

This page contains a grid of 100 small diagnostic test diagrams, arranged in 10 rows and 10 columns. Each diagram is a technical drawing or flowchart used for testing a specific component of the PDP-11/70 CPU. The diagrams are organized into sections, with the first column containing the most detailed tests and the last column containing the simplest or most common tests. The diagrams include various tables, flowcharts, and text blocks, all rendered in a light blue color on a dark background.

PDP11/70

CPU DIAGNOSTIC
MD-11-DEKBB-C

EP-DEKBB-C-DL-A

JAN 1977

COPYRIGHT © 1977

digital

FICHE 2 OF 2

MADE IN USA

This microfiche card contains a grid of frames, likely representing a diagnostic test sequence or data collection. The frames are arranged in approximately 10 rows and 4 columns. Each frame contains a different set of data, possibly including test results, error codes, or system status information. The data is presented in a structured, tabular format, with some frames showing what appears to be a list of values or a small table. The overall layout is consistent with standard microfiche conventions for technical documentation.

IDENTIFICATION

SEG 0001

PRODUCT CODE: M.11NDEC-11-DEKBB-C-D
PRODUCT NAME: PDP-11/70 CPU DIAGNOSTIC PART 2
DATE : SEPTEMBER, 1976
MAINTAINER: DIAGNOSTIC ENGINEERING
AUTHORS: DON MONROE

THE INFORMATION IN THIS DOCUMENT IS SUBJECT TO CHANGE WITHOUT NOTICE AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT CORPORATION. DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR ANY ERRORS THAT MAY APPEAR IN THIS MANUAL.

THE SOFTWARE DESCRIBED IN THIS DOCUMENT IS FURNISHED TO THE PURCHASER UNDER A LICENSE FOR USE ON A SINGLE COMPUTER SYSTEM AND CAN BE COPIED (WITH INCLUSION OF DIGITAL'S COPYRIGHT NOTICE) ONLY FOR USE IN SUCH SYSTEM, EXCEPT AS MAY OTHERWISE BE PROVIDED IN WRITING BY DIGITAL.

DIGITAL EQUIPMENT CORPORATION ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS SOFTWARE ON EQUIPMENT THAT IS NOT SUPPLIED BY DIGITAL.

COPYRIGHT (C) 1975, 1976 BY DIGITAL EQUIPMENT CORPORATION

CONTENTS

1. ABSTRACT
2. REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 STORAGE
 - 2.3 PRELIMINARY PROGRAMS
3. LOADING PROCEDURE
 - 3.1 METHOD
4. STARTING PROCEDURE
 - 4.1 CONTROL SWITCH SETTINGS
 - 4.2 STARTING ADDRESS
 - 4.3 PROGRAM AND OPERATOR ACTION
5. OPERATING PROCEDURE
 - 5.1 OPERATIONAL SWITCH SETTINGS
 - 5.2 SUBROUTINE ABSTRACTS
 - 5.3 OPERATOR ACTION
6. ERRORS
 - 6.1 ERROR HALTS AND DESCRIPTION
 - 6.2 ERROR RECOVERY
7. RESTRICTIONS
 - 7.1 STARTING RESTRICTIONS
 - 7.2 OPERATING RESTRICTIONS
8. MISCELLANEOUS
 - 8.1 EXECUTION TIME
 - 8.2 STACK POINTER
 - 8.3 PASS COUNT
 - 8.4 ITERATIONS
 - 8.5 SPECIAL REGISTERS
 - 8.6 T BIT TRAPPING
 - 8.7 OSCILLOSCOPE SYNC POINTS
 - 8.8 CACHE CONTROL
9. PROGRAM DESCRIPTION
 - 9.1 DEKBB
10. LISTINGS
 - 10.1 DEKBB

1. ABSTRACT

DEKBA/B ARE PROGRAMS DESIGNED TO DETECT AND REPORT LOGIC FAULTS IN THE PDP 11/70 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

DEKBA 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 DEKBA.

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.

B. ADVANCED INSTRUCTION AND MISCELLANEOUS LOGIC TESTS

DEKBB 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 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

E01

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.

SEQ 0004

C. IMPORTANT NOTE

THE PROGRAM ANNOTATION IN DEKBA AND THE TYPED ERROR REPORTS IN DEKBB 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 CPU WITH OPERATORS CONSOLE
LA30 OR EQUIVALENT TERMINAL

2.2 STORAGE

DEKBA REQUIRES 16K TO LOAD AND RUN
DEKBB REQUIRES 16K TO LOAD AND 32K TO RUN

2.3 PRELIMINARY PROGRAMS

DEKBA REQUIRES THAT TWO INSTRUCTIONS WORK:
"BR" AND "HALT"

DEKBB REQUIRES THAT DEKBA RUN

3. LOADING PROCEDURE

3.1 METHOD

BOTH DEKBA AND DEKBB ARE LOADED FROM THE XXDP MEDIA.
REFER TO THE XXDP MANUAL FOR FURTHER INFORMATION.

F01

SEQ 0005

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE 5.1

4.2 STARTING ADDRESS

200

4.3 PROGRAM AND OPERATOR ACTION

A. DEKBA

1. LOAD PROGRAM INTO MEMORY (SEE SECTION 3)
2. LOAD ADDRESS 200
3. PRESS START
5. THE PROGRAM WILL LOOP UNTIL THE HALT SWITCH IS PRESSED

B. DEKBB

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. DEKBA

NONE

B. DEKBB

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<3>=1 ... SKIP MEMORY MANAGEMENT TESTING
 SW<2>=1 ... SKIP CACHE TESTING
 SW<1>=1 ... SKIP MA BOX TESTING
 SW<0>=1 ... SKIP OPERATOR INTERVENTION TESTING

5.2 SUBROUTINE ABSTRACTS

A. DEKBA

SEE 5.2.4 AND 5.2.5

B. DEKBB

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 ITERATION, 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

H01

SEQ 0007

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 \$STNM IN THE "TEST ADDRESS TABLE". IF THEY DON'T COMPARE, A MESSAGE IS TYPED, INDICATING THAT A TEST WAS SKIPPED

5.2.9 PERIPHERAL DETERMINATOR AND INTERRUPT ENABLE ROUTINES

SEQ 0008

5.2.9.1 PERIPHERAL DETERMINATOR

THIS ROUTINE IS EXECUTED, IN LINE, BETWEEN TESTS 32 AND 33 OF DEKBB. IT CHECKS THE SYSTEM TO DETERMINE IF ONE OF FOUR PERIPHERALS IS AVAILABLE (RS04, RPO4, 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 "INTSSU" (INTERRUPT 5 SUBROUTINE) AND \$KWILL/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 DEKBB 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. DEKBA

EVERY ERROR IN DEKBA 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. DEKBB

NONE OF THE ERRORS IN DEKBB 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. DEKBA

ERROR RECOVERY IS STRICTLY BY USER INTERVENTION.

B. DEKBB

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. DEKBA

NONE

B. DEKBB

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 RPO4, THIS REPRESENTS A NON-STANDARD CONFIGURATION AND LOCATION 1244 SHOULD BE CHANGED FROM 176700 TO 176714.

7.2 OPERATING RESTRICTIONS

A. DEKBA

NONE

B. DEKBB

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 LOCATIONS 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. DEKBA

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

B. DEKBB

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. DEKBA

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, 10000(8) PASSES ARE MADE FOR EACH END OF PASS MESSAGE. THE PASS COUNT IS DISPLAYED IN THE DATA LIGHTS.

B. DEKBB

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

8.4 ITERATIONS

A. DEKBA

NONE

B. DEKBB

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

8.5 SPECIAL REGISTERS

A. DEKBA

R0 IS RESERVED FOR THE TEST NUMBER.

B. DEKBB

NONE

8.6 T BIT TRAPPING

A. DEKBA

NONE

B. DEKBB

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.

MO1

8.7 OSCILLOSCOPE SYNC POINTS

SEQ 0012

A. DEKBA

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. DEKBB

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

9.1 DEKBB

DOCUMENT

PDP 11/70 CPU DIAGNOSTIC PART 2

COPYRIGHT 1975
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASS. 01754

TABLE OF CONTENTS

41	BASIC DEFINITIONS
166	CACHE REGISTER DEFINITIONS
177	CPU REGISTER DEFINITIONS
191	MEMORY MANAGEMENT DEFINITIONS
340	UNIBUS MAP REGISTER DEFINITIONS
434	TRAP CATCHER
441	STARTING ADDRESS(ES)
447	ACT11 HOOKS
473	COMMON TAGS
547	ERROR POINTER TABLE
3188	PERIPHERAL DETERMINATOR & INTERRUPT ENABLE ROUTINES
5295	MEMORY MANAGEMENT SETUP
6074	END OF PASS ROUTINE
6146	SPURIOUS ERROR HANDLER
6211	SCOPE HANDLER ROUTINE
6277	ERROR HANDLER ROUTINE
6328	ERROR MESSAGE TYPEOUT ROUTINE
6371	STACK LIMIT TEST TYPE OUT ROUTINE
6489	MONITOR RESTORE ROUTINE
6527	CHECK TEST SEQUENCE ROUTINE
6588	TYPE ROUTINE
6661	BINARY TO OCTAL (ASCII) AND TYPE

TABLE OF CONTENTS

6739	CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
6807	TRAP DECODER
6822	TRAP TABLE
6837	POWER DOWN AND UP ROUTINES

17 COPYRIGHT (C) AUGUST 21, 1975
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).

41

BASIC DEFINITIONS

43 INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***

57 MISCELLANEOUS DEFINITIONS

63 GENERAL PURPOSE REGISTER DEFINITIONS

84 PRIORITY LEVEL DEFINITIONS

94 "SWITCH REGISTER" SWITCH DEFINITIONS

122 DATA BIT DEFINITIONS (BIT00 TO BIT15)

150 BASIC "CPU" TRAP VECTOR ADDRESSES

166

CACHE REGISTER DEFINITIONS

177

CPU REGISTER DEFINITIONS

191

MEMORY MANAGEMENT DEFINITIONS

194 MEMORY MANAGEMENT STATUS REGISTER ADDRESSES

205 USER "I" PAGE DESCRIPTOR REGISTERS

216 USER "D" PAGE DESCRIPTOR REGISTORS

227 USER "I" PAGE ADDRESS REGISTERS

238 USER "D" PAGE ADDRESS REGISTERS

- 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 LOGICAL
 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(AFIRD4(1)*AFIRO5(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 EB 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 BRAB04 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 DROO IS STUCK HIGH OR NOT GETTING THRU TO
RACK E64(C1) OR RACK E64 IS BAD (INPUT C1 FAILED HIGH)
THE MULTIPPLICAND WILL BE MULTPLIED BY 177777.

IF EITHER GRAD DROO IS STUCK LOW OR NOT GETTING THRU TO
RACK E64(C1) OR RACK E64 IS BAD (INPUT C1 FAILED LOW)
THE MULTIPPLICAND WILL BE MULTPLIED BY ZERO.

IF GRAJ SC=0 IS NOT GETTING TO RACK E50(C1) AND RACK E50
IS BAD (INPUT C1 FAILED LOW) THE MULTIPPLICAND WILL ONLY
BE MULTPLIED 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

222C 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*DRO(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 10000000000 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*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 MTP1 AND MTPD.

ROM FLOW-45, 151, 146, 205

2627 TEST 16 MTP*DMI

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*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

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

3
1032 -

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:
PJP.00, BRK.20, OR SER.00.
PJP.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 15-2.

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 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.

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 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.
- 4039 TEST 45 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.
- 4058 TEST 46 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.
- 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 GOREG 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 LOCALLOCATION 15.

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.

5289

THE NEXT 6 TESTS USE MEMORY MANAGEMENT. IF THESE TESTS ARE NOT TO
BE EXECUTED, THEN SWITCH 3 SHOULD BE PUT ON.
IT SHOULD BE REMEMBERED THAT A FAILURE IN THESE TESTS
COULD BE DUE TO MEMORY MANAGEMENT.

5295

MEMORY MANAGEMENT SETUP

5297

THIS ROUTINE SETS UP THE KERNEL AND SUPERVISOR PAR'S AND
PDR'S TO MAP VIRTUAL ADDRESSES TO THE SAME PHYSICAL ADDRESSES.

5338 TEST 64 MEMORY MANAGEMENT ABORT

THIS TEST SETS UP PDR6 TO CAUSE A MEMORY MANAGEMENT ABORT.
IF TMCC AERF(1) DOES NOT GO HIGH IT WILL LOOK LIKE THE TRAP FAILED.

IF TMCC ABORT DOES NOT GO HIGH THE TEST WILL NOT TRAP AT ALL.
IF TMCB SEGT DOES NOT GO LOW A TRAP TO 4 WILL OCCUR.

IF TMCE CACHE BEND DOES NOT GO HIGH A TRAP TO 350 WILL OCCUR.

5383 TEST 65 MEMORY MANAGEMENT TRAP

THIS TEST ENSURES THAT THE MEMORY MANAGEMENT TRAP LOGIC WORKS.
IF TMCA HONOR SEGT DOES NOT GO LOW OR DOES NOT GET THRU
TO TMCB BRQ TRUE THE TRAP WILL NOT OCCUR.

IF TMCB SEGT DOES NOT GO LOW BEN13 WILL FAIL TO BRK.20.
THE FLOW WILL THEN GO TO RTI.60 WHICH MEANS AN ACKNOWLEDGE
IS NEVER GIVEN AND THE PROCESSOR WILL HANG UP.

AN INSTRUCTION IS THEN EXECUTED THAT CAUSES A MEMORY MANAGEMENT
TRAP ON THE SOURCE OPERAND BUT NOT ON THE DESTINATION.

5424 TEST 66 NON EXISTANT MEMORY ABORT

THIS TEST ENSURES THAT A NON EXISTANT MEMORY
REFERENCE FUNCTIONS PROPERLY.

IF TMCC AERF(1) DOES NOT GO HIGH IT WILL LOOK LIKE THE ABORT FAILED.
IF TMCC ABORT DOES NOT GO HIGH THE ABORT WILL STILL OCCUR,
BUT THE ERROR REGISTER WILL NOT BE LOADED.

5464 TEST 67 KT BEND

THIS TEST ENSURES THAT TMCE KT BEND GOES LOW ON AN ODD ADDRESS ERROR, SL RED, AND NEXM. THIS IS DONE BY EXECUTING AN INSTRUCTION FOR EACH OF THESE THREE CASES THAT ALSO CAUSES A KT ABORT. THE ABORT SHOULD NOT BE HONORED.

5521 TEST 70 SL REGISTER COMPARATOR TEST 2

THIS TEST IS THE SAME AS TEST 153 EXCEPT IT TESTS THE ADDRESSES ON EVERY PAGE. THIS IS DONE BY MAPPING I/O PAGE ADDRESSES TO MEMORY IN KERNEL MODE. THIS MAKES THE I/O PAGE INACCESSIBLE IN KERNEL MODE SO AN IOT INSTRUCTION IS USED TO RETURN TO SUPERVISOR MODE WHEN THE I/O PAGE IS NEEDED.

5645 TEST 71 PS RESTORE

THIS TEST ENSURES THAT BENG WORKS ON A PS RESTORE. THIS IS DONE BY SETTING THE STACK TO A NON RESIDENT PAGE AND DOING A TRAP INSTRUCTION. WHEN THE PROCESSOR TRYS TO PUSH THE OLD PSW ON THE STACK A KT ABORT WILL OCCUR. SINCE IT WAS A KERNEL R6 OPERATION THIS WILL CAUSE BEN13 TO GO TO STATE SER.00 WHICH WILL PUSH THE PSW AND PC INTO LOCATIONS 2 AND 0, AND THEN TRAP TO LOCATION 4. THE PSW IN LOCATION 2 SHOULD BE THE PSW BEFORE THE TRAP INSTRUCTION AND NOT THE PSW IN THE TRAP VECTOR.

5676

THE NEXT 4 TESTS USE CERTAIN FUNCTIONS IN THE CACHE. IF THESE TESTS ARE NOT TO BE EXECUTED, THEN SWITCH 2 SHOULD BE PUT ON.

IT SHOULD BE REMEMBERED THAT A FAILURE IN THESE TESTS COULD BE DUE TO THE CACHE.

5699 TEST 72 PARITY ERROR ABORT

THIS TEST ENSURES THAT A CACHE PARITY ERROR FLAG CAUSES AN ABORT. THIS IS DONE BY FORCING A PARITY ERROR ON AN EVEN WORD.

5753 TEST 73 PARITY ERROR TRAP

THIS TEST ENSURES THAT A PARITY TRAP FUNCTIONS PROPERLY. THIS IS DONE BY MAKING THE ODD WORD HAVE BAD PARITY. IF THE TRAP DOESN'T OCCUR THEN THE PROBLEM IS ON TMCA. IF A TRAP OCCURS TO THE WRONG VECTOR THE PROBLEM COULD BE ON TMCA OR UCB.

5798 TEST 74 MEM MGT AND PE TRAP PRIORITY ARBITRATION

THIS TEST ENSURES THAT THE ARBITRATION LOGIC WORKS FOR MEMORY MANAGEMENT AND PARITY ERROR TRAPS.

NOTE: BOTH SWITCHES 3 AND 2 MUST BE DOWN TO EXECUTE THIS TEST AND THE NEXT TEST.

5918 THE NEXT TEST USES THE MAPPING BOX AND THE CACHE TO GENERATE A PARITY ERROR ON THE UNIBUS. SWITCHES 3, 2 AND 1 MUST BE OFF TO EXECUTE THIS TEST.

5938 TEST 75 UNIBUS PARITY ERROR
THIS TEST MAKES A REFERENCE TO MEMORY THRU THE MAPPING BOX THAT WILL CAUSE A PARITY ERROR. IF THE ABORT DOESN'T HAPPEN THEN THE PROBLEM IS ON UBCB.
NOTE: MAP REGISTER 0 AND 1 ARE NOT USED INCASE THE PROGRAM IS RUNNING UNDER ACT11.

6011 TEST 76 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 ETYPOD 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 ADDRESS 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   N              ;;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

L03

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17

```
.TITLE PDP 11/70 CPU DIAGNOSTIC PART 2  
;*COPYRIGHT (C) AUGUST 21, 1975  
;*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).  
*  
$TN=1      $SWR=177400
```

000001
177400

18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40

.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	DISABLE MEMORY MANAGEMENT TESTS
2	DISABLE CACHE TESTS
1	DISABLE MAP BOX TEST
0	DISABLE OPERATOR INTERVENTION TEST

```

41
42      .SBTTL BASIC DEFINITIONS
43
44      ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
45      001100  STACK= 1100          ;; FIRST ADDRESS OF THE STACK
46      001100  KERSTK= STACK      ;; KERNEL STACK
47      000700  SUPSTK= STACK-200  ;; SUPERVISOR STACK
48      000600  USESTK= STACK-300  ;; USER STACK
49      .EQUIV  EMT,ERROR          ;; BASIC DEFINITION OF ERROR CALL
50      .EQUIV  IOT,SCOPE         ;; BASIC DEFINITION OF SCOPE CALL
51      177776  PS= 177776        ;; PROCESSOR STATUS WORD
52      .EQUIV  PS,PSW
53      177774  STKLMT= 177774     ;; STACK LIMIT REGISTER
54      177772  PIRQ= 177772      ;; PROGRAM INTERRUPT REQUEST REGISTER
55      177570  SWR= 177570       ;; SWITCH REGISTER
56      177570  DISPLAY=SWR
57
58      ;*MISCELLANEOUS DEFINITIONS
59      000011  HT= 11             ;; CODE FOR HORIZONTAL TAB
60      000012  LF= 12             ;; CODE LINE FEED
61      000015  CR= 15             ;; CODE CARRIAGE RETURN
62      000200  CRLF= 200         ;; CODE FOR CARRIAGE RETURN-LINE FEED
63
64      ;*GENERAL PURPOSE REGISTER DEFINITIONS
65      000000  R0= %0             ;; GENERAL REGISTER
66      000001  R1= %1             ;; GENERAL REGISTER
67      000002  R2= %2             ;; GENERAL REGISTER
68      000003  R3= %3             ;; GENERAL REGISTER
69      000004  R4= %4             ;; GENERAL REGISTER
70      000005  R5= %5             ;; GENERAL REGISTER
71      000006  R6= %6             ;; GENERAL REGISTER
72      000007  R7= %7             ;; GENERAL REGISTER
73      .EQUIV  R0,R10            ;; GENERAL REGISTER
74      .EQUIV  R1,R11            ;; GENERAL REGISTER
75      .EQUIV  R2,R12            ;; GENERAL REGISTER
76      .EQUIV  R3,R13            ;; GENERAL REGISTER
77      .EQUIV  R4,R14            ;; GENERAL REGISTER
78      .EQUIV  R5,R15            ;; GENERAL REGISTER
79      .EQUIV  R6,SP             ;; STACK POINTER
80      .EQUIV  SP,KSP            ;; KERNEL STACK POINTER
81      .EQUIV  SP,SSP            ;; SUPERVISOR STACK POINTER
82      .EQUIV  SP,USP            ;; USER STACK POINTER
83      .EQUIV  R7,PC             ;; PROGRAM COUNTER
84
85      ;*PRIORITY LEVEL DEFINITIONS
86      000000  PR0= 0             ;; PRIORITY LEVEL 0
87      000040  PR1= 40            ;; PRIORITY LEVEL 1
88      000100  PR2= 100           ;; PRIORITY LEVEL 2
89      000140  PR3= 140           ;; PRIORITY LEVEL 3
90      000200  PR4= 200           ;; PRIORITY LEVEL 4
91      000240  PR5= 240           ;; PRIORITY LEVEL 5
92      000300  PR6= 300           ;; PRIORITY LEVEL 6
93      000340  PR7= 340           ;; PRIORITY LEVEL 7
94
95      ;*"SWITCH REGISTER" SWITCH DEFINITIONS
96      100000  SW15= 100000
    
```

97	040000	SW14=	40000
98	020000	SW13=	20000
99	010000	SW12=	10000
100	004000	SW11=	4000
101	002000	SW10=	2000
102	001000	SW09=	1000
103	000400	SW08=	400
104	000200	SW07=	200
105	000100	SW06=	100
106	000040	SW05=	40
107	000020	SW04=	20
108	000010	SW03=	10
109	000004	SW02=	4
110	000002	SW01=	2
111	000001	SW00=	1
112		.EQUIV	SW09, SW9
113		.EQUIV	SW08, SW8
114		.EQUIV	SW07, SW7
115		.EQUIV	SW06, SW6
116		.EQUIV	SW05, SW5
117		.EQUIV	SW04, SW4
118		.EQUIV	SW03, SW3
119		.EQUIV	SW02, SW2
120		.EQUIV	SW01, SW1
121		.EQUIV	SW00, SW0

