOV     #12,@#RESVEC    ;RESTORE RESVEC
007446  000404                          BR      TST10           ;;GO TO NEXT TEST
007450  012737  000012  000010  1$:     MOV     #12,@#RESVEC    ;RESTORE RESVEC
007456  104040                          ERROR   40              ;FORK A FAILED
                                ;;*************************************************************
                                ;*TEST 10        ASHC*DM0
                                ;*
                                ;*      NEITHER FORK A NOR BEN03 SHOULD FAIL.
                                ;*
                                ;*      IF THE INSTRUCTION FAILS, ONE OF THE ASC STATES IS BAD.
                                ;*
                                ;*      ONCE IT IS DETERMINED THAT THE INSTRUCTION WORKS,
                                ;*      A BIT STUCK TEST IS PERFORMED ON THE SHIFT COUNTER.
                                ;*
                                ;*      ROM FLOW-53,306,267,227     RIGHT SHIFT
                                ;*               53,306,247,176,136 LEFT SHIFT
                                ;*               53,306,207         NO SHIFT
                                ;;*************************************************************
007460  000004                  TST10:  SCOPE
007462  012767  010302  171600          MOV     #TST11,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
                                ;
                                ;ARITHMETIC RIGHT SHIFT COMBINED
007470  012700  100001                  MOV     #100001,R0      ;SETUP R0 FOR RIGHT SHIFT
007474  012701  000001                  MOV     #BIT0,R1        ;SETUP R1 FOR RIGH SHIFT
007500  012702  000077                  MOV     #77,R2          ;PUT SHIFT COUNT (-1) IN R2
007504  000262                          SEV                     ;ENSURE V SET
                                                                ;AND OSCILLOSCOPE SYNC POINT
007506  073002                          ASHC    R2,R0           ;EXECUTE INSTRUCTION UNDER TEST
007510  013767  177776  171470          MOV     @#PSW,$ERPSW    ;SAVE PSW
007516  042767  177760  171462          BIC     #177760,$ERPSW  ;MASK OFF THE CC'S
007524  022767  000011  171454          CMP     #11,$ERPSW      ;DID CC'S LOAD PROPERLY?
007532  001014                          BNE     1$              ;BRANCH IF NO
007534  005701                          TST     R1              ;DID R1 GET SIGN BIT SET
007536  100017                          BPL     2$              ;BRANCH IF NO
007540  022700  140000                  CMP     #140000,R0      ;DID R0 SIGN FILL?
007544  001427                          BEQ     3$              ;BRANCH IF YES
007546  012767  140000  171406          MOV     #140000,$TMP0   ;EXPECTED VALUE
007554  010067  171374                  MOV     R0,$REG0        ;SAVE R0 FOR TYPEOUT
007560  104041                          ERROR   41              ;R0 DID NOT SIGN FILL
007562  000420                          BR      3$
007564  012767  000011  171370  1$:     MOV     #11,$TMP0       ;EXPECTED VALUE
007572  104042                          ERROR   42              ;BAD CC'S
007574  000413                          BR      3$
007576  012767  140000  171356  2$:     MOV     #140000,$TMP0   ;GET EXPECTED VALUES
007604  012767  100000  171352          MOV     #100000,$TMP1   ;FOR TYPEOUT
007612  010067  171336                  MOV     R0,$REG0        ;SAVE R0 & R1 FOR TYPEOUT
007616  010167  171334                  MOV     R1,$REG1
007622  104043                          ERROR   43              ;R0<0> DID NOT GO TO R1<15>
                                ;;*************************************************************
                                ;ARITHMETIC LEFT SHIFT
```

```
007624  012767  007632  171256  3$:     MOV     #64$,$LPERR       ;SETUP ERROR LOOP
007632  012700  100000          64$:    MOV     #BIT15,R0         ;SETUP R0 FOR LEFT SHIFT
007636  012701  100001                  MOV     #100001,R1        ;SETUP R1 FOR LEFT SHIFT
007642  012702  000001                  MOV     #BIT0,R2          ;PUT SHIFT COUNT (+1) IN R2
007646  000274                          +SEZ!SEN                  ;ENSURE Z AND N SET
                                                                  ;AND OSCILLOSCOPE SYNC POINT
007650  073002                          ASHC    R2,R0             ;EXECUTE INSTRUCTION UNDER TEST
007652  013767  177776  171326          MOV     @#PSW,$ERPSW      ;SAVE PSW
007660  042767  177760  171320          BIC     #177760,$ERPSW    ;MASK OFF THE CC'S
007666  010067  171262                  MOV     R0,$REG0          ;SAVE REG0
007672  010167  171260                  MOV     R1,$REG1          ;SAVE REG1
007676  012767  000001  171256          MOV     #1,$TMP0          ;GET EXPECTED VALUES
007704  012767  000002  171252          MOV     #2,$TMP1          ;OF REGISTERS
007712  022767  000003  171266          CMP     #3,$ERPSW         ;ARE CC'S CORRECT?
007720  001010                          BNE     7$                ;BRANCH IF NO
007722  022701  000002                  CMP     #2,R1             ;DID R1 SHIFT PROPERLY?
007726  001012                          BNE     11$               ;BRANCH IF NO
007730  022700  000001                  CMP     #1,R0             ;DID R0 SHIFT PROPERLY?
007734  001410                          BEQ     12$               ;BRANCH IF YES
007736  104044                          ERROR   44                ;R0 DID NOT GET SHIFTED PROPERLY
007740  000406                          BR      12$
007742  012767  000003  171212  7$:     MOV     #3,$TMP0          ;SAVE EXPECTED VALUE
007750  104045                          ERROR   45                ;BAD CC'S
007752  000401                          BR      12$
007754  104046                  11$:    ERROR   46                ;R1 DID NOT SHIFT PROPERLY
                                ;;**************************************************************
                                ;NO SHIFT
007756  012767  007764  171124  12$:    MOV     #63$,$LPERR       ;SETUP ERROR LOOP
007764  012701  040000          63$:    MOV     #40000,R1         ;SETUP R1 FOR NO SHIFT
007770  005002                          CLR     R2                ;PUT SHIFT COUNT (0) IN R2
007772  012700  100000                  MOV     #BIT15,R0         ;SET SIGN BIT IN R0 & SET N
007776  000277                          SCC                       ;ENSURE ALL CC'S SET
010000  000250                          CLN                       ;ENSURE N CLEAR
                                                                  ;AND OSCILLOSCOPE SYNC POINT
010002  073002                          ASHC    R2,R0             ;EXECUTE INSTRUCTION UNDER TEST
010004  013767  177776  171174          MOV     @#PSW,$ERPSW      ;SAVE PSW
010012  012767  000010  171142          MOV     #10,$TMP0         ;SAVE EXPECTED VALUE
010020  042767  177760  171160          BIC     #177760,$ERPSW    ;MASK OFF THE CC'S
010026  022767  000010  171152          CMP     #10,$ERPSW        ;DID THE CC'S COME OUT CORRECT?
010034  001401                          BEQ     14$               ;BRANCH IF YES
010036  104047                  13$:    ERROR   47                ;BAD CC'S ON NO SHIFT
                                ;;**************************************************************
                                ;CHECK ROTATE
010040  012767  010046  171042  14$:    MOV     #62$,$LPERR       ;SETUP ERROR LOOP
010046  012701  100000          62$:    MOV     #BIT15,R1         ;SETUP R1 FOR A ROTATE
010052  010700                          MOV     PC,R0             ;PUT RANDOM NUMBER IN R0
010054  010067  171102                  MOV     R0,$TMP0          ;SAVE R0
010060  012702  177761                  MOV     #-17,R2           ;PUT SHIFT COUNT (-17) IN R2
010064  012767  000001  171072          MOV     #1,$TMP1          ;EXPECTED VALUE OF R1 IN $TMP1
010072  000240                          NOP                       ;OSCILLOSCOPE SYNC POINT
010074  073102                          ASHC    R2,R1             ;EXECUTE INSTRUCTION UNDER TEST
010076  022701  000001                  CMP     #1,R1             ;DID R1 ROTATE CORRECTLY?
010102  001405                          BEQ     15$               ;BRANCH IF YES
010104  010067  171044                  MOV     R0,$REG0          ;SAVE R0 FOR TYPEOUT
010110  010167  171042                  MOV     R1,$REG1          ;SAVE R1 FOR TYPEOUT
010114  104050                          ERROR   50                ;R1 DID NOT ROTATE PROPERLY
```

```
                                      ;;********************************************************************
                                      ;THE FOLLOWING CODE CHECKS THE BITS IN THE SC
010116  012767  010124  170764  15$:      MOV     #61$,$LPERR        ;SETUP ERROR LOOP
010124  012700  000040          61$:      MOV     #40,R0             ;SETUP R1 FOR RIGHT SHIFT
010130  012702  000052                    MOV     #52,R2             ;PUT SHIFT COUNT (-26) IN R2
010134  005001                            CLR     R1                 ;ENSURE R0 CLEAR
010136  005067  171020                    CLR     $TMP0              ;PUT EXPECTED VALUES OF
010142  005067  171016                    CLR     $TMP1              ;R0 & R1 IN STORAGE
010146  012767  000001  171012            MOV     #1,$TMP2           ;C BIT EXPECTED
010154  000240                            NOP                        ;OSCILLOSCOPE SYNC POINT
010156  073002                            ASHC    R2,R0              ;EXECUTE SHIFT
010160  013767  177776  171020            MOV     @#PSW,$ERPSW       ;SAVE PSW
010166  103410                            BCS     16$                ;BRANCH IF CORRECT SHIFT
010170  042767  177776  171010            BIC     #177776,$ERPSW     ;MASK OFF C BIT
010176  010067  170752                    MOV     R0,$REG0
010202  010167  170750                    MOV     R1,$REG1
010206  104051                            ERROR   51                 ;STUCK BITS IN SC
010210  012767  010216  170672  16$:      MOV     #60$,$LPERR        ;SETUP ERROR LOOP
010216  012701  004000          60$:      MOV     #4000,R1           ;SET UP R1 FOR LEFT SHIFT
010222  005000                            CLR     R0                 ;ENSURE R0 CLEAR
010224  012702  000025                    MOV     #25,R2             ;PUT SHIFT COUNT (+25) IN R2
010230  000240                            NOP                        ;OSCILLOSCOPE SYNC POINT
010232  073002                            ASHC    R2,R0              ;EXECUTE SHIFT
010234  013767  177776  170744            MOV     @#PSW,$ERPSW
010242  042767  177776  170736            BIC     #177776,$ERPSW     ;MASK OFF C BIT
010250  103414                            BCS     TST11              ;;GO TO NEXT TEST IF CORRECT SHIFT
010252  012767  000000  170674            MOV     #R0,$REG0          ;SAVE R0 FOR TYPEOUT
010260  012767  000001  170670            MOV     #R1,$REG1          ;SAVE R1 FOR TYPEOUT
010266  005067  170670                    CLR     $TMP0              ;SAVE EXPECTED
010272  012767  000001  170664            MOV     #1,$TMP1           ;VALUES
010300  104052                            ERROR   52                 ;STUCK BITS IN SC
                                      ;;********************************************************************
                                      ;*TEST 11           ASHC*DM1
                                      ;*
                                      ;*         THE ONLY POSSIBLE FAILURES ARE FORK B OR STATE ASC.00.
                                      ;*
                                      ;*         IF FORK B FAILS EXECUTION WILL EITHER GO TO RSD.00 OR
                                      ;*         ASH.00.
                                      ;*         ASH.00 WOULD PERFORM AN ASH INSTRUCTION INSTEAD OF AN ASHC.
                                      ;*
                                      ;*         ROM FLOW-1,175,63,53,306,267,227
                                      ;;********************************************************************
010302  000004                  TST11:    SCOPE
010304  012767  010442  170662            MOV     #TST12,$ESCAPE     ;SAVE START ADDRESS OF NEXT TEST
010312  012767  010442  170750            MOV     #TST12,NEXTTST     ;SAVE START ADDRESS OF NEXT TEST
010320  012706  001100                    MOV     #STACK,SP          ;INITIALIZE THE SP
010324  012737  010432  000010            MOV     #1$,@#RESVEC       ;SETUP RESERVED VECTOR
010332  012702  001166                    MOV     #$TMP2,R2          ;PUT ADDRESS OF SHIFT COUNT IN R2
010336  012700  000001                    MOV     #1,R0              ;SETUP R0 FOR RIGHT SHIFT
010342  005001                            CLR     R1                 ;SETUP R1 FOR RIGHT SHIFT
010344  012712  000077                    MOV     #77,(R2)           ;PUT SHIFT COUNT (-1) IN $TMP2
010350  005067  170606                    CLR     $TMP0              ;PUT EXPECTED VALUES OF
010354  012767  100000  170602            MOV     #BIT15,$TMP1       ;R0 & R1 IN STORAGE
010362  000240                            NOP                        ;OSCILLOSCOPE SYNC POINT
010364  073012                            ASHC    (R2),R0            ;EXECUTE INSTRUCTION UNDER TEST
010366  103012                            BCC     2$                 ;BRANCH IF FORK B DID NOT FAIL
```

```
010370  013767  177776  170610               MOV     @#PSW,$ERPSW      ;SAVE PSW FOR TYPEOUT
010376  005067  170560                       CLR     $TMP0             ;SAVE EXPECTED VALUE
010402  010067  170546                       MOV     R0,$REG0
010406  010167  170544                       MOV     R1,$REG1
010412  104053                               ERROR   53                ;FORK B FAILED TO ASH.00
010414  005701                       2$:     TST     R1                ;DID R1 SIGN BIT SET?
010416  100411                               BMI     TST12             ;;BRANCH IF YES
010420  010067  170530                       MOV     R0,$REG0          ;SAVE R0 & R1
010424  010167  170526                       MOV     R1,$REG1          ;FOR TYPEOUT
010430  104054                               ERROR   54                ;STATE ASC.00 FAILED
010432  012737  000012  000010       1$:     MOV     #12,@#RESVEC      ;RESTORE RESVEC
010440  104055                               ERROR   55                ;FORK A FAILED TO RSD.00
                                     ;;*************************************************************
                                     ;*TEST 12            MUL*DM0
                                     ;*
                                     ;*        FORK A SHOULD NOT FAIL.
                                     ;*        THE FOLLOWING WOULD BE BEN11 FAILURES:
                                     ;*        IF EITHER GRAD DR00 IS STUCK HIGH OR NOT GETTING THRU TO
                                     ;*        RACK E64(C1) OR RACK E64 IS BAD (INPUT C1 FAILED HIGH)
                                     ;*        THE MULITPLICAND WILL BE MULITPLIED BY 177777.
                                     ;*        IF EITHER GRAD DR00 IS STUCK LOW OR NOT GETTING THRU TO
                                     ;*        RACK E64(C1) OR RACK E64 IS BAD (INPUT C1 FAILED LOW)
                                     ;*        THE MULTIPLICAND WILL BE MULTIPLIED BY ZERO.
                                     ;*        IF GRAJ SC=0 IS NOT GETTING TO RACK E50(C1) AND RACK E50
                                     ;*         IS BAD (INPUT C1 FAILED LOW) THE MULTIPLICAND WILL ONLY
                                     ;*        BE MULTIPLIED BY BIT 0 OF THE MULTIPLIER.
                                     ;*        IF RACK E50(C1) FAILS HIGH THE PROCESSOR WILL HANG UP IN
                                     ;*        STATE MUL.20(266).
                                     ;*        IF THE INSTRUCTION FAILS TO MULTIPLY CORRECTLY AND ONE
                                     ;*        OF THE ABOVE CONDITIONS CANNOT BE DETERMINED THEN THE
                                     ;*        FAILURE COULD BE IN THE INSTRUCTION DECODE ROM.
                                     ;*
                                     ;*        IF THE CC'S COME UP BAD THEN THE FAILURE COULD BE IN
                                     ;*        STATES MUL.40,50, OR 60, OR IN THE CONDITION CODE ROM.
                                     ;*
                                     ;*        ROM FLOW-50,102,266/246,226/206,310
                                     ;;*************************************************************
010442  000004                       TST12:  SCOPE
010444  012767  011004  170616               MOV     #TST13,NEXTTST    ;SAVE ADDRESS OF NEXT TEST
                                     ;
                                     ;MULTIPLY 3 TIMES 2
010452  012700  000002                       MOV     #2,R0             ;PUT MULTIPLICAND IN R0
010456  012702  000003                       MOV     #3,R2             ;PUT MULTIPLIER IN R2
010462  005001                               CLR     R1                ;ENSURE R1 CLEAR
010464  005067  170472                       CLR     $TMP0             ;PUT EXPECTED VALUE OF
010470  012767  000006  170466               MOV     #6,$TMP1          ;R0 & R1 IN STORAGE
010476  000277                               SCC                       ;MULTIPLY SHOULD CLEAR ALL OF THEM
                                                                       ;AND OSCILLOSCOPE SYNC POINT
010500  070002                               MUL     R2,R0             ;EXECUTE INSTRUCTION UNDER TEST
010502  013767  177776  170476               MOV     @#PSW,$ERPSW      ;SAVE PSW
010510  010067  170440                       MOV     R0,$REG0          ;SAVE R0 & R1
010514  010167  170436                       MOV     R1,$REG1          ;FOR TYPEOUT
010520  020127  000006                       CMP     R1,#6             ;DID R1 GET LOW PRODUCT
010524  001002                               BNE     1$                ;BRANCH IF NO
010526  005700                               TST     R0                ;DID R0 GET UPPER PRODUCT?
010530  001423                               BEQ     2$                ;BRANCH IF YES
```

```
010532  020127  177776          1$:     CMP     R1,#177776      ;DID R0 GET MULTIPLIED BY 177777?
010536  001002                          BNE     3$              ;BRANCH IF NO
010540  104056                          ERROR   56              ;EITHER GRAD DR00 STUCK H OR RACK E64 BAD
010542  000430                          BR      8$
010544  005701                  3$:     TST     R1              ;DID R1 STAY ZERO?
010546  001401                          BEQ     4$              ;BRANCH IF YES
010550  000404                          BR      5$              ;CONTINUE ERROR ANALYSIS
010552  005700                  4$:     TST     R0              ;DID R1 & R0 GO TO ZERO?
010554  001002                          BNE     5$              ;BRANCH IF NO
010556  104057                          ERROR   57              ;EITHER GRAD DR00 STUCK L OR RACK E64 BAD
010560  000421                          BR      8$
010562  020127  100000          5$:     CMP     R1,#BIT15       ;DID R0 ONLY GET MULTIPLIED BY BIT 0?
010566  001002                          BNE     6$              ;BRANCH IF NO
010570  104060                          ERROR   60              ;RACH E50(C1) FAILED LOW
010572  000414                          BR      8$
010574  104061                  6$:     ERROR   61              ;CAN'T DETERMINE WHAT HAPPENED
010576  000412                          BR      8$
010600  042767  177760  170400  2$:     BIC     #177760,$ERPSW  ;MASK OFF THE CC'S
010606  022767  000000  170372          CMP     #0,$ERPSW
010614  001403                          BEQ     8$              ;BRANCH IF YES
010616  005067  170340          7$:     CLR     $TMP0           ;PUT EXPECTED PSW IN $TM0
010622  104062                          ERROR   62              ;BAD CC'S
                                ;;*****************************************************************
                                ;THE FOLLOWING SECTION TESTS STATE MUL.50 (-1 TIMES 3)
010624  012767  010632  170256  8$:     MOV     #64$,$LPERR     ;SETUP ERROR LOOP
010632  012700  177777          64$:    MOV     #177777,R0      ;PUT -1 (MULTIPLICAND) IN R0
                                                                ;R2 HAS MULTIPLIER (+3)
010636  000277                          SCC                     ;
010640  000250                          CLN                     ;CC'S=0111
                                                                ;AND OSCILLOSCOPE SYNC POINT
010642  070002                          MUL     R2,R0           ;EXECUTE INSTRUCTION UNDER TEST
010644  013767  177776  170334          MOV     @#PSW,$ERPSW    ;SAVE PSW
010652  010067  170276                  MOV     R0,$REG0        ;SAVE R0 & R1
010656  010167  170274                  MOV     R1,$REG1        ;FOR TYPEOUT
010662  012767  177777  170272          MOV     #-1,$TMP0       ;SAVE EXPECTED
010670  012767  177775  170266          MOV     #-3,$TMP1       ;VALUES
010676  020027  177777                  CMP     R0,#177777      ;DID R0 LOAD CORRECTLY?
010702  001402                          BEQ     9$              ;BRANCH IF YES
010704  104063                          ERROR   63              ;R0 DID NOT LOAD CORRECTLY
010706  000420                          BR      12$
010710  020127  177775          9$:     CMP     R1,#177775      ;DID R1 LOAD PROPERLY?
010714  001402                          BEQ     10$             ;BRANCH IF YES
010716  104064                          ERROR   64              ;R1 DID NOT LOAD CORRECTLY.
010720  000413                          BR      12$
010722  042767  177760  170256  10$:    BIC     #177760,$ERPSW  ;MASK OFF THE CC'S
010730  022767  000010  170250          CMP     #10,$ERPSW      ;DID THE CC'S LOAD PROPERLY?
010736  001404                          BEQ     12$             ;BRANCH IF YES
010740  012767  000010  170214  11$:    MOV     #10,$TMP0       ;PUT EXPECTED PSW IN $TMP0
010746  104065                          ERROR   65              ;BAD CC'S DUE TO STATE MUL.50
                                ;;*****************************************************************
                                ;THE FOLLOWING VERIFIES THAT C SETS IF MULTIPLICATION OVERFLOWS OR UNDER FLOWS
010750  012767  010756  170132  12$:    MOV     #63$,$LPERR     ;SETUP ERROR LOOP
010756  012700  077777          63$:    MOV     #77777,R0       ;PUT LARGEST POSITIVE NO. IN R0
                                                                ;R2 STILL CONTAINS +2
010762  000240                          NOP                     ;OSCILLOSCOPE SYNC POINT
010764  070002                          MUL     R2,R0           ;EXECUTE INSTRUCTION
```

```
010766  103401                                  BCS     13$             ;BRANCH IF C SET
010770  104066                                  ERROR   66              ;C DID NOT SET ON OVERFLOW
010772  012700  100000                  13$:    MOV     #BIT15,R0       ;PUT SMALLEST NEGATIVE NO IN R0
                                                                        ;R2 STILL CONTAINS +2
010776  070002                                  MUL     R2,R0           ;EXECUTE INSTRUCTION
011000  103401                                  BCS     TST13           ;;BRANCH IF C SET
011002  104067                                  ERROR   67              ;C DID NOT SET ON UNDERFLOW
                                        ;;*****************************************************************
                                        ;*TEST 13       MUL*DM1
                                        ;*
                                        ;*      FORK A SHOULD NOT FAIL.
                                        ;*
                                        ;*      FORK B WILL FAIL TO RSD.00 IF THE R(MUL:ASHC+MFP) FIELD OF
                                        ;*      THE INSTRUCTION DECODE ROM IS BAD.
                                        ;*      IF STATE MUL.00 FAILS THE RESULT WILL BE BAD.
                                        ;*
                                        ;*      ROM FLOW-1,175,60,102,266/246,226/206,310
                                        ;;*****************************************************************
011004  000004                          TST13:  SCOPE
011006  012767  011120  170160                  MOV     #TST14,$ESCAPE  ;SAVE START ADDRESS OF NEXT TEST
011014  012767  011120  170246                  MOV     #TST14,NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
011022  012706  001100                          MOV     #STACK,SP       ;INITIALIZE THE SP
011026  012737  011110  000010                  MOV     #1$,@#RESVEC    ;SETUP RESVEC
011034  012701  000002                          MOV     #2,R1           ;PUT MULTIPLICAND IN R1
011040  012702  001166                          MOV     #$TMP2,R2       ;PUT ADDRESS OF MULTIPLIER IN R2
011044  012712  000002                          MOV     #2,(R2)         ;PUT MULTIPLIER (+2) IN $TMP2
011050  005000                                  CLR     R0              ;ENSURE R0 CLEAR
011052  005067  170104                          CLR     $TMP0           ;STORE EXPECTED VALUE
011056  012767  000004  170100                  MOV     #4,$TMP1        ;OF R0 & R1
011064  000240                                  NOP                     ;OSCILLOSCOPE SYNC POINT
011066  070112                                  MUL     (R2),R1         ;EXECUTE INSTRUCTION UNDER TEST
011070  020127  000004                          CMP     R1,#4           ;DID MULTIPLY WORK?
011074  001411                                  BEQ     TST14           ;;BRANCH IF YES
011076  010067  170052                          MOV     R0,$REG0        ;SAVE R0 & R1
011102  010167  170050                          MOV     R1,$REG1        ;FOR TYPEOUT
011106  104070                                  ERROR   70              ;STATE MUL.00 FAILED
011110  012737  000012  000010          1$:     MOV     #12,@#RESVEC    ;RESTORE RESVEC
011116  104071                                  ERROR   71              ;FORK B FAILED

                                        ;;*****************************************************************
                                        ;*TEST 14       DIV*DM0
                                        ;*
                                        ;*      FORK A SHOULD NOT FAIL.
                                        ;*   SECTION 1
                                        ;*      THE FIRST SECTION HAS A ZERO DIVISOR. IF STATE DIV.00
                                        ;*      FAILS OR IRCF Z2(1) DOES NOT GET TO RACK E49 OR RACK
                                        ;*      E49(B1) IS STUCK LOW EXECUTION WILL GO FROM DIV.10 TO
                                        ;*      DIV.20. THIS WILL CAUSE THE DIVIDE TO ABORT (GO TO DVE.20)
                                        ;*      AFTER STATE DIV.60 AND THE C BIT WILL BE CLEAR.
                                        ;*
                                        ;*   SECTION 2
                                        ;*      THE NEXT SECTION DIVIDES 4 BY 2. IF RACK E49(B1) IS STUCK
                                        ;*      HIGH THE ALGORITHM WILL ABORT THINKING THAT THE DIVISOR
                                        ;*      IS ZERO. IF BEN04 FAILS THE DIVIDE WILL ABORT THINKING
                                        ;*      THAT THE DIVIDEND IS THE MOST NEGATIVE NUMBER.
                                        ;*      IF BEN16 FAILS THE DIVIDE WILL COMPLETE BUT R1 WILL CONTAIN
                                        ;*      155776. IF BEN05 FAILS THE ALGORITHM WILL ABORT THRU STATE
                                        ;*      DVE.20. IF BEN04 (AFTER DIV.70) FAILS R0 WILL END UP WITH 177777
                                        ;*      AND R1 WILL HAVE 177774. IF BEN03 FAILS R0 WILL END UP WITH
                                        ;*      20 AND R1 WILL HAVE 177774. IF BEN16 FAILS AFTER DVC.00 R0
                                        ;*      WILL HAVE 2 BUT R1 WILL HAVE 177776.
                                        ;*
                                        ;*   SECTION 3
                                        ;*      THE NEXT SECTION DIVIDES 6 BY 2 TO TEST BEN16*DR0(1).
                                        ;*      A FAILURE WILL LEAVE THE REMAINDER (R1)=2 INSTEAD OF ZERO.
                                        ;*
                                        ;*   SECTION 4
                                        ;*      THE NEXT SECTION DIVIDES 4 BY -2 TO TEST BEN15*SR15(1).
                                        ;*      IF THIS FAILS R0 WILL CONTAIN 27777 AND R1 WILL CONTAIN 4.
                                        ;*
                                        ;*   SECTION 5
                                        ;*      THE NEXT SECTION DIVIDES 1177777 BY 1 TO TEST BEN05*DIV QUIT.
                                        ;*      IF THIS FAILS R0 WILL CONTAIN 177777.
                                        ;*
                                        ;*   SECTION 6
                                        ;*      THE NEXT SECTION DIVIDES 1000000 B2 -2 TO TEST BEN05*DIV QUIT.
                                        ;*      THIS SECTION WILL ONLY FAIL IF GRAJ E5 (Z2(0)*LEFT SAVE (1)) IS BAD.
                                        ;*
                                        ;*   SECTION 7
                                        ;*      THE NEXT SECTION DIVIDES 100000000000 BY 2 TO TEST
                                        ;*      BEN04*NEGATIVE DIVIDEND.
                                        ;*
                                        ;*   SECTION 8
                                        ;*      THE NEXT SECTION DIVIDED 177776 177777 BY -1 TO TEST BEN05*DIV QUIT.
                                        ;*      THIS TEST WILL ONLY FAIL IF GRAJ E5(N(1)*SR15(1)) IS BAD.
                                        ;*
                                        ;*   SECTION 9
                                        ;*      THE NEXT SECTION DIVIDES -5 BY 2 TO ENSURE THAT THE REMAINDER IS
                                        ;*      STORED AS A NEGATIVE NUMBER.
                                        ;*
                                        ;*   SECTION 10
                                        ;*      THE NEXT SECTION DIVIDES -5 BY -2 TO TEST STATES DVC.20,DVC.40, & DVC.60
                                        ;*
                                        ;*   SECTION 11
                                        ;*      THE NEXT SECTION DIVIDES -2**16 BY 2**14 TO TEST STATE DVN.20
                                        ;*
                                        ;*   SECTION 12
                                        ;*      THE NEXT SECTION DIVIDES 100 000200 BY -177 TO TEST STATES
                                        ;*      DVD.00 AND DVD.10.
                                        ;;*************************************************************
 011120  000004                         TST14:  SCOPE
 011122  012767  012600  170140                 MOV     #TST15,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
                                        ;DIVISOR=0 SECTION (ALSO TESTS CONDITION CODE ROM FOR DIV)
 011130  005000                                 CLR     R0              ;PUT DIVIDEND IN
 011132  012701  000004                         MOV     #4,R1           ;R0 AND R1
 011136  005002                                 CLR     R2              ;MAKE DIVISOR=0
 011140  000270                                 SEN                     ;ENSURE N SET
                                                                        ;AND OSCILLOSCOPE SYNC POINT
 011142  071002                                 DIV     R2,R0           ;EXECUTE INSTRUCTION UNDER TEST
 011144  013767  177776  170034                 MOV     @#PSW,$ERPSW    ;SAVE PSW
 011152  042767  177760  170026                 BIC     #177760,$ERPSW  ;MASK OFF THE CC'S
 011160  022767  000007  170020                 CMP     #7,$ERPSW       ;DID THE CC'S COME OUT OK?
```

```
011166  001412                          BEQ     2$                      ;BRANCH IF YES
011170  022767  000002  170010          CMP     #2,$ERPSW               ;DID INSTR GO THROUGH DVE.20?
011176  001002                          BNE     3$                      ;BRANCH IF NO
011200  104075                          ERROR   75                      ;BEN02 FAILED
011202  000404                          BR      2$
011204  012767  000007  167750  3$:     MOV     #7,$TMP0                ;SAVE EXPECTED VALUE
011212  104076                          ERROR   76                      ;CC'S BAD IN STATE DVE.00
                                ;;**************************************************************
                                ;SECTION TWO (ALSO CHECKS CC LOAD OF STATE DVC.70)
011214  012767  011222  167666  2$:     MOV     #64$,$LPERR             ;SETTUP ERROR LOOP
011222  012702  000002          64$:    MOV     #2,R2                   ;PUT DIVISOR IN R2 (R0 & R1 HOLD DIVIDEND)
011226  000277                          SCC                             ;ENSURE CC'S SET
                                                                        ;AND OSCILLOSCOPE SYNC POINT
011230  071002                          DIV     R2,R0                   ;EXECUTE INSTRUCTION UNDER TEST
011232  013767  177776  167746          MOV     @#PSW,$ERPSW            ;SAVE PSW
011240  020027  000002                  CMP     R0,#2                   ;DID QUOTIENT COME OUT CORRECT?
011244  001042                          BNE     4$                      ;BRANCH IF NO
011246  005701                          TST     R1                      ;DID REMAINDER COME OUT CORRECT?
011250  001013                          BNE     5$                      ;BRANCH IF NO
011252  042767  177760  167726          BIC     #177760,$ERPSW          ;MASK OFF CC'S
011260  022767  000000  167720          CMP     #0,$ERPSW               ;ARE CC'S CORRECT?
011266  001513                          BEQ     7$                      ;BRANCH IF YES
011270  005067  167666          6$:     CLR     $TMP0                   ;SAVE EXPECTED VALUE
011274  104077                          ERROR   77                      ;BAD CC'S DUE TO STATE DVC.70
011276  000507                          BR      7$
011300  012767  000002  167654  5$:     MOV     #2,$TMP0                ;SAVE EXPECTED
011306  005067  167652                  CLR     $TMP1                   ;VALUES
011312  020127  177776                  CMP     R1,#177776              ;DID BEN16*DR0(0) FAIL?
011316  001002                          BNE     8$                      ;BRANCH IF NO
011320  104100                          ERROR   100                     ;RACK E50(B0) STUCK HIGH
011322  000475                          BR      7$
011324  020127  000004          8$:     CMP     R1,#4                   ;DID BEN04 FAIL?
011330  001002                          BNE     20$                     ;BRANCH IF NO
011332  104101                          ERROR   101                     ;BEN04 FAILED
011334  000470                          BR      7$
011336  010067  167612          20$:    MOV     R0,$REG0                ;SAVE R0 &
011342  010167  167610                  MOV     R1,$REG1                ;R1 FOR TYPEOUT
011346  104102                          ERROR   102                     ;QUOTIENT OK BUT REMAINDER BAD
011350  000462                          BR      7$
011352  012767  000002  167602  4$:     MOV     #2,$TMP0                ;SAVE EXPECTED
011360  005067  167600                  CLR     $TMP1                   ;VALUES
011364  022700  000020                  CMP     #20,R0                  ;DID BEN03 FAIL?
011370  001002                          BNE     9$                      ;BRANCH IF NO
011372  104103                          ERROR   103                     ;RACK E49(A1) STUCK LOW
011374  000450                          BR      7$
011376  022700  077777          9$:     CMP     #77777,R0               ;DID BEN04 (-DIV SUB) FAIL?
011402  001002                          BNE     10$                     ;BRANCH IF NO
011404  104104                          ERROR   104                     ;BEN04 STUCK TO DIV SUB
011406  000443                          BR      7$
011410  022700  177777          10$:    CMP     #-1,R0                  ;DID R0 GET A -1?
011414  001002                          BNE     13$                     ;BRANCH IF NO
011416  104110                          ERROR   110                     ;BEN04 (-N) FAILED
011420  000436                          BR      7$
011422  020027  000000          13$:    CMP     R0,#0                   ;DID R0 STAY 0?
011426  001406                          BEQ     11$                     ;BRANCH IF YES
011430  010067  167520                  MOV     R0,$REG0                ;SAVE R0 & R1
```

```
011434  010167  167516                   MOV     R1,$REG1          ;FOR TYPEOUT
011440  104105                           ERROR   105               ;INSTRUCTION FAILED
011442  000425                           BR      7$
011444  020127  155776          11$:     CMP     R1,#155776        ;DID BEN16*SR15(0) FAIL?
011450  001002                           BNE     12$               ;BRANCH IF NO
011452  104106                           ERROR   106               ;BEN16 FAILED TO DIV.50
011454  000420                           BR      7$
011456  020127  000004          12$:     CMP     R1,#4             ;DID R1 CHANGE?
011462  001406                           BEQ     14$               ;BRANCH IF NO
011464  010067  167464                   MOV     R0,$REG0          ;SAVE R0 & R1
011470  010167  167462                   MOV     R1,$REG1          ;FOR TYPEOUT
011474  104107                           ERROR   107               ;CAN'T DETERMINE FAILURE
011476  000407                           BR      7$
011500  032767  000001  167500  14$:     BIT     #BIT0,$ERPSW      ;DID BEN02 FAIL?
011506  001402                           BEQ     15$               ;BRANCH IF NO
011510  104111                           ERROR   111               ;BEN02 FAILED
011512  000401                           BR      7$
011514  104112                  15$:     ERROR   112               ;BEN05 FAILED
                                ;;*****************************************************************
                                ;SECTION THREE
011516  012767  011524  167364  7$:      MOV     #63$,$LPERR       ;SETUP ERROR LOOP
011524  005000                  63$:     CLR     R0                ;PUT DIVIDEND IN R0
011526  012701  000006                   MOV     #6,R1             ;PUT DIVIDEND IN R1 (DIVISOR=2 IN R2)
011532  000240                           NOP                       ;OSCILLOSCOPE SYNC POINT
011534  071002                           DIV     R2,R0             ;EXECUTE INSTRUCTION UNDER TEST
011536  020127  000002                   CMP     R1,#2             ;DID BEN16*DR0(1) FAIL?
011542  001001                           BNE     16$               ;BRANCH IF NO
011544  104113                           ERROR   113               ;RACK E50(B0) STUCK LOW
                                ;;*****************************************************************
                                ;SECTION FOUR (ALSO CHECK CC LOAD OF STATE DVC.90)
011546  012767  011554  167334  16$:     MOV     #62$,$LPERR       ;SETUP ERROR LOOP
011554  005000                  62$:     CLR     R0                ;
011556  012701  000004                   MOV     #4,R1             ;PUT DIVIDEND IN R0 & R1
011562  012702  177776                   MOV     #-2,R2            ;PUT DIVISOR IN R2
011566  012767  177776  167366           MOV     #-2,$TMP0         ;SAVE EXPECTED
011574  005067  167364                   CLR     $TMP1             ;VALUES
011600  000240                           NOP                       ;OSCILLOSCOPE SYNC POINT
011602  071002                           DIV     R2,R0             ;EXECUTE INSTRUCTION UNDER TEST
011604  013767  177776  167374           MOV     @#PSW,$ERPSW      ;SAVE CC'S
011612  020027  177776                   CMP     R0,#177776        ;DID DIVIDE WORK?
011616  001413                           BEQ     17$               ;BRANCH IF YES
011620  020027  027777                   CMP     R0,#27777         ;DID BEN16*SR15(1) FAIL?
011624  001002                           BNE     18$               ;BRANCH IF NO
011626  104114                           ERROR   114               ;RACK E64(B0) STUCK LOW
011630  000432                           BR      21$
011632  010067  167316          18$:     MOV     R0,$REG0          ;SAVE R0 & R1 FOR
011636  010167  167314                   MOV     R1,$REG1          ;TYPEOUT
011642  104115                           ERROR   115               ;EITHER STATE DVC.20 OR DVC.40 OR DVC.80
                                                                   ;OR DVC.90 FAILED
011644  000424                           BR      21$
011646  020127  000000          17$:     CMP     R1,#0             ;DID REMAINDER COME OUT CORRECT?
011652  001406                           BEQ     19$               ;BRANCH IF YES
011654  010067  167274                   MOV     R0,$REG0          ;SAVE R0 & R1
011660  010167  167272                   MOV     R1,$REG1          ;FOR TYPEOUT
011664  104116                           ERROR   116               ;QUOTIENT OK, REMAINDER BAD
011666  000413                           BR      21$
```

```
011670  042767  177760  167310  19$:    BIC     #177760,$ERPSW  ;MASK OF CC'S
011676  022767  000010  167302          CMP     #10,$ERPSW      ;DID CC'S COME OUT CORRECT?
011704  001404                          BEQ     21$             ;CONTINUE IF YES
011706  012767  000010  167246          MOV     #10,$TMPO       ;SAVE EXPECTED VALUE
011714  104117                          ERROR   117             ;BAD CC'S DUE TO STATE DVC.90

                                ;;*************************************************************
                                ;SECTION FIVE (DIV QUIT CAUSED BY N(0)*SR15(0))
011716  012767  011724  167164  21$:    MOV     #61$,$LPERR     ;SETUP ERROR LOOP
011724  012700  000002          61$:    MOV     #2,R0           ;SETUP R0 AND R1
011730  005001                          CLR     R1              ;TO CAUSE DIV QUIT ABORT
011732  012702  000001                  MOV     #1,R2           ;PUT DIVISOR IN R2
011736  000277                          SCC                     ;SETUP CC'S TO COMPLIMENT
011740  000242                          CLV                     ;OF EXPECTED VALUE
                                                                ;AND OSCILLOSCOPE SYNC POINT
011742  071002                          DIV     R2,R0           ;EXECUTE INSTRUCTION UNDER TEST
011744  013767  177776  167234          MOV     @#PSW,$ERPSW    ;SAVE CC'S
011752  020027  000002                  CMP     R0,#2           ;DID DIVIDE ABORT?
011756  001402                          BEQ     22$             ;BRANCH IF YES
011760  104120                          ERROR   120             ;DIV QUIT DID NOT GO LOW
011762  000413                          BR      23$
011764  042767  177760  167214  22$:    BIC     #177760,$ERPSW  ;MASK OFF CC'S
011772  022767  000002  167206          CMP     #2,$ERPSW       ;DID CC'S COME OUT OK?
012000  001404                          BEQ     23$             ;BRANCH IF YES
012002  012767  000002  167152          MOV     #2,$TMPO        ;STORE EXPECTED CC'S FOR TYPEOUT
012010  104121                          ERROR   121             ;CC'S BAD DUE TO EITHER DIV.30 OR DVE.20

                                ;;*************************************************************
                                ;SECTION SIX (DIV QUIT CAUSED BY SHFTR=0)
012012  012767  012020  167070  23$:    MOV     #60$,$LPERR     ;SETUP ERROR LOOP
012020  012700  000002          60$:    MOV     #2,R0           ;SETUP R0 & R1 TO
012024  005001                          CLR     R1              ;CAUSE DIV QUIT TO ABORT
012026  012702  177776                  MOV     #-2,R2          ;SETUP DIVISOR (-2)
012032  000240                          NOP                     ;OSCILLOSCOPE SYNC POINT
012034  071002                          DIV     R2,R0           ;EXECUTE INSTRUCTION UNDER TEST
012036  022700  000002                  CMP     #2,R0           ;DID DIVIDE ABORT?
012042  001401                          BEQ     24$             ;BRANCH IF YES
012044  104122                          ERROR   122             ;GRAJ E5 IS BAD (Z2(0)*LEFT SAVE (1))

                                ;;*************************************************************
                                ;SECTION SEVEN
012046  012767  012054  167034  24$:    MOV     #57$,$LPERR     ;SETUP ERROR LOOP
012054  012700  100000          57$:    MOV     #BIT15,R0       ;MAKE DIVIDEND
012060  005001                          CLR     R1              ;MOST NEGATIVE NUMBER
012062  012702  000001                  MOV     #1,R2           ;SETUP DIVISOR
012066  000257                          CCC                     ;SET CC'S TO COMPLIMENT OF EXPECTED
012070  000264                          SEZ                     ;OSCILLOSCOPE SYNC POINT
012072  071002                          DIV     R2,R0           ;EXECUTE INSTRUCTION UNDER TEST
012074  013767  177776  167104          MOV     @#PSW,$ERPSW    ;SAVE PSW
012102  020027  100000                  CMP     R0,#BIT15       ;DID DIVIDE ABORT?
012106  001402                          BEQ     25$             ;BRANCH IF YES
012110  104123                          ERROR   123             ;BEN04*N FAILED
```

```
012112  000413                                  BR      26$
012114  042767  177760  167064          25$:    BIC     #177760,$ERPSW  ;MASK OFF CC'S
012122  022767  000016  167056                  CMP     #16,$ERPSW      ;DID CC'S LOAD PROPERLY?
012130  001404                                  BEQ     26$             ;BRANCH IF YES
012132  012767  000016  167022                  MOV     #16,$TMP0       ;STORE EXPECTED VALUE
012140  104124                                  ERROR   124             ;CC'S DID NOT LOAD PROPERLY
                                        ;;*******************************************************************
                                        ;SECTION EIGHT (DIV QUIT CAUSED BY N(1)*SR15(1))
012142  012767  012150  166740          26$:    MOV     #56$,$LPERR     ;SETUP ERROR LOOP
012150  012700  177776                  56$:    MOV     #177776,R0      ;SETUP R0 & R1 TO
012154  012701  177777                          MOV     #177777,R1      ;CAUSE DIV QUIT TO ABORT
012160  012702  177777                          MOV     #177777,R2      ;SETUP DIVISOR (-1)
012164  000240                                  NOP                     ;OSCILLOSCOPE SYNC POINT
012166  071002                                  DIV     R2,R0           ;EXECUTE INSTRUCTION UNDER TEST
012170  020027  000001                          CMP     R0,#1           ;DID DIVIDE ABORT LEAVE R0=1
012174  001401                                  BEQ     27$             ;BRANCH IF YES
012176  000403                                  BR      35$
012200  020127  000001                  27$:    CMP     R1,#1           ;DID DIVIDE ABORT?
012204  001413                                  BEQ     28$             ;BRANCH IF YES
012206  012767  000001  166746          35$:    MOV     #1,$TMP0        ;STORE EXPECTED VALUE
012214  010067  166734                          MOV     R0,$REG0        ;SAVE R0
012220  012767  000001  166736                  MOV     #1,$TMP1
012226  010167  166724                          MOV     R1,$REG1        ;SAVE R1
012232  104125                                  ERROR   125             ;EITHER GRAJ E5 BAD OR MICROSTATE BAD
                                        ;;*******************************************************************
                                        ;SECTION NINE
012234  012767  012242  166646          28$:    MOV     #55$,$LPERR     ;SETUP ERROR LOOP
012242  012700  177777                  55$:    MOV     #-1,R0          ;PUT DIVIDEND IN
012246  012701  177773                          MOV     #-5,R1          ;R0 & R1
012252  012702  000002                          MOV     #2,R2           ;PUT DIVISOR IN R2
012256  012767  177776  166676                  MOV     #-2,$TMP0       ;SAVE EXPECTED VALUE
012264  012767  177777  166672                  MOV     #-1,$TMP1       ;SAVE EXPECTED VALUE
012272  000240                                  NOP                     ;OSCILLOSCOPE SYNC POINT
012274  071002                                  DIV     R2,R0           ;EXECUTE INSTRUCTION UNDER TEST
012276  010067  166652                          MOV     R0,$REG0        ;SAVE R0
012302  010167  166650                          MOV     R1,$REG1        ;SAVE R1
012306  020127  177777                          CMP     R1,#-1          ;IS REMAINDER CORRECT?
012312  001402                                  BEQ     29$             ;BRANCH IF YES
012314  104126                                  ERROR   126             ;REMAINDER BAD
012316  000404                                  BR      30$
012320  022700  177776                  29$:    CMP     #-2,R0          ;IS QUOTIENT CORRECT?
012324  001401                                  BEQ     30$             ;BRANCH IF YES
012326  104127                                  ERROR   127             ;QUOTIENT IS INCORRECT
                                        ;;*******************************************************************
                                        ;SECTION TEN
012330  012767  012336  166552          30$:    MOV     #54$,$LPERR     ;SETUP ERROR LOOP
012336  012700  177777                  54$:    MOV     #-1,R0          ;SETUP
012342  012701  177773                          MOV     #-5,R1          ;DIVIDEND
012346  012702  177776                          MOV     #-2,R2          ;AND DIVISOR
012352  012767  000002  166602                  MOV     #2,$TMP0        ;SAVE EXPECTED VALUES
012360  012767  177777  166576                  MOV     #-1,$TMP1       ;OF R0 & R1
012366  000240                                  NOP                     ;OSCILLOSCOPE SYNC POINT
012370  071002                                  DIV     R2,R0           ;EXECUTE INSTRUCTION UNDER TEST
012372  010067  166556                          MOV     R0,$REG0        ;SAVE R0
012376  010167  166554                          MOV     R1,$REG1        ;SAVE R1
012402  020027  000002                          CMP     R0,#2           ;IS QUOTIENT CORRECT?
```

```
012406  001402                               BEQ     31$                ;BRANCH IF YES
012410  104130                               ERROR   130                ;QUOTIENT BAD
012412  000404                               BR      32$
012414  020127  177777               31$:    CMP     R1,#-1             ;IS REMAINDER CORRECT
012420  001401                               BEQ     32$                ;BRANCH IF YES
012422  104131                               ERROR   131                ;REMAINDER INCORRECT (QUOT. OK)
                                     ;;*************************************************************
                                     ;SECTION ELEVEN
012424  012767  012432  166456       32$:    MOV     #53$,$LPERR        ;SETUP ERROR LOOP
012432  012700  177776               53$:    MOV     #177776,R0         ;SETUP DIVIDEND
012436  005001                               CLR     R1                 ;AND DIVISOR TO GET A
012440  012702  040000                       MOV     #40000,R2          ;QUOT OF -10 & REMAINDER=0
012444  012767  177770  166510               MOV     #-10,$TMP0         ;SAVE EXPECTED VALUE
012452  005067  166506                       CLR     $TMP1              ;SAVE EXPECTED VALUE
012456  000240                               NOP                        ;OSCILLOSCOPE SYNC POINT
012460  071002                               DIV     R2,R0              ;EXECUTE INSTRUCTION UNDER TEST
012462  010067  166466                       MOV     R0,$REG0           ;SAVE R0
012466  010167  166464                       MOV     R1,$REG1           ;SAVE R1
012472  020027  177770                       CMP     R0,#-10            ;IS QUOTIENT CORRECT?
012476  001402                               BEQ     33$                ;BRANCH IF YES
012500  104132                               ERROR   132                ;QUOTIENT IS BAD
012502  000404                               BR      34$
012504  020127  000000               33$:    CMP     R1,#0              ;IS REMAINDER CORRECT?
012510  001401                               BEQ     34$                ;BRANCH IF YES
012512  104133                               ERROR   133                ;REMAINDER BAD (QUOT. OK)
                                     ;;*************************************************************
                                     ;SECTION TWELVE
012514  012767  012522  166366       34$:    MOV     #52$,$LPERR        ;SETUP ERROR LOOP
012522  012700  000100               52$:    MOV     #100,R0            ;SETUP DIVIDEND
012526  012701  000200                       MOV     #200,R1            ;AND DIVISOR TO GENERATE
012532  012702  177601                       MOV     #-177,R2           ;DIVISION OVERFLOW
012536  000277                               SCC                        ;
012540  000242                               CLV                        ;AND OSCILLOSCOPE SYNC POINT
012542  071002                               DIV     R2,R0              ;EXECUTE INSTRUCTION UNDER TEST
012544  013767  177776  166434               MOV     @#PSW,$ERPSW       ;SAVE PSW
012552  042767  177760  166426               BIC     #177760,$ERPSW     ;MASK OFF CC'S
012560  022767  000002  166420               CMP     #2,$ERPSW          ;ARE CC'S CORRECT?
012566  001404                               BEQ     TST15              ;;TEST OK, GO TO NEXT TEST
012570  012767  000002  166364               MOV     #2,$TMP0           ;SAVE EXPECTED
012576  104134                               ERROR   134                ;BAD CC'S ON DIVISION OVERFLOW
                                     ;;*************************************************************
                                     ;*TEST 15        MTP*DMO
                                     ;*
                                     ;*       IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
                                     ;*
                                     ;*       THE ONLY OTHER POSSIBLE FAILURES WOULD BE IN ROM STATES
                                     ;*       MTP.00 OR MTP.10.
                                     ;*
                                     ;*       NOTE: THIS TEST ONLY TESTS THE CPU FUNCTIONS OF THIS INSTRUCTION.
                                     ;*       THE MEMORY MANAGEMENT TEST VERIFIES THE INTERMODE TRANSFER.
                                     ;*       AS FAR AS THE CPU IS CONCERNED THERE IS NO DIFFERENCE
                                     ;*       BETWEEN MTPI AND MTPD.
                                     ;*
                                     ;*       ROM FLOW-45,151,146,205
                                     ;;*************************************************************
012600  000004                       TST15:  SCOPE
```

```
012602  012767  012746  166364          MOV     #TST16,$ESCAPE  ;SAVE START ADDRESS OF NEXT TEST
012610  012767  012746  166452          MOV     #TST16,NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
012616  012737  012736  000010          MOV     #1$,@#RESVEC    ;SETUP RESVEC
012624  005037  177776                  CLR     @#PSW           ;ENSURE PREVIOUS MODE KERNAL
012630  012706  001074                  MOV     #1074,SP        ;SETUP THE SP
012634  012716  100000                  MOV     #BIT15,(SP)     ;SET SIGN BIT ON STACK
012640  005000                          CLR     R0              ;ENSURE R0 CLEAR
012642  000277                          SCC                     ;SET CC'S TO COMPLIMENT
012644  000250                          CLN                     ;OF EXPECTED VALUE (EXCEPT FOR C)
                                                                ;AND OSCILLOSCOPE SYNC POINT
012646  006600                          MTPI    R0              ;EXECUTE INSTRUCTION UNDER TEST
012650  013767  177776  166330          MOV     @#PSW,$ERPSW    ;SAVE PSW
012656  010067  166272                  MOV     R0,$REG0        ;SAVE R0 & SEC CC'S
012662  012767  100000  166272          MOV     #BIT15,$TMP0    ;SAVE EXPECTED VALUE
012670  100403                          BMI     2$              ;BRANCH IF R0 LOADED
012672  012700  100000                  MOV     #BIT15,R0       ;SAVE EXPECTED VALUE
012676  104135                          ERROR   135             ;R0 DID NOT LOAD
012700  022706  001076          2$:     CMP     #1076,SP        ;DID SP INCREMENT?
012704  001401                          BEQ     3$              ;BRANCH IF YES
012706  104136                          ERROR   136             ;SP DID NOT INCREMENT
012710  042767  177760  166270  3$:     BIC     #177760,$ERPSW  ;MASK OFF CC'S
012716  022767  000011  166262          CMP     #11,$ERPSW      ;DID CC'S COME OUT CORRECT?
012724  001410                          BEQ     TST16           ;;BRANCH IF YES
012726  012767  000011  166226          MOV     #11,$TMP0       ;SAVE EXPECTED VALUE
012734  104137                          ERROR   137             ;CC'S BAD (CC ROM)
012736  012737  000012  000010  1$:     MOV     #12,@#RESVEC    ;RESTORE RESVEC
012744  104140                          ERROR   140             ;FORK A FAILED
                                ;;*************************************************************
                                ;*TEST 16        MTP*DM1
                                ;*
                                ;*              IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
                                ;*              THIS WILL ONLY HAPPEN IF RACF E20(4) IS STUCK HIGH.
                                ;*
                                ;*              THIS TEST ENSURES STATE MTP.10 RELOADS THE
                                ;*              DR IF THE DESTINATION IS R6 AND THAT IT PUTS THE PC IN THE
                                ;*              DR IF THE DESTINATION FIELD IS R7.
                                ;*
                                ;*              ROM FLOW-45,151,146,111,155,312
                                ;*************************************************************
012746  000004                  TST16:  SCOPE
012750  012767  013120  166312          MOV     #TST17,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
012756  012737  013110  000010          MOV     #1$,@#RESVEC    ;SETUP RESVEC
012764  012706  001072                  MOV     #1072,SP        ;SETUP THE SP
012770  012716  100000                  MOV     #BIT15,(SP)     ;SET THE SIGN BIT ON THE STACK
012774  005066  000002                  CLR     2(SP)           ;ENSURE 1074 IS CLEAR
013000  000240                          NOP                     ;OSCILLOSCOPE SYNC POINT
013002  006616                          MTPI    (SP)            ;EXECUTE INSTRUCTION UNDER TEST
013004  005737  001074                  TST     @#1074          ;DID 1074 GET SIGN BIT SET?
013010  100413                          BMI     2$              ;BRANCH IF YES
013012  022706  001074                  CMP     #1074,SP        ;DID MTP.10 FAIL TO RELOAD THE DR?
013016  001002                          BNE     3$              ;BRANCH IF NO
013020  104141                          ERROR   141             ;STATE MTP.10 FAILED
013022  000406                          BR      2$
013024  010667  166124          3$:     MOV     SP,$REG0        ;SAVE SP
013030  012767  001074  166120          MOV     #1074,$REG1     ;SAVE EXPECTED VALUE
013036  104142                          ERROR   142             ;DON'T KNOW WHAT HAPPENED
```

```
013040  012767  013046  166042  2$:     MOV     #64$,$LPERR       ;SETUP ERROR LOOP
013046  012706  001074          64$:    MOV     #1074,SP          ;SETUP THE SP
013052  000240                          NOP                       ;OSCILLOSCOPE SYNC POINT
013054  006617                          MTPI    (PC)              ;EXECUTE INSTRUCTION UNDER TEST
013056  000000                  4$:     .WORD   0
013060  005767  177772                  TST     4$                ;DID 4$ GET BIT15 SET?
013064  100403                          BMI     5$                ;BRANCH IF YES
013066  005067  177764                  CLR     4$                ;RESTORE 4$
013072  104143                          ERROR   143               ;STATE MTP.10 DID NOT PUT PCB IN DR
013074  005067  177756          5$:     CLR     4$                ;RESTORE 4$
013100  012737  000012  000010          MOV     #12,@#RESVEC      ;RESTORE RESVEC
013106  000404                          BR      TST17             ;;GO TO NEXT TEST
013110  012737  000012  000010  1$:     MOV     #12,@#RESVEC      ;RESTORE RESVEC
013116  104144                          ERROR   144               ;FORK A FAILED
                                ;;*********************************************************************
                                ;*TEST 17               MFP*DMO
                                ;*
                                ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
                                ;*      IF ANYTHING ELSE FAILS, THEN A ROM STATE IS BAD.
                                ;*
                                ;*      ROM FLOW-46,304,250,222,300
                                ;;*********************************************************************
013120  000004                  TST17:  SCOPE
013122  012767  013306  166044          MOV     #TST20,$ESCAPE    ;SAVE START ADDRESS OF NEXT TEST
013130  012767  013306  166132          MOV     #TST20,NEXTTST    ;SAVE START ADDRESS OF NEXT TEST
013136  012767  013172  165742          MOV     #4$,$LPADR        ;SET UP LOOP
013144  012767  013172  165736          MOV     #4$,$LPERR        ;SETUP ERROR LOOP
013152  013767  177776  166010          MOV     @#PSW,$TMP3       ;SAVE PSW
013160  012746  000340                  MOV     #PR7,-(SP)        ;PUT NEW PSW ON STACK
013164  012746  013172                  MOV     #4$,-(SP)         ;PUT RETURN ADDR ON STACK
013170  000006                          RTT                       ;TURN T BIT OFF
013172  012737  013276  000010  4$:     MOV     #1$,@#RESVEC      ;SETUP RESVEC
013200  012706  001076                  MOV     #1076,SP          ;SETUP THE SP
013204  005016                          CLR     (SP)              ;ENSURE 1076 CLEAR
013206  005066  177776                  CLR     -2(SP)            ;ENSURE 1074 CLEAR
013212  012700  100000                  MOV     #BIT15,R0         ;SET THE SIGN BIT IN R0
013216  000277                          SCC                       ;SETUP CC'S TO COMPLIMENT
013220  000250                          CLN                       ;OF EXPECTED VALUE (EXCEPT FOR C)
                                                                  ;AND OSCILLOSCOPE SYNC POINT
013222  006500                          MFPI    R0                ;EXECUTE INSTRUCTION UNDER TEST
013224  013767  177776  165754          MOV     @#PSW,$ERPSW      ;SAVE THE PSW
013232  022706  001074                  CMP     #1074,SP          ;DID THE SP DECREMENT?
013236  001401                          BEQ     2$                ;BRANCH IF YES
013240  104145                          ERROR   145               ;STATE MFP.10 DID NOT DEC. THE SP
013242  005716                  2$:     TST     (SP)              ;DID R0 GET PUT ON THE STACK?
013244  100401                          BMI     3$                ;BRANCH IF YES
013246  104146                          ERROR   146               ;R0 DID NOT GET PUT ON THE STACK
013250  042767  177760  165730  3$:     BIC     #177760,$ERPSW    ;MASK OFF THE CC'S
013256  022767  000011  165722          CMP     #11,$ERPSW        ;DID THE CC'S SET CORRECTLY?
013264  001410                          BEQ     TST20             ;;BRANCH IF YES
013266  012767  000011  165666          MOV     #11,$TMP0         ;SAVE EXPECTED VALUE
013274  104147                          ERROR   147               ;INCORRECT CC'S
013276  012737  000012  000010  1$:     MOV     #12,@#RESVEC      ;RESTORE RESVEC
013304  104150                          ERROR   150               ;FORK A FAILED
                                ;;*********************************************************************
                                ;*TEST 20               MFP*DM2
```

```
                                        ;*
                                        ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
                                        ;*      IF FORK B FAILS EXECUTION WILL ALSO GO TO RSD.00 BUT THE DR
                                        ;*      WILL HAVE BEEN INCREMENTED. IF ANYTHING ELSE FAILS THEN
                                        ;*      STATE MFP.00 IS BAD.
                                        ;*
                                        ;*      ROM FLOW-2,175,66,250,222,300
                                        ;;*************************************************************
 013306  000004                         TST20:  SCOPE
 013310  012767  013446  165656                 MOV     #TST21,$ESCAPE  ;SAVE START ADDRESS OF NEXT TEST
 013316  012767  013446  165744                 MOV     #TST21,NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
 013324  012737  013426  000010                 MOV     #1$,@#RESVEC    ;SETUP RESVEC
 013332  012706  001076                         MOV     #1076,SP        ;SETUP THE SP
 013336  005016                                 CLR     (SP)            ;ENSURE WORDS ON
 013340  005066  177776                         CLR     -2(SP)          ;STACK CLEAR
 013344  012700  001164                         MOV     #$TMP1,R0       ;PUT ADDRESS OF $TMP1 IN R0
 013350  012710  100000                         MOV     #BIT15,(R0)     ;SET THE SIGN BIT IN $TMP1
 013354  000277                                 SCC                     ;SETUP CC'S TO COMPLIMENT OF
 013356  000250                                 CLN                     ;EXPECTED VALUE (EXCEPT FOR C)
                                                                        ;AND OSCILLOSCOPE SYNC POINT
 013360  006520                                 MFPI    (R0)+           ;EXECUTE INSTRUCTION UNDER TEST
 013362  013767  177776  165616                 MOV     @#PSW,$ERPSW    ;SAVE PSW
 013370  022706  001074                         CMP     #1074,SP        ;DID THE SP DECREMENT?
 013374  001401                                 BEQ     2$              ;BRANCH IF YES
 013376  104151                                 ERROR   151             ;STATE MFP.00 IS BAD
 013400  042767  177760  165600         2$:     BIC     #177760,$ERPSW  ;MASK OFF THE CC'S
 013406  022767  000011  165572                 CMP     #11,$ERPSW      ;DID THE CC'S SET CORRECTLY?
 013414  001414                                 BEQ     TST21           ;;BRANCH IF YES
 013416  012767  000011  165536                 MOV     #11,$TMP0       ;SAVE EXPECTED VALUE
 013424  104152                                 ERROR   152             ;INCORRECT CC'S DUE TO STATE MFP.00
 013426  012737  000012  000010         1$:     MOV     #12,@#RESVEC    ;RESTORE RESVEC
 013434  022700  001166                         CMP     #$TMP2,R0       ;DID R0 INCREMENT?
 013440  001401                                 BEQ     3$              ;BRANCH IF YES
 013442  104153                                 ERROR   153             ;FORK A FAILED
 013444  104154                         3$:     ERROR   154             ;FORK B FAILED
                                        ;;*************************************************************
                                        ;*TEST 21       BPT
                                        ;*
                                        ;*      FORK A SHOULD NOT FAIL.
                                        ;*      IF THE TRAP VECTOR LOGIC FAILS THE TRAP VECTOR WOULD
                                        ;*      COME OUT TO BE 4.
                                        ;*       THE ONLY OTHER FAILURE WOULD BE TRP.00.
                                        ;*       IF THIS STATE FAILS TO LOAD THE DR THE TRAP VECTOR WILL
                                        ;*      BE WHATEVER IS IN R3.  IF IT FAILS TO LOAD THE BR THE OLD
                                        ;*      PS WILL FAIL TO BE STACKED.
                                        ;;*************************************************************
 013446  000004                         TST21:  SCOPE
 013450  012767  013562  165516                 MOV     #TST22,$ESCAPE  ;SAVE START ADDRESS OF NEXT TEST
 013456  012767  013562  165604                 MOV     #TST22,NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
 013464  012737  013522  000014         4$:     MOV     #1$,@#BPTVEC    ;SETUP BPT VECTOR
 013472  012706  001100                         MOV     #STACK,SP       ;SETUP THE SP
 013476  012737  013542  000004                 MOV     #2$,@#ERRVEC    ;SETUP LOCATION 4
 013504  012737  013544  000024                 MOV     #3$,@#24        ;SETUP LOCATION 24
 013512  012703  000024                         MOV     #24,R3          ;SETUP R3
 013516  000237                                 SPL     7               ;SETUP PSW
 013520  000003                                 BPT                     ;EXECUTE INSTRUCTION UNDER TEST
```

```
013522  042737  177437  001076  1$:     BIC     #177437,@#1076   ;MASK OFF PRIORITIES OF OLD PS
013530  022737  000340  001076          CMP     #340,@#1076      ;DID OLD PS GET STACKED?
013536  001403                          BEQ     5$               ;BRANCH IF YES
013540  104160                          ERROR   160              ;STATE TRP.00 FAILED
013542  104161                  2$:     ERROR   161              ;TRAP VECTOR CAME UP BAD
013544  104162                  3$:     ERROR   162              ;STATE TRP.00 FAILED
013546  012737  032654  000004  5$:     MOV     #CPUSPUR,@#ERRVEC        ;RESTORE ERR VEC
013554  012737  032640  000014          MOV     #$RTRN,@#TBITVEC ;RESTORE T BIT VECTOR

                                ;;*************************************************************************
                                ;;*ALL THE LOGIC FOR (JMP+JSR)*DM0 HAS BEEN TESTED.
                                ;;*************************************************************************

                                ;;*************************************************************************
                                ;;*TEST 22          BIT TEST OF PIRQ REGISTER
                                ;;*
                                ;;*         IF ONE OF THE BLOCK LEVELS IS STUCK HIGH OR TMCB
                                ;;*         PS07(0)'S STUCK HIGH OR PDRD PRIORITY=0 IS STUCK HIGH,
                                ;;*         A PIRQ TRAP LOOP WILL OCCUR WHEN THAT PIR LEVEL IS ENABLED.
                                ;;*
                                ;;*         A COUNT PATTERN IS THEN RUN THRU THE REGISTER TO ENSURE THAT
                                ;;*         THE ENCODER FUNCTIONS PROPERLY.
                                ;;*************************************************************************
013562  000004                  TST22:  SCOPE
013564  012767  013766  165402          MOV     #TST23,$ESCAPE   ;SAVE START ADDRESS OF NEXT TEST
013572  012767  013766  165470          MOV     #TST23,NEXTTST   ;SAVE START ADDRESS OF NEXT TEST
013600  012706  001100                  MOV     #STACK,SP        ;INITIALIZE THE SP
013604  012767  013642  165274          MOV     #4$,$LPADR       ;SETUP LOOP ADR
013612  012767  013642  165270          MOV     #4$,$LPERR       ;SETUP ERROR LOOP
013620  032767  000020  165342          BIT     #BIT4,$TMP3      ;WAS T BIT ON?
013626  001405                          BEQ     4$               ;BRANCH IF NO
013630  012746  000360                  MOV     #360,-(SP)       ;PUT NEW PSW ON STACK
013634  012746  013642                  MOV     #4$,-(SP)        ;PUT RETURN ADDR ON STACK
013640  000006                          RTT                      ;TURN T BIT ON
013642  000237                  4$:     SPL     7                ;SET THE CPU PRIORITY AT 7.
013644  005067  165312                  CLR     $TMP0            ;SETUP COMPARISON LOCATION
013650  005037  177772                  CLR     @#PIRQ           ;CLEAR PIRQ REGISTER
013654  026737  165302  177772          CMP     $TMP0,@#PIRQ     ;DID PIRQ CLEAR?
013662  001035                          BNE     1$               ;BRANCH IF NO
013664  012700  000177                  MOV     #177,R0          ;SETUP ITTERATION COUNT
013670  012767  001042  165264          MOV     #1042,$TMP0      ;SETUP COMPARISON LOCATION
013676  012701  000002                  MOV     #2,R1            ;SETUP R1
013702  062737  001000  177772  2$:     ADD     #1000,@#PIRQ     ;START COUNT PATTERN
013710  026737  165246  177772          CMP     $TMP0,@#PIRQ     ;DID REGISTER SET CORRECT?
013716  001017                          BNE     1$               ;BRANCH IF NO
013720  120137  177773                  CMPB    R1,@#PIRQ+1      ;IS PIRQ READY TO GO TO NEXT LEVEL?
013724  001005                          BNE     3$               ;BRANCH IF NO
013726  062767  000042  165226          ADD     #42,$TMP0        ;INCREMENT ENCODED VALUES IN TEST LOC.
013734  005201                          INC     R1               ;SETUP R1 FOR ROTATE
013736  006101                          ROL     R1               ;SET R1 TO NEXT CHECK LEVEL
013740  062767  001000  165214  3$:     ADD     #1000,$TMP0      ;INC. PIRQ LEVEL IN TEST LOCATION
013746  077023                          SOB     R0,2$            ;CONTINUE COUNT
013750  005037  177772                  CLR     @#PIRQ           ;ENSURE PIRQ CLEAR
013754  000404                          BR      TST23            ;;GO TO NEXT TEST
013756  013767  177772  165232  1$:     MOV     @#PIRQ,$EPIRQ    ;SAVE PIRQ FOR TYPEOUT
013764  104163                          ERROR   163              ;PIRQ REG. FAILED
```

```
                                ;;*******************************************************************
                                ;*TEST 23          PIR LEVEL 1 INTERRUPT
                                ;*
                                ;*          IF BEN13 FAILS EXECUTION WOULD GO TO ONE OF THE FOLLOWING:
                                ;*          PUP.00, BRK.20, OR SER.00.
                                ;*          PUP.00 WOULD START THE POWER UP ROUTINE.
                                ;*          BRK.20 WOULD CAUSE A TRAP TO ZERO.
                                ;*          SER.00 WOULD PUT 4 IN THE SP AND PERFORM A RED ZONE TRAP.
                                ;*          IF TMCB PIRQ DOES NOT GET TO DAPE OR IF DAPE TV05*07 DOES NOT
                                ;*          GO HIGH OR DOES NOT GET THRU TO THE ALU A TRAP TO 4 WILL OCCUR.
                                ;*          IF THE INTERRUPT DOESN'T OCCUR AN ATTEMPT IS MADE TO ISOLATE THE
                                ;*          FAILURE.
                                ;;*******************************************************************
013766  000004                  TST23:  SCOPE
013770  012767  014336  165176          MOV     #TST24,$ESCAPE  ;SAVE START ADDRESS OF NEXT TEST
013776  012767  014336  165264          MOV     #TST24,NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
014004  012737  014274  000024          MOV     #10$,@#PWRVEC   ;SETUP
014012  012737  014326  000000          MOV     #12$,@#0        ;SETUP LOCATION 0
014020  012737  000340  000002          MOV     #PR7,@#2        ;PUT PRIORITY 7 IN 2
014026  012737  014302  000004          MOV     #11$,@#4        ;FAILURE TRAP VECTORS
014034  012737  014252  000240          MOV     #1$,@#PIRQVEC   ;SETUP PIRQ VECTOR
014042  032737  000020  177776          BIT     #BIT4,@#PSW     ;IS T BIT ON?
014050  001404                          BEQ     13$             ;BRANCH IF NO
014052  012737  000360  000242          MOV     #360,@#PIRQVEC+2 ;SET T BIT IN PIRQ VECTOR
014060  000403                          BR      14$             ;
014062  012737  000340  000242  13$:    MOV     #PR7,@#PIRQVEC+2  ;SETUP PIRQ VECTOR PSW
014070  012706  001100          14$:    MOV     #STACK,SP       ;SETUP THE SP
014074  000230                          SPL     0               ;SET PRIORITY LEVEL AT ZERO
014076  052737  001000  177772          BIS     #BIT9,@#PIRQ    ;SET LEVEL ONE
014104  005037  177772                  CLR     @#PIRQ          ;CLEAR LEVEL ONE
                                ;TRY TO GET INTERRUPT AT FET.01 INSTEAD OF TST.10.
014110  012737  014132  000240          MOV     #2$,@#PIRQVEC   ;SETUP PIRQ VEC
014116  012737  001000  177772          MOV     #BIT9,@#PIRQ    ;SET LEVEL ONE
014124  005037  177772                  CLR     @#PIRQ
014130  000403                          BR      3$              ;BRANCH IF NO INTERRUPT
014132  005037  177772          2$:     CLR     @#PIRQ          ;CLEAR LEVEL 1
014136  104164                          ERROR   164             ;STATE TST.10 HAS BAD BEN FIELD
                                ;TRY PIR LEVEL 4
014140  012737  014162  000240  3$:     MOV     #4$,@#PIRQVEC   ;SETUP PIRQ VECTOR
014146  052737  010000  177772          BIS     #BIT12,@#PIRQ   ;SET LEVEL 4
014154  005037  177772                  CLR     @#PIRQ
014160  000403                          BR      5$              ;BRANCN IF NO INTERRUPT
014162  005037  177772          4$:     CLR     @#PIRQ          ;CLEAR LEVEL 4
014166  104165                          ERROR   165             ;EITHER TMCB E62(1) IS BAD OR SOMETHING
                                                                ;IN THE HONOR PIR 1 LOGIC IS BAD.
                                ;TRY PIR 6
014170  012737  014212  000240  5$:     MOV     #6$,@#PIRQVEC   ;SETUP PIRQ VECTOR
014176  052737  040000  177772          BIS     #BIT14,@#PIRQ   ;SET LEVEL 6
014204  005037  177772                  CLR     @#PIRQ
014210  000403                          BR      7$              ;BRAHCN IF NO INTERRUPT
014212  005037  177772          6$:     CLR     @#PIRQ          ;
014216  104166                          ERROR   166             ;EITHER TMCB E51(9) OR E55(10-11) OR
                                                                ;E62 IS BAD OR TMCA INH BELOW BR6 IS
                                                                ;STUCK LOW
                                ;TRY PIR 7
014220  012737  014242  000240  7$:     MOV     #8$,@#PIRQVEC   ;SETUP PIRQ VECTOR
```

```
014226  052737  100000  177772                 BIS     #BIT15,@#PIRQ      ;SET LEVEL 7
014234  005037  177772                         CLR     @#PIRQ             ;CLEAR LEVEL 7
014240  000403                                 BR      9$                 ;BRANCH IF NO INTERRUPT
014242  005037  177772                 8$:     CLR     @#PIRQ
014246  104167                                 ERROR   167                ;TMCA ABOVE BR7 MIGHT BE STUCK LOW
014250  104170                         9$:     ERROR   170                ;TMCE BRQ CLOCK MIGHT BE STUCK LOW
014252  005037  177772                 1$:     CLR     @#PIRQ             ;CLEAR LEVEL 1
014256  012737  032654  000004                 MOV     #CPUSPUR,@#ERRVEC          ;RESTORE ERR VEC
014264  012737  036266  000024                 MOV     #$PWRDN,@#PWRVEC           ;RESTORE THE POWER VECTOR
014272  000421                                 BR      TST24              ;;GO TO NEXT TEST
014274  005037  177772                 10$:    CLR     @#PIRQ             ;CLEAR LEVEL 1
014300  104171                                 ERROR   171                ;BEN 13 FAILED
014302  005037  177772                 11$:    CLR     @#PIRQ
014306  012737  032654  000004                 MOV     #CPUSPUR,@#ERRVEC          ;RESTORE LOCATION 4
014314  012706  001100                         MOV     #STACK,SP          ;RESTORE THE SP
014320  005037  177766                         CLR     @#CPUERR           ;CLEAR ERROR REG
014324  104172                                 ERROR   172                ;BEN 13 FAILED OR TRAP VECTOR FAILED
014326  005037  177772                 12$:    CLR     @#PIRQ             ;CLEAR LEVEL 1
014332  104377                                 ERROR   377                ;EITHER TMCB E53 IS NOT GOING HIGH
014334  000454                                 454                        ;OR TMCB PIRQ DOES NOT GO LOW. (BEN 13 FAILURE)
                                       ;;*************************************************************************
                                       ;*TEST 24          PIR LEVEL 2 INTERRUPT
                                       ;*
                                       ;*      IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
                                       ;*      THIS WILL ONLY HAPPEN IF TMCB E63(2) IS BAD.
                                       ;*
                                       ;*      IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(2)
                                       ;*      IS BAD OR TMCB HONOR PIR2 IS BEING HELD HIGH.
                                       ;;*************************************************************************
014336  000004                         TST24:  SCOPE
014340  012767  014474  164722                 MOV     #TST25,NEXTTST     ;SAVE ADDRESS OF NEXT TEST
014346  012706  001100                         MOV     #STACK,SP          ;SETUP THE SP
014352  012737  014406  000000                 MOV     #1$,@#0            ;SETUP LOC. 0 TO CATCH BEN 13 FAILURE
014360  012737  014422  000240                 MOV     #2$,@#PIRQVEC      ;SETUP PIRQ VECTOR
014366  000231                                 SPL     1                  ;SET CPU PRIORITY TO 1
014370  052737  002000  177772                 BIS     #BIT10,@#PIRQ      ;SET LEVEL 2
014376  005037  177772                         CLR     @#PIRQ             ;CLEAR LEVEL 2
014402  104174                                 ERROR   174                ;PIR 2 DID NOT INTERRUPT
014404  000406                                 BR      2$
014406  005037  177772                 1$:     CLR     @#PIRQ             ;CLEAR LEVEL 2
014412  012737  032654  000004                 MOV     #CPUSPUR,@#ERRVEC          ;RESTORE LOCATION 4
014420  104175                                 ERROR   175                ;BEN 13 FAILED
                                       ;ENSURE PIR09 & 10 NOT SHORTED
014422  012767  014430  164460         2$:     MOV     #64$,$LPERR        ;SETUP ERROR LOOP
014430  005037  177772                 64$:    CLR     @#PIRQ             ;CLEAR LEVEL 2
014434  012737  014460  000240                 MOV     #3$,@#PIRQVEC      ;SETUP VECTOR
014442  000231                                 SPL     1                  ;SET LEVEL 1
014444  052737  001000  177772                 BIS     #BIT9,@#PIRQ       ;SET PIR 1
014452  005037  177772                         CLR     @#PIRQ             ;CLEAR PIR 1
014456  000406                                 BR      TST25              ;;GO TO NEXT TEST
014460  013767  177772  164530         3$:     MOV     @#PIRQ,$EPIRQ      ;SAVE PIRQ FOR TYPEOUT
014466  005037  177772                         CLR     @#PIRQ             ;CLEAR LEVEL 1
014472  104176                                 ERROR   176                ;PIR09 & 10 SHORTED
                                       ;;*************************************************************************
                                       ;*TEST 25          PIR LEVEL 3 INTERRUPT
                                       ;*
```

```
                                         ;*      IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
                                         ;*      THIS WILL ONLY HAPPEN IF TMCB E63(3) IS BAD.
                                         ;*
                                         ;*      IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(3)
                                         ;*      IS BAD OR TMCB HONOR PIR3 IS BEING HELD HIGH.
                                         ;;***************************************************************
014474  000004                           TST25:  SCOPE
014476  012767  014632  164564                   MOV     #TST26,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
014504  012706  001076                           MOV     #1076,SP        ;INITIALIZE THE SP
014510  012737  014560  000240                   MOV     #1$,@#PIRQVEC   ;SETUP PIRQ VECTOR
014516  012737  014544  000000                   MOV     #2$,@#0         ;SETUP LOCATION 0
014524  000232                                   SPL     2               ;SET CPU AT LEVEL 2
014526  052737  004000  177772                   BIS     #BIT11,@#PIRQ   ;SET LEVEL 3 PIR
014534  005037  177772                           CLR     @#PIRQ          ;CLEAR LEVEL 3
014540  104177                                   ERROR   177             ;INTERRUPT FAILED
014542  000406                                   BR      1$
014544  005037  177772                   2$:     CLR     @#PIRQ          ;CLEAR LEVEL 3
014550  012737  032654  000004                   MOV     #CPUSPUR,@#ERRVEC       ;RESTORE LOCATION 4
014556  104200                                   ERROR   200             ;BEN 13 FAILED
014560  012767  014566  164322           1$:     MOV     #64$,$LPERR     ;SETUP ERROR LOOP
014566  005037  177772                   64$:    CLR     @#PIRQ          ;CLEAR LEVEL 3
014572  012737  014616  000240                   MOV     #3$,@#PIRQVEC   ;SETUP PIRQ VECTOR
014600  000232                                   SPL     2               ;SET CPU PRIORITY AT 2
014602  052737  002000  177772                   BIS     #BIT10,@#PIRQ   ;ENABLE PIR2
014610  005037  177772                           CLR     @#PIRQ          ;CLEAR LEVEL 2
014614  000406                                   BR      TST26           ;;GO TO NEXT TEST
014616  013767  177772  164372           3$:     MOV     @#PIRQ,$EPIRQ   ;SAVE PIRQ
014624  005037  177772                           CLR     @#PIRQ          ;CLEAR IT
014630  104201                                   ERROR   201             ;LEVEL 2 INTERRUPT WHEN CPU LEVEL
                                                                         ;2 ENABLED.
                                         ;;***************************************************************
                                         ;*TEST 26        PIR LEVEL 4 INTERRUPT
                                         ;*
                                         ;*      IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
                                         ;*      THIS WILL ONLY HAPPEN IF TMCB E63(5) IS BAD.
                                         ;*
                                         ;*      IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(5)
                                         ;*      IS BAD OR TMCA HONOR PIR 4 IS BEING HELD HIGH.
                                         ;;***************************************************************
014632  000004                           TST26:  SCOPE
014634  012767  015002  164426                   MOV     #TST27,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
014642  012706  001100                           MOV     #STACK,SP       ;INITIALIZE THE SP
014646  012737  014720  000240                   MOV     #1$,@#PIRQVEC   ;SETUP PIRQ VEC
014654  012737  014704  000000                   MOV     #2$,@#0         ;SETUP LOCATION 0
014662  000233                                   SPL     3               ;SET CPU AT 3
014664  052737  010000  177772                   BIS     #BIT12,@#PIRQ   ;ENABLE PIR 4
014672  012737  032654  000004                   MOV     #CPUSPUR,@#ERRVEC
014700  104202                                   ERROR   202             ;INTERRUPT DID NOT OCCUR
014702  000406                                   BR      1$
014704  005037  177772                   2$:     CLR     @#PIRQ          ;CLEAR LEVEL 4
014710  012737  032654  000004                   MOV     #CPUSPUR,@#ERRVEC
014716  104203                                   ERROR   203             ;BEN 13 FAILED
014720  012767  014726  164162           1$:     MOV     #64$,$LPERR     ;SETUP ERROR LOOP
014726  005037  177772                   64$:    CLR     @#PIRQ          ;CLEAR LEVEL 4
014732  012737  032654  000004                   MOV     #CPUSPUR,@#ERRVEC       ;RESTORE LOCATION 4
014740  012737  014766  000240                   MOV     #3$,@#PIRQVEC   ;SETUP PIRQ VECTOR
```

```
014746  000233                                  SPL     3                       ;SET CPU AT LEVEL 3
014750  052737  004000  177772                  BIS     #BIT11,@#PIRQ           ;SET LEVEL 3
014756  005037  177772                          CLR     @#PIRQ                  ;CLEAR LEVEL 3
014762  000237                                  SPL     7
014764  000406                                  BR      TST27                   ;;GO TO NEXT TEST
014766  013767  177772  164222  3$:             MOV     @#PIRQ,$EPIRQ           ;SAVE PIRQ
014774  005037  177772                          CLR     @#PIRQ
015000  104204                                  ERROR   204                     ;LEVEL 3 INTERRUPT WHEN CPU LEVEL 3 ENABLED
                                        ;;*************************************************************
                                        ;*TEST 27       PIR LEVEL 5 INTERRUPT
                                        ;*
                                        ;*      IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
                                        ;*      THIS WILL ONLY HAPPEN IF TMCB E63(11) IS BAD.
                                        ;*
                                        ;*      IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(11)
                                        ;*      IS BAD OR TMCA HONOR PIR 5 IS BEING HELD HIGH.
                                        ;;*************************************************************
015002  000004                          TST27:  SCOPE
015004  012767  015156  164256                  MOV     #TST30,NEXTTST          ;SAVE ADDRESS OF NEXT TEST
015012  012706  001100                          MOV     #STACK,SP               ;INITIALIZE THE SP
015016  012737  015074  000240                  MOV     #1$,@#PIRQVEC           ;SETUP THE PIRQ VECTOR
015024  012737  015060  000000                  MOV     #2$,@#0                 ;SETUP LOCATION 0
015032  000234                                  SPL     4                       ;SET CPU ST LEVEL 4
015034  052737  020000  177772                  BIS     #BIT13,@#PIRQ           ;SET LEVEL 5
015042  005037  177772                          CLR     @#PIRQ                  ;CLEAR LEVEL 5
015046  012737  032654  000004                  MOV     #CPUSPUR,@#ERRVEC               ;RESTORE LOCATION 4
015054  104205                                  ERROR   205                     ;INTERRUPT DID NOT OCCUR
015056  000406                                  BR      1$
015060  005037  177772                  2$:     CLR     @#PIRQ                  ;CLEAR LEVEL 5
015064  012737  032654  000004                  MOV     #CPUSPUR,@#ERRVEC               ;RESTORE LOCATION 4
015072  104206                                  ERROR   206                     ;BEN 13 FAILED
015074  012767  015102  164006  1$:             MOV     #64$,$LPERR             ;SETUP ERROR LOOP
015102  012737  032654  000004  64$:            MOV     #CPUSPUR,@#ERRVEC               ;RESTORE LOCATION 4
015110  005037  177772                          CLR     @#PIRQ                  ;CLEAR LEVEL 5
015114  012737  015142  000240                  MOV     #3$,@#PIRQVEC           ;SETUP PIRQ VECTOR
015122  000234                                  SPL     4                       ;SET CPU AT LEVEL 4
015124  012737  010000  177772                  MOV     #BIT12,@#PIRQ           ;SET LEVEL 4
015132  005037  177772                          CLR     @#PIRQ                  ;CLEAR LEVEL 4
015136  000237                                  SPL     7
015140  000406                                  BR      TST30                   ;;GO TO NEXT TEST
015142  013767  177772  164046  3$:             MOV     @#PIRQ,$EPIRQ           ;SAVE PIRQ
015150  005037  177772                          CLR     @#PIRQ                  ;CLEAR LEVELS
015154  104207                                  ERROR   207                     ;LEVEL 4 INT. WHEN CPU LEVEL 4 ENABLED
                                        ;;*************************************************************
                                        ;*TEST 30       PIR LEVEL 6 INTERRUPT
                                        ;*
                                        ;*      IF BEN13 FAILS EXECUTION WILL GO TO BRK.20.
                                        ;*
                                        ;*      THIS WILL ONLY HAPPEN IF EITHER TMCB E63(12)
                                        ;*      IS BAD, OR E61(1) IS BAD.
                                        ;*
                                        ;*      IF THE INTERRUPT DOES NOT OCCUR LEVEL 7 IS TRYED, TO TRY
                                        ;*      AND ISOLATE THE FAILURE BEFORE TMCB E55(9-8).
                                        ;;*************************************************************
015156  000004                          TST30:  SCOPE
015160  012767  015356  164102                  MOV     #TST31,NEXTTST          ;SAVE ADDRESS OF NEXT TEST
```

```
015166  012706  001100                    MOV     #STACK,SP         ;INITIALIZE THE SP
015172  012737  015274  000240            MOV     #1$,@#PIRQVEC     ;SETUP THE PIRQ VECTOR
015200  012737  015260  000000            MOV     #2$,@#0           ;SETUP LOCATION 0
015206  000235                            SPL     5                 ;SET THE CPU AT LEVEL 5
015210  052737  040000  177772            BIS     #BIT14,@#PIRQ     ;SET LEVEL 6
015216  012737  032654  000004            MOV     #CPUSPUR,@#ERRVEC       ;RESTORE LOCATION 4
015224  012737  015250  000240            MOV     #3$,@#PIRQVEC     ;SETUP THE PIRQ VECTOR
015232  012737  100000  177772            MOV     #BIT15,@#PIRQ     ;SET LEVEL 7
015240  005037  177772                    CLR     @#PIRQ            ;CLEAR LEVEL 7
015244  104210                            ERROR   210               ;FAILURE IS AFTER TMCB E70
015246  000412                            BR      1$
015250  005037  177772            3$:     CLR     @#PIRQ            ;CLEAR LEVEL 7
015254  104211                            ERROR   211               ;FAILURE IS IN TMCB E70 OR BEFORE
015256  000406                            BR      1$
015260  012737  032654  000004    2$:     MOV     #CPUSPUR,@#ERRVEC       ;RESTORE LOCATION 4
015266  005037  177772                    CLR     @#PIRQ            ;CLEAR LEVEL 6
015272  104212                            ERROR   212               ;BEN 13 FAILED
015274  012767  015302  163606    1$:     MOV     #64$,$LPERR       ;SETUP ERROR LOOP
015302  005037  177772            64$:    CLR     @#PIRQ            ;CLEAR LEVEL 6
015306  012737  032654  000004            MOV     #CPUSPUR,@#ERRVEC       ;RESTORE LOCATION 4
015314  012737  015342  000240            MOV     #4$,@#PIRQVEC     ;SETUP PIRQ VECTOR
015322  000235                            SPL     5                 ;SET PRIORITY AT 5
015324  012737  020000  177772            MOV     #BIT13,@#PIRQ     ;SET LEVEL 5
015332  005037  177772                    CLR     @#PIRQ            ;CLEAR LEVEL 5
015336  000237                            SPL     7
015340  000406                            BR      TST31             ;;GO TO NEXT TEST
015342  013767  177772  163646    4$:     MOV     @#PIRQ,$EPIRQ     ;SAVE PIRQ
015350  005037  177772                    CLR     @#PIRQ            ;CLEAR LEVEL 5
015354  104213                            ERROR   213               ;LEVEL 5 INTERRUPT WHEN CPU 5 ENABLED
                                  ;;*****************************************************************
                                  ;*TEST 31         PIR LEVEL 7 INTERRUPT
                                  ;*
                                  ;*        IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
                                  ;*        THIS WILL ONLY HAPPEN IF TMCB E63(6) IS BAD.
                                  ;*
                                  ;*        IF THE INTERRUPT DOES NOT OCCUR THEN EITHER TMCB E70(6)
                                  ;*        IS BAD OR TMCA HONOR PIR7 IS BEING HELD HIGH.
                                  ;;*****************************************************************
015356  000004                    TST31:  SCOPE
015360  012767  015530  163702            MOV     #TST32,NEXTTST    ;SAVE ADDRESS OF NEXT TEST
015366  012706  001100                    MOV     #STACK,SP         ;INITIALIZE THE SP
015372  012737  015450  000240            MOV     #1$,@#PIRQVEC     ;SETUP THE PIRQ VECTOR
015400  012737  015434  000000            MOV     #2$,@#0           ;SETUP LOCATION 0
015406  000236                            SPL     6                 ;SET PRIORITY AT LEVEL 6
015410  052737  100000  177772            BIS     #BIT15,@#PIRQ     ;SET LEVEL 7
015416  005037  177772                    CLR     @#PIRQ            ;CLEAR LEVEL 7
015422  012737  032654  000004            MOV     #CPUSPUR,@#ERRVEC       ;RESTORE LOCATION 4
015430  104214                            ERROR   214               ;LEVEL 7 DID NOT INTERRUPT
015432  000406                            BR      1$
015434  005037  177772            2$:     CLR     @#PIRQ            ;CLEAR LEVEL 7
015440  012737  032654  000004            MOV     #CPUSPUR,@#ERRVEC       ;RESTORE LOCATION 4
015446  104215                            ERROR   215               ;BEN 13 FAILED
015450  012767  015456  163432    1$:     MOV     #64$,$LPERR       ;SETUP ERROR LOOP
015456  012737  032654  000004    64$:    MOV     #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
015464  012737  015514  000240            MOV     #3$,@#PIRQVEC     ;SETUP PIRQ VECTOR
015472  005037  177772                    CLR     @#PIRQ            ;CLEAR LEVEL 7
```

```
015476  000236                                  SPL     6                       ;SET LEVEL 6 IN CPU
015500  012737  040000  177772                  MOV     #BIT14,@#PIRQ           ;SET PIR 6
015506  005037  177772                          CLR     @#PIRQ                  ;CLEAR PIR 6
015512  000406                                  BR      TST32                   ;;GO TO NEXT TEST
015514  013767  177772  163474  3$:             MOV     @#PIRQ,$EPIRQ
015522  005037  177772                          CLR     @#PIRQ                  ;CLEAR LEVEL 6
015526  104216                                  ERROR   216                     ;LEVEL 6 INT. WHEN CPU AT 6

                                        ;;*****************************************************************
                                        ;*TEST 32           UNIBUS TIMEOUT
                                        ;*
                                        ;*              IF TMCC ABORT DOES NOT GO HIGH OR DOES NOT GET TO RACA
                                        ;*              OR IF RACA ZAP DOES NOT GO LOW THE PROCESSOR WILL NOT TRAP TO 4.
                                        ;*
                                        ;*              IF BEN06 FAILS THE STACKED PC WILL BE 160000 INSTEAD OF 1$-2.
                                        ;*
                                        ;*              IF BEN 13 FAILS EITHER TMCC AERF(1) L IS NOT GOING LOW
                                        ;*              OR TMCB E53(11) IS BAD.
                                        ;*              A TEST IS THEN MADE TO ENSURE THAT TMCC PRIORITY CLEAR GOES LOW.
                                        ;;*****************************************************************
015530  000004                          TST32:  SCOPE
015532  012767  016650  163530                  MOV     #TST33,NEXTTST          ;SAVE ADDRESS OF NEXT TEST
015540  012737  015632  000004                  MOV     #2$,@#ERRVEC            ;SETUP TIMEOUT VECTOR
015546  032737  000020  177776                  BIT     #BIT4,@#PSW             ;IS T BIT ON?
015554  001404                                  BEQ     19$                     ;BRANCH IF NO
015556  012737  000360  000006                  MOV     #360,@#ERRVEC+2         ;SETUP ERROR VEC FOR T BIT
015564  000403                                  BR      17$                     ;
015566  012737  000340  000006  19$:            MOV     #PR7,@#ERRVEC+2         ;SETUP ERROR VEC WITHOUT T BIT
015574  012767  015602  163306  17$:            MOV     #12$,$LPERR             ;SETUP ERROR LOOP
015602  012706  001100                  12$:    MOV     #STACK,SP               ;INITIALIZE THE SP
                                        ;
                                        ;EXECUTE A TIMEOUT ON A DATI
015606  013700  160000                          MOV     @#160000,R0             ;EXECUTE TIMEOUT ON DATI
                                        ;FAILURE-DID NOT TIMEOUT OR AERF DID NOT GO LOW
015612  032737  000020  177766                  BIT     #BIT4,@#CPUERR          ;DID TIMEOUT FLAG SET?
015620  001002                                  BNE     1$                      ;BRANCH IF YES
015622  104217                                  ERROR   217                     ;DID NOT TIMEOUT
015624  000423                                  BR      7$
015626  104220                          1$:     ERROR   220                     ;BEN 13 FAILED
015630  000421                                  BR      7$
015632  022716  160000                  2$:     CMP     #160000,(SP)            ;DID 160000 GET STACKED?
015636  001003                                  BNE     16$                     ;BRANCH IF NO
015640  104377                                  ERROR   377                     ;EITHER PS RESTORE STUCK HIGH
                                                                                ;OR RACK E63(B0) BAD
015642  000455                                  455
015644  000413                                  BR      7$
015646  012737  015674  000004  16$:            MOV     #7$,@#ERRVEC            ;SETUP TIMEOUT VECTOR
015654  012767  015662  163226                  MOV     #13$,$LPERR             ;SETUP ERROR LOOP
015662  012706  001100                  13$:    MOV     #STACK,SP               ;INITIALIZE THE SP
                                        ;
                                        ;EXECUTE A TIMEOUT ON A DATO
015666  010037  160000                          MOV     R0,@#160000             ;EXECUTE TIMEOUT ON DATO
015672  104221                                  ERROR   221                     ;DID NOT TIMEOUT
                                        ;PRIORITY CLEAR ON ABORT
                                        ;    ENSURES PIR VECTOR DOES NOT COME IN ON ABORT
015674  000237                          7$:     SPL     7                       ;ENSURE CPU AT LEVEL 7
```

```
015676  012767  015746  163204          MOV     #15$,$LPERR         ;SETUP THE ERROR LOOP
015704  012737  015766  000004          MOV     #11$,@#ERRVEC       ;SETUP ERRVEC
015712  012737  015774  000240          MOV     #9$,@#PIRQVEC       ;SETUP PIRQ VECTOR
015720  032737  000020  177776          BIT     #BIT4,@#PSW         ;IS T BIT ON?
015726  001404                          BEQ     18$                 ;BRANCH IF NO
015730  012737  000360  000242          MOV     #360,@#PIRQVEC+2 ;SETUP PIRQ PSW
015736  000403                          BR      15$
015740  012737  000340  000006  18$:    MOV     #PR7,@#ERRVEC+2 ;SETUP NEW PSW
015746  012706  001100          15$:    MOV     #STACK,SP           ;INITIALIZE THE SP
015752  052737  100000  177772          BIS     #BIT15,@#PIRQ       ;SET PIR LEVEL 7
015760  000236                          SPL     6
015762  005237  160000                  INC     @#160000            ;EXECUTE REFERENCE TO BAD ADDRESS
015766  005037  177772          11$:    CLR     @#PIRQ              ;CLEAR PIRQ REGISTER
015772  000404                          BR      10$                 ;SKIP OVER ERROR CALL
015774  005037  177772          9$:     CLR     @#PIRQ              ;CLEAR PIRQ REG
016000  104377                          ERROR   377                 ;TMCC PRIORITY CLEAR DID NOT GO LOW
016002  000435                          435
016004  012737  032654  000004  10$:    MOV     #CPUSPUR,@#ERRVEC   ;RESTORE ERRVEC
                                ;CPU ERROR REGISTER BIT 4 TEST
016012  022737  000020  177766          CMP     #BIT4,@#CPUERR      ;DID CPU ERROR REG TIMEOUT BIT SET?
016020  001407                          BEQ     5$                  ;BRANCH IF YES
016022  013767  177766  163124          MOV     @#CPUERR,$REG0      ;SAVE REG FOR TYPEOUT
016030  012767  000020  163124          MOV     #BIT4,$TMP0         ;SAVE EXPECTED VALUE
016036  104255                          ERROR   255                 ;CPU ERROR REG FAILED TO SET
016040  005037  177766          5$:     CLR     @#CPUERR            ;CLEAR OUT BIT 4
016044  022737  000000  177766          CMP     #0,@#CPUERR         ;DID REGISTER CLEAR?
016052  001401                          BEQ     PDINTE              ;;BRANCH IF YES
016054  104256                          ERROR   256                 ;CPU ERROR REG DOES NOT CLEAR
                                ;;*******************************************************************
                                .SBTTL  PERIPHERAL DETERMINATOR & INTERRUPT ENABLE ROUTINES
                                ;*      THIS SECTION OF CODE TRYS TO FIND A DEVICE ON BR5 AND BR6.
                                ;*      WHEN IT FINDS A DEVICE IT PUTS THE ADDRESS OF THAT DEVICES
                                ;*      SUBROUTINE IN A LOCATION.
                                ;*
                                ;*      THE CODE TO INITIATE AN INTERRUPT SEQUENCE ON CERTAIN
                                ;*      DEVICES IS ALSO HERE.  WHEN A TEST REQUIRES AN INTERRUPT ON
                                ;*      A CERTAIN LEVEL IT LOOKS AT INTERX TO DETERMINE IF A
                                ;*      DEVICE IS AVAILABLE AND IF SO DOES A JSR TO THAT DEVICES
                                ;*      INTERRUPT ENABLE ROUTINE.
                                ;;*******************************************************************
016056                          PDINTE:
016056  012767  016650  163110          MOV     #TST33,$ESCAPE      ;SAVE START ADDRESS OF NEXT TEST
016064  012767  016650  163176          MOV     #TST33,NEXTTST      ;SAVE START ADDRESS OF NEXT TEST
016072  005767  163002                  TST     $PASS               ;IS THIS FIRST PASS?
016076  001004                          BNE     1$                  ;BRANCH IF NO
016100  032737  040000  177570          BIT     #SW14,@#SWR         ;IS SWITCH 14 ON?
016106  001402                          BEQ     2$                  ;BRANCH IF NO
016110  000177  163154          1$:     JMP     @NEXTTST            ;GO TO NEXT TEST
016114  012706  001100          2$:     MOV     #STACK,SP           ;INITIALIZE THE SP
016120  012737  016134  000004          MOV     #3$,@#ERRVEC        ;SETUP ERROR VECTOR
016126  005777  163106                  TST     @RSCS1              ;IS RS AVAILABLE?
016132  000430                          BR      6$                  ;YES
016134  012737  016150  000004  3$:     MOV     #4$,@#ERRVEC        ;SETUP ERROR VECTOR
016142  005777  163076                  TST     @RPCS1              ;IS RP AVAILABLE?
016146  000431                          BR      7$                  ;YES
016150  012737  016164  000004  4$:     MOV     #5$,@#ERRVEC        ;SETUP ERROR VECTOR
```

```
016156  005777  163072                          TST     @TMCS1          ;IS TM AVAILABLE?
016162  000432                                  BR      8$              ;YES
016164  012737  016274  000004  5$:             MOV     #10$,@#ERRVEC   ;SETUP ERROR VECTOR
016172  005777  163052                          TST     @RKCS1          ;IS RK AVAILABLE?
016176  016767  163046  163024                  MOV     RKCS1,INT5ST    ;YES,SAVE RK STATUS
016204  016767  163042  163014                  MOV     RKVEC,INT5VEC   ;SAVE RK VECTOR
016212  000424                                  BR      9$              ;EXIT
016214  016767  163020  163006  6$:             MOV     RSCS1,INT5ST    ;SAVE RS STATUS
016222  016767  163014  162776                  MOV     RSVEC,INT5VEC   ;SAVE RS VECTOR
016230  000415                                  BR      9$              ;EXIT
016232  016767  163006  162770  7$:             MOV     RPCS1,INT5ST    ;SAVE RP STATUS
016240  016767  163002  162760                  MOV     RPVEC,INT5VEC   ;SAVE RP VECTOR
016246  000406                                  BR      9$              ;EXIT
016250  016767  163000  162752  8$:             MOV     TMCS1,INT5ST    ;SAVE TM STATUS
016256  016767  162774  162742                  MOV     TMVEC,INT5VEC   ;SAVE TM VECTOR
016264  012767  016464  162732  9$:             MOV     #INT5SU,INTER5  ;SAVE ADDRESS OF INTER 5 SUBROUTINE
016272  000406                                  BR      LEVE6           ;GO CHECK LEVEL 6
016274  032737  000440  177570  10$:            BIT     #440,@#SWR      ;SWITCH 8 OR 5 ON?
016302  001002                                  BNE     LEVE6           ;BRANCH IF YES
016304  104400  072420                          TYPE    ,EM710
                                ;
                                ;FIND OUT WHAT IS AVAILABLE ON LEVEL 6
016310  012737  016360  000004  LEVE6:  MOV     #1$,@#ERRVEC    ;SETUP LOCATION 4
016316  005777  162736                          TST     @LKSTAT         ;IS LINE CLOCK AVAILABLE?
016322  012767  016530  162702                  MOV     #$KW11L,INTER6  ;PUT ADDRESS OF KW11-L SUBROUTINE IN STORAGE
016330  016767  162726  162676                  MOV     LKVEC,INT6VEC   ;SAVE BR 6 INTERR VEC
016336  016767  162716  162672                  MOV     LKSTAT,INT6ST   ;SAVE ADDRESS OF BR 6 STATUS
016344  012737  032654  000004                  MOV     #CPUSPUR,@#ERRVEC
016352  005037  177766                          CLR     @#CPUERR        ;ENSURE TIMEOUT BIT CLEAR
016356  000534                                  BR      TST33           ;;PERIPHERAL DETERMINATOR FINISHED
016360  104400  073107          1$:             TYPE    ,EM725
                                ;
                                ;LINE CLOCK NOT AVAILABLE, SEE IF PROGRAMMABLE CLOCK AVAILABLE
016364  012737  016434  000004                  MOV     #2$,@#ERRVEC    ;SETUP LOCATION 4
016372  005777  162666                          TST     @PLKSTAT        ;IS PROGRAMMABLE AVAILABLE?
016376  012767  016574  162626                  MOV     #$KW11P,INTER6  ;YES, PUT ADDRESS OF KW11-P SUBROUTINE IN STORAGE
016404  016767  162656  162622                  MOV     PLKVEC,INT6VEC  ;SAVE BR 6 INTERR VEC
016412  016767  162646  162616                  MOV     PLKSTAT,INT6ST  ;SAVE ADDRESS OF BR 6 STATUS
016420  005037  177766                          CLR     @#CPUERR        ;ENSURE TIMEOUT BIT CLEAR
016424  012737  032654  000004                  MOV     #CPUSPUR,@#ERRVEC       ;RESTORE ERROR VECTOR
016432  000506                                  BR      TST33           ;;PERIPHERAL DETERMINATOR FINISHED
016434  012737  032654  000004  2$:             MOV     #CPUSPUR,@#ERRVEC   ;RESTORE LOCATION 4
016442  005037  177766                          CLR     @#CPUERR        ;ENSURE TIMEOUT BIT CLEAR
016446  032737  000500  177570                  BIT     #500,@#SWR      ;SWITCH 6 OR 8 ON?
016454  001002                                  BNE     3$              ;BRANCH IF YES
016456  104400  072440                          TYPE    ,EM711          ;TYPE MESSAGE(NO DEVICE AVAILABLE)
016462                          3$:
016462  000472                                  BR      TST33           ;;PERIPHERAL DETERMINATOR FINISHED
                                ;THIS CODE SETS UP THE DEVICE ON LEVEL 5 TO INTERRUPT
                                ;AND IS CALLED BY A JSR PC,@INTER5
016464  011600                  INT5SU: MOV     (SP),R0         ;SAVE RETURN PC
016466  012777  016520  162532                  MOV     #ENDBR5,@INT5VEC  ;SETUP LEVEL 5 VECTOR
016474  005001                                  CLR     R1              ;SETUP WAIT COUNTER
016476  012777  000311  162524                  MOV     #311,@INT5ST    ;SET LEVEL 5 INTERRUPT
016504  005201                  2$:             INC     R1              ;WAIT FOR
016506  001376                                  BNE     2$              ;INTERRUPT
```

```
016510  005077  162514                   CLR     @INT5ST         ;CLEAR INTERRUPT FLAG
016514  062700  000002                   ADD     #2,R0           ;ADJUST RETURN PC
016520  005077  162504          ENDBR5:  CLR     @INT5ST         ;CLEAR LEVEL 5 STATUS
016524  010046                           MOV     R0,-(SP)        ;PUT RETURN PC ON STACK
016526  000207                           RTS     PC              ;RETURN

                                ;
                                ;THIS CODE SETS UP THE KW11-L TO INTERRUPT AND IS CALLED BY A JSR PC,@INTER6
016530  011600                  $KW11L:  MOV     (SP),R0         ;SAVE THE RETURN PC
016532  012777  016566  162474           MOV     #1$,@INT6VEC    ;SETUP INTERRUPT VECTOR
016540  005001                           CLR     R1              ;SETUP COUNTER
016542  012777  000100  162466           MOV     #BIT6,@INT6ST   ;SET INTERRUPT ENABLE BIT & CLR MONITOR BIT
016550  005201                  2$:      INC     R1              ;WAIT FOR
016552  001376                           BNE     2$              ;INTERRUPT
016554  005077  162456                   CLR     @INT6ST         ;CLEAR INTERRUPT BIT
016560  062700  000002                   ADD     #2,R0           ;ADJUST RETURN PC
016564  000427                           BR      $ENDBR6         ;RETURN
016566  005077  162444          1$:      CLR     @INT6ST         ;CLEAR INTERRUPT FLAG
016572  000424                           BR      $ENDBR6         ;RETURN

                                ;
                                ;THIS CODE SETS UP THE KW11-P TO INTERRUPT AND IS CALLED BY A JSR PC,@INTER6
016574  011600                  $KW11P:  MOV     (SP),R0         ;SAVE THE RETURN PC
016576  012777  016640  162430           MOV     #1$,@INT6VEC    ;SETUP THE INTERRUPT VECTOR
016604  012737  000001  172544           MOV     #1,@#PLKC       ;SET THE COUNTER FOR ONE COUNT
016612  005001                           CLR     R1              ;SETUP THE WAIT COUNTER
016614  012777  000105  162414           MOV     #105,@INT6ST    ;START COUNTER
016622  005201                  2$:      INC     R1              ;WAIT FOR
016624  001376                           BNE     2$              ;INTERRUPT
016626  005077  162404                   CLR     @INT6ST         ;CLEAR INTERRUPT BIT
016632  062700  000002                   ADD     #2,R0           ;ADJUST R0 FOR RETURN
016636  000402                           BR      $ENDBR6         ;RETURN
016640  005077  162372          1$:      CLR     @INT6ST         ;CLEAR INTERRUPT FLAG
016644  010046                  $ENDBR6:MOV      R0,-(SP)        ;PUT RETURN PC ON STACK.
016646  000207                           RTS     PC              ;RETURN

                                ;;*****************************************************************
                                ;*TEST 33         BR LEVEL 4 INTERRUPT
                                ;*
                                ;*       BEN 13 SHOULD NOT FAIL.
                                ;*       IF THE INTERRUPT DOESN'T OCCUR AN ATTEMPT IS MADE TO
                                ;*       ISOLATE THE FAILURE.
                                ;;*****************************************************************
016650  000004                  TST33:   SCOPE
016652  012767  017144  162410           MOV     #TST34,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
016660  012767  003706  162216           MOV     #^D1990,$ICNT   ;ADJUST ITTERATION COUNT
016666  012767  016674  162212           MOV     #14$,$LPADR     ;SETUP LOOP ADR
016674  032737  000400  177570  14$:     BIT     #SW8,@#SWR      ;SWITCH 8 ON?
016702  001012                           BNE     8$              ;BRANCH IF YES
016704  032737  000020  177570           BIT     #SW4,@#SWR      ;IS SWITCH 4 ON?
016712  001406                           BEQ     8$              ;BRANCH IF NO
016714  005767  162160                   TST     $PASS           ;IS THIS FIRST PASS?
016720  001002                           BNE     9$              ;BRANCH IF NO
016722  104400  072460                   TYPE    ,EM712          ;TYPE MESSAGE(SKIPPING TEST)
016726                          9$:
016726  000506                           BR      TST34           ;;GO TO NEXT TEST
016730  012706  001100          8$:      MOV     #STACK,SP       ;INITIALIZE THE SP
016734  012737  017010  000064           MOV     #1$,@#TPVEC     ;SETUP THE TERMINAL INTERRUPT VECTOR
```

```
016742 032737 000020 177776           BIT     #BIT4,@#PSW      ;IS T BIT ON?
016750 001404                         BEQ     10$              ;BRANCH IF NO
016752 012737 000360 000066           MOV     #360,@#TPVEC+2   ;SETUP TPVEC PSW
016760 000403                         BR      11$
016762 012737 000340 000066   10$:    MOV     #PR7,@#TPVEC+2   ;PUT PRIORITY 7 IN NEW PSW
016770 000237                 11$:    SPL     7                ;SET CPU AT LEVEL 7
016772 005000                         CLR     R0               ;SETUP R0
016774 052777 000100 162140           BIS     #100,@$TPS       ;GET INTERRUPT ON BR 4
017002 005200                 2$:     INC     R0               ;WAIT AND SEE
017004 001376                         BNE     2$               ;IF INTERRUPT OCCURS
017006 000403                         BR      3$               ;NO INTERRUPT
017010 005077 162126          1$:     CLR     @$TPS            ;CLEAR INTERRUPT ENABLE
017014 104222                         ERROR   222              ;EITHER TMCB PS07(0) IS NOT GETTING TO
                                                               ;TMCB E77 OR E77 IS BAD
017016                        3$:
017016 012767 017144 162150           MOV     #TST34,$ESCAPE   ;SAVE START ADDRESS OF NEXT TEST
017024 012767 017144 162236           MOV     #TST34,NEXTTST   ;SAVE START ADDRESS OF NEXT TEST
017032 012737 017140 000064           MOV     #4$,@#TPVEC      ;SETUP THE INTERRUPT VECTOR
017040 005000                         CLR     R0               ;SETUP R0
017042 000233                         SPL     3                ;SET CPU AT LEVEL 3
017044 052777 000100 162070           BIS     #100,@$TPS       ;GET A BR 4
017052 005200                 5$:     INC     R0               ;WAIT FOR
017054 001376                         BNE     5$               ;INTERRUPT
017056 005077 162060                  CLR     @$TPS            ;CLEAR TP INTERRUPT BIT
                              ;BR 4 INTERRUPT FAILED.  IS THERE A BR 6 DEVICE AVAILABLE?
017062 005767 162144                  TST     INTER6           ;TEST LEVEL 6
017066 001422                         BEQ     7$
017070 016700 162140                  MOV     INT6VEC,R0       ;GET ADDR OF INTER 6 VECTOR
017074 062700 000002                  ADD     #2,R0            ;ADJUST TO PSW VEC
017100 032737 000020 177776           BIT     #BIT4,@#PSW      ;IS T BIT ON?
017106 001403                         BEQ     12$              ;BRANCH IF NO
017110 012710 000360                  MOV     #360,(R0)        ;SETUP PSW
017114 000402                         BR      13$
017116 012710 000340          12$:    MOV     #PR7,(R0)        ;SETUP PSW NO T BIT
017122 000235                 13$:    SPL     5                ;SET CPU AT 5
017124 004777 162102                  JSR     PC,@INTER6       ;EXECUTE INTERRUPT
017130 000402                         BR      6$               ;RETURN HERE IF 6 INTERRUPTS
017132 104223                         ERROR   223              ;BOTH BR 4 AND BR 6 FAILED
017134 104224                 7$:     ERROR   224              ;BR 4 FAILED
017136 104225                 6$:     ERROR   225              ;BR 4 FAILED BUT BR 6 OK
017140 005077 161776          4$:     CLR     @$TPS            ;CLEAR PRINTER INTERRUPT FLAG

                              ;;*****************************************************************
                              ;;*TEST 34        BR LEVEL 5 INTERRUPT
                              ;;*
                              ;;*       THE ONLY POSSIBLE FAILURE IS THAT TMCA HONOR BR 5
                              ;;*       DOES NOT GO LOW OR TMCB E62(6) IS BAD.
                              ;;*****************************************************************
017144 000004                 TST34:  SCOPE
017146 012767 017274 162020           MOV     #TST35,$ESCAPE   ;SAVE START ADDRESS OF NEXT TEST
017154 012767 017274 162106           MOV     #TST35,NEXTTST   ;SAVE START ADDRESS OF NEXT TEST
017162 032737 000400 177570           BIT     #SW8,@#SWR       ;SWITCH 8 ON?
017170 001012                         BNE     1$               ;BRANCH IF YES
017172 032737 000040 177570           BIT     #SW5,@#SWR       ;IS SWITCH 5 UP?
017200 001406                         BEQ     1$               ;BRANCH IF NO
017202 005767 161672                  TST     $PASS            ;IS THIS FIRST PASS?
```

```
017206  001002                          BNE     3$              ;BRANCH IF NO
017210  104400  072505                  TYPE    ,EM713          ;TYPE MESSAGE(TEST BEING SKIPPED)
017214                          3$:
017214  000427                          BR      TST35           ;;GO TO NEXT TEST
017216  005767  162002          1$:     TST     INTER5          ;IS THERE A DEVICE AVAILABLE?
017222  001424                          BEQ     TST35           ;;BRANCH IF NO
017224  012706  001100          2$:     MOV     #STACK,SP       ;INITIALIZE THE SP
017230  016700  161772                  MOV     INT5VEC,R0      ;GET ADDR OF BR 5 VECTOR
017234  062700  000002                  ADD     #2,R0           ;ADJUST TO PSW ADDR
017240  032737  000020  177776          BIT     #BIT4,@#PSW     ;IS T BIT ON?
017246  001403                          BEQ     5$              ;BRANCH IF NO
017250  012710  000360                  MOV     #360,(R0)       ;PUT NEW PSW IN VECTOR
017254  000402                          BR      6$              ;
017256  012710  000340          5$:     MOV     #PR7,(R0)       ;PUT NEW PSW IN VEC NO T BIT
017262  000234                  6$:     SPL     4               ;SET CPU AT LEVEL 4
017264  004777  161734                  JSR     PC,@INTER5      ;EXECUTE LEVEL 5 INTERRUPT
017270  000401                          BR      TST35           ;;GO TO NEXT TEST
017272  104226                          ERROR   226             ;BR 5 DID NOT INTERRUPT

                                ;;*************************************************************
                                ;*TEST 35       BR LEVEL 6 INTERRUPT
                                ;*
                                ;*      THE ONLY POSSIBLE FAILURE IS THAT TMCA HONOR BR 6 DOES NOT GO LOW
                                ;*      OR TMCB E62(12) IS BAD.
                                ;;*************************************************************
017274  000004                  TST35:  SCOPE
017276  012767  017424  161670          MOV     #TST36,$ESCAPE  ;SAVE START ADDRESS OF NEXT TEST
017304  012767  017424  161756          MOV     #TST36,NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
017312  032737  000400  177570          BIT     #SW8,@#SWR      ;SWITCH 8 ON?
017320  001012                          BNE     1$              ;BRANCH IF YES
017322  032737  000100  177570          BIT     #SW6,@#SWR      ;IS SWITCH 6 UP?
017330  001406                          BEQ     1$              ;BRANCH IF NO
017332  005767  161542                  TST     $PASS           ;IS THIS FIRST PASS?
017336  001002                          BNE     3$              ;BRANCH IF NO
017340  104400  072532                  TYPE    ,EM715          ;TYPE MESSAGE(TEST BEING SKIPPED)
017344                          3$:
017344  000427                          BR      TST36           ;;GO TO NEXT TEST
017346  005767  161660          1$:     TST     INTER6          ;IS THERE A DEVICE AVAILABLE?
017352  001424                          BEQ     TST36           ;;BRANCH IF NO
017354  012706  001100          2$:     MOV     #STACK,SP       ;INITIALIZE THE SP
017360  016700  161650                  MOV     INT6VEC,R0      ;GET ADR OF BR 6 VECTOR
017364  062700  000002                  ADD     #2,R0           ;ADJUST TO PSW ADDR
017370  032737  000020  177776          BIT     #BIT4,@#PSW     ;IS T BIT ON?
017376  001403                          BEQ     5$              ;BRANCH IF NO
017400  012710  000360                  MOV     #360,(R0)       ;SETUP NEW PSW
017404  000402                          BR      6$
017406  012710  000340          5$:     MOV     #PR7,(R0)       ;SETUP NEW PSW
017412  000235                  6$:     SPL     5               ;SET CPU AT LEVEL 5
017414  004777  161612                  JSR     PC,@INTER6      ;EXECUTE LEVEL 6 INTERRUPT
017420  000401                          BR      TST36           ;;GO TO NEXT TEST
017422  104227                          ERROR   227             ;BR 6 DID NOT INTERRUPT

                                ;;*************************************************************
                                ;*TEST 36       YELLOW ZONE TRAP
                                ;*
                                ;*      A YELLOW ZONE IS FIRST ATTEMPTED WITH THE SP AT 376.
```

```
                                         ;*        IF BEN 13 FAILS THE TRAP WILL NOT OCCUR.
                                         ;*
                                         ;*        IF THE PROCESSOR FAILS TO TRAP EITHER TMCD SL YEL IS
                                         ;*        NOT GOING HIGH OR TMCA HONOR SLY IS NOT GOING LOW
                                         ;*        OR TMCB E70(3) IS BAD OR BEN13 FAILED.
                                         ;*
                                         ;*        IF TMCC PRIORITY CLEAR DOES NOT GO LOW THE PROCESSOR WILL HANG
                                         ;*        UP IN A RED ZONE TRAP LOOP.
                                         ;*
                                         ;*        A JSR WITH A BAD SP IS THEN EXECUTED TO ENSURE TMCC
                                         ;*        KERNAL R6 GOES HIGH WHEN ENABLED BY ''STACK REFERENCE * KERNAL MODE''.
                                         ;*
                                         ;*        IF THE TRAP WORKS TESTS WILL BE PERFORMED TO ENSURE ALL THE
                                         ;*        APPROPRIATE CONDITIONS DISABLE THE TRAP EXCEPT
                                         ;*        THE PRIORITY ARBITRATOR.
                                         ;;*************************************************************
 017424  000004                          TST36:    SCOPE
 017426  012767  017472  161452                    MOV     #11$,$LPADR         ;SETUP LP ADR
 017434  012767  017472  161446                    MOV     #11$,$LPERR         ;SETUP ERROR LOOP
 017442  013767  177776  161520                    MOV     @#PSW,$TMP3         ;SAVE PSW
 017450  032737  000020  177776                    BIT     #BIT4,@#PSW         ;IS T BIT ON?
 017456  001405                                    BEQ     11$                 ;BRANCH IF NO
 017460  012746  000340                            MOV     #PR7,-(SP)          ;PUT NEW PSW ON STACK
 017464  012746  017472                            MOV     #11$,-(SP)          ;PUT RETURN ADR ON STACK
 017470  000006                                    RTT                         ;TURN T BIT OFF
 017472  012737  000340  000006          11$:      MOV     #PR7,@#ERRVEC+2     ;RESTORE ERR VEC PSW
 017500  012737  017536  000004                    MOV     #1$,@#ERRVEC        ;SETUP ERRVEC
 017506  012767  017514  161374                    MOV     #2$,$LPERR          ;SETUP ERROR LOOP
 017514  012706  000376                  2$:       MOV     #376,SP             ;SETUP THE SP
 017520  012700  177777                            MOV     #-1,R0              ;SETUP R0
 017524  010016                                    MOV     R0,(SP)             ;EXECUTE INSTRUCTION UNDER TEST
 017526  012706  001100                            MOV     #STACK,SP           ;REINITIALIZE THE SP
 017532  104230                                    ERROR   230                 ;YELLOW ZONE DID NOT OCCUR
 017534  000407                                    BR      8$
 017536  022737  177777  000376          1$:       CMP     #-1,@#376           ;DID RED ZONE OCCUR?
 017544  001403                                    BEQ     8$                  ;BRANCH IF NO
 017546  012706  001100                            MOV     #STACK,SP           ;RESTORE SP
 017552  104254                                    ERROR   254                 ;RED ZONE IN YELLOW REGION
                                         ;
                                         ;JSR WITH A BAD SP
 017554  012737  017614  000004          8$:       MOV     #6$,@#ERRVEC        ;SETUP ERRVEC
 017562  012767  017570  161320                    MOV     #63$,$LPERR         ;SETUP THE ERROR LOOP
 017570  012706  000376                  63$:      MOV     #376,SP             ;SETUP THE SP
 017574  004767  000000                            JSR     PC,7$               ;EXECUTE INSTRUCTION UNDER TEST
 017600  012706  001100                  7$:       MOV     #STACK,SP           ;RESET THE SP
 017604  012737  032654  000004                    MOV     #CPUSPUR,@#ERRVEC   ;RESTORE ERRVEC
 017612  104232                                    ERROR   232                 ;TMCC KERNAL R6 DID NOT GO HIGH ON JSR
                                         ;
                                         ;DISABLE WITH STACK LIMIT REGISTER
 017614  012737  017642  000004          6$:       MOV     #3$,@#ERRVEC        ;SETUP ERRVEC
 017622  012767  017630  161260                    MOV     #62$,$LPERR         ;SETUP ERROR LOOP
 017630  012706  000740                  62$:      MOV     #740,SP             ;SETUP THE SP
 017634  013716  000742                            MOV     @#742,(SP)          ;EXECUTE INSTRUCTION UNDER TEST
 017640  000406                                    BR      4$                  ;GO TO NEXT SECTION
 017642  012737  032654  000004          3$:       MOV     #CPUSPUR,@#ERRVEC   ;RESTORE ERRVEC
 017650  012706  001100                            MOV     #STACK,SP           ;RESTORE THE SP
```

```
017654  104233                             ERROR   233                ;PDRC STACK LIMIT DID NOT DISABLE TRAP
                                    ;
                                    ;DISABLE WITH TMCC KERNAL R6
                                    ;  DISABLE KERNAL R6 WITH DATI
017656  012737  017702  000004      4$:    MOV     #5$,@#ERRVEC       ;SETUP ERRVEC
017664  012767  017672  161216             MOV     #61$,$LPERR        ;SETUP THE ERROR LOOP
017672  012706  000376              61$:   MOV     #376,SP            ;SETUP THE SP
017676  011606                             MOV     (SP),SP            ;EXECUTE INSTRUCTION UNDER TEST
017700  000415                             BR      9$                 ;GO TO NEXT SECTION
017702  012706  001100              5$:    MOV     #STACK,SP          ;RESTORE THE SP
017706  012737  032654  000004             MOV     #CPUSPUR,@#ERRVEC  ;RESTORE ERRVEC
017714  104234                             ERROR   234                ;TMCC KERNAL R6 DID NOT DISABLE ON DATI
017716  000406                             BR      9$
                                    ;
                                    ;USED FOR ERROR LOOP IF BIT 3 IN ERROR REG
                                    ;DOES NOT SET
017720  012706  000376              60$:   MOV     #376,SP            ;SETUP THE SP
017724  012737  017734  000004             MOV     #9$,@#ERRVEC       ;SETUP ERROR VECTOR
017732  005016                             CLR     (SP)               ;CAUSE TRAP
017734  012767  017720  161146      9$:    MOV     #60$,$LPERR        ;SETUP ERROR LOOP
017742  012706  001100                     MOV     #STACK,SP          ;RESTORE THE SP
017746  022737  000010  177766             CMP     #BIT3,@#CPUERR     ;DID YEL ZONE BIT SET?
017754  001407                             BEQ     10$                ;BRANCH IF YES
017756  013767  177766  161170             MOV     @#CPUERR,$REG0     ;SAVE FOR TYPEOUT
017764  012767  000010  161170             MOV     #BIT3,$TMP0        ;SAVE EXPECTED VALUE
```

```
017772  104257                          ERROR   257             ;YEL ZONE BIT DID NOT SET IN CPUERR
017774  012767  020002  161106  10$:    MOV     #57$,$LPERR     ;SETUP ERROR LOOP
020002  005037  177766          57$:    CLR     @#CPUERR        ;CLEAR YEL ZONE BIT
020006  005737  177766                  TST     @#CPUERR        ;DID IT CLEAR?
020012  001401                          BEQ     TST37           ;;BRANCH IF YES
020014  104260                          ERROR   260             ;BIT3 DID NOV CLEAR IN CPUERR
                                ;;*************************************************************
                                ;*TEST 37       ROM FIELD CHECK OF PC MANIPULATOR STATES
                                ;*
                                ;*      THIS TEST EXECUTES THE MACHINE STATES THAT MANIPULATE
                                ;*      THE PC TO ENSURE THAT THE PCB ROM FIELD OR DRX ROM FIELD
                                ;*      OR SRX ROM FIELD OF THESE STATES IS FUNCTIONAL.
                                ;*      THESE STATES ARE S13.00, S45.00, MTP.10, D45.80, D45.90,
                                ;*      D45.00, AND D45.01.
                                ;;*************************************************************
020016  000004                  TST37:  SCOPE
020020  012767  020050  161060          MOV     #18$,$LPADR     ;SETUP LOOP ADR
020026  032767  000020  161134          BIT     #BIT4,$TMP3     ;WAS T BIT ON?
020034  001405                          BEQ     18$             ;BRANCH IF NO
020036  012746  000360                  MOV     #360,-(SP)      ;PUT NEW PSW ON STACK
020042  012746  020050                  MOV     #18$,-(SP)      ;PUT RETURN ADDR ON STACK
020046  000006                          RTT                     ;TURN T BIT ON
                                ;BIN*SM1*SF7*DM0(S13.00)
020050  012767  020056  161032  18$:    MOV     #64$,$LPERR     ;SETUP ERROR LOOP
020056  011700                  64$:    MOV     (PC),R0         ;EXECUTE INSTRUCTION UNDER TEST
020060  020027  020027                  CMP     R0,#20027       ;DID TEST WORK
020064  001402                          BEQ     1$              ;BRANCH IF YES
020066  104377                          ERROR   377             ;STATE S13.00 FAILED
020070  000443                          443
                                ;BIN*SM4*SF7*DM2*DF7 (S45.00)
020072  012767  020100  161010  1$:     MOV     #63$,$LPERR     ;SETUP ERROR LOOP
020100  012767  024727  000000  63$:    MOV     #24727,2$       ;PUT INSTRUCTION IN 2$
020106  024727                  2$:     CMP     -(PC),(PC)+     ;EXECUTE INSTRUCTION UNDER TEST
020110  000240                          NOP                     ;USED TO BE SAFE
020112  001402                          BEQ     3$              ;BRANCH IF TEST OK
020114  104377                          ERROR   377             ;STATE S45.00 FAILED
020116  000444                          444
                                ;MTP*DM2*DF7 (MTP.10)
020120  012767  020126  160762  3$:     MOV     #62$,$LPERR     ;SETUP ERROR LOOP
020126  012706  001100          62$:    MOV     #STACK,SP       ;INITIALIZE THE SP
020132  012746  177777                  MOV     #-1,-(SP)       ;SET 1076 TO ALL ONES
020136  005067  000002                  CLR     4$              ;ENSURE 4$ CLEAR
020142  006627                          MTPI    (PC)+           ;EXECUTE INSTRUCTION UNDER TEST
020144  000000                  4$:     .WORD   0
020146  022767  177777  177770          CMP     #-1,4$          ;DID INSTRUCTION WORK?
020154  001402                          BEQ     7$              ;BRANCH IF YES
020156  104377                          ERROR   377             ;STATE MTP.10 FAILED
020160  000445                          445
                                ;BIN*SM2*SF7*DM4*DF7 (D45.80)
020162  012767  020170  160720  7$:     MOV     #61$,$LPERR     ;SETUP ERROR LOOP
020170  012747                  61$:    MOV     (PC)+,-(PC)     ;EXECUTE INSTRUCTION UNDER TEST
020172  000402                          BR      10$             ;WILL EXECUTE THIS IF INSTR. WORKS
020174  104377                          ERROR   377             ;STATE D45.80 FAILED
020176  000447                          447
                                ;BIN*SM1*SF0*DM4*DF7 (D45.90)
020200  012767  020206  160702  10$:    MOV     #60$,$LPERR     ;SETUP ERROR LOOP
```