122			
123		.*DATA	BIT DEFINITIONS (BIT00 TO BIT15)
124	100000	BIT15=	100000
125	040000	BIT14=	40000
126	020000	BIT13=	20000
127	010000	BIT12=	10000
128	004000	BIT11=	4000
129	002000	BIT10=	2000
130	001000	BIT09=	1000
131	000400	BIT08=	400
132	000200	BIT07=	200
133	000100	BIT06=	100
134	000040	BIT05=	40
135	000020	BIT04=	20
136	000010	BIT03=	10
137	000004	BIT02=	4
138	000002	BIT01=	2
139	000001	BIT00=	1

140		.EQUIV	BIT09, BIT9
141		.EQUIV	BIT08, BIT8
142		.EQUIV	BIT07, BIT7
143		.EQUIV	BIT06, BIT6
144		.EQUIV	BIT05, BIT5
145		.EQUIV	BIT04, BIT4
146		.EQUIV	BIT03, BIT3
147		.EQUIV	BIT02, BIT2
148		.EQUIV	BIT01, BIT1
149		.EQUIV	BIT00, BIT0

150			
151		.*BASIC	"CPU" TRAP VECTOR ADDRESSES
152	000004	ERRVEC=	4 ;; TIME OUT AND OTHER ERRORS

153	000010	RESVEC= 10	:: RESERVED AND ILLEGAL INSTRUCTIONS
154	000014	TBITVEC=14	:: "T" BIT
155	000014	TRTVEC= 14	:: TRACE TRAP
156	000014	BPTVEC= 14	:: BREAKPOINT TRAP (BPT)
157	000020	IOTVEC= 20	:: INPUT/OUTPUT TRAP (IOT) **SCOPE**
158	000024	PWRVEC= 24	:: POWER FAIL
159	000030	EMTVEC= 30	:: EMULATOR TRAP (EMT) **ERROR**
160	000034	TRAPVEC=34	:: "TRAP" TRAP
161	000060	TKVEC= 60	:: TTY KEYBOARD VECTOR
162	000064	TFVEC= 64	:: TTY PRINTER VECTOR
163	000114	CACHVEC=114	:: CACHE ERROR INTERRUPT VECTOR
164	000240	PIRQVEC=240	:: PROGRAM INTERRUPT REQUEST VECTOR
165	000250	MMVEC= 250	:: MEMORY MANAGEMENT VECTOR

.SBTTL CACHE REGISTER DEFINITIONS

170	!77740	LOADRS = 177740	:: LOWER 16 BITS OF ADDRESS THAT CAUSED ERROR
171	177742	HIADRS = 177742	:: UPPER SIX BITS OF ADDRESS THAT CAUSED ERROR
172	177744	MEMERR = 177744	:: CACHE ERROR REGISTER
173	177746	CONTRL = 177746	:: MEMORY CONTROL REGISTER
174	177750	MAINT = 177750	:: MEMORY MAINTENANCE REGISTER
175	177752	HITMIS = 177752	:: HIT MISS REGISTER "1" IMPLIES HIT IN CACHE

.SBTTL CPU REGISTER DEFINITIONS

181	177760	SIZELO = 177760	:: MEMORY SIZE REGISTER NUMBER TO PUT INTO A PAR
182			:: TO GET TO THE LAST 32 WORDS OF MEMORY
183	177762	SIZEHI = 177762	:: HIGH SIZE REGISTER, RESERVED FOR FUTURE USE
184			:: CURRENTLY ALL ZERO
185	177764	SYSTID = 177764	:: SYSTEM ID REGISTER
186	177766	CPUERR = 177766	:: CPU ERROR REGISTER HOLDS CONDITION THAT CAUSED
187			:: THE TRAP TO ERRVEC (000004)

.SBTTL MEMORY MANAGEMENT DEFINITIONS

;*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES

197	177572	MMR0= 177572	
198	177574	MMR1= 177574	
199	177576	MMR2= 177576	
200	172516	MMR3= 172516	
201		.EQUIV MMR0,SR0	
202		.EQUIV MMR1,SR1	
203		.EQUIV MMR2,SR2	
204		.EQUIV MMR3,SR3	

;*USER "I" PAGE DESCRIPTOR REGISTERS

207			
209	177600	UIPDRO= 177600	

```

209      177602      UIPDR1= 177602
210      177604      UIPDR2= 177604
211      177606      UIPDR3= 177606
212      177610      UIPDR4= 177610
213      177612      UIPDR5= 177612
214      177614      UIPDR6= 177614
215      177616      UIPDR7= 177616
216
217      ;*USER "D" PAGE DESCRIPTOR REGISTORS
218
219      177620      UDPDR0= 177620
220      177622      UDPDR1= 177622
221      177624      UDPDR2= 177624
222      177626      UDPDR3= 177626
223      177630      UDPDR4= 177630
224      177632      UDPDR5= 177632
225      177634      UDPDR6= 177634
226      177636      UDPDR7= 177636
227
228      ;*USER "I" PAGE ADDRESS REGISTERS
229
230      177640      UIPAR0= 177640
231      177642      UIPAR1= 177642
232      177644      UIPAR2= 177644
233      177646      UIPAR3= 177646
234      177650      UIPAR4= 177650
235      177652      UIPAR5= 177652
236      177654      UIPAR6= 177654
237      177656      UIPAR7= 177656
238
239      ;*USER "D" PAGE ADDRESS REGISTERS
240
241      177660      UDPAR0= 177660
242      177662      UDPAR1= 177662
243      177664      UDPAR2= 177664
244      177666      UDPAR3= 177666
245      177670      UDPAR4= 177670
246      177672      UDPAR5= 177672
247      177674      UDPAR6= 177674
248      177676      UDPAR7= 177676
249
250      ;*SUPERVISOR "I" PAGE DESCRIPTOR REGISTERS
251
252      172200      SIPDR0= 172200
253      172202      SIPDR1= 172202
254      172204      SIPDR2= 172204
255      172206      SIPDR3= 172206
256      172210      SIPDR4= 172210
257      172212      SIPDR5= 172212
258      172214      SIPDR6= 172214
259      172216      SIPDR7= 172216
260
261      ;*SUPERVISOR "D" PAGE DESCRIPTOR REGISTERS
262
263      172220      SDPDR0= 172220
264      172222      SDPDR1= 172222

```

265	172224	SOPDR2= 172224
266	172226	SOPDR3= 172226
267	172230	SOPDR4= 172230
268	172232	SOPDR5= 172232
269	172234	SOPDR6= 172234
270	172236	SOPDR7= 172236

;*SUPERVISOR "I" PAGE ADDRESS REGISTERS

274	172240	SIPAR0= 172240
275	172242	SIPAR1= 172242
276	172244	SIPAR2= 172244
277	172246	SIPAR3= 172246
278	172250	SIPAR4= 172250
279	172252	SIPAR5= 172252
280	172254	SIPAR6= 172254
281	172256	SIPAR7= 172256

;*SUPERVISOR "D" PAGE ADDRESS REGISTERS

285	172260	SDPAR0= 172260
286	172262	SDPAR1= 172262
287	172264	SDPAR2= 172264
288	172266	SDPAR3= 172266
289	172270	SDPAR4= 172270
290	172272	SDPAR5= 172272
291	172274	SDPAR6= 172274
292	172276	SDPAR7= 172276

;*KERNEL "I" PAGE DESCRIPTOR REGISTERS

296	172300	KIPDR0= 172300
297	172302	KIPDR1= 172302
298	172304	KIPDR2= 172304
299	172306	KIPDR3= 172306
300	172310	KIPDR4= 172310
301	172312	KIPDR5= 172312
302	172314	KIPDR6= 172314
303	172316	KIPDR7= 172316

;*KERNEL "D" PAGE DESCRIPTOR REGISTERS

307	172320	KDPDR0= 172320
308	172322	KDPDR1= 172322
309	172324	KDPDR2= 172324
310	172326	KDPDR3= 172326
311	172330	KDPDR4= 172330
312	172332	KDPDR5= 172332
313	172334	KDPDR6= 172334
314	172336	KDPDR7= 172336

;*KERNEL "I" PAGE ADDRESS REGISTERS

318	172340	KIPAR0= 172340
319	172342	KIPAR1= 172342
320	172344	KIPAR2= 172344

321	172346	KIPAR3= 172346
322	172350	KIPAR4= 172350
323	172352	KIPAR5= 172352
324	172354	KIPAR6= 172354
325	172356	KIPAR7= 172356

;*KERNEL "D" PAGE ADDRESS REGISTERS

329	172360	KDPAR0= 172360
330	172362	KDPAR1= 172362
331	172364	KDPAR2= 172364
332	172366	KDPAR3= 172366
333	172370	KDPAR4= 172370
334	172372	KDPAR5= 172372
335	172374	KDPAR6= 172374
336	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'

348	170200	MAPL00 = 170200
349	170202	MAPH00 = 170202
350	170204	MAPL01 = 170204
351	170206	MAPH01 = 170206
352	170210	MAPL02 = 170210
353	170212	MAPH02 = 170212
354	170214	MAPL03 = 170214
355	170216	MAPH03 = 170216
356	170220	MAPL04 = 170220
357	170222	MAPH04 = 170222
358	170224	MAPL05 = 170224
359	170226	MAPH05 = 170226
360	170230	MAPL06 = 170230
361	170232	MAPH06 = 170232
362	170234	MAPL07 = 170234
363	170236	MAPH07 = 170236
364	170240	MAPL10 = 170240
365	170242	MAPH10 = 170242
366	170244	MAPL11 = 170244
367	170246	MAPH11 = 170246
368	170250	MAPL12 = 170250
369	170252	MAPH12 = 170252
370	170254	MAPL13 = 170254
371	170256	MAPH13 = 170256
372	170260	MAPL14 = 170260
373	170262	MAPH14 = 170262
374	170264	MAPL15 = 170264
375	170266	MAPH15 = 170266
376	170270	MAPL16 = 170270

377	170272	MAPH16 = 170272
378	170274	MAPL17 = 170274
379	170276	MAPH17 = 170276
380	170300	MAPL20 = 170300
381	170302	MAPH20 = 170302
382	170304	MAPL21 = 170304
383	170306	MAPH21 = 170306
384	170310	MAPL22 = 170310
385	170312	MAPH22 = 170312
386	170314	MAPL23 = 170314
387	170316	MAPH23 = 170316
388	170320	MAPL24 = 170320
389	170320	MAPH24 = 170320
390	170324	MAPL25 = 170324
391	170326	MAPH25 = 170326
392	170330	MAPL26 = 170330
393	170332	MAPH26 = 170332
394	170334	MAPL27 = 170334
395	170336	MAPH27 = 170336
396	170340	MAPL30 = 170340
397	170342	MAPH30 = 170342
398	170344	MAPL31 = 170344
399	170346	MAPH31 = 170346
400	170350	MAPL32 = 170350
401	170352	MAPH32 = 170352
402	170354	MAPL33 = 170354
403	170356	MAPH33 = 170356
404	170360	MAPL34 = 170360
405	170362	MAPH34 = 170362
406	170364	MAPL35 = 170364
407	170366	MAPH35 = 170366
408	170370	MAPL36 = 170370
409	170372	MAPH36 = 170372
410	170374	MAPL37 = 170374
411	170376	MAPH37 = 170376
412		.EQUIV MAPL00, MAPL0
413		.EQUIV MAPH00, MAPH0
414		.EQUIV MAPL01, MAPL1
415		.EQUIV MAPH01, MAPH1
416		.EQUIV MAPL02, MAPL2
417		.EQUIV MAPH02, MAPH2
418		.EQUIV MAPL03, MAPL3
419		.EQUIV MAPH03, MAPH3
420		.EQUIV MAPL04, MAPL4
421		.EQUIV MAPH04, MAPH4
422		.EQUIV MAPL05, MAPL5
423		.EQUIV MAPH05, MAPH5
424		.EQUIV MAPL06, MAPL6
425		.EQUIV MAPH06, MAPH6
426		.EQUIV MAPL07, MAPL7
427		.EQUIV MAPH07, MAPH7
428		
429		
430		
431		
432	172544	PLKC=172544

```

433
434
435          .SBTTL  TRAP CATCHER
436
437          000000          .=0
438          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
439          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
440          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
441
442          .SBTTL  STARTING ADDRESS(ES)
443          000200          .=200
444
445          000200  000137  004740          JMP      @#START          ;;JUMP TO STARTING ADDRESS OF PROGRAM
446          ;;*****
447
448          .SBTTL  ACT11 HOOKS
449
450          ;*THE FOLLOWING LOCATIONS ARE SETUP TO BE USED WITH ACT11
451          ;*
452          ;*LOCATION 46 WILL CONTAIN THE ADDRESS OF THE LOGICAL
453          ;*END OF THE PROGRAM.
454          ;*LOCATION 52 IS USED TO SPECIFY PROGRAM OPERATING REQUIREMENTS
455          ;*AND/OR RESTRICTIONS. THIS IS ACCOMPLISHED BY SETTING VARIOUS BITS
456          ;*TO A ONE OR A ZERO. THE BITS USED AND THERE MEANING ARE:
457          ;*
458          ;*      BIT 15=1 PROGRAM SHOULD BE POWER FAILED WHILE RUNNING
459          ;*      =0 NO POWER FAIL DESIRED
460          ;*
461          ;*      BIT 14=1 PROGRAM RUN TIME IS MEMORY SIZE DEPENDENT
462          ;*      =0 RUN TIME IS NOT MEMORY SIZE DEPENDENT
463          ;*
464          ;*      BITS 13-0 MUST BE ZERO'S
465
466          000204          $SVPC=.          ;;SAVE LOCATION COUNTER
467          000046          .=46          ;;SET LOCATION COUNTER
468          000046  036160          .WORD  $ENDAD          ;;SET LOC.46 TO ADDRESS $ENDAD
469          000052          .=52          ;;SET LOCATION COUNTER
470          000052  000000          .WORD  0          ;;SET LOC.52 TO ZERO
471          000204          .=52          ;;RESTORE LOCATION COUNTER

```

```

472 ;*****
473
474 .SBTTL COMMON TAGS
475
476 ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
477 ;*USED IN THE PROGRAM.
478
479         001100             .=1100
480
481 001100 $CMTAG:           ; START OF COMMON TAGS
482 001100 000000 $PASS: .WORD 0 ; CONTAINS PASS COUNT
483 001102 000 $STSTM: .BYTE 0 ; CONTAINS THE TEST NUMBER
484 001103 000 $SERFLG: .BYTE 0 ; CONTAINS ERROR FLAG
485 001104 000000 $ICNT: .WORD 0 ; CONTAINS SUBTEST ITERATION COUNT
486 001106 000000 $LPADR: .WORD 0 ; CONTAINS SCOPE LOOP
487 001110 000000 $LPERR: .WORD 0 ; CONTAINS SCOPE RETURN FOR ERRORS
488 001112 000000 $ERTTL: .WORD 0 ; CONTAINS TOTAL ERRORS DETECTED
489 001114 000 $ITEMB: .BYTE 0 ; CONTAINS ITEM CONTROL BYTE
490 001115 001 $ERMAX: .BYTE 1 ; CONTAINS MAX. ERRORS PER TEST
491 001116 000000 $ERRPC: .WORD 0 ; CONTAINS PC OF LAST ERROR INSTRUCTION
492 001120 000000 $GDADR: .WORD 0 ; CONTAINS OF 'GOOD' DATA
493 001122 000000 $BDADR: .WORD 0 ; CONTAINS OF 'BAD' DATA
494 001124 000000 $GDDAT: .WORD 0 ; CONTAINS 'GOOD' DATA
495 001126 000000 $BDDAT: .WORD 0 ; CONTAINS 'BAD' DATA
496 001130 000000 000000 000000 .WORD 0,0,0 ; RESERVED--NOT TO BE USED
497 001136 177560 $TKS: 177560 ; TTY KBD STATUS
498 001140 177562 $TKB: 177562 ; TTY KBD BUFFER
499 001142 177564 $TPS: 177564 ; TTY PRINTER STATUS REG.
500 001144 177566 $TPB: 177566 ; TTY PRINTER BUFFER REG.
501 001146 000 $NULL: .BYTE 0 ; CONTAINS NULL CHARACTER FOR FILLS
502 001147 002 $FILLS: .BYTE 2 ; CONTAINS # OF FILLER CHARACTERS REQUIRED
503 001150 012 $FILLC: .BYTE 12 ; INSERT FILL CHARS. AFTER A "LINE FEED"
504 001151 000 $TPFLG: .BYTE 0 ; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
505 001152 000000 $REGAD: .WORD 0 ; CONTAINS THE FROM
506 ; WHICH ($REGO) WAS OBTAINED
507 001154 000000 $REGO: .WORD 0 ; CONTAINS (($REGAD)+0)
508 001156 000000 $REG1: .WORD 0 ; CONTAINS (($REGAD)+2)
509 001160 000000 $REG2: .WORD 0 ; CONTAINS (($REGAD)+4)
510 001162 000000 $TMPO: .WORD 0 ; USER DEFINED
511 001164 000000 $TMP1: .WORD 0 ; USER DEFINED
512 001166 000000 $TMP2: .WORD 0 ; USER DEFINED
513 001170 000000 $TMP3: .WORD 0 ; USER DEFINED
514 001172 000000 $TIMES: 0 ; MAX. NUMBER OF ITERATIONS
515 001174 000000 $ESCAPE: 0 ; ESCAPE ON ERROR
516 001176 177607 000377 $BELL: .ASCIZ <207><377><377> ; CODE FOR BELL
517 001202 077 $QUES: .ASCII /?/ ; QUESTION MARK
518 001203 015 $CRLF: .ASCII <15> ; CARRIAGE RETURN
519 001204 000012 $LF: .ASCIZ <12> ; LINE FEED
520 001206 000000 $ERPSW: .WORD 0 ; ERROR PSW
521 001210 000000 $STSTM: .WORD 0 ; TEST NUMBER STORAGE
522 001212 000100 $PR2: .WORD PR2 ; PRIORITY LEVEL 2
523 001214 000240 $PR5: .WORD PR5 ; PRIORITY LEVEL 5
524 001216 000000 $EPIRQ: .WORD 0 ; ERROR PIRQ
525 001220 000000 E1STKLM: .WORD 0 ; STACK LIMIT REGISTER ERROR 1
526 001222 000000 E2STKLM: .WORD 0 ; STACK LIMIT REGISTER ERROR 2
527 001224 000000 INTER5: .WORD 0 ; ADDRESS OF BR5 INTER SUBROUTINE

```


528	001226	000000	INTSVEC: .WORD	0	;BR 5 INTERRUPT VECTOR
529	001230	000000	INTSST: .WORD	0	;BR 5 STATUS REG
530	001232	000000	INTER6: .WORD	0	;ADDRESS OF BR6 INTERRUPT SUBROUTINE
531	001234	000000	INT6VEC: .WORD	0	;BR 6 INTERRUPT VECTOR
532	001236	000000	INT6ST: .WORD	0	;BR 6 STATUS REG
533	001240	172040	RSCS1: .WORD	172040	;ADDRESS OF RS STATUS REGISTER
534	001242	000204	RSVEC: .WORD	204	;ADDRESS OF RS VECTOR
535	001244	176700	RPCS1: .WORD	176700	;ADDRESS OF RP STATUS REGISTER
536	001246	000254	RPVEC: .WORD	254	;ADDRESS OF RP VECTOR
537	001250	177404	RKCS1: .WORD	177404	;ADDRESS OF RK STATUS REGISTER
538	001252	000220	RKVEC: .WORD	220	;ADDRESS OF RK VECTOR
539	001254	172440	TMCS1: .WORD	172440	;ADDRESS OF TM STATUS REG
540	001256	000224	TMVEC: .WORD	224	;ADDRESS OF TM VECTOR
541	001260	177546	LKSTAT: .WORD	177546	;ADDRESS OF LINE CLOCK STATUS REGISTER
542	001262	000100	LKVEC: .WORD	100	;ADDRESS OF LINE CLOCK VECTOR
543	001264	172540	PLKSTAT: .WORD	172540	;ADDRESS OF PROG LINE CLOCK STATUS REG
544	001266	000104	PLKVEC: .WORD	104	;ADDRESS OF PROG LINE CLOCK VECTOR
545	001270	000000	NEXTTST: .WORD	0	;ADDRESS OF NEXT TEST

```

546 ;*****
547
548 .SBTTL ERROR POINTER TABLE
549
550 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
551 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
552 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
553 ;*NOTE1: IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
554 ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
555
556 ;* EM ;:POINTS TO THE ERROR MESSAGE
557 ;* DH ;:POINTS TO THE DATA HEADER
558 ;* DT ;:POINTS TO THE DATA
559 ;* DF ;:POINTS TO THE DATA FORMAT
560
561
562 001272 $ERRTB:
563
564 001272 042006 ITEM1: EM1 ;EITHER SPL FAILED OR BITS STUCK
565 001274 042063 DH1 ;ERRORPC SPL 5 PSW SPL 2 PSW TEST NUMBER
566 ; EXPECT ACTUAL EXPECT ACTUAL
567 001276 042202 DT1 ;$ERRPC,$PRS,$ERPSW,$PR2,$TMPO
568 001300 042220 ITEM2: EM2 ;PR5 LOADS OK BUT PR2 DOESN'T
569 001302 042063 DH1
570 001304 042202 DT1
571 001306 042241 ITEM3: EM3 ;PR2 LOADS OK BUT PR5 DOESN'T
572 001310 042063 DH1
573 001312 042202 DT1
574 001314 042266 ITEM4: EM4 ;FORK A FAILED .0 D30.00
575 001316 042331 DH4 ;ERRORPC TEST NUMBER
576 001320 042356 DT4 ;$ERRPC,$$STSTNM
577 001322 042364 ITEMS: EM5 ;FORK A FAILED TO RSD.02
578 001324 042331 DH4
579 001326 042356 DT4
580 001330 042452 ITEM6: EM6 ;RESET DID NOT WORK
581 001332 042331 DH4
582 001334 042356 DT4
583 001336 042506 ITEM7: EM7 ;FORK A FAILED INTO TRP.02
584 001340 042331 DH4
585 001342 042356 DT4
586 001344 042522 ITEM10: EM10 ;FORK A FAILED INTO WAT.00
587 001346 042331 DH4
588 001350 042356 DT4
589 001352 042536 ITEM11: EM11 ;FORK A FAILED TO MTP.00
590 001354 042331 DH4
591 001356 042356 DT4
592 001360 042575 ITEM12: EM12 ;FORK A FAILED TO MFP.80
593 001362 042331 DH4
594 001364 042356 DT4
595 001366 042634 ITEM13: EM13 ;PCB DID NOT LOAD FROM R5
596 001370 042331 DH4
597 001372 042356 DT4
598 001374 042665 ITEM14: EM14 ;MARK DID NOT LOAD SP PROPERLY
599 001376 042716 DH14 ;ERRORPC SP TEST NUMBER
600 ; EXPECT ACTUAL
601 001400 043000 DT14 ;$ERRPC,$REG1,$REG0,$$STSTNM