```
020206  012767  141047  000024  60$:    MOV     #141047,11$         ;SETUP INSTRUCTION TO EXECUTE
020214  012700  001162                  MOV     #$TMP0,R0           ;PUT ADDRESS OF $TMP0 IN R0
020220  005200                          INC     R0
020222  112710  000002                  MOVB    #BIT1,(R0)          ;SET BIT1 IN $TMP1 HIGH BYTE
020226  012705  001166                  MOV     #$TMP2,R5           ;SETUP R5
020232  012767  001000  160724          MOV     #BIT9,$TMP1         ;SETUP $TMP1
020240  141047                  11$:    BICB    (R0),-(PC)          ;EXECUTE INSTRUCTION UNDER TEST
020242  005767  160716                  TST     $TMP1               ;DID $TMP1 GET CLEARED?
020246  001402                          BEQ     12$                 ;BRANCH IF YES
020250  104377                          ERROR   377                 ;STATE D45.00 FAILED
020252  000450                          450
                                ;DAC*DM4*DF7 (D45.00)
020254  012767  020262  160626  12$:    MOV     #57$,$LPERR         ;SETUP ERROR LOOP
020262  012767  140047  000016  57$:    MOV     #140047,13$         ;SETUP INSTRUCTION TO EXECUTE
020270  012700  000002                  MOV     #BIT1,R0            ;SETUP R0 TO CHANGE DR FROM 7 TO 5
020274  012705  001165                  MOV     #$TMP1+1,R5         ;PUT ADDRESS OF $TMP1 HIGH BYTE IN R5
020300  012767  000002  160656          MOV     #BIT1,$TMP1         ;SET UP $TMP1 SO INSTRUCTION CLEARS IT
020306  140047                  13$:    BICB    R0,-(PC)            ;EXECUTE INSTRUCTION UNDER TEST
020310  005767  160650                  TST     $TMP1               ;DID $TMP1 CLEAR?
020314  001402                          BEQ     14$                 ;BRANCH IF YES
020316  104377                          ERROR   377                 ;STATE D45.90 FAILED
020320  000451                          451
                                ;DAC*DM5*DF7 (D45.01)
020322  012767  020330  160560  14$:    MOV     #56$,$LPERR         ;SETUP ERROR LOOP
020330  012767  130057  000052  56$:    MOV     #130057,15$         ;SETUP INSTRUCTION TO EXECUTE
020336  032737  000020  177776          BIT     #BIT4,@#PSW         ;IS T BIT ON?
020344  001404                          BEQ     20$                 ;BRANCH IF NO
020346  012737  000360  000066          MOV     #360,@#TPVEC+2      ;SETUP TP VEC PSW
020354  000403                          BR      17$
020356  012737  000340  000066  20$:    MOV     #PR7,@#TPVEC+2
020364  012737  020424  000064  17$:    MOV     #16$,@#TPVEC        ;SETUP BR4 INTERRUPT VECTOR
020372  000233                          SPL     3                   ;SET PROCESSOR PRIORITY BELOW BR4
020374  152777  000100  160540          BISB    #BIT6,@$TPS         ;SET INTERRUPT BIT
020402  012777  000015  160534          MOV     #15,@$TPB           ;SEND CHARACTER TO PRINTER
020410  130057                  15$:    BITB    R0,@-(PC)           ;EXECUTE INSTRUCTION UNDER TEST
020412  142777  000100  160522          BICB    #BIT6,@$TPS         ;WILL EXECUTE IF STATE FAILS
020420  104377                          ERROR   377                 ;STATE D45.01 FAILED
020422  000452                          452
020424  142777  000100  160510  16$:    BICB    #BIT6,@$TPS         ;TEST OK, CLEAR INTERRUPT BIT
                                                                    ;CONTINUE

                                ;;*****************************************************************
                                ;*TEST 40        RED ZONE TRAP
                                ;*
                                ;*      A RED ZONE TRAP IS FIRST ATTEMPTED WITH THE SP AT 336.
                                ;*      IF BEN13 FAILS EXECUTION WILL GO TO EITHER BRK.80 OR BRK.20
                                ;*      OR PUP.00.
                                ;*      BRK.80 WILL CAUSE THE OLD PSW AND PC TO BE STACKED ON THE
                                ;*      OLD STACK INSTEAD OF LOCATIONS 2 AND 0.
                                ;*      BRK.20  WILL MAKE IT LOOK LIKE THE RED ZONE FAILED.
                                ;*      PUP.00 WILL CAUSE A TRAP TO LOCATION 24.
                                ;*
                                ;*      IF THE PROCESSOR FAILS TO TRAP EITHER TMCD SL RED IS
                                ;*      NOT GOING LOW OR TMCC ABORT IS NOT GOING LOW.
                                ;*
                                ;*      IF UBCB ABORT RESTART FAILS TO GO LOW OR E10(13)
```

```
                                        ;*      IS BAD THE PROCESSOR WILL HANG IN THE PAUSE STATE.
                                        ;;*****************************************************************
020432  000004                          TST40:  SCOPE
020434  012767  021166  160626                  MOV     #TST41,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
020442  012767  020500  160436                  MOV     #14$,$LPADR     ;SETUP LOOP ADR
020450  013767  177776  160512                  MOV     @#PSW,$TMP3     ;SAVE PSW
020456  032737  000020  177776                  BIT     #BIT4,@#PSW     ;IS T BIT ON?
020464  001405                                  BEQ     14$             ;BRANCH IF NO
020466  012746  000340                          MOV     #PR7,-(SP)      ;PUT NEW PSW ON STACK
020472  012746  020500                          MOV     #14$,-(SP)      ;PUT RETURN ADR ON STACK
020476  000006                                  RTT                     ;TURN T BIT OFF
020500  012737  000340  000006  14$:            MOV     #PR7,@#ERRVEC+2 ;RESTORE ERRVEC PSW
020506  012767  020522  160374                  MOV     #64$,$LPERR     ;SETUP ERROR LOOP
020514  012737  020560  000024                  MOV     #3$,@#PWRVEC    ;SETUP LOCATION 24
020522  012737  020576  000004  64$:            MOV     #1$,@#ERRVEC    ;SETUP ERRVEC
020530  012706  000336                          MOV     #336,SP         ;SET THE SP TO RED ZONE
020534  012700  177777                          MOV     #-1,R0          ;SETUP R0
020540  010016                                  MOV     R0,(SP)         ;EXECUTE THE TRAP INSTRUCTION
020542  012706  001100          6$:             MOV     #STACK,SP       ;RESET THE SP
020546  012737  032654  000004                  MOV     #CPUSPUR,@#ERRVEC  ;RESTORE ERRVEC
020554  104235                                  ERROR   235             ;RED ZONE REFERENCE FAILED TO TRAP
020556  000431                                  BR      8$
020560  012706  001100          3$:             MOV     #STACK,SP       ;RESET THE SP
020564  012737  032654  000004                  MOV     #CPUSPUR,@#ERRVEC  ;RESTORE LOCATION 4
020572  104237                                  ERROR   237             ;BEN 13 FAILED TO PUP.00
020574  000422                                  BR      8$
020576  023727  000000  020542  1$:             CMP     @#0,#6$         ;DID BEN 13 FAIL?
020604  001407                                  BEQ     7$              ;BRANCH IF NO
020606  012706  001100                          MOV     #STACK,SP       ;RESET THE SP
020612  012737  032654  000004                  MOV     #CPUSPUR,@#ERRVEC  ;RESTORE ERRVEC
020620  104240                                  ERROR   240             ;BEN 13 FAILED TO BRK.80
020622  000407                                  BR      8$
020624  022737  177777  000336  7$:             CMP     #-1,@#336       ;;DID YEL ZONE OCCUR?
020632  001003                                  BNE     8$              ;BRANCH IF NO
020634  005037  000336                          CLR     @#336           ;SETUP FOR LOOPING
020640  104251                                  ERROR   251             ;YEL ZONE IN RED REGION
                                        ;
                                        ;TEST TO ENSURE PSW REFERENCE VIA THE SP CAUSES A RED ZONE TRAP
020642  012767  020650  160240  8$:             MOV     #63$,$LPERR     ;SETUP ERROR LOOP
020650  012737  020700  000004  63$:            MOV     #4$,@#ERRVEC    ;SETUP ERRVEC
020656  012706  177776                          MOV     #PSW,SP         ;PUT ADDRESS OF PSW IN SP
020662  005016                                  CLR     (SP)            ;EXECUTE THE TRAP CAUSING INSTRUCTION
020664  012706  001076                          MOV     #1076,SP        ;RESET THE SP
020670  012737  032654  000004                  MOV     #CPUSPUR,@#ERRVEC  ;RESTORE ERRVEC
020676  104241                                  ERROR   241             ;NO RED ZONE ON STACK OVERFLOW
                                        ;
                                        ;TEST TO ENSURE SL REG GREATER THAN BADRR CAUSES A RED ZONE
020700  012767  020706  160202  4$:             MOV     #62$,$LPERR     ;SETUP ERROR LOOP
020706  012737  020750  000004  62$:            MOV     #5$,@#ERRVEC    ;SETUP RESVEC
020714  012737  000400  177774                  MOV     #400,@#STKLMT   ;SET STACK LIMIT REGISTER TO 400
020722  012706  000336                          MOV     #336,SP         ;SET THE SP
020726  005016                                  CLR     (SP)            ;EXECUTE THE TRAP CAUSING INSTRUCTION
020730  012706  001100                          MOV     #STACK,SP       ;RESET THE SP
020734  005037  177774                          CLR     @#STKLMT        ;ENSURE SL REG. CLEAR
020740  012737  032654  000004                  MOV     #CPUSPUR,@#ERRVEC  ;RESTORE ERRVEC
020746  104242                                  ERROR   242             ;NO RED ZONE WHEN SL REG>BUS ADDR
```

```
                                       ;
                                       ;TEST OF TMCD YEL ZONE (E15)
020750  012767  020762  160132  5$:     MOV     #61$,$LPERR     ;SETUP ERROR LOOP
020756  005037  177774                  CLR     @#STKLMT        ;ENSURE SL CLEAR
020762  012737  021002  000004  61$:    MOV     #9$,@#ERRVEC    ;SETUP ERRVEC
020770  012706  000240                  MOV     #240,SP         ;SETUP THE SP
020774  012700  177777                  MOV     #-1,R0          ;SETUP R0
021000  010016                          MOV     R0,(SP)         ;EXECUTE THE TRAP CAUSING INSTRUCTION
021002  022737  177777  000240  9$:     CMP     #-1,@#240       ;DID YEL ZONE OCCUR?
021010  001010                          BNE     10$             ;BRANCH IF NO
021012  012706  001100                  MOV     #STACK,SP       ;RESTORE THE SP
021016  012737  032654  000004          MOV     #CPUSPUR,@#ERRVEC  ;RESTORE ERRVEC
021024  005037  000240                  CLR     @#240           ;FOR LOOPING
021030  104252                          ERROR   252             ;TMCD YEL ZONE DID NOT GO LOW
021032  012767  021040  160050  10$:    MOV     #60$,$LPERR     ;SETUP ERROR LOOP
021040  012737  021054  000004  60$:    MOV     #11$,@#ERRVEC   ;SETUP ERRVEC
021046  012706  000140                  MOV     #140,SP         ;SETUP THE SP
021052  010016                          MOV     R0,(SP)         ;EXECUTE THE TRAP CAUSING INSTR.
021054  022737  177777  000140  11$:    CMP     #-1,@#140       ;DID YEL ZONE OCCUR?
021062  001010                          BNE     12$             ;BRANCH IF NO
021064  012706  001100                  MOV     #STACK,SP       ;RESTORE SP
021070  012737  032654  000004          MOV     #CPUSPUR,@#ERRVEC  ;RESTORE ERRVEC
021076  005037  000140                  CLR     @#140           ;FOR LOOPING
021102  104253                          ERROR   253             ;TMCD YEL ZONE DID NOT GO LOW
021104  012706  001100          12$:    MOV     #STACK,SP       ;RESTORE SP
021110  012737  032654  000004          MOV     #CPUSPUR,@#ERRVEC  ;RESTORE ERRVEC
021116  022737  000004  177766          CMP     #BIT2,@#CPUERR  ;DID RED ZONE BIT IN CPU ERROR SET?
021124  001407                          BEQ     13$             ;BRANCH IF YES
021126  013767  177766  160020          MOV     @#CPUERR,$REG0  ;SAVE FOR TYPEOUT
021134  012767  000004  160020          MOV     #BIT2,$TMP0     ;SAVE EXPECTED VALUE
021142  104261                          ERROR   261             ;RED ZONE BIT IN CPU ERROR DID NOT SET
021144  012767  021152  157736  13$:    MOV     #57$,$LPERR     ;SETUP ERROR LOOP
021152  005037  177766          57$:    CLR     @#CPUERR        ;CLEAR RED ZONE BIT
021156  005737  177766                  TST     @#CPUERR        ;DID REG CLEAR?
021162  001401                          BEQ     TST41           ;;BRANCH IF YES
021164  104262                          ERROR   262             ;RED ZONE BIT DID NOT CLEAR

                                       ;;*****************************************************************
                                       ;;*TEST 41        BIT TEST OF STACK LIMIT REGISTER
                                       ;;*
                                       ;;*       FIRST A 125252 AND 52525 PATTERN IS PUT IN THE REGISTER TO ENSURE
                                       ;;*       THAT THE REGISTER DOESN'T HAVE ANY STUCK BITS AND THAT
                                       ;;*       THE DMUX SELECT AND INPUT LINES WORK.
                                       ;;*
                                       ;;*       IF SCCE SL ADRS DOES NOT GET TO TMCD OR IF TMCD E28 OR E14
                                       ;;*       IS BAD THE BR WILL BE SELECTED.  THE PB REGISTER IS LOADED
                                       ;;*       WITH 200 SO IF TMCD LO BYTE EN DOES NOT GO LOW AN
                                       ;;*       ERROR WILL BE DETECTED.
                                       ;;*****************************************************************
021166  000004                  TST41:  SCOPE
021170  012767  021432  157776          MOV     #TST42,$ESCAPE  ;SAVE START ADDRESS OF NEXT TEST
021176  012767  021432  160064          MOV     #TST42,NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
021204  012737  036266  000024          MOV     #$PWRDN,@#PWRVEC ;RESTORE POWER VECTOR
021212  005005                          CLR     R5              ;CLEAR R5
021214  012737  125252  177774          MOV     #125252,@#STKLMT  ;PUT PATTERN IN STACK LIMIT REG
021222  012737  000200  177770          MOV     #200,@#177770   ;PUT 200 IN PB REGISTER
```

```
021230  012700  125000                          MOV     #125000,R0         ;SETUP R0 TO LOOK LIKE STK LIMIT
021234  000237                                  SPL     7                  ;PUT PRIORITY BITS IN KNOWN CONFIGURATION
021236  020037  177774                          CMP     R0,@#STKLMT        ;EXECUTE TEST ON STKLMT REG.
021242  001404                                  BEQ     1$                 ;BRANCH IF TEST OK
021244  005205                                  INC     R5                 ;INCREMENT TEST FAIL INDICATOR
021246  013767  177774  157744                  MOV     @#STKLMT,E1STKLM   ;SAVE ERROR VALUE
021254  012737  052400  177774  1$:             MOV     #52400,@#STKLMT    ;PUT COMPLEMENT PATTERN REG.
021262  012700  052400                          MOV     #52400,R0          ;PUT IN R0
021266  020037  177774                          CMP     R0,@#STKLMT        ;EXECUTE TEST ON REG.
021272  001415                                  BEQ     2$                 ;BRANCH IF TEST OK
021274  005705                                  TST     R5                 ;DID FIRST TEST FAIL?
021276  001023                                  BNE     3$                 ;BRANCH IF YES
021300  013767  177774  157714                  MOV     @#STKLMT,E2STKLM   ;SAVE ERROR VALUE
021306  012767  052400  157646                  MOV     #52400,$TMP0       ;SAVE EXPECTED VALUE
021314  005037  177774                          CLR     @#STKLMT           ;CLEAR THE REG
021320  012706  001100                          MOV     #STACK,SP          ;RESTORE THE SP
021324  104243                                  ERROR   243                ;52400 PATTERN FAILED
021326  005705                          2$:     TST     R5                 ;DID FIRST TEST FAIL?
021330  001436                                  BEQ     4$                 ;BRANCH IF NO
021332  012767  125000  157622                  MOV     #125000,$TMP0      ;SAVE EXPECTED VALUE
021340  005037  177774                          CLR     @#STKLMT           ;CLEAR REG.
021344  104244                                  ERROR   244                ;125252 PATTERN FAILED
021346  005037  177774                  3$:     CLR     @#STKLMT           ;CLEAR STACK LIMIT REG
021352  026727  157642  013767                  CMP     E1STKLM,#13767     ;DID BR GET SELECTED ON STACK LIMIT REF.?
021360  001001                                  BNE     5$                 ;BRANCH IF NO
021362  104245                                  ERROR   245                ;BR SELECTED BY DMUX
021364  026727  157630  125200  5$:             CMP     E1STKLM,#125200    ;DID PB GET GATED ALSO?
021372  001001                                  BNE     6$                 ;BRANCH IF NO
021374  104246                                  ERROR   246                ;TMCD LO BYTE EN DOES NOT GO LOW
021376  022767  000340  157614  6$:             CMP     #340,E1STKLM       ;DID PS GET SELECTED?
021404  001001                                  BNE     7$                 ;BRANCH IF NO
021406  104247                                  ERROR   247                ;D MUX SELECTED PSW
021410  012767  125000  157544  7$:             MOV     #125000,$TMP0
021416  012767  052400  157540                  MOV     #52400,$TMP1
021424  104250                                  ERROR   250                ;BOTH PATTERNS FAILED BUT DON'T KNOW WHY
021426  005037  177774                  4$:     CLR     @#STKLMT           ;CLEAR THE STACK LIMIT REG.

                                        ;;*******************************************************************
                                        ;*TEST 42         SL REGISTER COMPARATOR TEST 1
                                        ;*
                                        ;*       THIS TEST RUNS A HIGH BYTE COUNT PATTERN THRU THE BUS
                                        ;*       ADDRESS MUX FOR EACH PATTERN OF THE STACK LIMIT REGISTER.
                                        ;*       FOR EACH PATTERN OF ADDRESSES THERE WILL BE ONE YEL TRAP
                                        ;*       AT THE ADDRESS CORRESPONDING TO THE SL REG+340 AND A RED
                                        ;*       TRAP AT EVERY ADDRESS BELOW THIS.
                                        ;*       THIS TEST ONLY TESTS ADDRESSES UP TO THE I/O PAGE.
                                        ;*       THE I/O PAGE ADDRESSES WILL BE TESTED SEPARATELY WITH
                                        ;*       MEMORY MANAGEMENT ENABLED AND THE I/O PAGE MAPPED INTO RESIDENT
                                        ;*       MEMORY
                                        ;*
                                        ;*       THE FOLLOWING ARE THE TYPES OF ERRORS THAT CAN OCCUR IN THIS TEST:
                                        ;*              TYPE            DESCRIPTION
                                        ;*               0        RED ZONE TRAP ON YELLOW ZONE ADDRESS
                                        ;*               2        RED ZONE TRAP ON LEGAL ADDRESS
                                        ;*               4        YELLOW ZONE TRAP ON RED ZONE ADDRESS
                                        ;*               6        YELLOW ZONE TRAP ON LEGAL ADDRESS
                                        ;*              10       NO TRAP ON RED ZONE ADDRESS
                                        ;*              12       NO TRAP ON YELLOW ZONE ADDRESS
                                        ;*
                                        ;*      THE LOW BYTE ADDRESS IN THE STACK POINTER IS ALWAYS
                                        ;*      340 AND WILL NOT BE TYPED ON AN ERROR.
                                        ;;************************************************************
021432  000004                          TST42:  SCOPE
                                        ;;************************************************************
                                        ;*NOTE: IF THE LOOP ON ERROR SWITCH IS UP (SWITCH 9) THE TEST WILL
                                        ;*      LOOP ON THE FIRST ERROR WITH NO ERROR TYPEOUT.  OTHERWISE ALL
                                        ;*      ERRORS WILL BE RECORDED IN A TABLE AND TYPED OUT AT THE
                                        ;*      END OF THE TEST.
                                        ;*      IF SWITCH 3 (DISABLE MEMORY MANAGEMENT TESTS) IS NOT ON,
                                        ;*      THIS TEST IS SKIPPED AND TEST 70 WILL EXECUTE.
                                        ;;************************************************************
021434  012767  022042  157532                  MOV     #TST43,$ESCAPE  ;SAVE START ADDRESS OF NEXT TEST
021442  012767  022042  157620                  MOV     #TST43,NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
021450  012767  003706  157426                  MOV     #^D1990,$ICNT   ;SETUP ITTERATION COUNT
021456  012767  021464  157422                  MOV     #13$,$LPADR     ;SETUP LOOP ADDRESS
021464  032737  000010  177570  13$:    BIT     #BIT3,@#SWR     ;IS SWITCH 3 ON?
021472  001002                                  BNE     11$             ;BRANCH IF YES
021474  000177  157474                          JMP     @$ESCAPE        ;GO TO NEXT TEST
021500  012737  021610  000004  11$:    MOV     #1$,@#ERRVEC    ;SETUP ERRVEC
021506  005067  157454                          CLR     $TMP2           ;INITIALIZE ERROR OVERFLOW FLAG
021512  012703  000340                          MOV     #340,R3         ;SET SOB COUNT FOR SL REGISTER
021516  012700  120000                          MOV     #120000,R0      ;INITIALIZE ERROR DATA POINTER
021522  005060  000002                          CLR     2(R0)           ;INITIALIZE ERROR DATA BUFFER
021526  012701  000340                          MOV     #340,R1         ;INITIALIZE YELLOW ZONE ADDRESS(TRAP CASE)
021532  012704  000344                          MOV     #344,R4         ;SETUP YELLOW ZONE ADDR(NO TRAP)
021536  012702  000340                  2$:     MOV     #340,R2         ;SET SOB COUNT FOR SP
021542  012705  000344                          MOV     #344,R5         ;INITIALIZE SECONDARY STORAGE FOR SP
021546  016567  177776  157406  3$:     MOV     -2(R5),$TMP0    ;SAVE WORDS AT
021554  016567  177774  157402                  MOV     -4(R5),$TMP1    ;STACK UNDER TEST
021562  010506                          4$:     MOV     R5,R6           ;SET THE SP
021564  011616                                  MOV     (R6),(R6)       ;EXECUTE TEST INSTRUCTION
                                        ;NO TRAP. DETERMINE IF THIS IS CORRECT.
021566  020604                                  CMP     R6,R4           ;IS ADDRESS > YELL ZONE BOUNDRY?
021570  101066                                  BHI     5$              ;BRANCH IF YES
021572  001403                                  BEQ     6$              ;BRANCH IF ADDRESS = YELL ZONE BOUNDRY
                                        ;NO TRAP ADDRESS IS LESS THAN YELLOW ZONE BOUNDRY
021574  052710  000010                          BIS     #BIT3,(R0)      ;SET ERROR TYPE IN DATA BUFFER
021600  000442                                  BR      10$             ;GO RECORD DATA
                                        ;NO TRAP EQUALS YELLOW ZONE BOUNDRY
021602  052710  000012                  6$:     BIS     #12,(R0)        ;SET ERROR TYPE IN DATA BUFFER
021606  000437                                  BR      10$             ;GO RECORD DATA
                                        ;
                                        ;GOT A TRAP.  RESTORE AND DETERMINE IF IT IS CORRECT.
021610  016765  157346  177776  1$:     MOV     $TMP0,-2(R5)    ;RESTORE WORDS AT
021616  016765  157342  177774                  MOV     $TMP1,-4(R5)    ;OLD STACK UNDER TEST
021624  032737  000004  177766                  BIT     #BIT2,@#CPUERR  ;WAS IT A RED ZONE?
021632  001406                                  BEQ     8$              ;BRANCH IF NO
021634  020106                                  CMP     R1,R6           ;IS ADDRESS < YELL ZONE BOUNDRY?
021636  101043                                  BHI     5$              ;BRANCH IF YES
021640  001422                                  BEQ     10$             ;BRANCH IF ADDRESS = YELL ZONE BOUNDRY
                                        ;RED ZONE TRAP ON LEGAL ADDRESS
021642  052710  000002                          BIS     #BIT1,(R0)      ;SET ERROR TYPE IN DATA BUFFER
```

```
021646  000417                            BR      10$                ;GO RECORD DATA
                                  ;NOT A RED ZONE.  IS IT A YELLOW ZONE?
021650  032737  000010  177766    8$:     BIT     #BIT3,@#CPUERR     ;IS THIS A YELLOW ZONE TRAP?
021656  001002                            BNE     12$                ;BRANCH IF NO
021660  000167  010770                    JMP     CPUSPUR            ;GO TO SPURIOUS ROUTINE
021664  020106                    12$:    CMP     R1,R6              ;IS ADDRESS = YELL ZONE BOUNDRY?
021666  001427                            BEQ     5$                 ;BRANCH IF YES
021670  101003                            BHI     9$                 ;BRANCH IF ADDRESS IS < YELL ZONE BOUNDRY
                                  ;YELLOW ZONE TRAP ON LEGAL ADDRESS
021672  052710  000006                    BIS     #6,(R0)            ;SET ERROR TYPE IN DATA BUFFER
021676  000403                            BR      10$                ;GO RECORD DATA
                                  ;YELLOW ZONE TRAP ON RED ZONE ADDRESS
021700  052710  000004            9$:     BIS     #BIT2,(R0)         ;SET ERROR TYPE IN DATA BUFFER
021704  000400                            BR      10$                ;GO RECORD DATA
                                  ;
                                  ;RECORD ERROR DATA
021706  032737  001000  177570    10$:    BIT     #BIT9,@#SWR        ;IS LOOP ON ERROR ENABLED?
021714  001322                            BNE     4$                 ;BRANCH IF YES
021716  005767  157244                    TST     $TMP2              ;HAS ERROR BUFFER OVERFLOWED?
021722  001011                            BNE     5$                 ;BRANCH IF YES
021724  005200                            INC     R0                 ;SET POINTER TO HIGH BYTE
021726  113720  177775                    MOVB    @#STKLMT+1,(R0)+   ;SAVE ERROR STACK LIMIT
021732  010620                            MOV     R6,(R0)+           ;SAVE ERROR SP
021734  020027  157774                    CMP     R0,#157774         ;IS BUFFER AT PAGE 7?
021740  001002                            BNE     5$                 ;BRANCH IF NO
021742  005267  157220                    INC     $TMP2              ;SET BUFFER OVERFLOW FLAG
                                  ;
021746  062705  000400            5$:     ADD     #400,R5            ;GO TO NEXT STACK ADDRESS
021752  005037  177766                    CLR     @#CPUERR           ;CLEAR ERROR REG
021756  000240                            NOP
021760  005302                            DEC     R2                 ;REPLACES A
021762  001271                            BNE     3$                 ;SOB INSTRUCTION
021764  062737  000400  177774            ADD     #400,@#STKLMT      ;GO TO NEXT SL ADDRESS
021772  062701  000400                    ADD     #400,R1            ;SET NEXT YELLOW ZONE ADDRESS
021776  000240                            NOP
022000  062704  000400                    ADD     #400,R4            ;SET NEXT YELLOW ZONE ADDR(NO TRAP)
022004  005303                            DEC     R3                 ;THIS REPLACES
022006  001253                            BNE     2$                 ;A SOB
                                  ;
                                  ;DONE WITH TEST.  WAS THERE AN ERROR?
022010  005037  177774                    CLR     @#STKLMT           ;RESET THE SL REG
022014  012706  001100                    MOV     #STACK,SP          ;AND SP
022020  012737  032654  000004            MOV     #CPUSPUR,@#ERRVEC  ;RESTORE ERRVEC
022026  005737  120002                    TST     @#120002           ;WAS THERE AN ERROR?
022032  001403                            BEQ     TST43              ;;BRANCH IF NO
022034  010067  157114                    MOV     R0,$REG0           ;SAVE ERROR DATA POINTER
022040  104263                            ERROR   263                ;STACK LIMIT COMPARATORS FAILED
                                  ;;*****************************************************************
                                  ;*TEST 43          ODD ADDRESS ERROR
                                  ;*
                                  ;*        BEN 13 SHOULD NOT FAIL.
                                  ;*        IF THE PROCESSOR FAILS TO TRAP IN ALL SECTIONS EITHER TMCC ODD
                                  ;*        ADRS ERR IS NOT GOING LOW OR TMCC BUS ERROR IS NOT GOING LOW.
                                  ;*
                                  ;*        EACH TYPE OF ODD ADDRESS ERROR IS TESTED INDIVIDUALLY TO
                                  ;*        ALLOW MAXIMUM ISOLATION.
```

```
                                   ;*     NOTE:AN ODD ADDRESS ON "KERNEL DATI" CANNOT BE TESTED.
                                   ;*           THIS SIGNAL COMES UP WHEN A TRAP VECTOR IS READ IN FROM THE BUS.
                                   ;;***********************************************************************
022042  000004                     TST43:  SCOPE
022044  012767  022476  157216             MOV     #TST44,NEXTTST   ;SAVE ADDRESS OF NEXT TEST
022052  005005                             CLR     R5               ;INITIALIZE ERROR COUNT
                                   ;NON BYTE REFERENCE ON DATIP
022054  012706  001100                     MOV     #STACK,SP        ;INITIALIZE THE SP
022060  012737  022140  000004             MOV     #1$,@#ERRVEC     ;SETUP ERRVEC
022066  012700  001163                     MOV     #$TMP0+1,R0      ;PUT ODD ADDRESS IN R0
022072  012767  177777  157062             MOV     #-1,$TMP0        ;PUT -1 IN $TMP0 TO SEE IF TEST CHANGES IT
022100  005210                             INC     (R0)             ;EXECUTE ODD ADDRESS INSTRUCTION
                                   ;TRAP DID NOT OCCUR. TRY A DATI.
022102  012737  022126  000004             MOV     #2$,@#ERRVEC     ;SETUP ERRVEC
022110  110030                             MOVB    R0,@(R0)+        ;EXECUTE DATI TO CAUSE ODD ADDR
022112  012737  032654  000004             MOV     #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
022120  005205                             INC     R5               ;SET ERROR COUNT
022122  104265                             ERROR   265              ;NEITHER - BYIN OR DATI CAUSE ODD ADDR
022124  000423                             BR      3$
022126  012737  032654  000004     2$:     MOV     #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
022134  104266                             ERROR   266              ;DATI TRAPPED BUT -BYIN DIDN'T
022136  000416                             BR      3$
                                   ;
                                   ;NON BYTE REFERENCE WORKED. NOW TRY DATI.
022140  005037  177766             1$:     CLR     @#CPUERR         ;CLEAR ERROR REG
022144  012706  001100                     MOV     #STACK,SP        ;INITIALIZE THE SP
022150  012767  022140  156732             MOV     #1$,$LPERR       ;CHANGE LPERR ADDRESS TO THIS SECTION
022156  012737  022174  000004             MOV     #3$,@#ERRVEC     ;SETUP ERRVEC
022164  012700  001163                     MOV     #$TMP0+1,R0      ;PUT ODD ADDR IN R0
022170  110030                             MOVB    R0,@(R0)+        ;EXECUTE ODD ADDRESS INSTRUCTION
022172  000404                             BR      13$
022174  032737  000100  177766     3$:     BIT     #BIT6,@#CPUERR   ;DID ODD ADDR BIT SET?
022202  001037                             BNE     12$              ;BRANCH IF YES
                                   ;TRAP FAILED OR ERROR REG FAILED.  TRY A DATO. (REVERSES INPUTS TO TMCC E5(12,13))
022204  005037  177766             13$:    CLR     @#CPUERR         ;CLEAR ODD ADDR BIT
022210  012706  001077                     MOV     #STACK-1,SP      ;MAKE SP AN ODD ADDRESS
022214  012737  022250  000004             MOV     #4$,@#ERRVEC     ;SETUP ERRVEC
022222  012737  022232  000030             MOV     #5$,@#EMTVEC     ;SETUP EMT VECTOR
022230  104000                             EMT     0                ;EXECUTE DATO TO SP
022232  012706  001100             5$:     MOV     #STACK,SP        ;RESTORE SP
022236  012737  032654  000004             MOV     #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
022244  104267                             ERROR   267              ;BOTH DATI AND DATO FAILED
022246  000446                             BR      6$
022250  012706  001100             4$:     MOV     #STACK,SP        ;RESTORE THE SP
022254  012737  032654  000004             MOV     #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
022262  032737  000100  177766             BIT     #BIT6,@#CPUERR   ;DID ODD ADDR BIT SET?
022270  001401                             BEQ     14$              ;BRANCH IF NO
022272  104270                             ERROR   270              ;DATO WORKS BUT DATI FAILED
022274  104377                     14$:    ERROR   377
022276  000456                             456
022300  000431                             BR      6$
                                   ;
                                   ;EITHER DATI WORKED OR -BYIN AND DATI FAILED. NOW TRY DATO.
022302  012767  022174  156600     12$:    MOV     #3$,$LPERR       ;CHANGE LPERR ADDRESS TO THIS SECTION
022310  012706  001077                     MOV     #STACK-1,SP      ;MAKE SP AN ODD ADDRESS
022314  012737  022364  000004             MOV     #6$,@#ERRVEC     ;SETUP ERRVEC
```

```
022322  012737  022332  000030          MOV     #7$,@#EMTVEC      ;SETUP EMTVEC TO CATCH A FAILURE
022330  104000                          EMT     0                 ;EXECUTE DATO TO ODD ADDRESS
                                ;TRAP FAILED
022332  012706  001076          7$:     MOV     #1076,SP          ;RESET SP
022336  012737  032654  000004          MOV     #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
022344  012737  033536  000030          MOV     #$ERROR,@#EMTVEC  ;RESTORE EMTVEC
022352  005705                          TST     R5
022354  001002                          BNE     11$
022356  104271                          ERROR   271               ;NO TRAP ON DATO BUT DATI & -BYIN OK
022360  000401                          BR      6$
022362  104274                  11$:    ERROR   274               ;NO TRAPS
                                ;
                                ;EITHER DATO WORKED OR -BYIN AND DATI AND DATO FAILED OR DATO
                                ;FAILED BUT -BYIN AND DATI OK. TRY SM357 AND SRC1 DATI.
022364  012737  033536  000030  6$:     MOV     #$ERROR,@#EMTVEC  ;RESTORE EMT VEC
022372  012767  022400  156510          MOV     #8$,$LPERR        ;CHANGE LOOP ERROR ADDR TO THIS SECT.
022400  012706  001076          8$:     MOV     #1076,SP          ;RESET SP
022404  012737  022452  000004          MOV     #9$,@#ERRVEC      ;SETUP ERRVEC
022412  012700  001163                  MOV     #$TMP0+1,R0       ;PUT ODD ADDRESS IN R0
022416  012767  001164  156536          MOV     #$TMP1,$TMP0      ;PUT EVEN ADDRESS IN $TMP0
022424  113001                          MOVB    @(R0)+,R1         ;EXECUTE INSTRUCTION TO CAUSE TRAP
022426  012737  032654  000004          MOV     #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
022434  012767  022476  156532          MOV     #TST44,$ESCAPE    ;SAVE START ADDRESS OF NEXT TEST
022442  012767  022476  156620          MOV     #TST44,NEXTTST    ;SAVE START ADDRESS OF NEXT TEST
022450  104272                          ERROR   272               ;SM357*SRC1 DATI FAILED TO TRAP
                                ;
                                ;SM357*SRC1 DATI WORKS CHECK CPU ERROR REG.
022452  012737  032654  000004  9$:     MOV     #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
022460  032737  000100  177766          BIT     #BIT6,@#CPUERR    ;IS ODD ADDRESS BIT SET?
022466  001001                          BNE     10$               ;BRANCH IF YES
022470  104273                          ERROR   273               ;CPU ERROR REG BIT DOES NOT SET
022472  005037  177766          10$:    CLR     @#CPUERR          ;CLEAR ERROR REG.
                                                                  ;CONTINUE
                                ;;*****************************************************************
                                ;*TEST 44        ILLEGAL INSTRUCTIONS
                                ;*       THIS TEST ENSURES THAT ILLEGAL OP CODES TRAP TO LOCATION 10.
                                ;*       ONLY THOSE OF CODES THAT HAVE A SINGLE BIT THAT
                                ;*       DISTINGUISHES THEM FROM A LEGAL INSTRUCTION WILL BE TESTED.
                                ;;*****************************************************************
022476  000004                  TST44:  SCOPE
022500  012767  023142  156562          MOV     #KBTST,NEXTTST    ;SAVE ADDRESS OF NEXT TEST
022506  012767  022542  156372          MOV     #31$,$LPADR       ;SETUP LOOP ADR
022514  032737  000020  177776          BIT     #BIT4,@#PSW       ;IS T BIT ON?
022522  001404                          BEQ     30$               ;BRANCH IF NO
022524  012737  000360  000012          MOV     #360,@#RESVEC+2   ;SETUP RESVEC PSW
022532  000403                          BR      31$               ;GO DO TEST
                                ;SECTION 1-10 THRU 77
022534  012737  000340  000012  30$:    MOV     #PR7,@#RESVEC+2   ;RESTORE RESVEC PSW
022542  012767  022556  156340  31$:    MOV     #1$,$LPERR        ;SETUP ERROR LOOP
022550  012737  022574  000010          MOV     #2$,@#RESVEC      ;SETUP RESVEC
022556  012706  001100          1$:     MOV     #STACK,SP         ;INITIALIZE THE SP
022562  012746  022570                  MOV     #3$,-(SP)         ;PUT ADDRESS OF 3$ ON STACK
022566  000010                          10                        ;EXECUTE OP CODE 10
                                ;FAILURE-RTT OCCURED
022570  104377                  3$:     ERROR   377               ;OP CODE 10 FAILED TO TRAP
022572  000453                          453
```

```
022574  012767  022610  156306  2$:     MOV     #4$,$LPERR          ;SETUP ERROR LOOP
022602  012737  022616  000010          MOV     #5$,@#RESVEC        ;SETUP RESVEC
022610  000015                  4$:     15                          ;EXECUTE OP CODE 15
                                ;FAILURE
022612  104377                          ERROR   377                 ;OP CODE 15 FAILED TO TRAP
022614  000453                          453
022616  012767  022632  156264  5$:     MOV     #6$,$LPERR          ;SETUP ERROR LOOP
022624  012737  022640  000010          MOV     #7$,@#RESVEC        ;SETUP RESVEC
022632  000025                  6$:     25                          ;EXECUTE OP CODE 25
                                ;FAILURE
022634  104377                          ERROR   377                 ;OP CODE 25 FAILED TO TRAP
022636  000453                          453
022640  012767  022654  156242  7$:     MOV     #8$,$LPERR          ;SETUP ERROR LOOP
022646  012737  022662  000010          MOV     #9$,@#RESVEC        ;SETUP RESVEC
022654  000045                  8$:     45                          ;EXECUTE OP CODE 45
                                ;FAILURE
022656  104377                          ERROR   377
022660  000453                          453                         ;OP CODE 45 FAILED TO TRAP
                                ;
                                ;;**********************************************************************
                                ;SECTION 2-210 THRU 227
022662  012767  022676  156220  9$:     MOV     #10$,$LPERR         ;SETUP ERROR LOOP
022670  012737  022714  000010          MOV     #11$,@#RESVEC       ;SETUP RESVEC
022676  012706  001100          10$:    MOV     #STACK,SP           ;INITIALIZE THE SP
022702  012700  022710                  MOV     #12$,R0             ;SETUP R0 INCASE RTS
022706  000210                          210                         ;EXECUTE OP CODE 210
                                ;FAILURE-RTS EXECUTED
022710  104377                  12$:    ERROR   377                 ;OP CODE 210 FAILED TO TRAP
022712  000453                          453
022714  012767  022730  156166  11$:    MOV     #13$,$LPERR         ;SETUP ERROR LOOP
022722  012737  022746  000010          MOV     #14$,@#RESVEC       ;SETUP RESVEC
022730  012706  001100          13$:    MOV     #STACK,SP           ;INITIALIZE THE SP
022734  012700  022742                  MOV     #15$,R0             ;SETUP R0 TO CATCH RTS
022740  000220                          220                         ;EXECUTE OP CODE 220
                                ;FAILURE-RTS EXECUTED
022742  104377                  15$:    ERROR   377                 ;OP CODE 220 FAILED TO TRAP
022744  000453                          453
                                ;
                                ;;**********************************************************************
                                ;SECTION 3-7000 THRU 7777
022746  012767  022762  156134  14$:    MOV     #27$,$LPERR         ;SETUP ERROR LOOP
022754  012737  022770  000010          MOV     #16$,@#RESVEC       ;SETUP RESVEC
022762  007000                  27$:    7000                        ;EXECUTE OP CODE 7000
                                ;FAILURE
022764  104377                          ERROR   377                 ;OP CODE 7000 FAILED TO TRAP
022766  000453                          453
                                ;
                                ;;**********************************************************************
                                ;SECTION 4-75000 THRU 76000
022770  012767  023004  156112  16$:    MOV     #17$,$LPERR         ;SETUP ERROR LOOP
022776  012737  023012  000010          MOV     #18$,@#RESVEC       ;SETUP RESVEC
023004  075000                  17$:    75000                       ;EXECUTE OP CODE 75000
                                ;FAILURE
023006  104377                          ERROR   377                 ;OP CODE 75000 FAILED TO TRAP
023010  000453                          453
023012  012767  023026  156070  18$:    MOV     #19$,$LPERR         ;SETUP ERROR LOOP
```

```
023020  012737  023034  000010          MOV     #20$,@#RESVEC       ;SETUP RESVEC
023026  076000                  19$:    76000                       ;EXECUTE OP CODE 76000
                                ;FAILURE
023030  104377                          ERROR   377                 ;OP CODE 76000 FAILED TO TRAP
023032  000453                          453
                                ;
                                ;;**************************************************************
                                ;SECTION 5-106400 THRU 106477
023034  012767  023050  156046  20$:    MOV     #21$,$LPERR         ;SETUP ERROR LOOP
023042  012737  023056  000010          MOV     #22$,@#RESVEC       ;SETUP RESVEC
023050  106400                  21$:    106400                      ;EXECUTE OP CODE 106400
                                ;FAILURE
023052  104377                          ERROR   377                 ;OP CODE 106400 FAILED TO TRAP
023054  000453                          453
                                ;
                                ;;**************************************************************
                                ;SECTION 6-106700 THRU 107777
023056  012767  023072  156024  22$:    MOV     #23$,$LPERR         ;SETUP ERROR LOOP
023064  012737  023104  000010          MOV     #24$,@#RESVEC       ;SETUP RESVEC
023072  012706  001100          23$:    MOV     #STACK,SP           ;INITIALIZE THE SP
023076  106700                          106700                      ;EXECUTE OP CODE 106700
                                ;FAILURE
023100  104377                          ERROR   377                 ;OP CODE 106700 FAILED TO TRAP
023102  000453                          453
023104  012767  023120  155776  24$:    MOV     #25$,$LPERR         ;SETUP ERROR LOOP
023112  012737  023126  000010          MOV     #26$,@#RESVEC       ;SETUP RESVEC
023120  107000                  25$:    107000                      ;EXECUTE OP CODE 107000
                                ;FAILURE
023122  104377                          ERROR   377                 ;OP CODE 107000 FAILED TO TRAP
023124  000453                          453
023126  012737  000012  000010  26$:    MOV     #12,@#RESVEC        ;RESTORE RESVEC
023134  012737  000340  000012          MOV     #PR7,@#RESVEC+2     ;RESTORE RESVEC PSW
                                                                    ;CONTINUE

                                ;;**************************************************************
                                ;
                                ; THIS ROUTINE IS USED TO DETERMINE WHICH CPU WE ARE RUNNING ON FOR THE NEXT
                                ; TEST.  THE ROUTINE EXECUTES AN MFPT INSTRUCTION WHICH ONLY EXIESTS ON A KB11-E
                                ; OR KB11-EM.  ON A KB11-B/C OR KB11-CM THE MFPT WILL TRAP.  AFTER DETERMINING
                                ; WHICH CPU IS BEING RUN ON A MESSAGE WILL BE PRINTED.  THIS TEST COULD NOT
                                ; BE RUN BEFORE THE PREVIOUS RESERVED INSTRUCTION TEST.
                                ;
023142  005227  177777          KBTST:  INC     #-1                 ;FIRST TIME?
023146  001032                          BNE     KBDONE              ;BRANCH IF NOT
023150  005037  001272                  CLR     @#KB11E             ;CLEAR KB11E AND KB11EM FLAGS
023154  012737  023174  000010          MOV     #MFPTTR,@#RESVEC    ;SET UP TRAP ADDRESS FOR MFPT AT RESERV VECTOR
023162  000007                          MFPT                        ;EXECUTE MFPT. WILL TRAP ON 1170 (KB11B/C) OR
                                                                    ;KB11-CM
023164  012737  000001  001272          MOV     #1,@#KB11E          ;HERE IF KB11E OR KB11EM. SET FLAG
023172  000403                          BR      ENDKB               ;DONE DETERMINING WHICH CPU

023174                          MFPTTR:                             ;HERE IF MFPT TRAPPED.
023174  012716  023202                  MOV     #ENDKB,(SP)         ;SET UP RETURN ADDRESS FOR RTI
023200  000002                          RTI                         ;RETURN
023202  012737  000012  000010  ENDKB:  MOV     #12,@#RESVEC        ;RESTORE RESERVE VECTOR
```

```
023210  104400  036426                  TYPE    ,MSG1            ;<15><12>CPU UNDER TEST FOUND TO BE A
023214  005737  001272                  TST     @#KB11E          ;IS THIS A KB11-E OR KB11-EM?
023220  001003                          BNE     101$             ;BR IF EITHER ONE
023222  104400  036465                  TYPE    ,MSG3            ;KB11-B/C OR KB11-CM<15><12>
023226  000402                          BR      KBDONE           ;SKIP OTHER MESSAGE
023230  104400  036532          101$:   TYPE    ,MSG5            ;KB11-E OR KB11EM<15><12>
023234                          KBDONE:
                                ;;*****************************************************************
                                ;;*****************************************************************
                                ;*TEST 45          T BIT TRAP
                                ;*
                                ;*        IF BEN 13 FAILS EXECUTION WOULD GO TO BRK.20.
                                ;*        THIS WOULD LOOK LIKE THE TRAP DIDN'T OCCUR.
                                ;*
                                ;*        IF THE TRAP DOESN'T OCCUR THEN EITHER PDRD PS04(1) DOES NOT GET
                                ;*        TO TMCB AS A HIGH OR IT DOES NOT GET TO TMCB E51(10)
                                ;*        AS A LOW OR E51 IS BAD OR IRCD RTT DOES NOT GET TO TMCB AS A HIGH.
                                ;*
                                ;*        THIS TEST ALSO CHECKS THAT PS<08> SET WILL INHIBIT A T BIT TRAP IF
                                ;*        THIS IS A KB11-E OR KB11-EM.
                                ;;*****************************************************************
023234  000004                  TST45:  SCOPE
023236  012767  023504  155730          MOV     #TST46,$ESCAPE   ;SAVE START ADDRESS OF NEXT TEST
023244  012767  023504  156016          MOV     #TST46,NEXTTST   ;SAVE START ADDRESS OF NEXT TEST
                                ;;*****************************************************************
                                ;THIS CODE TURNS THE T BIT OFF IF IT IS ON.
023252  012746  000340                  MOV     #PR7,-(SP)       ;PUT PRIORITY LEVEL 7 ON SP
023256  012746  023272                  MOV     #1$,-(SP)        ;PUT RETURN ADDRESS ON SP
023262  013767  177776  155700          MOV     @#PSW,$TMP3      ;SAVE PSW
023270  000006                          RTT                      ;TURN OFF T BIT
                                ;;*****************************************************************
023272  012767  023300  155610  1$:     MOV     #2$,$LPERR       ;SETUP LOOP ADDRESS
023300  012706  001100          2$:     MOV     #1100,SP         ;INITIALIZE THE SP
023304  012737  023370  000014          MOV     #3$,@#TBITVEC    ;SET UP T BIT VECTOR
023312  012737  023366  000010          MOV     #8$,@#RESVEC     ;SETUP RESVEC
023320  012746  000360                  MOV     #360,-(SP)       ;PUT NEW PSW ON STACK (ENABLE T BIT)
023324  012746  023332                  MOV     #5$,-(SP)        ;PUT RETURN LOCATION ON STACK
023330  000002                          RTI                      ;TURN ON T BIT
023332  012746  000340          5$:     MOV     #PR7,-(SP)       ;SETUP STACK TO
023336  012746  023352                  MOV     #6$,-(SP)        ;TURN OFF T BIT
023342  013767  177776  155636          MOV     @#PSW,$ERPSW     ;SAVE PSW
023350  000006                          RTT                      ;TURN T BIT OFF
023352  032767  000020  155626  6$:     BIT     #BIT4,$ERPSW     ;DID T BIT SET?
023360  001401                          BEQ     7$               ;BRANCH IF NO
023362  104275                          ERROR   275              ;T BIT TRAP FAILED
023364  104276                  7$:     ERROR   276              ;T BIT NEVER SET
023366  104300                  8$:     ERROR   300              ;TRAP VECTOR DECODE FAILED
023370  005767  155676          3$:     TST     KB11E            ;KB11-E OR KB11-EM?
023374  001443                          BEQ     11$              ;DONE IF NOT
023376  012746  000340                  MOV     #PR7,-(SP)       ;PUT PRIORITY LEVEL 7 ON SP
023402  012746  023416                  MOV     #15$,-(SP)       ;PUT RETURN ADDRESS ON SP
023406  013767  177776  155554          MOV     @#PSW,$TMP3      ;SAVE PSW
023414  000006                          RTT                      ;TURN OFF T BIT
023416  012767  023424  155464  15$:    MOV     #12$,$LPERR      ;SET UP LOOP ADDRESS
023424  012706  001100          12$:    MOV     #STACK,SP        ;INIT STACK POINTER
023430  012737  023450  000014          MOV     #10$,@#TBITVEC   ;SET UP T BIT VECTOR
```

```
023436  012746  000760                   MOV     #760,-(SP)        ;NEW PSW ON STACK, ENABLE T BIT AND
                                                                   ;SET PS<08> (SUSPEND BIT)
023442  012746  023504                   MOV     #11$,-(SP)        ;RETURN LOCATION ON STACK. SHOULD GO TO 11$
                                                                   ;INSTEAD OF 10$ ON RTI
023446  000002                           RTI                       ;TURN ON T BIT AND PS<08>
023450  012746  000340           10$:    MOV     #PR7,-(SP)        ;SETUP STACK TO
023454  012746  023470                   MOV     #13$,-(SP)        ;TURN OFF T BIT
023460  013767  177776  155520           MOV     @#PSW,$ERPSW      ;SAVE PSW
023466  000006                           RTT                       ;TURN T BIT OFF
023470  032767  000400  155510   13$:    BIT     #BIT8,$ERPSW      ;DID PS<08> SET?
023476  001401                           BEQ     14$               ;BR IF NOT
023500  104277                           ERROR   277               ;SETTING PS<08> FAILED TO DISABLE T-TRAP
023502  104376                   14$:    ERROR   376               ;PS<08> FAILED TO SET
023504                           11$:                              ;CONTINUE
                                 ;;*****************************************************************
                                 ;*TEST 46       T BIT TRAP AND RTT
                                 ;*
                                 ;*      IF THE INSTRUCTION AFTER THE RTT DOES NOT GET EXECUTED THEN
                                 ;*      EITHER IRCD RTT DOES NOT GO LOW OR IT DOES NOT GET TO
                                 ;*      TMCB E74(11) OR TMCB E74 IS BAD.
                                 ;;*****************************************************************
023504  000004                   TST46:  SCOPE
023506  012767  023556  155554           MOV     #TST47,NEXTTST    ;SAVE ADDRESS OF NEXT TEST
023514  005000                           CLR     R0                ;ENSURE R0 CLEAR
023516  012706  001100                   MOV     #STACK,SP         ;INITIALIZE THE SP
023522  012746  000360                   MOV     #360,-(SP)        ;PUT NEW PSW ON STACK (ENABLE T BIT)
023526  012746  023542                   MOV     #1$,-(SP)         ;PUT PC ON STACK
023532  012737  023546  000014           MOV     #2$,@#TBITVEC     ;SETUP T BIT VECTOR
023540  000006                           RTT                       ;TURN T BIT ON WITH RTT
023542  012700  000001           1$:     MOV     #1,R0             ;THIS SHOULD EXECUTE
023546  022700  000001           2$:     CMP     #1,R0             ;DID MOV INSTRUCTION EXECUTE?
023552  001401                           BEQ     TST47             ;;BRANCH IF YES
023554  104301                           ERROR   301               ;RTT DID NOT DISABLE T BIT
                                 ;;*****************************************************************
                                 ;*TEST 47       PRIORITY ARBITRATION
                                 ;*
                                 ;*      THIS TEST ASSURES THAT EACH NECESSARY INPUT TO AN HONOR FLAG
                                 ;*      CAN DISABLE THAT FLAG.
                                 ;*
                                 ;*      EACH SECTION WILL PERFORM A SETUP SO THAT A TIGHT ERROR LOOP
                                 ;*      CAN BE OBTAINED.
                                 ;*
                                 ;*      THE FOLLOWING IS A TABLE OF CONTENTS OF THIS TEST:
                                 ;*              SECTION NUMBER  LEVEL UNDER TEST        DISABLING FUNCTION
                                 ;*                     1            PIR 1                      BR  4
                                 ;*                     2            PIR 1                   SL YELLOW
                                 ;*                     3            PIR 2                   SL YELLOW
                                 ;*                     4            PIR 3                   SL YELLOW
                                 ;*                     5            BR  4                      PIR 4
                                 ;*                     6            BR  4                      PIR 5
                                 ;*                     7            BR  4                      BR  5
                                 ;*                     8            BR  4                      PIR 6
                                 ;*                     9            BR  4                      PIR 7
                                 ;*                    10            PIR 4                      BR  5
                                 ;*                    11            PIR 4                      BR  6
                                 ;*                    12            PIR 4                   SL YELLOW
```

```
                                  ;*                 13           BR  5               PIR 5
                                  ;*                 14           BR  5               PIR 6
                                  ;*                 15           BR  5               PIR 7
                                  ;*                 16           PIR 5               BR  6
                                  ;*                 17           PIR 5            SL YELLOW
                                  ;*                 18           BR  6               PIR 6
                                  ;*                 19           BR  6               PIR 7
                                  ;*                 20           PIR 6            SL YELLOW
                                  ;*                 21           PIR 7            SL YELLOW
                                  ;;*************************************************************
023556 000004                     TST47:  SCOPE
023560 012767 026366 155502               MOV     #TST50,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
023566 012767 003706 155310               MOV     #^D1990,$ICNT   ;SETUP ITTERATION COUNT
023574 012767 023610 155304               MOV     #103$,$LPADR    ;SETUP LOOP ADDRESS
023602 012737 032640 000014               MOV     #$RTRN,@#TBITVEC ;RESTORE T BIT VEC
023610 005005                     103$:   CLR     R5              ;CLEAR BR5 DISABLE FLAG
023612 005004                             CLR     R4              ;CLEAR BR6 DISABLE FLAG
023614 005003                             CLR     R3              ;CLEAR BR4 DISABLE FLAG
023616 032737 000020 001170               BIT     #BIT4,@#$TMP3   ;WAS T BIT ON?
023624 001417                             BEQ     71$             ;BRANCH IF NO
023626 012737 000360 000066               MOV     #360,@#TPVEC+2
023634 012737 000360 000242               MOV     #360,@#PIRQVEC+2
023642 012737 000360 000006               MOV     #360,@#ERRVEC+2
023650 012746 000360                      MOV     #360,-(SP)      ;SET UP STACK PSW
023654 012746 023706                      MOV     #72$,-(SP)      ;PUT RETURN ADDRESS ON STACK
023660 000006                             RTT                     ;TURN T BIT ON
023662 000411                             BR      72$
023664 012737 000340 000066       71$:    MOV     #PR7,@#TPVEC+2  ;PUT PRIORITY 7 IN PRINTER VECTOR
023672 012737 000340 000006               MOV     #PR7,@#ERRVEC+2 ;RESTORE ERRVEC PSW
023700 012737 000340 000242               MOV     #PR7,@#PIRQVEC+2 ;PUT PRIORITY 7 IN PIRQ VECTOR
023706 032737 000400 177570       72$:    BIT     #SW8,@#SWR      ;SWITCH 8 ON?
023714 001006                             BNE     100$            ;BRANCH IF YES
023716 032737 000040 177570               BIT     #SW5,@#SWR      ;IS BR 5 TESTING DISABLED?
023724 001402                             BEQ     100$            ;BRANCH IF NO
023726 005205                     102$:   INC     R5              ;SET BR 5 DISABLE FLAG
023730 000403                             BR      101$            ;CONTINUE
023732 005767 155266              100$:   TST     INTER5          ;IS THERE A BR 5 DEVICE?
023736 001773                             BEQ     102$            ;BRANCH IF NO
                                  ;;*************************************************************
                                  ;SECTION 1 - BR4 AND PIR1
023740 032737 000400 177570       101$:   BIT     #SW8,@#SWR      ;SWITCH 8 ON?
023746 001006                             BNE     68$             ;BRANCH IF YES
023750 032737 000020 177570               BIT     #SW4,@#SWR      ;IS BR4 TESTING DISABLED?
023756 001402                             BEQ     68$             ;BRANCH IF NO
023760 005203                             INC     R3              ;SET BR4 FLAG
023762 000430                             BR      2$              ;GO TO NEXT SECTION
023764 012767 024006 155116       68$:    MOV     #1$,$LPERR      ;SETUP ERROR LOOP
023772 012737 024044 000064               MOV     #2$,@#TPVEC     ;SETUP PRINTER VECTOR
024000 012737 024032 000240               MOV     #3$,@#PIRQVEC   ;SETUP PIRQ VECTOR
024006 012706 001076              1$:     MOV     #1076,SP        ;INITIALIZE THE SP
024012 000237                             SPL     7               ;ENSURE CPU AT LEVEL 7
024014 052737 001000 177772               BIS     #BIT9,@#PIRQ    ;SET PIR LEVEL 1
024022 004767 002204                      JSR     PC,LEVEL4       ;GO GET BR4
024026 000230                             SPL     0               ;SET CPU AT ZERO
024030 000240                             NOP                     ;USED WITH SPL
                                  ;FAILURE PIR CAME THRU
```

```
024032  005037  177772          3$:     CLR     @#PIRQ          ;CLEAR THE PIRQ
024036  005077  155100                  CLR     @$TPS           ;CLEAR THE PRINTER FLAG
024042  104302                          ERROR   302             ;PIR 1 DID NOT DISABLE ON BR
                                ;;*****************************************************************
                                ;SECTION 2 - PIR 1 AND SL YELLOW
024044  005077  155072          2$:     CLR     @$TPS           ;CLEAR PRINTER INT FLAG
024050  012767  024076  155032          MOV     #4$,$LPERR      ;SETUP ERROR LOOP
024056  012737  024134  000004          MOV     #5$,@#ERRVEC    ;SETUP LOCATION 4
024064  012737  024114  000240          MOV     #6$,@#PIRQVEC   ;SETUP PIRQ VECTOR
024072  005037  177772                  CLR     @#PIRQ          ;CLEAR LEVEL 1
024076  012706  000376          4$:     MOV     #376,SP         ;SETUP THE SP
024102  052737  001000  177772          BIS     #BIT9,@#PIRQ    ;SET LEVEL 1
024110  000230                          SPL     0               ;SET CPU AT ZERO
024112  011616                          MOV     (SP),(SP)       ;EXECUTE YELL ZONE INSTR
                                ;FAILURE, PIR CAME THRU
024114  012706  001100          6$:     MOV     #STACK,SP       ;RESET SP
024120  012737  032654  000004          MOV     #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
024126  005037  177772                  CLR     @#PIRQ          ;CLEAR LEVEL 1
024132  104303                          ERROR   303             ;PIR CAME THRU ON YELLOW ZONE
                                ;
                                ;;*****************************************************************
                                ;SECTION 3 - PIR 2 AND SL YELLOW
024134  005037  177772          5$:     CLR     @#PIRQ          ;CLEAR LEVEL 1
024140  012767  024162  154742          MOV     #7$,$LPERR      ;SETUP ERROR LOOP
024146  012737  024220  000004          MOV     #8$,@#ERRVEC    ;SETUP LOCATION 4
024154  012737  024200  000240          MOV     #9$,@#PIRQVEC   ;SETUP PIRQ VECTOR
024162  012706  000376          7$:     MOV     #376,SP         ;SETUP THE SP
024166  052737  002000  177772          BIS     #BIT10,@#PIRQ   ;SET LEVEL 2
024174  000231                          SPL     1               ;SET CPU AT LEVEL 1
024176  011616                          MOV     (SP),(SP)       ;EXECUTE TRAP CAUSING INSTR
                                ;FAILURE, PIR CAME THRU
024200  012706  001100          9$:     MOV     #STACK,SP       ;RESET SP
024204  012737  032654  000004          MOV     #CPUSPUR,@#ERRVEC ;RESET ERRVEC
024212  005037  177772                  CLR     @#PIRQ          ;CLEAR LEVEL 2
024216  104304                          ERROR   304             ;PIR 2 CAME THRU ON YELLOW ZONE
                                ;
                                ;;*****************************************************************
                                ;SECTION 4 - PIR 3 AND YEL ZONE
024220  005037  177772          8$:     CLR     @#PIRQ          ;CLEAR LEVEL 2
024224  012767  024246  154656          MOV     #10$,$LPERR     ;SETUP ERROR LOOP
024232  012737  024304  000004          MOV     #11$,@#ERRVEC   ;SETUP ERR VECTOR
024240  012737  024264  000240          MOV     #12$,@#PIRQVEC  ;SETUP PIRQ VECTOR
024246  012706  000376          10$:    MOV     #376,SP         ;SETUP THE SP
024252  052737  004000  177772          BIS     #BIT11,@#PIRQ   ;SET LEVEL 3
024260  000232                          SPL     2               ;SET CPU AT LEVEL 2
024262  011616                          MOV     (SP),(SP)       ;EXECUTE TRAP CAUSING INSTR
                                ;FAILURE, PIR CAME THRU
024264  012706  001100          12$:    MOV     #STACK,SP       ;RESET SP
024270  012737  032654  000004          MOV     #CPUSPUR,@#ERRVEC ;RESET LOCATION 4
024276  005037  177772                  CLR     @#PIRQ          ;CLEAR LEVEL 3
024302  104305                          ERROR   305             ;PIR 3 CAME THRU ON YELLOW ZONE
                                ;;*****************************************************************
                                ;SECTION 5- PIR4 AND BR4
024304  012706  001100          11$:    MOV     #STACK,SP       ;RESTORE THE SP
024310  005703                          TST     R3              ;IS BR4 TESTING DISABLED?
024312  001402                          BEQ     70$             ;BRANCH IF NO
```

```
024314  000167  000410                   JMP     29$                 ;SKIP BR4 SECTIONS
024320  012767  024342  154562   70$:    MOV     #13$,$LPERR         ;SETUP ERROR LOOP
024326  012737  024376  000240           MOV     #15$,@#PIRQVEC      ;SETUP PIRQ VEC
024334  012737  024364  000064           MOV     #14$,@#TPVEC        ;SETUP TPVEC
024342  012706  001100           13$:    MOV     #STACK,SP           ;INITIALIZE THE SP
024346  012737  010000  177772           MOV     #BIT12,@#PIRQ       ;SET PIR LEVEL 4
024354  004767  001652                   JSR     PC,LEVEL4           ;GO GET BR 4
024360  000233                           SPL     3                   ;LOWER CPU
024362  000240                           NOP                         ;REQUIRED BECAUSE OF SPL

                                 ;FAILURE, BR4 CAME THRU
024364  005077  154552           14$:    CLR     @$TPS               ;CLEAR PRINTER INT FLAG
024370  005037  177772                   CLR     @#PIRQ              ;CLEAR LEVEL 4
024374  104306                           ERROR   306                 ;BR4 CAME THRU ON PIR4

                                 ;
                                 ;;*************************************************************
                                 ;SECTION 6- PIR 5 AND BR4
024376  012767  024420  154504   15$:    MOV     #16$,$LPERR         ;SETUP ERROR LOOP
024404  012737  024442  000064           MOV     #17$,@#TPVEC        ;SETUP BR4 VEC
024412  012737  024454  000240           MOV     #18$,@#PIRQVEC      ;SETUP PIRQ VEC
024420  012706  001100           16$:    MOV     #STACK,SP           ;INITIALIZE THE SP
024424  012737  020000  177772           MOV     #BIT13,@#PIRQ       ;SET PIR LEVEL 5
024432  004767  001574                   JSR     PC,LEVEL4           ;GO GET BR4
024436  000233                           SPL     3                   ;SET CPU AT 3
024440  000240                           NOP                         ;REQUIRED BECAUSE OF SPL

                                 ;FAILURE, BR4 CAME THRU
024442  005077  154474           17$:    CLR     @$TPS               ;CLEAR BR4 INT FLAG
024446  005037  177772                   CLR     @#PIRQ              ;CLEAR PIR5
024452  104307                           ERROR   307                 ;BR4 CAME THRU ON PIR5


                                 ;
                                 ;;*************************************************************
                                 ;SECTION 7- BR5 AND BR4
024454  005037  177772           18$:    CLR     @#PIRQ              ;CLEAR PIR LEVEL 5
024460  005077  154456                   CLR     @$TPS               ;CLEAR PRINTER INT FLAG
024464  005705                           TST     R5                  ;IS BR 5 TESTING DISABLED?
024466  001040                           BNE     23$                 ;BRANCH IF YES
024470  012767  024536  154412   20$:    MOV     #21$,$LPERR         ;SETUP ERROR LOOP
024476  012737  024556  000064           MOV     #22$,@#TPVEC        ;SETUP BR4 VECTOR
024504  016700  154516                   MOV     INT5VEC,R0          ;GET BR5 VECTOR
024510  012720  024570                   MOV     #23$,(R0)+          ;PUT ADDRESS OF 23$ IN VECTOR
024514  032737  000020  177776           BIT     #BIT4,@#PSW         ;IS T BIT ON?
024522  001403                           BEQ     73$                 ;BRANCH IF NO
024524  012710  000360                   MOV     #360,(R0)           ;SETUP VECTOR PSW
024530  000402                           BR      21$
024532  012710  000340           73$:    MOV     #PR7,(R0)           ;PUT PR7 IN VECTOR+2
024536  012706  001100           21$:    MOV     #STACK,SP           ;INITIALIZE THE SP
024542  004767  001474                   JSR     PC,LEVEL5           ;GO GET BR5
024546  004767  001460                   JSR     PC,LEVEL4           ;GO GET BR4
024552  000233                           SPL     3                   ;SET CPU AT LEVEL 3
024554  000240                           NOP                         ;REQUIRED BECAUSE OF SPL

                                 ;FAILURE, BR4 CAME IN
024556  004767  001526           22$:    JSR     PC,KILBR5           ;GET RID OF BR 5
024562  005077  154354                   CLR     @$TPS               ;CLEAR BR4 INT ENABLE
024566  104310                           ERROR   310                 ;BR4 CAME THRU ON BR5