```

602	001402	043012	ITEM15: EM15	;R5 DID NOT LOAD PROPERLY
603	001404	043043	DH15	;ERRORPC R5 TEST NUMBER
604	001406	043000	DT14	
605	001410	043124	ITEM16: EM16	;FORK A FAILED TO RSD.00
606	001412	042331	DH4	
607	001414	042356	DT4	
608	001416	043161	ITEM17: EM17	;FORK A FAILED TO D67.01
609	001420	042331	DH4	
610	001422	042356	DT4	
611	001424	043245	ITEM20: EM20	;R1 SHIFTED RIGHT INSTEAD OF LEFT
612	001426	042331	DH4	
613	001430	042356	DT4	
614	001432	043337	ITEM21: EM21	;R1 DID NOT SHIFT
615	001434	042331	DH4	
616	001436	042356	DT4	
617	001440	043360	ITEM22: EM22	;R1 SHIFTED BUT CARRY DID NOT SET
618	001442	042331	DH4	
619	001444	042356	DT4	
620	001446	043456	ITEM23: EM23	;R1 SHIFTED LEFT INSTEAD OF RIGHT
621	001450	042331	DH4	
622	001452	042356	DT4	
623	001454	043547	ITEM24: EM24	;SHIFT RIGHT DID NOT SIGN FILL
624	001456	042331	DH4	
625	001460	042356	DT4	
626	001462	043603	ITEM25: EM25	;R1 SHIFTED BUT DON'T KNOW WHERE
627	001464	043636	DH25	;ERRORPC R5 TEST NUMBER
628				; EXPECT ACTUAL
629	001466	043712	DT25	;SERRPC,\$TMPI,\$REGI,\$\$STNM
630	001470	043724	ITEM26: EM26	;SHIFT OK BUT CARRY DID NOT LOAD
631	001472	042331	DH4	
632	001474	042356	DT4	
633	001476	043765	ITEM27: EM27	;ASH.20 DID NOT LOAD CC'S CORRECTLY
634	001500	044774	DH42	;ERRORPC PSW TEST NUMBER
635				; EXPECT ACTUAL
636	001502	045052	DT42	;SERRPC,\$TMPO,\$ERPSW,\$\$STNM
637	001504	044030	ITEM30: EM30	;R1 SHIFTED WITH A SHIFT COUNT OF 0
638	001506	042331	DH4	
639	001510	042356	DT4	
640	001512	044066	ITEM31: EM31	;ASH.40 DID NOT LOAD CC'S CORRECTLY
641	001514	044774	DH42	
642	001516	045052	DT42	
643	001520	044131	ITEM32: EM32	;FORK A FAILED TO RSD.00
644	001522	042331	DH4	
645	001524	042356	DT4	
646	001526	044176	ITEM33: EM33	;STATE ASH.00 FAILED
647	001530	042331	DH4	
648	001532	042356	DT4	
649	001534	044214	ITEM34: EM34	;FORK B FAILED INTO MUL.00
650	001536	042331	DH4	
651	001540	042356	DT4	
652	001542	044247	ITEM35: EM35	;FORK B FAILED TO RSD.00
653	001544	042331	DH4	
654	001546	042356	DT4	
655	001550	044375	ITEM36: EM36	;FORK A FAILED TO RSD.00
656	001552	042331	DH4	
657	001554	042356	DT4	

658	001556	044473	ITEM37: EM37	;RACE E45 BAD (AFIRO4(1)*MUL:ASHC+MFP))
659	001560	042331	DH4	
660	001562	042356	DT4	
661	001564	044540	ITEM40: EM40	;RACE E33 BAD (AFIRO5(1)*(MUL:ASHC+MFP))
662	001566	042331	DH4	
663	001570	042356	DT4	
664	001572	044610	ITEM41: EM41	;RO DID NOT SIGN FILL ON RIGHT SHIFT
665	001574	044654	DH41	;ERROR PC RO TEST NUMBER
666				; EXPECT ACTUAL
667	001576	044732	DT41	;SERRPC, \$TMPO, \$REGO, \$STSTNM
668	001600	044744	ITEM42: EM42	;BAD CC'S ON RIGHT SHIFT
669	001602	044774	DH42	;ERRORPC PSW TEST NUMBER
670				; EXPECT ACTUAL
671	001604	045052	DT42	;SERRPC, \$TMPO, \$ERPSW, \$STSTNM
672	001606	045064	ITEM43: EM43	;RO<0> DID NOT GO TO R1<15>
673	001610	045117	DH43	;ERRORPC RO R1 TEST NUMBER
674				; EXPECT ACTUAL EXPECT ACTUAL
675	001612	045230	DT43	;SERRPC, \$TMPO, \$REGO, \$TMP1, \$REG1, \$STSTNM
676	001614	045246	ITEM44: EM44	;RO DID NOT SHIFT LEFT PROPERLY
677	001616	045117	DH43	
678	001620	045230	DT43	
679	001622	045313	ITEM45: EM45	;BAD CC'S ON LEFT SHIFT
680	001624	044774	DH42	
681	001626	045052	DT42	
682	001630	043603	ITEM46: EM25	;R1 DID NOT SHIFT LEFT PROPERLY
683	001632	045117	DH43	
684	001634	045230	DT43	
685	001636	045342	ITEM47: EM47	;BAD CC'S ON NO SHIFT
686	001640	044774	DH42	
687	001642	045052	DT42	
688	001644	045367	ITEM50: EM50	;R1 DID NOT ROTATE PROPERLY
689	001646	045117	DH43	
690	001650	045230	DT43	
691	001652	045422	ITEM51: EM51	;BITS STUCK IN SC (52 PATTERN)
692	001654	045462	DH51	;ERRORPC RO R1 C BIT
693				; EXPECT ACTUAL EXPECT ACTUAL EXPECT ACTUAL
694	001656	045622	DT51	;SERRPC, \$TMPO, \$REGO, \$TMP1, \$REG1, \$TMP2, \$ERPSW, \$STSTNM
695	001660	045644	ITEM52: EM52	;BITS STUCK IN SC (25 PATTERN)
696	001662	045462	DH51	
697	001664	045622	DT51	
698	001666	045717	ITEM53: EM53	;IRCB B FORK MUX INPUT B3 NOT GOING LOW
699	001670	045462	DH51	
700	001672	045622	DT51	
701	001674	045752	ITEM54: EM54	;STATE ASC.00 FAILED
702	001676	045117	DH43	
703	001700	045230	DT43	
704	001702	045770	ITEM55: EM55	;FORK A FAILED TO RSD.00
705	001704	042331	DH4	
706	001706	042356	DT4	
707	001710	046046	ITEM56: EM56	;EITHER GRAD DROO STUCK H OR RACK E64 BAD
708	001712	045117	DH43	
709	001714	045230	DT43	
710	001716	046126	ITEM57: EM57	;EITHER GRAD DROO STUCK LOW OR RACK E64 BAD
711	001720	045117	DH43	
712	001722	045230	DT43	
713	001724	046205	ITEM60: EM60	;EITHER GRAJ SC=0 NOT GETTING TO RACK E50

714	001726	045117	DH43		;OR RACK E50 BAD
715	001730	045230	DT43		
716	001732	046315	ITEM61: EM61		;INSTRUCTION FAILED TO LOAD R0 & R1 CORRECTLY
717	001734	045117	DH43		;ON POSITIVE
718	001736	045230	DT43		
719	001740	046402	ITEM62: EM62		;BAD CC'S
720	001742	044774	DH42		
721	001744	045052	DT42		
722	001746	046444	ITEM63: EM63		;R0 DID NOT LOAD ON NEG.
723	001750	045117	DH43		
724	001752	045230	DT43		
725	001754	046504	ITEM64: EM64		;R1 DID NOT LOAD ON NEG.
726	001756	045117	DH43		
727	001760	045230	DT43		
728	001762	046544	ITEM65: EM65		;BAD CC'S DUE TO STATE MUL.50
729	001764	044774	DH42		
730	001766	045052	DT42		
731	001770	046601	ITEM66: EM66		;C DID NOT SET ON OVERFLOW
732	001772	042331	DH4		
733	001774	042356	DT4		
734	001776	046633	ITEM67: EM67		;C DID NOT SET ON UNDERFLOW
735	002000	042331	DH4		
736	002002	042356	DT4		
737	002004	046666	ITEM70: EM70		;STATE MUL.00 FAILED
738	002006	045117	DH43		
739	002010	045230	DT43		
740	002012	045770	ITEM71: EM55		;BAD FIELD IN IR DECODE ROM
741	002014	042331	DH4		
742	002016	042356	DT4		
743	002020	046704	ITEM72: EM72		;STATE ASH.30 DID NOT LOAD CC'S CORRECTLY
744	002022	044774	DH42		
745	002024	045052	DT42		
746	002026	046747	ITEM73: EM73		;STATE ASH.41 FAILED
747	002030	043636	DH25		
748	002032	043712	DT25		
749	002034	046765	ITEM74: EM74		;BAD CC'S DUE TO STATE ASH.41
750	002036	044774	DH42		
751	002040	045052	DT42		
752	002042	047030	ITEM75: EM75		;BEN2 FAILED
753	002044	042331	DH4		
754	002046	042356	DT4		
755	002050	047125	ITEM76: EM76		;BAD CC'S DUE TO DVE.00
756	002052	044774	DH42		
757	002054	045052	DT42		
758	002056	047150	ITEM77: EM77		;BAD CC'S DUE TO DVC.70
759	002060	044774	DH42		
760	002062	045052	DT42		
761	002064	047173	ITE100: EM100		;BEN16 FAILED
762	002066	042331	DH4		
763	002070	042356	DT4		
764	002072	047223	ITE101: EM101		;BEN4 FAILED
765	002074	042331	DH4		
766	002076	042356	DT4		
767	002100	047356	ITE102: EM102		;QUOTIENT OK REMAINDER BAD
768	002102	045117	DH43		
769	002104	045230	DT43		

770	002106	047430	ITE103:	EM103	;BEN3 FAILED
771	002110	042331		DH4	
772	002112	042356		DT4	
773	002114	047457	ITE104:	EM104	;BEN04 STUCK TO DIV SUB
774	002116	042331		DH4	
775	002120	042356		DT4	
776	002122	047556	ITE105:	EM105	;CAN'T DETERMINE CAUSE OF FAILURE
777	002124	045117		DH43	
778	002126	045230		DT43	
779	002130	047607	ITE106:	EM106	;BEN16 FAILED TO DIV.50
780	002132	042331		DH4	
781	002134	042356		DT4	
782	002136	047556	ITE107:	EM105	;EM107 SAME AS EM105
783	002140	045117		DH43	
784	002142	045230		DT43	
785	002144	047706	ITE110:	EM110	;BEN04*-N FAILED
786	002146	042331		DH4	
787	002150	042356		DT4	
788	002152	050005	ITE111:	EM111	;BEN02 FAILED
789	002154	042331		DH4	
790	002156	042356		DT4	
791	002160	050035	ITE112:	EM112	;BEN05 FAILED
792	002162	042331		DH4	
793	002164	042356		DT4	
794	002166	050164	ITE113:	EM113	;BEN16 FAILED (RACK E50(80))
795	002170	042331		DH4	
796	002172	042356		DT4	
797	002174	050213	ITE114:	EM114	;BEN16 FAILED (RACK E64(80))
798	002176	042331		DH4	
799	002200	042356		DT4	
800	002202	050242	ITE115:	EM115	;ROM STATE FAILED
801	002204	045117		DH43	
802	002206	045230		DT43	
803	002210	047356	ITE116:	EM102	;QUOTIENT OK, REMAINDER BAD
804	002212	045117		DH43	
805	002214	045230		DT43	
806	002216	050317	ITE117:	EM117	;BAD CC'S IN DVC.90 OR RACK E63 BAD
807	002220	044774		DH42	
808	002222	045052		DT42	
809	002224	050403	ITE120:	EM120	;BEN05 DIV QUIT (N(0)*SR15(0)) DID NOT
810	002226	042331		DH4	;GO LOW OR RACK E63(C0) STUCK HIGH
811	002230	042356		DT4	
812	002232	050502	ITE121:	EM121	;CC'S BAD DUE TO DIV.30 OR DVE.20
813	002234	044774		DH42	
814	002236	045052		DT42	
815	002240	050552	ITE122:	EM122	;GRAJ E5 BAD
816	002242	042331		DH4	
817	002244	042356		DT4	
818	002246	050612	ITE123:	EM123	;RACK E63(D0) STUCK LOW
819	002250	042331		DH4	
820	002252	042356		DT4	
821	002254	050641	ITE124:	EM124	;CC'S DID NOT LOAD PROPERLY
822	002256	044774		DH42	
823	002260	045052		DT42	
824	002262	050674	ITE125:	EM125	;EITHER GRAJ E5(N(1)*SR15(1)) BAD
825	002264	045117		DH43	;OR ROM STATE BAD

826	002266	045230		DT43	
827	002270	047356	ITE126:	EM102	;QUOT. OK REMAINDER BAD
828	002272	045117		DH43	
829	002274	045230		DT43	
830	002276	050756	ITE127:	EM127	;QUOT. BAD, REMAINDER OK
831	002300	045117		DH43	
832	002302	045230		DT43	
833	002304	051030	ITE130:	EM130	;QUOTIENT BAD
834	002306	045117		DH43	
835	002310	045230		DT43	
836	002312	047356	ITE131:	EM102	;QUOT. OK, REMAINDER BAD
837	002314	045117		DH43	
838	002316	045230		DT43	
839	002320	051030	ITE132:	EM130	;QUOT. BAD
840	002322	045117		DH43	
841	002324	045230		DT43	
842	002326	047356	ITE133:	EM102	;QUOT. OK, REMAIN. BAD
843	002330	045117		DH43	
844	002332	045230		DT43	
845	002334	051071	ITE134:	EM134	;BAD CC'S ON DIVISION OVERFLOW
846	002336	044774		DH42	
847	002340	045052		DT42	
848	002342	051140	ITE135:	EM135	;RO DID NOT LOAD CORRECTLY
849	002344	044654		DH41	
950	002346	044732		DT41	
851	002350	051147	ITE136:	EM136	;THE SP DID NOT INCREMENT
852	002352	042331		DH4	
853	002354	042356		DT4	
854	002356	050641	ITE137:	EM124	;CC'S DID NOT LOAD CORRECTLY
855	002360	044774		DH42	
856	002362	045052		DT42	
857	002364	051203	ITE140:	EM140	;FORK A FAILED
858	002366	042331		DH4	
859	002370	042356		DT4	
860	002372	051241	ITE141:	EM141	;STATE MTP.10 FAILED
861	002374	042331		DH4	
862	002376	042356		DT4	
863	002400	051300	ITE142:	EM142	;SP LOADED INCORRECTLY
864	002402	042716		DH14	
865	002404	043000		DT14	
866	002406	051326	ITE143:	EM143	;STATE MTP.00 DID NOT PUT PCB IN DR
867	002410	042331		DH4	
868	002412	042356		DT4	
869	002414	051363	ITE144:	EM144	;FORK A FAILED
870	002416	042331		DH4	
871	002420	042356		DT4	
872	002422	051412	ITE145:	EM145	;STATE MFP.10 DID NOT DEC. THE SP
873	002424	042331		DH4	
874	002426	042356		DT4	
875	002430	051452	ITE146:	EM146	;RO DID NOT GET PUT ON THE STACK
876	002432	042331		DH4	
877	002434	042356		DT4	
878	002436	051512	ITE147:	EM147	;BAD CC'S
879	002440	044774		DH42	
880	002442	045052		DT42	
881	002444	051203	ITE150:	EM140	;RACF X/CLASS DOES NOT GO HIGH

882	002446	042331	DH4	
883	002450	042356	DT4	
884	002452	051543	ITE151: EM151	;STATE MFP.00 IS BAD
885	002454	042331	DH4	
886	002456	042356	DT4	
887	002460	051556	ITE152: EM152	;BAD CC'S
888	002462	044774	DH42	
889	002464	045052	DT42	
890	002466	043124	ITE153: EM16	;RACE X/CLASS DOES NOT GO HIGH
891	002470	042331	DH4	
892	002472	042356	DT4	
893	002474	045770	ITE154: EM55	;R(NUL:ASHC+MFP) FIELD OF IR ROM BAD
894	002476	042331	DH4	
895	002500	042356	DT4	
896	002502	051605	ITE155: EM155	;STATE TRP.01 FAILED
897	002504	042331	DH4	
898	002506	042356	DT4	
899	002510	051636	ITE156: EM156	;TRAP VECTOR DECODE FAILED
900	002512	042331	DH4	
901	002514	042356	DT4	
902	002516	051767	ITE157: EM157	;STATE TRP.01 FAILED
903	002520	042331	DH4	
904	002522	042356	DT4	
905	002524	052020	ITE160: EM160	;STATE TRP.01 FAILED
906	002526	042331	DH4	
907	002530	042356	DT4	
908	002532	052051	ITE161: EM161	;TRAP VECTOR DECODE FAILED
909	002534	042331	DH4	
910	002536	042356	DT4	
911	002540	052157	ITE162: EM162	;STATE TRP.01 FAILED
912	002542	042331	DH4	
913	002544	042356	DT4	
914	002546	052210	ITE163: EM163	;BIT FAILED IN PIRQ REG
915	002550	052237	DH163	;ERRORPC PIRQ TEST NUMBER
916				; EXPECT ACTUAL
917	002552	052316	DT163	;SERRPC,\$TMPO,\$EPIRQ,\$\$TSTNM
918	002554	052330	ITE164: EM164	;STATE TST.10 HAS BAD BEN FIELD
919	002556	042331	DH4	
920	002560	042356	DT4	
921	002562	052361	ITE165: EM165	;HONOR PIR 1 DOES NOT GO LOW
922	002564	042331	DH4	
923	002566	042356	DT4	
924	002570	052452	ITE166: EM166	;PIR 6 WORKS BUT 4 & 1 DON'T
925	002572	042331	DH4	
926	002574	042356	DT4	
927	002576	052566	ITE167: EM167	;TMCA ABOVE BR7 MIGHT BE STUCK LOW
928	002600	042331	DH4	
929	002602	042356	DT4	
930	002604	052630	ITE170: EM170	;TMCE BRQ CLDCK MIGH BE STUCH LOW
931	002606	042331	DH4	
932	002610	042356	DT4	
933	002612	052672	ITE171: EM171	;BEN 13 FAILED TO PUP.00
934	002614	042331	DH4	
935	002616	042356	DT4	
936	002620	053022	ITE172: EM172	;BEN 13 FAILED TO SER.00
937	002622	042331	DH4	

938	002624	042356		
939	002626	000000	ITE173:	DT4 0 ;DELETED
940	002630	042331		
941	002632	042356		
942	002634	053304	ITE174:	DT4 EM174 ;PIR 2 DID NOT INTERRUPT
943	002636	042331		
944	002640	042356		
945	002642	053375	ITE175:	EM175 ;BEN 13 FAILED
946	002644	042331		
947	002646	042356		
948	002650	053412	ITE176:	EM176 ;LEVEL 1 INT. WHEN CPU LEVEL 1 ENABLED
949	002652	053613		
950	002654	053644		
951	002656	053454	ITE177:	EM177 ;ERRORPC PIRQ TEST NUMBER ;SERRPC,SEPIRQ,\$STSTNM ;PIR 3 DID NOT INTERRUPT
952	002660	042331		
953	002662	042356		
954	002664	053375	ITE200:	EM175 ;BEN 13 FAILED
955	002666	042331		
956	002670	042356		
957	002672	053545	ITE201:	EM201 ;LEVEL 2 WHEN CPU LEVEL 2 ENABLED
958	002674	053613		
959	002676	053644		
960	002700	053654	ITE202:	EM202 ;PIR 4 DID NOT INTERRUPT
961	002702	042331		
962	002704	042356		
963	002706	053375	ITE203:	EM175 ;BEN 13 FAILED
964	002710	042331		
965	002712	042356		
966	002714	053745	ITE204:	EM204 ;LEVEL 3 WHEN CPU LEVEL 3 ENABLED
967	002716	053613		
968	002720	053644		
969	002722	054013	ITE205:	EM205 ;PIR 5 DID NOT INTERRUPT
970	002724	042331		
971	002726	042356		
972	002730	053375	ITE206:	EM175 ;BEN 13 FAILED
973	002732	042331		
974	002734	042356		
975	002736	054105	ITE207:	EM207 ;LEVEL 4 INTERRUPT WHEN NOT SUPPOSE TO
976	002740	053613		
977	002742	053644		
978	002744	054153	ITE210:	EM210 ;FAILURE AFTER TMCB E35
979	002746	042331		
980	002750	042356		
981	002752	054230	ITE211:	EM211 ;FAILURE IN TMCB E35 OR BEFORE
982	002754	042331		
983	002756	042356		
984	002760	054320	ITE212:	EM212 ;BEN 13 FAILED
985	002762	042331		
986	002764	042356		
987	002766	054373	ITE213:	EM213 ;LEVEL 5 INTERRUPT WHEN NOT SUPPOSE TO
988	002770	053613		
989	002772	053644		
990	002774	054441	ITE214:	EM214 ;LEVEL 7 DID NOT INTERRUPT
991	002776	042331		
992	003000	042356		
993	003002	053375	ITE215:	EM175 ;BEN 13 FAILED

994	003004	042331	DH4	
995	003006	042356	DT4	
996	003010	054532	ITE216:	EM216 ;LEVEL 6 INTERRUPT WHEN NOT SUPPOSE TO
997	003012	053613	DH201	
998	003014	053644	DT201	
999	003016	054600	ITE217:	EM217 ;NO TIMEOUT ON DATI
1000	003020	042331	DH4	
1001	003022	042356	DT4	
1002	003024	054635	ITE220:	EM220 ;BEN 13 FAILED
1003	003026	042331	DH4	
1004	003030	042356	DT4	
1005	003032	054725	ITE221:	EM221 ;NO TIMEOUT ON DATO
1006	003034	042331	DH4	
1007	003036	042356	DT4	
1008	003040	054762	ITE222:	EM222 ;BR 4 INTERRUPTS WHEN CPU AT LEVEL 7
1009	003042	042331	DH4	
1010	003044	042356	DT4	
1011	003046	055051	ITE223:	EM223 ;BOTH BR 4 & BR 6 FAILED
1012	003050	042331	DH4	
1013	003052	042356	DT4	
1014	003054	055072	ITE224:	EM224 ;BR 4 FAILED
1015	003056	042331	DH4	
1016	003060	042356	DT4	
1017	003062	055241	ITE225:	EM225 ;BR 4 FAILED BUT BR 6 OK
1018	003064	042331	DH4	
1019	003066	042356	DT4	
1020	003070	055352	ITE226:	EM226 ;BR 5 INTERRUPT FAILED
1021	003072	042331	DH4	
1022	003074	042356	DT4	
1023	003076	055441	ITE227:	EM227 ;BR 6 INTERRUPT FAILED
1024	003100	042331	DH4	
1025	003102	042356	DT4	
1026	003104	055531	ITE230:	EM230 ;YEL ZONE TRAP FAILED
1027	003106	042331	DH4	
1028	003110	042356	DT4	
1029	003112	000000	ITE231:	0 ;DELETED
1030	003114	042331	DH4	
1031	003116	042356	DT4	
1032	003120	056016	ITE232:	EM232 ;JSR WITH BAD STACK FAILED
1033	003122	042331	DH4	
1034	003124	042356	DT4	
1035	003126	056077	ITE233:	EM233 ;STACK LIMIT REG DID NOT DISABLE YEL ZONE
1036	003130	042331	DH4	
1037	003132	042356	DT4	
1038	003134	056203	ITE234:	EM234 ;TMCC KERNAL R6 DID NOT DISABLE YEL ZONE
1039	003136	042331	DH4	
1040	003140	042356	DT4	
1041	003142	056317	ITE235:	EM235 ;RED ZONE REFERENCE FAILED
1042	003144	042331	DH4	
1043	003146	042356	DT4	
1044	003150	000000	ITE236:	0 ;DELETED
1045	003152	042331	DH4	
1046	003154	042356	DT4	
1047	003156	056474	ITE237:	EM237 ;BEN 13 FAILED TO PUP.00
1048	003160	042331	DH4	
1049	003162	042356	DT4	

1050	003164	056576	ITE240:	EM240	;BEN 13 FAILED TO BRK.80
1051	003166	042331		DH4	
1052	003170	042356		DT4	
1053	003172	056707	ITE241:	EM241	;NO RED ZONE ON STACK OVERFLOW
1054	003174	042331		DH4	
1055	003176	042356		DT4	
1056	003200	057027	ITE242:	EM242	;NO RED ZONE WHEN SL REG>BADDR
1057	003202	042331		DH4	
1058	003204	042356		DT4	
1059	003206	057172	ITE243:	EM243	;52400 PATTERN FAILED IN SL REG
1060	003210	057242		DH243	;ERRORPC SL REG TEST NUMBER
1061					; EXPECT ACTUAL
1062	003212	057324		DT243	;SERRPC, \$TMP0, E2STKLMT, \$STSTNM
1063	003214	057336	ITE244:	EM244	;125000 PATTERN FAILED IN SL REG
1064	003216	057242		DH243	
1065	003220	057406		DT244	
1066	003222	057420	ITE245:	EM245	;SERRPC, \$TMP0, E1STKLMT, \$STSTNM
1067	003224	042331		DH4	;BR SELECTED INSTEAD OF SL
1068	003226	042356		DT4	
1069	003230	057522	ITE246:	EM246	;SL AND PB BOTH SELECTED
1070	003232	042331		DH4	
1071	003234	042356		DT4	
1072	003236	057653	ITE247:	EM247	;PSW SELECTED INSTEAD OF SL
1073	003240	042331		DH4	
1074	003242	042356		DT4	
1075	003244	057774	ITE250:	EM250	;WHAT HAPPENED?
1076	003246	060021		DH250	;BOTH PATTERNS FAILED
1077	003250	060140		DT250	
1078	003252	060156	ITE251:	EM251	;YEL ZONE IN RED REGION (SP=330)
1079	003254	042331		DH4	
1080	003256	042356		DT4	
1081	003260	060335	ITE252:	EM252	;YEL ZONE IN RED REGION (SP=240)
1082	003262	042331		DH4	
1083	003264	042356		DT4	
1084	003266	060406	ITE253:	EM253	;YEL ZONE IN RED REGION (SP=140)
1085	003270	042331		DH4	
1086	003272	042356		DT4	
1087	003274	060457	ITE254:	EM254	;RED ZONE IN YELLOW REGION (SP=376)
1088	003276	042331		DH4	
1089	003300	042356		DT4	
1090	003302	060560	ITE255:	EM255	;CPU ERR REG BIT 4 DOES NOT SET
1091	003304	060640		DH255	
1092	003306	044732		DT41	
1093	003310	060721	ITE256:	EM256	;CPU ERR REG DOES NOT CLEAR
1094	003312	042331		DH4	
1095	003314	042356		DT4	
1096	003316	060757	ITE257:	EM257	;CPU ERROR BIT 3 DOES NOT SET
1097	003320	060640		DH255	
1098	003322	044732		DT41	
1099	003324	061031	ITE260:	EM260	;CPU ERROR BIT 3 DOES NOT CLEAR
1100	003326	042331		DH4	
1101	003330	042356		DT4	
1102	003332	061103	ITE261:	EM261	;CPU ERROR BIT 2 DOES NOT SET
1103	003334	060640		DH255	
1104	003336	044732		DT41	
1105	003340	061156	ITE262:	EM262	;CPU ERROR BIT 2 DOES NOT CLEAR

1106	003342	042331	DH4	
1107	003344	042356	DT4	
1108	003346	061227	ITE263:	EM263
1109	003350	061416		DH263
1110	003352	000000		0
1111	003354	062040	ITE264:	EM264
1112	003356	000000		0
1113	003360	000000		0
1114	003362	062063	ITE265:	EM265
1115	003364	042331		DH4
1116	003366	042356		DT4
1117	003370	062401	ITE266:	EM266
1118	003372	042331		DH4
1119	003374	042356		DT4
1120	003376	062527	ITE267:	EM267
1121	003400	042331		DH4
1122	003402	042356		DT4
1123	003404	062617	ITE270:	EM270
1124	003406	042331		DH4
1125	003410	042356		DT4
1126	003412	062725	ITE271:	EM271
1127	003414	042331		DH4
1128	003416	042356		DT4
1129	003420	063066	ITE272:	EM272
1130	003422	042331		DH4
1131	003424	042356		DT4
1132	003426	063125	ITE273:	EM273
1133	003430	042331		DH4
1134	003432	042356		DT4
1135	003434	063174	ITE274:	EM274
1136	003436	042331		DH4
1137	003440	042356		DT4
1138	003442	063214	ITE275:	EM275
1139	003444	042331		DH4
1140	003446	042356		DT4
1141	003450	063600	ITE276:	EM276
1142	003452	042331		DH4
1143	003454	042356		DT4
1144	003456	000000	ITE277:	0
1145	003460	042331		DH4
1146	003462	042356		DT4
1147	003464	063632	ITE300:	EM300
1148	003466	042331		DH4
1149	003470	042356		DT4
1150	003472	063731	ITE301:	EM301
1151	003474	042331		DH4
1152	003476	042356		DT4
1153	003500	064026	ITE302:	EM302
1154	003502	042331		DH4
1155	003504	042356		DT4
1156	003506	064103	ITE303:	EM303
1157	003510	042331		DH4
1158	003512	042356		DT4
1159	003514	064245	ITE304:	EM304
1160	003516	042331		DH4
1161	003520	042356		DT4

;GOING TO NEXT TEST
 ;NEITHER - BYIN NOR DATI CAUSED ODD ADDR.
 ;DATI TRAPPED BUT - BYIN DIDN'T
 ;BOTH DATI & DATO FAILED
 ;DATO WORKS BUT DATI FAILED
 ;NO TRAP ON DATO
 ;NO TRAP ON SM357*SRC1 DATI
 ;ODD ADDR BIT IN CPU ERROR FAILED
 ;NO TRAP ON DATO
 ;T BIT TRAP FAILED
 ;T BIT NEVER SET
 ;DELETED
 ;TRAP VECTOR DECODE FAILED
 ;RTT DID NOT DISABLE T BIT
 ;PIR 1 DID NOT DISABLE ON BR4
 ;PIR 1 CAME THRU ON YELLOW ZONE
 ;PIR 2 CAME THRU ON YELLOW ZONE

1162	003522	064342	ITE305:	EM305	;PIR 3 CAME THRU ON YELLOW ZONE
1163	003524	042331		DH4	
1164	003526	042356		DT4	
1165	003530	064440	ITE306:	EM306	;BR4 CAME THRU ON PIR 5
1166	003532	042331		DH4	
1167	003534	042356		DT4	
1168	003536	064573	ITE307:	EM307	;BR4 CAME THRU ON PIR 5
1169	003540	042331		DH4	
1170	003542	042356		DT4	
1171	003544	064647	ITE310:	EM310	;BR4 CAME THRU ON BR 5
1172	003546	042331		DH4	
1173	003550	042356		DT4	
1174	003552	064722	ITE311:	EM311	;BR4 CAME IN ON PIR 6
1175	003554	042331		DH4	
1176	003556	042356		DT4	
1177	003560	064774	ITE312:	EM312	;BR4 CAME IN ON PIR 7
1178	003562	042331		DH4	
1179	003564	042356		DT4	
1180	003566	065131	ITE313:	EM313	;PIR 4 CAME IN ON BR5
1181	003570	042331		DH4	
1182	003572	042356		DT4	
1183	003574	065202	ITE314:	EM314	;PIR 4 CAME IN ON BR6
1184	003576	042331		DH4	
1185	003600	042356		DT4	
1186	003602	065323	ITE315:	EM315	;PIR 4 CAME THRU ON SL YELLOW
1187	003604	042331		DH4	
1188	003606	042356		DT4	
1189	003610	065455	ITE316:	EM316	;BR5 CAME IN ON PIR 5
1190	003612	042331		DH4	
1191	003614	042356		DT4	
1192	003616	065600	ITE317:	EM317	;BR5 CAME IN ON PIR 6
1193	003620	042331		DH4	
1194	003622	042356		DT4	
1195	003624	065620	ITE320:	EM320	;BR5 CAME IN ON PIR 7
1196	003626	042331		DH4	
1197	003630	042356		DT4	
1198	003632	065746	ITE321:	EM321	;PIR 5 CAME IN ON BR6
1199	003634	042331		DH4	
1200	003636	042356		DT4	
1201	003640	066017	ITE322:	EM322	;PIR 5 CAME IN ON SL YELLOW
1202	003642	042331		DH4	
1203	003644	042356		DT4	
1204	003646	066104	ITE323:	EM323	;BR6 CAME IN ON PIR 6
1205	003650	042331		DH4	
1206	003652	042356		DT4	
1207	003654	066223	ITE324:	EM324	;BR6 CAME IN ON PIR 7
1208	003656	042331		DH4	
1209	003660	042356		DT4	
1210	003662	066313	ITE325:	EM325	;PIR 6 CAME IN ON SL YELLOW
1211	003664	042331		DH4	
1212	003666	042356		DT4	
1213	003670	066400	ITE326:	EM326	;PIR 7 CAME IN ON SL YELLOW
1214	003672	042331		DH4	
1215	003674	042356		DT4	
1216	003676	066470	ITE327:	EM327	;PSW BIT 11 DID NOT SET
1217	003700	042331		DH4	

1218	003702	042356		DT4	
1219	003704	066644	ITE330:	EM330	;R12 CLEARED R2
1220	003706	042331		DH4	
1221	003710	042356		DT4	
1222	003712	066766	ITE331:	EM331	;R2 SRC WAS AFFECTED BY CLR R12
1223	003714	042331		DH4	
1224	003716	042356		DT4	
1225	003720	067112	ITE332:	EM332	;R2 DST WAS AFFECTED BY (R12)+
1226	003722	042331		DH4	
1227	003724	042356		DT4	
1228	003726	067203	ITE333:	EM333	;R2 SRC WAS AFFECTED BY (R12)+
1229	003730	042331		DH4	
1230	003732	042356		DT4	
1231	003734	067237	ITE334:	EM334	;R15 DST AFTER UPAD 2
1232	003736	042331		DH4	
1233	003740	042356		DT4	
1234	003742	067273	ITE335:	EM335	;R15 SRC DOES NOT SELECT ON UPAD 2
1235	003744	042331		DH4	
1236	003746	042356		DT4	
1237	003750	067330	ITE336:	EM336	;PDRD PS11 DOES NOT GET TO GRAC
1238	003752	042331		DH4	
1239	003754	042356		DT4	
1240	003756	067400	ITE337:	EM337	;BAD BITS IN GPR SET 1 SRC
1241	003760	067445		DH337	;ERRORPC PATTERN TESTNUMBER
1242	003762	067502		DT337	;SERRPC, STMP1, SSTSTNM
1243	003764	067512	ITE340:	EM340	;BAD BITS IN GPR SET 1 DST
1244	003766	067445		DH337	
1245	003770	067502		DT337	
1246	003772	067564	ITE341:	EM341	;GRAB DST SET 1 DOES NOT GO LOW ON R14
1247	003774	042331		DH4	
1248	003776	042356		DT4	
1249	004000	067632	ITE342:	EM342	;GRAB SRC SET 1 DOES NOT GO LOW ON R14
1250	004002	042331		DH4	
1251	004004	042356		DT4	
1252	004006	067700	ITE343:	EM343	;50000 PATTERN FAILED IN PSW
1253	004010	044774		DH42	
1254	004012	045052		DT42	
1255	004014	067734	ITE344:	EM344	;164000 PATTERN FAILED IN PSW
1256	004016	044774		DH42	
1257	004020	045052		DT42	
1258	004022	067771	IRE345:	EM345	;PSW HIGH BYTE DID NOT CLEAR
1259	004024	044774		DH42	
1260	004026	045052		DT42	
1261	004030	070025	ITE346:	EM346	;SUPER SP DOES NOT SELECT ON UPAD 5
1262	004032	042331		DH4	
1263	004034	042356		DT4	
1264	004036	070106	ITE347:	EM347	;SUPER SP DOES NOT SELECT ON UPAD 0
1265	004040	042331		DH4	
1266	004042	042356		DT4	
1267	004044	070167	ITE350:	EM350	;USER SP DOES NOT SELECT ON UPAD 5
1268	004046	042331		DH4	
1269	004050	042356		DT4	
1270	004052	070245	ITE351:	EM351	;USER SP DOES NOT SELECT ON UPAD 0
1271	004054	042331		DH4	
1272	004056	042356		DT4	
1273	004060	070323	ITE352:	EM352	;EITHER USER OR SUPER SP DST FAILED

1274	004062	067445	DH337	
1275	004064	067502	DT337	
1276	004066	070377	ITE353: EM353	;EITHER USER OR SUPER SP SRC FAILED
1277	004070	067445	DH337	
1278	004072	067502	DT337	
1279	004074	070453	ITE354: EM354	;KSP SRC CHANGED ON MTP
1280	004076	042331	DH4	
1281	004100	042356	DT4	
1282	004102	070505	ITE355: EM355	;KSP SRC & DST CHANGED ON MTP
1283	004104	042331	DH4	
1284	004106	042356	DT4	
1285	004110	070564	ITE356: EM356	;KSP DST CHANGED ON MTP
1286	004112	042331	DH4	
1287	004114	042356	DT4	
1288	004116	070617	ITE357: EM357	;SSP DID NOT LOAD PROPERLY ON MTPD
1289	004120	070710	DH357	;ERRORPC SSP TEST NUMBER
1290				; EXPECT ACTUAL
1291	004122	043000	DT14	
1292	004124	070762	ITE360: EM360	;USER SP DID NOT LOAD ON MTP
1293	004126	042331	DH4	
1294	004130	042356	DT4	
1295	004132	071042	ITE361: EM361	;BAD FIELD IN IR DECODE ROM
1296	004134	042331	DH4	
1297	004136	042356	DT4	
1298	004140	071112	ITE362: EM362	;SSP WAS READ AND USP WAS WRITTEN
1299	004142	042331	DH4	
1300	004144	042356	DT4	
1301	004146	071150	ITE363: EM363	;SSP WAS READ AND WRITTEN
1302	004150	042331	DH4	
1303	004152	042356	DT4	
1304	004154	071221	ITE364: EM364	;CAN'T DETERMINE WHAT HAPPENED
1305	004156	042331	DH4	;SSP WAS READ BUT THE WRITE FAILED
1306	004160	042356	DT4	
1307	004162	071303	ITE365: EM365	;USP WAS READ BUT SSP WAS WRITTEN
1308	004164	042331	DH4	
1309	004166	042356	DT4	
1310	004170	071341	ITE366: EM366	;USP WAS READ BUT REG 7 WAS WRITTEN
1311	004172	042331	DH4	
1312	004174	042356	DT4	
1313	004176	071376	ITE367: EM367	;SPL WORKED IN SUPERVISOR MODE
1314	004200	044774	DH42	
1315	004202	045052	DT42	
1316	004204	071522	ITE370: EM370	;BIT FAILED TO SET IN PSW
1317	004206	044774	DH42	
1318	004210	045052	DT42	
1319	004212	071650	ITE371: EM371	;BIT FAILED TO CLEAR IN PSW
1320	004214	044774	DH42	
1321	004216	045052	DT42	
1322	004220	071755	ITE372: EM372	;BITS <13:11> DID NOT PRESET
1323	004222	044774	DH42	
1324	004224	045052	DT42	
1325	004226	072106	ITE373: EM373	;IOT DID NOT CHANGE PSW CORRECTLY
1326	004230	044774	DH42	
1327	004232	045052	DT42	
1328	004234	072220	ITE374: EM374	;PREVIOUS MODE BITS DID NOT CLEAR
1329	004236	044774	DH42	

1330	004240	045052		
1331	004242	072311	ITE375:	DT42 EM375 ;NO KT ABORT
1332	004244	042331		DH4
1333	004246	042356		DT4
1334	004250	000000	ITE376:	0 ;DELETED
1335	004252	042331		DH4
1336	004254	042356		DT4
1337	004256	000000	ITE377:	0
1338	004260	000000		0
1339	004262	000000		0
1340	004264	072475	ITE400:	EM400 ;KT ABORT TRAPPED TO 4
1341	004266	042331		DH4
1342	004270	042356		DT4
1343	004272	072607	ITE401:	EM401 ;KT ABORT TRAPPED TO 10
1344	004274	042331		DH4
1345	004276	042356		DT4
1346	004300	072655	ITE402:	EM402 ;KT ABORT TRAPPED TO 240
1347	004302	042331		DH4
1348	004304	042356		DT4
1349	004306	072720	ITE403:	EM403 ;KT TRAP DID NOT WORK
1350	004310	042331		DH4
1351	004312	042356		DT4
1352	004314	000000	ITE404:	0 ;DELETED
1353	004316	042331		DH4
1354	004320	042356		DT4
1355	004322	073034	ITE405:	EM405 ;NO KT TRAP ON SOURCE OPERAND
1356	004324	042331		DH4
1357	004326	042356		DT4
1358	004330	073163	ITE406:	EM406 ;NO ABORT ON NEXM
1359	004332	042331		DH4
1360	004334	042356		DT4
1361	004336	000000	ITE407:	0 ;DELETED
1362	004340	042331		DH4
1363	004342	042356		DT4
1364	004344	073325	ITE410:	EM410 ;NEXM BIT DID NOT SET IN CPUERR REG
1365	004346	060640		DH255
1366	004350	044732		DT41
1367	004352	073432	ITE411:	EM411 ;NEXM BIT DID NOT CLEAR IN CPUERR REG
1368	004354	042331		DH4
1369	004356	042356		DT4
1370	004360	073502	ITE412:	EM412 ;KT ABORT ON NEXM
1371	004362	042331		DH4
1372	004364	042356		DT4
1373	004366	073560	ITE413:	EM413 ;KT ABORT ON SL RED
1374	004370	042331		DH4
1375	004372	042356		DT4
1376	004374	073634	ITE414:	EM414 ;KT ABORT ON ODD ADDRESS
1377	004376	042331		DH4
1378	004400	042356		DT4
1379	004402	000000	ITE415:	0 ;DELETED
1380	004404	042331		DH4
1381	004406	042356		DT4
1382	004410	073716	ITE416:	EM416 ;TMCE CACHE BEND DID NOT GO
1383	004412	042331		DH4 ;HIGH ON KT ABORT
1384	004414	042356		DT4
1385	004416	073772	ITE417:	EM417 ;ILLEGAL HALT DID NOT TRAP

M05

1386	004420	042331	DH4	
1387	004422	042356	DT4	
1388	004424	074073	ITE420:	EM420 ;CPU ERROR REG BIT 5 DID NOT SET
1389	004426	060640	DH255	
1390	004430	044732	DT41	
1391	004432	074151	ITE421:	EM421 ;BEN 6 FAILED ON PS RESTORE
1392	004434	042331	DH4	
1393	004436	042356	DT4	
1394	004440	074252	ITE422:	EM422 ;NO PE ABORT
1395	004442	042331	DH4	
1396	004444	042356	DT4	
1397	004446	074474	ITE423:	EM423 ;PE ABORTED TO 4
1398	004450	042331	DH4	
1399	004452	042356	DT4	
1400	004454	074576	ITE424:	EM424 ;PE ABORTED TO 14
1401	004456	042331	DH4	
1402	004460	042356	DT4	
1403	004462	074665	ITE425:	EM425 ;PE ABORTED TO 104
1404	004464	042331	DH4	
1405	004466	042356	DT4	
1406	004470	074704	ITE426:	EM426 ;NO PE TRAP
1407	004472	042331	DH4	
1408	004474	042356	DT4	
1409	004476	074771	ITE427:	EM427 ;PE TRAP, TRAPPED TO
1410	004500	042331	DH4	;WRONG VECTOR
1411	004502	042356	DT4	
1412	004504	075105	ITE430:	EM430 ;PIR 6 CAME IN ON MEM MGT TRAP
1413	004506	042331	DH4	
1414	004510	042356	DT4	
1415	004512	075151	ITE431:	EM431 ;PIR 3 CAME IN ON MEM MGT TRAP
1416	004514	042331	DH4	
1417	004516	042356	DT4	
1418	004520	075216	ITE432:	EM432 ;YEL ZONE CAME IN ON PE TRAP
1419	004522	042331	DH4	
1420	004524	042356	DT4	
1421	004526	075312	ITE433:	EM433 ;MEM MGT TRAP CAME IN ON PE
1422	004530	042331	DH4	
1423	004532	042356	DT4	
1424	004534	000000	ITE434:	0 ;DELETED
1425	004536	042331	DH4	
1426	004540	042356	DT4	
1427	004542	075361	ITE435:	EM435 ;TMCC PRIORITY CLEAR DID NOT WORK
1428	004544	042331	DH4	
1429	004546	042356	DT4	
1430	004550	075512	ITE436:	EM436 ;UNIBUS PE ABORT DID NOT HAPPEN
1431	004552	042331	DH4	
1432	004554	042356	DT4	
1433	004556	075556	ITE437:	EM437 ;UNIBUS PE TRAPPED TO 0
1434	004560	042331	DH4	
1435	004562	042356	DT4	
1436	004564	075627	ITE440:	EM440 ;WAIT INSTRUCTION FAILED
1437	004566	042331	DH4	
1438	004570	042356	DT4	
1439	004572	075703	ITE441:	EM441 ;T BIT INTERRUPTED WAIT
1440	004574	042331	DH4	
1441	004576	042356	DT4	