                                 ;
                                 ;;*************************************************************
                                 ;SECTION 8- PIR6 AND BR4
024570  004767  001514           23$:    JSR     PC,KILBR5         ;GET RID OF BR 5
024574  012767  024616  154306           MOV     #24$,$LPERR       ;SETUP ERROR LOOP
024602  012737  024640  000064           MOV     #25$,@#TPVEC      ;SETUP BR4 INTERRUPT VECTOR.
024610  012737  024652  000240           MOV     #26$,@#PIRQVEC    ;SETUP PIRQ VECTOR
024616  012706  001076           24$:    MOV     #1076,SP          ;INITIALIZE THE SP
024622  012737  040000  177772           MOV     #BIT14,@#PIRQ     ;SET PIR LEVEL 6
024630  004767  001376                   JSR     PC,LEVEL4         ;GO GET BR4
024634  000233                           SPL     3                 ;SET CPU AT LEVEL 3
024636  000240                           NOP                       ;REQUIRED BECAUSE OF SPL
                                 ;FAILURE, BR4 CAME IN
024640  005037  177772           25$:    CLR     @#PIRQ            ;CLEAR PIR LEVEL 6
024644  005077  154272                   CLR     @$TPS             ;CLEAR PRINTER INTER FLAG
024650  104311                           ERROR   311               ;BR4 CAME IN ON PIR6
                                 ;
                                 ;;**************************************************************
                                 ;SECTION 9- PIR 7 AND BR4
024652  012767  024674  154230   26$:    MOV     #27$,$LPERR       ;SETUP ERROR LOOP
024660  012737  024716  000064           MOV     #28$,@#TPVEC      ;SETUP BR4 VECTOR
024666  012737  024730  000240           MOV     #29$,@#PIRQVEC    ;SETUP PIRQ VECTOR
024674  012706  001076           27$:    MOV     #1076,SP          ;SETUP THE SP
024700  012737  100000  177772           MOV     #BIT15,@#PIRQ     ;SET PIR LEVEL 7
024706  004767  001320                   JSR     PC,LEVEL4         ;GO GET BR4
024712  000233                           SPL     3                 ;LOWER CPU TO 3
024714  000240                           NOP                       ;REQUIRED BECAUSE OF SPL
                                 ;FAILURE, BR4 CAME IN
024716  005077  154220           28$:    CLR     @$TPS             ;CLEAR PRINTER INTERR FLAG
024722  005037  177772                   CLR     @#PIRQ            ;CLEAR PIR LEVEL 7
024726  104312                           ERROR   312               ;BR4 CAME IN ON PIR7
                                 ;
                                 ;;**************************************************************
                                 ;SECTION 10- BR5 AND PIR4
024730  005077  154206           29$:    CLR     @$TPS             ;CLEAR BR4 INTERR FLAG
024734  005037  177772                   CLR     @#PIRQ            ;CLEAR PIRQ LEVEL 7
024740  005705                           TST     R5                ;IS THERE A BR5 DEVICE?
024742  001041                           BNE     32$               ;BRANCH TO SECTION 11 IF NO
024744  012767  025012  154136           MOV     #30$,$LPERR       ;SETUP ERROR LOOP
024752  012737  025034  000240           MOV     #31$,@#PIRQVEC    ;SETUP PIRQ VECTOR
024760  016700  154242                   MOV     INT5VEC,R0        ;GET ADDR OF BR 5 VECTOR
024764  012720  025046                   MOV     #32$,(R0)+        ;SETUP BR5 VECTOR
024770  032737  000020  177776           BIT     #BIT4,@#PSW       ;IS T BIT ON?
024776  001403                           BEQ     74$               ;BRANCH IF NO
025000  012710  000360                   MOV     #360,(R0)
025004  000462                           BR      75$
025006  012710  000340           74$:    MOV     #PR7,(R0)
025012  012706  001100           30$:    MOV     #STACK,SP         ;SETUP THE SP
025016  012737  010000  177772           MOV     #BIT12,@#PIRQ     ;SET PIR4
025024  004767  001212                   JSR     PC,LEVEL5         ;GO GET BR5
025030  000233                           SPL     3                 ;LOWER CPU TO LEVEL 3
025032  000240                           NOP                       ;REQUIRED BECAUSE OF SPL
                                 ;FAILURE, PIR4 CAME IN
025034  004767  001250           31$:    JSR     PC,KILBR5         ;GET RID OF BR 5
025040  005037  177772                   CLR     @#PIRQ            ;CLEAR PIR LEVEL 4
025044  104313                           ERROR   313               ;PIR4 CAME IN ON BR5
                                 ;
                                 ;;**************************************************************
```

```
                                        ;SECTION 11- BR6 AND PIR4
 025046  005037  177772                 32$:    CLR     @#PIRQ          ;CLEAR PIR LEVEL 4
 025052  004767  001232                         JSR     PC,KILBR5       ;GET RID OF BR 5
 025056  032737  000400  177570                 BIT     #SW8,@#SWR      ;SWITCH 8 ON?
 025064  001004                                 BNE     80$             ;BRANCH IF YES
 025066  032737  000100  177570                 BIT     #SW6,@#SWR      ;IS BR6 TESTING DISABLED?
 025074  001003                                 BNE     33$             ;BRANCH IF YES
 025076  005767  154130                 80$:    TST     INTER6          ;IS THERE A BR6 DEVICE?
 025102  001002                                 BNE     34$             ;BRANCH IF YES
 025104  005204                         33$:    INC     R4              ;SET BR 6 DISABLE FLAG
 025106  000441                                 BR      37$             ;GO TO SECTION 12
 025110  012767  025156  153772         34$:    MOV     #35$,$LPERR     ;SETUP ERROR LOOP
 025116  012737  025200  000240                 MOV     #36$,@#PIRQVEC  ;SETUP PIR VECTOR
 025124  016701  154104                         MOV     INT6VEC,R1      ;GET BR 6 VECTOR
 025130  012721  025212                         MOV     #37$,(R1)+      ;PU ADDRESS OF 37$ IN VECTOR
 025134  032737  000020  177776                 BIT     #BIT4,@#PSW     ;IS T BIT ON?
 025142  001403                                 BEQ     75$             ;BRANCH IF NO
 025144  012711  000360                         MOV     #360,(R1)
 025150  000402                                 BR      35$
 025152  012711  000340                 75$:    MOV     #PR7,(R1)       ;PUT PRIORITY OF 7 IN VECTOR+2
 025156  012706  001100                 35$:    MOV     #STACK,SP       ;INITIALIZE THE SP
 025162  012737  010000  177772                 MOV     #BIT12,@#PIRQ   ;SET PIR LEVEL 4
 025170  004767  001056                         JSR     PC,LEVEL6       ;GO GET BR6
 025174  000233                                 SPL     3               ;LOWER CPU TO LEVEL 3
 025176  000240                                 NOP                     ;REQUIRED BECAUSE OF SPL
                                        ;FAILURE, PIR4 CAME IN
 025200  005077  154032                 36$:    CLR     @INT6ST         ;CLEAR BR6 INTERR FLAG
 025204  005037  177772                         CLR     @#PIRQ          ;CLEAR PIR LEVEL 4
 025210  104314                                 ERROR   314             ;PIR4 CAME IN ON BR6
                                        ;
                                        ;;****************************************************************
                                        ;SECTION 12- PIR4 AND STACK LIMIT YELLOW
 025212  005077  154020                 37$:    CLR     @INT6ST         ;CLEAR BR6 INTERR. FLAG
 025216  005037  177772                         CLR     @#PIRQ          ;CLEAR LEVEL 4
 025222  012767  025244  153660                 MOV     #38$,$LPERR     ;SETUP ERROR LOOP
 025230  012737  025262  000240                 MOV     #39$,@#PIRQVEC  ;SETUP PIRQ VECTOR
 025236  012737  025302  000004                 MOV     #40$,@#ERRVEC   ;SETUP YELL ZONE VECTOR
 025244  012706  000376                 38$:    MOV     #376,SP         ;SETUP THE SP
 025250  052737  010000  177772                 BIS     #BIT12,@#PIRQ   ;SET LEVEL 4
 025256  000233                                 SPL     3               ;SET CPU AT LEVEL 3
 025260  011616                                 MOV     (SP),(SP)       ;EXECUTE TRAP CAUSING INSTR
                                        ;FAILURE, PIR 4 CAME THRU
 025262  012706  001100                 39$:    MOV     #STACK,SP       ;RESET THE SP
 025266  012737  032654  000004                 MOV     #CPUSPUR,@#ERRVEC ;RESTORE ERR VEC
 025274  005037  177772                         CLR     @#PIRQ          ;CLEAR PIR LEVEL 4
 025300  104315                                 ERROR   315             ;PIR4 CAME IN ON SL YELLOW
                                        ;
                                        ;;****************************************************************
                                        ;SECTION 13- BR5 AND PIR5
 025302  012706  001100                 40$:    MOV     #STACK,SP       ;RESTORE THE SP
 025306  012737  032654  000004                 MOV     #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
 025314  005037  177772                         CLR     @#PIRQ          ;CLEAR PIR 4
 025320  005705                                 TST     R5              ;IS BR5 DISABLED?
 025322  001105                                 BNE     49$             ;BRANCH IF YES TO SECTION 16
 025324  012767  025346  153556                 MOV     #41$,$LPERR     ;SETUP ERROR LOOP
 025332  012777  025370  153666                 MOV     #42$,@INT5VEC   ;SETUP BR5 VECTOR
```

```
4539 025340 012737 025402 000240             MOV     #43$,@#PIRQVEC  ;SETUP PIRQ VECTOR
4540 025346 012706 001076           41$:     MOV     #1076,SP        ;INITIALIZE THE SP
4541 025352 012737 020000 177772             MOV     #BIT13,@#PIRQ   ;SET PIR LEVEL 5
4542 025360 004767 000656                    JSR     PC,LEVEL5       ;GO GET BR5
```

```
025364  000234                                  SPL     4                   ;LOWER CPU TO LEVEL 4
025366  000240                                  NOP                         ;REQUIRED BECAUSE OF SPL
                                        ;FAILURE, BR5 CAME IN
025370  004767  000714                  42$:    JSR     PC,KILBR5           ;GET RID OF BR 5
025374  005037  177772                          CLR     @#PIRQ              ;CLEAR PIR5
025400  104316                                  ERROR   316                 ;BR5 CAME IN ON PIR5
                                        ;
                                        ;;**************************************************************
                                        ;SECTION 14- BR5 AND PIR6
025402  012767  025424  153500          43$:    MOV     #44$,$LPERR         ;SETUP ERROR LOOP
025410  012777  025446  153610                  MOV     #45$,@INT5VEC       ;SETUP BR5 VEC
025416  012737  025460  000240                  MOV     #46$,@#PIRQVEC      ;SETUP PIRQ VECTOR
025424  012706  001076                  44$:    MOV     #1076,SP            ;INITIALIZE THE SP
025430  012737  040000  177772                  MOV     #BIT14,@#PIRQ       ;SET PIR LEVEL 6
025436  004767  000600                          JSR     PC,LEVEL5           ;GO GET BR5
025442  000234                                  SPL     4                   ;SET CPU AT LEVEL 4
025444  000240                                  NOP                         ;REQUIRED BECAUSE OF SPL
                                        ;FAILURE, BR5 CAME IN
025446  004767  000636                  45$:    JSR     PC,KILBR5           ;GET RID OF BR5
025452  005037  177772                          CLR     @#PIRQ              ;CLEAR PIR LEVEL 6
025456  104317                                  ERROR   317                 ;BR5 CAME IN ON PIR6
                                        ;
                                        ;;**************************************************************
                                        ;SECTION 15- BR5 AND PIR7
025460  012767  025502  153422          46$:    MOV     #47$,$LPERR         ;SETUP ERROR LOOP
025466  012777  025524  153532                  MOV     #48$,@INT5VEC       ;SETUP BR5 VECTOR
025474  012737  025536  000240                  MOV     #49$,@#PIRQVEC      ;SETUP PIRQ VECTOR
025502  012706  001076                  47$:    MOV     #1076,SP            ;INITIALIZE THE SP
025506  012737  100000  177772                  MOV     #BIT15,@#PIRQ       ;SET PIR LEVEL 7
025514  004767  000522                          JSR     PC,LEVEL5           ;GO GET BR5
025520  000234                                  SPL     4                   ;ALLOW BRQ
025522  000240                                  NOP                         ;REQUIRED BECAUSE OF SPL
                                        ;FAILURE, BR5 CAME IN
025524  004767  000560                  48$:    JSR     PC,KILBR5           ;GET RID OF BR5
025530  005037  177772                          CLR     @#PIRQ              ;CLEAR PIR LEVEL 7
025534  104320                                  ERROR   320                 ;BR5 CAME IN ON PIR7
                                        ;
                                        ;;**************************************************************
                                        ;SECTION 16- PIR5 AND BR6
025536  004767  000546                  49$:    JSR     PC,KILBR5           ;GET RID OF BR 5
025542  005704                                  TST     R4                  ;IS BR6 TESTING DISABLED?
025544  001030                                  BNE     52$                 ;BRANCH IF YES TO SECTION 17
025546  012767  025570  153334                  MOV     #50$,$LPERR         ;SETUP ERROR LOOP
025554  012737  025614  000240                  MOV     #51$,@#PIRQVEC      ;SETUP PIR VECTOR
025562  012777  025626  153444                  MOV     #52$,@INT6VEC       ;SETUP BR6 VECTOR
025570  012706  001076                  50$:    MOV     #1076,SP            ;INITIALIZE THE SP
025574  012737  020000  177772                  MOV     #BIT13,@#PIRQ       ;SET IR LEVEL 5
025602  004767  000444                          JSR     PC,LEVEL6           ;GO GET BR6
025606  000234                                  SPL     4                   ;ALLOW BRQ
025610  000240                                  NOP                         ;REQUIRED BECAUSE OF SPL
                                        ;FAILURE, PIR 5 CAME IN
025612  000000                                  HALT                        ;DEBUG ONLY
025614  005077  153416                  51$:    CLR     @INT6ST             ;CLEAR BR6 INTERR FLAG
025620  005037  177772                          CLR     @#PIRQ              ;CLEAR PIR LEVEL 5
025624  104321                                  ERROR   321                 ;PIR 5 CAME IN ON BR6
                                        ;
                                        ;;*******************************************************************
                                        ;SECTION 17- PIR 5 AND STACK LIMIT YELLOW
025626  005077  153404                  52$:    CLR     @INT6ST         ;CLEAR BR6 INTERR FLAG
025632  012767  025654  153250                  MOV     #53$,$LPERR     ;SETUP ERROR LOOP
025640  012737  025672  000240                  MOV     #54$,@#PIRQVEC  ;SETUP PIRQVEC
025646  012737  025712  000004                  MOV     #55$,@#ERRVEC   ;SETUP LOCATION 4
025654  012706  000376                  53$:    MOV     #376,SP         ;SETUP THE SP
025660  052737  020000  177772                  BIS     #BIT13,@#PIRQ   ;SET LEVEL 5
025666  000234                                  SPL     4               ;SET CPU AT 4
025670  011616                                  MOV     (SP),(SP)       ;EXECUTE YEL ZONE INSTR
                                        ;FAILURE, PIR5 CAME IN ON SL YELLOW
025672  005037  177772                  54$:    CLR     @#PIRQ          ;CLEAR LEVEL 5
025676  012737  032654  000004                  MOV     #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
025704  012706  001100                          MOV     #STACK,SP       ;RESET THE SP
025710  104322                                  ERROR   322             ;PIR5 CAME IN ON SL YELLOW
                                        ;
                                        ;;*******************************************************************
                                        ;SECTION 18- BR6 AND PIR6
025712  012706  001100                  55$:    MOV     #STACK,SP       ;RESTORE THE SP
025716  005704                                  TST     R4              ;IS BR6 TESTING DISABLED?
025720  001056                                  BNE     61$             ;BRANCH IF YES TO SECTION 20
025722  012767  025744  153160                  MOV     #56$,$LPERR     ;SETUP ERROR LOOP
025730  012777  025766  153276                  MOV     #57$,@INT6VEC   ;SETUP BR6 INTERR VECTOR
025736  012737  026000  000240                  MOV     #58$,@#PIRQVEC  ;SETUP PIR VECTOR
025744  012706  001076                  56$:    MOV     #1076,SP        ;SETUP THE SP
025750  012737  040000  177772                  MOV     #BIT14,@#PIRQ   ;SET PIR LEVEL 6
025756  004767  000270                          JSR     PC,LEVEL6       ;GO GET BR6
025762  000235                                  SPL     5               ;LOWER CPU TO 5
025764  000240                                  NOP                     ;REQUIRED BECAUSE OF SPL
                                        ;FAILURE, BR6 CAME IN ON PIR6
025766  005077  153244                  57$:    CLR     @INT6ST         ;CLEAR BR6 INTERR FLAG
025772  005037  177772                          CLR     @#PIRQ          ;CLEAR LEVEL 6
025776  104323                                  ERROR   323             ;BR6 CAME IN ON PIR6
                                        ;
                                        ;;*******************************************************************
                                        ;SECTION 19- BR6 AND PIR7
026000  012767  026022  153102          58$:    MOV     #59$,$LPERR     ;SETUP ERROR LOOP
026006  012777  026044  153220                  MOV     #60$,@INT6VEC   ;SETUP BR6 VECTOR
026014  012737  026056  000240                  MOV     #61$,@#PIRQVEC  ;SETUP PIRQ VECTOR
026022  012706  001076                  59$:    MOV     #1076,SP        ;SETUP THE SP
026026  012737  100000  177772                  MOV     #BIT15,@#PIRQ   ;SET PIR 7
026034  004767  000212                          JSR     PC,LEVEL6       ;GO GET BR6
026040  000235                                  SPL     5               ;LOWER CPU TO 5
026042  000240                                  NOP                     ;REQUIRED BECAUSE OF SPL
                                        ;FAILURE, BR6 CAME IN ON PIR7
026044  005077  153166                  60$:    CLR     @INT6ST         ;CLEAR BR6 INTERR FLAG
026050  005037  177772                          CLR     @#PIRQ          ;CLEAR LEVEL 7
026054  104324                                  ERROR   324             ;BR6 CAME IN ON PIR7
                                        ;
                                        ;;*******************************************************************
                                        ;SECTION 20- PIR6 AND SL YELLOW
026056  012767  026100  153024          61$:    MOV     #62$,$LPERR     ;SETUP LOOP ADDRESS
026064  012737  026116  000240                  MOV     #63$,@#PIRQVEC  ;SETUP PIRQ VECTOR
026072  012737  026130  000004                  MOV     #64$,@#ERRVEC   ;SETUP LOCATION 4
026100  012706  000376                  62$:    MOV     #376,SP         ;SETUP THE SP
026104  052737  040000  177772                  BIS     #BIT14,@#PIRQ   ;SET LEVEL 6
```

```
 026112  000235                               SPL     5                  ;SET CPU AT LEVEL 5
 026114  011616                               MOV     (SP),(SP)          ;EXECUTE YELL ZONE INSTR
                                      ;FAILURE, PIR 6 CAME IN ON SL YELLOW
 026116  005037  177772               63$:    CLR     @#PIRQ             ;CLEAR PIR 6
 026122  012706  001100                       MOV     #STACK,SP          ;RESTORE THE SP
 026126  104325                               ERROR   325                ;PIR 6 CAME IN ON SL YELLOW
                                      ;
                                      ;;*************************************************************
                                      ;SECTION 21-PIR7 AND STACK LIMIT YELLOW
 026130  005077  153102               64$:    CLR     @INT6ST            ;ENSURE BR6 FLAG CLEAR
 026134  012767  026156  152746               MOV     #65$,$LPERR        ;SETUP ERROR LOOP
 026142  012737  026174  000240               MOV     #66$,@#PIRQVEC     ;SETUP PIRQ VECTOR
 026150  012737  026206  000004               MOV     #67$,@#ERRVEC      ;SETUP THE ERROR VECTOR
 026156  012706  000376               65$:    MOV     #376,SP            ;SETUP THE SP
 026162  052737  100000  177772               BIS     #BIT15,@#PIRQ      ;SET LEVEL 7
 026170  000236                               SPL     6                  ;SET UP AT 6
 026172  011616                               MOV     (SP),(SP)          ;CAUSE YELL ZONE
                                      ;FAILURE, PIR 7 CAME IN ON SL YELLOW
 026174  005037  177772               66$:    CLR     @#PIRQ             ;CLEAR LEVEL 7
 026200  012706  001100                       MOV     #STACK,SP          ;RESTORE THE SP
 026204  104326                               ERROR   326                ;PIR7 CAME IN ON SL YELLOW
                                      ;
                                      ;END OF TEST
 026206  012706  001100               67$:    MOV     #STACK,SP          ;RESET THE SP
 026212  005037  177772                       CLR     @#PIRQ             ;CLEAR LEVEL 7
 026216  012737  032654  000004               MOV     #CPUSPUR,@#ERRVEC  ;RESTORE LOCATION 4
 026224  005037  177766                       CLR     @#CPUERR           ;CLEAR OUT ERROR REG
 026230  000456                               BR      TST50              ;;CONTINUE
                                      ;
                                      ;ROUTINES TO GET BUS REQUESTS
 026232  052777  000100  152702       LEVEL4: BIS     #100,@$TPS         ;GET A BR 4
 026240  000445                               BR      WAIT               ;GO WAIT
 026242  012777  000311  152760       LEVEL5: MOV     #311,@INT5ST       ;START BR 5 INTERRUPT
 026250  000441                               BR      WAIT               ;GO WAIT
 026252  026727  152756  000100       LEVEL6: CMP     INT6VEC,#100       ;IS BR6 DEVICE KW11-L?
 026260  001004                               BNE     1$                 ;BRANCH IF NO
 026262  012777  000100  152746               MOV     #BIT6,@INT6ST      ;SET INTERR BIT
 026270  000431                               BR      WAIT               ;GO WAIT
 026272  012737  000001  172544       1$:     MOV     #1,@#PLKC          ;SET KW11-P COUNTER
 026300  012777  000105  152730               MOV     #105,@INT6ST       ;SET INTERR BIT
 026306  000422                               BR      WAIT
 026310  012777  026346  152710       KILBR5: MOV     #2$,@INT5VEC       ;SET UP INTER 5 VEC
 026316  042777  000100  152704               BIC     #BIT6,@INT5ST      ;ENSURE INTERRUPT FLAG CLEAR
 026324  005037  177772                       CLR     @#PIRQ             ;ENSURE PIRQ REG CLEAR
 026330  005077  152606                       CLR     @$TPS
 026334  000230                               SPL     0                  ;LET BR 5 COME IN
 026336  005000                               CLR     R0
 026340  005200                       3$:     INC     R0
 026342  001376                               BNE     3$
 026344  000406                               BR      ENDGET             ;DON'T CHANGE STACK
 026346  062706  000004               2$:     ADD     #4,SP              ;RESTORE THE SP
 026352  000403                               BR      ENDGET
 026354  005000                       WAIT:   CLR     R0                 ;WAIT FOR
 026356  005200                       2$:     INC     R0                 ;THE INTERRUPT
 026360  001376                               BNE     2$                 ;TO COME IN
 026362  000237                       ENDGET: SPL     7
```

```
026364  000207                          RTS     PC                      ;RETURN
                                ;;*************************************************************
                                ;*TEST 50       GPR SET 1 SELECT TEST
                                ;*
                                ;*      THIS TEST FIRST ENSURES THAT PSW BIT 11 SETS AND CLEARS.
                                ;*      IT THEN ENSURES THAT GRAC GDREG SET 1 AND GSREG SET 1 GOES
                                ;*      HIGH FOR THE MUX SELECTS LL,LH,AND HL.
                                ;*      MUX SELECT HH WILL BE TESTED IN SUPERVISOR MODE.
                                ;;*************************************************************
026366  000004                  TST50:  SCOPE
026370  012767  027016  152576          MOV     #TST51,$ESCAPE          ;SAVE START ADDRESS OF NEXT TEST
026376  012767  027016  152664          MOV     #TST51,NEXTTST          ;SAVE START ADDRESS OF NEXT TEST
026404  052737  004000  177776          BIS     #BIT11,@#PSW            ;SET REG SET SELECT BIT
026412  032737  004000  177776          BIT     #BIT11,@#PSW            ;DID BIT 11 SET?
026420  001004                          BNE     20$                     ;BRANCH IF YES
026422  042737  004000  177776          BIC     #BIT11,@#PSW
026430  104327                          ERROR   327                     ;EITHER PSW BIT 11 DOES NOT SET OR TMCF
                                                                        ;CLK HI PS DOES NOT GO LOW OR PSW BIT
                                                                        ;11 DOES NOT GET TO OR THRU THE DMUX.
026432  105037  177777          20$:    CLRB    @#PSW+1                 ;CLEAR BIT 11
026436  032737  004000  177776          BIT     #BIT11,@#PSW            ;DID BIT 11 CLEAR?
026444  001402                          BEQ     1$                      ;BRANCH IF YES
026446  104377                          ERROR   377
026450  000457                          457                             ;PSW BIT 11 DID NOT CLEAR
                                ;;*************************************************************
                                ;UPAD 5 TEST
026452  005067  152516          1$:     CLR     $ESCAPE
026456  012767  026464  152424          MOV     #10$,$LPERR
026464  012702  177777          10$:    MOV     #-1,R2                  ;SETUP R2
026470  052737  004000  177776          BIS     #BIT11,@#PSW            ;SELECT REG SET 1
026476  005002                          CLR     R12                     ;CLEAR R12
026500  042737  004000  177776          BIC     #BIT11,@#PSW            ;GO BACK TO REG SET 0
026506  005702                          TST     R2                      ;DID R2 CLEAR?
026510  001002                          BNE     2$                      ;BRANCH IF NO
026512  104330                          ERROR   330                     ;CLEAR R10 ACTUALLY CLEARED R2
026514  000403                          BR      3$
026516  020202                  2$:     CMP     R2,R2                   ;WAS R2 SOURCE AFFECTED BY R12?
026520  001401                          BEQ     3$                      ;BRANCH IF NO
026522  104331                          ERROR   331                     ;R2 SRC WAS AFFECTED BY CLR R12
                                ;;*************************************************************
                                ;UPAD 0 TEST
026524  012767  026532  152356  3$:     MOV     #11$,$LPERR
026532  005002                  11$:    CLR     R2                      ;SETUP R2
026534  052737  004000  177776          BIS     #BIT11,@#PSW            ;GO TO REG SET 1
026542  012202                          MOV     (R12)+,R12              ;EXECUTE A UPAD 0 INSTRUCTION
026544  042737  004000  177776          BIC     #BIT11,@#PSW            ;COME BACK TO SET 0.
026552  005702                          TST     R2                      ;DID DESTINATION CHANGE
026554  001402                          BEQ     4$                      ;BRANCH IF NO
026556  104332                          ERROR   332                     ;R2 DST GOT CHANGED ON (R12)+
026560  000403                          BR      5$
026562  020202                  4$:     CMP     R2,R2                   ;DID SRC R2 GET CHANGED?
026564  001401                          BEQ     5$                      ;BRANCH IF NO
026566  104333                          ERROR   333                     ;R2 SRC GOT CHANGED ON (R12)+
                                ;;*************************************************************
                                ;UPAD 2 TEST
026570  012767  026576  152312  5$:     MOV     #12$,$LPERR
```

```
026576  012705  026660          12$:    MOV     #9$,R5          ;SETUP R5
026602  052737  004000  177776          BIS     #BIT11,@#PSW    ;GO TO SET 1
026610  012705  026634                  MOV     #6$,R15         ;SETUP R15
026614  020527  026634                  CMP     R15,#6$         ;IS THERE STUCK BITS IN R15 SRC?
026620  001401                          BEQ     7$              ;BRANCH IF NO
026622  000475                          BR      TST51           ;;CAN'T DO THIS TEST, GO TO STUCK BIT TEST
026624  020505                  7$:     CMP     R15,R15         ;IS THERE STUCK BITS IN R15 DST?
026626  001401                          BEQ     8$              ;BRANCH IF NO
026630  000472                          BR      TST51           ;;CAN'T DO THIS TEST, GO TO STUCK BIT TEST
026632  006400                  8$:     MARK    0               ;EXECUTE A UPAD 2 INSTRUCTION
026634  026705  177774          6$:     CMP     6$,R15          ;DID R15 DST GET LOADED?
026640  001425                          BEQ     16$             ;BRANCH IF YES
026642  042737  004000  177776          BIC     #BIT11,@#PSW    ;GO BACK TO SET 0
026650  012706  001100                  MOV     #STACK,SP       ;RESTORE THE SP
026654  104334                          ERROR   334             ;R15 DST BAD AFTER UPAD2
026656  000416                          BR      16$
026660  022705  026660          9$:     CMP     #9$,R15         ;IS DST ALSO BAD?
026664  001407                          BEQ     15$             ;BRANCH IF YES
026666  042737  004000  177776          BIC     #BIT11,@#PSW
026674  012706  001100                  MOV     #STACK,SP       ;RESTORE THE SP
026700  104335                          ERROR   335             ;R15 SRC DOES NOT SELECT ON UPAD2
026702  000404                          BR      16$
026704  042737  004000  177776  15$:    BIC     #BIT11,@#PSW
026712  104336                          ERROR   336             ;PDRD PS11 DOES NOT GET TO GRAC
                                ;;*************************************************************
                                ;TEST GRAB E19(2 & 3)
026714  012706  001100          16$:    MOV     #STACK,SP       ;RESET THE SP
026720  012767  026726  152162          MOV     #14$,$LPERR
026726  042737  004000  177776  14$:    BIC     #BIT11,@#PSW    ;GO BACK TO SET 0
026734  005004                          CLR     R4              ;ENSURE R4 CLEAR
026736  052737  004000  177776          BIS     #BIT11,@#PSW    ;GO TO SET 1
026744  005204                          INC     R14             ;INCREMENT R14
026746  042737  004000  177776          BIC     #BIT11,@#PSW    ;GO BACK TO SET 0
026754  005704                          TST     R4              ;WAS R4 AFFECTED?
026756  001401                          BEQ     17$             ;BRANCH IF NO
026760  104341                          ERROR   341             ;GRAB DST SET 1 DID NOT GO LOW ON R14
                                ;;*************************************************************
                                ;TEST GRAB E18(2 & 3)
026762  012767  026770  152120  17$:    MOV     #13$,$LPERR
026770  052737  004000  177776  13$:    BIS     #BIT11,@#PSW    ;GO TO SET 1
026776  005004                          CLR     R14             ;ENSURE EVEN ADDRESS IN R14
027000  012404                          MOV     (R14)+,R14      ;EXECUTE A UPAD 0
027002  042737  004000  177776          BIC     #BIT11,@#PSW    ;GO BACK TO SET 0
027010  005704                          TST     R4              ;WAS R4 AFFECTED?
027012  001401                          BEQ     TST51           ;;BRANCH IF NO
027014  104342                          ERROR   342             ;GRAB SRC SET 1 DID NOT GO LOW ON R14

                                ;;*************************************************************
                                ;;*TEST 51        REGISTER SET 1 STUCK BIT TEST
                                ;;*
                                ;;*       THIS TEST ENSURES THAT ALL BITS IN GPR'S R10 THRU R15 WORK
                                ;;*************************************************************
027016  000004                  TST51:  SCOPE
027020  012767  027232  152146          MOV     #TST52,$ESCAPE  ;SAVE START ADDRESS OF NEXT TEST
027026  012767  027232  152234          MOV     #TST52,NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
027034  005067  152122                  CLR     $TMP0           ;INITIALIZE PASS COUNT
```

```
027040  052737  004000  177776          BIS     #BIT11,@#PSW     ;GO TO REG SET 1
027046  012737                  4$:     MOV     (PC)+,@(PC)+     ;SAVE EXPECTED VALUE
027050  125252                  5$:     .WORD   125252
027052  001164                          .WORD   $TMP1            ;ADDRESS OF $TEMP1
027054  012700                          MOV     (PC)+,R10        ;PUT PATTERN IN R10
027056  125252                  6$:     .WORD   125252
027060  022700                          CMP     (PC)+,R10        ;IS R10 DST OK?
027062  125252                  7$:     .WORD   125252
027064  001040                          BNE     1$               ;BRANCH IF NO
027066  020027                          CMP     R10,(PC)+        ;IS R10 SRC OK?
027070  125252                  8$:     .WORD   125252
027072  001034                          BNE     2$               ;BRANCH IF NO
027074  010001                          MOV     R10,R11
027076  020001                          CMP     R10,R11          ;IS R11 DST OK?
027100  001032                          BNE     1$               ;BRANCH IF NO
027102  020100                          CMP     R11,R10          ;IS R11 SRC OK?
027104  001027                          BNE     2$               ;BRANCH IF NO
027106  010102                          MOV     R11,R12
027110  020002                          CMP     R10,R12          ;IS R12 DST OK?
027112  001025                          BNE     1$               ;BRANCH IF NO
027114  020200                          CMP     R12,R10          ;IS R12 SRC OK?
027116  001022                          BNE     2$               ;BRANCH IF NO
027120  010003                          MOV     R10,R13
027122  020003                          CMP     R10,R13          ;IS R13 DST OK
027124  001020                          BNE     1$               ;BRANCH IF NO
027126  020300                          CMP     R13,R10          ;IS R13 SRC OK?
027130  001015                          BNE     2$               ;BRANCH IF NO
027132  010004                          MOV     R10,R14
027134  020004                          CMP     R10,R14          ;IS R14 DST OK?
027136  001013                          BNE     1$               ;BRANCH IF NO
027140  020400                          CMP     R14,R10          ;IS R14 SRC OK?
027142  001010                          BNE     2$               ;BRANCH IF NO
027144  010005                          MOV     R10,R15          ;
027146  020005                          CMP     R10,R15          ;IS R15 DST OK
027150  001006                          BNE     1$               ;BRANCH IF NO
027152  020500                          CMP     R15,R10          ;IS R15 SRC OK
027154  001410                          BEQ     3$               ;BRANCH IF YES
027156  042737  004000  177776          BIC     #BIT11,@#PSW
027164  104337                  2$:     ERROR   337              ;BAD BITS IN GPR SET 1 SRC
027166  042737  004000  177776  1$:     BIC     #BIT11,@#PSW
027174  104340                          ERROR   340              ;BAD BITS IN GPR SET 1 DST
027176  005767  151760          3$:     TST     $TMP0            ;IS THIS FIRST PASS?
027202  001013                          BNE     TST52            ;;BRANCH IF NO
027204  005267  151752                  INC     $TMP0            ;SET PASS COUNT
027210  005167  177634                  COM     5$               ;GO TO
027214  005167  177636                  COM     6$               ;COMPLIMENT
027220  005167  177636                  COM     7$               ;PATTERN
027224  005167  177640                  COM     8$
027230  000706                          BR      4$               ;GO TEST COMPLIMENT PATTERN

                                ;;*****************************************************************
                                ;*TEST 52         PSW HIGH BYTE BIT TEST
                                ;*
                                ;*        THIS TEST ENSURES THAT THE PRESENT AND PREVIOUS MODE BITS OF THE PSW
                                ;*        CAN BE SET AND CLEARED AND THAT THEY ARE NOT STUCK TOGETHER.
                                ;;*****************************************************************
```

```
027232  000004                          TST52:  SCOPE
027234  012767  027402  152026                  MOV     #TST53,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
027242  112737  000120  177777                  MOVB    #120,@#PSW+1    ;SET TO SUPER & PREVIOUS SUPER
027250  122737  000120  177777                  CMPB    #120,@#PSW+1    ;IS IT OK?
027256  001412                                  BEQ     1$              ;BRANCH IF YES
027260  012767  050000  151674                  MOV     #50000,$TMP0    ;SAVE EXPECTED VALUE
027266  013767  177776  151712                  MOV     @#PSW,$ERPSW    ;SAVE ERROR VALUE
027274  042767  000377  151704                  BIC     #377,$ERPSW     ;MASK OFF HIGH BYTE
027302  104343                                  ERROR   343             ;PATTERN FAILED
027304  112737  000350  177777          1$:     MOVB    #350,@#PSW+1    ;SET COMPLIMENT  PATTERN
027312  122737  000350  177777                  CMPB    #350,@#PSW+1    ;IS IT OK?
027320  001412                                  BEQ     2$              ;BRANCH IF YES
027322  012767  164000  151632                  MOV     #164000,$TMP0   ;SAVE EXPECTED VALUE
027330  013767  177776  151650                  MOV     @#PSW,$ERPSW    ;SAVE ERROR VALUE
027336  042767  000377  151642                  BIC     #377,$ERPSW     ;MASK OFF HIGH BYTE
027344  104344                                  ERROR   344             ;PATTERN FAILED
027346  105037  177777                  2$:     CLRB    @#PSW+1         ;CLEAR ALL BITS
027352  105737  177777                          TSTB    @#PSW+1         ;DID THEY ALL CLEAR?
027356  001411                                  BEQ     TST53           ;;BRANCH IF YES
027360  005067  151576                          CLR     $TMP0           ;SAVE EXPECTED VALUE
027364  013767  177776  151614                  MOV     @#PSW,$ERPSW    ;SAVE ERROR VALUE
027372  042767  000377  151606                  BIC     #377,$ERPSW     ;MASK OFF HIGH BYTE
027400  104345                                  ERROR   345             ;ALL BITS IN PSW DID NOT CLEAR

                                        ;;*************************************************************
                                        ;*TEST 53        SP SELECTION TEST IN SUPER AND USER MODE
                                        ;*
                                        ;*       THIS TEST ENSURES THAT THE CORRECT STACK POINTERS ARE
                                        ;*       SELECTED IN SUPERVISOR AND USER MODE
                                        ;;*************************************************************
027402  000004                          TST53:  SCOPE
027404  012767  027652  151656                  MOV     #TST54,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
                                        ;UPAD 5-SSP
027412  012767  027420  151470                  MOV     #5$,$LPERR      ;SETUP ERROR LOOP
027420  012706  001100                  5$:     MOV     #STACK,KSP      ;SETUP THE KERNAL SP
027424  052737  040000  177776                  BIS     #BIT14,@#PSW    ;GO TO SUPER MODE
027432  005006                                  CLR     SSP             ;CLEAR THE SUPER SP (UPAD 5)
027434  042737  040000  177776                  BIC     #BIT14,@#PSW    ;GO BACK TO KERNEL MODE
027442  005706                                  TST     KSP             ;DID THE KSP CHANGE?
027444  001003                                  BNE     1$              ;BRANCH IF NO
027446  012706  001100                          MOV     #STACK,KSP      ;RESTORE THE KSP
027452  104346                                  ERROR   346             ;SUPER SP DOES NOT SELECT ON UPAD 5

                                        ;
                                        ;UPAD 0-SSP
027454  012767  027462  151426          1$:     MOV     #6$,$LPERR      ;SETUP ERROR LOOP
027462  012706  001100                  6$:     MOV     #STACK,KSP      ;INITIALIZE THE SP
027466  052737  040000  177776                  BIS     #BIT14,@#PSW    ;GO TO SUPER MODE
027474  012706  001776                          MOV     #1776,SSP       ;SETUP SUPER SP
027500  012606                                  MOV     (SSP)+,SSP      ;EXECUTE A UPAD 0 TYPE OPERATION
027502  042737  040000  177776                  BIC     #BIT14,@#PSW    ;GO BACK TO KERNEL
027510  020627  001100                          CMP     KSP,#STACK      ;DID KSP CHANGE?
027514  001403                                  BEQ     2$              ;BRANCH IF NO
027516  012706  001100                          MOV     #STACK,KSP      ;RESTORE THE KSP
027522  104347                                  ERROR   347             ;SUPER SP DOES NOT SELECTON UPAD 0

                                        ;
                                        ;UPAD 5-USP
```

```
027524  012767  027532  151356  2$:     MOV     #7$,$LPERR          ;SETUP ERROR LOOP
027532  052737  040000  177776  7$:     BIS     #BIT14,@#PSW        ;GO TO SUPER MODE
027540  012706  177777                  MOV     #-1,SSP             ;SET THE SSP TO A NON-ZERO NUMBER
027544  052737  100000  177776          BIS     #BIT15,@#PSW        ;GO TO USER MODE
027552  005006                          CLR     USP                 ;CLEAR R17
027554  042737  100000  177776          BIC     #BIT15,@#PSW        ;GO BACK TO SUPER MODE
027562  005706                          TST     SSP                 ;DID SSP CLEAR?
027564  001003                          BNE     3$                  ;BRANCH IF NO
027566  105037  177777                  CLRB    @#PSW+1             ;GO BACK TO KERNEL
027572  104350                          ERROR   350                 ;USER SP DOES NOT SELECT ON UPAD 5
                                ;
                                ;UPAD 0-USP
027574  012767  027602  151306  3$:     MOV     #8$,$LPERR          ;SETUP ERROR LOOP
027602  052737  040000  177776  8$:     BIS     #BIT14,@#PSW        ;GO TO SUPER MODE
027610  005006                          CLR     SSP                 ;ENSURE SSP CLEAR
027612  052737  100000  177776          BIS     #BIT15,@#PSW        ;GO TO USER MODE
027620  012706  001776                  MOV     #1776,USP           ;SET USP TO EVEN NUMBER
027624  012606                          MOV     (USP)+,USP          ;EXECUTE A UPAD 0 TYPE INSTURCTION
027626  042737  100000  177776          BIC     #BIT15,@#PSW        ;GO BACK TO SUPER
027634  005706                          TST     SSP                 ;DID SSP CHANGE?
027636  001403                          BEQ     4$                  ;BRANCH IF NO
027640  105037  177777                  CLRB    @#PSW+1             ;GO BACK TO KERNEL
027644  104351                          ERROR   351                 ;USER SP DOES NOT SELECT ON UPAD 0
027646  105037  177777          4$:     CLRB    @#PSW+1             ;GO BACK TO KERNAL
                                                                    ;CONTINUE

                                ;;*****************************************************************
                                ;*TEST 54        SUPER AND USER SP BIT TEST
                                ;*
                                ;*       THIS TEST ENSURES THAT THE SUPERVISOR AND USER STACK POINTERS
                                ;*       DON'T HAVE ANY STUCK BITS.
                                ;;*****************************************************************
027652  000004                  TST54:  SCOPE
027654  012767  030016  151312          MOV     #TST55,$ESCAPE      ;SAVE START ADDRESS OF NEXT TEST
027662  012767  030016  151400          MOV     #TST55,NEXTTST      ;SAVE START ADDRESS OF NEXT TEST
027670  005067  151270                  CLR     $TMP1               ;CLEAR LOOP COUNT
027674  052737  040000  177776          BIS     #BIT14,@#PSW        ;GO TO SUPER MODE
027702  012706                  4$:     MOV     (PC)+,SP            ;PUT PATTERN IN SP
027704  052525                  6$:     .WORD   52525               ;DATA PATTERN
027706  022706                          CMP     (PC)+,SP            ;IS DST OK?
027710  052525                  7$:     .WORD   52525               ;DATA PATTERN
027712  001403                          BEQ     1$                  ;BRANCH IF YES
027714  105037  177777                  CLRB    @#PSW+1             ;GO BACK TO KERNEL
027720  104352                          ERROR   352                 ;DST FAILED
027722  020627                  1$:     CMP     SP,(PC)+            ;IS SRC OK?
027724  052525                  8$:     .WORD   52525               ;DATA PATTERN
027726  001403                          BEQ     2$                  ;BRANCH IF YES
027730  105037  177777                  CLRB    @#PSW+1             ;GO BACK TO KERNEL
027734  104353                          ERROR   353                 ;SRC FAILED
027736  005737  177776          2$:     TST     @#PSW               ;IS BIT 15 SET?
027742  100404                          BMI     3$                  ;BRANCH IF YES
027744  052737  100000  177776          BIS     #BIT15,@#PSW        ;GO TO USER MODE
027752  000753                          BR      4$                  ;GO CHECK USER STACK POINTER
027754  005767  151204          3$:     TST     $TMP1               ;IS THIS SECOND PASS?
027760  001014                          BNE     5$                  ;BRANCH IF YES
027762  005167  177716                  COM     6$                  ;CHANGE TEST
```

```
027766  005167  177716                    COM     7$               ;TO COMPLIMENT
027772  005167  177726                    COM     8$               ;PATTERN
027776  042737  100000  177776            BIC     #BIT15,@#PSW     ;GO BACK TO SUPER MODE
030004  005267  151154                    INC     $TMP1            ;SET PASS COUNT
030010  000734                            BR      4$               ;GO TEST COMPLIMENT PATTERN
030012  105037  177777            5$:     CLRB    @#PSW+1          ;GO BACK TO KERNAL
                                                                   ;CONTINUE

                                  ;;*************************************************************
                                  ;*TEST 55         MTP*DMO*DF6*PREVIOUS MODE(SUPER*USER)
                                  ;*
                                  ;*        THIS TEST ENSURES THAT THE CORRECT SP'S ARE SELECTED WHEN EXECUTING
                                  ;*        A MTP WITH DIFFERENT PREVIOUS MODE BITS SELECTED.
                                  ;;*************************************************************
030016  000004                    TST55:  SCOPE
030020  012767  030252  151242            MOV     #TST56,NEXTTST   ;SAVE ADDRESS OF NEXT TEST
                                  ;
                                  ;SECTION 1-POP THE KSP AND PUT IT IN THE SSP
030026  012706  001100                    MOV     #STACK,KSP       ;SETUP THE KERNEL SP
030032  012746  177777                    MOV     #-1,-(KSP)       ;PUT -1 ON THE STACK
030036  005000                            CLR     R0               ;CLEAR ERROR COUNT
030040  052737  010000  177776            BIS     #BIT12,@#PSW     ;SET PREVIOUS MODE SUPER
030046  106606                            MTPD    SP               ;EXECUTE INSTRUCTION UNDER TEST
030050  022706  001100                    CMP     #STACK,KSP       ;DID THE KSP DST COME OUT OK?
030054  001401                            BEQ     2$               ;BRANCH IF YES
030056  005200                            INC     R0               ;SET THE ERROR COUNT
030060  020627  001100            2$:     CMP     KSP,#STACK       ;DID THE KSP SRC COME OUT OK?
030064  001410                            BEQ     3$               ;BRANCH IF YES
030066  012706  001100                    MOV     #STACK,KSP       ;RESTORE THE SP
030072  005700                            TST     R0               ;DID DST ALSO FAIL?
030074  001002                            BNE     4$               ;BRANCH IF YES
030076  104354                            ERROR   354              ;KSP SRC WAS CHANGED
030100  000431                            BR      9$
030102  104355                    4$:     ERROR   355              ;BOTH KSP SRC AND DST CHANGED
030104  000427                            BR      9$
030106  005700                    3$:     TST     R0               ;DID KSP DST CHANGE
030110  001404                            BEQ     5$               ;BRANCH IF NO
030112  012706  001100                    MOV     #STACK,KSP       ;RESTORE THE SP
030116  104356                            ERROR   356              ;KSP DST WAS CHANGED
030120  000421                            BR      9$
030122  052737  040000  177776    5$:     BIS     #BIT14,@#PSW     ;GO TO SUPER MODE
030130  022706  177777                    CMP     #-1,SSP          ;DID SSP LOAD OK?
030134  001412                            BEQ     6$               ;BRANCH IF YES
030136  010667  151012                    MOV     SSP,$REG0        ;SAVE THE SSP
030142  005006                            CLR     SSP              ;FOR LOOPING
030144  042737  040000  177776            BIC     #BIT14,@#PSW     ;GO BACK TO KERNEL
030152  012767  177777  150776            MOV     #-1,$REG1        ;SAVE EXPECTED VALUE
030160  104357                            ERROR   357              ;SSP DID NOT LOAD PROPERLY
                                  ;
                                  ;SECTION 2-POP THE KSP AND PUT IT IN THE USP
030162  005006                    6$:     CLR     SSP              ;ENSURE SSP CLEAR FOR ITTERATIONS
030164  042737  040000  177776    9$:     BIC     #BIT14,@#PSW     ;GO BACK TO KERNEL
030172  012767  030200  150710            MOV     #7$,$LPERR       ;SET UP ERROR LOOP
030200  012706  001100            7$:     MOV     #STACK,KSP       ;SETUP THE SP
030204  012746  177777                    MOV     #-1,-(KSP)       ;PUT -1 ON THE KERNEL STACK
030210  052737  030000  177776            BIS     #30000,@#PSW     ;SET PREVIOUS MODE USER
```

```
030216  106606                          MTPD    SP                ;EXECUTE INSTRUCTION UNDER TEST
030220  052737  140000  177776          BIS     #140000,@#PSW     ;GO TO USER MODE
030226  020627  177777                  CMP     USP,#-1           ;DID USER SP GET LOADED?
030232  001404                          BEQ     8$                ;BRANCH IF YES
030234  005006                          CLR     USP               ;CLEAR USER SP
030236  105037  177777                  CLRB    @#PSW+1           ;GO BACK TO KERNEL
030242  104360                          ERROR   360               ;USER SP DID NOT LOAD ON MTP
030244  005006                  8$:     CLR     USP               ;ENSURE USP CLEAR, IF ITTERATING
030246  105037  177777                  CLRB    @#PSW+1           ;GO BACK TO KERNEL
                                                                  ;CONTINUE
                                ;;*****************************************************************
                                ;*TEST 56        MFP*DMO*DF6*PREVIOUS MODE SUPER
                                ;*
                                ;*          THE ONLY POSSIBLE WAY THIS TEST CAN FAIL IS THAT
                                ;*          IRCC DMO (MFP+MTP) DOES NOT GO HIGH ON MFP.
                                ;*           THIS WILL ONLY HAPPEN IF THE IR DECODE ROM HAS
                                ;*          A BAD FIELD (R(MFP+MTP)).
                                ;;*****************************************************************
030252  000004                  TST56:  SCOPE
030254  012767  030400  150712          MOV     #TST57,$ESCAPE    ;SAVE START ADDRESS OF NEXT TEST
030262  012767  030400  151000          MOV     #TST57,NEXTTST    ;SAVE START ADDRESS OF NEXT TEST
                                ;
                                ;PUSH THE SSP ONTO THE KERNEL STACK
030270  012767  030334  150610          MOV     #1$,$LPADR        ;SETUP LOOP ADR
030276  012767  030334  150604          MOV     #1$,$LPERR        ;SETUP ERROR LOOP
030304  013767  177776  150656          MOV     @#PSW,$TMP3       ;SAVE PSW
030312  032737  000020  177776          BIT     #BIT4,@#PSW       ;IS T BIT ON?
030320  001405                          BEQ     1$                ;BRANCH IF NO
030322  012746  000340                  MOV     #PR7,-(SP)        ;PUT NEW PSW ON STACK
030326  012746  030334                  MOV     #1$,-(SP)         ;PUT RETURN ADDR ON STACK
030332  000006                          RTT                       ;TURN T BIT OFF
030334  052737  040000  177776  1$:     BIS     #BIT14,@#PSW      ;GO TO SUPER MODE
030342  005006                          CLR     SSP               ;CLEAR THE SUPER SP
030344  105037  177777                  CLRB    @#PSW+1           ;GO BACK TO KERNEL
030350  012706  001100                  MOV     #STACK,KSP        ;SETUP THE KSP
030354  012766  177777  177776          MOV     #-1,-2(KSP)       ;PUT KNOWN VALUE ON STACK
030362  052737  010000  177776          BIS     #BIT12,@#PSW      ;SET PREVIOUS MODE SUPER
030370  106506                          MFPD    SP                ;EXECUTE INSTRUCTION UNDER TEST
030372  005716                          TST     (KSP)             ;DID STACK GET SUPER SP?
030374  001401                          BEQ     TST57             ;;BRANCH IF YES
030376  104361                          ERROR   361               ;IR DECODE ROM BAD

                                ;;*****************************************************************
                                ;*TEST 57        UPAD 7 IN USER MODE
                                ;*
                                ;*          THIS TEST ENSURES THAT A UPAD 7(OCCURS IN RTI) CAUSES THE USER
                                ;*          STACK POINTER TO BE USED TO FETCH THE NEW PS AND PC.
                                ;*
                                ;*          IF PDRD PS14(1) DOES NOT GET TO THE GSAM(ON GRAC)
                                ;*          THE TEST WILL BLOW UP.
                                ;;*****************************************************************
030400  000004                  TST57:  SCOPE
030402  012767  030610  150660          MOV     #TST60,NEXTTST    ;SAVE ADDRESS OF NEXT TEST
030410  012706  001100                  MOV     #STACK,KSP        ;SETUP THE KERNAL SP
030414  052737  040000  177776          BIS     #BIT14,@#PSW      ;GO TO SUPER MODE
030422  012706  001064                  MOV     #1064,SSP         ;SETUP THE SSP
```

```
030426  012716  030466                      MOV     #1$,(SSP)          ;PUT ADDRESS OF 1$ ON SUPER STACK
030432  012766  140340  000002              MOV     #140340,2(SSP)     ;PUT NEW PSW ON SUPER STACK
030440  052737  100000  177776              BIS     #BIT15,@#PSW       ;GO TO USER MODE
030446  012706  001060                      MOV     #1060,USP          ;SETUP THE USP
030452  012716  030536                      MOV     #2$,(USP)          ;PUT ADDRESS OF 2$ ON USER STACK
030456  012766  140340  000002              MOV     #140340,2(USP)     ;PUT NEW PSW ON USER STACK
030464  000002                              RTI                        ;EXECUTE INSTRUCTION THAT USES UPAD 7
                                    ;FAILURE, SUPER STACK POINTER WAS READ
030466  022706  001070              1$:     CMP     #1070,USP          ;DID USP DST GET WRITTEN?
030472  001004                              BNE     3$                 ;BRANCH IF NO
030474  105037  177777                      CLRB    @#PSW+1            ;GO BACK TO KERNEL
030500  104362                              ERROR   362                ;SSP WAS READ AND USP WAS WRITTEN
030502  000415                              BR      2$
030504  042737  100000  177776      3$:     BIC     #BIT15,@#PSW       ;GO TO SUPER MODE
030512  022706  001070                      CMP     #1070,SSP          ;WAS THE SSP WRITTEN>
030516  001004                              BNE     4$                 ;BRANCH IF NO
030520  105037  177777                      CLRB    @#PSW+1            ;GO BACK TO KERNEL
030524  104363                              ERROR   363                ;SSP WAS READ AND WRITTEN
030526  000403                              BR      2$
030530  105037  177777              4$:     CLRB    @#PSW+1            ;GO BACK TO KERNEL
030534  104364                              ERROR   364                ;DON'T KNOW WHAT HAPPENED
                                    ;READ OK,NOW CHECK WRITE
030536  052737  140000  177776      2$:     BIS     #140000,@#PSW      ;SETUP PSW
030544  022706  001064                      CMP     #1064,USP          ;DID THE USP GET WRITTEN?
030550  001415                              BEQ     5$                 ;BRANCH IF YES
030552  042737  100000  177776              BIC     #BIT15,@#PSW       ;GO TO SUPER MODE
030560  022706  001064                      CMP     #1064,SSP          ;DID THE SSP GET WRITTEN?
030564  001004                              BNE     6$                 ;BRANCH IF NO
030566  105037  177777                      CLRB    @#PSW+1            ;GO TO KERNEL MODE
030572  104365                              ERROR   365                ;USP WAS READ BUT SSP WAS WRITTEN
030574  000403                              BR      5$
030576  105037  177777              6$:     CLRB    @#PSW+1            ;GO TO KERNEL MODE
030602  104366                              ERROR   366                ;USP WAS READ BUT REG 7 WAS WRITTEN
030604  105037  177777              5$:     CLRB    @#PSW+1            ;GO TO KERNEL
                                                                       ;CONTINUE



                                    ;;*************************************************************
                                    ;*TEST 60       SPL*SUPERVISOR MODE
                                    ;*
                                    ;*      THIS TEST ENSURES THAT SPL DOES NOT LOAD THE PSW IN SUPER+USER MODE.
                                    ;;*************************************************************
030610  000004                      TST60:  SCOPE
030612  000237                              SPL     7                  ;SET CPU AT LEVEL 7
030614  052737  040000  177776              BIS     #BIT14,@#PSW       ;GO TO SUPER MODE
030622  000230                              SPL     0                  ;TRY AND CLEAR PRIORITIES
030624  042737  040000  177776              BIC     #BIT14,@#PSW       ;GO BACK TO KERNEL
030632  013767  177776  150346              MOV     @#PSW,$ERPSW       ;SAVE PSW
030640  042767  177437  150340              BIC     #^C<PR7>,$ERPSW    ;MASK OFF PRIORITES
030646  022767  000340  150332              CMP     #PR7,$ERPSW        ;DID SPL CLR PRIORITIES
030654  001404                              BEQ     TST61              ;;BRANCH IF NO
030656  012767  000340  150276              MOV     #PR7,$TMP0         ;SAVE EXPECTED VALUE
030664  104367                              ERROR   367                ;SPL WORKED IN SUPER MODE
                                    ;;*************************************************************
                                    ;*TEST 61       PSW CLOCKING TEST
                                    ;*
```

```
                                        ;*      THIS TEST ENSURES THAT ALL THE BITS IN THE PSW GET LOADED WHEN THE
                                        ;*      FOLLOWING SIGNALS ARE TRUE:
                                        ;*      1) LOAD PS*KERNEL MODE, AND 2) LOAD PS*KERNEL DATI.
                                        ;*
                                        ;*      IT ALSO ENSURES THAT THE PRESET LOGIC ON BITS 11, 12, 13,
                                        ;*      14, AND 15 FUNCTIONS PROPERLY.
                                        ;*
                                        ;*      FOLLOWING IS A TABLE TO DESCRIBE THE PSW FOR EACH SECTION
                                        ;*
                                        ;*      SECTION PSW AT START    PSW ON STK(OR VECTOR)   EXPEC PSW
                                        ;*         1       000XXX              174XXX            174XXX (175XXX IF KB11-E/EM)
                                        ;*         2       174XXX              000XXX            174XXX
                                        ;*         3       040XXX              134XXX            174XXX
                                        ;*         4       144XXX              000XXX            030XXX
                                        ;*         5       030XXX              000XXX            000XXX
                                        ;;*****************************************************************
 030666 000004                          TST61:  SCOPE
 030670 013767 177776 150266                    MOV     @#PSW,$TMP1         ;SAVE PSW
 030676 012767 174000 150256                    MOV     #174000,$TMP0       ;SAVE EXPECTED VALUE OF PSW
 030704 005767 150362                           TST     KB11E               ;KB11-E OR KB11-EM PROCESSOR?
 030710 001403                                  BEQ     22$                 ;BR IF NOT
 030712 052767 000400 150242                    BIS     #400,$TMP0          ;KB11-E HAS PS08 WHICH SHOULD SET ON NEXT TEST
 030720                                 22$:
 030720 012767 031432 150342                    MOV     #TST62,NEXTTST      ;SAVE ADDRESS OF NEXT TEST
 030726 005067 150254                           CLR     $ERPSW              ;ENSURE $ERPSW CLEAR
                                        ;;*****************************************************************
                                        ;RTI IN KERNEL MODE WITH NEW PSW<15:11>174 (175 IN KB11-E) AND OLD PSW<15:11>000
 030732 012767 030740 150150                    MOV     #11$,$LPERR         ;SETUP ERROR LOOP
 030740 012706 001072                   11$:    MOV     #1072,SP            ;SETUP THE SP
 030744 012716 030760                           MOV     #1$,(SP)            ;PUT ADDRESS OF 1$ ON THE STACK
 030750 012766 177757 000002                    MOV     #177757,2(SP)       ;SET ALL BITS IN NEW PSW EXCEPT T
 030756 000002                                  RTI                         ;EXECUTE INSTRUCTION
 030760 005767 150306                   1$:     TST     KB11E               ;IS THIS A KB11-E OE KB11-EM PROCESSOR?
 030764 001405                                  BEQ     20$                 ;BR IF NOT
 030766 122737 000371 177777                    CMPB    #371,@#PSW+1        ;NEW PSW LOAD OK?
 030774 001413                                  BEQ     2$                  ;BR IF YES
 030776 000404                                  BR      21$                 ;OTHERWISE REPORT ERROR
 031000 122737 000370 177777            20$:    CMPB    #370,@#PSW+1        ;DID NEW PSW LOAD OK?
 031006 001406                                  BEQ     2$                  ;BRANCH IF YES
 031010 113767 177777 150171            21$:    MOVB    @#PSW+1,$ERPSW+1    ;SAVE ERROR PSW
 031016 105037 177777                           CLRB    @#PSW+1             ;GO BACK TO KERNEL
 031022 104370                                  ERROR   370                 ;A HIGH BYTE BIT DID NOT SET IN PSW
                                        ;;*****************************************************************
                                        ;RTI IN USER MODE WITH NEW PSW<15:11>000 AND OLD PSW<15:11>174
 031024 012767 174000 150130            2$:     MOV     #174000,$TMP0       ;SAVE EXPECTED VALUE OF PSW
 031032 012767 031040 150050                    MOV     #12$,$LPERR         ;SETUP ERROR LOOP
 031040 152737 000370 177777            12$:    BISB    #370,@#PSW+1        ;SETUP THE PSW
 031046 012706 001072                           MOV     #1072,USP           ;SETUP USER SP
 031052 012716 031064                           MOV     #3$,(USP)           ;PUT ADDRESS OF 3$ ON STACK
 031056 005066 000002                           CLR     2(USP)              ;PUT NEW PSW ON STACK
 031062 000002                                  RTI                         ;EXECUTE INSTRUCTION
 031064 122737 000370 177777            3$:     CMPB    #370,@#PSW+1        ;DID PSW STAY THE SAME?
 031072 001406                                  BEQ     4$                  ;BRANCH IF YES
 031074 113767 177777 150105                    MOVB    @#PSW+1,$ERPSW+1    ;SAVE PSW
 031102 105037 177777                           CLRB    @#PSW+1             ;GO BACK TO KERNEL
 031106 104371                                  ERROR   371                 ;A HIGH BYTE BIT CLEARED IN PSW
```

```
                                        ;;*******************************************************************
                                        ;RTI IN SUPERVISOR MODE WITH NEW PSW<15:11>134 AND OLD PSW<15:11>040
                                        ;    CHECKS PRESETS ON BITS 11, 12, 13, AND 15
031110  012767  031116  147772  4$:     MOV     #13$,$LPERR       ;SETUP ERROR LOOP
031116  012737  040340  177776  13$:    MOV     #40340,@#PSW      ;GO TO SUPER AND CLEAR PREVIOUS MODE AND REG BITS
031124  012706  001072                  MOV     #1072,SSP         ;SET THE SSP
031130  012716  031144                  MOV     #5$,(SSP)         ;PUT ADDRESS OF 5$ ON STACK
031134  012766  134000  000002          MOV     #134000,2(SSP)    ;PUT NEW PSW ON STACK
031142  000002                          RTI                       ;EXECUTE INSTRUCTION
031144  122737  000370  177777  5$:     CMPB    #370,@#PSW+1      ;DID ALL BITS SET?
031152  001406                          BEQ     10$               ;BRANCH IF YES
031154  113767  177777  150025          MOVB    @#PSW+1,$ERPSW+1  ;SAVE PSW HIGH BYTE
031162  105037  177777                  CLRB    @#PSW+1           ;GO BACK TO KERNEL
031166  104372                          ERROR   372               ;BITS <15,13:11> DID NOT PRESET
                                        ;;*******************************************************************
                                        ;IOT IN USER MODE WITH NEW PSW<15:11>000 AND OLD PSW<15:11>144
031170  012767  031176  147712  10$:    MOV     #14$,$LPERR       ;SETUP ERROR LOOP
031176  112737  000310  177777  14$:    MOVB    #310,@#PSW+1      ;SETUP PSW
031204  012737  031222  000020          MOV     #6$,@#IOTVEC      ;PUT ADDRESS OF 6$ IN IOT VECTOR
031212  012737  000340  000022          MOV     #PR7,@#IOTVEC+2   ;PUT NEW PSW IN IOT VECTOR+2
031220  000004                          IOT                       ;EXECUTE INSTRUCTION
031222  122737  000060  177777  6$:     CMPB    #60,@#PSW+1       ;DID NEW PSW COME UP OK?
031230  001411                          BEQ     7$                ;BRANCH IF YES
031232  113767  177777  147747          MOVB    @#PSW+1,$ERPSW+1  ;SAVE PSW
031240  012767  030000  147714          MOV     #30000,$TMP0      ;SAVE EXPECTED VALUE
031246  105037  177777                  CLRB    @#PSW+1           ;MAKE SURE KERNEL MODE
031252  104373                          ERROR   373               ;PSW HIGH BYTE WRONG
                                        ;;*******************************************************************
                                        ;IOT IN KERNEL MODE WITH NEW PSW<15:11>000 AND OLD PSW<15:11>030
031254  012767  031262  147626  7$:     MOV     #15$,$LPERR       ;SETUP ERROR LOOP
031262  112737  000060  177777  15$:    MOVB    #60,@#PSW+1       ;SETUP PSW
031270  012737  031306  000020          MOV     #8$,@#IOTVEC      ;SETUP IOT VECTOR
031276  012737  030340  000022          MOV     #30340,@#IOTVEC+2 ;SETUP NEW PSW
031304  000004                          IOT                       ;EXECUTE INSTRUCTION
031306  105737  177777          8$:     TSTB    @#PSW+1           ;IS PSW HIGH BYTE CLEAR?
031312  001430                          BEQ     9$                ;BRANCH IF YES
031314  113767  177777  147665          MOVB    @#PSW+1,$ERPSW+1  ;SAVE PSW
031322  005067  147634                  CLR     $TMP0             ;SAVE EXPECTED VALUE
031326  104374                          ERROR   374               ;PREVIOUS MODE BITS DID NOT CLEAR
                                        ;;*******************************************************************
                                        ;RESET IN SUPER MODE
031330  012737  074357  177776          MOV     #74357,@#PSW      ;SETUP PSW
031336  000005                          RESET                     ;EXECUTE INSTRUCTION
031340  013767  177776  147640          MOV     @#PSW,$ERPSW      ;SAVE PSW
031346  026727  147634  074340          CMP     $ERPSW,#74340     ;WAS PSW OK?
031354  001412                          BEQ     16$               ;BRANCH IF YES
031356  012767  074340  147576          MOV     #74340,$TMP0      ;SAVE EXPECTED VALUE
031364  005037  177776                  CLR     @#PSW
031370  104377                          ERROR   377               ;RESET AFFECTED BITS IN PSW
031372  000461                          461
031374  012737  033270  000020  9$:     MOV     #$SCOPE,@#IOTVEC          ;RESTORE IOTVEC
031402  032767  000020  147554  16$:    BIT     #BIT4,$TMP1       ;WAS T BIT ON?
031410  001410                          BEQ     TST62             ;;BRANCH IF NO
031412  012706  001074                  MOV     #1074,SP          ;SETUP THE STACK
031416  016716  147646                  MOV     NEXTTST,(SP)      ;PUT ADDRESS OF NEXT TEST ON STACK
031422  012766  000360  000002          MOV     #360,2(SP)        ;SET T BIT ON STACK
```

```
031430  000002                                  RTI                             ;TURN T BIT ON
                                        ;;*********************************************************************
                                        ;;*TEST 62          ILLEGAL HALT
                                        ;;*         THIS TEST ENSURES THAT A HALT IN SUPER OR USER MODE WILL TRAP TO
                                        ;;*         LOCATION 4.
                                        ;;*
                                        ;;*         IF BEN6 FAILS EXECUTION WOULD GO TO FET.04 WHICH WOULD
                                        ;;*         CAUSE THE HALT TO LOOK LIKE A NOP.
                                        ;;*          IF TMCE SET CONF GOES LOW THE PROCESSOR WILL HALT AT LOCALOCATION 1$.
                                        ;;*
                                        ;;*         THE CPU ERROR REGISTER BIT 7 IS ALSO TESTED HERE.
                                        ;;*********************************************************************
031432  000004                          TST62:  SCOPE
031434  012767  031546  147626                  MOV     #TST63,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
031442  012706  001100                          MOV     #STACK,SP       ;SETUP THE SP
031446  012737  031474  000004                  MOV     #1$,@#ERRVEC    ;SETUP ERRVEC
031454  052737  040000  177776                  BIS     #BIT14,@#PSW    ;GO TO SUPER MODE
031462  000000                                  HALT                    ;EXECUTE INSTRUCTION UNDER TEST
                                        ;FAILURE, NO TRAP
031464  105037  177777                          CLRB    @#PSW+1         ;GO BACK TO KERNEL
031470  104377                                  ERROR   377             ;ILLEGAL HALT DID NOT
031472  000417                                  417                     ;TRAP TO 4
031474  032737  000200  177766          1$:     BIT     #BIT7,@#CPUERR  ;IS ERROR REG OK?
031502  001010                                  BNE     2$              ;BRANCH IF YES
031504  013767  177766  147442                  MOV     @#CPUERR,$REG0  ;SAVE FOR TYPOUT
031512  012767  000200  147442                  MOV     #200,$TMP0      ;SAVE EXPECTED VALUE
031520  104377                                  ERROR   377             ;BIT 7 DID NOT SET
031522  000420                                  420
031524  005037  177766                  2$:     CLR     @#CPUERR        ;CLEAR BIT 7
031530  005737  177766                          TST     @#CPUERR        ;DID BIT 7 CLEAR?
031534  001401                                  BEQ     3$              ;BRANCH IF YES
031536  104256                                  ERROR   256             ;BIT 7 DID NOT CLEAR
031540  012737  032654  000004          3$:     MOV     #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
                                                                        ;CONTINUE
                                        ;;*********************************************************************
                                        ;;*TEST 63          WAIT
                                        ;;*
                                        ;;*         THIS TEST ENSURES THAT THE WAIT INSTRUCTION WORKS PROPERLY.
                                        ;;*         IT FIRST EXECUTES WITH A LEVEL 4 INTERRUPT.
                                        ;;*         THEN THE T BIT IS ENABLED TO ENSURE THAT THE INTERRUPT
                                        ;;*         OCCURS AND NOT THE T BIT TRAP.
                                        ;;*********************************************************************
031546  000004                          TST63:  SCOPE
031550  012767  032042  147512                  MOV     #TST64,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
031556  012767  003706  147320                  MOV     #^D1990,$ICNT   ;ADJUST ITTERATION COUNT
031564  012767  031572  147314                  MOV     #10$,$LPADR     ;SETUP LOOP ADR
031572  012737  031642  000064          10$:    MOV     #2$,@#TPVEC     ;SETUP TELEPRINTER VECTOR
031600  012767  031606  147302                  MOV     #1$,$LPERR      ;SETUP ERROR LOOP
031606  012706  001100                  1$:     MOV     #STACK,SP       ;INITIALIZE THE SP
031612  000233                                  SPL     3
031614  012777  000100  147320                  MOV     #BIT6,@$TPS     ;SET PRINTER INTERRUPT FLAG
031622  012777  000015  147314                  MOV     #15,@$TPB       ;SEND CHARACTER
031630  000001                                  WAIT                    ;EXECUTE INSTRUCTION UNDER TEST
                                        ;FAILURE
031632  005077  147304                          CLR     @$TPS           ;CLEAR INTERR ENABLE BIT
031636  104377                                  ERROR   377             ;WAIT INSTRUCTION DID NOT
```

```
031640  000440                                 440                      ;WAIT FOR INTERRUPT
                                ;TEST OK, NO TRY WITH T BIT
031642  012767  031664  147240  2$:     MOV     #3$,$LPERR              ;SETUP ERROR LOOP
031650  012737  031720  000014          MOV     #4$,@#TBITVEC           ;SETUP T BIT VECTOR
031656  012737  031742  000064          MOV     #5$,@#TPVEC             ;SETUP TELEPRINTER VECTOR
031664  012706  001100          3$:     MOV     #STACK,SP               ;INITIALIZE THE SP
031670  012746  000020                  MOV     #20,-(SP)               ;PUT SP ON STACK TO TURN ON T BIT
031674  012746  031716                  MOV     #6$,-(SP)               ;PUT RETURN ADDRESS ON STACK
031700  052777  000100  147234          BIS     #BIT6,@$TPS             ;SET PRINTER INTERRUPT BIT
031706  012777  000015  147230          MOV     #15,@$TPB               ;SEND CHARACTER
031714  000006                          RTT                             ;TURN T BIT ON
031716  000001                  6$:     WAIT                            ;EXECUTE INSTRUCTION UNDER TEST
                                ;FAILURE, T BIT CAME IN
031720  005077  147216          4$:     CLR     @$TPS                   ;CLEAR PRINTER STATUS
031724  012706  001100                  MOV     #STACK,SP               ;RESTORE THE SP
031730  012737  032640  000014          MOV     #$RTRN,@#TBITVEC                ;RESTORE T BIT VECTOR
031736  104377                          ERROR   377                     ;T BIT CAUSED WAIT TO
031740  000441                          441                             ;QUIT
                                ;TEST OK, NOW TRY PIRQ
031742  012737  032640  000014  5$:     MOV     #$RTRN,@#TBITVEC                ;RESTORE T BIT VECTOR
031750  012767  031772  147132          MOV     #7$,$LPERR              ;SETUP ERROR LOOP
031756  012737  032016  000064          MOV     #8$,@#TPVEC             ;SETUP TP VECTOR
031764  012737  032032  000240          MOV     #9$,@#PIRQVEC           ;SETUP PIRQ VECTOR
031772  012706  001100          7$:     MOV     #STACK,SP               ;SETUP THE SP
031776  012737  100000  177772          MOV     #BIT15,@#PIRQ           ;SET PIR LEVEL 7
032004  012777  000015  147132          MOV     #15,@$TPB               ;SEND CHARACTER
032012  000233                          SPL     3                       ;LOWER CPU
032014  000001                          WAIT                            ;EXECUTE INSTRUCTION UNDER TEST
                                ;FAILURE, PIRQ DID NOT INTERRUPT
032016  005077  147120          8$:     CLR     @$TPS                   ;CLEAR TP STATUS
032022  005037  177772                  CLR     @#PIRQ                  ;CLEAR PIR LEVEL 7
032026  104377                          ERROR   377                     ;PIRQ DID NOT CAUSE
032030  000442                          442                             ;INTERRUPT
                                ;TEST OK
032032  005077  147104          9$:     CLR     @$TPS                   ;CLEAR INTERRUPT FLAG
032036  005037  177772                  CLR     @#PIRQ                  ;CLEAR PIR LEVEL 7
                                                                        ;CONTINUE

                                ;;*****************************************************************
                                ;*TEST 64       CHECK MFPT INSTRUCTION (KB-11E/EM ONLY)
                                ;*      THE MFPT INSTRUCTION IS NOT AVAILABLE ON THE 11/70 BUT
                                ;*      IF THIS IS AN KB-11E/EM THIS TEST IS RUN.  MFPT RETURNS
                                ;*      DATA TO R0 IN THE FOLLOWING FORMAT:
                                ;*      BIT 0 - 1  INDICATES 11/44 CPU
                                ;*      BIT 1 - 1 INDICATES KB-11E/EM CPU (SHOULD ALWAYS COME UP IN THIS TEST)
                                ;*      BIT 8 - 1 INDICATES CISP PRESENT
                                ;*      BIT 9 - 1 INDICATES FP PRESENT
                                ;*
                                ;*      THIS TEST MASKS OFF BITS 8 AND 9 AND ONLY CHECKS THAT BITS 0 AND 1
                                ;*      COME BACK CORRECTLY AND THAT THE OTHER BITS ARE CLEAR.
                                ;;*****************************************************************
032042  000004                  TST64:  SCOPE
032044  012767  032114  147216          MOV     #MMSET,NEXTTST          ;SAVE ADDRESS OF NEXT TEST
032052  005737  001272                  TST     @#KB11E                 ;IS THIS A KB11-E OR KB11-EM?
032056  001002                          BNE     DOMFPT                  ;BR IF IT IS
032060  000167  000030                  JMP     DONE7                   ;SKIP THIS TEST IF NOT. MFPT NOT THERE
```

```
032064                          DOMFPT:
032064  012703  000002                   MOV     #2,R3              ;R3 IS DATA PATTERN. BIT 1 WILL ALWAYS BE SET
032070  000007                           MFPT                       ;EXECUTE INSTRUCTION
032072  042700  001400                   BIC     #1400,R0           ;MASK OFF CISP AND FP BITS
032076  020003                           CMP     R0,R3              ;MATCH?
032100  001405                           BEQ     DONE7              ;DONE IF SO
032102  010337  001164                   MOV     R3,@#$TMP1         ;SET UP EXPECTED (GOOD) DATA
032106  010037  001162                   MOV     R0,@#$TMP0         ;SET UP RECIEVED (BAD) DATA
032112  104173                           ERROR   173                ;ERROR PRINTOUT
032114                          DONE7:
032114                          MMSET:
                                ;;*****************************************************************
                                ;*TEST 65        OPERATOR INTERVENTION TEST
                                ;*
                                ;*       THIS TEST ENSURES THAT THE RESET AND WAIT FLOWS PUT R0
                                ;*       AND THE PC IN THE LIGHTS. THE TEST IS ONLY EXECUTED ON
                                ;*       PASS 1 AND CAN BE DISABLED ALTOGETHER WITH SWITCH 0.
                                ;*
                                ;*       THE RESET LOOP IS STOPED BY CHANGING THE POSITION OF
                                ;*       SWITCH 7.
                                ;*
                                ;*       THE WAIT IS EXITED BY TYPING A CHARACTER
                                ;*       ON THE TERMINAL.
                                ;;*****************************************************************
032114  000004                  TST65:   SCOPE
032116  012767  032354  147144           MOV     #$EOP,NEXTTST      ;SAVE ADDRESS OF $EOP
032124  005767  146750                   TST     $PASS              ;IS THIS FIRST PASS?
032130  001006                           BNE     1$                 ;BRANCH IF NO
032132  032737  000001  177570           BIT     #SW0,@#SWR         ;IS SWITCH 0 ON?
032140  001403                           BEQ     2$                 ;BRANCH IF NO
032142  104400  072557                   TYPE    ,EM717             ;TYPE MESSAGE(SKIPPING TEST)
032146  000502                  1$:      BR      $EOP               ;EXIT
032150  005737  000042          2$:      TST     @#42               ;IS MONITOR ACT11?
032154  001077                           BNE     $EOP               ;BRANCH IF YES
032156  012700  166667                   MOV     #166667,R0         ;SETUP R0
032162  104400  072602                   TYPE    ,EM720             ;TYPE MESSAGE
032166  032737  000200  177570           BIT     #SW7,@#SWR         ;IS SWITCH 7 ON?
032174  001416                           BEQ     3$                 ;BRANCH IF NO
032176  012767  032220  146756           MOV     #4$+2,$TMP0        ;SAVE PC
032204  016746  146752                   MOV     $TMP0,-(SP)        ;;SAVE $TMP0 FOR TYPEOUT
                                                                    ;;TYPE ADDRESS
032210  104402                           TYPOC                      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
032212  104400  072741                   TYPE    ,EM721             ;TYPE MESSAGE
032216  000005                  4$:      RESET                      ;EXECUTE INSTRUCTION
032220  032737  000200  177570           BIT     #SW7,@#SWR         ;HAS SWITCH 7 CHANGED?
032226  001416                           BEQ     5$                 ;BRANCH IF YES
032230  000772                           BR      4$                 ;LOOP
032232  012767  032254  146722  3$:      MOV     #6$+2,$TMP0        ;SAVE PC
032240  016746  146716                   MOV     $TMP0,-(SP)        ;;SAVE $TMP0 FOR TYPEOUT
                                                                    ;;TYPE ADDRESS
032244  104402                           TYPOC                      ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
032246  104400  072741                   TYPE    ,EM721             ;TYPE MESSAGE
032252  000005                  6$:      RESET                      ;EXECUTE INSTRUCTION
032254  032737  000200  177570           BIT     #SW7,@#SWR         ;HAS SWITCH 7 CHANGED?
032262  001773                           BEQ     6$                 ;BRANCH IF NO
                                ;
```

```
                                         ;WAIT TEST
032264  104400  072602                   5$:     TYPE    ,EM720          ;TYPE MESSAGE
032270  012767  032334  146664                   MOV     #7$,$TMPO       ;SAVE PC
032276  016746  146660                           MOV     $TMPO,-(SP)     ;;SAVE $TMPO FOR TYPEOUT
                                                                         ;;TYPE ADDRESS
032302  104402     .                             TYPOC                   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
032304  104400  072777                           TYPE    ,EM722          ;TYPE MESSAGE
032310  005077  146624                           CLR     @$TKB           ;CLEAR KEYBOARD BUFFER
032314  012737  032334  000060                   MOV     #7$,@#TKVEC     ;SETUP KEYBOARD VECTOR
032322  052777  000100  146606                   BIS     #BIT6,@$TKS     ;SET INTERRUPT ENABLE BIT
032330  000233                                   SPL     3               ;LOWER PRIORITY FOR INTERRUPT
032332  000001                                   WAIT                    ;EXECUTE INSTRUCTION
032334  012737  034644  000060   7$:     MOV     #QUIT,@#TKVEC   ;RESTORE TKVECTOR
032342  005077  146572                           CLR     @$TKB           ;ENSURE BUFFER CLEAR
032346  152777  000100  146562                   BISB    #BIT6,@$TKS     ;SET INTERRUPT ENABLE FLAG
                                                                         ;CONTINUE

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

                                         .SBTTL  END OF PASS ROUTINE

                                         ;*INCREMENT THE PASS NUMBER ($PASS)
                                         ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
                                         ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
                                         ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
                                         ;*IF SW12=1 INHIBIT TRACE TRAP
                                         ;*IF THERES A MONITOR GO TO IT
                                         ;*IF THERE ISN'T JUMP TO LOOP

032354                                   $EOP:
032354  000004                                   SCOPE
032356  005067  146520                           CLR     $TSTNM          ;;ZERO THE TEST NUMBER
032362  005067  146604                           CLR     $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
032366  005267  146506                           INC     $PASS           ;;INCREMENT THE PASS NUMBER
032372  042767  100000  146500                   BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
032400  005327                                   DEC     (PC)+           ;;LOOP?
032402  000001                           $EOPCT: .WORD   1
032404  003076                                   BGT     $DOAGN          ;;YES
032406  012737                                   MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
032410  000001                           $ENDCT: .WORD   1
032412  032402                                   $EOPCT
032414  104400  032422                           TYPE    ,65$            ;;TYPE ASCIZ STRING
032420  000407                                   BR      64$             ;;GET OVER THE ASCIZ
                                         ;;65$:  .ASCIZ  <12><15>/END PASS #/
032440                                   64$:
032440  016746  146434                           MOV     $PASS,-(SP)     ;;SAVE $PASS FOR TYPEOUT
                                                                         ;;TYPE PASS NUMBER
032444  104410                                   TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
032446  104400  032454                           TYPE    ,67$            ;;TYPE ASCIZ STRING
032452  000421                                   BR      66$             ;;GET OVER THE ASCIZ
                                         ;;67$:  .ASCIZ  /  TOTAL ERRORS SINCE LAST REPORT /
032516                                   66$:
032516  016746  146370                           MOV     $ERTTL,-(SP)    ;;SAVE $ERTTL FOR TYPEOUT
                                                                         ;;TOTAL NUMBER OF ERRORS
032522  104410                                   TYPDS                   ;;GO TYPE--DECIMAL ASCII WITH SIGN
032524  104400  001203                           TYPE    ,$CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
032530  005067  146356                           CLR     $ERTTL          ;;CLEAR ERROR TOTAL
```

```
032534  013700  000042          $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
032540  001420                          BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
032542  005046                          CLR     -(SP)           ;;INSURE THE "T" BIT IS CLEAR
032544  012746  032552                  MOV     #$CLR.T,-(SP)   ;;SETUP FOR AN RTI OR RTT
032550  000433                          BR      $RTRN           ;;GO DO AN RTI OR RTT TO LOAD THE PSW
                                                                ;;WITH A CLEARED "T" BIT
032552                          $CLR.T:
032552  012703  000001                  MOV     #1,R3           ;MAKE R3=1
032556  004767  002120                  JSR     PC,CHAINQ       ;GO RESTORE MONITOR
032562  013700  000042                  MOV     @#42,R0         ;;INSURE R0 CONTAINS THE MONITORS
032566  001405                          BEQ     $DOAGN          ;;RETURN ADDRESS
032570  000005                          RESET                   ;;CLEAR THE WORLD
032572  004710                  $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
032574  000240                          NOP                     ;;SAVE ROOM
032576  000240                          NOP                     ;;FOR
032600  000240                          NOP                     ;;ACT11
032602                          $DOAGN:
032602  013746  177776                  MOV     @#PS,-(SP)      ;;PUT THE PS ON THE STACK AND
032606  042716  000020                  BIC     #20,(SP)        ;;CLEAR THE "T" BIT
032612  032737  010000  177570          BIT     #BIT12,@#SWR    ;;RUN WITH TRACE TRAP?
032620  001005                          BNE     1$              ;;BR IF NO
032622  005167  000020                  COM     $TBIT           ;;IS IT TIME FOR TRACE TRAP
032626  100402                          BMI     1$              ;;BR IF NO
032630  052716  000020                  BIS     #20,(SP)        ;;SET TRACE TRAP
032634  012746  032642          1$:     MOV     #$LOOP,-(SP)    ;;JUMP TO START OF TEST
032640  000002                  $RTRN:  RTI                     ;;RETURN--THIS IS CHANGED TO
                                                                ;;AN "RTT" IF "RTT" IS A LEGAL
                                                                ;;INSTRUCTION
032642                          $LOOP:
032642  000137  005340                  JMP     @#LOOP          ;;RETURN
032646  000000                  $TBIT:  .WORD   0               ;;"T" BIT STATE INDICATOR
032650     377     377     000  $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
        032654                          .EVEN
                                ;;**************************************************************
                                .SBTTL  SPURIOUS ERROR HANDLER
                                ;*      THIS ROUTINE IS ENTERED BY AN UNEXPECTED TRAP TO 4 OR 114.
                                ;*       IF SWITCH 13 IS OFF, AN ERROR MESSAGE, THE ERROR REGISTER,
                                ;*      THE ERROR PC, AND THE TEST NUMBER ARE TYPED OUT.
                                ;*       IF SWITCH 13 IS ON, ONLY THE ERROR MESSAGE WILL BE TYPED.
                                ;;**************************************************************
032654  011667  146236          CPUSPUR:MOV     (SP),$ERRPC             ;SAVE THE ERROR PC
032660  012706  001100                  MOV     #STACK,SP       ;RESTORE THE SP
032664  104400  032672                  TYPE    ,65$            ;;TYPE ASCIZ STRING
032670  000414                          BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /UNEXPECTED TRAP TO 4/<15><12>
032722                          64$:
032722  032737  020000  177570          BIT     #BIT13,@#SWR    ;IS INHIBIT ERROR TYPEOUT ON?
032730  001045                          BNE     1$              ;BRANCH IF YES
032732  104400  032740                  TYPE    ,67$            ;;TYPE ASCIZ STRING
032736  000417                          BR      66$             ;;GET OVER THE ASCIZ
                                ;;67$:  .ASCIZ  /ERRORPC TSTNUM  CPUERR REG/<15><12>
032776                          66$:
032776  013767  177766  146156          MOV     @#CPUERR,$TMP0  ;GET CPU ERROR REG
033004  116767  146072  146176          MOVB    $TSTNM,$$TSTNM  ;GET TEST NUMBER
033012  016746  146100                  MOV     $ERRPC,-(SP)    ;;SAVE $ERRPC FOR TYPEOUT
                                                                ;;TYPE ERROR PC
```

```
033016  104402                            TYPOC                          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
033020  104400  034640                    TYPE     ,TWOSP
033024  016746  146160                    MOV      $$TSTNM,-(SP)         ;;SAVE $$TSTNM FOR TYPEOUT
                                                                         ;;TYPE TEST NUMBER
033030  104402                            TYPOC                          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
033032  104400  034640                    TYPE     ,TWOSP
033036  016746  146120                    MOV      $TMP0,-(SP)           ;;SAVE $TMP0 FOR TYPEOUT
                                                                         ;;TYPE ERROR REGISTER
033042  104402                            TYPOC                          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
033044  005037  177766            1$:     CLR      @#CPUERR              ;CLEAR CPU ERROR REG
033050  016767  146214  146116            MOV      NEXTTST,$ESCAPE       ;SETUP ESCAPE ADDRESS
033056  104264                            ERROR    264                   ;MAKE THE ERROR CALL TO SYSMAC
033060  011667  146032            CACHSPU:MOV      (SP),$ERRPC           ;SAVE ERROR PC
033064  012706  001100                    MOV      #STACK,SP             ;RESTORE STACK
033070  104400  033076                    TYPE     ,65$                  ;;TYPE ASCIZ STRING
033074  000415                            BR       64$                   ;;GET OVER THE ASCIZ
                                  ;;65$:  .ASCIZ   /UNEXPECTED TRAP TO 114/<15><12>
033130                            64$:
033130  032737  020000  177570            BIT      #BIT13,@#SWR          ;IS SWITCH 13 ON?
033136  001045                            BNE      1$                    ;BRANCH IF YES
033140  104400  033146                    TYPE     ,67$                  ;;TYPE ASCIZ STRING
033144  000417                            BR       66$                   ;;GET OVER THE ASCIZ
                                  ;;67$:  .ASCIZ   /ERRORPC TSTNUM  MEMERR REG/<15><12>
033204                            66$:
033204  013767  177744  145750            MOV      @#MEMERR,$TMP0        ;SAVE MEMORY ERROR REG
033212  116767  145664  145770            MOVB     $TSTNM,$$TSTNM        ;SAVE TEST NUMBER
033220  016746  145672                    MOV      $ERRPC,-(SP)          ;;SAVE $ERRPC FOR TYPEOUT
                                                                         ;;TYPE ERROR PC
033224  104402                            TYPOC                          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
033226  104400  034640                    TYPE     ,TWOSP
033232  016746  145752                    MOV      $$TSTNM,-(SP)         ;;SAVE $$TSTNM FOR TYPEOUT
                                                                         ;;TYPE TEST NUMBER
033236  104402                            TYPOC                          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
033240  104400  034640                    TYPE     ,TWOSP
033244  016746  145712                    MOV      $TMP0,-(SP)           ;;SAVE $TMP0 FOR TYPEOUT
                                                                         ;;TYPE MEM ERROR REG
033250  104402                            TYPOC                          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
033252  013737  177744  177744    1$:     MOV      @#MEMERR,@#MEMERR  ;CLEAR MEMERR REG.
033260  016767  146004  145706            MOV      NEXTTST,$ESCAPE       ;SETUP ESCAPE ADDRESS
033266  104264                            ERROR    264
                                  ;;*************************************************************


                                  .SBTTL  SCOPE HANDLER ROUTINE

                                  ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                  ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
                                  ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
                                  ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                  ;*SW14=1          LOOP ON TEST
                                  ;*SW11=1          INHIBIT ITERATIONS
                                  ;*SW09=1          LOOP ON ERROR
                                  ;*SW08=1          LOOP ON TEST IN SWR<7:0>
                                  ;*CALL
                                  ;*       SCOPE                 ;;SCOPE=IOT

033270                            $SCOPE:
```

```
033270  006137  177570                          ROL     @#SWR               ;;LOOP ON PRESENT TEST?
033274  100511                                  BMI     $OVER               ;;YES IF SW14=1
                                        ;#####START OF CODE FOR THE XOR TESTER#####
033276  000416                          $XTSTR: BR      6$                  ;;IF RUNNING ON THE ''XOR'' TESTER CHANGE
                                                                            ;;THIS INSTRUCTION TO A ''NOP'' (NOP=240)
033300  013746  000004                          MOV     @#ERRVEC,-(SP)      ;;SAVE THE CONTENTS OF THE ERROR VECTOR
033304  012737  033324  000004                  MOV     #5$,@#ERRVEC        ;;SET FOR TIMEOUT
033312  005737  177060                          TST     @#177060            ;;TIME OUT ON XOR?
033316  012637  000004                          MOV     (SP)+,@#ERRVEC      ;;RESTORE THE ERROR VECTOR
033322  000463                                  BR      $SVLAD              ;;GO TO THE NEXT TEST
033324  022626                          5$:     CMP     (SP)+,(SP)+         ;;CLEAR THE STACK AFTER A TIME OUT
033326  012637  000004                          MOV     (SP)+,@#ERRVEC      ;;RESTORE THE ERROR VECTOR
033332  000423                                  BR      7$                  ;;LOOP ON THE PRESENT TEST
033334                                  6$:;#####END OF CODE FOR THE XOR TESTER#####
033334  032737  000400  177570                  BIT     #BIT08,@#SWR        ;;LOOP ON SPEC. TEST?
033342  001404                                  BEQ     2$                  ;;BR IF NO
033344  123767  177570  145530                  CMPB    @#SWR,$TSTNM        ;;ON THE RIGHT TEST?     SWR<7:0>
033352  001462                                  BEQ     $OVER               ;;BR IF YES
033354  105767  145523                  2$:     TSTB    $ERFLG              ;;HAS AN ERROR OCCURRED?
033360  001421                                  BEQ     3$                  ;;BR IF NO
033362  126767  145527  145513                  CMPB    $ERMAX,$ERFLG       ;;MAX. ERRORS FOR THIS TEST OCCURRED?
033370  101015                                  BHI     3$                  ;;BR IF NO
033372  032737  001000  177570                  BIT     #BIT09,@#SWR        ;;LOOP ON ERROR?
033400  001404                                  BEQ     4$                  ;;BR IF NO
033402  016767  145502  145476  7$:     MOV     $LPERR,$LPADR       ;;SET LOOP ADDRESS TO LAST SCOPE
033410  000443                                  BR      $OVER
033412  105067  145465                  4$:     CLRB    $ERFLG              ;;ZERO THE ERROR FLAG
033416  005067  145550                          CLR     $TIMES              ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
033422  000415                                  BR      1$                  ;;ESCAPE TO THE NEXT TEST
033424  032737  004000  177570  3$:     BIT     #BIT11,@#SWR        ;;INHIBIT ITERATIONS?
033432  001011                                  BNE     1$                  ;;BR IF YES
033434  005767  145440                          TST     $PASS               ;;IF FIRST PASS OF PROGRAM
033440  001406                                  BEQ     1$                  ;;        INHIBIT ITERATIONS
033442  005267  145436                          INC     $ICNT               ;;INCREMENT ITERATION COUNT
033446  026767  145520  145430                  CMP     $TIMES,$ICNT        ;;CHECK THE NUMBER OF ITERATIONS MADE
033454  002021                                  BGE     $OVER               ;;BR IF MORE ITERATION REQUIRED
033456  012767  000001  145420  1$:     MOV     #1,$ICNT            ;;REINITIALIZE THE ITERATION COUNTER
033464  016767  000044  145500                  MOV     $MXCNT,$TIMES       ;;SET NUMBER OF ITERATIONS TO DO
033472  105267  145404                  $SVLAD: INCB    $TSTNM              ;;COUNT TEST NUMBERS
033476  011667  145404                          MOV     (SP),$LPADR         ;;SAVE SCOPE LOOP ADDRESS
033502  011667  145402                          MOV     (SP),$LPERR         ;;SAVE ERROR LOOP ADDRESS
033506  005067  145462                          CLR     $ESCAPE             ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
033512  112767  000001  145375                  MOVB    #1,$ERMAX           ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
033520  016737  145356  177570  $OVER:  MOV     $TSTNM,@#DISPLAY    ;;DISPLAY TEST NUMBER
033526  016716  145354                          MOV     $LPADR,(SP)         ;;FUDGE RETURN ADDRESS
033532  000002                                  RTI                         ;;FIXES PS
033534  003720                          $MXCNT: 2000.                       ;;MAX. NUMBER OF ITERATIONS
                                        ;;*************************************************************


                                        .SBTTL  ERROR HANDLER ROUTINE

                                        ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
                                        ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
                                        ;*AND GO TO ETYPDM ON ERROR
                                        ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                        ;*SW15=1          HALT ON ERROR
```

```
                                    ;*              HALT CAN OCCUR BEFORE AND AFTER THE ERROR TYPEOUT
                                    ;*SW13=1        INHIBIT ERROR TYPEOUTS
                                    ;*SW10=1        BELL ON ERROR
                                    ;*SW09=1        LOOP ON ERROR
                                    ;*CALL
                                    ;*      ERROR   N       ;;ERROR=EMT AND N=ERROR ITEM NUMBER

033536                              $ERROR:
033536  116767  145340  145444              MOVB    $TSTNM,$$TSTNM  ;GET TEST NUMBER FOR TYPE OUT
033544  105267  145333              7$:     INCB    $ERFLG          ;;SET THE ERROR FLAG
033550  001775                              BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
033552  016737  145324  177570              MOV     $TSTNM,@#DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
033560  005737  177570                      TST     @#SWR           ;;HALT ON ERROR = 1?
033564  100001                              BPL     8$              ;;BRANCH IF NO
033566  000000                              HALT                    ;;YES--HALT
033570  032737  002000  177570      8$:     BIT     #BIT10,@#SWR    ;;BELL ON ERROR?
033576  001402                              BEQ     1$              ;;NO - SKIP
033600  104400  001176                      TYPE    ,$BELL          ;;RING BELL
033604  005267  145302              1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
033610  011667  145302                      MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
033614  162767  000002  145274              SUB     #2,$ERRPC
033622  117767  145270  145264              MOVB    @$ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
033630  032737  020000  177570              BIT     #BIT13,@#SWR    ;;SKIP TYPEOUT IF SET
033636  001004                              BNE     2$              ;;SKIP TYPEOUTS
033640  004767  000056                      JSR     PC,ETYPDM       ;;GO TO USER ERROR ROUTINE
033644  104400  001203                      TYPE    ,$CRLF
033650  005737  177570              2$:     TST     @#SWR           ;;HALT ON ERROR
033654  100001                              BPL     9$              ;;SKIP IF CONTINUE
033656  000000                              HALT                    ;;HALT ON ERROR!
033660  022767  032572  144154      9$:     CMP     #$ENDAD,42      ;;ACT-11?
033666  001001                              BNE     3$              ;;BRANCH IF NO
033670  000000                              HALT                    ;;YES
033672  032737  001000  177570      3$:     BIT     #BIT09,@#SWR    ;;LOOP ON ERROR SWITCH SET?
033700  001402                              BEQ     4$              ;;BR IF NO
033702  016716  145202                      MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
033706  005767  145262              4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
033712  001402                              BEQ     5$              ;;BR IF NONE
033714  016716  145254                      MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
033720                              5$:
033720  000002                              RTI                     ;;RETURN
                                    ;;*************************************************************************
                                    .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
                                    ;*THIS ROUTINE USES THE ''ITEM CONTROL BYTE'' ($ITEMB) TO DETERMINE WHICH
                                    ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE ''ERROR TABLE'' ($ERRTB),
                                    ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
                                    ;;*************************************************************************
033722  026727  145262  000042      ETYPDM: CMP     $$TSTNM,#42     ;IS THIS TEST 42?
033730  001464                              BEQ     TYPSLE          ;BRANCH IF YES
033732  026727  145252  000070              CMP     $$TSTNM,#70     ;IS THIS TEST 70?
033740  001460                              BEQ     TYPSLE          ;BRANCH IF YES
033742  104400  001203                      TYPE    ,$CRLF          ;;''CARRIAGE RETURN'' & ''LINE FEED''
033746  005000                              CLR     R0              ;;PICKUP THE ITEM INDEX
033750  153700  001114                      BISB    @#$ITEMB,R0
033754  126727  145134  000377              CMPB    $ITEMB,#377     ;IS MESSABE NUMBER OVER 377
033762  001006                              BNE     1$              ;BRANCH IF NO
033764  016600  000002                      MOV     2(SP),R0        ;GET RETURN PC FROM STACK
```

```
033770  011000                          MOV     (R0),R0                 ;GET MESSAGE NUMBER
033772  062766  000002  000002          ADD     #2,2(SP)                ;CORRECT PC
034000  005300                  1$:     DEC     R0                      ;;ADJUST THE INDEX SO THAT IT WILL
034002  010001                          MOV     R0,R1
034004  070127  000006                  MUL     #6,R1                   ;;WORK FOR THE ERROR TABLE
034010  010100                          MOV     R1,R0
034012  062700  001274                  ADD     #$ERRTB,R0              ;;FORM TABLE POINTER
034016  012067  000004                  MOV     (R0)+,2$                ;;PICKUP ''ERROR MESSAGE'' POINTER
034022  001404                          BEQ     3$                      ;;SKIP TYPEOUT IF NO POINTER
034024  104400                          TYPE                            ;;TYPE THE ''ERROR MESSAGE''
034026  000000                  2$:     .WORD   0                       ;;''ERROR MESSAGE'' POINTER GOES HERE
034030  104400  001203                  TYPE    ,$CRLF                  ;;''CARRIAGE RETURN'' & ''LINE FEED''
034034  012067  000004          3$:     MOV     (R0)+,4$                ;;PICKUP ''DATA HEADER'' POINTER
034040  001404                          BEQ     5$                      ;;SKIP TYPEOUT IF 0
034042  104400                          TYPE                            ;;TYPE THE ''DATA HEADER''
034044  000000                  4$:     .WORD   0                       ;;''DATA HEADER'' POINTER GOES HERE
034046  104400  001203                  TYPE    ,$CRLF                  ;;''CARRIAGE RETURN'' & ''LINE FEED''
034052  011000                  5$:     MOV     (R0),R0                 ;;PICKUP ''DATA TABLE'' POINTER
034054  001003                          BNE     7$                      ;;GO TYPE THE DATA
034056  104400  001203          6$:     TYPE    ,$CRLF                  ;;''CARRIAGE RETURN'' & ''LINE FEED''
034062  000207                          RTS     PC                      ;;RETURN
034064                          7$:
034064  013046                          MOV     @(R0)+,-(SP)            ;;SAVE @(R0)+ FOR TYPEOUT
034066  104402                          TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
034070  005710                          TST     (R0)                    ;;IS THERE ANOTHER NUMBER?
034072  001771                          BEQ     6$                      ;;BR IF NO
034074  104400  034640                  TYPE    ,TWOSP                  ;;TYPE TWO(2) SPACES
034100  000771                          BR      7$                      ;;LOOP
                                ;;*****************************************************************
                                .SBTTL  STACK LIMIT TEST TYPE OUT ROUTINE
                                ;*      THIS ROUTINE TYPES THE ADDRESS AND STACK LIMIT REGISTER
                                ;*VALUES THAT FAILED IN TEST 153 OR 201 IN OCTAL AND BINARY.
                                ;;*****************************************************************
034102  104400                  TYPSLE: TYPE                            ;TYPE
034104  055262                          EM263                           ;ERROR MESSAGE
034106  104400                          TYPE                            ;TYPE
034110  055451                          DH263                           ;DATA HEADER
034112  016727  145000                  MOV     $ERRPC,(PC)+            ;GET ERROR PC
034116  000000                  1$:     .WORD                           ;ERROR PC
034120  016746  177772                  MOV     1$,-(SP)                ;;SAVE 1$ FOR TYPEOUT
                                                                        ;;TYPE IT
034124  104402                          TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
034126  104400                          TYPE                            ;TYPE TWO SPACES
034130  034640                          TWOSP
034132  016727  145052                  MOV     $$TSTNM,(PC)+           ;GET TEST NUMBER
034136  000000                  2$:     .WORD   0                       ;TEST NUMBER
034140  016746  177772                  MOV     2$,-(SP)                ;;SAVE 2$ FOR TYPEOUT
                                                                        ;;TYPE IT
034144  104402                          TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
034146  104400                          TYPE                            ;TYPE CR LF
034150  001203                          $CRLF
034152  104400                          TYPE                            ;TYPE DATA HEADER
034154  055476                          DH263A
034156  005067  000020                  CLR     $TYPEE                  ;INITIALIZE $TYPEE
```

```
034162  005001                          STYPE:  CLR     R1                ;ENSURE R1 CLEAR
034164  005067  000134                          CLR     SLDATA            ;
034170  005067  000174                          CLR     SPDATA            ;
034174  012700  120000                          MOV     #120000,R0        ;GET ERROR DATA BUFFER POINTER
034200  122710                          BUFFDP: CMPB    (PC)+,(R0)        ;IS THIS CORRECT TYPE?
034202  000001                          $TYPEE: .WORD   1
034204  001417                                  BEQ     TYPEHE            ;BRANCH IF YES, GO TYPE IT
034206  062700  000004                          ADD     #4,R0             ;STEP R0 TO NEXT ENTRY
034212  020067  144736                  NEXTEN: CMP     R0,$REG0          ;LAST ENTRY?
034216  001401                                  BEQ     2$                ;BRANCH IF YES
034220  000767                                  BR      BUFFDP            ;GO CHECK NEXT ENTRY
034222  000241                          2$:     CLC                       ;ENSURE C CLEAR
034224  062767  000002  177750                  ADD     #2,$TYPEE         ;SELECT NEXT ERROR TYPE
034232  026727  177744  000014                  CMP     $TYPEE,#14        ;IS TYPE ROUTINE DONE?
034240  001350                                  BNE     STYPE             ;BRANCH IF NO
034242  000467                                  BR      $$DONE            ;RETURN
                                        ;OUTPUT THE TYPE HEADER IF FIRST ERROR OF THIS TYPE
034244  005701                          TYPEHE: TST     R1                ;FIRST ERROR OF THIS TYPE?
034246  001014                                  BNE     4$                ;BRANCH IF NO
034250  016701  177726                          MOV     $TYPEE,R1         ;GET TYPE NUMBER
034254  062701  056060                          ADD     #INDEX,R1         ;GET ADDRESS OF TYPE HEADER ADDRESS
034260  011167  000006                          MOV     (R1),3$           ;GET ADDRESS OF HEADER
034264  104400  034622                          TYPE    ,SP16             ;TYPE SPACES
034270  104400                                  TYPE                      ;GO TYPE THE HEADER
034272  000000                          3$:     .WORD                     ;ADDRESS OF HEADER
034274  104400                                  TYPE                      ;TYPE A CRLF
034276  001203                                  $CRLF
                                        ;OUTPUT THE DATA
034300  126067  000001  000017  4$:     CMPB    1(R0),SLDATA+1    ;IS SL DATA SAME AS LAST SL DATA?
034306  001005                                  BNE     8$                ;BRANCH IF NO
034310  104400                                  TYPE                      ;TYPE 38 SPACES
034312  034600                                  SP38
034314  062700  000002                          ADD     #2,R0             ;STEP R0 TO SP DATA
034320  000422                                  BR      SPDATA-2          ;GO TO SP DATA OUTPUT
034322  012027                          8$:     MOV     (R0)+,(PC)+       ;GET SL DATA
034324  000000                          SLDATA: .WORD                     ;
034326  042767  000377  177770                  BIC     #377,SLDATA       ;MASK HIGH BYTE
034334  016746  177764                          MOV     SLDATA,-(SP)      ;;SAVE SLDATA FOR TYPEOUT
                                                                          ;;TYPE IT
034340  104402                                  TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
034342  016746  177756                          MOV     SLDATA,-(SP)      ;PUT DATA ON STACK
034346  004767  000150                          JSR     PC,TYPEBN         ;GO TYPE BINARY
034352  021067  000012                          CMP     (R0),SPDATA       ;IS SP DATA SAME AS LAST SP DATA?
034356  001003                                  BNE     9$                ;BRANCH IF NO
034360  062700  000002                          ADD     #2,R0             ;STEP R0 TO NEXT SL DATA
034364  000413                                  BR      TERM              ;GO TERMINATE LINE
034366  012027                          9$:     MOV     (R0)+,(PC)+       ;GET SP ERROR DATA
034370  000000                          SPDATA: .WORD
034372  104400  034636                          TYPE    ,FOURSP           ;TYPE FOUR SPACES
034376  016746  177766                          MOV     SPDATA,-(SP)      ;;SAVE SPDATA FOR TYPEOUT
                                                                          ;;GO TYPE IT IN OCTAL
034402  104402                                  TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
034404  016746  177760                          MOV     SPDATA,-(SP)      ;PUT DATA ON STACK
034410  004767  000106                          JSR     PC,TYPEBN         ;GO TYPE BINARY
034414  104400                          TERM:   TYPE                      ;TYPE A CR LF
034416  001203                                  $CRLF
```

```
034420  000674                                   BR      NEXTEN          ;GO CHECK NEXT ENTRY
034422  026727  144526  157774  $$DONE: CMP      $REGO,#157774   ;DID BUFFER OVERFLOW?
034430  001033                                   BNE     1$              ;BRANCH IF NO
034432  104400  034440                           TYPE    ,65$            ;;TYPE ASCIZ STRING
034436  000430                                   BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:   .ASCIZ  <15><12>/PROBABLY MORE ERRORS BUT BUFFER OVERFLOWED/<15><12>
034520                                  64$:
034520  000207                          1$:      RTS     PC              ;RETURN TO $ERROR

034522  016602  000002                  TYPEBN:  MOV     2(SP),R2        ;GET NUMBER TO BE TYPED
034526  104400                                   TYPE                    ;TYPE FOUR SPACES
034530  034636                                   FOURSP
034532  012703  000010                           MOV     #10,R3          ;SETUP SOB COUNT
034536  112767  000060  000030  3$:      MOVB    #60,BINARY      ;PUT ASCII 0 IN LOCATION BINARY
034544  006102                                   ROL     R2              ;GET BIT TO BE TYPED
034546  103005                                   BCC     1$              ;BRANCH IF IT IS 0
034550  005567  000020                           ADC     BINARY          ;MAKE IT ASCII

034554  104400  034574                           TYPE    ,BINARY         ;GO TYPE IT
034560  000402                                   BR      2$              ;GO TO END OF LOOP
034562  104400  034637                  1$:      TYPE    ,THRESP         ;BIT WAS 0 TYPE 3 SPACES
034566  077315                          2$:      SOB     R3,3$
034570  012616                                   MOV     (SP)+,(SP)      ;READ JUST STACK
034572  000207                                   RTS     PC              ;RETURN
034574     000     040     040  BINARY:  .BYTE   0,40,40,0       ;STORAGE FOR ASCII CHARACTERS & TERMINATOR
034577     000
034600  020040  020040  020040  SP38:    .ASCII  /                                      /
034606  020040  020040  020040
034614  020040  020040  020040
034622  020040  020040  020040  SP16:    .ASCII  /                /
034630  020040  020040     040
034635     040                  FIVESP:  .ASCII  / /
034636     040                  FOURSP:  .ASCII  / /
034637     040                  THRESP:  .ASCII  / /
034640  020040     000          TWOSP:   .ASCIZ  /  /
        034644                                   .EVEN
                                ;;*******************************************************************
                                .SBTTL  MONITOR RESTORE ROUTINE
                                ;*THIS ROUTINE IS ENTERED BY TYPING A CHARACTER ON THE KEYBOARD
                                ;*IF THE CHARACTER IS NOT A CTRL C EXECUTION IS RETURNED TO THE
                                ;*TEST FOLLOWING THE ONE THAT WAS INTERRUPTED.
                                ;*IF IT IS A CTRL C THE MONITOR IS RESTORED AND THE
                                ;*PROCESSOR HALTS.
034644  005003                  QUIT:    CLR     R3              ;NOT ENTERED THROUGH XXDP CHAIN
034646  017700  144266                   MOV     @$TKB,R0        ;GET CHARACTER
034652  042700  000200                   BIC     #BIT7,R0        ;GET RID OF PARITY BIT
034656  022700  000003                   CMP     #3,R0           ;WAS IT A CONTROL C?
034662  001041                           BNE     DUMMY           ;BRANCH IF NO
034664  000005                           RESET                   ;CLEAR THE WORLD
034666  104400  034674                   TYPE    ,65$            ;;TYPE ASCIZ STRING
034672  000403                           BR      64$             ;;GET OVER THE ASCIZ
                                ;;65$:  .ASCIZ  /^C/<15><12>
034702                          64$:
034702  012700  002734          CHAINQ:  MOV     #^D1500,R0      ;SETUP SOB COUNT
034706  012701  073300                   MOV     #SUBTAB+2,R1    ;GET END ADDRESS OF PROGRAM
034712  012702  160000                   MOV     #160000,R2      ;GET TOP ADDRESS OF MONITOR
```

```
034716  012142                          1$:     MOV     (R1)+,-(R2)     ;RESTORE THE MONITOR
034720  077002                                  SOB     R0,1$
034722  005303                                  DEC     R3
034724  001001                                  BNE     2$
034726  000207                                  RTS     PC
034730                                  2$:
034730  104400  034736                          TYPE    ,65$            ;;TYPE ASCIZ STRING
034734  000413                                  BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  /MONITOR RESTORED/<15><12><15><12>
034764                                  64$:
034764  000000                                  HALT
034766  005077  144146                  DUMMY:  CLR     @$TKB           ;CLEAR KEYBOARD BUFFER
034772  152777  000100  144136                  BISB    #BIT6,@$TKS     ;RESET INTERRUPT ENABLE BIT
035000  104400  035006                          TYPE    ,65$            ;;TYPE ASCIZ STRING
035004  000417                                  BR      64$             ;;GET OVER THE ASCIZ
                                        ;;65$:  .ASCIZ  <15><12>/TYPE A CTRL C TO QUIT!!!!/<15><12>
035044                                  64$:
035044  000177  144220                          JMP     @NEXTTST        ;RETURN
                                        ;;*****************************************************************
                                        .SBTTL  CHECK TEST SEQUENCE ROUTINE
                                        ;*THIS ROUTINE IS CALLED IN THE SCOPE ROUTINE. IT VERIFYS
                                        ;*THAT A TEST HAS NOT BEEN SKIPPED.
035050  016605  000002                  SEQENC: MOV     2(SP),R5        ;GET ADDRESS OF SCOPE+2
035054  162705  000002                          SUB     #2,R5           ;GET ADDRESS OF SCOPE CALL
035060  005004                                  CLR     R4              ;ENSURE R4 CLEAR
035062  116704  144014                          MOVB    $TSTNM,R4       ;GET TEST NUMBER
035066  006104                                  ROL     R4              ;ADJUST
035070  026405  073124                          CMP     ADRTAB(R4),R5   ;HAS TEST BEEN SKIPPED?
035074  001523                                  BEQ     1$              ;BRANCH IF NO
035076  006004                                  ROR     R4
035100  005304                                  DEC     R4
035102  010467  144046                          MOV     R4,$REG0        ;SAVE PREVIOUS TST NUM FOR TYPOUT
                                        ;
                                        ;FIND OUT WHICH TEST THIS IS.
035106  012767  000002  144046                  MOV     #2,$TMP0        ;SET BUFFER HEADER POINTER
035114  012767  000176  144042                  MOV     #176,$TMP1      ;SET BUFFER END POINTER
035122  012703  000100                          MOV     #100,R3         ;SET R3 AT MIDDLE OF BUFFER
035126  026305  073124                  4$:     CMP     ADRTAB(R3),R5   ;IS IT THIS TEST?
035132  001426                                  BEQ     2$              ;BRANCH IF YES
035134  101014                                  BHI     3$              ;MOVE UP TABLE
                                        ;CORRECT TEST IS FURTHER DOWN TABLE
035136  010367  144020                          MOV     R3,$TMP0        ;UPDATE HEADER POINTER
035142  016702  144016                          MOV     $TMP1,R2        ;GET END POINTER
035146  166702  144010                          SUB     $TMP0,R2        ;FIND RANGE OF REMAINING TABLE
035152  000241                                  CLC
035154  006002                                  ROR     R2              ;FIND MIDPOINT OF RANGE
035156  060203                                  ADD     R2,R3           ;MAKE R3 MID POINT OF REMAINING TABLE
035160  042703  000001                  5$:     BIC     #BIT0,R3        ;ENSURE EVEN ADDRESS
035164  000760                                  BR      4$              ;GO CHECK AGAIN
                                        ;
                                        ;CORRECT TEST IS FURTHER UP THE TABLE
035166  010367  143772                  3$:     MOV     R3,$TMP1        ;UPDATE END POINTER
035172  010302                                  MOV     R3,R2           ;GET END POINTER
035174  166702  143762                          SUB     $TMP0,R2        ;FIND RANGE OF REMAINING TABLE
035200  000241                                  CLC
035202  006002                                  ROR     R2              ;FIND MID POINT OF RANGE
```

```
035204  160203                  SUB     R2,R3                   ;MAKE R3 MIDPOINT OF REMAINING TABLE
035206  000764                  BR      5$                      ;MAKE EVEN ADDRESS AND CHECK AGAIN
                        ;
                        ;FOUND THE CORRECT TEST
035210  000241          2$:     CLC
035212  006003                  ROR     R3                      ;GET TEST NUMBER
035214  110367  143662          MOVB    R3,$TSTNM               ;UPDATE TEST NUMBER
035220  010367  143732          MOV     R3,$REG1                ;SAVE FOR TYPEOUT
035224  104400  035232          TYPE    ,65$                    ;;TYPE ASCIZ STRING
035230  000404                  BR      64$                     ;;GET OVER THE ASCIZ
                        ;;65$:  .ASCIZ  /TEST  /
035242                  64$:
035242  016746  143706          MOV     $REG0,-(SP)             ;;SAVE $REG0 FOR TYPEOUT
035246  104402                  TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
035250  104400  035256          TYPE    ,67$                    ;;TYPE ASCIZ STRING
035254  000426                  BR      66$                     ;;GET OVER THE ASCIZ
                        ;;67$:  .ASCIZ  /  FAILED, CAUSING ECECUTION TO GO TO TEST  /
035332                  66$:
035332  016746  143620          MOV     $REG1,-(SP)             ;;SAVE $REG1 FOR TYPEOUT
035336  104402                  TYPOC                           ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
035340  104400  001203          TYPE    ,$CRLF
035344  000207          1$:     RTS     PC                      ;RETURN
                        ;;*******************************************************************


                        .SBTTL  TYPE ROUTINE

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

035346  105767  143577  $TYPE:  TSTB    $TPFLG                  ;;IS THERE A TERMINAL?
035352  100002                  BPL     1$                      ;;BR IF YES
035354  000000                  HALT                            ;;HALT HERE IF NO TERMINAL
035356  000407                  BR      3$                      ;;LEAVE
035360  010046          1$:     MOV     R0,-(SP)                ;;SAVE R0
035362  017600  000002          MOV     @2(SP),R0               ;;GET ADDRESS OF ASCIZ STRING
035366  112046          2$:     MOVB    (R0)+,-(SP)             ;;PUSH CHARACTER TO BE TYPED ONTO STACK
035370  001005                  BNE     4$                      ;;BR IF IT ISN'T THE TERMINATOR
035372  005726                  TST     (SP)+                   ;;IF TERMINATOR POP IT OFF THE STACK
035374  012600                  MOV     (SP)+,R0                ;;RESTORE R0
035376  062716  000002  3$:     ADD     #2,(SP)                 ;;ADJUST RETURN PC
035402  000002                  RTI                             ;;RETURN
```

```
035404  122716  000011          4$:     CMPB    #HT,(SP)                ;;BRANCH IF <HT>
035410  001426                          BEQ     8$
035412  122716  000200                  CMPB    #CRLF,(SP)              ;;BRANCH IF NOT
035416  001004                          BNE     5$
035420  005726                          TST     (SP)+                   ;;POP  <CR><LF> EQUIV
035422  104400  001203                  TYPE    ,$CRLF
035426  000757                          BR      2$                      ;;GET NEXT CHARACTER
035430  004767  000056          5$:     JSR     PC,$TYPEC               ;;GO TYPE THIS CHARACTER
035434  126726  143510          6$:     CMPB    $FILLC,(SP)+            ;;IS IT TIME FOR FILLER CHARS.?
035440  001352                          BNE     2$                      ;;IF NO GO GET NEXT CHAR.
035442  016746  143500                  MOV     $NULL,-(SP)             ;;GET # OF FILLER CHARS. NEEDED
                                                                        ;;AND THE NULL CHAR.
035446  105366  000001          7$:     DECB    1(SP)                   ;;DOES A NULL NEED TO BE TYPED?
035452  002770                          BLT     6$                      ;;BR IF NO--GO POP THE NULL OFF OF STACK
035454  004767  000032                  JSR     PC,$TYPEC               ;;GO TYPE A NULL
035460  105367  000072                  DECB    $CHARCNT                ;;DON'T COUNT THE NULL AS A CHARACTER
035464  000770                          BR      7$                      ;;LOOP

                                ;;HORIZONTAL TAB PROCESSOR

035466  112716  000040          8$:     MOVB    #' ,(SP)                ;;REPLACE TAB WITH SPACE
035472  004767  000014          9$:     JSR     PC,$TYPEC               ;;TYPE A SPACE
035476  132767  000007  000052          BITB    #7,$CHARCNT             ;;BRANCH IF NOT AT
035504  001372                          BNE     9$                      ;;TAB STOP
035506  005726                          TST     (SP)+                   ;;POP SPACE OFF STACK
035510  000726                          BR      2$                      ;;GET NEXT CHARACTER
035512  105777  143424          $TYPEC: TSTB    @$TPS                   ;;WAIT UNTIL PRINTER IS READY
035516  100375                          BPL     $TYPEC
035520  116677  000002  143416          MOVB    2(SP),@$TPB             ;;LOAD CHAR TO BE TYPED INTO DATA REG.
035526  122766  000015  000002          CMPB    #CR,2(SP)               ;;BRANCH IF
035534  001003                          BNE     1$                      ;;NOT <CR>
035536  105067  000014                  CLRB    $CHARCNT                ;;
035542  000406                          BR      $TYPEX                  ;;EXIT
035544  122766  000012  000002  1$:     CMPB    #LF,2(SP)               ;;BRANCH IF
035552  001402                          BEQ     $TYPEX                  ;;<LF>
035554  105227                          INCB    (PC)+                   ;;INC SPACE
035556  000000                  $CHARCNT:.WORD  0                       ;;COUNT
035560  000207                  $TYPEX: RTS     PC

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

                                .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

035562  017646  000000                     $TYPOS:  MOV     @(SP),-(SP)         ;;PICKUP THE MODE
035566  116667  000001  000211                      MOVB    1(SP),$0FILL        ;;LOAD ZERO FILL SWITCH
035574  112667  000207                              MOVB    (SP)+,$OMODE+1      ;;NUMBER OF DIGITS TO TYPE
035600  062716  000002                              ADD     #2,(SP)             ;;ADJUST RETURN ADDRESS
035604  000406                                      BR      $TYPON
035606  112767  000001  000171     $TYPOC:  MOVB    #1,$0FILL           ;;SET THE ZERO FILL SWITCH
035614  112767  000006  000165              MOVB    #6,$OMODE+1         ;;SET FOR SIX(6) DIGITS
035622  112767  000005  000154     $TYPON:  MOVB    #5,$OCNT            ;;SET THE ITERATION COUNT
035630  010346                                      MOV     R3,-(SP)            ;;SAVE R3
035632  010446                                      MOV     R4,-(SP)            ;;SAVE R4
035634  010546                                      MOV     R5,-(SP)            ;;SAVE R5
035636  116704  000145                              MOVB    $OMODE+1,R4         ;;GET THE NUMBER OF DIGITS TO TYPE
035642  005404                                      NEG     R4
035644  062704  000006                              ADD     #6,R4               ;;SUBTRACT IT FOR MAX. ALLOWED
035650  110467  000132                              MOVB    R4,$OMODE           ;;SAVE IT FOR USE
035654  116704  000125                              MOVB    $0FILL,R4           ;;GET THE ZERO FILL SWITCH
035660  016605  000012                              MOV     12(SP),R5           ;;PICKUP THE INPUT NUMBER
035664  005003                                      CLR     R3                  ;;CLEAR THE OUTPUT WORD
035666  006105                             1$:      ROL     R5                  ;;ROTATE MSB INTO ''C''
035670  000404                                      BR      3$                  ;;GO DO MSB
035672  006105                             2$:      ROL     R5                  ;;FORM THIS DIGIT
035674  006105                                      ROL     R5
035676  006105                                      ROL     R5
035700  010503                                      MOV     R5,R3
035702  006103                             3$:      ROL     R3                  ;;GET LSB OF THIS DIGIT
035704  105367  000076                              DECB    $OMODE              ;;TYPE THIS DIGIT?
035710  100016                                      BPL     7$                  ;;BR IF NO
035712  042703  177770                              BIC     #177770,R3          ;;GET RID OF JUNK
035716  001002                                      BNE     4$                  ;;TEST FOR 0
035720  005704                                      TST     R4                  ;;SUPPRESS THIS 0?
035722  001403                                      BEQ     5$                  ;;BR IF YES
035724  005204                             4$:      INC     R4                  ;;DON'T SUPPRESS ANYMORE 0'S
035726  052703  000060                              BIS     #'0,R3              ;;MAKE THIS DIGIT ASCII
035732  052703  000040                     5$:      BIS     #' ,R3              ;;MAKE ASCII IF NOT ALREADY
035736  110367  000040                              MOVB    R3,8$               ;;SAVE FOR TYPING
035742  104400  036002                              TYPE    ,8$                 ;;GO TYPE THIS DIGIT
035746  105367  000032                     7$:      DECB    $OCNT               ;;COUNT BY 1
035752  003347                                      BGT     2$                  ;;BR IF MORE TO DO
035754  002402                                      BLT     6$                  ;;BR IF DONE
035756  005204                                      INC     R4                  ;;INSURE LAST DIGIT ISN'T A BLANK
035760  000744                                      BR      2$                  ;;GO DO THE LAST DIGIT
035762  012605                             6$:      MOV     (SP)+,R5            ;;RESTORE R5
035764  012604                                      MOV     (SP)+,R4            ;;RESTORE R4
035766  012603                                      MOV     (SP)+,R3            ;;RESTORE R3
035770  016666  000002  000004                      MOV     2(SP),4(SP)         ;;SET THE STACK FOR RETURNING
035776  012616                                      MOV     (SP)+,(SP)
036000  000002                                      RTI                         ;;RETURN
```

```
036002     000                     8$:     .BYTE   0                       ;;STORAGE FOR ASCII DIGIT
036003     000                             .BYTE   0                       ;;TERMINATOR FOR TYPE ROUTINE
036004     000                     $OCNT:  .BYTE   0                       ;;OCTAL DIGIT COUNTER
036005     000                     $OFILL: .BYTE   0                       ;;ZERO FILL SWITCH
036006  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

036010                             $TYPDS:
036010  010046                             MOV     R0,-(SP)                ;;PUSH R0 ON STACK
036012  010146                             MOV     R1,-(SP)                ;;PUSH R1 ON STACK
036014  010246                             MOV     R2,-(SP)                ;;PUSH R2 ON STACK
036016  010346                             MOV     R3,-(SP)                ;;PUSH R3 ON STACK
036020  010546                             MOV     R5,-(SP)                ;;PUSH R5 ON STACK
036022  012746  020200                     MOV     #20200,-(SP)            ;;SET BLANK SWITCH AND SIGN
036026  016605  000020                     MOV     20(SP),R5               ;;GET THE INPUT NUMBER
036032  100004                             BPL     1$                      ;;BR IF INPUT IS POS.
036034  005405                             NEG     R5                      ;;MAKE THE BINARY NUMBER POS.
036036  112766  000055  000001             MOVB    #'-,1(SP)               ;;MAKE THE ASCII NUMBER NEG.
036044  005000                     1$:     CLR     R0                      ;;ZERO THE CONSTANTS INDEX
036046  012703  036224                     MOV     #$DBLK,R3               ;;SETUP THE OUTPUT POINTER
036052  112723  000040                     MOVB    #' ,(R3)+               ;;SET THE FIRST CHARACTER TO A BLANK
036056  005002                     2$:     CLR     R2                      ;;CLEAR THE BCD NUMBER
036060  016001  036214                     MOV     $DTBL(R0),R1            ;;GET THE CONSTANT
036064  160105                     3$:     SUB     R1,R5                   ;;FORM THIS BCD DIGIT
036066  002402                             BLT     4$                      ;;BR IF DONE
036070  005202                             INC     R2                      ;;INCREASE THE BCD DIGIT BY 1
036072  000774                             BR      3$
036074  060105                     4$:     ADD     R1,R5                   ;;ADD BACK THE CONSTANT
036076  005702                             TST     R2                      ;;CHECK IF BCD DIGIT=0
036100  001002                             BNE     5$                      ;;FALL THROUGH IF 0
036102  105716                             TSTB    (SP)                    ;;STILL DOING LEADING 0'S?
036104  100407                             BMI     7$                      ;;BR IF YES
036106  106316                     5$:     ASLB    (SP)                    ;;MSD?
036110  103003                             BCC     6$                      ;;BR IF NO
036112  116663  000001  177777             MOVB    1(SP),-1(R3)            ;;YES--SET THE SIGN
036120  052702  000060             6$:     BIS     #'0,R2                  ;;MAKE THE BCD DIGIT ASCII
036124  052702  000040             7$:     BIS     #' ,R2                  ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
036130  110223                             MOVB    R2,(R3)+                ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
036132  005720                             TST     (R0)+                   ;;JUST INCREMENTING
036134  020027  000010                     CMP     R0,#10                  ;;CHECK THE TABLE INDEX
036140  002746                             BLT     2$                      ;;GO DO THE NEXT DIGIT
036142  003002                             BGT     8$                      ;;GO TO EXIT
036144  010502                             MOV     R5,R2                   ;;GET THE LSD
036146  000764                             BR      6$                      ;;GO CHANGE TO ASCII
036150  105726                     8$:     TSTB    (SP)+                   ;;WAS THE LSD THE FIRST NON-ZERO?
```

```
036152  100003                          BPL     9$                      ;;BR IF NO
036154  116663  177777  177776          MOVB    -1(SP),-2(R3)           ;;YES--SET THE SIGN FOR TYPING
036162  105013                  9$:     CLRB    (R3)                    ;;SET THE TERMINATOR
036164  012605                          MOV     (SP)+,R5                ;;POP STACK INTO R5
036166  012603                          MOV     (SP)+,R3                ;;POP STACK INTO R3
036170  012602                          MOV     (SP)+,R2                ;;POP STACK INTO R2
036172  012601                          MOV     (SP)+,R1                ;;POP STACK INTO R1
036174  012600                          MOV     (SP)+,R0                ;;POP STACK INTO R0
036176  104400  036224                  TYPE    ,$DBLK                  ;;NOW TYPE THE NUMBER
036202  016666  000002  000004          MOV     2(SP),4(SP)             ;;ADJUST THE STACK
036210  012616                          MOV     (SP)+,(SP)
036212  000002                          RTI                             ;;RETURN TO USER
036214  023420                  $DTBL:  10000.
036216  001750                          1000.
036220  000144                          100.
036222  000012                          10.
036224  000004                  $DBLK:  .BLKW   4
                                ;;*************************************************************

                                .SBTTL  TRAP DECODER

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

036234  010046                  $TRAP:  MOV     R0,-(SP)                ;;SAVE R0
036236  016600  000002                  MOV     2(SP),R0                ;;GET TRAP ADDRESS
036242  005740                          TST     -(R0)                   ;;BACKUP BY 2
036244  111000                          MOVB    (R0),R0                 ;;GET RIGHT BYTE OF TRAP
036246  016000  036254                  MOV     $TRPAD(R0),R0           ;;INDEX TO TABLE
036252  000200                          RTS     R0                      ;;GO TO ROUTINE


                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
036254                          $TRPAD:
036254  035346                          $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
036256  035606                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
036260  035562                          $TYPOS  ;;CALL=TYPOS    TRAP+4(104404)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
036262  035622                          $TYPON  ;;CALL=TYPON    TRAP+6(104406)  TYPE OCTAL NUMBER (AS PER LAST CALL)
036264  036010                          $TYPDS  ;;CALL=TYPDS    TRAP+10(104410) TYPE DECIMAL NUMBER (WITH SIGN)
                                ;;*************************************************************

                                .SBTTL  POWER DOWN AND UP ROUTINES

                                ;POWER DOWN ROUTINE
036266  012737  036410  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
036274  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;PRIO:7
036302  010046                          MOV     R0,-(SP)                ;;PUSH R0 ON STACK
036304  010146                          MOV     R1,-(SP)                ;;PUSH R1 ON STACK
```

```
036306  010246                              MOV     R2,-(SP)            ;;PUSH R2 ON STACK
036310  010346                              MOV     R3,-(SP)            ;;PUSH R3 ON STACK
036312  010446                              MOV     R4,-(SP)            ;;PUSH R4 ON STACK
036314  010546                              MOV     R5,-(SP)            ;;PUSH R5 ON STACK
036316  010667  000072                      MOV     SP,$SAVR6           ;;SAVE SP
036322  012737  036334  000024              MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
036330  000000                              HALT
036332  000776                              BR      .-2                 ;;HANG UP