DEKBB.CP11 ERROR POINTER TABLE

1498	005004	022706	001136			CMP	#STKS,R6	:: DONE?	
1499	005010	001374				BNE	.-6	:: LOOP BACK IF NO	
1500	005012	012706	001100			MOV	#STACK,SP	:: SETUP THE STACK POINTER	
1501	005016	012737	036656	000020		MOV	#SCOPE,2#IOTVEC	:: IOT VECTOR FOR SCOPE ROUTINE	
1502	005024	012737	000340	000022		MOV	#340,2#IOTVEC+2	:: LEVEL 7	
1503	005032	012737	037124	000030		MOV	#ERROR,2#EMTVEC	:: EMT VECTOR FOR ERROR ROUTINE	
1504	005040	012737	000340	000032		MOV	#340,2#EMTVEC+2	:: LEVEL 7	
1505	005046	012737	041614	000034		MOV	#STRAP,2#TRAPVEC	:: TRAP VECTOR FOR TRAP CALLS	
1506	005054	012737	000340	000036		MOV	#340,2#TRAPVEC+2	:: LEVEL 7	
1507	005062	012737	041646	000024		MOV	#SPWRDN,2#PWRVEC	:: POWER FAILURE VECTOR	
1508	005070	012737	000340	000026		MOV	#340,2#PWRVEC+2	:: LEVEL 7	
1509	005076	016767	030674	030664		MOV	SENDCT,SEOPCT	:: SETUP END-OF-PROGRAM COUNTER	
1510	005104	005067	174062			CLR	\$TIMES	:: INITIALIZE NUMBER OF ITERATIONS	
1511	005110	005067	174060			CLR	\$ESCAPE	:: CLEAR THE ESCAPE ON ERROR ADDRESS	
1512	005114	112767	000001	173773		MOV	#1,\$ERMAX	:: ALLOW ONE ERROR PER TEST	
1513	005122	012737	036226	000014		MOV	#\$RTRN,2#TBITVEC	:: SET "T" BIT VECTOR TO \$RTRN	
1514	005130	012737	000340	000016		MOV	#340,2#TBITVEC+2	:: LEVEL 7	
1515	005136	012767	000002	031062		MOV	#RTI,\$RTRN	:: SET \$RTRN TO A RTI	
1516	005144	012737	005172	000010		MOV	#65\$,\$RESVEC	:: TRY TO DO A RTT	
1517	005152	005046				CLR	-(SP)	:: DUMMY PS	
1518	005154	012746	005162			MOV	#64\$,-(SP)	:: AND PC	
1519	005160	000006				RTT		:: TRY THE RTT	
1520	005162	012767	000006	031036	64\$:	MOV	#RTT,\$RTRN	:: RTT IS LEGAL--SET \$RTRN TO A RTT	
1521	005170	000402				BR	66\$		
1522	005172	062706	000010		65\$:	ADD	#10,SP	:: RTT ILLEGAL--CLEAN OFF THE STACK	
1523	005176	012737	000012	000010	56\$:	MOV	#RESVEC+2,2#RESVEC	:: RESTORE TRAP CATCHER	
1524	005204	005067	031024			CLR	\$TBIT	:: CLEAR "T" BIT SWITCH	
1525	005210	012767	005210	173670		MOV	\$.,\$SLPADR	:: INITIALIZE THE LOOP ADDRESS FOR SCOPE	
1526	005216	012767	005216	173664		MOV	\$.,\$SLPERR	:: SETUP THE ERROR LOOP ADDRESS	
1527	005224	005227	177777			INC	#-1	:: FIRST TIME?	
1528	005230	001042				BNE	67\$:: BRANCH IF NO	
1529	005232	022737	036160	000042		CMP	#SENDAD,2#42	:: ACT-11?	
1530	005240	001436				BEQ	67\$:: BRANCH IF YES	
1531	005242	104400	005250			TYPE	68\$:: TYPE ASCIZ STRING	
1532	005246	000433				BR	67\$:: GET OVER THE ASCIZ	
1533						:: 68\$:	.ASCIZ	<CRLF>"MAINDEC-11-DEKBB-C POP11/70 CPU DIAGNOSTIC PART 2"<CRLF>	
1534	005336					67\$:			
1535						:: *****			
1536						:: SAVE THE MONITOR IF INITIAL LOAD			
1537	005336	005737	000042			LOOP:	TST	2#42	:: RUNNING UNDER XXDP CHAIN?
1538	005342	001003				BNE	3\$:: BRANCH IF YES	
1539	005344	005227	177777			INC	#-1	:: INITIAL LOAD?	
1540	005350	001010				BNE	2\$:: BRANCH IF NO	
1541	005352	012700	002734		3\$:	MOV	#101500,R0	:: SETUP SOB COUNT	
1542	005356	012701	077734			MOV	#SUBTAB+2,R1	:: GET END ADDRESS OF PROGRAM	
1543	005362	012702	160000			MOV	#160000,R2	:: GET END ADDRESS OF MONITOR	
1544	005366	014221			1\$:	MOV	-(R2),(R1)+	:: SAVE 1500 DECIMAL WORDS	
1545	005370	077002				SOB	R0,1\$		
1546	005372	012737	040224	000060	2\$:	MOV	#QUIT,2#TKVEC	:: SETUP KEYBOARD VECTOR	
1547	005400	012737	000340	000062		MOV	#PR7,2#TKVEC+2	:: SETUP KEYBOARD PSW	
1548	005406	005077	173526			CLR	2#TKB	:: ENSURE BUFFER CLEAR	
1549	005412	152777	000100	173516		BISB	#BIT6,2#STKS	:: SET INTERRUPT ENABLE BIT	
1550	005420	012737	036242	000004		MOV	#CPUSPUR,2#ERRVEC		
1551	005426	012737	000340	000006		MOV	#PR7,2#ERRVEC+2		
1552	005434	012737	036446	000114		MOV	#CACHSPU,2#CACHVEC		
1553	005442	012737	000340	000116		MOV	#PR7,2#CACHVEC+2		

1554	005450	005037	177766	
1555	005454	012737	177777	177744
1556	005462	005767	173412	
1557	005466	001402		
1558	005470	005037	177746	
1559				
1560				
1561				
1562				
1563				
1564				
1565				
1566				
1567				
1568				
1569				
1570				
1571				
1572				
1573				
1574				
1575				
1576				
1577				
1578				
1579				
1580				
1581				
1582	005474	000004		
1583	005476	012767	005656	173470
1584	005504	012767	005656	173556
1585	005512	012706	001100	
1586	005516	005005		
1587	005520	012737	005640	000010
1588	005526	012737	000113	177770
1589	005534	000235		
1590	005536	013767	177776	173442
1591	005544	042767	177437	173434
1592	005552	022767	000240	173426
1593	005560	001401		
1594	005562	005205		
1595	005564	012737	000044	177770 25:
1596	005572	000240		
1597	005574	000232		
1598	005576	013767	177776	173356
1599	005604	042767	177437	173350
1600	005612	022767	000100	173342
1601	005620	001404		
1602	005622	005705		
1603	005624	001401		
1604	005626	104001		
1605	005630	104002		45:
1606	005632	005705		35:
1607	005634	001410		
1608	005636	104003		
1609	005640	012737	000012	000010 15:

```

;*****
;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(AFI04(1)*AFI05(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
;*****
TST1:  SCOPE
MOV     #TST2,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
MOV     #TST2,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
MOV     #STACK,SP     ;INITIALIZE THE SP
CLR     R5            ;INITIALIZE ERROR RECORD
MOV     #15,#RESVEC  ;SETUP RESERVED VECTOR TO TRAP TO THIS ROUTINE
MOV     #113,#177770 ;SETUP MICROPROCESSOR BREAK REG
SPL     5             ;EXECUTE INSTRUCTION UNDER TEST
MOV     #PSW,$ERPSW  ;SAVE PSW
BIC     #177437,$ERPSW ;MASK OFF THE PRIORITY
CMP     #PR5,$ERPSW  ;DID PRIORITY LEVEL 5 GET SET?
BEQ     25           ;BRANCH IF YES
INC     R5           ;SET ERROR RECORD
MOV     #44,#177770 ;SETUP MICROPROCESSOR BREAK REG
NOP     ;SYNC INSTRUCTION
SPL     2            ;TEST TO SEE IF BITS 5&7 CLEAR & BIT 6 SETS
MOV     #PSW,$TMP0   ;SAVE PSW
BIC     #177437,$TMP0 ;MASK OFF THE PRIORITY
CMP     #PR2,$TMP0   ;DID BIT 6 SET & 5&7 CLEAR?
BEQ     35           ;BRANCH IF YES
TST     R5           ;DID SPL 5 FAIL?
BEQ     45           ;BRANCH IF NO
ERROR   1            ;EITHER SPL DOES NOT LOAD PSW OR BITS STUCK
ERROR   2            ;PR5 OK BUT PR2 BAD
TST     R5           ;DID SPL 5 FAIL?
BEQ     TST2        ;BRANCH IF NO
ERROR   3            ;PR2 OK BUT PR5 FAILED
MOV     #12,#RESVEC  ;RESTORE RESERVEC VECTOR ADDRESS

```

```

1610 005646 005705          TST      R5          ;DID R5 GET INCREMENTED?
1611 005650 001401          BEQ      5$          ;BRANCH IF NO
1612 005652 104004          ERROR    4          ;FORK A FAILED TO D30.00
1613 005654 104005          ERROR    5          ;FORK A FAILED TO RSD.02
1614                                     5$:
1615                                     *****
1616                                     *TEST 2          RESET
1617                                     *
1618                                     * IF FORK A FAILS EXECUTION WILL EITHER GO TO WAT.00 OR TRP.02.
1619                                     * THE LA30 PRINTER WILL BE STARTED SUCH THAT A FAILURE INTO
1620                                     * WAT.00 WILL RECOVER.
1621                                     *
1622                                     * IF TRP.01 IS ENTERED A TRAP SEQUENCE WILL EXECUTE
1623                                     * WITH A TRAP VECTOR OF 4.
1624                                     *
1625                                     * ROM FLOW-15,255,374
1626                                     *****
1627 005656 000004          TST2:  SCOPE
1628 005660 012767 003674 173216      MOV      #101980,$ICNT ;SET ITERATION COUNT
1629 005666 012767 006152 173300      MOV      #TST3,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
1630 005674 012767 006152 173366      MOV      #TST3,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
1631 005702 013767 177776 173260      MOV      @#PSW,$TMP3   ;SAVE PSW
1632 005710 012767 005740 173170      MOV      #4$,$LPADR   ;SETUP LOOP ADR
1633 005716 012767 005740 173164      MOV      #4$,$LPERR   ;SETUP ERROR LOOP
1634 005724 012737 006142 000064      MOV      #2$,@#TPVEC  ;PUT ADDRESS OF 2$ IN PRINTER INTERRUPT VEC
1635 005732 012737 006132 000004      MOV      #1$,@#ERRVEC ;PUT ADDRESS OF 1$ IN LOCATION 4
1636 005740 012706 001100          4$:  MOV      #STACK,$P    ;INITIALIZE THE SP
1637                                     ;TO CATCH A FAILURE TO THE TRP.02 STATE
1638 005744 000230          SPL      0          ;SET CPU AT 0
1639 005746 012777 000015 173170      MOV      #15,@#STPB   ;SEND A CR TO THE PRINTER
1640 005754 105777 173162          7$:  TSTB    @#STPS      ;WAIT FOR CR TO FINISH
1641 005760 100375          BPL      7$          ;INCREASE TP DOUBLE BUFFERED
1642 005762 012777 000101 173154      MOV      #101,@#STPB  ;SEND AN "A"
1643 005770 052777 000100 173144      BIS      #BIT6,@#STPS ;SET THE INTERRUPT FLAG
1644 005776 012737 034157 177776      MOV      #34157,@#PSW ;SET AS MANY BITS AS POSSIBLE IN PSW
1645 006004 000240          NOP
1646 006006 000005          RESET
1647 006010 013767 177776 173170      MOV      @#PSW,$ERPSW ;EXECUTE INSTRUCTION UNDER TEST
1648 006016 017700 173120      MOV      @#STPS,$R0   ;SAVE PSW
1649 006022 042777 000100 173114      BIC      #BIT6,@#STPB ;GET THE PRINTER STATUS
1650 006030 032700 000100          BIT      #BIT6,$R0   ;CLEAR INTERRUPT FLAG INCASE RESET FAILED
1651 006034 001401          BEQ      3$          ;DID INTERRUPT FLAG CLEAR?
1652 006036 104006          ERROR    6          ;BRANCH IF YES
1653 006040 105777 173076          3$:  TSTB    @#STPS      ;RESET DID NOT WORK
1654 006044 100375          BPL      3$          ;WAIT FOR CHARACTER TO FINISH
1655 006046 012737 036242 000004      MOV      #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
1656 006054 012767 034157 173100      MOV      #34157,$TMP0 ;SAVE EXPECTED VALUE
1657 006062 032737 000020 177776      BIT      #BIT4,@#PSW  ;IS T BIT ON?
1658 006070 001407          BEQ      5$          ;BRANCH IF NO
1659 006072 012767 034177 173062      MOV      #34177,$TMP0 ;SAVE EXPECTED VALUE
1660 006100 022767 034177 173100      CMP      #34177,$ERPSW ;DID PSW COME OUT OK?
1661 006110 022767 034157 173070          BR      6$          ;DID PSW CHANGE?
1662 006116          5$:  CMP      #34157,$ERPSW
1663 006116 001415          BEQ      TST3      ;BRANCH IF NO
1664 006120 012767 034157 173034      MOV      #34157,$TMP0 ;SAVE EXPECTED VALUE
1665 006126 104377          ERROR    377      ;PSW CHANGED ON RESET

```

E06

DEKBBC.P11 T2 RESET

SEQ 0069

1666	006130	000460				460		
1667	006132	042777	000100	173002	1\$:	BIC	#BIT6,2STPS	;CLEAR PRINTER INTERRUPT FLAG
1668	006140	104007				ERROR	7	;FORK A FAILED INTO TRP.02
1669	006142	042777	000100	172772	2\$:	BIC	#BIT6,2STPS	;CLEAR PRINTER INTERRUPT FLAG
1670	006150	104010				ERROR	10	;FORK A FAILED TO WAT.00

1671
1672
1673
1674
1675
1676
1677
1678
1679
1680
1681
1682
1683
1684
1685

```

*****
;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 EB IS BAD.
; D67.01 WILL STEP THE PCB AND TRAP TO LOCATION 10 AFTER STATE D10.60.
;
; ROM FLOW-47,252,235,234
*****

```

1686	006152	000004				TST3:	SCOPE	
1687	006154	152777	000100	172754			BISB	#BIT6,2STKS ;RESTORE INTER FLAG AFTER RESET
1688	006162	012767	006250	172716			MOV	#9\$, \$LPADR ;SETUP LOOP ADDRESS
1689	006170	012767	006250	172712			MOV	#9\$, \$LPERR ;SETUP ERROR LOOP
1690	006176	013767	177776	172764			MOV	2#PSW, \$TMP3 ;SAVE PSW
1691	006204	032737	000020	177776			BIT	#BIT4, 2#PSW ;IS T BIT ON?
1692	006212	001416					BEQ	9\$;BRANCH IF NO
1693	006214	012746	000340				MOV	#PR7, -(SP) ;PUT NEW PSW ON STACK
1694	006220	012746	006250				MOV	#9\$, -(SP) ;PUT RETURN ADDR ON STACK
1695	006224	000006					RTT	;TURN T BIT OFF
1696	006226	012767	006434	172740			MOV	#TST4, \$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
1697	006234	012767	006434	173026			MOV	#TST4, NEXTTST ;SAVE START ADDRESS OF NEXT TEST
1698	006242	012737	006404	000010			MOV	#2\$, 2#RESVEC ;SETUP LOC 10 TO TRAP TO THIS ROUTINE
1699	006250	012706	001100		9\$:		MOV	#STACK, SP ;INITIALIZE STACK
1700	006254	012746	100000				MOV	#BIT15, -(SP) ;SET SIGN BIT ON STACK
1701	006260	012766	100000	177776			MOV	#BIT15, -2(SP) ;SET SIGN BIT AT LOCATION 1074
1702	006266	012705	006324				MOV	#1\$, R5 ;SETUP R5 FOR MARK INSTRUCTION
1703	006272	000240					NOP	;SYNC POINT
1704	006274	006401					MARK	1 ;EXECUTE INSTRUCTION UNDER TEST
1705	006276	022701	006276		8\$:		CMP	#8\$, R1 ;DID MTP OCCUR?
1706	006302	001401					BEQ	3\$;BRANCH IF NO
1707	006304	104011			5\$:		ERROR	11 ;FORK A FAILED TO MTP
1708	006306	013701	001074		3\$:		MOV	2#1074, R1 ;DID MFP OCCUR?
1709	006312	100401					BMI	4\$;BRANCH IF NO
1710	006314	104012					ERROR	12 ;FORK A FAILED TO MFP
1711	006316	012706	001076		4\$:		MOV	#1076, SP ;INITIALIZE SP INCASE MARK CHANGED IT
1712	006322	104013					ERROR	13 ;PCB DID NOT LOAD FROM R5
1713	006324	020627	006302		1\$:		CMP	SP, #5\$-2 ;DID SP GET LOADED PROPERLY?
1714	006330	001410					BEQ	6\$;BRANCH IF YES
1715	006332	010667	172616				MOV	SP, \$REGD ;SAVE SP FOR TYPEOUT
1716	006336	012767	006304	172612			MOV	#5\$, \$REG1 ;STORE ADDRESS OF 5\$ FOR TYPEOUT
1717	006344	012706	001100				MOV	#STACK, SP ;REINITIALIZE SP
1718	006350	104014					ERROR	14 ;MARK DID NOT LOAD SP PROPERLY
1719	006352	012706	001100		6\$:		MOV	#STACK, SP ;RESTORE THE SP
1720	006356	020527	006276				CMP	R5, #8\$;DID R5 GET LOADED PROPERLY?
1721	006362	001424					BEQ	TST4 ;BRANCH IF YES

```

1722 006364 010567 172564          MOV      R5,$REG0          ;SAVE R5 FOR TYPEOUT
1723 006370 013767 006302 172550  MOV      2#5$-2,$REG1      ;STORE ADDRESS OF 5$-2 FOR TYPEOUT
1724 006376 012706 001076          MOV      #1076,$P          ;RESTORE THE SP
1725 006402 104015          ERROR    15                ;RS DID NOT LOAD PROPERLY
1726 006404 012737 000012 000010 2$:  MOV      #12,2#RESVEC      ;RESTORE RESERVED INSTR VECTOR
1727 006412 021627 006276          CMP      (SP),#8$         ;DID PCB GET INCREMENTED BY D67.01?
1728 006416 001003          BNE      7$                ;BRANCH IF NO
1729 006420 012706 001100          MOV      #STACK,$P        ;RESTORE THE SP
1730 006424 104016          ERROR    16                ;FORK A FAILED TO RSD.00
1731 006426 012706 001100 7$:  MOV      #STACK,$P        ;RESTORE THE SP
1732 006432 104017          ERROR    17                ;FORK A FAILED TO D67.01
1733                                     ;*****
1734                                     ;TEST 4          ASH*DMO
1735                                     ;*
1736                                     ;*          IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
1737                                     ;*
1738                                     ;*          IF GRAJ SC05 L DOES NOT GO LOW OR DOES NOT GET THRU TO
1739                                     ;*          RACK BRCA04 L A SHIFT RIGHT WILL OCCUR.
1740                                     ;*          IF THE SHIFT COUNTER DOES NOT SHIFT OR GRAJ SC=0 DOES NOT GO
1741                                     ;*          LOW THE PROCESSOR WILL HANG UP IN STATE ASH.41.
1742                                     ;*
1743                                     ;*          ROM FLOW-52,305,257,166 LEFT SHIFT
1744                                     ;*          52,305,277          RIGHT SHIFT
1745                                     ;*****
1746 006434 000004          TST4:  SCOPE
1747 006436 012767 007176 172624  MOV      #TST5,NEXTTST     ;SAVE ADDRESS OF NEXT TEST
1748 006444 012767 006510 172434  MOV      #19$,$LPADR       ;SETUP LOOP ADR
1749 006452 012767 006510 172430  MOV      #19$,$LPERR       ;SETUP ERROR LOOP
1750 006460 032767 000020 172502  BIT      #BIT4,$TMP3       ;WAS T BIT ON?
1751 006466 001405          BEQ      18$                ;BRANCH IF NO
1752 006470 012746 000360          MOV      #360,-(SP)        ;PUT NEW PSW ON STACK
1753 006474 012746 006502          MOV      #18$,-(SP)       ;PUT RETURN ADR ON STACK
1754 006500 000006          RTT                        ;TURN T BIT ON
1755
1756                                     ;ARITHMETIC LEFT SHIFT
1757 006502 012737 007064 000010 18$:  MOV      #1$,2#RESVEC      ;SET UP FOR FORK A FAILURE
1758 006510 012706 001100          19$:  MOV      #STACK,$P        ;INITIALIZE THE SP
1759 006514 012700 177701          MOV      #177701,$R0       ;PUT SHIFT COUNT IN R0 (+1)
1760 006520 012701 100000          MOV      #BIT15,$R1        ;SET BIT 15 IN REGISTER TO BE SHIFTED
1761 006524 000240          NOP                        ;SYNC POINT
1762 006526 072100          ASH     R0,$R1             ;EXECUTE INSTRUCTION UNDER TEST
1763 006530 103414          BCS     2$                ;BRANCH IF SHIFT WORKED
1764 006532 020127 140000          CMP     R1,#140000        ;DID SHIFT GO IN WRONG DIRECTION?
1765 006536 001002          BNE     3$                ;BRANCH IF NO
1766 006540 104020          ERROR   20                ;SHIFTED RIGHT INSTEAD OF LEFT
1767 006542 000425          BR      9$
1768 006544 020127 100000 3$:  CMP     R1,#BIT15         ;DID R1 SHIFT AT ALL?
1769 006550 001002          BNE     4$                ;BRANCH IF YES
1770 006552 104021          ERROR   21                ;R1 DID NOT SHIFT
1771 006554 000420          BR      9$
1772 006556 104022 4$:  ERROR   22                ;R1 SHIFTED BUT CARRY DID NOT SET
1773                                     ;SHIFT COUNTER COULD BE STUCK
1774 006560 000416          BR      9$
1775 006562 102003 2$:  BVC     10$                ;BRANCH IF V DID NOT SET
1776 006564 100402          BMI     10$                ;BRANCH IF N DID NOT CLEAR
1777 006566 001001          BNE     10$                ;BRANCH IF Z DID NOT SET