                                    ;POWER UP ROUTINE
036334  016706  000054              $PWRUP: MOV     $SAVR6,SP           ;;GET SP
036340  005067  000050                      CLR     $SAVR6              ;;WAIT LOOP FOR THE TTY
036344  005267  000044              1$:     INC     $SAVR6              ;;WAIT FOR THE INC
036350  001375                              BNE     1$                  ;;OF  WORD
036352  012605                              MOV     (SP)+,R5            ;;POP STACK INTO R5
036354  012604                              MOV     (SP)+,R4            ;;POP STACK INTO R4
036356  012603                              MOV     (SP)+,R3            ;;POP STACK INTO R3
036360  012602                              MOV     (SP)+,R2            ;;POP STACK INTO R2
036362  012601                              MOV     (SP)+,R1            ;;POP STACK INTO R1
036364  012600                              MOV     (SP)+,R0            ;;POP STACK INTO R0
036366  012737  036266  000024              MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
036374  012737  000340  000026              MOV     #340,@#PWRVEC+2 ;;PRIO:7
036402  104400                              TYPE                        ;REPORT THE POWER FAILURE
036404  036416                      $PWRMG: .WORD   $POWER              ;;POWER FAIL MESSAGE POINTER
036406  000002                              RTI
036410  000000                      $ILLUP: HALT                        ;;THE POWER UP SEQUENCE WAS STARTED
036412  000776                              BR      .-2                 ;; BEFORE THE POWER DOWN WAS COMPLETE
036414  000000                      $SAVR6: 0                           ;;PUT THE SP HERE
036416  005015  047520  042527      $POWER: .ASCIZ  <15><12>'POWER''
036424  000122
                                            .EVEN
036426  041600  052520  052440      MSG1:   .ASCIZ<CRLF>    ''CPU UNDER TEST FOUND TO BE A ''
036434  042116  051105  052040
036442  051505  020124  047506
036450  047125  020104  047524
036456  041040  020105  020101
036464     000
036465     113  030502  026461      MSG3:   .ASCIZ  'KB11-B/C OR KB11-CM                    '<CRLF>
036472  027502  020103  051117
036500  045440  030502  026461
036506  046503  020040  020040
036514  020040  020040  020040
036522  020040  020040  020040
036530  000200
036532  020040  000040              MSG5:   .ASCIZ  ''  ''
036536  047200  052117  035105      MSG6:   .ASCIZ  <CRLF>'NOTE:''
036544     000
        036546                              .EVEN

036546  050123  020114  040506      EM1:    .ASCIZ  /SPL FAILED OR PSW PRIORITY BITS STUCK/
036554  046111  042105  047440
036562  020122  051520  020127
036570  051120  047511  044522
036576  054524  041040  052111
036604  020123  052123  041525
036612  000113
```

```
036614  051105  047522  020122  DH1:    .ASCII  /ERROR PC    SPL 5 PSW       SPL 2 PSW    TST NUM/<CRLF>
036622  041520  020040  051440
036630  046120  032440  050040
036636  053523  020040  020040
036644  020040  051440  046120
036652  031040  050040  053523
036660  020040  052040  052123
036666  047040  046525     200
036673     011  042440  050130          .ASCIZ  /         EXPECT ACTUAL   EXPECT ACTUAL/
036700  041505  020124  041501
036706  052524  046101  020040
036714  042440  050130  041505
036722  020124  041501  052524
036730  046101     000
        036734                          .EVEN
036734  001116  001214  001206  DT1:    .WORD   $ERRPC,$PR5,$ERPSW,$PR2,$TMP0,$$TSTNM,0
036742  001212  001162  001210
036750  000000
036752  051120  032440  047440  EM2:    .ASCIZ  /PR 5 OK PR 2 BAD/
036760  020113  051120  031040
036766  041040  042101     000
036773     120  020122  020062  EM3:    .ASCIZ  /PR 2 OK BUT PR 5 BAD/
037000  045517  041040  052125
037006  050040  020122  020065
037014  040502  000104
037020  040522  043103  042440  EM4:    .ASCIZ  /RACF E3 BAD(NOT GOING HIGH ON SPL)/
037026  020063  040502  024104
037034  047516  020124  047507
037042  047111  020107  044510
037050  044107  047440  020116
037056  050123  024514     000
037063     105  051122  051117  DH4:    .ASCIZ  /ERRORPC TEST NUMBER/
037070  041520  052040  051505
037076  020124  052516  041115
037104  051105     000
        037110                          .EVEN
037110  001116  001210  000000  DT4:    .WORD   $ERRPC,$$TSTNM,0
037116  040522  043103  042440  EM5:    .ASCIZ  /RACF E5 BAD(AFIR04(1)*AFIR03(1)*AFIR05(1)*(RTS:CCOP))/
037124  020065  040502  024104
037132  043101  051111  032060
037140  030450  025051  043101
037146  051111  031460  030450
037154  025051  043101  051111
037162  032460  030450  025051
037170  051050  051524  041472
037176  047503  024520  000051
037204  042522  042523  020124  EM6:    .ASCIZ  /RESET DIDN'T SEND OUT INIT/
037212  044504  047104  052047
037220  051440  047105  020104
037226  052517  020124  047111
037234  052111     000
037237     122  041501  020106  EM7:    .ASCIZ  /RACF E5 BAD/
037244  032505  041040  042101
037252     000
037253     122  041501  020106  EM10:   .ASCIZ  /RACF E8 BAD/
037260  034105  041040  042101
```

```
037266     000
037267     122  041501  020106  EM11:   .ASCIZ  "RACF E17 BAD(AFIR07(0)*X/CLASS"
037274  030505  020067  040502
037302  024104  043101  051111
037310  033460  030050  025051
037316  027530  046103  051501
037324  000123
037326  040522  043103  042440  EM12:   .ASCIZ  "RACF E5 BAD(AFIR06(0)*X/CLASS)"
037334  020065  040502  024104
037342  043101  051111  033060
037350  030050  025051  027530
037356  046103  051501  024523
037364     000
037365     120  041103  042040  EM13:   .ASCIZ  /PCB DIDN'T LOAD FROM R5/
037372  042111  023516  020124
037400  047514  042101  043040
037406  047522  020115  032522
037414     000
037415     123  020120  044504  EM14:   .ASCIZ  /SP DIDN'T LOAD PROPERLY/
037422  047104  052047  046040
037430  040517  020104  051120
037436  050117  051105  054514
037444     000
037445     105  051122  051117  DH14:   .ASCII  /ERRORPC        SP        TST NUM/<CRLF>
037452  041520  020040  020040
037460  020040  051440  004520
037466  052040  052123  047040
037474  046525     200
037477     040  020040  020040          .ASCIZ  /         EXPECT  ACTUAL/
037504  020040  042440  050130
037512  041505  020124  040440
037520  052103  040525  000114
                                        .EVEN
037526  001116  001156  001154  DT14:   .WORD   $ERRPC,$REG1,$REG0,$$TSTNM,0
037534  001210  000000
037540  032522  042040  042111  EM15:   .ASCIZ  /R5 DIDN'T LOAD PROPERLY/
037546  023516  020124  047514
037554  042101  050040  047522
037562  042520  046122  000131
037570  051105  047522  050122  DH15:   .ASCII  /ERRORPC        R5        TST NUM/<CRLF>
037576  020103  020040  020040
037604  020040  032522  020040
037612  020040  020040  020040
037620  051524  020124  052516
037626  100115
037630  020011  054105  042520          .ASCIZ  /         EXPECT  ACTUAL/
037636  052103  020040  041501
037644  052524  046101     000
037651     122  041501  020106  EM16:   .ASCIZ  "RACF X/CLASS DIDN'T GO HIGH"
037656  027530  046103  051501
037664  020123  044504  047104
037672  052047  043440  020117
037700  044510  044107     000
037705     122  041501  020106  EM17:   .ASCIZ  /RACF E34 BAD OR NOT GETTING THRU RACL RADR05/
037712  031505  020064  040502
037720  020104  051117  047040
```

```
037726  052117  043440  052105
037734  044524  043516  052040
037742  051110  020125  040522
037750  046103  051040  042101
037756  030122  000065
037762  051107  045101  051440  EM20:   .ASCIZ  /GRAJ SC05 L DOESN'T GET THRU TO RACK BRCAB04 L AS A HIGH/
037770  030103  020065  020114
037776  047504  051505  023516
040004  020124  042507  020124
040012  044124  052522  052040
040020  020117  040522  045503
040026  041040  041522  041101
040034  032060  046040  040440
040042  020123  020101  044510
040050  044107     000
040053     122  020061  044504  EM21:   .ASCIZ  /R1 DIDN'T SHIFT/
040060  047104  052047  051440
040066  044510  052106     000
040073     122  020061  044123  EM22:   .ASCII  /R1 SHIFTED BUT CARRY DIDN'T SET/<CRLF>
040100  043111  042524  020104
040106  052502  020124  040503
040114  051122  020131  044504
040122  047104  052047  051440
040130  052105     200
040133     123  044510  052106          .ASCIZ  /SHIFT COUNTER COULD BE STUCK/
040140  041440  052517  052116
040146  051105  041440  052517
040154  042114  041040  020105
040162  052123  041525  000113
040170  051107  045101  051440  EM23:   .ASCIZ  /GRAJ SC05 L DOESN'T GET THRU TO RACK BRCAB04 L AS A LOW/
040176  030103  020065  020114
040204  047504  051505  023516
040212  020124  042507  020124
040220  044124  052522  052040
040226  020117  040522  045503
040234  041040  041522  041101
040242  032060  046040  040440
040250  020123  020101  047514
040256  000127
040260  051501  020110  044522  EM24:   .ASCIZ  /ASH RIGHT DIDN'T SIGN FILL/
040266  044107  020124  044504
040274  047104  052047  051440
040302  043511  020116  044506
040310  046114     000
040313     122  020061  044504  EM25:   .ASCIZ  /R1 DIDN'T SHIFT CORRECTLY/
040320  047104  052047  051440
040326  044510  052106  041440
040334  051117  042522  052103
040342  054514     000
040345     105  051122  051117  DH25:   .ASCII  /ERRORPC        R1        TST NUM/<CRLF>
040352  041520  020040  020040
040360  020040  051040  004461
040366  052040  052123  047040
040374  046525     200
040377     011  042440  050130          .ASCIZ  /        EXPECT  ACTUAL/
040404  041505  020124  040440
```

```
040412  052103  040525  000114
                                          .EVEN
040420  001116  001164  001156  DT25:     .WORD    $ERRPC,$TMP1,$REG1,$$TSTNM,0
040426  001210  000000
040432  030522  051440  044510  EM26:     .ASCIZ   /R1 SHIFTED BUT CARRY DIDN'T SET/
040440  052106  042105  041040
040446  052125  041440  051101
040454  054522  042040  042111
040462  023516  020124  042523
040470  000124
040472  051501  027110  030062  EM27:     .ASCIZ   /ASH.20 DIDN'T LOAD CC'S CORRECTLY/
040500  042040  042111  023516
040506  020124  047514  042101
040514  041440  023503  020123
040522  047503  051122  041505
040530  046124  000131
040534  030522  051440  044510  EM30:     .ASCIZ   /R1 SHIFTED WHEN SHIFT COUNT=0/
040542  052106  042105  053440
040550  042510  020116  044123
040556  043111  020124  047503
040564  047125  036524  000060
040572  051501  027110  030064  EM31:     .ASCIZ   /ASH.40 DIDN'T LOAD CC'S CORRECTLY/
040600  042040  042111  023516
040606  020124  047514  042101
040614  041440  023503  020123
040622  047503  051122  041505
040630  046124  000131
040634  040522  043103  052440  EM32:     .ASCIZ   'RACF U/CLASS DOESN'T GO HIGH ON ASH'
040642  041457  040514  051523
040650  042040  042517  047123
040656  052047  043440  020117
040664  044510  044107  047440
040672  020116  051501  000110
040700  051501  027110  030060  EM33:     .ASCIZ   /ASH.00 FAILED/
040706  043040  044501  042514
040714  000104
040716  051111  041103  042440  EM34:     .ASCIZ   /IRCB E35(B2) NOT GOING LOW/
040724  032463  041050  024462
040732  047040  052117  043440
040740  044517  043516  046040
040746  053517     000
040751     111  041522  020102  EM35:     .ASCII   /IRCB (MUL:ASHC)+MFP L DIDN'T GO LOW OR IRCB E37 BAD/<CRLF>
040756  046450  046125  040472
040764  044123  024503  046453
040772  050106  046040  042040
041000  042111  023516  020124
041006  047507  046040  053517
041014  047440  020122  051111
041022  041103  042440  033463
041030  041040  042101     200
041035     117  020122  051111            .ASCIZ   /OR IRCB E35(B1) STUCK LOW/
041042  041103  042440  032463
041050  041050  024461  051440
041056  052524  045503  046040
041064  053517     000
041067     122  041501  020105  EM36:     .ASCIZ   /RACE (MUL:ASHC+MFP) DIDN'T GO HIGH OR RACE E44 BAD/
```

```
041074  046450  046125  040472
041102  044123  025503  043115
041110  024520  042040  042111
041116  023516  020124  047507
041124  044040  043511  020110
041132  051117  051040  041501
041140  020105  032105  020064
041146  040502  000104
041152  040522  042503  042440  EM37:   .ASCIZ  /RACE E45 BAD (AFIR04(1)*(MUL:ASHC+MFP))/
041160  032464  041040  042101
041166  024040  043101  051111
041174  032060  030450  025051
041202  046450  046125  040472
041210  044123  025503  043115
041216  024520  000051
041222  040522  042503  042440  EM40:   .ASCIZ  /RACE E33 BAD (AFIR05(1)*(MUL:ASHC+MFP))/
041230  031463  041040  042101
041236  024040  043101  051111
041244  032460  030450  025051
041252  046450  046125  040472
041260  044123  025503  043115
041266  024520  000051
041272  030122  042040  042111  EM41:   .ASCIZ  /R0 DIDN'T SIGN FILL ON RIGHT SHIFT/
041300  023516  020124  044523
041306  047107  043040  046111
041314  020114  047117  051040
041322  043511  052110  051440
041330  044510  052106     000
041335     105  051122  051117  DH41:   .ASCII  /ERRORPC        R0        TST NUM/<CRLF>
041342  041520  020040  020040
041350  020040  020040  030122
041356  020011  052040  052123
041364  047040  046525     200
041371     011  020040  054105          .ASCIZ  /         EXPECT  ACTUAL/
041376  042520  052103  020040
041404  041501  052524  046101
041412     000
        041414                          .EVEN
041414  001116  001162  001154  DT41:   .WORD   $ERRPC,$TMP0,$REG0,$$TSTNM,0
041422  001210  000000
041426  040502  020104  041503  EM42:   .ASCIZ  /BAD CC'S ON RIGHT SHIFT/
041434  051447  047440  020116
041442  044522  044107  020124
041450  044123  043111  000124
041456  051105  047522  050122  DH42:   .ASCII  /ERRORPC        PSW       TST NUM/<CRLF>
041464  020103  020040  020040
041472  020040  051520  004527
041500  020040  051524  020124
041506  052516  100115
041512  020011  054105  042520          .ASCIZ  /         EXPECT  ACTUAL/
041520  052103  020040  041501
041526  052524  046101     000
        041534                          .EVEN
041534  001116  001162  001206  DT42:   .WORD   $ERRPC,$TMP0,$ERPSW,$$TSTNM,0
041542  001210  000000
041546  030122  030074  020076  EM43:   .ASCIZ  /R0<0> DIDN'T GO TO R1<15>/
```

```
041554  044504  047104  052047
041562  043440  020117  047524
041570  051040  036061  032461
041576  000076
041600  051105  047522  050122  DH43:   .ASCII  /ERRORPC        RO              R1          TST NUM/<CRLF>
041606  020103  020040  020040
041614  020040  030122  020011
041622  020040  020040  020040
041630  051040  004461  020040
041636  051524  020124  052516
041644  100115
041646  020011  054105  042520          .ASCIZ  /        EXPECT  ACTUAL   EXPECT  ACTUAL/
041654  052103  020040  041501
041662  052524  046101  020040
041670  042440  050130  041505
041676  020124  040440  052103
041704  040525  000114
                                        .EVEN
041710  001116  001162  001154  DT43:   .WORD   $ERRPC,$TMP0,$REG0,$TMP1,$REG1,$$TSTNM,0
041716  001164  001156  001210
041724  000000
041726  030122  042040  042111  EM44:   .ASCIZ  /R0 DIDN'T GET SHIFTED LEFT PROPERLY/
041734  023516  020124  042507
041742  020124  044123  043111
041750  042524  020104  042514
041756  052106  050040  047522
041764  042520  046122  000131
041772  040502  020104  041503  EM45:   .ASCIZ  /BAD CC'S ON LEFT SHIFT/
042000  051447  047440  020116
042006  042514  052106  051440
042014  044510  052106     000
042021     115  050106  020124  EM46:   .ASCIZ  /MFPT LOADED R0 INCORRECTLY/
042026  047514  042101  042105
042034  051040  020060  047111
042042  047503  051122  041505
042050  046124  000131
042054  051105  047522  050122  DH46:   .ASCII  /ERRORPC        RO          TST NUM/<CRLF>
042062  020103  020040  020040
042070  020040  030122  020011
042076  051524  020124  052516
042104  100115
042106  020040  020040  020040          .ASCIZ  /        EXPECT  ACTUAL/
042114  020040  054105  042520
042122  052103  020040  041501
042130  052524  046101     000
        042136                          .EVEN
042136  001116  001164  001162  DT46:   .WORD   $ERRPC,$TMP1,$TMP0,$$TSTNM,0
042144  001210  000000
042150  040502  020104  041503  EM47:   .ASCIZ  /BAD CC'S ON NO SHIFT/
042156  051447  047440  020116
042164  047516  051440  044510
042172  052106     000
042175     122  020061  044504  EM50:   .ASCIZ  /R1 DIDN'T ROTATE PROPERLY/
042202  047104  052047  051040
042210  052117  052101  020105
042216  051120  050117  051105
```

```
042224  054514     000
042227     102  052111  051440  EM51:    .ASCIZ  /BIT STUCK IN SC WITH 52 PATTERN/
042234  052524  045503  044440
042242  020116  041523  053440
042250  052111  020110  031065
042256  050040  052101  042524
042264  047122     000
042267     105  051122  051117  DH51:    .ASCII  /ERRORPC        R0              R1            C BIT        TST NUM/<CRLF>
042274  041520  020040  020040
042302  020040  051040  004460
042310  020040  020040  020040
042316  051040  004461  020040
042324  020040  020103  044502
042332  004524  052040  052123
042340  047040  046525     200
042345     011  042440  050130           .ASCIZ  /        EXPECT  ACTUAL  EXPECT  ACTUAL  EXPECT  ACTUAL/
042352  041505  020124  040440
042360  052103  040525  020114
042366  042440  050130  041505
042374  020124  040440  052103
042402  040525  020114  042440
042410  050130  041505  020124
042416  040440  052103  040525
042424  000114
                                         .EVEN
042426  001116  001162  001154  DT51:    .WORD   $ERRPC,$TMP0,$REG0,$TMP1,$REG1,$TMP2,$ERPSW,$$TSTNM,0
042434  001164  001156  001166
042442  001206  001210  000000
042450  044502  020124  052123  EM52:    .ASCIZ  /BIT STUCK IN SHIFT COUNTER WITH 25 PATTERN/
042456  041525  020113  047111
042464  051440  044510  052106
042472  041440  052517  052116
042500  051105  053440  052111
042506  020110  032462  050040
042514  052101  042524  047122
042522     000
042523     111  041522  020102  EM53:    .ASCIZ  /IRCB E35(B3) NOT GOING LOW/
042530  031505  024065  031502
042536  020051  047516  020124
042544  047507  047111  020107
042552  047514  000127
042556  051501  027103  030060  EM54:    .ASCIZ  /ASC.00 FAILED/
042564  043040  044501  042514
042572  000104
042574  024122  052515  035114  EM55:    .ASCIZ  /R(MUL:ASHC+MFP) FIELD IN INSTR DECODE ROM BAD/
042602  051501  041510  046453
042610  050106  020051  044506
042616  046105  020104  047111
042624  044440  051516  051124
042632  042040  041505  042117
042640  020105  047522  020115
042646  040502  000104
042652  051107  042101  042040  EM56:    .ASCIZ  /GRAD DR00 STUCK HIGH OR RACK E64(C1) BAD/
042660  030122  020060  052123
042666  041525  020113  044510
042674  044107  047440  020122
```

```
042702  040522  045503  042440
042710  032066  041450  024461
042716  041040  042101     000
042723     107  040522  020104  EM57:   .ASCIZ  /GRAD DROO STUCK LOW OR RACK E64(C1) BAD/
042730  051104  030060  051440
042736  052524  045503  046040
042744  053517  047440  020122
042752  040522  045503  042440
042760  032066  041450  024461
042766  041040  042101     000
042773     107  040522  020112  EM60:   .ASCII  /GRAJ SC=0 L NOT GETTING TO RACK E50(C1) OR/<CRLF>
043000  041523  030075  046040
043006  047040  052117  043440
043014  052105  044524  043516
043022  052040  020117  040522
043030  045503  042440  030065
043036  041450  024461  047440
043044  100122
043046  042440  030065  041040          .ASCIZ  / E50 BAD (FAILED LOW)/
043054  042101  024040  040506
043062  046111  042105  046040
043070  053517  000051
043074  030122  047440  020122  EM61:   .ASCIZ  /R0 OR R1 OR BOTH BAD ON POSITIVE MULTIPLICAND/
043102  030522  047440  020122
043110  047502  044124  041040
043116  042101  047440  020116
043124  047520  044523  044524
043132  042526  046440  046125
043140  044524  046120  041511
043146  047101  000104
043152  040502  020104  041503  EM62:   .ASCIZ  /BAD CC'S ON POSITIVE MULTIPLICAND/
043160  051447  047440  020116
043166  047520  044523  044524
043174  042526  046440  046125
043202  044524  046120  041511
043210  047101  000104
043214  030122  041040  042101  EM63:   .ASCIZ  /R0 BAD ON NEGATIVE MULTIPLICAND/
043222  047440  020116  042516
043230  040507  044524  042526
043236  046440  046125  044524
043244  046120  041511  047101
043252  000104
043254  030522  041040  042101  EM64:   .ASCIZ  /R1 BAD ON NEGATIVE MULTIPLICAND/
043262  047440  020116  042516
043270  040507  044524  042526
043276  046440  046125  044524
043304  046120  041511  047101
043312  000104
043314  040502  020104  041503  EM65:   .ASCIZ  /BAD CC'S DUE TO STATE MUL.50/
043322  051447  042040  042525
043330  052040  020117  052123
043336  052101  020105  052515
043344  027114  030065     000
043351     103  042040  042111  EM66:   .ASCIZ  /C DIDN'T SET ON OVERFLOW/
043356  023516  020124  042523
043364  020124  047117  047440
```

```
043372  042526  043122  047514
043400  000127
043402  020103  044504  047104  EM67:     .ASCIZ  /C DIDN'T SET ON UNDERFLOW/
043410  052047  051440  052105
043416  047440  020116  047125
043424  042504  043122  047514
043432  000127
043434  052515  027114  030060  EM70:     .ASCIZ  /MUL.00 FAILED/
043442  043040  044501  042514
043450  000104
043452  051501  027110  030063  EM72:     .ASCIZ  /ASH.30 DIDN'T LOAD CC'S CORRECTLY/
043460  042040  042111  023516
043466  020124  047514  042101
043474  041440  023503  020123
043502  047503  051122  041505
043510  046124  000131
043514  051501  027110  030464  EM73:     .ASCIZ  /ASH.41 FAILED/
043522  043040  044501  042514
043530  000104
043532  051501  027110  030464  EM74:     .ASCIZ  /ASH.41 DIDN'T LOAD CC'S CORRECTLY/
043540  042040  042111  023516
043546  020124  047514  042101
043554  041440  023503  020123
043562  047503  051122  041505
043570  046124  000131
043574  051111  043103  055040  EM75:     .ASCIZ  /IRCF Z2(1) DOESN'T GO HIGH OR RACK E49(B1) STUCK LOW/
043602  024062  024461  042040
043610  042517  047123  052047
043616  043440  020117  044510
043624  044107  047440  020122
043632  040522  045503  042440
043640  034464  041050  024461
043646  051440  052524  045503
043654  046040  053517     000
043661     103  023503  020123  EM76:     .ASCIZ  /CC'S BAD IN DVE.00/
043666  040502  020104  047111
043674  042040  042526  030056
043702  000060
043704  041503  051447  041040  EM77:     .ASCIZ  /CC'S BAD IN DVC.70/
043712  042101  044440  020116
043720  053104  027103  030067
043726     000
043727     122  041501  020113  EM100:    .ASCIZ  /RACK E50(B0) STUCK HIGH/
043734  032505  024060  030102
043742  020051  052123  041525
043750  020113  044510  044107
043756     000
```

```
043757     107  040522  020112  EM101:  .ASCIZ  /GRAJ DIV SUB L NOT GOING LOW OR NOT GETTING THRU/<CRLF>
043764  044504  020126  052523
043772  020102  020114  047516
044000  020124  047507  047111
044006  020107  047514  020127
044014  051117  047040  052117
044022  043440  052105  044524
044030  043516  052040  051110
044036  100125     000
044041     124  020117  040522          .ASCIZ  /TO RACK E49 OR E49(D0) STUCK HIGH/
044046  045503  042440  034464
044054  047440  020122  032105
044062  024071  030104  020051
044070  052123  041525  020113
044076  044510  044107     000
044103     121  047525  020124  EM102:  .ASCIZ  /QUOT OK REMAINDER BAD (ROM STATE FAILURE)/
044110  045517  051040  046505
044116  044501  042116  051105
044124  041040  042101  024040
044132  047522  020115  052123
044140  052101  020105  040506
044146  046111  051125  024505
044154     000
044155     122  041501  020113  EM103:  .ASCIZ  /RACK E49(A1) STUCK LOW/
044162  032105  024071  030501
044170  020051  052123  041525
044176  020113  047514  000127
044204  051107  045101  042040  EM104:  .ASCIZ  /GRAJ DIV SUB L NOT GOING HIGH OR RACK E49(D0) STUCK LOW/
044212  053111  051440  041125
044220  046040  047040  052117
044226  043440  044517  043516
044234  044040  043511  020110
044242  051117  051040  041501
044250  020113  032105  024071
044256  030104  020051  052123
044264  041525  020113  047514
044272  000127
044274  052521  047524  042511  EM105:  .ASCIZ  /QUTOIENT & REMAINDER BAD/
044302  052116  023040  051040
044310  046505  044501  042116
044316  051105  041040  042101
044324     000
044325     107  040522  020110  EM106:  .ASCIZ  /GRAH SR15 NOT GETTING TO RACK E64 OR E64(B0) STUCK HIGH/
044332  051123  032461  047040
044340  052117  043440  052105
044346  044524  043516  052040
044354  020117  040522  045503
044362  042440  032066  047440
044370  020122  033105  024064
044376  030102  020051  052123
044404  041525  020113  044510
044412  044107     000
044415     111  041522  020110  EM110:  .ASCIZ  /IRCH N(1) NOT GETTING TO RACK E63 OR E63(D0) STUCK HIGH/
044422  024116  024461  047040
044430  052117  043440  052105
044436  044524  043516  052040
```

```
044444  020117  040522  045503
044452  042440  031466  047440
044460  020122  033105  024063
044466  030104  020051  052123
044474  041525  020113  044510
044502  044107     000
044505     122  041501  020113  EM111:  .ASCIZ  /RACK E49(B1) STUCK HIGH/
044512  032105  024071  030502
044520  020051  052123  041525
044526  020113  044510  044107
044534     000
044535     107  040522  020112  EM112:  .ASCIZ  /GRAJ DIV QUIT L NOT GOING HIGH OR NOT GETTING/<CRLF>
044542  044504  020126  052521
044550  052111  046040  047040
044556  052117  043440  044517
044564  043516  044040  043511
044572  020110  051117  047040
044600  052117  043440  052105
044606  044524  043516  000200
044614  047524  051040  041501          .ASCIZ  /TO RACK E63 OR E63(C0) STUCK LOW/
044622  020113  033105  020063
044630  051117  042440  031466
044636  041450  024460  051440
044644  052524  045503  046040
044652  053517     000
044655     122  041501  020113  EM113:  .ASCIZ  /RACK E50(B0) STUCK LOW/
044662  032505  024060  030102
044670  020051  052123  041525
044676  020113  047514  000127
044704  040522  045503  042440  EM114:  .ASCIZ  /RACK E64(B0) STUCK LOW/
044712  032066  041050  024460
044720  051440  052524  045503
044726  046040  053517     000
044733     104  041526  031056  EM115:  .ASCIZ  /DVC.20,DVC.40,DVC.80,OR DVC.90 FAILED/
044740  026060  053104  027103
044746  030064  042054  041526
044754  034056  026060  051117
044762  042040  041526  034456
044770  020060  040506  046111
044776  042105     000
045001     102  042101  041440  EM117:  .ASCIZ  /BAD CC'S IN DVC.90 OR RACK E63(D0) STUCK LOW/
045006  023503  020123  047111
045014  042040  041526  034456
045022  020060  051117  051040
045030  041501  020113  033105
045036  024063  030104  020051
045044  052123  041525  020113
045052  047514  000127
045056  051107  045101  042040  EM120:  .ASCIZ  /GRAJ DIV QUIT DIDN'T GO LOW OR RACK E63(C0) STUCK HIGH/
045064  053111  050440  044525
045072  020124  044504  047104
045100  052047  043440  020117
045106  047514  020127  051117
045114  051040  041501  020113
045122  033105  024063  030103
045130  020051  052123  041525
```

```
045136  020113  044510  044107
045144     000
045145     103  023503  020123  EM121:  .ASCIZ  /CC'S BAD DUE TO EITHER DIV.30 OR DVE.20/
045152  040502  020104  052504
045160  020105  047524  042440
045166  052111  042510  020122
045174  044504  027126  030063
045202  047440  020122  053104
045210  027105  030062     000
045215     107  040522  020112  EM122:  .ASCIZ  /GRAJ E5 BAD(Z2(0)*LEFT SAVE(1))/
045222  032505  041040  042101
045230  055050  024062  024460
045236  046052  043105  020124
045244  040523  042526  030450
045252  024451     000
045255     122  041501  020113  EM123:  .ASCIZ  /RACK E63(D0) STUCK LOW/
045262  033105  024063  030104
045270  020051  052123  041525
045276  020113  047514  000127
045304  041503  051447  042040  EM124:  .ASCIZ  /CC'S DIDN'T LOAD PROPERLY/
045312  042111  023516  020124
045320  047514  042101  050040
045326  047522  042520  046122
045334  000131
045336  051107  045101  042440  EM125:  .ASCIZ  /GRAJ E5(N(1)*SR15(1)) BAD OR ROM STATE BAD/
045344  024065  024116  024461
045352  051452  030522  024065
045360  024461  020051  040502
045366  020104  051117  051040
045374  046517  051440  040524
045402  042524  041040  042101
045410     000
045411     121  047525  020124  EM127:  .ASCIZ  /QUOT BAD REMAINDER OK (ROM STATE FAILURE)/
045416  040502  020104  042522
045424  040515  047111  042504
045432  020122  045517  024040
045440  047522  020115  052123
045446  052101  020105  040506
045454  046111  051125  024505
045462     000
045463     121  047525  044524  EM130:  .ASCIZ  /QUOTIENT BAD (ROM STATE FAILURE)/
045470  047105  020124  040502
045476  020104  051050  046517
045504  051440  040524  042524
045512  043040  044501  052514
045520  042522  000051
045524  040502  020104  041503  EM134:  .ASCIZ  /BAD CC'S ON DIV OVERFLOW, STATE DVD.10/
045532  051447  047440  020116
045540  044504  020126  053117
045546  051105  046106  053517
045554  020054  052123  052101
045562  020105  053104  027104
045570  030061     000
045573     122  020060  040502  EM135:  .ASCIZ  /R0 BAD/
045600  000104
045602  052123  052101  020105  EM136:  .ASCIZ  /STATE MTP.00 DIDN'T INC SP/
```

```
045610  052115  027120  030060
045616  042040  042111  023516
045624  020124  047111  020103
045632  050123     000
045635     122  041501  020106   EM140:  .ASCIZ  'RACF X/CLASS DOESN'T GO HIGH''
045642  027530  046103  051501
045650  020123  047504  051505
045656  023516  020124  047507
045664  044040  043511  000110
045672  052115  027120  030061   EM141:  .ASCIZ  /MTP.10 FAILED TO RELOAD THE DR/
045700  043040  044501  042514
045706  020104  047524  051040
045714  046105  040517  020104
045722  044124  020105  051104
045730     000
045731     123  020120  047514   EM142:  .ASCIZ  /SP LOADED INCORRECTLY/
045736  042101  042105  044440
045744  041516  051117  042522
045752  052103  054514     000
045757     115  050124  030456   EM143:  .ASCIZ  /MTP.10 DIDN'T PUT PCB IN DR/
045764  020060  044504  047104
045772  052047  050040  052125
046000  050040  041103  044440
046006  020116  051104     000
046013     122  041501  020106   EM144:  .ASCIZ  /RACF E20(4) STUCK HIGH/
046020  031105  024060  024464
046026  051440  052524  045503
046034  044040  043511  000110
046042  043115  027120  030061   EM145:  .ASCIZ  /MFP.10 DIDN'T DECREMENT THE SP/
046050  042040  042111  023516
046056  020124  042504  051103
046064  046505  047105  020124
046072  044124  020105  050123
046100     000
046101     122  020060  044504   EM146:  .ASCIZ  /RO DIDN'T GET PUT ON THE STACK/
046106  047104  052047  043440
046114  052105  050040  052125
046122  047440  020116  044124
046130  020105  052123  041501
046136  000113
046140  040502  020104  041503   EM147:  .ASCIZ  /BAD CC'S, CC CONTROL ROM/
046146  051447  020054  041503
046154  041440  047117  051124
046162  046117  051040  046517
046170     000
046171     115  050106  030056   EM151:  .ASCIZ  /MFP.00 BAD/
046176  020060  040502  000104
046204  040502  020104  041503   EM152:  .ASCIZ  /BAD CC'S DUE TO MFP.00/
046212  051447  042040  042525
046220  052040  020117  043115
046226  027120  030060     000
046233     124  050122  030056   EM155:  .ASCIZ  /TRP.01 FAILED TO LOAD BR/
046240  020061  040506  046111
046246  042105  052040  020117
046254  047514  042101  041040
046262  000122
```

```
046264  051111  042103  044440  EM156:  .ASCII  /IRCD IOT DOESN'T GO LOW OR DAPE TV04 DOES/<CRLF>
046272  052117  042040  042517
046300  047123  052047  043440
046306  020117  047514  020127
046314  051117  042040  050101
046322  020105  053124  032060
046330  042040  042517  100123
046336  047516  020124  047507          .ASCIZ  /NOT GO HIGH OR DOESN'T GET TO THE ALU/
046344  044040  043511  020110
046352  051117  042040  042517
046360  047123  052047  043440
046366  052105  052040  020117
046374  044124  020105  046101
046402  000125
046404  051124  027120  030460  EM157:  .ASCIZ  /TRP.01 FAILED TO LOAD DR/
046412  043040  044501  042514
046420  020104  047524  046040
046426  040517  020104  051104
046434     000
046435     124  050122  030056  EM160:  .ASCIZ  /TRP.00 FAILED TO LOAD BR/
046442  020060  040506  046111
046450  042105  052040  020117
046456  047514  042101  041040
046464  000122
046466  051111  042103  047440  EM161:  .ASCII  /IRCD OPCODE3 DOESN'T GO LOW OR DOESN'T/<CRLF>
046474  041520  042117  031505
046502  042040  042517  047123
046510  052047  043440  020117
046516  047514  020127  051117
046524  042040  042517  047123
046532  052047     200
046535     107  052105  052040          .ASCIZ  /GET THRU TO DAPE TV03/
046542  051110  020125  047524
046550  042040  050101  020105
046556  053124  031460     000
046563     124  050122  030056  EM162:  .ASCIZ  /TRP.00 FAILED TO LOAD DR/
046570  020060  040506  046111
046576  042105  052040  020117
046604  047514  042101  042040
046612  000122
046614  044502  020124  040506  EM163:  .ASCIZ  /BIT FAILED IN PIRQ REG/
046622  046111  042105  044440
046630  020116  044520  050522
046636  051040  043505     000
046643     105  051122  051117  DH163:  .ASCII  /ERRORPC        PIRQ       TST NUM/<CRLF>
046650  041520  020040  020040
046656  020040  050040  051111
046664  004521  020040  051524
046672  020124  052516  100115
046700  020011  054105  042520          .ASCIZ  /         EXPECT  ACTUAL/
046706  052103  020040  041501
046714  052524  046101     000
        046722                          .EVEN
046722  001116  001162  001216  DT163:  .WORD   $ERRPC,$TMP0,$EPIRQ,$$TSTNM,0
046730  001210  000000
046734  042506  027124  030060  EM164:  .ASCIZ  /FET.00 HAD BAD BEN FIELD/
```

```
 7052  046742  044040  042101  041040
 7053  046750  042101  041040  047105
 7054  046756  043040  042511  042114
 7055  046764     000
 7056  046765     124  041515  020102  EM165:  .ASCIZ  /TMCB E62(1) BAD OR TMCB HONOR PIR 1 NOT GOING LOW/
 7057  046772  033105  024062  024461
 7058  047000  041040  042101  047440
 7059  047006  020122  046524  041103
 7060  047014  044040  047117  051117
 7061  047022  050040  051111  030440
 7062  047030  047040  052117  043440
 7063  047036  044517  043516  046040
 7064  047044  053517     000
 7065  047047     124  041515  020102  EM166:  .ASCII  /TMCB E51(9) OR E55(10,11) OR E62 BAD OR/<CRLF>
 7066  047054  032505  024061  024471
 7067  047062  047440  020122  032505
 7068  047070  024065  030061  030454
 7069  047076  024461  047440  020122
 7070  047104  033105  020062  040502
 7071  047112  020104  051117     200
 7072  047117     124  041515  020101           .ASCIZ  /TMCA INH BELOW BR6 STUCK LOW/
 7073  047124  047111  020110  042502
 7074  047132  047514  020127  051102
 7075  047140  020066  052123  041525
 7076  047146  020113  047514  000127
 7077  047154  046524  040503  040440  EM167:  .ASCIZ  /TMCA ABOVE BR7 MIGHT BE STUCK LOW/
 7078  047162  047502  042526  041040
 7079  047170  033522  046440  043511
 7080  047176  052110  041040  020105
 7081  047204  052123  041525  020113
 7082  047212  047514  000127
 7083  047216  046524  042503  041040  EM170:  .ASCIZ  /TMCE BRQ CLOCK MIGHT BE STUCK LOW/
 7084  047224  050522  041440  047514
 7085  047232  045503  046440  043511
 7086  047240  052110  041040  020105
 7087  047246  052123  041525  020113
 7088  047254  047514  000127
 7089  047260  046524  041103  050040  EM171:  .ASCII  /TMCB PF(0)*(SF+TF) NOT GOING HIGH OR NOT/<CRLF>
 7090  047266  024106  024460  024052
 7091  047274  043123  052053  024506
 7092  047302  047040  052117  043440
 7093  047310  044517  043516  044040
 7094  047316  043511  020110  051117
 7095  047324  047040  052117     200
 7096  047331     107  052105  044524           .ASCIZ  /GETTING TO RACK E50 OR RACK E50(A1) BAD/
 7097  047336  043516  052040  020117
 7098  047344  040522  045503  042440
 7099  047352  030065  047440  020122
 7100  047360  040522  045503  042440
 7101  047366  030065  040450  024461
 7102  047374  041040  042101     000
 7103  047401     124  041515  020102  EM172:  .ASCII  /TMCB PF(0)*(SF+-TF) NOT GOING LOW OR/<CRLF>
 7104  047406  043120  030050  025051
 7105  047414  051450  025506  052055
 7106  047422  024506  047040  052117
 7107  047430  043440  044517  043516
```

```
047436  046040  053517  047440
047444  100122
047446  047516  020124  042507                   .ASCII  /NOT GETTING TO RACK E64 OR RACK E64(A1) BAD/<CRLF>
047454  052124  047111  020107
047462  047524  051040  041501
047470  020113  033105  020064
047476  051117  051040  041501
047504  020113  033105  024064
047512  030501  020051  040502
047520  100104
047522  051117  052040  041515                   .ASCII  /OR TMCB PIRQ NOT GETTING TO DAPE OR DAPE TV05*07/<CRLF>
047530  020102  044520  050522
047536  047040  052117  043440
047544  052105  044524  043516
047552  052040  020117  040504
047560  042520  047440  020122
047566  040504  042520  052040
047574  030126  025065  033460
047602     200
047603     116  052117  043440                   .ASCIZ  /NOT GOING HIGH OR NOT GETTING TO THE ALU/
047610  044517  043516  044040
047616  043511  020110  051117
047624  047040  052117  043440
047632  052105  044524  043516
047640  052040  020117  044124
047646  020105  046101  000125
047654  046524  041103  042440  EM174:   .ASCIZ  /TMCB E62(2) BAD OR TMCB HONOR PIR 2 NOT GOING LOW/
047662  031066  031050  020051
047670  040502  020104  051117
047676  052040  041515  020102
047704  047510  047516  020122
047712  044520  020122  020062
047720  047516  020124  047507
047726  047111  020107  047514
047734  000127
047736  046524  041103  042440  EM175:   .ASCIZ  /TMCB E63 BAD/
047744  031466  041040  042101
047752     000
047753     114  053105  046105  EM176:   .ASCIZ  /LEVEL 2 INTERRUPT WHEN LEVEL 1 ON/
047760  031040  044440  052116
047766  051105  052522  052120
047774  053440  042510  020116
050002  042514  042526  020114
050010  020061  047117     000
050015     124  041515  020102  EM177:   .ASCIZ  /TMCB E62(3) BAD OR TMCB HONOR PIR 3 NOT GOING LOW/
050022  033105  024062  024463
050030  041040  042101  047440
050036  020122  046524  041103
050044  044040  047117  051117
050052  050040  051111  031440
050060  047040  052117  043440
050066  044517  043516  046040
050074  053517     000
050077     114  053105  046105  EM201:   .ASCIZ  /LEVEL 2 INTERRUPT WHEN CPU LEVEL 2 ON/
050104  031040  044440  052116
050112  051105  052522  052120
```

```
050120  053440  042510  020116
050126  050103  020125  042514
050134  042526  020114  020062
050142  047117     000
050145     105  051122  051117  DH201:  .ASCIZ  /ERRORPC  PIRQ     TST NUM/
050152  041520  020040  044520
050160  050522  020040  020040
050166  051524  020124  052516
050174  000115
                                        .EVEN
050176  001116  001216  001210  DT201:  .WORD   $ERRPC,$EPIRQ,$$TSTNM,0
050204  000000                          .
050206  046524  041103  042440  EM202:  .ASCIZ  /TMCB E62(5) BAD OR TMCA HONOR PIR 4 NOT GOING LOW/
050214  031066  032450  020051
050222  040502  020104  051117
050230  052040  041515  020101
050236  047510  047516  020122
050244  044520  020122  020064
050252  047516  020124  047507
050260  047111  020107  047514
050266  000127
050270  042514  042526  020114  EM204:  .ASCIZ  /LEVEL 3 INTERRUPT WHEN CPU LEVEL 3 ON/
050276  020063  047111  042524
050304  051122  050125  020124
050312  044127  047105  041440
050320  052520  046040  053105
050326  046105  031440  047440
050334  000116
050336  046524  041103  042440  EM205:  .ASCIZ  /TMCB E62(11) BAD OR TMCA HONOR PIR 5 NOT GOING LOW/
050344  031066  030450  024461
050352  041040  042101  047440
050360  020122  046524  040503
050366  044040  047117  051117
050374  050040  051111  032440
050402  047040  052117  043440
050410  044517  043516  046040
050416  053517     000
050421     114  053105  046105  EM207:  .ASCIZ  /LEVEL 4 INTERRUPT WHEN CPU LEVEL 4 ON/
050426  032040  044440  052116
050434  051105  052522  052120
050442  053440  042510  020116
050450  050103  020125  042514
050456  042526  020114  020064
050464  047117     000
050467     124  041515  020102  EM210:  .ASCIZ  /TMCB E51(11) BAD OR TMCB E55(8-9) BAD/
050474  032505  024061  030461
050502  020051  040502  020104
050510  051117  052040  041515
050516  020102  032505  024065
050524  026470  024471  041040
050532  042101     000
050535     124  041515  020102  EM211:  .ASCIZ  /TMCB E70(1) BAD OR TMCA HONOR PIR6 NOT GOING LOW/
050542  033505  024060  024461
050550  041040  042101  047440
050556  020122  046524  040503
050564  044040  047117  051117
```

```
050572  050040  051111  020066
050600  047516  020124  047507
050606  047111  020107  047514
050614  000127
050616  046524  041103  042440   EM212:  .ASCIZ  /TMCB E63(12) BAD OR TMCB E61(1) BAD/
050624  031466  030450  024462
050632  041040  042101  047440
050640  020122  046524  041103
050646  042440  030466  030450
050654  020051  040502  000104
050662  042514  042526  020114   EM213:  .ASCIZ  /LEVEL 5 INTERRUPT WHEN CPU LEVEL 5 ON/
050670  020065  047111  042524
050676  051122  050125  020124
050704  044127  047105  041440
050712  052520  046040  053105
050720  046105  032440  047440
050726  000116
050730  046524  041103  042440   EM214:  .ASCIZ  /TMCB E70(6) BAD OR TMCA HONOR PIR 7 NOT GOING LOW/
050736  030067  033050  020051
050744  040502  020104  051117
050752  052040  041515  020101
050760  047510  047516  020122
050766  044520  020122  020067
050774  047516  020124  047507
051002  047111  020107  047514
051010  000127
051012  042514  042526  020114   EM216:  .ASCIZ  /LEVEL 6 INTERRUPT WHEN CPU LEVEL 6 ON/
051020  020066  047111  042524
051026  051122  050125  020124
051034  044127  047105  041440
051042  052520  046040  053105
051050  046105  033040  047440
051056  000116
051060  044524  042515  052517   EM217:  .ASCIZ  /TIMEOUT ON DATI DIDN'T WORK/
051066  020124  047117  042040
051074  052101  020111  044504
051102  047104  052047  053440
051110  051117  000113
051114  046524  041503  040440   EM220:  .ASCII  /TMCC AERF(1) L NOT GOING LOW/<CRLF>
051122  051105  024106  024461
051130  046040  047040  052117
051136  043440  044517  043516
051144  046040  053517     200
051151     117  020122  046524           .ASCIZ  /OR TMCB E53(11) BAD/
051156  041103  042440  031465
051164  030450  024461  041040
051172  042101     000
051175     124  046511  047505   EM221:  .ASCIZ  /TIMEOUT ON DATO DIDN'T WORK/
051202  052125  047440  020116
051210  040504  047524  042040
051216  042111  023516  020124
051224  047527  045522     000
051231     124  041515  020102   EM222:  .ASCIZ  /TMCB PS07(0) NOT GETTING TO TMCB E77 OR E77 BAD/
051236  051520  033460  030050
051244  020051  047516  020124
051252  042507  052124  047111
```

```
051260  020107  047524  052040
051266  041515  020102  033505
051274  020067  051117  042440
051302  033467  041040  042101
051310     000
051311     102  032122  023040    EM223:  .ASCIZ  /BR4 & BR6 FAILED/
051316  041040  033122  043040
051324  044501  042514  000104
051332  051102  020064  040506    EM224:  .ASCII  /BR4 FAILED. EITHER TMCB HONOR BR4 NOT GOING LOW OR/<CRLF>
051340  046111  042105  020056
051346  044505  044124  051105
051354  052040  041515  020102
051362  047510  047516  020122
051370  051102  020064  047516
051376  020124  047507  047111
051404  020107  047514  020127
051412  051117     200
051415     124  041515  020102            .ASCIZ  /TMCB E62(4) BAD OR INTERRUPT OR BG LOGIC ON UBC BAD/
051422  033105  024062  024464
051430  041040  042101  047440
051436  020122  047111  042524
051444  051122  050125  020124
051452  051117  041040  020107
051460  047514  044507  020103
051466  047117  052440  041502
051474  041040  042101     000
051501     102  032122  043040    EM225:  .ASCII  /BR4 FAILED BR6 OK EITHER TMCB HONOR BR4 NOT GOING/<CRLF>
051506  044501  042514  020104
051514  051102  020066  045517
051522  042440  052111  042510
051530  020122  046524  041103
051536  044040  047117  051117
051544  041040  032122  047040
051552  052117  043440  044517
051560  043516     200
051563     114  053517  047440            .ASCIZ  /LOW OR TMCB E62(4) BAD/
051570  020122  046524  041103
051576  042440  031066  032050
051604  020051  040502  000104
051612  046524  040503  044040    EM226:  .ASCIZ  /TMCA HONOR BR5 NOT GOING LOW OR TMCB E62(6) BAD/
051620  047117  051117  041040
051626  032522  047040  052117
051634  043440  044517  043516
051642  046040  053517  047440
051650  020122  046524  041103
051656  042440  031066  033050
051664  020051  040502  000104
051672  046524  040503  044040    EM227:  .ASCIZ  /TMCA HONOR BR6 NOT GOING LOW OR TMCB E62(12) BAD/
051700  047117  051117  041040
051706  033122  047040  052117
051714  043440  044517  043516
051722  046040  053517  047440
051730  020122  046524  041103
051736  042440  031066  030450
051744  024462  041040  042101
051752     000
```

```
7332 051753    131 046105 055040 EM230:  .ASCII  /YEL ZONE FAILED EITHER TMCD SL YEL NOT GOING HIGH OR/<CRLF>
7333 051760 047117 020105 040506
7334 051766 046111 042105 042440
7335 051774 052111 042510 020122
7336 052002 046524 042103 051440
7337 052010 020114 042531 020114
7338 052016 047516 020124 047507
7339 052024 047111 020107 044510
7340 052032 044107 047440 100122
7341 052040 051117 052040 041515         .ASCII  /OR TMCA HONOR SLY NOT GOING LOW OR TMCB E70(3) BAD/<CRLF>
7342 052046 020101 047510 047516
7343 052054 020122 046123 020131
7344 052062 047516 020124 047507
7345 052070 047111 020107 047514
7346 052076 020127 051117 052040
7347 052104 041515 020102 033505
7348 052112 024060 024463 041040
7349 052120 042101    200
7350 052123    117 020122 042502         .ASCII  /OR BEN13 FAILED- EITHER TMCA HONOR SLY NOT GETTING/<CRLF>
7351 052130 030516 020063 040506
7352 052136 046111 042105 020055
7353 052144 044505 044124 051105
7354 052152 052040 041515 020101
7355 052160 047510 047516 020122
7356 052166 046123 020131 047516
7357 052174 020124 042507 052124
7358 052202 047111 100107
7359 052206 047524 052040 041515         .ASCIZ  /TO TMCB E53 OR E53(3) BAD/
7360 052214 020102 032505 020063
7361 052222 051117 042440 031465
7362 052230 031450 020051 040502
7363 052236 000104
7364 052240 046524 041503 042440 EM232:  .ASCIZ  /TMCC E16(8) NOT GOING LOW OR TMCC E36 BAD/
7365 052246 033061 034050 020051
7366 052254 047516 020124 047507
7367 052262 047111 020107 047514
7368 052270 020127 051117 052040
7369 052276 041515 020103 031505
7370 052304 020066 040502 000104
7371 052312 042120 041522 051440 EM233:  .ASCII  /PDRC STACK LIMIT NOT GETTING TO TMCD AS A LOW OR/<CRLF>
7372 052320 040524 045503 046040
7373 052326 046511 052111 047040
7374 052334 052117 043440 052105
7375 052342 044524 043516 052040
7376 052350 020117 046524 042103
7377 052356 040440 020123 020101
7378 052364 047514 020127 051117
7379 052372    200
7380 052373    124 041515 020104         .ASCIZ  /TMCD E8 BAD/
7381 052400 034105 041040 042101
7382 052406    000
7383 052407    125 041502 020103 EM234:  .ASCII  /UBCC DATI NOT GETTING TO TMCC AS A LOW OR EITHER/<CRLF>
7384 052414 040504 044524 047040
7385 052422 052117 043440 052105
7386 052430 044524 043516 052040
7387 052436 020117 046524 041503
```

```
052444  040440  020123  020101
052452  047514  020127  051117
052460  042440  052111  042510
052466  100122
052470  046524  041503  042440            .ASCIZ  /TMCC E30 OR E36 BAD/
052476  030063  047440  020122
052504  031505  020066  040502
052512  000104
052514  046524  042103  051440  EM235:    .ASCII  /TMCD SL RED NOT GOING LOW OR TMCC ABORT/<CRLF>
052522  020114  042522  020104
052530  047516  020124  047507
052536  047111  020107  047514
052544  020127  051117  052040
052552  041515  020103  041101
052560  051117  100124
052564  047516  020124  047507            .ASCII  /NOT GOING LOW /
052572  047111  020107  047514
052600  020127
052602  051117  052040  041515            .ASCIZ  /OR TMCB E50(6) DIDN'T GO HIGH ON TMCC SERF(1)L/
052610  020102  032505  024060
052616  024466  042040  042111
052624  023516  020124  047507
052632  044040  043511  020110
052640  047117  052040  041515
052646  020103  042523  043122
052654  030450  046051     000
052661     124  041515  020103  EM237:    .ASCII  /TMCC SERF(1) NOT GOING LOW OR/<CRLF>
052666  042523  043122  030450
052674  020051  047516  020124
052702  047507  047111  020107
052710  047514  020127  051117
052716     200
052717     116  052117  043440            .ASCIZ  /NOT GETTING TO TMCB E50(2&1)/
052724  052105  044524  043516
052732  052040  020117  046524
052740  041103  042440  030065
052746  031050  030446  000051
052754  046524  041103  050040  EM240:    .ASCII  /TMCB PF(0)*(SF+-TF) NOT GOING HIGH OR/<CRLF>
052762  024106  024460  024052
052770  043123  026453  043124
052776  020051  047516  020124
053004  047507  047111  020107
053012  044510  044107  047440
053020  100122
053022  047516  020124  042507            .ASCIZ  /NOT GETTING TO RACK BRCAB04/
053030  052124  047111  020107
053036  047524  051040  041501
053044  020113  051102  040503
053052  030102  000064
053056  041523  042503  051440  EM241:    .ASCII  /SCCE STACK OVERFLOW NOT GOING HIGH OR/<CRLF>
053064  040524  045503  047440
053072  042526  043122  047514
053100  020127  047516  020124
053106  047507  047111  020107
053114  044510  044107  047440
053122  100122
```

```
053124  047516  020124  042507          .ASCIZ  /NOT GETTING TO TMCD E31 OR E31 BAD/
053132  052124  047111  020107
053140  047524  052040  041515
053146  020104  031505  020061
053154  051117  042440  030463
053162  041040  042101     000
053167     120  051104  020103  EM242:  .ASCII  /PDRC RED ZONE NOT GOING HIGH OR/<CRLF>
053174  042522  020104  047532
053202  042516  047040  052117
053210  043440  044517  043516
053216  044040  043511  020110
053224  051117     200
053227     116  052117  043440          .ASCIZ  /NOT GETTING TO TMCD E31 OR TMCD E31 BAD OR SL REG BIT 0 BAD/
053234  052105  044524  043516
053242  052040  020117  046524
053250  042103  042440  030463
053256  047440  020122  046524
053264  042103  042440  030463
053272  041040  042101  047440
053300  020122  046123  051040
053306  043505  041040  052111
053314  030040  041040  042101
053322     000
053323     065  032062  030060  EM243:  .ASCIZ  /52400 PATTERN FAILED, 125000 PATTERN OK/
053330  050040  052101  042524
053336  047122  043040  044501
053344  042514  026104  030440
053352  032462  030060  020060
053360  040520  052124  051105
053366  020116  045517     000
053373     105  051122  051117  DH243:  .ASCII  /ERRORPC        SL REG      TST NUM/<CRLF>
053400  041520  020040  020040
053406  020040  046123  051040
053414  043505  020040  020040
053422  020040  051524  020124
053430  052516  100115
053434  020011  054105  042520          .ASCIZ  /        EXPECT  ACTUAL/
053442  052103  020040  041501
053450  052524  046101     000
        053456                          .EVEN
053456  001116  001162  001222  DT243:  .WORD   $ERRPC,$TMPO,E2STKLM,$$TSTNM,0
053464  001210  000000
053470  031061  030065  030060  EM244:  .ASCIZ  /125000 PATTERN FAILED 52400 PATTERN OK/
053476  050040  052101  042524
053504  047122  043040  044501
053512  042514  020104  031065
053520  030064  020060  040520
053526  052124  051105  020116
053534  045517     000
        053540                          .EVEN
053540  001116  001162  001220  DT244:  .WORD   $ERRPC,$TMPO,E1STKLM,$$TSTNM,0
053546  001210  000000
053552  041523  042503  051440  EM245:  .ASCII  /SCCE SL ADDRESS NOT GETTING TO TMCD OR/<CRLF>
053560  020114  042101  051104
053566  051505  020123  047516
053574  020124  042507  052124
```

```
053602  047111  020107  047524
053610  052040  041515  020104
053616  051117     200
053621     124  041515  020104          .ASCIZ  /TMCD E28 OR E14 BAD/
053626  031105  020070  051117
053634  042440  032061  041040
053642  042101     000
053645     124  041515  020104  EM246:  .ASCII  /TMCD LOW BYTE EN DOESN'T GO LOW OR/<CRLF>
053652  047514  020127  054502
053660  042524  042440  020116
053666  047504  051505  023516
053674  020124  047507  046040
053702  053517  047440  100122
053710  047516  020124  042507          .ASCIZ  /NOT GETTING THRU TO THE DMUX (PDRE) AS A LOW /
053716  052124  047111  020107
053724  044124  052522  052040
053732  020117  044124  020105
053740  046504  054125  024040
053746  042120  042522  020051
053754  051501  040440  046040
053762  053517  000040
053766  046524  042103  042040  EM247:  .ASCII  /TMCD DMX S1 STUCK HIGH OR IT DOESN'T/<CRLF>
053774  054115  051440  020061
054002  052123  041525  020113
054010  044510  044107  047440
054016  020122  052111  042040
054024  042517  047123  052047
054032     200
054033     107  052105  052040          .ASCIZ  /GET THRU TO THE DMUX(PDRE) AS A LOW/
054040  051110  020125  047524
054046  052040  042510  042040
054054  052515  024130  042120
054062  042522  020051  051501
054070  040440  046040  053517
054076     000
054077     102  052117  020110  EM250:  .ASCIZ  /BOTH PATTERNS FAILED/
054104  040520  052124  051105
054112  051516  043040  044501
054120  042514  000104
054124  051105  047522  050122  DH250:  .ASCII  /ERRORPC       SL REG          SL REG      TST NUM/<CRLF>
054132  020103  020040  020040
054140  051440  020114  042522
054146  004507  020040  020040
054154  020040  046123  051040
054162  043505  020040  020040
054170  052040  052123  047040
054176  046525     200
054201     011  042440  050130          .ASCIZ  /           EXPECT  ACTUAL  EXPECT  ACTUAL/
054206  041505  020124  040440
054214  052103  040525  020114
054222  042440  050130  041505
054230  020124  040440  052103
054236  040525  000114
                                        .EVEN
054242  001116  001162  001220  DT250:  .WORD   $ERRPC,$TMP0,E1STKLM,$TMP1,E2STKLM,$$TSTNM,0
054250  001164  001222  001210
```

```
054256  000000
054260  046524  042103  054440  EM251:.ASCII    /TMCD YEL ZONE DIDN'T GO LOW OR IT DIDN'T GET THRU TO E31/<CRLF>
054266  046105  055040  047117
054274  020105  044504  047104
054302  052047  043440  020117
054310  047514  020127  051117
054316  044440  020124  044504
054324  047104  052047  043440
054332  052105  052040  051110
054340  020125  047524  042440
054346  030463     200
054351     117  020122  046524          .ASCIZ  /OR TMCE CACHE BEND DIDN'T GO HIGH ON SL RED/
054356  042503  041440  041501
054364  042510  041040  047105
054372  020104  044504  047104
054400  052047  043440  020117
054406  044510  044107  047440
054414  020116  046123  051040
054422  042105     000
054425     124  041515  020104  EM252:  .ASCIZ  /TMCD YEL ZONE DOESN'T GO LOW ON ADR 240/
054432  042531  020114  047532
054440  042516  042040  042517
054446  047123  052047  043440
054454  020117  047514  020127
054462  047117  040440  051104
054470  031040  030064     000
054475     124  041515  020104  EM253:  .ASCIZ  /TMCD YEL ZONE DOESN'T GO LOW ON ADR 140/
054502  042531  020114  047532
054510  042516  042040  042517
054516  047123  052047  043440
054524  020117  047514  020127
054532  047117  040440  051104
054540  030440  030064     000
054545     124  041515  020104  EM254:  .ASCIZ  /TMCD YEL ZONE DOESN'T GO HIGH OR DIDN'T GET THRU TO E31/
054552  042531  020114  047532
054560  042516  042040  042517
054566  047123  052047  043440
054574  020117  044510  044107
054602  047440  020122  044504
054610  047104  052047  043440
054616  052105  052040  051110
054624  020125  047524  042440
054632  030463     000
054635     125  044516  052502  EM255:  .ASCIZ  /UNIBUS TIMEOUT BIT IN CPU ERROR REG DIDN'T SET/
054642  020123  044524  042515
054650  052517  020124  044502
054656  020124  047111  041440
054664  052520  042440  051122
054672  051117  051040  043505
054700  042040  042111  023516
054706  020124  042523  000124
054714  051105  047522  050122  DH255:  .ASCII  /ERRORPC     CPUERR REG   TST NUM/<CRLF>
054722  020103  020040  041440
054730  052520  051105  020122
054736  042522  020107  020040
054744  051524  020124  052516
```

```
054752  100115
054754  020011  054105  042520                    .ASCIZ  /        EXPECT  ACTUAL/
054762  052103  020040  041501
054770  052524  046101     000
054775     103  052520  042440        EM256:  .ASCIZ  /CPU ERRPROR REG DIDN'T CLEAR/
055002  051122  051120  051117
055010  051040  043505  042040
055016  042111  023516  020124
055024  046103  040505  000122
055032  042531  020114  047532        EM257:  .ASCIZ  /YEL ZONE BIT IN CPU ERROR REG DIDN'T SET/
055040  042516  041040  052111
055046  044440  020116  050103
055054  020125  051105  047522
055062  020122  042522  020107
055070  044504  047104  052047
055076  051440  052105     000
055103     124  041515  020104        EM260:  .ASCIZ  /TMCD E4(4) NOT GOING LOW OR E4 BAD/
055110  032105  032050  020051
055116  047516  020124  047507
055124  047111  020107  047514
055132  020127  051117  042440
055140  020064  040502  000104
055146  042522  042101  055040        EM261:  .ASCIZ  /READ ZONE BIT IN CPU ERROR REG DIDN'T SET/
055154  047117  020105  044502
055162  020124  047111  041440
055170  052520  042440  051122
055176  051117  051040  043505
055204  042040  042111  023516
055212  020124  042523  000124
055220  046524  042103  042440        EM262:  .ASCIZ  /TMCD E15(6) OR E18(14) OR E18 BAD/
055226  032461  033050  020051
055234  051117  042440  034061
055242  030450  024464  047440
055250  020122  030505  020070
055256  040502  000104
055262  047506  046114  053517        EM263:  .ASCII  /FOLLOWING IS A LIST OF THE STACK LIMIT REG/<CRLF>
055270  047111  020107  051511
055276  040440  046040  051511
055304  020124  043117  052040
055312  042510  051440  040524
055320  045503  046040  046511
055326  052111  051040  043505
055334     200
055335     046  051440  020120                .ASCII  /& SP VALUES THAT CAUSED AN ERROR. THEY ARE/<CRLF>
055342  040526  052514  051505
055350  052040  040510  020124
055356  040503  051525  042105
055364  040440  020116  051105
055372  047522  027122  052040
055400  042510  020131  051101
055406  100105
055410  051107  052517  042520                .ASCIZ  /GROUPED ACCORDING TO ERROR TYPES/
055416  020104  041501  047503
055424  042122  047111  020107
055432  047524  042440  051122
055440  051117  052040  050131
```

```
055446  051505     000
055451     105  051122  051117  DH263:  .ASCIZ  /ERRORPC TEST NUMBER/<CRLF>
055456  041520  052040  051505
055464  020124  052516  041115
055472  051105  000200
055476  051411  040524  045503  DH263A: .ASCII  /       STACK LIMIT REGISTER                    STACK POINTER/<CRLF>
055504  046040  046511  052111
055512  051040  043505  051511
055520  042524  004522  020011
055526  020040  020040  020040
055534  052123  041501  020113
055542  047520  047111  042524
055550  100122
055552  041517  040524  020114          .ASCII  /OCTAL    15 14 13 12 11 10  9  8      OCTAL/
055560  020040  030440  020065
055566  032061  030440  020063
055574  031061  030440  020061
055602  030061  020040  020071
055610  034040  020040  020040
055616  020040  041517  040524
055624     114
055625     040  020040  030440          .ASCIZ  /    15 14 13 12 11 10  9  8/<CRLF>
055632  020065  032061  030440
055640  020063  031061  030440
055646  020061  030061  020040
055654  020071  034040  000200
055662  042522  020104  051124  DH263B: .ASCIZ  /RED TRAP ON YEL ADR/
055670  050101  047440  020116
055676  042531  020114  042101
055704  000122
055706  042522  020104  051124  DH263C: .ASCIZ  /RED TRAP ON LEGAL ADR/
055714  050101  047440  020116
055722  042514  040507  020114
055730  042101  000122
055734  042531  046114  053517  DH263D: .ASCIZ  /YELLOW TRAP ON RED ADR/
055742  052040  040522  020120
055750  047117  051040  042105
055756  040440  051104     000
055763     131  046105  052040  DH263E: .ASCIZ  /YEL TRAP ON LEGAL ADR/
055770  040522  020120  047117
055776  046040  043505  046101
056004  040440  051104     000
056011     116  020117  051124  DH263F: .ASCIZ  /NO TRAP ON RED ADR/
056016  050101  047440  020116
056024  042522  020104  042101
056032  000122
056034  047516  052040  040522  DH263G: .ASCIZ  /NO TRAP ON YEL ADR/
056042  020120  047117  054440
056050  046105  040440  051104
056056     000
        056060                          .EVEN
056060  055662                  INDEX:  DH263B
056062  055706                          DH263C
056064  055734                          DH263D
056066  055763                          DH263E
056070  056011                          DH263F
```

```
056072  056034                          DH263G
056074  047507  047111  020107  EM264:  .ASCIZ  /GOING TO NEXT TEST/
056102  047524  047040  054105
056110  020124  042524  052123
056116     000

056117     116  052117  035105  EM265:  .ASCII  /NOTE: IF NONE OF THE ODD ADR ERRORS TRAP/<CRLF>
056124  044440  020106  047516
056132  042516  047440  020106
056140  044124  020105  042117
056146  020104  042101  020122
056154  051105  047522  051522
056162  052040  040522  100120
056170  020040  020040  020040          .ASCII  /       THEN EITHER TMCC ODD ADRS ERR NOT GETTING TO/<CRLF>
056176  044124  047105  042440
056204  052111  042510  020122
056212  046524  041503  047440
056220  042104  040440  051104
056226  020123  051105  020122
056234  047516  020124  042507
056242  052124  047111  020107
056250  047524     200
056253     040  020040  020040          .ASCII  /       TMCC BUS ERROR OR DAPB BAMX00 NOT/<CRLF>
056260  052040  041515  020103
056266  052502  020123  051105
056274  047522  020122  051117
056302  042040  050101  020102
056310  040502  054115  030060
056316  047040  052117     200
056323     040  020040  020040          .ASCII  /       GETTING TO TMCC E7 AS A HIGH/<CRLF><CRLF>
056330  043440  052105  044524
056336  043516  052040  020117
056344  046524  041503  042440
056352  020067  051501  040440
056360  044040  043511  100110
056366     200
056367     116  044505  044124          .ASCIZ  /NEITHER -BYIN NOR DATI CAUSED A TRAP/<CRLF>
056374  051105  026440  054502
056402  047111  047040  051117
056410  042040  052101  020111
056416  040503  051525  042105
056424  040440  052040  040522
056432  100120     000
056435     111  041522  020104  EM266:  .ASCII  /IRCD BYIN DOESN'T GET TO TMCC E12 AS A HIGH/<CRLF>
056442  054502  047111  042040
056450  042517  047123  052047
056456  043440  052105  052040
056464  020117  046524  041503
056472  042440  031061  040440
056500  020123  020101  044510
056506  044107     200
056511     117  020122  046524          .ASCIZ  /OR TMCC E12(9,8) BAD OR E7(2) BAD/
056516  041503  042440  031061
056524  034450  034054  020051
056532  040502  020104  051117
056540  042440  024067  024462
```

```
056546  041040  042101     000
056553     104  050101  020102  EM267:  .ASCIZ  /DAPB BAMX00 DOESN'T GET TO TMCC E7(4) OR E7 BAD/
056560  040502  054115  030060
056566  042040  042517  047123
056574  052047  043440  052105
056602  052040  020117  046524
056610  041503  042440  024067
056616  024464  047440  020122
056624  033505  041040  042101
056632     000
056633     122  041501  020103  EM270:  .ASCII  /RACC UBSC00 DOESN'T GET TO TMCC E5(13) AS/<CRLF>
056640  041125  041523  030060
056646  042040  042517  047123
056654  052047  043440  052105
056662  052040  020117  046524
056670  041503  042440  024065
056676  031461  020051  051501
056704     200
056705     101  046040  053517          .ASCIZ  /A LOW OR E5(13) BAD/
056712  047440  020122  032505
056720  030450  024463  041040
056726  042101     000
056731     122  041501  020103  EM271:  .ASCII  /RACC UBSC02 DOESN'T GET TO TMCC E12(5) AS/<CRLF>
056736  041125  041523  031060
056744  042040  042517  047123
056752  052047  043440  052105
056760  052040  020117  046524
056766  041503  042440  031061
056774  032450  020051  051501
057002     200
057003     101  044040  043511          .ASCIZ  /A HIGH OR E12(6) DOESN'T GO LOW OR E5(12) BAD/
057010  020110  051117  042440
057016  031061  033050  020051
057024  047504  051505  023516
057032  020124  047507  046040
057040  053517  047440  020122
057046  032505  030450  024462
057054  041040  042101     000
057061     123  031515  033465  EM272:  .ASCIZ  /SM357*SRC1 DATI FAILED TO TRAP/
057066  051452  041522  020061
057074  040504  044524  043040
057102  044501  042514  020104
057110  047524  052040  040522
057116  000120
057120  042117  020104  042101  EM273:  .ASCIZ  /ODD ADR BIT IN CPUERR REG DOESN'T SET/
057126  020122  044502  020124
057134  047111  041440  052520
057142  051105  020122  042522
057150  020107  047504  051505
057156  023516  020124  042523
057164  000124
057166  047516  052040  040522  EM274:  .ASCIZ  /NO TRAP ON DATO/
057174  020120  047117  042040
057202  052101  000117
057206  051520  032060  030450  EM275:  .ASCII  /PS04(1) DOESN'T GET TO TMCB E74(9) AS A HIGH/<CRLF>
057214  020051  047504  051505
```

```
057222  023516  020124  042507
057230  020124  047524  052040
057236  041515  020102  033505
057244  024064  024471  040440
057252  020123  020101  044510
057260  044107     200
057263     117  020122  052111           .ASCII  /OR IT DOESN'T GET TO E51(10) AS A LOW OR E51 BAD/<CRLF>
057270  042040  042517  047123
057276  052047  043440  052105
057304  052040  020117  032505
057312  024061  030061  020051
057320  051501  040440  046040
057326  053517  047440  020122
057334  032505  020061  040502
057342  100104
057344  051117  044440  041522           .ASCII  /OR IRCD RTT DOESN'T GET TO TMCB E74(11) AS A HIGH/<CRLF>
057352  020104  052122  020124
057360  047504  051505  023516
057366  020124  042507  020124
057374  047524  052040  041515
057402  020102  033505  024064
057410  030461  020051  051501
057416  040440  044040  043511
057424  100110
057426  051117  052040  041515           .ASCII  /OR TMCB HONOR T DIDN'T GO LOW OR TMCB TOK DID/<CRLF>
057434  020102  047510  047516
057442  020122  020124  044504
057450  047104  052047  043440
057456  020117  047514  020127
057464  051117  052040  041515
057472  020102  047524  020113
057500  044504  100104
057504  047516  020124  047507           .ASCIZ  /NOT GO LOW OR IT DIDN'T GET THRU TMCB E53/
057512  046040  053517  047440
057520  020122  052111  042040
057526  042111  023516  020124
057534  042507  020124  044124
057542  052522  052040  041515
057550  020102  032505  000063
057556  044502  020124  020064   EM276:  .ASCIZ  /BIT 4 IN PSW DOESN'T SET/
057564  047111  050040  053523
057572  042040  042517  047123
057600  052047  051440  052105
057606     000
057607     123  052105  044524   EM277:  .ASCIZ  /SETTING PS<08> FAILED TO DISABLE T BIT TRAP/
057614  043516  050040  036123
057622  034060  020076  040506
057630  046111  042105  052040
057636  020117  044504  040523
057644  046102  020105  020124
057652  044502  020124  051124
057660  050101     000
057663     124  041515  020102   EM300:  .ASCIZ  /TMCB TOK DOESN'T GET TO DAPE E7(11) AS A LOW OR E7 BAD/
057670  047524  020113  047504
057676  051505  023516  020124
057704  042507  020124  047524
```

```
057712  042040  050101  020105
057720  033505  030450  024461
057726  040440  020123  020101
057734  047514  020127  051117
057742  042440  020067  040502
057750  000104
057752  051111  042103  051040   EM301:  .ASCII  /IRCD RTT DOESN'T GET TO TMCB E74 AS A LOW/<CRLF>
057760  052124  042040  042517
057766  047123  052047  043440
057774  052105  052040  020117
060002  046524  041103  042440
060010  032067  040440  020123
060016  020101  047514  100127
060024  051117  042440  032067            .ASCIZ  /OR E74 BAD/
060032  041040  042101     000
060037     124  041515  020102   EM302:  .ASCIZ  /TMCB E58(5) DIDN'T GO LOW OR E64 BAD/
060044  032505  024070  024465
060052  042040  042111  023516
060060  020124  047507  046040
060066  053517  047440  020122
060074  033105  020064  040502
060102  000104
060104  046524  040503  040440   EM303:  .ASCII  /TMCA ABOVE BR7 DOESN'T GO LOW ON A YEL ZONE OR/<CRLF>
060112  047502  042526  041040
060120  033522  042040  042517
060126  047123  052047  043440
060134  020117  047514  020127
060142  047117  040440  054440
060150  046105  055040  047117
060156  020105  051117     200
060163     111  020124  047504            .ASCIZ  /IT DOESN'T GET TO TMCB E84(13) OR E84 BAD/
060170  051505  023516  020124
060176  042507  020124  047524
060204  052040  041515  020102
060212  034105  024064  031461
060220  020051  051117  042440
060226  032070  041040  042101
060234     000
060235     124  041515  020101   EM304:  .ASCIZ  /TMCA ABOVE BR7 DOESN'T GET TO TMCB E84(1) OR E84 BAD/
060242  041101  053117  020105
060250  051102  020067  047504
060256  051505  023516  020124
060264  042507  020124  047524
060272  052040  041515  020102
060300  034105  024064  024461
060306  047440  020122  034105
060314  020064  040502  000104
060322  046524  040503  040440   EM305:  .ASCIZ  /TMCA ABOVE BR7 DOESN'T GET TO TMCB E77(13) OR E77 BAD/
060330  047502  042526  041040
060336  033522  042040  042517
060344  047123  052047  043440
060352  052105  052040  020117
060360  046524  041103  042440
060366  033467  030450  024463
060374  047440  020122  033505
060402  020067  040502  000104
```

```
060410  046524  040503  041040  EM306:  .ASCII  /TMCA BLOCK BR4 DOESN'T GO LOW ON PIR4 OR/<CRLF>
060416  047514  045503  041040
060424  032122  042040  042517
060432  047123  052047  043440
060440  020117  047514  020127
060446  047117  050040  051111
060454  020064  051117     200
060461     111  020124  047504          .ASCIZ  /IT DOESN'T GET TO TMCB E77(5) OR E77 BAD/
060466  051505  023516  020124
060474  042507  020124  047524
060502  052040  041515  020102
060510  033505  024067  024465
060516  047440  020122  033505
060524  020067  040502  000104
060532  046524  040503  044440  EM307:  .ASCIZ  /TMCA INH BELOW PIR4 DOESN'T GO LOW ON PIR5/
060540  044116  041040  046105
060546  053517  050040  051111
060554  020064  047504  051505
060562  023516  020124  047507
060570  046040  053517  047440
060576  020116  044520  032522
060604     000
060605     124  041515  020101  EM310:  .ASCIZ  /TMCA INH BELOW PIR4 DOESN'T GO LOW ON BR5/
060612  047111  020110  042502
060620  047514  020127  044520
060626  032122  042040  042517
060634  047123  052047  043440
060642  020117  047514  020127
060650  047117  041040  032522
060656     000
060657     124  041515  020101  EM311:  .ASCIZ  /TMCA BLOCK LEVEL 4 DIDN'T GO LOW ON PIR6/
060664  046102  041517  020113
060672  042514  042526  020114
060700  020064  044504  047104
060706  052047  043440  020117
060714  047514  020127  047117
060722  050040  051111  000066
060730  046524  040503  040440  EM312:  .ASCII  /TMCA ABOVE BR7 DOESN'T GO LOW(ON PIR 7) OR/<CRLF>
060736  047502  042526  041040
060744  033522  042040  042517
060752  047123  052047  043440
060760  020117  047514  024127
060766  047117  050040  051111
060774  033440  020051  051117
061002     200
061003     111  020124  047504          .ASCIZ  /IT DOESN'T GET TO TMCB E77(1) OR E77 BAD/
061010  051505  023516  020124
061016  042507  020124  047524
061024  052040  041515  020102
061032  033505  024067  024461
061040  047440  020122  033505
061046  020067  040502  000104
061054  041125  042103  042440  EM313:  .ASCIZ  /UBCD EXT BRQ DIDN'T GO LOW ON HONOR BR5/
061062  052130  041040  050522
061070  042040  042111  023516
061076  020124  047507  046040
```

```
061104  053517  047440  020116
061112  047510  047516  020122
061120  051102  000065
061124  047111  020110  042502  EM314:  .ASCII  /INH BELOW BR6 DOESN'T GO LOW(ON BR6) OR IT DOES/<CRLF>
061132  047514  020127  051102
061140  020066  047504  051505
061146  023516  020124  047507
061154  046040  053517  047450
061162  020116  051102  024466
061170  047440  020122  052111
061176  042040  042517  100123
061204  047516  020124  042507          .ASCIZ  /NOT GET THRU TMCA E82(6)/
061212  020124  044124  052522
061220  052040  041515  020101
061226  034105  024062  024466
061234     000
061235     124  041515  020101  EM315:  .ASCII  /TMCA E67(8) DOESN'T GO LOW (ON SL YEL) OR IT IS NOT/<CRLF>
061242  033105  024067  024470
061250  042040  042517  047123
061256  052047  043440  020117
061264  047514  020127  047450
061272  020116  046123  054440
061300  046105  020051  051117
061306  044440  020124  051511
061314  047040  052117     200
061321     107  052105  044524          .ASCIZ  /GETTING TO E60(12) OR E60 BAD/
061326  043516  052040  020117
061334  033105  024060  031061
061342  020051  051117  042440
061350  030066  041040  042101
061356     000
061357     124  041515  020101  EM316:  .ASCIZ  /TMCA E81(12) DOESN'T GO LOW(ON PIR5) OR IT DOES/<CRLF>
061364  034105  024061  031061
061372  020051  047504  051505
061400  023516  020124  047507
061406  046040  053517  047450
061414  020116  044520  032522
061422  020051  051117  044440
061430  020124  047504  051505
061436  000200
061440  042507  020124  047524          .ASCIZ  /GET TO E83(10) OR E83 BAD/
061446  042440  031470  030450
061454  024460  047440  020122
061462  034105  020063  040502
061470  000104
061472  046524  040503  042440  EM317:  .ASCIZ  /TMCA E81(2) BAD/
061500  030470  031050  020051
061506  040502  000104
061512  046524  040503  042440  EM320:  .ASCII  /TMCA E67(8) DOESN'T GO LOW(ON PIR 7) OR IT DOES/<CRLF>
061520  033466  034050  020051
061526  047504  051505  023516
061534  020124  047507  046040
061542  053517  047450  020116
061550  044520  020122  024467
061556  047440  020122  052111
061564  042040  042517  100123
```

```
061572  047516  020124  042507                   .ASCIZ  /NOT GET TO E83(13) OR E83 BAD/
061600  020124  047524  042440
061606  031470  030450  024463
061614  047440  020122  034105
061622  020063  040502  000104
061630  041125  042103  042440   EM321:  .ASCIZ  /UBCD EXT BRQ DIDN'T GO LOW ON HONOR BR6/
061636  052130  041040  050522
061644  042040  042111  023516
061652  020124  047507  046040
061660  053517  047440  020116
061666  047510  047516  020122
061674  051102  000066
061700  046524  040503  042440   EM322:  .ASCIZ  /TMCA E67(8) DOESN'T GET TO E83(1) OR E83 BAD/
061706  033466  034050  020051
061714  047504  051505  023516
061722  020124  042507  020124
061730  047524  042440  031470
061736  030450  020051  051117
061744  042440  031470  041040
061752  042101     000
061755     124  041515  020101   EM323:  .ASCII  /TMCA E81(8) IS NOT GOING LOW OR IT IS NOT/<CRLF>
061762  034105  024061  024470
061770  044440  020123  047516
061776  020124  047507  047111
062004  020107  047514  020127
062012  051117  044440  020124
062020  051511  047040  052117
062026     200
062027     107  052105  044524           .ASCIZ  /GETTING TO E76(12) OR E76 BAD/
062034  043516  052040  020117
062042  033505  024066  031061
062050  020051  051117  042440
062056  033067  041040  042101
062064     000
062065     124  041515  020101   EM324:  .ASCIZ  /TMCA E67(8) NOT GETTING TO E76(13) OR E76 IS BAD/
062072  033105  024067  024470
062100  047040  052117  043440
062106  052105  044524  043516
062114  052040  020117  033505
062122  024066  031461  020051
062130  051117  042440  033067
062136  044440  020123  040502
062144  000104
062146  046524  040503  042440   EM325:  .ASCIZ  /TMCA E67(8) DOESN'T GET TO E76(1) OR E76 BAD/
062154  033466  034050  020051
062162  047504  051505  023516
062170  020124  042507  020124
062176  047524  042440  033067
062204  030450  020051  051117
062212  042440  033067  041040
062220  042101     000
062223     124  041515  020101   EM326:  .ASCIZ  /TMCA E67(6) NOT GETTING TO E69(12) OR E69 IS BAD/
062230  033105  024067  024466
062236  047040  052117  043440
062244  052105  044524  043516
062252  052040  020117  033105
```

```
062260  024071  031061  020051
062266  051117  042440  034466
062274  044440  020123  040502
062302  000104
062304  051520  020127  044502   EM327:  .ASCII  /PSW BIT 11 DOESN'T SET OR TMCF CLK HI PS DOES/<CRLF>
062312  020124  030461  042040
062320  042517  047123  052047
062326  051440  052105  047440
062334  020122  046524  043103
062342  041440  045514  044040
062350  020111  051520  042040
062356  042517  100123
062362  047516  020124  047507           .ASCIZ  /NOT GO LOW OR BIT 11 DOESN'T GET TO OR THRU THE DMUX/
062370  046040  053517  047440
062376  020122  044502  020124
062404  030461  042040  042517
062412  047123  052047  043440
062420  052105  052040  020117
062426  051117  052040  051110
062434  020125  044124  020105
062442  046504  054125     000
062447     107  040522  020103   EM330:  .ASCII  /GRAC GDREG SET 1 DOESN'T GO HIGH (ON PAD 5) OR/<CRLF>
062454  042107  042522  020107
062462  042523  020124  020061
062470  047504  051505  023516
062476  020124  047507  044040
062504  043511  020110  047450
062512  020116  040520  020104
062520  024465  047440  100122
062526  047504  051505  023516           .ASCIZ  /DOESN'T GET TO THE GD REG/
062534  020124  042507  020124
062542  047524  052040  042510
062550  043440  020104  042522
062556  000107
062560  051107  041501  043440   EM331:  .ASCII  /GRAC GSREG SET 1 DOESN'T GO HIGH(ON PAD 5) OR/<CRLF>
062566  051123  043505  051440
062574  052105  030440  042040
062602  042517  047123  052047
062610  043440  020117  044510
062616  044107  047450  020116
062624  040520  020104  024465
062632  047440  100122
062636  052111  042040  042517           .ASCIZ  /IT DOESN'T GET TO THE GS REG/
062644  047123  052047  043440
062652  052105  052040  020117
062660  044124  020105  051507
062666  051040  043505     000
062673     107  040522  020102   EM332:  .ASCIZ  /GRAB SRC SET 1 DOESN'T GO LOW OR GRAC E23(6) BAD/
062700  051123  020103  042523
062706  020124  020061  047504
062714  051505  023516  020124
062722  047507  046040  053517
062730  047440  020122  051107
062736  041501  042440  031462
062744  033050  020051  040502
062752  000104
```

```
062754  051107  041501  042440  EM333:  .ASCIZ  /GRAC E24(6) DOESN'T GO LOW/
062762  032062  033050  020051
062770  047504  051505  023516
062776  020124  047507  046040
063004  053517     000
063007     107  040522  020103  EM334:  .ASCIZ  /GRAC E23(4) DOESN'T GO LOW/
063014  031105  024063  024464
063022  042040  042517  047123
063030  052047  043440  020117
063036  047514  000127
063042  051107  041501  042440  EM335:  .ASCIZ  /GRAC E23(4) IS NOT GOING LOW/
063050  031462  032050  020051
063056  051511  047040  052117
063064  043440  044517  043516
063072  046040  053517     000
063077     120  051104  020104  EM336:  .ASCIZ  /PDRD PS11 DOESN'T GET TO GRAC AS A LOW/
063104  051520  030461  042040
063112  042517  047123  052047
063120  043440  052105  052040
063126  020117  051107  041501
063134  040440  020123  020101
063142  047514  000127
063146  042522  044507  052123  EM337:  .ASCIZ  /REGISTER SET 1 SOURCE HAS STUCK BITS/
063154  051105  051440  052105
063162  030440  051440  052517
063170  041522  020105  040510
063176  020123  052123  041525
063204  020113  044502  051524
063212     000
063213     105  051122  051117  DH337:  .ASCIZ  /ERRORPC PATTERN TEST NUMBER/
063220  041520  050040  052101
063226  042524  047122  052040
063234  051505  020124  052516
063242  041115  051105     000
        063250                          .EVEN
063250  001116  001164  001210  DT337:  .WORD   $ERRPC,$TMP1,$$TSTNM,0
063256  000000
063260  042522  044507  052123  EM340:  .ASCIZ  /REGISTER SET 1 DESTINATION HAS STUCK BITS/
063266  051105  051440  052105
063274  030440  042040  051505
063302  044524  040516  044524
063310  047117  044040  051501
063316  051440  052524  045503
063324  041040  052111  000123
063332  051107  041101  042040  EM341:  .ASCIZ  /GRAB DST SET 1 DOESN'T GO LOW ON R14/
063340  052123  051440  052105
063346  030440  042040  042517
063354  047123  052047  043440
063362  020117  047514  020127
063370  047117  051040  032061
063376     000
063377     107  040522  020102  EM342:  .ASCIZ  /GRAB SRC SET 1 DOESN'T GO LOW ON R14/
063404  051123  020103  042523
063412  020124  020061  047504
063420  051505  023516  020124
063426  047507  046040  053517
```

```
063434  047440  020116  030522
063442  000064
063444  030065  030060  020060  EM343:  .ASCIZ  /50000 PATTERN FAILED IN PSW/
063452  040520  052124  051105
063460  020116  040506  046111
063466  042105  044440  020116
063474  051520  000127
063500  033061  030064  030060  EM344:  .ASCIZ  /164000 PATTERN FAILED IN PSW/
063506  050040  052101  042524
063514  047122  043040  044501
063522  042514  020104  047111
063530  050040  053523     000
063535     120  053523  044040  EM345:  .ASCIZ  /PSW HIGH BYTE DIDN'T CLEAR/
063542  043511  020110  054502
063550  042524  042040  042111
063556  023516  020124  046103
063564  040505  000122
063570  051107  041101  042040  EM346:  .ASCIZ  /GRAB DST SET 1 DOESN'T GO LOW ON DF6*SUPER MODE/
063576  052123  051440  052105
063604  030440  042040  042517
063612  047123  052047  043440
063620  020117  047514  020127
063626  047117  042040  033106
063634  051452  050125  051105
063642  046440  042117  000105
063650  051107  041101  051440  EM347:  .ASCIZ  /GRAB SRC SET 1 DOESN'T GO LOW ON SF6*SUPER MODE/
063656  041522  051440  052105
063664  030440  042040  042517
063672  047123  052047  043440
063700  020117  047514  020127
063706  047117  051440  033106
063714  051452  050125  051105
063722  046440  042117  000105
063730  051107  041501  042440  EM350:  .ASCIZ  /GRAC E35(8) DOESN'T GO HIGH ON DF6*USER MODE/
063736  032463  034050  020051
063744  047504  051505  023516
063752  020124  047507  044040
063760  043511  020110  047117
063766  042040  033106  052452
063774  042523  020122  047515
064002  042504     000
064005     107  040522  020103  EM351:  .ASCIZ  /GRAC E15(6) DOESN'T GO HIGH ON SF6*USER MODE/
064012  030505  024065  024466
064020  042040  042517  047123
064026  052047  043440  020117
064034  044510  044107  047440
064042  020116  043123  025066
064050  051525  051105  046440
064056  042117  000105
064062  051525  051105  047440  EM352:  .ASCIZ  /USER OR SUPER SP DST FAILED BIT TEST/
064070  020122  052523  042520
064076  020122  050123  042040
064104  052123  043040  044501
064112  042514  020104  044502
064120  020124  042524  052123
064126     000
```

```
064127     125  042523  020122  EM353:  .ASCIZ  /USER OR SUPER SP SRC FAILED BIT TEST/
064134  051117  051440  050125
064142  051105  051440  020120
064150  051123  020103  040506
064156  046111  042105  041040
064164  052111  052040  051505
064172  000124
064174  051107  041501  042440  EM354:  .ASCIZ  /GRAC E20(4) NOT GOING LOW/
064202  030062  032050  020051
064210  047516  020124  047507
064216  047111  020107  047514
064224  000127
064226  051107  041501  042440  EM355:  .ASCIZ  /GRAC E20(12) NOT GOING LOW ON MTP*DM0*DF6*PS12/
064234  030062  030450  024462
064242  047040  052117  043440
064250  044517  043516  046040
064256  053517  047440  020116
064264  052115  025120  046504
064272  025060  043104  025066
064300  051520  031061     000
064305     107  040522  020103  EM356:  .ASCIZ  /GRAC E20(11) NOT GOING LOW/
064312  031105  024060  030461
064320  020051  047516  020124
064326  047507  047111  020107
064334  047514  000127
064340  051523  020120  040502  EM357:  .ASCII  /SSP BAD ON MTP EITHER GRAC GRWE/<CRLF>
064346  020104  047117  046440
064354  050124  042440  052111
064362  042510  020122  051107
064370  041501  043440  053522
064376  100105
064400  044510  020102  051117          .ASCIZ  /HIB OR LOB NOT GOING LOW/
064406  046040  041117  047040
064414  052117  043440  044517
064422  043516  046040  053517
064430     000
064431     105  051122  051117  DH357:  .ASCII  /ERRORPC       SSP        TST NUM/<CRLF>
064436  041520  020040  020040
064444  020040  051523  004520
064452  051524  020124  052516
064460  100115
064462  020011  054105  042520          .ASCIZ  /         EXPECT  ACTUAL/
064470  052103  020040  041501
064476  052524  046101     000
064503     107  040522  020103  EM360:  .ASCIZ  /GRAC E35(8) DIDN'T GO HIGH ON MTP*DM0*DF6*PS13/
064510  031505  024065  024470
064516  042040  042111  023516
064524  020124  047507  044040
064532  043511  020110  047117
064540  046440  050124  042052
064546  030115  042052  033106
064554  050052  030523  000063
064562  051111  041503  042040  EM361:  .ASCIZ  /IRCC DM0(MFP+MTP) NOT GOING HIGH ON MFP/
064570  030115  046450  050106
064576  046453  050124  020051
064604  047516  020124  047507
```

```
064612  047111  020107  044510
064620  044107  047440  020116
064626  043115  000120
064632  051107  041501  042440  EM362:  .ASCIZ  /GRAC E24(13) DOESN'T GO HIGH/
064640  032062  030450  024463
064646  042040  042517  047123
064654  052047  043440  020117
064662  044510  044107     000
064667     120  051104  020104  EM363:  .ASCIZ  /PDRD PS15 DOESN'T GET TO GRAC AS A HIGH/
064674  051520  032461  042040
064702  042517  047123  052047
064710  043440  052105  052040
064716  020117  051107  041501
064724  040440  020123  020101
064732  044510  044107     000
064737     123  050123  053440  EM364:  .ASCIZ  /SSP WAS READ BUT NEITHER USP NOR SSP WERE WRITTEN/
064744  051501  051040  040505
064752  020104  052502  020124
064760  042516  052111  042510
064766  020122  051525  020120
064774  047516  020122  051523
065002  020120  042527  042522
065010  053440  044522  052124
065016  047105     000
065021     107  040522  020103  EM365:  .ASCIZ  /GRAC E23(13) DOESN'T GO HIGH/
065026  031105  024063  031461
065034  020051  047504  051505
065042  023516  020124  047507
065050  044040  043511  000110
065056  051107  041501  042440  EM366:  .ASCIZ  /GRAC E23(3) DOESN'T GO HIGH/
065064  031462  031450  020051
065072  047504  051505  023516
065100  020124  047507  044040
065106  043511  000110
065112  042120  042122  050040  EM367:  .ASCII  /PDRD PS14(0) DOESN'T GET TO PDRD E73(13)/<CRLF>
065120  030523  024064  024460
065126  042040  042517  047123
065134  052047  043440  052105
065142  052040  020117  042120
065150  042122  042440  031467
065156  030450  024463     200
065163     101  020123  020101          .ASCIZ  /AS A HIGH OR E73(12) NOT GOING LOW/
065170  044510  044107  047440
065176  020122  033505  024063
065204  031061  020051  047516
065212  020124  047507  047111
065220  020107  047514  000127
065226  020101  051520  020127  EM370:  .ASCII  /A PSW HIGH BYTE BIT(S) DIDN'T SET ON LOAD PS &/<CRLF>
065234  044510  044107  041040
065242  052131  020105  044502
065250  024124  024523  042040
065256  042111  023516  020124
065264  042523  020124  047117
065272  046040  040517  020104
065300  051520  023040     200
065305     113  051105  042516          .ASCIZ  /KERNEL MODE WITH A BR VALUE OF 174000/
```

```
 065312  020114  047515  042504
 065320  053440  052111  020110
 065326  020101  051102  053040
 065334  046101  042525  047440
 065342  020106  033461  030064
 065350  030060     000
 065353     101  041040  052111  EM371:  .ASCII  /A BIT(S) IN THE PSW CLEARED ON AN RTI IN USER MODE/<CRLF>
 065360  051450  020051  047111
 065366  052040  042510  050040
 065374  053523  041440  042514
 065402  051101  042105  047440
 065410  020116  047101  051040
 065416  044524  044440  020116
 065424  051525  051105  046440
 065432  042117  100105
 065436  020046  020101  051102           .ASCIZ  /& A BR VALUE OF 0/
 065444  053040  046101  042525
 065452  047440  020106  000060
 065460  020101  044502  024124  EM372:  .ASCII  /A BIT(S) OF PSW <15,13:11> DIDN'T PRESET ON AN RTI/<CRLF>
 065466  024523  047440  020106
 065474  051520  020127  030474
 065502  026065  031461  030472
 065510  037061  042040  042111
 065516  023516  020124  051120
 065524  051505  052105  047440
 065532  020116  047101  051040
 065540  044524     200
 065543     111  020116  052523           .ASCIZ  /IN SUPER MODE & A BR VALUE OF 134000/
 065550  042520  020122  047515
 065556  042504  023040  040440
 065564  041040  020122  040526
 065572  052514  020105  043117
 065600  030440  032063  030060
 065606  000060
 065610  020101  044502  024124  EM373:  .ASCII  /A BIT(S) IN PSW HIGH BYTE FAILED ON AN IOT IN USER/<CRLF>
 065616  024523  044440  020116
 065624  051520  020127  044510
 065632  044107  041040  052131
 065640  020105  040506  046111
 065646  042105  047440  020116
 065654  047101  044440  052117
 065662  044440  020116  051525
 065670  051105     200
 065673     115  042117  020105           .ASCIZ  /MODE & A BR VALUE OF 0/
 065700  020046  020101  051102
 065706  053040  046101  042525
 065714  047440  020106  000060
 065722  020101  051120  053105  EM374:  .ASCIZ  /A PREVIOUS MODE BIT(S) DIDN'T CLEAR ON AN IOT IN KERNEL/
 065730  047511  051525  046440
 065736  042117  020105  044502
 065744  024124  024523  042040
 065752  042111  023516  020124
 065760  046103  040505  020122
 065766  047117  040440  020116
 065774  047511  020124  047111
 066002  045440  051105  042516
```

```
066010  000114
066012  051523  041522  045440  EM375:  .ASCII  /SSRC KT ABORT FLG DOESN'T GET TO TMCC E34(10) OR/<CRLF>
066020  020124  041101  051117
066026  020124  046106  020107
066034  047504  051505  023516
066042  020124  042507  020124
066050  047524  052040  041515
066056  020103  031505  024064
066064  030061  020051  051117
066072     200
066073     105  032063  030450          .ASCIZ  /E34(10) BAD OR TMCC AERF(1) DOESN'T GO HIGH ON A KT ABORT/
066100  024460  041040  042101
066106  047440  020122  046524
066114  041503  040440  051105
066122  024106  024461  042040
066130  042517  047123  052047
066136  043440  020117  044510
066144  044107  047440  020116
066152  020101  052113  040440
066160  047502  052122     000
066165     120  053523  041040  EM376:  .ASCIZ  /PSW BIT 8 FAILED TO SET/
066172  052111  034040  043040
066200  044501  042514  020104
066206  047524  051440  052105
066214     000
066215     124  041515  020102  EM400:  .ASCII  /TMCB SEGT DOESN'T GO LOW ON A KT ABORT OR/<CRLF>
066222  042523  052107  042040
066230  042517  047123  052047
066236  043440  020117  047514
066244  020127  047117  040440
066252  045440  020124  041101
066260  051117  020124  051117
066266     200
066267     111  020124  047504          .ASCIZ  /IT DOESN'T GET TO DAPE/
066274  051505  023516  020124
066302  042507  020124  047524
066310  042040  050101  000105
066316  040504  042520  052040  EM401:  .ASCIZ  /DAPE TV05*07 DOESN'T GO HIGH ON SEGT/
066324  030126  025065  033460
066332  042040  042517  047123
066340  052047  043440  020117
066346  044510  044107  047440
066354  020116  042523  052107
066362     000
066363     104  050101  020105  EM402:  .ASCIZ  /DAPE TV03 DOESN'T GO HIGH ON SEGT/
066370  053124  031460  042040
066376  042517  047123  052047
066404  043440  020117  044510
066412  044107  047440  020116
066420  042523  052107     000
066425     124  041515  020101  EM403:  .ASCII  /TMCA SEG+CON+PAR DOESN'T GO LOW OR IT DOES/<CRLF>
066432  042523  025507  047503
066440  025516  040520  020122
066446  047504  051505  023516
066454  020124  047507  046040
066462  053517  047440  020122
```

```
066470  052111  042040  042517
066476  100123
066500  047516  020124  042507          .ASCIZ  /NOT GET THRU TMCB E70(2)/
066506  020124  044124  052522
066514  052040  041515  020102
066522  033505  024060  024462
066530     000
066531     124  041515  020101  EM405:  .ASCII  /TMCA SEGTF DOESN'T GET TO E44 OR TMCE PAUSES H/<15><12>
066536  042523  052107  020106
066544  047504  051505  023516
066552  020124  042507  020124
066560  047524  042440  032064
066566  047440  020122  046524
066574  042503  050040  052501
066602  042523  020123  006510
066610     012
066611     104  042517  047123          .ASCIZ  /DOESN'T GET TO E44 OR E44 BAD/
066616  052047  043440  052105
066624  052040  020117  032105
066632  020064  051117  042440
066640  032064  041040  042101
066646     000
066647     124  041515  020103  EM406:  .ASCII  /TMCC NEXM DOESN'T GO LOW OR IT DOESN'T GET THRU E34/<CRLF>
066654  042516  046530  042040
066662  042517  047123  052047
066670  043440  020117  047514
066676  020127  051117  044440
066704  020124  047504  051505
066712  023516  020124  042507
066720  020124  044124  052522
066726  042440  032063     200
066733     117  020122  052111          .ASCIZ  /OR IT DOESN'T GET TO E14 OR E40 BAD/
066740  042040  042517  047123
066746  052047  043440  052105
066754  052040  020117  030505
066762  020064  051117  042440
066770  030064  041040  042101
066776     000
066777     116  054105  020115  EM410:  .ASCII  /NEXM BIT DIDN'T SET IN CPU ERROR REG/<CRLF>
067004  044502  020124  044504
067012  047104  052047  051440
067020  052105  044440  020116
067026  050103  020125  051105
067034  047522  020122  042522
067042  100107
067044  051117  052040  041515          .ASCIZ  /OR TMCC ABORT DOESN'T GO HIGH/
067052  020103  041101  051117
067060  020124  047504  051505
067066  023516  020124  047507
067074  044040  043511  000110
067102  042516  046530  041040  EM411:  .ASCIZ  /NEXM BIT DIDN'T CLEAR IN CPU ERROR REG/
067110  052111  042040  042111
067116  023516  020124  046103
067124  040505  020122  047111
067132  041440  052520  042440
067140  051122  051117  051040
```

```
067146  043505     000
067151     124  041515  020105  EM412:  .ASCIZ  /TMCE KT BEND DOESN'T GO LOW ON TMCC NEXM LOW/
067156  052113  041040  047105
067164  020104  047504  051505
067172  023516  020124  047507
067200  046040  053517  047440
067206  020116  046524  041503
067214  047040  054105  020115
067222  047514  000127
067226  046524  042503  045440  EM413:  .ASCIZ  /TMCE KT BEND DOESN'T GO LOW ON TMCD SL RED/
067234  020124  042502  042116
067242  042040  042517  047123
067250  052047  043440  020117
067256  047514  020127  047117
067264  052040  041515  020104
067272  046123  051040  042105
067300     000
067301     124  041515  020105  EM414:  .ASCIZ  /TMCE KT BEND DOESN'T GO LOW ON TMCC ODD ADRS ERR/
067306  052113  041040  047105
067314  020104  047504  051505
067322  023516  020124  047507
067330  046040  053517  047440
067336  020116  046524  041503
067344  047440  042104  040440
067352  051104  020123  051105
067360  000122
067362  052113  040440  047502  EM415:  .ASCIZ  ?KT ABORT FAILED TO OVER-RIDE NEXM TRAP (KB11-E/EM)?
067370  052122  043040  044501
067376  042514  020104  047524
067404  047440  042526  026522
067412  044522  042504  047040
067420  054105  020115  051124
067426  050101  024040  041113
067434  030461  042455  042457
067442  024515     000
067445     124  041515  020105  EM416:  .ASCIZ  /TMCE CACHE BEND DIDN'T GO HIGH ON KT ABORT/
067452  040503  044103  020105
067460  042502  042116  042040
067466  042111  023516  020124
067474  047507  044040  043511
067502  020110  047117  045440
067510  020124  041101  051117
067516  000124
067520  040504  040520  041040  EM417:  .ASCII  /DAPA BR14 DOESN'T GET TO RACK E49 AS A LOW/<CRLF>
067526  030522  020064  047504
067534  051505  023516  020124
067542  042507  020124  047524
067550  051040  041501  020113
067556  032105  020071  051501
067564  040440  046040  053517
067572     200
067573     117  020122  032105          .ASCIZ  /OR E49(5) BAD/
067600  024071  024465  041040
067606  042101     000
067611     111  046114  043505  EM420:  .ASCIZ  /ILLEGAL HALT BIT IN CPU ERROR REG DIDN'T SET/
067616  046101  044040  046101
```

```
067624  020124  044502  020124
067632  047111  041440  052520
067640  042440  051122  051117
067646  051040  043505  042040
067654  042111  023516  020124
067662  042523  000124
067666  051523  040522  050040   EM421:  .ASCII  /SSRA PS RESTORE(1) DOESN'T GET TO RACK E63/<CRLF>
067674  020123  042522  052123
067702  051117  024105  024461
067710  042040  042517  047123
067716  052047  043440  052105
067724  052040  020117  040522
067732  045503  042440  031466
067740     200
067741     117  020122  033105            .ASCIZ  /OR E63(5) BAD/
067746  024063  024465  041040
067754  042101     000
067757     125  041502  020102   EM422:  .ASCII  /UBCB PE ABORT DOESN'T GO LOW OR/<CRLF>
067764  042520  040440  047502
067772  052122  042040  042517
070000  047123  052047  043440
070006  020117  047514  020127
070014  051117     200
070017     111  020124  047504            .ASCII  /IT DOESN'T GET TO TMCC E33 OR E33 BAD/<CRLF>
070024  051505  023516  020124
070032  042507  020124  047524
070040  052040  041515  020103
070046  031505  020063  051117
070054  042440  031463  041040
070062  042101     200
070065     117  020122  041125            .ASCII  /OR UBCB PARITY ERR DOESN'T GET TO TMCB E53/<CRLF>
070072  041103  050040  051101
070100  052111  020131  051105
070106  020122  047504  051505
070114  023516  020124  042507
070122  020124  047524  052040
070130  041515  020102  032505
070136  100063
070140  051501  040440  046040            .ASCIZ  /AS A LOW OR E53(5) BAD/
070146  053517  047440  020122
070154  032505  024063  024465
070162  041040  042101     000
070167     125  041502  020102   EM423:  .ASCII  /UBCB PARITY ERR DOESN'T GO LOW OR IT DOES/<CRLF>
070174  040520  044522  054524
070202  042440  051122  042040
070210  042517  047123  052047
070216  043440  020117  047514
070224  020127  051117  044440
070232  020124  047504  051505
070240     200
070241     116  052117  043440            .ASCIZ  /NOT GET TO DAPE/
070246  052105  052040  020117
070254  040504  042520     000
070261     104  050101  020105   EM424:  .ASCIZ  /DAPE E11(4) BAD OR TV06 DOESN'T GET TO THE ALU/
070266  030505  024061  024464
070274  041040  042101  047440
```

```
 8676  070302  020122  053124  033060
 8677  070310  042040  042517  047123
 8678  070316  052047  043440  052105
 8679  070324  052040  020117  044124
 8680  070332  020105  046101  000125
 8681  070340  040504  042520  042440  EM425:  .ASCIZ  /DAPE E7(1) BAD/
 8682  070346  024067  024461  041040
 8683  070354  042101     000
 8684  070357     124  041515  020101  EM426:  .ASCIZ  /TMCA SEG+CON+PAR DOESN'T GO LOW ON CCBJ PARITY TRAP/
 8685  070364  042523  025507  047503
 8686  070372  025516  040520  020122
 8687  070400  047504  051505  023516
 8688  070406  020124  047507  046040
 8689  070414  053517  047440  020116
 8690  070422  041503  045102  050040
 8691  070430  051101  052111  020131
 8692  070436  051124  050101     000
 8693  070443     124  041515  020102  EM427:  .ASCII  /TMCB PART DOESN'T GO LOW OR DOES/<CRLF>
 8694  070450  040520  052122  042040
 8695  070456  042517  047123  052047
 8696  070464  043440  020117  047514
 8697  070472  020127  051117  042040
 8698  070500  042517  100123
 8699  070504  047516  020124  042507          .ASCIZ  /NOT GET TO UBCB OR UBCB E18(1) BAD/
 8700  070512  020124  047524  052440
 8701  070520  041502  020102  051117
 8702  070526  052440  041502  020102
 8703  070534  030505  024070  024461
 8704  070542  041040  042101     000
 8705  070547     124  041515  020101  EM430:  .ASCIZ  /TMCA E67(8) DOESN'T GO LOW ON MGMT/
 8706  070554  033105  024067  024470
 8707  070562  042040  042517  047123
 8708  070570  052047  043440  020117
 8709  070576  047514  020127  047117
 8710  070604  046440  046507  000124
 8711  070612  046524  040503  042440  EM431:  .ASCIZ  /TMCA E67(12) DOESN'T GO LOW ON MGMT/
 8712  070620  033466  030450  024462
 8713  070626  042040  042517  047123
 8714  070634  052047  043440  020117
 8715  070642  047514  020127  047117
 8716  070650  046440  046507  000124
 8717  070656  046524  040503  042440  EM432:  .ASCII  /TMCA E68(6) DOESN'T GO LOW ON PAR TRP/<CRLF>
 8718  070664  034066  033050  020051
 8719  070672  047504  051505  023516
 8720  070700  020124  047507  046040
 8721  070706  053517  047440  020116
 8722  070714  040520  020122  051124
 8723  070722  100120
 8724  070724  051117  042440  032464          .ASCIZ  /OR E45(4) BAD/
 8725  070732  032050  020051  040502
 8726  070740  000104
 8727  070742  046524  040503  042440  EM433:  .ASCIZ  /TMCA E68(8) DOESN'T GO LOW ON PAR TRP/
 8728  070750  034066  034050  020051
 8729  070756  047504  051505  023516
 8730  070764  020124  047507  046040
 8731  070772  053517  047440  020116
```

```
071000  040520  020122  051124
071006  000120
071010  046524  041503  050040  EM435:  .ASCII  /TMCC PRIORITY CLEAR DIDN'T GO LOW OR DIDN'T/<CRLF>
071016  044522  051117  052111
071024  020131  046103  040505
071032  020122  044504  047104
071040  052047  043440  020117
071046  047514  020127  051117
071054  042040  042111  023516
071062  100124
071064  042507  020124  044124          .ASCIZ  /GET THRU TMCA E43(2) ON ABORT CLEAR/
071072  052522  052040  041515
071100  020101  032105  024063
071106  024462  047440  020116
071114  041101  051117  020124
071122  046103  040505  000122
071130  052502  020123  041120  EM436:  .ASCIZ  /BUS PB DIDN'T GET TO UBCB PE ABORT/
071136  042040  042111  023516
071144  020124  042507  020124
071152  047524  052440  041502
071160  020102  042520  040440
071166  047502  052122     000
071173     125  041502  020102  EM437:  .ASCIZ  /UBCB PARITY ERR DIDN'T GO LOW ON BUS PB/
071200  040520  044522  054524
071206  042440  051122  042040
071214  042111  023516  020124
071222  047507  046040  053517
071230  047440  020116  052502
071236  020123  041120     000
071243     127  044501  020124  EM440:  .ASCIZ  /WAIT INSTRUCTION DIDN'T WAIT FOR INTERRUPT/
071250  047111  052123  052522
071256  052103  047511  020116
071264  044504  047104  052047
071272  053440  044501  020124
071300  047506  020122  047111
071306  042524  051122  050125
071314  000124
071316  040527  052111  044440  EM441:  .ASCIZ  /WAIT INSTRUCTION FELL THRU ON T BIT TRAP/
071324  051516  051124  041525
071332  044524  047117  043040
071340  046105  020114  044124
071346  052522  047440  020116
071354  020124  044502  020124
071362  051124  050101     000
071367     125  041502  020103  EM442:  .ASCIZ  /UBCC (PWRF+INTR) DOESN'T GO LOW ON TMCB PIRQ/
071374  050050  051127  025506
071402  047111  051124  020051
071410  047504  051505  023516
071416  020124  047507  046040
071424  053517  047440  020116
071432  046524  041103  050040
071440  051111  000121
071444  030523  027063  030060  EM443:  .ASCIZ  /S13.00 FAILED/
071452  043040  044501  042514
071460  000104
071462  032123  027065  030060  EM444:  .ASCIZ  /S45.00 FAILED/
```

```
071470  043040  044501  042514
071476  000104
071500  052115  027120  030061  EM445:  .ASCIZ  /MTP.10 FAILED/
071506  043040  044501  042514
071514  000104
071516  042516  027107  030071  EM446:  .ASCIZ  /NEG.90 FAILED/
071524  043040  044501  042514
071532  000104
071534  032104  027065  030070  EM447:  .ASCIZ  /D45.80 FAILED/
071542  043040  044501  042514
071550  000104
071552  032104  027065  030071  EM450:  .ASCIZ  /D45.90 FAILED/
071560  043040  044501  042514
071566  000104
071570  032104  027065  030060  EM451:  .ASCIZ  /D45.00 FAILED/
071576  043040  044501  042514
071604  000104
071606  032104  027065  030460  EM452:  .ASCIZ  /D45.01 FAILED/
071614  043040  044501  042514
071622  000104
071624  047101  044440  046114  EM453:  .ASCIZ  /AN ILLEGAL INSTRUCTION FAILED TO TRAP/
071632  043505  046101  044440
071640  051516  051124  041525
071646  044524  047117  043040
071654  044501  042514  020104
071662  047524  052040  040522
071670  000120
071672  046524  041103  042440  EM454:  .ASCIZ  /TMCB E53 DOESN'T GO HIGH OR TMCB PIRQ DOESN'T GO LOW/
071700  031465  042040  042517
071706  047123  052047  043440
071714  020117  044510  044107
071722  047440  020122  046524
071730  041103  050040  051111
071736  020121  047504  051505
071744  023516  020124  047507
071752  046040  053517     000
071757     123  051123  020101  EM455:  .ASCII  /SSRA PS RESTORE IS STUCK HIGH OR IT IS NOT/<CRLF>
071764  051520  051040  051505
071772  047524  042522  044440
072000  020123  052123  041525
072006  020113  044510  044107
072014  047440  020122  052111
072022  044440  020123  047516
072030  100124
072032  042507  052124  047111          .ASCIZ  /GETTING THRU RACK E63(B0) AS A LOW/
072040  020107  044124  052522
072046  051040  041501  020113
072054  033105  024063  030102
072062  020051  051501  040440
072070  046040  053517     000
072075     124  041515  020103  EM456:  .ASCIZ  /TMCC E5(11) DOESN'T GO HIGH ON DATI OR DATO/
072102  032505  030450  024461
072110  042040  042517  047123
072116  052047  043440  020117
072124  044510  044107  047440
072132  020116  040504  044524
```

```
072140  047440  020122  040504
072146  047524     000
072151     120  053523  041040  EM457:  .ASCIZ  /PSW BIT 11 DOESN'T CLEAR/
072156  052111  030440  020061
072164  047504  051505  023516
072172  020124  046103  040505
072200  000122
072202  051520  020127  044103  EM460:  .ASCIZ  /PSW CHANGED ON RESET IN KERNEL MODE/
072210  047101  042507  020104
072216  047117  051040  051505
072224  052105  044440  020116
072232  042513  047122  046105
072240  046440  042117  000105
072246  051520  020127  044103  EM461:  .ASCIZ  /PSW CHANGES ON RESET IN SUPER MODE/
072254  047101  042507  020123
072262  047117  051040  051505
072270  052105  044440  020116
072276  052523  042520  020122
072304  047515  042504     000
072311     115  046505  046440  EM703:  .ASCIZ  /MEM MGT TESTS DISABLED/<CRLF>
072316  052107  052040  051505
072324  051524  042040  051511
072332  041101  042514  100104
072340     000
072341     103  041501  042510  EM704:  .ASCIZ  /CACHE TESTS DISABLED/<CRLF>
072346  052040  051505  051524
072354  042040  051511  041101
072362  042514  100104     000
072367     125  044516  052502  EM706:  .ASCIZ  /UNIBUS PE TEST DISABLED/<CRLF>
072374  020123  042520  052040
072402  051505  020124  044504
072410  040523  046102  042105
072416  000200
072420  047516  041040  020122  EM710:  .ASCIZ  /NO BR 5 DEVICE/<CRLF>
072426  020065  042504  044526
072434  042503  000200
072440  047516  041040  020122  EM711:  .ASCIZ  /NO BR 6 DEVICE/<CRLF>
072446  020066  042504  044526
072454  042503  000200
072460  051102  032040  052040  EM712:  .ASCIZ  /BR 4 TESTS DISABLED/<CRLF>
072466  051505  051524  042040
072474  051511  041101  042514
072502  100104     000
072505     102  020122  020065  EM713:  .ASCIZ  /BR 5 TESTS DISABLED/<CRLF>
072512  042524  052123  020123
072520  044504  040523  046102
072526  042105  000200
072532  051102  033040  052040  EM715:  .ASCIZ  /BR 6 TESTS DISABLED/<CRLF>
072540  051505  051524  042040
072546  051511  041101  042514
072554  100104     000
072557     117  051120  052040  EM717:  .ASCIZ  /OPR TEST DISABLED/<CRLF>
072564  051505  020124  044504
072572  040523  046102  042105
072600  000200
072602  047514  045517  040440  EM720:  .ASCII  /LOOK AT THE CONSOLE LIGHTS/<CRLF>
```

```
072610  020124  044124  020105
072616  047503  051516  046117
072624  020105  044514  044107
072632  051524     200
072635     124  042510  042040            .ASCII  /THE DATA LIGHTS SHOULD READ 166667/<CRLF>
072642  052101  020101  044514
072650  044107  051524  051440
072656  047510  046125  020104
072664  042522  042101  030440
072672  033066  033066  100067
072700  044124  020105  042101            .ASCIZ  /THE ADDRESS LIGHTS SHOULD READ  /
072706  051104  051505  020123
072714  044514  044107  051524
072722  051440  047510  046125
072730  020104  042522  042101
072736  020040     000
072741     200  044103  047101  EM721:    .ASCIZ  <CRLF>/CHANGE SWITCH 7 TO CONTINUE/<CRLF>
072746  042507  051440  044527
072754  041524  020110  020067
072762  047524  041440  047117
072770  044524  052516  100105
072776     000
072777     200  054524  042520  EM722:    .ASCIZ  <CRLF>/TYPE A CHARACTER TO CONTINUE/<CRLF>
073004  040440  041440  040510
073012  040522  052103  051105
073020  052040  020117  047503
073026  052116  047111  042525
073034  000200
073036  047200  020117  040515  EM724:    .ASCIZ  <CRLF>/NO MAP REGISTERS AVAILABLE FOR TEST 75/<CRLF>
073044  020120  042522  044507
073052  052123  051105  020123
073060  053101  044501  040514
073066  046102  020105  047506
073074  020122  042524  052123
073102  033440  100065     000
073107     116  020117  020114  EM725:    .ASCIZ  /NO L CLOCK/<CRLF>
073114  046103  041517  100113
073122     000
        073124                            .EVEN
073124  073124                  ADRTAB:   .WORD   .
073126  005476                            TST1
073130  005660                            TST2
073132  006154                            TST3
073134  006436                            TST4
073136  007200                            TST5
073140  007314                            TST6
073142  007374                            TST7
073144  007460                            TST10
073146  010302                            TST11
073150  010442                            TST12
073152  011004                            TST13
073154  011120                            TST14
073156  012600                            TST15
073160  012746                            TST16
073162  013120                            TST17
073164  013306                            TST20
```

```
 8956  073166  013446                  TST21
 8957  073170  013562                  TST22
 8958  073172  013766                  TST23
 8959  073174  014336                  TST24
 8960  073176  014474                  TST25
 8961  073200  014632                  TST26
 8962  073202  015002                  TST27
 8963  073204  015156                  TST30
 8964  073206  015356                  TST31
 8965  073210  015530                  TST32
 8966  073212  016650                  TST33
 8967  073214  017144                  TST34
 8968  073216  017274                  TST35
 8969  073220  017424                  TST36
 8970  073222  020016                  TST37
 8971  073224  020432                  TST40
 8972  073226  021166                  TST41
 8973  073230  021432                  TST42
 8974  073232  022042                  TST43
 8975  073234  022476                  TST44
 8976  073236  023234                  TST45
 8977  073240  023504                  TST46
 8978  073242  023556                  TST47
 8979  073244  026366                  TST50
 8980  073246  027016                  TST51
 8981  073250  027232                  TST52
 8982  073252  027402                  TST53
 8983  073254  027652                  TST54
 8984  073256  030016                  TST55
 8985  073260  030252                  TST56
 8986  073262  030400                  TST57
 8987  073264  030610                  TST60
 8988  073266  030666                  TST61
 8989  073270  031432                  TST62
 8990  073272  031546                  TST63
 8991  073274  032042                  TST64
 8992  073276  032354         SUBTAB:  $EOP
 8993          000001                  .END
```

```
ADRTAB  073124          5912     5922     8939#
BINARY  034574          5842*    5845*    5847     5853#
BIT0  = 000001           149#    1974     2004     2406     5932
BIT00 = 000001           139#     149
BIT01 = 000002           138#     148
BIT02 = 000004           137#     147
BIT03 = 000010           136#     146
BIT04 = 000020           135#     145
BIT05 = 000040           134#     144
BIT06 = 000100           133#     143
BIT07 = 000200           132#     142
BIT08 = 000400           131#     141      5621
BIT09 = 001000           130#     140      5629     5695
BIT1  = 000002           148#    3590     3601     3603     3866
BIT10 = 002000           129#    2925     2969     4342     5678
BIT11 = 004000           128#    2958     3003     4358     4723     4724     4726     4731     4740     4742     4754     4756     4768
                        4779     4785     4789     4795     4797     4799     4806     4809     4823     4860     4862     5636
BIT12 = 010000           127#    2873     2991     3038     4376     4473     4504     4521     5012     5083     5514
BIT13 = 020000           126#    3025     3080     4391     4541     4589     4606     5541     5569     5685
BIT14 = 040000           125#    2881     3062     3115     4433     4556     4624     4654     4914     4916     4925     4928     4936
                        4948     4971     5031     5036     5042     5078     5101     5147     5149     5287
BIT15 = 100000           124#    1710     1711     1770     1778     1821     1836     1846     1866     1869     1895     2002     2031
                        2045     2109     2180     2227     2488     2495     2615     2623     2625     2653     2701     2738     2890
                        3065     3102     3179     4448     4571     4639     4669     4938     4940     4950     4953     4986     4993
                        5105     5116     5128     5351
BIT2  = 000004           147#    3725     3728     3860     3879
BIT3  = 000010           146#    3527     3530     3830     3851     3869
BIT4  = 000020           145#    1666     1701     1760     2807     2855     3137     3148     3173     3189     3192     3338     3367
                        3403     3436     3470     3548     3612     3649     4021     4189     4277     4411     4467     4498     5073
                        5266
BIT5  = 000040           144#
BIT6  = 000100           143#    1559     1652     1658     1659     1677     1679     1697     3292     3619     3622     3625     3954
                        3968     4007     4691     4697     5321     5335     5448     5453     5897
BIT7  = 000200           142#    5293     5874
BIT8  = 000400           141#    4212
BIT9  = 001000           140#    2862     2866     2937     3592     3883     4310     4326
BPTVEC= 000014           156#    2771*
BUFFDP  034200          5777#    5783
CACHSP  033060          1562     5563#
CACHVE= 000114           163#    1562*    1563*
CHAINQ  034702          5503     5882#
CONTRL= 177746           173#    1495*    1568*
CPUERR= 177766           186#    1564*    2905*    3148     3189     3191     3194*    3195     3254*    3264*    3268*    3527     3529
                        3533*    3534     3725     3727     3731*    3732     3860     3869     3895*    3947*    3954     3957*    3968
                        4007     4010*    4681*    5293     5295     5299*    5300     5547     5560*
CPUSPU  032654          1560     1664     2784     2897     2903     2930     2963     2992     2996     3000     3027     3031     3034
                        3063     3072     3077     3104     3108     3111     3187     3253     3265     3267     3495     3504     3516
                        3662     3666     3672     3686     3697     3710     3720     3724     3871     3909     3938     3942     3963
                        3967     3983     4000     4006     4331     4347     4363     4526     4533     4611     4680     5303     5535#
CR    = 000015            61#    6026     6036
CRLF  = 000200            62#    1541     5999     6036     6252     6258     6266     6277     6356     6372     6407     6438     6486
                        6529     6545     6561     6593     6620     6679     6772     6839     6996     7020     7040     7065     7089
                        7103     7110     7118     7258     7284     7302     7332     7341     7350     7371     7383     7396     7414
                        7425     7437     7450     7474     7496     7507     7521     7539     7557     7607     7647     7655     7669
                        7673     7689     7730     7737     7746     7753     7760     7767     7790     7802     7834     7842     7851
                        7860     7898     7914     7948     7985     8007     8020     8035     8052     8080     8120     8137     8150
```

```
                                                                  J 16
PDP 11/70-74MP CPU DIAGNOSTIC PART 2      MACY11 30A(1052)  17-SEP-79  10:53  PAGE 169
CEKBBD.P11     17-SEP-79 10:22            CROSS REFERENCE TABLE -- USER SYMBOLS                                          SEQ 0204

                        8309     8320     8374     8387     8402     8414     8430     8453     8477     8502     8530     8546     8607
                        8626     8637     8643     8650     8662     8693     8717     8734     8824     8863     8868     8872     8877
                        8880     8883     8887     8891     8895     8899     8904     8916     8922     8928     8935
DH1      036614          570      574      577     6277#
DH14     037445          604      869     6356#
DH15     037570          608     6372#
DH163    046643          920     7040#
DH201    050145          954      963      972      981      993     1002     7168#
DH243    053373         1065     1069     7474#
DH25     040345          632      752     6438#
DH250    054124         1081     7539#
DH255    054714         1096     1102     1108     1370     1394     7607#
DH263    055451         1114     5755     7669#
DH263A   055476         5771     7673#
DH263B   055662         7694#    7719
DH263C   055706         7698#    7720
DH263D   055734         7702#    7721
DH263E   055763         7706#    7722
DH263F   056011         7710#    7723
DH263G   056034         7714#    7724
DH337    063213         1246     1249     1279     1282     8201#
DH357    064431         1294     8320#
DH4      037063          580      583      586      589      592      595      598      601      611      614      617      620      623
                         626      629      636      643      649      652      655      658      661      664      667      710      737
                         740      746      758      767      770      776      779      785      791      794      797      800      803
                         815      821      824      857      863      866      872      875      878      881      887      890      896
                         899      902      905      908      911      914      917      924      927      930      933      936      939
                         942      948      951      957      960      966      969      975      978      984      987      990      996
                         999     1005     1008     1011     1014     1017     1020     1023     1026     1029     1032     1035     1038
                        1041     1044     1047     1050     1053     1056     1059     1062     1072     1075     1078     1084     1087
                        1090     1093     1099     1105     1111     1120     1123     1126     1129     1132     1135     1138     1141
                        1144     1147     1150     1153     1156     1159     1162     1165     1168     1171     1174     1177     1180
                        1183     1186     1189     1192     1195     1198     1201     1204     1207     1210     1213     1216     1219
                        1222     1225     1228     1231     1234     1237     1240     1243     1252     1255     1267     1270     1273
                        1276     1285     1288     1291     1298     1301     1304     1307     1310     1313     1316     1337     1340
                        1346     1349     1352     1355     1358     1361     1364     1367     1373     1376     1379     1382     1385
                        1388     1391     1397     1400     1403     1406     1409     1412     1415     1418     1421     1424     1427
                        1430     1433     1436     1439     1442     1445     1448     1451     1454     1457     1460     1463     1466
                        1469     1472     1475     1478     1481     1484     1487     6308#
DH41     041335          670      854     6529#
DH42     041456          639      646      674      685      691      725      734      749      755      761      764      812      818
                         827      851      860      884      893     1258     1261     1264     1319     1322     1325     1328     1331
                        1334     1490     1493     6545#
DH43     041600          678      682      688      694      707      713      716      719      722      728      731      743      773
                         782      788      806      809      830      833      836      839      842      845      848     6561#
DH46     042054          945     6593#
DH51     042267          697      701      704     6620#
DISPLA=  177570           56#    5650*    5674*
DOMFPT   032064         5381     5383#
DONE7    032114         5382     5388     5392#
DT1      036734          572      575      578     6292#
DT14     037526          606      609      870     1296     6366#
DT163    046722          922     7049#
DT201    050176          955      964      973      982      994     1003     7174#
DT243    053456         1067     7484#
DT244    053540         1070     7494#
```

```
DT25     040420         634     753    6447#
DT250    054242        1082    7554#
DT337    063250        1247    1250    1280    1283    8207#
DT4      037110         581     584     587     590     593     596     599     602     612     615     618     621     624
                        627     630     637     644     650     653     656     659     662     665     668     711     738
                        741     747     759     768     771     777     780     786     792     795     798     801     804
                        816     822     825     858     864     867     873     876     879     882     888     891     897
                        900     903     906     909     912     915     918     925     928     931     934     937     940
                        943     949     952     958     961     967     970     976     979     985     988     991     997
                       1000    1006    1009    1012    1015    1018    1021    1024    1027    1030    1033    1036    1039
                       1042    1045    1048    1051    1054    1057    1060    1063    1073    1076    1079    1085    1088
                       1091    1094    1100    1106    1112    1121    1124    1127    1130    1133    1136    1139    1142
                       1145    1148    1151    1154    1157    1160    1163    1166    1169    1172    1175    1178    1181
                       1184    1187    1190    1193    1196    1199    1202    1205    1208    1211    1214    1217    1220
                       1223    1226    1229    1232    1235    1238    1241    1244    1253    1256    1268    1271    1274
                       1277    1286    1289    1292    1299    1302    1305    1308    1311    1314    1317    1338    1341
                       1347    1350    1353    1356    1359    1362    1365    1368    1374    1377    1380    1383    1386
                       1389    1392    1398    1401    1404    1407    1410    1413    1416    1419    1422    1425    1428
                       1431    1434    1437    1440    1443    1446    1449    1452    1455    1458    1461    1464    1467
                       1470    1473    1476    1479    1482    1485    1488    6313#
DT41     041414         672     855    1097    1103    1109    1371    1395    6539#
DT42     041534         641     647     676     686     692     726     735     750     756     762     765     813     819
                        828     852     861     885     894    1259    1262    1265    1320    1323    1326    1329    1332
                       1335    1491    1494    6554#
DT43     041710         680     683     689     695     708     714     717     720     723     729     732     744     774
                        783     789     807     810     831     834     837     840     843     846     849    6575#
DT46     042136         946    6603#
DT51     042426         699     702     705    6638#
DUMMY    034766        5876    5896#
EMTVEC=  000030         159#   1508*   1509*   3960*   3979*   3984*   3993*
EM1      036546         569    6270#
EM10     037253         591    6331#
EM100    043727         766    6767#
EM101    043757         769    6772#
EM102    044103         772     808     832     841     847    6787#
EM103    044155         775    6795#
EM104    044204         778    6799#
EM105    044274         781     787    6809#
EM106    044325         784    6814#
EM11     037267         594    6334#
EM110    044415         790    6824#
EM111    044505         793    6834#
EM112    044535         796    6839#
EM113    044655         799    6853#
EM114    044704         802    6857#
EM115    044733         805    6861#
EM117    045001         811    6868#
EM12     037326         597    6340#
EM120    045056         814    6876#
EM121    045145         817    6886#
EM122    045215         820    6893#
EM123    045255         823    6899#
EM124    045304         826     859    6903#
EM125    045336         829    6908#
EM127    045411         835    6916#
EM13     037365         600    6346#
```