```



```

1778 006570 000412          BR      9$          ;CC'S OK
1779 006572 013767 177776 172406 10$: MOV    2#PSW,$ERPSW ;SAVE PSW
1780 006600 042767 177760 172400      BIC    #177760,$ERPSW ;MASK OFF CC'S
1781 006606 012767 000007 172346      MOV    #7,$TMP0      ;PUT EXPECTED CC'S IN STORAGE
1782 006614 104031          ERROR   31          ;STATE ASH.40 DID NOT LOAD CC'S CORRECTLY
1783          ;:*****
1784          ;:ARITHMETIC RIGHT SHIFT TEST
1785 006616 012767 006624 172264 9$: MOV    #64$,$LPERR ;SETUP ERROR LOOP
1786 006624 012701 100001          64$: MOV    #100001,R1 ;SETUP R1 FOR RIGH SHIFT
1787 006630 012700 000077          MOV    #77,R0        ;PUT SHIFT COUNT IN RO (-1)
1788 006634 005002          CLR    R2            ;ENSURE R2 CLEAR
1789 006636 000240          NOP                    ;SYNC POINT
1790 006640 072100          ASH    RO,R1         ;EXECUTE RIGHT SHIFT
1791 006642 005502          ADC    R2            ;SAVE CARRY IN R2
1792 006644 020127 140000          CMP    R1,#140000    ;DID R1 SHIFT IN RIGHT DIRECTION?
1793 006650 001417          BEQ    5$            ;BRANCH IF YES
1794 006652 005701          TST    R1            ;DID R1 SHIFT LEFT INSTEAD OF RIGHT?
1795 006654 001002          BNE    6$            ;BRANCH IF NO
1796 006656 104023          ERROR   23          ;R1 SHIFTED WRONG DIRECTION
1797 006660 000416          BR      8$
1798 006662 022701 040000          6$: CMP    #40000,R1 ;DID SHIFT FAIL TO SIGN FILL?
1799 006666 001001          BNE    7$            ;BRANCH IF NO
1800 006670 104024          ERROR   24          ;SHIFT RIGHT DID NOT SIGN FILL
1801 006672 010167 172260          7$: MOV    R1,$REG1    ;SAVE R1 FOR TYPE OUT
1802 006676 012767 140000 172260      MOV    #140000,$TMP1 ;PUT EXPECTED VALUE IN STORAGE
1803 006704 104025          ERROR   25          ;R1 SHIFTED, BUT DON'T KNOW WHERE
1804 006706 000403          BR      8$
1805 006710 005702          5$: TST    R2            ;DID CARRY GET SET ON SHIFT?
1806 006712 001001          BNE    8$            ;BRANCH IF YES
1807 006714 104026          ERROR   26          ;SHIFT OK BUT CARRY DID NOT LOAD
1808          ;:*****
1809          ;:CONDITION CODE LOAD TEST(STATE ASH.30)
1810 006716 012767 006724 172164 8$: MOV    #63$,$LPERR ;SETUP ERROR LOOP
1811 006724 012701 100000          63$: MOV    #BIT15,R1 ;SETUP RO TO TEST CC'S IN STATE ASH.30
1812          ;RO HAS SHIFT COUNT (-1)
1813          ;SET UP CC'S TO
1814          ;COMPLIMENT OF EXPECTED RESULT
1815          ;AND OSCILLOSCOPE SYNC POINT
1816          ;EXECUTE THE SHIFT
1816 006734 072100          ASH    RO,R1         ;EXECUTE THE SHIFT
1817 006736 013767 177776 172242      MOV    2#PSW,$ERPSW ;SAVE PSW
1818 006744 042767 177760 172234      BIC    #177760,$ERPSW ;MASK OFF THE CC'S
1819 006752 022767 000010 172226      CMP    #10,$ERPSW    ;DID CC'S COME OUT OK?
1820 006760 001404          BEQ    12$           ;BRANCH IF YES
1821 006762 012767 000010 172172      MOV    #10,$TMP0     ;PUT EXPECTED VALUE IN STORAGE
1822 006770 104072          ERROR   72          ;STATE ASH.30 DID NOT LOAD CC'S CORRECTLY
1823          ;:*****
1824          ;:SHIFT COUNT=0
1825 006772 012767 007000 172110 12$: MOV    #62$,$LPERR ;SETUP ERROR LOOP
1826 007000 012701 100000          62$: MOV    #BIT15,R1 ;SET SIGN BIT IN REGISTER TO BE SHIFTED
1827 007004 005000          CLR    RO            ;SET SHIFT COUNT TO ZERO
1828 007006 000263          +SEC!SEV           ;SETUP CC'S TO COMPLIMENT
1829 007010 000250          CLN                    ;OF EXPECTED VALUE
1830          ;AND OSCILLOSCOPE SYNC POINT
1831          ;EXECUTE INSTRUCTION
1831 007012 072100          ASH    RO,R1         ;EXECUTE INSTRUCTION
1832 007014 013767 177776 172164      MOV    2#PSW,$ERPSW ;SAVE PSW
1833 007022 042767 177760 172156      BIC    #177760,$ERPSW ;MASK OFF THE CC'S

```

H06

```

1834 007030 022767 000010 172150      CMP      #10,$ERPSW      ;DID CC'S COME OUT OK?
1835 007036 001416                      BEQ      16$             ;BRANCH IF YES
1836 007040 022701 100000          13$:    CMP      #BIT15,R1      ;DID R1 SHIFT?
1837 007044 001005                      BNE      15$             ;BRANCH IF YES
1838 007046 012767 000010 172106      MOV      #10,$TMP0      ;SAVE EXPECTED VALUE
1839 007054 104027                      ERROR    27             ;STATE ASH.20 DID NOT LOAD CC'S CORRECTLY
1840 007056 000406                      BR       16$
1841 007060 104030          15$:    ERROR    30             ;R1 SHIFTED WHEN NOT SUPPOSE TO
1842 007062 000404                      BR       16$
1843 007064 012737 000012 000010  1$:    MOV      #12,$RESVEC    ;RESTORE RESERVED VECTOR
1844 007072 104032                      ERROR    32             ;FORK A FAILED
1845                                     ;*****
1846                                     ;CONDITION CODE TEST OF STATE ASH.41
1847 007074 012767 007102 172006  16$:    MOV      #61,$SLPERR    ;SETUP ERROR LOOP
1848 007102 012700 000002          61$:    MOV      #2,R0         ;PUT SHIFT COUNT IN R0
1849                                     ;TO TEST STATE ASH.41
1850 007106 012701 060000          MOV      #60000,R1      ;SETUP R1 FOR SHIFT LEFT
1851 007112 000274          +SEN!SEZ              ;SETUP CC'S TO COMPLEMENT
1852 007114 000243          +CLV!CLC              ;OF EXPECTED VALUE
1853                                     ;AND OSCILLOSCOPE SYNC POINT
1854 007116 072100          ASH      R0,R1         ;EXECUTE INSTRUCTION UNDER TEST
1855 007120 013767 177776 172060      MOV      #PSW,$ERPSW    ;SAVE PSW
1856 007126 020127 100000          CMP      R1,#BIT15      ;DID R1 SHIFT CORRECTLY?
1857 007132 001406                      BEQ      17$             ;BRANCH IF YES
1858 007134 010167 172016          MOV      R1,$REG1       ;SAVE R1 FOR TYPEOUT
1859 007140 012767 100000 172016      MOV      #BIT15,$TMP1   ;STORE EXPECTED VALUE
1860 007146 104073                      ERROR    73             ;STATE ASH.41 FAILED
1861 007150 042767 177760 172030  17$:    BIC      #177760,$ERPSW ;MASK OFF CC'S
1862 007156 022767 000013 172022      CMP      #13,$ERPSW     ;DID CC'S LOAD CORRECTLY?
1863 007164 001404                      BEQ      TST5           ;BRANCH IF YES
1864 007166 012767 000013 171766      MOV      #13,$TMP0      ;STORE EXPECTED VALUE
1865 007174 104074                      ERROR    74             ;BAD CC'S DUE TO ASH.41
1866                                     ;*****
1867                                     ;*TEST 5          ASH*DM1
1868                                     ;*
1869                                     ;* IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
1870                                     ;*
1871                                     ;* IF FORK B FAILS EXECUTION WILL EITHER GO TO RSD.00 OR
1872                                     ;* MUL.00.
1873                                     ;* MUL.00 WILL ONLY BE ENTERED IF EITHER B FORK MUX INPUT B2
1874                                     ;* DOES NOT GO HIGH OR THE MUX IS BAD.
1875                                     ;*
1876                                     ;* ROM FLOW-1,175,62,52,305,257
1877                                     ;*****
1878 007176 000004          TST5:  SCOPE
1879 007200 012767 007312 171766      MOV      #TST6,$ESCAPE  ;SAVE START ADDRESS OF NEXT TEST
1880 007206 012767 007312 172054      MOV      #TST6,NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
1881 007214 012706 001100          MOV      #STACK,SP      ;INITIALIZE THE STACK
1882 007220 012737 007260 000010      MOV      #1,$RESVEC     ;SETUP RESERVED VECTOR
1883 007226 012700 001164          MOV      #TMP1,R0       ;PUT ADDRESS OF SHIFT COUNT IN R0
1884 007232 012710 000001          MOV      #1,(R0)        ;SET SHIFT COUNT TO ONE
1885 007236 012701 100000          MOV      #BIT15,R1      ;SETUP REGISTER TO BE SHIFTED
1886 007242 000240          NOP                     ;OSCILLOSCOPE SYNC POINT
1887 007244 072110          ASH      (R0),R1        ;EXECUTE INSTRUCTION UNDER TEST
1888 007246 103421          BCS     TST6           ;TEST OK, GO TO NEXT TEST
1889 007250 005701          TST     R1              ;DID MULTIPLY OCCUR?

```

```

1890 007252 100401      BMI      2$      ;BRANCH IF YES
1891 007254 104033      ERROR    33      ;STATE ASH.00 FAILED
1892 007256 104034      2$:      ERROR    34      ;FORK B FAILED INTO MUL.00
1893 007260 012737 007274 000010 1$:      MOV      #3$,2#RESVEC ;SETUP RESVEC
1894 007266 005000      CLR      RO      ;PUT EVEN ADDRESS IN RO
1895 007270 000240      NOP      ;OSCILLOSCOPE SYNC POINT
1896 007272 072120      ASH      (RO)+,R1 ;EXECUTE DM2
1897 007274 012737 000012 000010 3$:      MOV      #12,2#RESVEC ;RESTORE RESERVED VECTOR
1898 007302 005700      TST      RO      ;DID RO AUTO INCREMENT?
1899 007304 001401      BEQ      4$      ;BRANCH IF NO
1900 007306 104035      ERROR    35      ;FORK B FAILED
1901 007310 104036      4$:      ERROR    36      ;FORK A FAILED
1902 ;*****
1903 ;*TEST 6      ASH*DM2
1904 ;*
1905 ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
1906 ;*
1907 ;*      ALL OTHER LOGIC HAS BEEN TESTED
1908 ;*
1909 ;*      ROM FLOW-2,175,62,52,305
1910 ;*****
1911 007312 000004      †ST6:    SCOPE
1912 007314 012767 007372 171652      MOV      #TST7,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
1913 007322 012767 007372 171740      MOV      #TST7,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
1914 007330 012706 001100      MOV      #STACK,SP      ;INITIALIZE THE SP
1915 007334 012737 007362 000010      MOV      #1$,2#RESVEC   ;PUT ADDRESS OF 1$ IN RESERVED VECTOR
1916 007342 012700 001164      MOV      #STMP1,RO      ;PUT ADDRESS OF SHIFT COUNT IN RO
1917 007346 005010      CLR      (RO)          ;PUT SHIFT COUNT OF ZERO IN $TMP1
1918 007350 072120      ASH      (RO)+,R1      ;EXECUTE INSTRUCTION UNDER TEST
1919 007352 012737 000012 000010      MOV      #12,2#RESVEC   ;RESTORE RESVEC
1920 007360 000404      BR      TST7          ;GO TO NEXT TEST
1921 007362 012737 000012 000010 1$:      MOV      #12,2#RESVEC   ;RESTORE RESERVED VECTOR
1922 007370 104037      ERROR    37          ;FORK A FAILED
1923 ;*****
1924 ;*TEST 7      ASH*DM4
1925 ;*
1926 ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
1927 ;*
1928 ;*      ALL OTHER LOGIC HAS BEEN TESTED.
1929 ;*
1930 ;*      ROM FLOW-4,122,177,62,52,305
1931 ;*****
1932 007372 000004      †ST7:    SCOPE
1933 007374 012767 007456 171572      MOV      #TST10,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
1934 007402 012767 007456 171660      MOV      #TST10,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
1935 007410 012706 001100      MOV      #STACK,SP      ;INITIALIZE THE SP
1936 007414 012737 007446 000010      MOV      #1$,2#RESVEC   ;PUT ADDRESS OF 1$ IN RESERVED VECTOR
1937 007422 012700 001166      MOV      #STMP2,RO      ;PUT ADDRESS OF SHIFT COUNT+2 IN RO
1938 007426 005060 177776      CLR      -2(RO)        ;PUT SHIFT COUNT IN $TMP1
1939 007432 000240      NOP      ;OSCILLOSCOPE SYNC POINT
1940 007434 072140      ASH      -(RO),R1      ;EXECUTE INSTRUCTION UNDER TEST
1941 007436 012737 000012 000010      MOV      #12,2#RESVEC   ;RESTORE RESVEC
1942 007444 000404      BR      TST10         ;GO TO NEXT TEST
1943 007446 012737 000012 000010 1$:      MOV      #12,2#RESVEC   ;RESTORE RESVEC
1944 007454 104040      ERROR    40          ;FORK A FAILED
1945 ;*****

```

J06

```

1946
1947
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959 007456 000004
1960 007460 012767 010300 171602
1961
1962
1963 007466 012700 100001
1964 007472 012701 000001
1965 007476 012702 000077
1966 007502 000262
1967
1968 007504 073002
1969 007506 013767 177776 171472
1970 007514 042767 177760 171464
1971 007522 022767 000011 171456
1972 007530 001014
1973 007532 005701
1974 007534 100017
1975 007536 022700 140000
1976 007542 001427
1977 007544 012767 140000 171410
1978 007552 010067 171376
1979 007556 104041
1980 007560 000420
1981 007562 012767 000011 171372 1$:
1982 007570 104042
1983 007572 000413
1984 007574 012767 140000 171360 2$:
1985 007602 012767 100000 171354
1986 007610 010067 171340
1987 007614 010167 171336
1988 007620 104043
1989
1990
1991 007622 012767 007630 171260
1992 007630 012700 100000
1993 007634 012701 100001
1994 007640 012702 000001
1995 007644 000274
1996
1997 007646 073002
1998 007650 013767 177776 171330
1999 007656 042767 177760 171322
2000 007664 010067 171264
2001 007670 010167 171262
    
```

```

*TEST 10 ASHC*DMO
*
* NEITHER FORK A NOR BEND3 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
*
*****
TST10: SCOPE
MOV #TST11,NEXTTST ;SAVE ADDRESS OF NEXT TEST
;
;ARITHMETIC RIGHT SHIFT COMBINED
MOV #100001,R0 ;SETUP R0 FOR RIGHT SHIFT
MOV #BIT0,R1 ;SETUP R1 FOR RIGH SHIFT
MOV #77,R2 ;PUT SHIFT COUNT (-1) IN R2
SEV ;ENSURE V SET
;AND OSCILLOSCOPE SYNC POINT
ASHC R2,R0 ;EXECUTE INSTRUCTION UNDER TEST
MOV @#PSW,$ERPSW ;SAVE PSW
BIC #177760,$ERPSW ;MASK OFF THE CC'S
CMP #11,$ERPSW ;DID CC'S LOAD PROPERLY?
BNE 1$ ;BRANCH IF NO
TST R1 ;DID R1 GET SIGN BIT SET
BPL 2$ ;BRANCH IF NO
CMP #140000,R0 ;DID R0 SIGN FILL?
BEQ 3$ ;BRANCH IF YES
MOV #140000,$TMP0 ;EXPECTED VALUE
MOV R0,$REG0 ;SAVE R0 FOR TYPEOUT
ERROR 41 ;R0 DID NOT SIGN FILL
BR 3$
MOV #11,$TMP0 ;EXPECTED VALUE
ERROR 42 ;BAD CC'S
BR 3$
MOV #140000,$TMP0 ;GET EXPECTED VALUES
MOV #100000,$TMP1 ;FOR TYPEOUT
MOV R0,$REG0 ;SAVE R0 & R1 FOR TYPEOUT
MOV R1,$REG1
ERROR 43 ;R0<0> DID NOT GO TO R1<15>
*****
;ARITHMETIC LEFT SHIFT
3$: MOV #64,$SLPERR ;SETUP ERROR LOOP
64$: MOV #BIT15,R0 ;SETUP R0 FOR LEFT SHIFT
MOV #100001,R1 ;SETUP R1 FOR LEFT SHIFT
MOV #BIT0,R2 ;PUT SHIFT COUNT (+1) IN R2
+SEZ!SEN ;ENSURE Z AND N SET
;AND OSCILLOSCOPE SYNC POINT
ASHC R2,R0 ;EXECUTE INSTRUCTION UNDER TEST
MOV @#PSW,$ERPSW ;SAVE PSW
BIC #177760,$ERPSW ;MASK OFF THE CC'S
MOV R0,$REG0 ;SAVE REG0
MOV R1,$REG1 ;SAVE REG1
    
```