```
EM130   045463        838    844    6924#
EM134   045524        850   6930#
EM135   045573        853   6937#
EM136   045602        856   6939#
EM14    037415        603   6351#
EM140   045635        862    886    6944#
EM141   045672        865   6949#
EM142   045731        868   6955#
EM143   045757        871   6959#
EM144   046013        874   6964#
EM145   046042        877   6968#
EM146   046101        880   6974#
EM147   046140        883   6980#
EM15    037540        607   6368#
EM151   046171        889   6985#
EM152   046204        892   6987#
EM155   046233        901   6991#
EM156   046264        904   6996#
EM157   046404        907   7010#
EM16    037651        610    895    6381#
EM160   046435        910   7015#
EM161   046466        913   7020#
EM162   046563        916   7031#
EM163   046614        919   7036#
EM164   046734        923   7051#
EM165   046765        926   7056#
EM166   047047        929   7065#
EM167   047154        932   7077#
EM17    037705        613   6386#
EM170   047216        935   7083#
EM171   047260        938   7089#
EM172   047401        941   7103#
EM174   047654        947   7134#
EM175   047736        950    959     968     977     998    7143#
EM176   047753        953   7146#
EM177   050015        956   7152#
EM2     036752        573   6295#
EM20    037762        616   6394#
EM201   050077        962   7161#
EM202   050206        965   7176#
EM204   050270        971   7185#
EM205   050336        974   7192#
EM207   050421        980   7201#
EM21    040053        619   6404#
EM210   050467        983   7208#
EM211   050535        986   7215#
EM212   050616        989   7224#
EM213   050662        992   7230#
EM214   050730        995   7237#
EM216   051012       1001   7246#
EM217   051060       1004   7253#
EM22    040073        622   6407#
EM220   051114       1007   7258#
EM221   051175       1010   7267#
EM222   051231       1013   7272#
EM223   051311       1016   7281#
```

```
EM224   051332            1019     7284#
EM225   051501            1022     7302#
EM226   051612            1025     7315#
EM227   051672            1028     7323#
EM23    040170             625     6418#
EM230   051753            1031     7332#
EM232   052240            1037     7364#
EM233   052312            1040     7371#
EM234   052407            1043     7383#
EM235   052514            1046     7396#
EM237   052661            1052     7414#
EM24    040260             628     6428#
EM240   052754            1055     7425#
EM241   053056            1058     7437#
EM242   053167            1061     7450#
EM243   053323            1064     7467#
EM244   053470            1068     7486#
EM245   053552            1071     7496#
EM246   053645            1074     7507#
EM247   053766            1077     7521#
EM25    040313             631      687     6433#
EM250   054077            1080     7535#
EM251   054260            1083     7557#
EM252   054425            1086     7575#
EM253   054475            1089     7582#
EM254   054545            1092     7589#
EM255   054635            1095     7599#
EM256   054775            1098     7616#
EM257   055032            1101     7621#
EM26    040432             635     6449#
EM260   055103            1104     7628#
EM261   055146            1107     7634#
EM262   055220            1110     7641#
EM263   055262            1113     5753     7647#
EM264   056074            1116     7725#
EM265   056117            1119     7730#
EM266   056435            1122     7767#
EM267   056553            1125     7781#
EM27    040472             638     6455#
EM270   056633            1128     7790#
EM271   056731            1131     7802#
EM272   057061            1134     7818#
EM273   057120            1137     7824#
EM274   057166            1140     7831#
EM275   057206            1143     7834#
EM276   057556            1146     7875#
EM277   057607            1149     7880#
EM3     036773             576     6298#
EM30    040534             642     6461#
EM300   057663            1152     7888#
EM301   057752            1155     7898#
EM302   060037            1158     7907#
EM303   060104            1161     7914#
EM304   060235            1164     7930#
EM305   060322            1167     7939#
EM306   060410            1170     7948#
```

```
                                                                  B  1
PDP 11/70-74MP CPU DIAGNOSTIC PART 2      MACY11 30A(1052)  17-SEP-79  10:53  PAGE 173
CEKBBD.P11     17-SEP-79 10:22            CROSS REFERENCE TABLE -- USER SYMBOLS                            SEQ 0208

EM307     060532             1173      7962#
EM31      040572              645      6466#
EM310     060605             1176      7970#
EM311     060657             1179      7978#
EM312     060730             1182      7985#
EM313     061054             1185      8000#
EM314     061124             1188      8007#
EM315     061235             1191      8020#
EM316     061357             1194      8035#
EM317     061472             1197      8049#
EM32      040634              648      6472#
EM320     061512             1200      8052#
EM321     061630             1203      8065#
EM322     061700             1206      8072#
EM323     061755             1209      8080#
EM324     062065             1212      8094#
EM325     062146             1215      8103#
EM326     062223             1218      8111#
EM327     062304             1221      8120#
EM33      040700              651      6478#
EM330     062447             1224      8137#
EM331     062560             1227      8150#
EM332     062673             1230      8163#
EM333     062754             1233      8172#
EM334     063007             1236      8177#
EM335     063042             1239      8182#
EM336     063077             1242      8187#
EM337     063146             1245      8194#
EM34      040716              654      6481#
EM340     063260             1248      8209#
EM341     063332             1251      8216#
EM342     063377             1254      8223#
EM343     063444             1257      8230#
EM344     063500             1260      8235#
EM345     063535             1263      8240#
EM346     063570             1266      8245#
EM347     063650             1269      8253#
EM35      040751              657      6486#
EM350     063730             1272      8261#
EM351     064005             1275      8269#
EM352     064062             1278      8277#
EM353     064127             1281      8284#
EM354     064174             1284      8291#
EM355     064226             1287      8296#
EM356     064305             1290      8304#
EM357     064340             1293      8309#
EM36      041067              660      6500#
EM360     064503             1297      8328#
EM361     064562             1300      8336#
EM362     064632             1303      8343#
EM363     064667             1306      8348#
EM364     064737             1309      8355#
EM365     065021             1312      8364#
EM366     065056             1315      8369#
EM367     065112             1318      8374#
EM37      041152              663      6509#
```

```
EM370   065226          1321    8387#
EM371   065353          1324    8402#
EM372   065460          1327    8414#
EM373   065610          1330    8430#
EM374   065722          1333    8443#
EM375   066012          1336    8453#
EM376   066165          1339    8472#
EM4     037020           579    6302#
EM40    041222           666    6516#
EM400   066215          1345    8477#
EM401   066316          1348    8489#
EM402   066363          1351    8496#
EM403   066425          1354    8502#
EM405   066531          1360    8515#
EM406   066647          1363    8530#
EM41    041272           669    6523#
EM410   066777          1369    8546#
EM411   067102          1372    8558#
EM412   067151          1375    8565#
EM413   067226          1378    8573#
EM414   067301          1381    8581#
EM415   067362          1384    8590#
EM416   067445          1387    8599#
EM417   067520          1390    8607#
EM42    041426           673    6541#
EM420   067611          1393    8618#
EM421   067666          1396    8626#
EM422   067757          1399    8637#
EM423   070167          1402    8662#
EM424   070261          1405    8673#
EM425   070340          1408    8681#
EM426   070357          1411    8684#
EM427   070443          1414    8693#
EM43    041546           677    6556#
EM430   070547          1417    8705#
EM431   070612          1420    8711#
EM432   070656          1423    8717#
EM433   070742          1426    8727#
EM435   071010          1432    8734#
EM436   071130          1435    8748#
EM437   071173          1438    8754#
EM44    041726           681    6578#
EM440   071243          1441    8761#
EM441   071316          1444    8769#
EM442   071367          1447    8776#
EM443   071444          1450    8784#
EM444   071462          1453    8787#
EM445   071500          1456    8790#
EM446   071516          1459    8793#
EM447   071534          1462    8796#
EM45    041772           684    6584#
EM450   071552          1465    8799#
EM451   071570          1468    8802#
EM452   071606          1471    8805#
EM453   071624          1474    8808#
EM454   071672          1477    8815#
```

```
                                                                      D  1
PDP 11/70-74MP CPU DIAGNOSTIC PART 2     MACY11 30A(1052)  17-SEP-79  10:53  PAGE 175
CEKBBD.P11     17-SEP-79 10:22           CROSS REFERENCE TABLE -- USER SYMBOLS                                    SEQ 0210

EM455    071757           1480     8824#
EM456    072075           1483     8838#
EM457    072151           1486     8846#
EM46     042021            944     6588#
EM460    072202           1489     8851#
EM461    072246           1492     8857#
EM47     042150            690     6605#
EM5      037116            582     6314#
EM50     042175            693     6609#
EM51     042227            696     6614#
EM52     042450            700     6641#
EM53     042523            703     6649#
EM54     042556            706     6654#
EM55     042574            709      745      898     6657#
EM56     042652            712     6665#
EM57     042723            715     6672#
EM6      037204            585     6323#
EM60     042773            718     6679#
EM61     043074            721     6691#
EM62     043152            724     6699#
EM63     043214            727     6705#
EM64     043254            730     6711#
EM65     043314            733     6717#
EM66     043351            736     6722#
EM67     043402            739     6727#
EM7      037237            588     6328#
EM70     043434            742     6732#
EM703    072311           8863#
EM704    072341           8868#
EM706    072367           8872#
EM710    072420           3245     8877#
EM711    072440           3271     8880#
EM712    072460           3333     8883#
EM713    072505           3395     8887#
EM715    072532           3428     8891#
EM717    072557           5413     8895#
EM72     043452            748     6735#
EM720    072602           5418     5440     8899#
EM721    072741           5425     5434     8916#
EM722    072777           5445     8922#
EM724    073036           8928#
EM725    073107           3256     8935#
EM73     043514            751     6741#
EM74     043532            754     6744#
EM75     043574            757     6750#
EM76     043661            760     6759#
EM77     043704            763     6763#
ENDBR5   016520           3277     3284#
ENDGET   026362           4704     4706     4710#
ENDKB    023202           4140     4143     4145#
ERRVEC=  000004            152#    1560*    1561*    1644*    1664*    2773*    2784*    2897*    2903*    2930*    2963*    2992*    2996*
                          3000*    3027*    3031*    3034*    3063*    3072*    3077*    3104*    3108*    3111*    3136*    3139*    3141*
                          3160*    3171*    3177*    3187*    3219*    3222*    3225*    3228*    3248*    3253*    3259*    3265*    3267*
                          3475*    3476*    3490*    3495*    3499*    3504*    3510*    3516*    3523*    3654*    3657*    3662*    3666*
                          3672*    3682*    3686*    3691*    3697*    3703*    3710*    3714*    3720*    3724*    3833*    3909*    3931*
                          3936*    3938*    3942*    3950*    3959*    3963*    3967*    3978*    3983*    3996*    4000*    4006*    4281*
```

```
                      4287*   4322*   4331*   4339*   4347*   4355*   4363*   4519*   4526*   4533*   4604*   4611*   4652*
                      4667*   4680*   5286*   5303*   5612    5613*   5615*   5618*
ETYPDM   033722       5687    5709#
E1STKL   001220        525#   3760*   3778    3781    3784    7494    7554
E2STKL   001222        526#   3767*   7484    7554
FIVESP   034635       5860#
FOURSP   034636       5821    5840    5861#
GNS    = ****** U      441    1540    5481    5488    5539    5545    5567    5573    5834    5880    5893    5900    5951
                      5957    6207    6208    6209    6210    6211
HIADRS= 177742         171#
HITMIS= 177752         175#
HT     = 000011         59#   5997    6036
INDEX    056060       5793    7719#
INTER5   001224        527#   3241*   3398    3409    4295
INTER6   001232        530#   3250*   3261*   3363    3373    3431    3442    4490
INT5ST   001230        529#   3230*   3233*   3236*   3239*   3279*   3282*   3284*   4687*   4697*
INT5SU   016464       3241    3276#
INT5VE   001226        528#   3231*   3234*   3237*   3240*   3277*   3401    4409    4465    4538*   4553*   4568*   4696*
INT6ST   001236        532#   3252*   3263*   3292*   3295*   3298*   3306*   3309*   3312*   4509*   4515*   4595*   4601*
                      4629*   4644*   4664*   4691*   4694*
INT6VE   001234        531#   3251*   3262*   3290*   3303*   3365    3434    4496    4587*   4621*   4636*   4689
IOTVEC= 000020         157#   1506*   1507*   5233*   5234*   5246*   5247*   5265*
IRE345   004024       1263#
ITEM1    001274        569#
ITEM10   001346        591#
ITEM11   001354        594#
ITEM12   001362        597#
ITEM13   001370        600#
ITEM14   001376        603#
ITEM15   001404        607#
ITEM16   001412        610#
ITEM17   001420        613#
ITEM2    001302        573#
ITEM20   001426        616#
ITEM21   001434        619#
ITEM22   001442        622#
ITEM23   001450        625#
ITEM24   001456        628#
ITEM25   001464        631#
ITEM26   001472        635#
ITEM27   001500        638#
ITEM3    001310        576#
ITEM30   001506        642#
ITEM31   001514        645#
ITEM32   001522        648#
ITEM33   001530        651#
ITEM34   001536        654#
ITEM35   001544        657#
ITEM36   001552        660#
ITEM37   001560        663#
ITEM4    001316        579#
ITEM40   001566        666#
ITEM41   001574        669#
ITEM42   001602        673#
ITEM43   001610        677#
ITEM44   001616        681#
```

```
ITEM45  001624        684#
ITEM46  001632        687#
ITEM47  001640        690#
ITEM5   001324        582#
ITEM50  001646        693#
ITEM51  001654        696#
ITEM52  001662        700#
ITEM53  001670        703#
ITEM54  001676        706#
ITEM55  001704        709#
ITEM56  001712        712#
ITEM57  001720        715#
ITEM6   001332        585#
ITEM60  001726        718#
ITEM61  001734        721#
ITEM62  001742        724#
ITEM63  001750        727#
ITEM64  001756        730#
ITEM65  001764        733#
ITEM66  001772        736#
ITEM67  002000        739#
ITEM7   001340        588#
ITEM70  002006        742#
ITEM71  002014        745#
ITEM72  002022        748#
ITEM73  002030        751#
ITEM74  002036        754#
ITEM75  002044        757#
ITEM76  002052        760#
ITEM77  002060        763#
ITE100  002066        766#
ITE101  002074        769#
ITE102  002102        772#
ITE103  002110        775#
ITE104  002116        778#
ITE105  002124        781#
ITE106  002132        784#
ITE107  002140        787#
ITE110  002146        790#
ITE111  002154        793#
ITE112  002162        796#
ITE113  002170        799#
ITE114  002176        802#
ITE115  002204        805#
ITE116  002212        808#
ITE117  002220        811#
ITE120  002226        814#
ITE121  002234        817#
ITE122  002242        820#
ITE123  002250        823#
ITE124  002256        826#
ITE125  002264        829#
ITE126  002272        832#
ITE127  002300        835#
ITE130  002306        838#
ITE131  002314        841#
```

```
                                                                 G 1
PDP 11/70-74MP CPU DIAGNOSTIC PART 2     MACY11 30A(1052)  17-SEP-79  10:53  PAGE 178
CEKBBD.P11     17-SEP-79 10:22           CROSS REFERENCE TABLE -- USER SYMBOLS                                    SEQ 0213

ITE132  002322            844#
ITE133  002330            847#
ITE134  002336            850#
ITE135  002344            853#
ITE136  002352            856#
ITE137  002360            859#
ITE140  002366            862#
ITE141  002374            865#
ITE142  002402            868#
ITE143  002410            871#
ITE144  002416            874#
ITE145  002424            877#
ITE146  002432            880#
ITE147  002440            883#
ITE150  002446            886#
ITE151  002454            889#
ITE152  002462            892#
ITE153  002470            895#
ITE154  002476            898#
ITE155  002504            901#
ITE156  002512            904#
ITE157  002520            907#
ITE160  002526            910#
ITE161  002534            913#
ITE162  002542            916#
ITE163  002550            919#
ITE164  002556            923#
ITE165  002564            926#
ITE166  002572            929#
ITE167  002600            932#
ITE170  002606            935#
ITE171  002614            938#
ITE172  002622            941#
ITE173  002630            944#
ITE174  002636            947#
ITE175  002644            950#
ITE176  002652            953#
ITE177  002660            956#
ITE200  002666            959#
ITE201  002674            962#
ITE202  002702            965#
ITE203  002710            968#
ITE204  002716            971#
ITE205  002724            974#
ITE206  002732            977#
ITE207  002740            980#
ITE210  002746            983#
ITE211  002754            986#
ITE212  002762            989#
ITE213  002770            992#
ITE214  002776            995#
ITE215  003004            998#
ITE216  003012           1001#
ITE217  003020           1004#
ITE220  003026           1007#
ITE221  003034           1010#
```

```
ITE222  003042          1013#
ITE223  003050          1016#
ITE224  003056          1019#
ITE225  003064          1022#
ITE226  003072          1025#
ITE227  003100          1028#
ITE230  003106          1031#
ITE231  003114          1034#
ITE232  003122          1037#
ITE233  003130          1040#
ITE234  003136          1043#
ITE235  003144          1046#
ITE236  003152          1049#
ITE237  003160          1052#
ITE240  003166          1055#
ITE241  003174          1058#
ITE242  003202          1061#
ITE243  003210          1064#
ITE244  003216          1068#
ITE245  003224          1071#
ITE246  003232          1074#
ITE247  003240          1077#
ITE250  003246          1080#
ITE251  003254          1083#
ITE252  003262          1086#
ITE253  003270          1089#
ITE254  003276          1092#
ITE255  003304          1095#
ITE256  003312          1098#
ITE257  003320          1101#
ITE260  003326          1104#
ITE261  003334          1107#
ITE262  003342          1110#
ITE263  003350          1113#
ITE264  003356          1116#
ITE265  003364          1119#
ITE266  003372          1122#
ITE267  003400          1125#
ITE270  003406          1128#
ITE271  003414          1131#
ITE272  003422          1134#
ITE273  003430          1137#
ITE274  003436          1140#
ITE275  003444          1143#
ITE276  003452          1146#
ITE277  003460          1149#
ITE300  003466          1152#
ITE301  003474          1155#
ITE302  003502          1158#
ITE303  003510          1161#
ITE304  003516          1164#
ITE305  003524          1167#
ITE306  003532          1170#
ITE307  003540          1173#
ITE310  003546          1176#
ITE311  003554          1179#
```

```
ITE312   003562          1182#
ITE313   003570          1185#
ITE314   003576          1188#
ITE315   003604          1191#
ITE316   003612          1194#
ITE317   003620          1197#
ITE320   003626          1200#
ITE321   003634          1203#
ITE322   003642          1206#
ITE323   003650          1209#
ITE324   003656          1212#
ITE325   003664          1215#
ITE326   003672          1218#
ITE327   003700          1221#
ITE330   003706          1224#
ITE331   003714          1227#
ITE332   003722          1230#
ITE333   003730          1233#
ITE334   003736          1236#
ITE335   003744          1239#
ITE336   003752          1242#
ITE337   003760          1245#
ITE340   003766          1248#
ITE341   003774          1251#
ITE342   004002          1254#
ITE343   004010          1257#
ITE344   004016          1260#
ITE346   004032          1266#
ITE347   004040          1269#
ITE350   004046          1272#
ITE351   004054          1275#
ITE352   004062          1278#
ITE353   004070          1281#
ITE354   004076          1284#
ITE355   004104          1287#
ITE356   004112          1290#
ITE357   004120          1293#
ITE360   004126          1297#
ITE361   004134          1300#
ITE362   004142          1303#
ITE363   004150          1306#
ITE364   004156          1309#
ITE365   004164          1312#
ITE366   004172          1315#
ITE367   004200          1318#
ITE370   004206          1321#
ITE371   004214          1324#
ITE372   004222          1327#
ITE373   004230          1330#
ITE374   004236          1333#
ITE375   004244          1336#
ITE376   004252          1339#
ITE377   004260          1342#
ITE400   004266          1345#
ITE401   004274          1348#
ITE402   004302          1351#
```

```
ITE403  004310           1354#
ITE404  004316           1357#
ITE405  004324           1360#
ITE406  004332           1363#
ITE407  004340           1366#
ITE410  004346           1369#
ITE411  004354           1372#
ITE412  004362           1375#
ITE413  004370           1378#
ITE414  004376           1381#
ITE415  004404           1384#
ITE416  004412           1387#
ITE417  004420           1390#
ITE420  004426           1393#
ITE421  004434           1396#
ITE422  004442           1399#
ITE423  004450           1402#
ITE424  004456           1405#
ITE425  004464           1408#
ITE426  004472           1411#
ITE427  004500           1414#
ITE430  004506           1417#
ITE431  004514           1420#
ITE432  004522           1423#
ITE433  004530           1426#
ITE434  004536           1429#
ITE435  004544           1432#
ITE436  004552           1435#
ITE437  004560           1438#
ITE440  004566           1441#
ITE441  004574           1444#
ITE442  004602           1447#
ITE443  004610           1450#
ITE444  004616           1453#
ITE445  004624           1456#
ITE446  004632           1459#
ITE447  004640           1462#
ITE450  004646           1465#
ITE451  004654           1468#
ITE452  004662           1471#
ITE453  004670           1474#
ITE454  004676           1477#
ITE455  004704           1480#
ITE456  004712           1483#
ITE457  004720           1486#
ITE460  004726           1489#
ITE461  004734           1492#
KBDONE  023234           4134     4151     4153#
KBTST   023142           4019     4133#
KB11E   001272            546#    4135*    4139*    4148     4194     5178     5191     5380
KB11EM  001273            547#
KDPAR0= 172360            329#
KDPAR1= 172362            330#
KDPAR2= 172364            331#
KDPAR3= 172366            332#
KDPAR4= 172370            333#
```

```
KDPAR5= 172372          334#
KDPAR6= 172374          335#
KDPAR7= 172376          336#
KDPDR0= 172320          307#
KDPDR1= 172322          308#
KDPDR2= 172324          309#
KDPDR3= 172326          310#
KDPDR4= 172330          311#
KDPDR5= 172332          312#
KDPDR6= 172334          313#
KDPDR7= 172336          314#
KERSTK= 001100           46#
KILBR5  026310         4422     4428     4478     4485     4546     4561     4576     4582     4696#
KIPAR0= 172340          318#
KIPAR1= 172342          319#
KIPAR2= 172344          320#
KIPAR3= 172346          321#
KIPAR4= 172350          322#
KIPAR5= 172352          323#
KIPAR6= 172354          324#
KIPAR7= 172356          325#
KIPDR0= 172300          296#
KIPDR1= 172302          297#
KIPDR2= 172304          298#
KIPDR3= 172306          299#
KIPDR4= 172310          300#
KIPDR5= 172312          301#
KIPDR6= 172314          302#
KIPDR7= 172316          303#
LEVEL4  026232         4311     4377     4392     4418     4434     4449     4685#
LEVEL5  026242         4417     4474     4542     4557     4572     4687#
LEVEL6  026252         4505     4590     4625     4640     4689#
LEVE6   016310         3242     3244     3248#
LF    = 000012           60#    6030     6036
LKSTAT  001260          541#    3249     3252
LKVEC   001262          542#    3251
LOADRS= 177740          170#
LOOP    005340         1547#    5524
MAINT = 177750          174#
MAPH0 = 170202          413#    1497*
MAPH00= 170202          349#     413
MAPH01= 170206          351#     415
MAPH02= 170212          353#     417
MAPH03= 170216          355#     419
MAPH04= 170222          357#     421
MAPH05= 170226          359#     423
MAPH06= 170232          361#     425
MAPH07= 170236          363#     427
MAPH1 = 170206          415#    1499*
MAPH10= 170242          365#
MAPH11= 170246          367#
MAPH12= 170252          369#
MAPH13= 170256          371#
MAPH14= 170262          373#
MAPH15= 170266          375#
MAPH16= 170272          377#
```

```
MAPH17= 170276            379#
MAPH2 = 170212            417#
MAPH20= 170302            381#
MAPH21= 170306            383#
MAPH22= 170312            385#
MAPH23= 170316            387#
MAPH24= 170320            389#
MAPH25= 170326            391#
MAPH26= 170332            393#
MAPH27= 170336            395#
MAPH3 = 170216            419#
MAPH30= 170342            397#
MAPH31= 170346            399#
MAPH32= 170352            401#
MAPH33= 170356            403#
MAPH34= 170362            405#
MAPH35= 170366            407#
MAPH36= 170372            409#
MAPH37= 170376            411#
MAPH4 = 170222            421#
MAPH5 = 170226            423#
MAPH6 = 170232            425#
MAPH7 = 170236            427#
MAPL0 = 170200            412#    1496*
MAPL00= 170200            348#     412
MAPL01= 170204            350#     414
MAPL02= 170210            352#     416
MAPL03= 170214            354#     418
MAPL04= 170220            356#     420
MAPL05= 170224            358#     422
MAPL06= 170230            360#     424
MAPL07= 170234            362#     426
MAPL1 = 170204            414#    1498*
MAPL10= 170240            364#
MAPL11= 170244            366#
MAPL12= 170250            368#
MAPL13= 170254            370#
MAPL14= 170260            372#
MAPL15= 170264            374#
MAPL16= 170270            376#
MAPL17= 170274            378#
MAPL2 = 170210            416#
MAPL20= 170300            380#
MAPL21= 170304            382#
MAPL22= 170310            384#
MAPL23= 170314            386#
MAPL24= 170320            388#
MAPL25= 170324            390#
MAPL26= 170330            392#
MAPL27= 170334            394#
MAPL3 = 170214            418#
MAPL30= 170340            396#
MAPL31= 170344            398#
MAPL32= 170350            400#
MAPL33= 170354            402#
MAPL34= 170360            404#
```

```
MAPL35= 170364      406#
MAPL36= 170370      408#
MAPL37= 170374      410#
MAPL4 = 170220      420#
MAPL5 = 170224      422#
MAPL6 = 170230      424#
MAPL7 = 170234      426#
MEMERR= 177744      172#    1565*   5575    5588*
MFPT  = 000007      550#    4137    5385
MFPTTR  023174     4136     4142#
MMR0  = 177572      197#     201
MMR1  = 177574      198#     202
MMR2  = 177576      199#     203
MMR3  = 172516      200#     204
MMSET   032114     5379     5393#
MMVEC = 000250      165#
MSG1    036426     4147     6252#
MSG3    036465     4150     6258#
MSG5    036532     4152     6265#
MSG6    036536     6266#
NEXTEN  034212     5781#    5829
NEXTTS  001270      545#    1594*   1639*   1707*   1757*   1890*   1923*   1944*   1970*   2101*   2151*   2245*   2326*
                   2611*    2650*   2690*   2732*   2770*   2803*   2849*   2920*   2953*   2986*   3020*   3057*   3097*
                   3135*    3212*   3217    3324*   3354*   3388*   3421*   3646*   3750*   3827*   3927*   4002*   4019*
                   4170*    4225*   4270*   4722*   4821*   4880*   4910*   4969*   5006*   5067*   5099*   5182*   5269
                   5284*    5314*   5379*   5408*   5561    5589    5902
PDINTE  016056     3196     3210#
PIRQ  = 177772       54#    2814*   2815    2820*   2821    2823    2830*   2832    2862*   2863*   2866*   2867*   2869*
                   2873*    2874*   2876*   2881*   2882*   2884*   2890*   2891*   2893*   2896*   2900*   2902*   2907*
                   2925*    2926*   2929*   2934*   2937*   2938*   2940    2941*   2958*   2959*   2962*   2966*   2969*
                   2970*    2972    2973*   2991*   2995*   2999*   3003*   3004*   3007    3008*   3025*   3026*   3030*
                   3035*    3038*   3039*   3042    3043*   3062*   3065*   3066*   3069*   3073*   3076*   3080*   3081*
                   3084     3085*   3102*   3103*   3107*   3113*   3115*   3116*   3118    3119*   3179*   3182*   3184*
                   4310*    4315*   4324*   4326*   4332*   4337*   4342*   4348*   4353*   4358*   4364*   4376*   4382*
                   4391*    4397*   4403*   4433*   4438*   4448*   4454*   4460*   4473*   4479*   4484*   4504*   4510*
                   4516*    4521*   4527*   4534*   4541*   4547*   4556*   4562*   4571*   4577*   4589*   4596*   4606*
                   4610*    4624*   4630*   4639*   4645*   4654*   4658*   4669*   4673*   4679*   4698*   5351*   5357*
                   5362*
PIRQVE= 000240      164#    2854*   2857*   2859*   2865*   2872*   2880*   2889*   2923*   2935*   2955*   2967*   2988*
                   3001*    3022*   3036*   3059*   3064*   3078*   3099*   3112*   3172*   3175*   4280*   4288*   4307*
                   4323*    4340*   4356*   4373*   4389*   4431*   4446*   4464*   4495*   4518*   4539*   4554*   4569*
                   4586*    4603*   4622*   4637*   4651*   4666*   5349*
PLKC  = 172544      432#    3304*   4693*
PLKSTA  001264      543#    3260    3263
PLKVEC  001266      544#    3262
PR0   = 000000       86#
PR1   = 000040       87#
PR2   = 000100       88#     522    1610
PR3   = 000140       89#
PR4   = 000200       90#
PR5   = 000240       91#     523    1602
PR6   = 000300       92#
PR7   = 000340       93#    1557    1561    1563    1703    2694    2852    2859    3141    3177    3342    3371    3407
                   3440     3472    3475    3616    3651    3654    4026    4122    4173    4185    4196    4208    4286
                   4287     4288    4415    4471    4502    5075    5151    5152    5154    5234
PS    = 177776       51#      52    1500*   5512
```

```
                                                              N  1
PDP 11/70-74MP CPU DIAGNOSTIC PART 2     MACY11 30A(1052)  17-SEP-79  10:53  PAGE 185
CEKBBD.P11    17-SEP-79 10:22            CROSS REFERENCE TABLE -- USER SYMBOLS                                   SEQ 0220

PSW   = 177776        52#   1600    1608    1640    1653*   1656    1666    1700    1701    1789    1827    1842    1865
                    1979    2008    2036    2068    2080    2113    2162    2200    2334    2351    2431    2464    2494
                    2588    2613*   2621    2693    2706    2743    2855    3137    3173    3338    3367    3403    3436
                    3469    3470    3612    3648    3649    3683    4021    4175    4187    4198    4210    4411    4467
                    4498    4723*   4724    4726*   4730*   4731    4740*   4742*   4754*   4756*   4768*   4779*   4785*
                    4789*   4795*   4797*   4799*   4806*   4809*   4823*   4860*   4862*   4881*   4882    4885    4888*
                    4889    4892    4895*   4896    4899    4914*   4916*   4925*   4928*   4936*   4938*   4940*   4943*
                    4948*   4950*   4953*   4956*   4958*   4971*   4977*   4982*   4984    4986*   4993*   4996*   5012*
                    5031*   5036*   5042*   5046*   5048*   5052*   5055*   5072    5073    5078*   5080*   5083*   5101*
                    5105*   5113*   5116*   5119*   5122*   5125*   5128*   5131*   5134*   5136*   5147*   5149*   5150
                    5176    5193    5196    5198    5199*   5205*   5210    5212    5213*   5219*   5224    5226    5227*
                    5232*   5236    5238    5240*   5245*   5249    5251    5256*   5258    5262*   5287*   5290*
PWRVEC= 000024       158#   1512*   1513*   2850*   2898*   3656*   3751*   6217*   6218*   6226*   6241*   6242*
QUIT    034644      1556    5451    5872#
RESVEC= 000010       153#   1521*   1528*   1597*   1619*   1708*   1736*   1767*   1853*   1892*   1903*   1907*   1925*
                    1929*   1931*   1946*   1951*   1953*   2103*   2123*   2247*   2261*   2612*   2635*   2651*   2676*
                    2678*   2697*   2718*   2733*   2752*   4023*   4026*   4028*   4036*   4042*   4048*   4057*   4065*
                    4076*   4085*   4091*   4100*   4109*   4116*   4121*   4122*   4136*   4145*   4181*
RKCS1   001250       537#   3229    3230
RKVEC   001252       538#   3231
RPCS1   001244       535#   3223            3236
RPVEC   001246       536#   3237
RSCS1   001240       533#   3220            3233
RSVEC   001242       534#   3234
SDPAR0= 172260       285#
SDPAR1= 172262       286#
SDPAR2= 172264       287#
SDPAR3= 172266       288#
SDPAR4= 172270       289#
SDPAR5= 172272       290#
SDPAR6= 172274       291#
SDPAR7= 172276       292#
SDPDR0= 172220       263#
SDPDR1= 172222       264#
SDPDR2= 172224       265#
SDPDR3= 172226       266#
SDPDR4= 172230       267#
SDPDR5= 172232       268#
SDPDR6= 172234       269#
SDPDR7= 172236       270#
SEQENC  035050      5907#
SIPAR0= 172240       274#
SIPAR1= 172242       275#
SIPAR2= 172244       276#
SIPAR3= 172246       277#
SIPAR4= 172250       278#
SIPAR5= 172252       279#
SIPAR6= 172254       280#
SIPAR7= 172256       281#
SIPDR0= 172200       252#
SIPDR1= 172202       253#
SIPDR2= 172204       254#
SIPDR3= 172206       255#
SIPDR4= 172210       256#
SIPDR5= 172212       257#
SIPDR6= 172214       258#
```

```
SIPDR7= 172216             259#
SIZEHI= 177762             183#
SIZELO= 177760             181#
SLDATA  034324            5774*   5801    5808#   5809*   5810    5813
SPDATA  034370            5775*   5806    5815    5820#   5822    5825
SP16    034622            5795    5858#
SP38    034600            5804    5855#
SR0   = 177572             201#
SR1   = 177574             202#
SR2   = 177576             203#
SR3   = 172516             204#
STACK = 001100              45#     46      47      48    1505    1595    1645    1709    1727    1729    1739    1741    1768
                          1891    1924    1945    2102    2246    2772    2804    2860    2904    2921    2987    3021    3058
                          3098    3143    3162    3178    3218    3336    3400    3433    3481    3486    3494    3505    3515
                          3526    3570    3661    3665    3671    3695    3709    3719    3723    3770    3908    3930    3948
                          3958    3962    3966    3977    4029    4058    4066    4110    4201    4227    4330    4346    4362
                          4368    4375    4390    4416    4472    4503    4525    4532    4612    4617    4659    4674    4678
                          4780    4786    4793    4913    4919    4924    4929    4931    5009    5014    5017    5019    5028
                          5044    5081    5100    5285    5319    5332    5341    5350    5536    5564
START   004742             445    1495#
STKLMT= 177774              53#   3692*   3696*   3702*   3753*   3757    3760    3761*   3763    3767    3769*   3775*   3777*
                          3790*   3888    3899*   3907*
STYPE   034162            5773#   5787
SUBTAB  073276            1552    5883    8992#
SUPSTK= 000700              47#
SWR   = 177570              55#     56    3215    3243    3269    3327    3329    3389    3391    3422    3424    3830    3883
                          4289    4291    4299    4301    4486    4488    5411    5419    5427    5436    5514    5541    5569
                          5607*   5621    5623    5629    5636    5675    5678    5685    5689    5695
SW0   = 000001             121#   5411
SW00  = 000001             111#    121
SW01  = 000002             110#    120
SW02  = 000004             109#    119
SW03  = 000010             108#    118
SW04  = 000020             107#    117
SW05  = 000040             106#    116
SW06  = 000100             105#    115
SW07  = 000200             104#    114
SW08  = 000400             103#    113
SW09  = 001000             102#    112
SW1   = 000002             120#
SW10  = 002000             101#
SW11  = 004000             100#
SW12  = 010000              99#
SW13  = 020000              98#
SW14  = 040000              97#   3215
SW15  = 100000              96#
SW2   = 000004             119#
SW3   = 000010             118#
SW4   = 000020             117#   3329    4301
SW5   = 000040             116#   3391    4291
SW6   = 000100             115#   3424    4488
SW7   = 000200             114#   5419    5427    5436
SW8   = 000400             113#   3327    3389    3422    4289    4299    4486
SW9   = 001000             112#
SYSTID= 177764             185#
TAPE1 = ****** U          5865
```

```
TAPE2 = ****** U         5865
TBITVE= 000014            154#    1518*   1519*   2785*   4180*   4202*   4230*   4273*   5330*   5342*   5346*
TERM    034414           5818     5827#
THRESP  034637           5849     5862#
TKVEC = 000060            161#    1556*   1557*   5447*   5451*
TMCS1   001254            539#    3226    3239
TMVEC   001256            540#    3240
TPVEC = 000064            162#    1643*   3337*   3340*   3342*   3355*   3614*   3616*   3617*   4279*   4286*   4306*   4374*
                         4388*    4408*   4430*   4445*   5317*   5331*   5348*
TRAPVE= 000034            160#    1510*   1511*
TRTVEC= 000014            155#
TST1    005476           1567     1592#   8940
TST10   007460           1943     1944    1952    1969#   8947
TST11   010302           1970     2082    2099#   8948
TST12   010442           2100     2101    2119    2150#   8949
TST13   011004           2151     2230    2243#   8950
TST14   011120           2244     2245    2257    2325#   8951
TST15   012600           2326     2591    2609#   8952
TST16   012746           2610     2611    2632    2649#   8953
TST17   013120           2650     2677    2688#   8954
TST2    005660           1593     1594    1617    1636#   8941
TST20   013306           2689     2690    2715    2730#   8955
TST21   013446           2731     2732    2749    2768#   8956
TST22   013562           2769     2770    2801#   8957
TST23   013766           2802     2803    2831    2847#   8958
TST24   014336           2848     2849    2899    2919#   8959
TST25   014474           2920     2939    2952#   8960
TST26   014632           2953     2971    2985#   8961
TST27   015002           2986     3006    3019#   8962
TST3    006154           1638     1639    1673    1696#   8942
TST30   015156           3020     3041    3056#   8963
TST31   015356           3057     3083    3096#   8964
TST32   015530           3097     3117    3134#   8965
TST33   016650           3135     3211    3212    3255    3266    3273    3323#   8966
TST34   017144           3324     3335    3353    3354    3386#   8967
TST35   017274           3387     3388    3397    3399    3410    3419#   8968
TST36   017424           3420     3421    3430    3432    3443    3466#   8969
TST37   020016           3535     3546#   8970
TST4    006436           1706     1707    1731    1756#   8943
TST40   020432           3645#    8971
TST41   021166           3646     3733    3748#   8972
TST42   021432           3749     3750    3817#   8973
TST43   022042           3826     3827    3911    3926#   8974
TST44   022476           3927     4001    4002    4018#   8975
TST45   023234           4168#    8976
TST46   023504           4169     4170    4224#   8977
TST47   023556           4225     4234    4269#   8978
TST5    007200           1757     1873    1888#   8944
TST50   026366           4270     4682    4720#   8979
TST51   027016           4721     4722    4772    4775    4811    4819#   8980
TST52   027232           4820     4821    4865    4879#   8981
TST53   027402           4880     4897    4909#   8982
TST54   027652           4910     4967#   8983
TST55   030016           4968     4969    5005#   8984
TST56   030252           5006     5065#   8985
TST57   030400           5066     5067    5086    5098#   8986
```

```
TST6     007314          1889    1890    1898    1921#   8945
TST60    030610          5099    5145#   8987
TST61    030666          5153    5175#   8988
TST62    031432          5182    5267    5283#   8989
TST63    031546          5284    5313#   8990
TST64    032042          5314    5378#   8991
TST65    032114          5407#
TST7     007374          1922    1923    1930    1942#   8946
TWOSP    034640          5552    5556    5580    5584    5745    5762    5863#
TYPDS  = 104410          5485    5492    6211#
TYPE   = 104400          1538    3245    3256    3271    3333    3395    3428    4147    4150    4152    5413    5418    5425
                         5434    5440    5445    5479    5486    5493    5537    5543    5552    5556    5565    5571    5580
                         5584    5680    5688    5713    5728    5730    5733    5735    5738    5745    5752    5754    5761
                         5768    5770    5795    5796    5798    5803    5821    5827    5832    5839    5847    5849    5878
                         5891    5898    5949    5955    5961    6002    6097    6173    6207#   6243
TYPEBN   034522          5814    5826    5838#
TYPEHE   034244          5779    5790#
TYPOC  = 104402          5424    5433    5444    5551    5555    5559    5579    5583    5587    5742    5760    5767    5812
                         5824    5954    5960    6208#
TYPON  = 104406          6210#
TYPOS  = 104404          6209#
TYPSLE   034102          5710    5712    5752#
UDPAR0=  177660           241#
UDPAR1=  177662           242#
UDPAR2=  177664           243#
UDPAR3=  177666           244#
UDPAR4=  177670           245#
UDPAR5=  177672           246#
UDPAR6=  177674           247#
UDPAR7=  177676           248#
UDPDR0=  177620           219#
UDPDR1=  177622           220#
UDPDR2=  177624           221#
UDPDR3=  177626           222#
UDPDR4=  177630           223#
UDPDR5=  177632           224#
UDPDR6=  177634           225#
UDPDR7=  177636           226#
UIPAR0=  177640           230#
UIPAR1=  177642           231#
UIPAR2=  177644           232#
UIPAR3=  177646           233#
UIPAR4=  177650           234#
UIPAR5=  177652           235#
UIPAR6=  177654           236#
UIPAR7=  177656           237#
UIPDR0=  177600           208#
UIPDR1=  177602           209#
UIPDR2=  177604           210#
UIPDR3=  177606           211#
UIPDR4=  177610           212#
UIPDR5=  177612           213#
UIPDR6=  177614           214#
UIPDR7=  177616           215#
USESTK=  000600            48#
WAIT     026354          4686    4688    4692    4695    4707#
```

```
$BDADR   001122             493#
$BDDAT   001126             495#
$BELL    001176             516#   5680    5703
$CHARC   035556            6012*   6019    6028*   6033#
$CLR.T   032552            5498    5501#
$CMTAG   001100             481#   1500    1501    1508    1514    1515    1516
$CM1   = 000003             507#    508#    509#    510#
$CM2   = 000006             507#    508#    509#    510#
$CM3   = 000003             505#    507
$CM4   = 000004             510#    511#    512#    513#    514#
$CRLF    001203             518#   5493    5688    5703    5713    5730    5735    5738    5769    5799    5828    5961    6002
                           6036
$DBLK    036224            6139    6173    6181#
$DOAGN   032602            5475    5496    5505    5511#
$DTBL    036214            6142    6177#
$ENDAD   032572             468    1536    5507#   5692
$ENDBR   016644            3297    3299    3311    3313#
$ENDCT   032410            1514    5477#
$ENULL   032650            5526#
$EOP     032354            5408    5414    5416    5467#   8992
$EOPCT   032402            1514*   5474#   5478
$EPIRQ   001216             524#   2832*   2940*   2972*   3007*   3042*   3084*   3118*   7049    7174
$ERFLG   001103             484#   5597    5625    5627    5633*   5654    5672*   5703
$ERMAX   001115             490#   1517*   5627    5649*   5654
$ERPSW   001206             520#   1600*   1601*   1602    1656*   1669    1671    1789*   1790*   1827*   1828*   1829    1842*
                           1843*   1844    1865*   1871*   1872    1979*   1980*   1981    2008*   2009*   2014    2036*   2038*
                           2039    2068*   2070*   2080*   2081*   2113*   2162*   2186*   2187    2200*   2213*   2214    2334*
                           2335*   2336    2338    2351*   2356*   2357    2406    2431*   2449*   2450    2464*   2469*   2470
                           2494*   2499*   2500    2588*   2589*   2590    2621*   2630*   2631    2706*   2713*   2714    2743*
                           2747*   2748    4187*   4189    4210*   4212    4885*   4886*   4892*   4893*   4899*   4900*   5150*
                           5151*   5152    5183*   5198*   5212*   5226*   5238*   5251*   5258*   5259    6292    6554    6638
$ERROR   033536            1508    3984    3993    5670#
$ERRPC   001116             491#   5535*   5549    5563*   5577    5682*   5683*   5684    5703    5756    6292    6313    6366
                           6447    6539    6554    6575    6603    6638    7049    7174    7484    7494    7554    8207
$ERRTB   001274             567#   5725
$ERTTL   001112             488#   5490    5494*   5681*   5703
$ESCAP   001174             515#   1516*   1593*   1638*   1706*   1889*   1922*   1943*   2100*   2244*   2610*   2689*   2731*
                           2769*   2802*   2848*   3211*   3353*   3387*   3420*   3749*   3826*   3832    4001*   4169*   4721*
                           4737*   4820*   4968*   5066*   5561*   5589*   5648*   5698    5700    5703
$FILLC   001150             503#   6005    6036
$FILLS   001147             502#   6036
$GDADR   001120             492#
$GDDAT   001124             494#
$GET42   032534            5495#
$HD    = 000001               1#     16      17
$ICNT    001104             485#   1637*   3325*   3828*   4271*   5315*   5640*   5641    5643*   5653
$ILLUP   036410            6217    6246#
$ITEMB   001114             489#   5684*   5703    5715    5716
$KW11L   016530            3250    3289#
$KW11P   016574            3261    3302#
$LF      001204             519#   5703    6036
$LOOP    032642            5519    5523#
$LPADR   001106             486#   1530*   1641*   1698*   1758*   2691*   2805*   3326*   3467*   3547*   3647*   3829*   4020*
                           4272*   5070*   5316*   5631*   5646*   5651    5653
$LPERR   001110             487#   1531*   1642*   1699*   1759*   1795*   1820*   1835*   1857*   2001*   2028*   2044*   2059*
                           2074*   2193*   2220*   2346*   2413*   2423*   2456*   2476*   2487*   2506*   2524*   2543*   2562*
```

```
                            2581*   2666*   2692*   2806*   2933*   2965*   2998*   3033*   3075*   3110*   3142*   3161*   3170*
                            3468*   3477*   3491*   3500*   3511*   3525*   3532*   3554*   3561*   3569*   3580*   3586*   3599*
                            3610*   3655*   3681*   3690*   3701*   3713*   3730*   3949*   3976*   3994*   4027*   4035*   4041*
                            4047*   4056*   4064*   4075*   4084*   4090*   4099*   4108*   4115*   4178*   4200*   4305*   4321*
                            4338*   4354*   4372*   4387*   4407*   4429*   4444*   4463*   4494*   4517*   4537*   4552*   4567*
                            4585*   4602*   4620*   4635*   4650*   4665*   4738*   4752*   4766*   4794*   4805*   4912*   4923*
                            4935*   4947*   5043*   5071*   5186*   5204*   5218*   5231*   5244*   5318*   5329*   5347*   5631
                            5647*   5653    5697
$MXCNT   033534             5644    5653#
$NULL    001146              501#   6007    6036
$NWTST=  000001             1570#   1572    1624#   1626    1682#   1684    1743#   1745    1876#   1878    1912#   1914    1933#
                            1935    1955#   1957    2088#   2090    2125#   2127    2232#   2234    2264#   2266    2594#   2596
                            2637#   2639    2680#   2682    2720#   2722    2757#   2759    2791#   2793    2834#   2836    2910#
                            2912    2943#   2945    2976#   2978    3010#   3012    3045#   3047    3087#   3089    3122#   3124
                            3316#   3318    3380#   3382    3413#   3415    3446#   3448    3537#   3539    3628#   3630    3736#
                            3738    3792#   3794    3914#   3916    4012#   4014    4155#   4157    4217#   4219    4236#   4238
                            4712#   4714    4814#   4816    4873#   4875    4903#   4905    4961#   4963    4999#   5001    5057#
                            5059    5089#   5091    5140#   5142    5156#   5158    5272#   5274    5305#   5307    5365#   5367
                            5394#   5396
$OCNT    036004             6069*   6098*   6111#
$OMODE   036006             6064*   6068*   6073    6076*   6087*   6113#
$OVER    033520             5608    5624    5632    5642    5650#
$PASS    001100              482#   1566    3213    3331    3393    3426    5409    5471*   5472*   5483    5525    5638    5654
$POWER   036416             6244    6249#
$PR2     001212              522#   6292
$PR5     001214              523#   6292
$PWRDN   036266             1512    2898    3751    6217#   6241
$PWRMG   036404             6244#
$PWRUP   036334             6226    6231#
$QUES    001202              517#   5703
$RDCHR=  ****** U           6212
$RDDEC=  ****** U           6212
$RDLIN=  ****** U           6212
$RDOCT=  ****** U           6212
$REGAD   001152              505#
$REG0    001154              507#   1725*   1732*   1988*   1996*   2010*   2054*   2071*   2083*   2115*   2120*   2163*   2201*
                            2258*   2372*   2392*   2402*   2438*   2445*   2518*   2532*   2551*   2570*   2622*   2663*   3191*
                            3529*   3727*   3912*   5034*   5295*   5781    5830    5916*   5953    6366    6539    6575    6638
$REG1    001156              508#   1726*   1733*   1811*   1868*   1997*   2011*   2055*   2072*   2084*   2116*   2121*   2164*
                            2202*   2259*   2373*   2393*   2403*   2439*   2446*   2520*   2533*   2552*   2571*   2664*   5037*
                            5948*   5959    6366    6447    6575    6638
$REG2    001160              509#
$RTRN    032640             1518    1520*   1525*   2785    4273    5342    5346    5499    5520#
$SAVRE=  ****** U           6212
$SAVR6   036414             6225*   6231    6232*   6233*   6248#
$SCOPE   033270             1506    5265    5606#
$SETUP=  000037              434#   1506    1508    1510    1512    1514    1515    1516    1518    1530    1536    5469    5689
$STUP =  177777              434#
$SVLAD   033472             5616    5645#
$SVPC =  000204              466#    471
$SWR  =  177400               16      17#     25      26      27      28      29      30      31      32     514     515     516
                            1515    1516    1518    1530    1531    1593    1637    1697    1757    1889    1922    1943    1970
                            2100    2151    2244    2326    2610    2650    2689    2731    2769    2802    2848    2920    2953
                            2986    3020    3057    3097    3135    3324    3387    3420    3467    3547    3646    3749    3818
                            3927    4019    4169    4225    4270    4721    4820    4880    4910    4968    5006    5066    5099
                            5146    5176    5284    5314    5379    5408    5463    5470    5497    5512    5525    5598    5599
```

```
                    5600     5601     5602     5607     5619     5621     5622     5625     5626     5627     5634     5635     5636
                    5647     5650     5653     5661     5662     5663     5664     5665     5666     5675     5678     5685     5689
                    5695     5703
$SWRMK= 000000        32       33     5602     5603     5623
$TBIT   032646      1529*    5516*    5525#
$TIMES  001172       514#    1515*    5470*    5634*    5641     5644*    5653
$TKB    001140       498#    1558*    5446*    5452*    5873     5896*
$TKS    001136       497#    1503     1559*    1697*    5448*    5453*    5897*
$TMPO   001162       510#    1608*    1609*    1610     1665*    1668*    1674*    1791*    1831*    1848*    1874*    1987*    1991*
                    1994*    2012*    2022*    2037*    2047*    2063*    2085*    2108*    2114*    2157*    2189*    2203*    2216*
                    2252*    2342*    2359*    2362*    2376*    2427*    2452*    2472*    2502*    2517*    2528*    2547*    2566*
                    2592*    2623*    2633*    2716*    2750*    2813*    2815     2818*    2821     2825*    2828*    3192*    3530*
                    3588     3728*    3768*    3774*    3787*    3842*    3858     3932     3933*    3951     3997     3998*    4822*
                    4864     4866*    4884*    4891*    4898*    5154*    5177*    5180*    5203*    5239*    5252*    5261*    5296*
                    5390*    5421*    5422     5430*    5431     5441*    5442     5547*    5557     5575*    5585     5919*    5926*
                    5928     5938     6292     6539     6554     6575     6603     6638     7049     7484     7494     7554
$TMP1   001164       511#    1812*    1869*    1893     1926     1995*    2013*    2049*    2064*    2086*    2109*    2158*    2204*
                    2253*    2363*    2377*    2428*    2519*    2529*    2548*    2567*    2737     3592*    3594     3602     3603*
                    3605     3788*    3843*    3859     3998     4826     4970*    4988     4994*    5176*    5266     5389*    5920*
                    5927     5936*    6447     6575     6603     6638     7554     8207
$TMP2   001166       512#    1947     2065*    2104     2249     2753     3591     3834*    3885     3892*    6638
$TMP3   001170       513#    1640*    1700*    1760     2693*    2807     3469*    3548     3648*    4175*    4198*    4277     5072*
$TN   = 000066        16#    1567     1570     1593#    1617     1624     1637#    1638     1673     1682     1697#    1706     1731
                    1743     1757#    1873     1876     1889#    1898     1912     1922#    1930     1933     1943#    1952     1955
                    1970#    2082     2088     2100#    2119     2125     2151#    2230     2232     2244#    2257     2264     2326#
                    2591     2594     2610#    2632     2637     2650#    2677     2680     2689#    2715     2720     2731#    2749
                    2757     2769#    2791     2802#    2831     2834     2848#    2899     2910     2920#    2939     2943     2953#
                    2971     2976     2986#    3006     3010     3020#    3041     3045     3057#    3083     3087     3097#    3117
                    3122     3135#    3210     3255     3266     3273     3316     3324#    3335     3352     3380     3387#    3397
                    3399     3410     3413     3420#    3430     3432     3443     3446     3467#    3535     3537     3547#    3628
                    3646#    3733     3736     3749#    3792     3818#    3826     3911     3914     3927#    4001     4012     4019#
                    4155     4169#    4217     4225#    4234     4236     4270#    4682     4712     4721#    4772     4775     4811
                    4814     4820#    4865     4873     4880#    4897     4903     4910#    4961     4968#    4999     5006#    5057
                    5066#    5086     5089     5099#    5140     5146#    5153     5156     5176#    5181     5267     5272     5284#
                    5305     5314#    5365     5379#    5394     5408#
$TPB    001144       500#    1648*    1651*    1658*    3620*    5322*    5336*    5352*    6025*    6036
$TPFLG  001151       504#    5985     6036
$TPS    001142       499#    1649     1652*    1657     1662     1677*    1679*    3345*    3349*    3358*    3361*    3378*    3619*
                    3622*    3625*    4316*    4320*    4381*    4396*    4404*    4423*    4439*    4453*    4459*    4685*    4699*
                    5321*    5325*    5335*    5340*    5356*    5361*    6023     6036
$TRAP   036234      1510     6191#
$TRP  = 000012      6198#    6208#    6209#    6210#    6211#    6212#
$TRPAD  036254      6195     6206#
$TSTNM  001102       483#    5469*    5548     5576     5597     5623     5645*    5650     5654     5671     5674     5703     5910
                    5947*
$TYPBN= ****** U    6212
$TYPDS  036010      6127#    6211
$TYPE   035346      5985#    6198     6207
$TYPEC  035512      6004     6011     6018     6023#    6024
$TYPEE  034202      5772*    5778#    5785*    5786     5792
$TYPEX  035560      6029     6031     6034#
$TYPOC  035606      6067#    6208
$TYPON  035622      6066     6069#    6210
$TYPOS  035562      6062#    6209
$XTSTR  033276      5610#
$$DONE  034422      5788     5830#
```

```
$$GET4= 000001          5497#   5504#
$$TRP = 000002          6197#   6208    6209    6210    6211    6212
$$TSTN  001210           521#   5548*   5553    5576*   5581    5671*   5709    5711    5763    6292    6313    6366    6447
                        6539    6554    6575    6603    6638    7049    7174    7484    7494    7554    8207
$OFILL  036005          6063*   6067*   6077    6112#
.     = 073300           437#    441     443#    466     467#    469#    471#    479#    520    1504    1530    1531    1541#
                        5482#   5525    5527#   5540#   5546#   5568#   5574#   5653    5654    5703    5835#   5864#   5881#
                        5894#   5952#   6036    6181#   6228    6247    6268#   6291#   6312#   6538#   6553#   6602#   7048#
                        7483#   7493#   7718#   8206#   8938#   8939
```

```
                                                                              I  2
PDP 11/70-74MP CPU DIAGNOSTIC PART 2    MACY11 30A(1052)  17-SEP-79  10:53  PAGE 194
CEKBBD.P11    17-SEP-79 10:22           CROSS REFERENCE TABLE -- MACRO NAMES                                          SEQ 0228

CHAIN     5455#   5502
COMMEN       1#    431#
DOC          1#
DOCEXP       1#
ENDCOM       1#    431#
ERROR       49#   1614    1615    1618    1622    1623    1661    1675    1678    1680    1717    1720    1722    1728    1735
          1740    1742    1776    1780    1782    1792    1806    1810    1813    1817    1832    1849    1851    1854    1870
          1875    1901    1902    1910    1911    1932    1954    1989    1992    1998    2020    2023    2025    2041    2056
          2073    2087    2117    2122    2124    2171    2178    2182    2184    2190    2207    2211    2217    2226    2231
          2260    2262    2340    2343    2360    2366    2370    2374    2380    2384    2388    2394    2398    2404    2408
          2410    2420    2436    2440    2447    2453    2467    2473    2484    2497    2503    2521    2536    2540    2555
          2559    2574    2578    2593    2626    2629    2634    2636    2661    2665    2674    2679    2709    2712    2717
          2719    2746    2751    2755    2756    2781    2782    2783    2833    2870    2877    2885    2894    2895    2901
          2906    2908    2927    2931    2942    2960    2964    2974    2993    2997    3009    3028    3032    3044    3067
          3070    3074    3086    3105    3109    3120    3150    3152    3156    3166    3185    3193    3197    3350    3375
          3376    3377    3411    3444    3482    3487    3496    3506    3517    3531    3536    3558    3566    3577    3583
          3596    3607    3623    3663    3667    3673    3678    3687    3698    3712    3722    3729    3734    3771    3776
          3780    3783    3786    3789    3913    3940    3943    3964    3970    3971    3987    3989    4003    4009    4033
          4039    4045    4051    4062    4070    4079    4088    4094    4103    4113    4119    4191    4192    4193    4214
          4215    4235    4317    4333    4349    4365    4383    4398    4424    4440    4455    4480    4511    4528    4548
          4563    4578    4597    4613    4631    4646    4660    4675    4727    4733    4745    4749    4759    4763    4781
          4787    4790    4802    4812    4861    4863    4887    4894    4901    4920    4932    4944    4957    4978    4983
          5022    5024    5029    5038    5053    5087    5114    5120    5123    5132    5135    5155    5200    5214    5228
          5241    5253    5263    5291    5297    5302    5326    5343    5358    5391    5562    5590
ESCAPE       1#      4#    431#
LABEL        1#
LABEL1       1#
MSGP      5365#   5367
MSG100    1570#   1572
MSG101    1624#   1626
MSG102    1682#   1684
MSG103    1743#   1745
MSG104    1876#   1878
MSG105    1912#   1914
MSG106    1933#   1935
MSG107    1955#   1957
MSG110    2088#   2090
MSG111    2125#   2127
MSG112    2232#   2234
MSG113    2263#   2266
MSG114    2594#   2596
MSG115    2637#   2639
MSG116    2680#   2682
MSG117    2720#   2722
MSG121    2757#   2759
MSG122    2791#   2793
MSG123    2834#   2836
MSG124    2910#   2912
MSG125    2943#   2945
MSG126    2976#   2978
MSG127    3010#   3012
MSG130    3045#   3047
MSG131    3087#   3089
MSG132    3122#   3124
MSG133    3316#   3318
MSG135    3380#   3382
```

```
MSG136     3413#    3415
MSG137     3446#    3448
MSG140     3628#    3630
MSG141     3736#    3738
MSG142     3792#    3794
MSG143     3914#    3916
MSG144     4155#    4157
MSG145     4217#    4219
MSG146     4236#    4238
MSG147     4712#    4714
MSG150     4814#    4816
MSG151     4873#    4875
MSG152     4903#    4905
MSG153     4961#    4963
MSG154     4999#    5001
MSG155     5057#    5059
MSG156     5089#    5091
MSG157     5140#    5142
MSG160     5156#    5158
MSG173     5394#    5396
MS137A     3537#    3539
MS145A     4012#    4014
MS160A     5272#    5274
MS160B     5305#    5307
MULT          1#     431#
MYTAGS        1#     520
NEWTST        1#       4#     431#    1570     1624     1682     1743     1876     1912     1933     1955     2088     2125     2232     2264
           2594     2637     2680     2720     2757     2791     2834     2910     2943     2976     3010     3045     3087     3122     3316
           3380     3413     3446     3537     3628     3736     3792     3914     4012     4155     4217     4236     4712     4814     4873
           4903     4961     4999     5057     5089     5140     5156     5272     5305     5365     5394
POP           1#     431#    6168     6235
PUSH          1#     431#    6127     6219
SAVEAD        1#    1593     1638     1706     1889     1922     1943     2100     2244     2610     2689     2731     2769     2802     2848
           3210     3352     3387     3420     3749     3826     4001     4169     4721     4820     4968     5066
SCOPE        50#    1592     1636     1696     1756     1888     1921     1942     1969     2099     2150     2243     2325     2609     2649
           2688     2730     2768     2801     2847     2919     2952     2985     3019     3056     3096     3134     3323     3386     3419
           3466     3546     3645     3748     3817     3926     4018     4168     4224     4269     4720     4819     4879     4909     4967
           5005     5065     5098     5145     5175     5283     5313     5378     5407     5468
SETTRA     6198#    6208     6209     6210     6211
SETUP         1#     431#    1500
SKIP          1#     431#    1567     1617     1672     1731     1873     1898     1930     1952     2082     2119     2230     2257     2591
           2632     2677     2715     2749     2831     2899     2939     2971     3006     3041     3083     3117     3196     3255     3266
           3272     3334     3396     3399     3410     3429     3432     3443     3535     3733     3911     4234     4682     4772     4775
           4811     4865     4897     5086     5153     5267
SLASH         1#     431#
SPACE       431#
STARS         1#       4#     431#     446      472      551     1533     1545     1570     1591     1624     1635     1682     1695     1743
           1755     1793     1818     1833     1855     1876     1887     1912     1920     1933     1941     1955     1968     1999     2026
           2042     2057     2088     2098     2125     2149     2191     2218     2232     2242     2264     2324     2344     2411     2421
           2454     2474     2485     2504     2522     2541     2560     2579     2594     2608     2637     2648     2680     2687     2720
           2729     2757     2767     2787     2789     2791     2800     2834     2846     2910     2918     2943     2951     2976     2984
           3010     3018     3045     3055     3087     3095     3122     3133     3198     3209     3316     3322     3380     3385     3413
           3418     3446     3465     3537     3545     3628     3644     3736     3747     3792     3816     3818     3825     3914     3925
           4012     4017     4054     4073     4082     4097     4106     4125     4154     4155     4167     4171     4177     4217     4223
           4236     4268     4297     4318     4335     4351     4366     4385     4401     4426     4442     4457     4482     4513     4530
           4550     4565     4580     4599     4615     4633     4648     4662     4712     4719     4735     4750     4764     4791     4803
```

```
          4814     4818     4873     4878     4903     4908     4961     4966     4999     5004     5057     5064     5089     5097     5140
          5144     5156     5174     5184     5201     5215     5229     5242     5254     5272     5282     5305     5312     5365     5377
          5394     5406     5455     5528     5534     5591     5654     5703     5708     5747     5751     5865     5903     5963     6036
          6114     6182     6212
TRMTRP    6198#
TYPBIN       1#     431#
TYPDEC       1#     431#    5483     5490
TYPNAM       1#     431#    1534
TYPNUM       1#     431#
TYPOCS       1#     431#
TYPOCT       1#     431#    5422     5431     5442     5549     5553     5557     5577     5581     5585     5740     5758     5765     5810
          5822     5953     5959
TYPTXT       1#     431#    5479     5486     5537     5543     5565     5571     5832     5878     5890     5898     5949     5955
$IUT         1#
$SAVEA       1#    1757     1970     2151     2326     2650     2920     2953     2986     3020     3057     3097     3135     3324     3646
          3927     4225     4270     4880     4910     5006     5099     5181     5284     5314
$SYNC        1#
$$CMRE     472#     507      508      509
$$CMTM     472#     510      511      512      513
$$ESCA       1#       4#     431#
$$NEWT       1#       4#     431#    1570     1624     1682     1743     1876     1912     1933     1955     2088     2125     2232     2264
          2594     2637     2680     2720     2757     2791     2834     2910     2943     2976     3010     3045     3087     3122     3316
          3380     3413     3446     3537     3628     3736     3792     3914     4012     4155     4217     4236     4712     4814     4873
          4903     4961     4999     5057     5089     5140     5156     5272     5305     5365     5394
$$SET     6198#    6208     6209     6210     6211
$$SETU       4#    1518#
$$SKIP       1#     431#    1567     1617     1673     1731     1873     1898     1930     1952     2082     2119     2230     2257     2591
          2632     2677     2715     2749     2831     2899     2939     2971     3006     3041     3083     3117     3255     3266     3273
          3335     3397     3399     3410     3430     3432     3443     3535     3733     3911     4234     4682     4772     4775     4811
          4865     4897     5086     5153     5267
.EQUAT       1#
.HEADE       1#       4#       6
.KT11        1#
.SETUP       1#       4#     434
.SWRHI       1#       4#      20
.SWRLO       4#      33#      39
.$ACT1       1#       4#     446
.$CATC       1#       4#     434
.$CMTA       1#       4#     472
.$DB2D       1#
.$DB2O       1#
.$DIV        1#
.$EOP        1#       4#    5455#
.$ERRO       1#       4#    5654
.$ERRT       1#       4#
.$MULT       1#
.$POWE       1#       4#    5591#    6212
.$RAND       1#
.$RDDE       1#
.$RDOC       1#
.$READ       1#
.$SAVE       1#
.$SB2D       1#
.$SB2O       1#
.$SCOP       1#       4#    5591
.$SIZE       1#
```

```
.$SUPR        1#
.$TRAP        1#        4#    5591#    6182
.$TYPB        1#
.$TYPD        1#        4#    5591#    6114
.$TYPE        1#        4#    5963
.$TYPO        1#        4#    5591#    6036
.1170         1#        4#      41


. ABS.  073300      000

 ERRORS DETECTED:  0

 DSKZ:CEKBBD,CEKBBD.LST/CRF/SOL=CEKBBD.SML,CEKBBD.P11
 RUN-TIME: 60 90 10 SECONDS
 RUN-TIME RATIO: 357/161=2.2
 CORE USED:  35K  (69 PAGES)
```