K06

```

2002 007674 012767 000001 171260 MOV #1,$TMP0 ;GET EXPECTED VALUES
2003 007702 012767 000002 171254 MOV #2,$TMP1 ;OF REGISTERS
2004 007710 022767 000003 171270 CMP #3,$SERPSW ;ARE CC'S CORRECT?
2005 007716 001010 BNE 7$ ;BRANCH IF NO
2006 007720 022701 000002 CMP #2,R1 ;DID R1 SHIFT PROPERLY?
2007 007724 001012 BNE 11$ ;BRANCH IF NO
2008 007726 022700 000001 CMP #1,R0 ;DID R0 SHIFT PROPERLY?
2009 007732 001410 BEQ 12$ ;BRANCH IF YES
2010 007734 104044 ERROR 44 ;R0 DID NOT GET SHIFTED PROPERLY
2011 007736 000406 BR 12$
2012 007740 012767 000003 171214 7$: MOV #3,$TMP0 ;SAVE EXPECTED VALUE
2013 007746 104045 ERROR 45 ;BAD CC'S
2014 007750 000401 BR 12$
2015 007752 104046 11$: ERROR 46 ;R1 DID NOT SHIFT PROPERLY
2016 ;*****
2017 ;NO SHIFT
2018 007754 012767 007762 171126 12$: MOV #63$,$LPERR ;SETUP ERROR LOOP
2019 007762 012701 040000 63$: MOV #40000,R1 ;SETUP R1 FOR NO SHIFT
2020 007766 005002 CLR R2 ;PUT SHIFT COUNT (0) IN R2
2021 007770 012700 100000 MOV #BIT15,R0 ;SET SIGN BIT IN R0 & SET N
2022 007774 000277 SCC ;ENSURE ALL CC'S SET
2023 007776 000250 CLN ;ENSURE N CLEAR
2024 ;AND OSCILLOSCOPE SYNC POINT
2025 010000 073002 ASHC R2,R0 ;EXECUTE INSTRUCTION UNDER TEST
2026 010002 013767 177776 171176 MOV #PSW,$SERPSW ;SAVE PSW
2027 010010 012767 000010 171144 MOV #10,$TMP0 ;SAVE EXPECTED VALUE
2028 010016 042767 177760 171162 BIC #177760,$SERPSW ;MASK OFF THE CC'S
2029 010024 022767 000010 171154 CMP #10,$SERPSW ;DID THE CC'S COME OUT CORRECT?
2030 010032 001401 BEQ 14$ ;BRANCH IF YES
2031 010034 104047 13$: ERROR 47 ;BAD CC'S ON NO SHIFT
2032 ;*****
2033 ;CHECK ROTATE
2034 010036 012767 010044 171044 14$: MOV #62$,$LPERR ;SETUP ERROR LOOP
2035 010044 012701 100000 62$: MOV #BIT15,R1 ;SETUP R1 FOR A ROTATE
2036 010050 010700 MOV PC,R0 ;PUT RANDOM NUMBER IN R0
2037 010052 010067 171104 MOV R0,$TMP0 ;SAVE R0
2038 010056 012702 177761 MOV #-17,R2 ;PUT SHIFT COUNT (-17) IN R2
2039 010062 012767 000001 171074 MOV #1,$TMP1 ;EXPECTED VALUE OF R1 IN $TMP0
2040 010070 000240 NOP ;OSCILLOSCOPE SYNC POINT
2041 010072 073102 ASHC R2,R1 ;EXECUTE INSTRUCTION UNDER TEST
2042 010074 022701 000001 CMP #1,R1 ;DID R1 ROTATE CORRECTLY?
2043 010100 001405 BEQ 15$ ;BRANCH IF YES
2044 010102 010067 171046 MOV R0,$REG0 ;SAVE R0 FOR TYPEOUT
2045 010106 010167 171044 MOV R1,$REG1 ;SAVE R1 FOR TYPEOUT
2046 010112 104050 ERROR 50 ;R1 DID NOT ROTATE PROPERLY
2047 ;*****
2048 ;THE FOLLOWING CODE CHECKS THE BITS IN THE SC
2049 010114 012767 010122 170766 15$: MOV #61$,$LPERR ;SETUP ERROR LOOP
2050 010122 012700 000040 61$: MOV #40,R0 ;SETUP R1 FOR RIGHT SHIFT
2051 010126 012702 000052 MOV #52,R2 ;PUT SHIFT COUNT (-26) IN R2
2052 010132 005001 CLR R1 ;ENSURE R0 CLEAR
2053 010134 005067 171022 CLR $TMP0 ;PUT EXPECTED VALUES OF
2054 010140 005067 171020 CLR $TMP1 ;R0 & R1 IN STORAGE
2055 010144 012767 000001 171014 MOV #1,$TMP2 ;C BIT EXPECTED
2056 010152 000240 NOP ;OSCILLOSCOPE SYNC POINT
2057 010154 073002 ASHC R2,R0 ;EXECUTE SHIFT
    
```

```

2058 010156 013767 177776 171022      MOV      2#PSW,$ERPSW      ;SAVE PSW
2059 010164 103410                    BCS      16$              ;BRANCH IF CORRECT SHIFT
2060 010166 042767 177776 171012      BIC      #177776,$ERPSW   ;MASK OFF C BIT
2061 010174 010067 170754            MOV      RO,$REGO
2062 010200 010167 170752            MOV      R1,$REG1
2063 010204 104051                    ERROR    51              ;STUCK BITS IN SC
2064 010206 012767 010214 170674 16$: MOV      #60$,$LPERR      ;SETUP ERROR LOOP
2065 010214 012701 004000 60$: MOV      #4000,R1      ;SET UP R1 FOR LEFT SHIFT
2066 010220 005000                    CLR      RO              ;ENSURE RO CLEAR
2067 010222 012702 000025            MOV      #25,R2          ;PUT SHIFT COUNT (+25) IN R2
2068 010226 000240                    NOP
2069 010230 073002                    ASHC     R2,RO           ;OSCILLOSCOPE SYNC POINT
2070 010232 013767 177776 170746      MOV      2#PSW,$ERPSW   ;EXECUTE SHIFT
2071 010240 042767 177776 170740      BIC      #177776,$ERPSW ;MASK OFF C BIT
2072 010246 103414                    BCS     TST11           ;GO TO NEXT TEST IF CORRECT SHIFT
2073 010250 012767 000000 170676      MOV      #RO,$REGO      ;SAVE RO FOR TYPEOUT
2074 010256 012767 000001 170672      MOV      #R1,$REG1      ;SAVE R1 FOR TYPEOUT
2075 010264 005067 170672            CLR      $TMP0          ;SAVE EXPECTED
2076 010270 012767 000001 170666      MOV      #1,$TMP1       ;VALUES
2077 010276 104052                    ERROR    52              ;STUCK BITS IN SC
2078
2079 ;*****
2080 ;*TEST 11 ASHC*DM1
2081 ;*
2082 ;* THE ONLY POSSIBLE FAILURES ARE FORK B OR STATE ASC.00.
2083 ;*
2084 ;* IF FORK B FAILS EXECUTION WILL EITHER GO TO RSD.00 OR
2085 ;* ASH.00.
2086 ;* ASH.00 WOULD PERFORM AN ASH INSTRUCTION INSTEAD OF AN ASHC.
2087 ;*
2088 ;* ROM FLOW-1,175,63,53,306,267,227
2089 ;*****
2090 ;*TST11: SCOPE
2091 MOV      #TST12,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
2092 MOV      #TST12,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
2093 MOV      #STACK,SP      ;INITIALIZE THE SP
2094 MOV      #15,2#RESVEC   ;SETUP RESERVED VECTOR
2095 MOV      #TMP2,R2       ;PUT ADDRESS OF SHIFT COUNT IN R2
2096 MOV      #1,RO         ;SETUP RO FOR RIGHT SHIFT
2097 CLR      R1            ;SETUP R1 FOR RIGHT SHIFT
2098 MOV      #77,(R2)      ;PUT SHIFT COUNT (-1) IN TMP2
2099 CLR      $TMP0         ;PUT EXPECTED VALUES OF
2100 MOV      #BIT15,$TMP1   ;RO & R1 IN STORAGE
2101 NOP
2102 ASHC     (R2),RO       ;OSCILLOSCOPE SYNC POINT
2103 BCC      2$            ;EXECUTE INSTRUCTION UNDER TEST
2104 MOV      2#PSW,$ERPSW  ;BRANCH IF FORK B DID NOT FAIL
2105 CLR      $TMP0         ;SAVE PSW FOR TYPEOUT
2106 MOV      RO,$REGO      ;SAVE EXPECTED VALUE
2107 MOV      R1,$REG1
2108 ERROR    53           ;FORK B FAILED TO ASH.00
2109 TST      R1            ;DID R1 SIGN BIT SET?
2110 BMI     TST12         ;BRANCH IF YES
2111 MOV      RO,$REGO      ;SAVE RO & R1
2112 MOV      R1,$REG1      ;FOR TYPEOUT
2113 ERROR    54           ;STATE ASC.00 FAILED
2114 MOV      #12,2#RESVEC  ;RESTORE RESVEC

```

M06

2114	010436	104055			ERROR 55 ;FORK A FAILED TO RSD.00
2115					*****
2116					TEST 12 MUL*DMO
2117					*
2118					* FORK A SHOULD NOT FAIL.
2119					* THE FOLLOWING WOULD BE BEN11 FAILURES:
2120					* IF EITHER GRAD DROO IS STUCK HIGH OR NOT GETTING THRU TO
2121					* RACK E64(C1) OR RACK E64 IS BAD (INPUT C1 FAILED HIGH)
2122					* THE MULTIPPLICAND WILL BE MULTIPLIED BY 177777.
2123					* IF EITHER GRAD DROO IS STUCK LOW OR NOT GETTING THRU TO
2124					* RACK E64(C1) OR RACK E64 IS BAD (INPUT C1 FAILED LOW)
2125					* THE MULTIPLICAND WILL BE MULTIPLIED BY ZERO.
2126					* IF GRAJ SC=0 IS NOT GETTING TO RACK E50(C1) AND RACK E50
2127					* IS BAD (INPUT C1 FAILED LOW) THE MULTIPLICAND WILL ONLY
2128					* BE MULTIPLIED BY BIT 0 OF THE MULTIPLIER.
2129					* IF RACK E50(C1) FAILS HIGH THE PROCESSOR WILL HANG UP IN
2130					* STATE MUL.20(266).
2131					* IF THE INSTRUCTION FAILS TO MULTIPLY CORRECTLY AND ONE
2132					* OF THE ABOVE CONDITIONS CANNOT BE DETERMINED THEN THE
2133					* FAILURE COULD BE IN THE INSTRUCTION DECODE ROM.
2134					*
2135					* IF THE CC'S COME UP BAD THEN THE FAILURE COULD BE IN
2136					* STATES MUL.40,50, OR 60, OR IN THE CONDITION CODE ROM.
2137					*
2138					* ROM FLOW-50,102,266/246,226/206,310
2139					*****
2140	010440	000004			TST12: SCOPE
2141	010442	012767	011002	170620	MOV #TST13,NEXTTST ;SAVE ADDRESS OF NEXT TEST
2142					*
2143					;MULTIPLY 3 TIMES 2
2144	010450	012700	000002		MOV #2,R0 ;PUT MULTIPLICAND IN R0
2145	010454	012702	000003		MOV #3,R2 ;PUT MULTIPLIER IN R2
2146	010460	005001			CLR R1 ;ENSURE R1 CLEAR
2147	010462	005067	170474		CLR \$TMP0 ;PUT EXPECTED VALUE OF
2148	010466	012767	000006	170470	MOV #6,\$TMP1 ;R0 & R1 IN STORAGE
2149	010474	000277			SCC ;MULTIPLY SHOULD CLEAR ALL OF THEM
2150					;AND OSCILLOSCOPE SYNC POINT
2151	010476	070002			MUL R2,R0 ;EXECUTE INSTRUCTION UNDER TEST
2152	010500	013767	177776	170500	MOV #PSW,\$ERPSW ;SAVE PSW
2153	010506	010067	170442		MOV R0,\$REG0 ;SAVE R0 & R1
2154	010512	010167	170440		MOV R1,\$REG1 ;FOR TYPEOUT
2155	010516	020127	000006		CMP R1,#6 ;DID R1 GET LOW PRODUCT
2156	010522	001002			BNE 1\$;BRANCH IF NO
2157	010524	005700			TST R0 ;DID R0 GET UPPER PRODUCT?
2158	010526	001423			BEQ 2\$;BRANCH IF YES
2159	010530	020127	177776	1\$:	CMP R1,#177776 ;DID R0 GET MULTIPLIED BY 177777?
2160	010534	001002			BNE 3\$;BRANCH IF NO
2161	010536	104056			ERROR 56 ;EITHER GRAD DROO STUCK H OR RACK E64 BAD
2162	010540	000430			BR 8\$
2163	010542	005701		3\$:	TST R1 ;DID R1 STAY ZERO?
2164	010544	001401			BEQ 4\$;BRANCH IF YES
2165	010546	000404			BR 5\$;CONTINUE ERROR ANALYSIS
2166	010550	005700		4\$:	TST R0 ;DID R1 & R0 GO TO ZERO?
2167	010552	001002			BNE 5\$;BRANCH IF NO
2168	010554	104057			ERROR 57 ;EITHER GRAD DROO STUCK L OR RACK E64 BAD
2169	010556	000421			BR 8\$


```

2226
2227
2228
2229
2230
2231
2232
2233 011002 000004
2234 011004 012767 011116 170162
2235 011012 012767 011116 170250
2236 011020 012706 001100
2237 011024 012737 011106 000010
2238 011032 012701 000002
2239 011036 012702 001166
2240 011042 012712 000002
2241 011046 005000
2242 011050 005067 170106
2243 011054 012767 000004 170102
2244 011062 000240
2245 011064 070112
2246 011066 020127 000004
2247 011072 001411
2248 011074 010067 170054
2249 011100 010167 170052
2250 011104 104070
2251 011106 012737 000012 000010 15:
2252 011114 104071
2253
2254
2255
2256
2257
2258
2259
2260
2261
2262
2263
2264
2265
2266
2267
2268
2269
2270
2271
2272
2273
2274
2275
2276
2277
2278
2279
2280
2281

```

```

:*
:* 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
:* *****
*ST13: SCOPE
MOV #TST14,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
MOV #TST14,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
MOV #STACK,SP ;INITIALIZE THE SP
MOV #15,$RESVEC ;SETUP RESVEC
MOV #2,R1 ;PUT MULTIPLICAND IN R1
MOV #STMP2,R2 ;PUT ADDRESS OF MULTIPLIER IN R2
MOV #2,(R2) ;PUT MULTIPLIER (+2) IN STMP2
CLR RO ;ENSURE RO CLEAR
CLR $TMPD ;STORE EXPECTED VALUE
MOV #4,$TMP1 ;OF RO & R1
NOP ;OSCILLOSCOPE SYNC POINT
MUL (R2),R1 ;EXECUTE INSTRUCTION UNDER TEST
CMP R1,#4 ;DID MULTIPLY WORK?
BEQ TST14 ;BRANCH IF YES
MOV RO,$REGO ;SAVE RO & R1
MOV R1,$REG1 ;FOR TYPEOUT
ERROR 70 ;STATE MUL.00 FAILED
MOV #12,$RESVEC ;RESTORE RESVEC
ERROR 71 ;FORK B FAILED
:* *****
*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 22(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 RO WILL END UP WITH 177777
:* AND R1 WILL HAVE 177774. IF BEN03 FAILS RO WILL END UP WITH
:* 20 AND R1 WILL HAVE 177774. IF BEN16 FAILS AFTER DVC.00 RO
:* WILL HAVE 2 BUT R1 WILL HAVE 177776.
:*
:* SECTION 3
:* THE NEXT SECTION DIVIDES 6 BY 2 TO TEST BEN16*DRO(1).
:* A FAILURE WILL LEAVE THE REMAINDER (R1)=2 INSTEAD OF ZERO.
:*
:* SECTION 4

```

```

2282
2283
2284
2285
2286
2287
2288
2289
2290
2291
2292
2293
2294
2295
2296
2297
2298
2299
2300
2301
2302
2303
2304
2305
2306
2307
2308
2309
2310
2311
2312
2313
2314
2315 011116 000004
2316 011120 012767 012576 170142
2317
2318 011126 005000
2319 011130 012701 000004
2320 011134 005002
2321 011136 000270
2322
2323 011140 071002
2324 011142 013767 177776 170036
2325 011150 042767 177760 170030
2326 011156 022767 000007 170022
2327 011164 001412
2328 011166 022767 000002 170012
2329 011174 001002
2330 011176 104075
2331 011200 000404
2332 011202 012767 000007 167752 3$:
2333 011210 104076
2334
2335
2336 011212 012767 011220 167670 2$:
2337 011220 012702 000002 64$:

```

```

* 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 10000000000 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.
* *****
* ST14: SCOPE
* MOV #TST15,NEXTTST ;SAVE ADDRESS OF NEXT TEST
* DIVISOR=0 SECTION (ALSO TESTS CONDITION CODE ROM FOR DIV)
* CLR R0 ;PUT DIVIDEND IN
* MOV #4,R1 ;R0 AND R1
* CLR R2 ;MAKE DIVISOR=0
* SEN ;ENSURE N SET
* ;AND OSCILLOSCOPE SYNC POINT
* DIV R2,R0 ;EXECUTE INSTRUCTION UNDER TEST
* MOV #PSW,SERPSW ;SAVE PSW
* BIC #177760,SERPSW ;MASK OFF THE CC'S
* CMP #7,SERPSW ;DID THE CC'S COME OUT OK?
* BEQ 2$ ;BRANCH IF YES
* CMP #2,SERPSW ;DID INSTR GO THROUGH DVE.20?
* BNE 3$ ;BRANCH IF NO
* ERROR 75 ;BEN02 FAILED
* BR 2$
* 3$: MOV #7,STMP0 ;SAVE EXPECTED VALUE
* ERROR 76 ;CC'S BAD IN STATE DVE.00
* *****
* SECTION TWO (ALSO CHECKS CC LOAD OF STATE DVC.70)
* 2$: MOV #64$,SLPERR ;SETUP ERROR LOOP
* 64$: MOV #2,R2 ;PUT DIVISOR IN R2 (R0 & R1 HOLD DIVIDEND)

```

DEKBBC.P11 T14 DIV*DMO

2338	011224	000277			SCC				;ENSURE CC'S SET
2339									;AND OSCILLOSCOPE SYNC POINT
2340	011226	071002			DIV	R2,RO			;EXECUTE INSTRUCTION UNDER TEST
2341	011230	013767	177776	167750	MOV	2#PSW,\$ERPSW			;SAVE PSW
2342	011236	020027	000002		CMP	RO,#2			;DID QUOTIENT COME OUT CORRECT?
2343	011242	001042			BNE	4\$;BRANCH IF NO
2344	011244	005701			TST	R1			;DID REMAINDER COME OUT CORRECT?
2345	011246	001013			BNE	5\$;BRANCH IF NO
2346	011250	042767	177760	167730	BIC	#177760,\$ERPSW			;MASK OFF CC'S
2347	011256	022767	000000	167722	CMP	#0,\$ERPSW			;ARE CC'S CORRECT?
2348	011264	001513			BEQ	7\$;BRANCH IF YES
2349	011266	005067	167670		6\$: CLR	\$TMPO			;SAVE EXPECTED VALUE
2350	011272	104077			ERROR	77			;BAD CC'S DUE TO STATE DVC.70
2351	011274	000507			BR	7\$			
2352	011276	012767	000002	167656	5\$: MOV	#2,\$TMPO			;SAVE EXPECTED
2353	011304	005067	167654		CLR	\$TMP1			;VALUES
2354	011310	020127	177776		CMP	R1,#177776			;DID BEN16*DRO(0) FAIL?
2355	011314	001002			BNE	8\$;BRANCH IF NO
2356	011316	104100			ERROR	100			;RACK E50(B0) STUCK HIGH
2357	011320	000475			BR	7\$			
2358	011322	020127	000004		8\$: CMP	R1,#4			;DID BEN04 FAIL?
2359	011326	001002			BNE	20\$;BRANCH IF NO
2360	011330	104101			ERROR	101			;BEN04 FAILED
2361	011332	000470			BR	7\$			
2362	011334	010067	167614		20\$: MOV	RO,\$REGO			;SAVE RO &
2363	011340	010167	167612		MOV	R1,\$REG1			;R1 FOR TYPEOUT
2364	011344	104102			ERROR	102			;QUOTIENT OK BUT REMAINDER BAD
2365	011346	000462			BR	7\$			
2366	011350	012767	000002	167604	4\$: MOV	#2,\$TMPO			;SAVE EXPECTED
2367	011356	005067	167602		CLR	\$TMP1			;VALUES
2368	011362	022700	000020		CMP	#20,RO			;DID BEN03 FAIL?
2369	011366	001002			BNE	9\$;BRANCH IF NO
2370	011370	104103			ERROR	103			;RACK E49(A1) STUCK LOW
2371	011372	000450			BR	7\$			
2372	011374	022700	077777		9\$: CMP	#77777,RO			;DID BEN04 (-DIV SUB) FAIL?
2373	011400	001002			BNE	10\$;BRANCH IF NO
2374	011402	104104			ERROR	104			;BEN04 STUCK TO DIV SUB
2375	011404	000443			BR	7\$			
2376	011406	022700	177777		10\$: CMP	#-1,RO			;DID RO GET A -1?
2377	011412	001002			BNE	13\$;BRANCH IF NO
2378	011414	104110			ERROR	110			;BEN04 (-N) FAILED
2379	011416	000436			BR	7\$			
2380	011420	020027	000000		13\$: CMP	RO,#0			;DID RO STAY 0?
2381	011424	001406			BEQ	11\$;BRANCH IF YES
2382	011426	010067	167522		MOV	RO,\$REGO			;SAVE RO & R1
2383	011432	010167	167520		MOV	R1,\$REG1			;FOR TYPEOUT
2384	011436	104105			ERROR	105			;INSTRUCTION FAILED
2385	011440	000425			BR	7\$			
2386	011442	020127	155776		11\$: CMP	R1,#155776			;DID BEN16*SR15(0) FAIL?
2387	011446	001002			BNE	12\$;BRANCH IF NO
2388	011450	104106			ERROR	106			;BEN16 FAILED TO DIV.50
2389	011452	000420			BR	7\$			
2390	011454	020127	000004		12\$: CMP	R1,#4			;DID R1 CHANGE?
2391	011460	001406			BEQ	14\$;BRANCH IF NO
2392	011462	010067	167466		MOV	RO,\$REGO			;SAVE RO & R1
2393	011466	010167	167464		MOV	R1,\$REG1			;FOR TYPEOUT

```

DEKBBC.P11      T14      DIV*DMO
2394 011472 104107          ERROR 107          ;CAN'T DETERMINE FAILURE
2395 011474 000407          BR      7$
2396 011476 032767 000001 167502 14$: BIT  #BIT0,$ERPSW ;DID BEN02 FAIL?
2397 011504 001402          BEQ     15$          ;BRANCH IF NO
2398 011506 104111          ERROR 111          ;BEN02 FAILED
2399 011510 000401          BR      7$
2400 011512 104112          15$:  ERROR 112          ;BEN05 FAILED
2401          ;*****
2402          ;SECTION THREE
2403 011514 012767 011522 167366 7$:  MOV  #63$,$LPERR ;SETUP ERROR LOOP
2404 011522 005000          63$: CLR  RO          ;PUT DIVIDEND IN RO
2405 011524 012701 000006          MOV  #6,R1          ;PUT DIVIDEND IN R1 (DIVISOR=2 IN R2)
2406 011530 000240          NOP                    ;OSCILLOSCOPE SYNC POINT
2407 011532 071002          DIV  R2,RO          ;EXECUTE INSTRUCTION UNDER TEST
2408 011534 020127 000002          CMP  R1,#2          ;DID BEN16*DRO(1) FAIL?
2409 011540 001001          BNE   16$          ;BRANCH IF NO
2410 011542 104113          ERROR 113          ;RACK E50(80) STUCK LOW
2411          ;*****
2412          ;SECTION FOUR (ALSO CHECK CC LOAD OF STATE DVC.90)
2413 011544 012767 011552 167336 16$: MOV  #62$,$LPERR ;SETUP ERROR LOOP
2414 011552 005000          62$: CLR  RO          ;
2415 011554 012701 000004          MOV  #4,R1          ;PUT DIVIDEND IN RO & R1
2416 011560 012702 177776          MOV  #-2,R2         ;PUT DIVISOR IN R2
2417 011564 012767 177776 167370          MOV  #-2,$TMP0     ;SAVE EXPECTED
2418 011572 005067 167366          CLR  $TMP1         ;VALUES
2419 011576 000240          NOP                    ;OSCILLOSCOPE SYNC POINT
2420 011600 071002          DIV  R2,RO          ;EXECUTE INSTRUCTION UNDER TEST
2421 011602 013767 177776 167376          MOV  $#PSW,$ERPSW ;SAVE CC'S
2422 011610 020027 177776          CMP  RO,#177776    ;DID DIVIDE WORK?
2423 011614 001413          BEQ   17$          ;BRANCH IF YES
2424 011616 020027 027777          CMP  RO,#27777    ;DID BEN16*SR15(1) FAIL?
2425 011622 001002          BNE   18$          ;BRANCH IF NO
2426 011624 104114          ERROR 114          ;RACK E64(80) STUCK LOW
2427 011626 000432          BR      21$
2428 011630 010067 167320          18$: MOV  RO,$REG0    ;SAVE RO & R1 FOR
2429 011634 010167 167316          MOV  R1,$REG1     ;TYPEOUT
2430 011640 104115          ERROR 115          ;EITHER STATE DVC.20 OR DVC.40 OR DVC.80
2431          ;OR DVC.90 FAILED
2432 011642 000424          BR      21$
2433 011644 020127 000000          17$: CMP  R1,#0          ;DID REMAINDER COME OUT CORRECT?
2434 011650 001406          BEQ   19$          ;BRANCH IF YES
2435 011652 010067 167276          MOV  RO,$REG0     ;SAVE RO & R1
2436 011656 010167 167274          MOV  R1,$REG1     ;FOR TYPEOUT
2437 011662 104116          ERROR 116          ;QUOTIENT OK, REMAINDER BAD
2438 011664 000413          BR      21$
2439 011666 042767 177760 167312          19$: BIC  #177760,$ERPSW ;MASK OF CC'S
2440 011674 022767 000010 167304          CMP  #10,$ERPSW   ;DID CC'S COME OUT CORRECT?
2441 011702 001404          BEQ   21$          ;CONTINUE IF YES
2442 011704 012767 000010 167250          MOV  #10,$TMP0    ;SAVE EXPECTED VALUE
2443 011712 104117          ERROR 117          ;BAD CC'S DUE TO STATE DVC.90
2444          ;*****
2445          ;SECTION FIVE (DIV QUIT CAUSED BY N(0)*SR15(0))
2446 011714 012767 011722 167166 21$: MOV  #61$,$LPERR ;SETUP ERROR LOOP
2447 011722 012700 000002          61$: MOV  #2,RO     ;SETUP RO AND R1
2448 011726 005001          CLR  R1          ;TO CAUSE DIV QUIT ABORT
2449 011730 012702 000001          MOV  #1,R2       ;PUT DIVISOR IN R2

```

```

2450 011734 000277          SCC          ;SETUP CC'S TO COMPLIMENT
2451 011736 000242          CLV          ;OF EXPECTED VALUE
2452                                ;AND OSCILLOSCOPE SYNC POINT
2453 011740 071002          DIV      R2,R0 ;EXECUTE INSTRUCTION UNDER TEST
2454 011742 013767 177776 167236 MOV      Q#PSW,$ERPSW ;SAVE CC'S
2455 011750 020027 000002 CMP      R0,#2      ;DID DIVIDE ABORT?
2456 011754 001402          BEQ      22$      ;BRANCH IF YES
2457 011756 104120          ERROR    120      ;DIV QUIT DID NOT GO LOW
2458 011760 000413          BR      23$
2459 011762 042767 177760 167216 22$: BIC      #177760,$ERPSW ;MASK OFF CC'S
2460 011770 022767 000002 167210 CMP      #2,$ERPSW ;DID CC'S COME OUT OK?
2461 011776 001404          BEQ      23$      ;BRANCH IF YES
2462 012000 012767 000002 167154 MOV      #2,$TMPO   ;STORE EXPECTED CC'S FOR TYPEOUT
2463 012006 104121          ERROR    121      ;CC'S BAD DUE TO EITHER DIV.30 OR DVE.20
2464                                ;*****
2465                                ;SECTION SIX (DIV QUIT CAUSED BY SHFTR=0)
2466 012010 012767 012016 167072 23$: MOV      #60,$SLPERR ;SETUP ERROR LOOP
2467 012016 012700 000002 60$: MOV      #2,R0   ;SETUP RO & R1 TO
2468 012022 005001          CLR      R1       ;CAUSE DIV QUIT TO ABORT
2469 012024 012702 177776          MOV      #-2,R2  ;SETUP DIVISOR (-2)
2470 012030 000240          NOP
2471 012032 071002          DIV      R2,R0   ;OSCILLOSCOPE SYNC POINT
2472 012034 022700 000002          CMP      #2,R0   ;EXECUTE INSTRUCTION UNDER TEST
2473 012040 001401          BEQ      24$      ;DID DIVIDE ABORT?
2474 012042 104122          ERROR    122      ;BRANCH IF YES
2475                                ;GRAJ ES IS BAD (Z2(0)*LEFT SAVE (1))
2476                                ;*****
2477                                ;SECTION SEVEN
2478 012044 012767 012052 167036 24$: MOV      #57,$SLPERR ;SETUP ERROR LOOP
2479 012052 012700 100000 57$: MOV      #BIT15,R0 ;MAKE DIVIDEND
2480 012056 005001          CLR      R1       ;MOST NEGATIVE NUMBER
2481 012060 012702 000001          MOV      #1,R2  ;SETUP DIVISOR
2482 012064 000257          CCC
2483 012066 000264          SEZ
2484 012070 071002          DIV      R2,R0   ;SET CC'S TO COMPLIMENT OF EXPECTED
2485 012072 013767 177776 167106 MOV      Q#PSW,$ERPSW ;OSCILLOSCOPE SYNC POINT
2486 012100 020027 100000          CMP      R0,#BIT15 ;EXECUTE INSTRUCTION UNDER TEST
2487 012104 001402          BEQ      25$      ;SAVE PSW
2488 012106 104123          ERROR    123      ;DID DIVIDE ABORT?
2489 012110 000413          BR      26$      ;BRANCH IF YES
2490 012112 042767 177760 167066 25$: BIC      #177760,$ERPSW ;BEND4*N FAILED
2491 012120 022767 000016 167060 CMP      #16,$ERPSW ;MASK OFF CC'S
2492 012126 001404          BEQ      26$      ;DID CC'S LOAD PROPERLY?
2493 012130 012767 000016 167024 MOV      #16,$TMPO ;BRANCH IF YES
2494 012136 104124          ERROR    124      ;STORE EXPECTED VALUE
2495                                ;CC'S DID NOT LOAD PROPERLY
2496                                ;*****
2497                                ;SECTION EIGHT (DIV QUIT CAUSED BY N(1)*SR15(1))
2498 012140 012767 012146 166742 26$: MOV      #56,$SLPERR ;SETUP ERROR LOOP
2499 012146 012700 177776 56$: MOV      #177776,R0 ;SETUP RO & R1 TO
2500 012152 012701 177777          MOV      #177777,R1 ;CAUSE DIV QUIT TO ABORT
2501 012156 012702 177777          MOV      #177777,R2 ;SETUP DIVISOR (-1)
2502 012162 000240          NOP
2503 012164 071002          DIV      R2,R0   ;OSCILLOSCOPE SYNC POINT
2504 012166 020027 000001          CMP      R0,#1   ;EXECUTE INSTRUCTION UNDER TEST
2505 012172 001401          BEQ      27$      ;DID DIVIDE ABORT LEAVE RO=1
2506 012174 000403          BR      35$      ;BRANCH IF YES
2507 012176 020127 000001 27$: CMP      R1,#1   ;DID DIVIDE ABORT?

```

```

2506 012202 001413          BEQ      28$          ;BRANCH IF YES
2507 012204 012767 000001 166750 35$:  MOV     #1,$TMP0    ;STORE EXPECTED VALUE
2508 012212 010067 166736          MOV     R0,$REG0    ;SAVE R0
2509 012216 012767 000001 166740          MOV     #1,$TMP1
2510 012224 010167 166726          MOV     R1,$REG1    ;SAVE R1
2511 012230 104125          ERROR    125        ;EITHER GRAJ E5 BAD OR MICROSTATE BAD
2512                                     ;*****
2513                                     ;SECTION NINE
2514 012232 012767 012240 166650 28$:  MOV     #55,$LPERR   ;SETUP ERROR LOOP
2515 012240 012700 177777          MOV     #-1,R0      ;PUT DIVIDEND IN
2516 012244 012701 177773          MOV     #-5,R1      ;RO & R1
2517 012250 012702 000002          MOV     #2,R2       ;PUT DIVISOR IN R2
2518 012254 012767 177776 166700          MOV     #-2,$TMP0   ;SAVE EXPECTED VALUE
2519 012262 012767 177777 166674          MOV     #-1,$TMP1   ;SAVE EXPECTED VALUE
2520 012270 000240          NOP
2521 012272 071002          DIV     R2,R0       ;OSCILLOSCOPE SYNC POINT
2522 012274 010067 166654          MOV     R0,$REG0    ;EXECUTE INSTRUCTION UNDER TEST
2523 012300 010167 166652          MOV     R1,$REG1    ;SAVE R0
2524 012304 020127 177777          CMP     R1,#-1      ;SAVE R1
2525 012310 001402          BEQ     29$          ;IS REMAINDER CORRECT?
2526 012312 104126          ERROR    126        ;BRANCH IF YES
2527 012314 000404          BR      30$          ;REMAINDER BAD
2528 012316 022700 177776          29$:  CMP     #-2,R0      ;IS QUOTIENT CORRECT?
2529 012322 001401          BEQ     30$          ;BRANCH IF YES
2530 012324 104127          ERROR    127        ;QUOTIENT IS INCORRECT
2531                                     ;*****
2532                                     ;SECTION TEN
2533 012326 012767 012334 166554 30$:  MOV     #54,$LPERR   ;SETUP ERROR LOOP
2534 012334 012700 177777          MOV     #-1,R0      ;SETUP
2535 012340 012701 177773          MOV     #-5,R1      ;DIVIDEND
2536 012344 012702 177776          MOV     #-2,R2      ;AND DIVISOR
2537 012350 012767 000002 166604          MOV     #2,$TMP0    ;SAVE EXPECTED VALUES
2538 012356 012767 177777 166600          MOV     #-1,$TMP1   ;OF RO & R1
2539 012364 000240          NOP
2540 012366 071002          DIV     R2,R0       ;OSCILLOSCOPE SYNC POINT
2541 012370 010067 166560          MOV     R0,$REG0    ;EXECUTE INSTRUCTION UNDER TEST
2542 012374 010167 166556          MOV     R1,$REG1    ;SAVE RO
2543 012400 020027 000002          CMP     R0,#2       ;SAVE R1
2544 012404 001402          BEQ     31$          ;IS QUOTIENT CORRECT?
2545 012406 104130          ERROR    130        ;BRANCH IF YES
2546 012410 000404          BR      32$          ;QUOTIENT BAD
2547 012412 020127 177777          31$:  CMP     R1,#-1      ;IS REMAINDER CORRECT
2548 012416 001401          BEQ     32$          ;BRANCH IF YES
2549 012420 104131          ERROR    131        ;REMAINDER INCORRECT (QUOT. OK)
2550                                     ;*****
2551                                     ;SECTION ELEVEN
2552 012422 012767 012430 166460 32$:  MOV     #53,$LPERR   ;SETUP ERROR LOOP
2553 012430 012700 177776          53$:  MOV     #177776,R0  ;SETUP DIVIDEND
2554 012434 005001          CLR     R1           ;AND DIVISOR TO GET A
2555 012436 012702 040000          MOV     #40000,R2   ;QUOT OF -10 & REMAINDER=0
2556 012442 012767 177770 166512          MOV     #-10,$TMP0  ;SAVE EXPECTED VALUE
2557 012450 005067 166510          CLR     $TMP1        ;SAVE EXPECTED VALUE
2558 012454 000240          NOP
2559 012456 071002          DIV     R2,R0       ;OSCILLOSCOPE SYNC POINT
2560 012460 010067 166470          MOV     R0,$REG0    ;EXECUTE INSTRUCTION UNDER TEST
2561 012464 010167 166466          MOV     R1,$REG1    ;SAVE RO
                          ;SAVE R1

```



```

2618 012702 001401 BEQ 3$ ;BRANCH IF YES
2619 012704 104136 ERROR 136 ;SP DID NOT INCREMENT
2620 012706 042767 177760 166272 3$: BIC #177760,$ERPSW ;MASK OFF CC'S
2621 012714 022767 000011 166264 CMP #11,$ERPSW ;DID CC'S COME OUT CORRECT?
2622 012722 001410 BEQ TST16 ;BRANCH IF YES
2623 012724 012767 000011 166230 MOV #11,$TMPO ;SAVE EXPECTED VALUE
2624 012732 104137 ERROR 137 ;CC'S BAD (CC ROM)
2625 012734 012737 000012 000010 1$: MOV #12,@#RESVEC ;RESTORE RESVEC
2626 012742 104140 ERROR 140 ;FORK A FAILED
2627 *****
2628 ;*TEST 16 MTP*DM1
2629 *
2630 * IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
2631 * THIS WILL ONLY HAPPEN IF RACF E20(4) IS STUCK HIGH.
2632 *
2633 * THIS TEST ENSURES STATE MTP.10 RELOADS THE
2634 * DR IF THE DESTINATION IS R6 AND THAT IT PUTS THE PC IN THE
2635 * DR IF THE DESTINATION FIELD IS R7.
2636 *
2637 * ROM FLOW-45,151,146,111,155,312
2638 *****
2639 012744 000004 TST16: SCOPE
2640 012746 012767 013116 166314 MOV #TST17,NEXTTST ;SAVE ADDRESS OF NEXT TEST
2641 012754 012737 013106 000010 MOV #15,@#RESVEC ;SETUP RESVEC
2642 012762 012706 001072 MOV #1072,SP ;SETUP THE SP
2643 012766 012716 100000 MOV #BIT15,(SP) ;SET THE SIGN BIT ON THE STACK
2644 012772 005066 000002 CLR 2(SP) ;ENSURE 1074 IS CLEAR
2645 012776 000240 NOP ;OSCILLOSCOPE SYNC POINT
2646 013000 006616 MTPI (SP) ;EXECUTE INSTRUCTION UNDER TEST
2647 013002 005737 001074 TST @#1074 ;DID 1074 GET SIGN BIT SET?
2648 013006 100413 BMI 2$ ;BRANCH IF YES
2649 013010 022706 001074 CMP #1074,SP ;DID MTP.10 FAIL TO RELOAD THE DR?
2650 013014 001002 BNE 3$ ;BRANCH IF NO
2651 013016 104141 ERROR 141 ;STATE MTP.10 FAILED
2652 013020 000406 BR 2$
2653 013022 010667 166126 3$: MOV SP,$REG0 ;SAVE SP
2654 013026 012767 001074 166122 MOV #1074,$REG1 ;SAVE EXPECTED VALUE
2655 013034 104142 ERROR 142 ;DON'T KNOW WHAT HAPPENED
2656 013036 012767 013044 166044 2$: MOV #64,$SLPERR ;SETUP ERROR LOOP
2657 013044 012706 001074 64$: MOV #1074,SP ;SETUP THE SP
2658 013050 000240 NOP ;OSCILLOSCOPE SYNC POINT
2659 013052 006617 MTPI (PC) ;EXECUTE INSTRUCTION UNDER TEST
2660 013054 000000 4$: .WORD 0
2661 013056 005767 177772 TST 4$ ;DID 4$ GET BIT15 SET?
2662 013062 100403 BMI 5$ ;BRANCH IF YES
2663 013064 005067 177764 CLR 4$ ;RESTORE 4$
2664 013070 104143 ERROR 143 ;STATE MTP.10 DID NOT PUT PCB IN DR
2665 013072 005067 177756 5$: CLR 4$ ;RESTORE 4$
2666 013076 012737 000012 000010 MOV #12,@#RESVEC ;RESTORE RESVEC
2667 013104 000404 BR TST17 ;GO TO NEXT TEST
2668 013106 012737 000012 000010 1$: MOV #12,@#RESVEC ;RESTORE RESVEC
2669 013114 104144 ERROR 144 ;FORK A FAILED
2670 *****
2671 ;*TEST 17 MFP*DMO
2672 *
2673 * IF FORK A FAILS EXECUTION WILL GO TO RSD.00.

```



```

2730 013354 000250 CLN ;EXPECTED VALUE (EXCEPT FOR C)
2731 ;AND OSCILLOSCOPE SYNC POINT
2732 013356 006520 MFPI (R0)+ ;EXECUTE INSTRUCTION UNDER TEST
2733 013360 013767 177776 165620 MOV @#PSW,$ERPSW ;SAVE PSW
2734 013366 022706 001074 CMP #1074,SP ;DID THE SP DECREMENT?
2735 013372 001401 BEQ 2$ ;BRANCH IF YES
2736 013374 104151 ERROR 151 ;STATE MFP.00 IS BAD
2737 013376 042767 177760 165602 2$: BIC #177760,$ERPSW ;MASK OFF THE CC'S
2738 013404 022767 000011 165574 CMP #11,$ERPSW ;DID THE CC'S SET CORRECTLY?
2739 013412 001414 BEQ TST21 ;BRANCH IF YES
2740 013414 012767 000011 165540 MOV #11,$TMP0 ;SAVE EXPECTED VALUE
2741 013422 104152 ERROR 152 ;INCORRECT CC'S DUE TO STATE MFP.00
2742 013424 012737 000012 000010 1$: MOV #12,@#RESVEC ;RESTORE RESVEC
2743 013432 022700 001166 CMP #$TMP2,R0 ;DID R0 INCREMENT?
2744 013436 001401 BEQ 3$ ;BRANCH IF YES
2745 013440 104153 ERROR 153 ;FORK A FAILED
2746 013442 104154 3$: ERROR 154 ;FORK B FAILED
2747 ;*****
2748 ;*TEST 21 BPT
2749 ;*
2750 ;* FORK A SHOULD NOT FAIL.
2751 ;* IF THE TRAP VECTOR LOGIC FAILS THE TRAP VECTOR WOULD
2752 ;* COME OUT TO BE 4.
2753 ;* THE ONLY OTHER FAILURE WOULD BE TRP.00.
2754 ;* IF THIS STATE FAILS TO LOAD THE DR THE TRAP VECTOR WILL
2755 ;* BE WHATEVER IS IN R3. IF IT FAILS TO LOAD THE BR THE OLD
2756 ;* PS WILL FAIL TO BE STACKED.
2757 ;*****
2758 013444 000004 †TST21: SCOPE
2759 013446 012767 013560 165520 MOV #TST22,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
2760 013454 012767 013560 165606 MOV #TST22,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
2761 013462 012737 013520 000014 4$: MOV #1$,@#BPTVEC ;SETUP BPT VECTOR
2762 013470 012706 001100 MOV #STACK,SP ;SETUP THE SP
2763 013474 012737 013540 000004 MOV #2$,@#ERRVEC ;SETUP LOCATION 4
2764 013502 012737 013542 000024 MOV #3$,@#24 ;SETUP LOCATION 24
2765 013510 012703 000024 MOV #24,R3 ;SETUP R3
2766 013514 000237 SPL 7 ;SETUP PSW
2767 01351F 000003 BPT ;EXECUTE INSTRUCTION UNDER TEST
2768 01352L 042737 177437 001076 1$: BIC #177437,@#1076 ;MASK OFF PRIORITIES OF OLD PS
2769 013526 022737 000340 001076 CMP #340,@#1076 ;DID OLD PS GET STACKED?
2770 013534 001403 BEQ 5$ ;BRANCH IF YES
2771 013536 104160 ERROR 160 ;STATE TRP.00 FAILED
2772 013540 104161 2$: ERROR 161 ;TRAP VECTOR CAME UP BAD
2773 013542 104162 3$: ERROR 162 ;STATE TRP.00 FAILED
2774 013544 012737 036242 000004 5$: MOV #CPUSPUR,@#ERRVEC ;RESTORE ERR VEC
2775 013552 012737 036226 000014 MOV #$RTRN,@#TBITVEC ;RESTORE T BIT VECTOR
2776 ;*****
2777 ;*ALL THE LOGIC FOR (JMP+JSR)*DMO HAS BEEN TESTED.
2778 ;*****
2779 ;*****
2780 ;*****
2781 ;*TEST 22 BIT TEST OF PIRQ REGISTER
2782 ;*
2783 ;*
2784 ;* IF ONE OF THE BLOCK LEVELS IS STUCK HIGH OR TMCB
2785 ;* PS07(0)'S STUCK HIGH OR PDRD PRIORITY=0 IS STUCK HIGH,

```

```

2786 ;* A PIRQ TRAP LOOP WILL OCCUR WHEN THAT PIR LEVEL IS ENABLED.
2787 ;*
2788 ;* A COUNT PATTERN IS THEN RUN THRU THE REGISTER TO ENSURE THAT
2789 ;* THE ENCODER FUNCTIONS PROPERLY.
2790 ;*****
2791 013560 000004 †ST22: SCOPE
2792 013562 012767 013764 165404 MOV #TST23,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
2793 013570 012767 013764 165472 MOV #TST23,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
2794 013576 012706 001100 MOV #STACK,SP ;INITIALIZE THE SP
2795 013602 012767 013640 165276 MOV #4,$,SLPADR ;SETUP LOOP ADR
2796 013610 012767 013640 165272 MOV #4,$,SLPERR ;SETUP ERROR LOOP
2797 013616 032767 000020 165344 BIT #BIT4,$TMP3 ;WAS T BIT ON?
2798 013624 001405 BEQ 4$ ;BRANCH IF NO
2799 013626 012746 000360 MOV #360,-(SP) ;PUT NEW PSW ON STACK
2800 013632 012746 013640 MOV #4,$,-(SP) ;PUT RETURN ADDR ON STACK
2801 013636 000006 RTT ;TURN T BIT ON
2802 013640 000237 4$: SPL 7 ;SET THE CPU PRIORITY AT 7.
2803 013642 005067 165314 CLR $TMP0 ;SETUP COMPARISON LOCATION
2804 013646 005037 177772 CLR @#PIRQ ;CLEAR PIRQ REGISTER
2805 013652 026737 165304 177772 CMP $TMP0,@#PIRQ ;DID PIRQ CLEAR?
2806 013660 001035 BNE 1$ ;BRANCH IF NO
2807 013662 012700 000177 MOV #177,R0 ;SETUP ITERATION COUNT
2808 013666 012767 001042 165266 MOV #1042,$TMP0 ;SETUP COMPARISON LOCATION
2809 013674 012701 000002 MOV #2,R1 ;SETUP R1
2810 013700 062737 001000 177772 2$: ADD #1000,@#PIRQ ;START COUNT PATTERN
2811 013706 026737 165250 177772 CMP $TMP0,@#PIRQ ;DID REGISTER SET CORRECT?
2812 013714 001017 BNE 1$ ;BRANCH IF NO
2813 013716 120137 177773 CMPB R1,@#PIRQ+1 ;IS PIRQ READY TO GO TO NEXT LEVEL?
2814 013722 001005 BNE 3$ ;BRANCH IF NO
2815 013724 062767 000042 165230 ADD #42,$TMP0 ;INCREMENT ENCODED VALUES IN TEST LOC.
2816 013732 005201 INC R1 ;SETUP R1 FOR ROTATE
2817 013734 006101 ROL R1 ;SET R1 TO NEXT CHECK LEVEL
2818 013736 062767 001000 165216 3$: ADD #1000,$TMP0 ;INC. PIRQ LEVEL IN TEST LOCATION
2819 013744 077023 SOB R0,2$ ;CONTINUE COUNT
2820 013746 005037 177772 CLR @#PIRQ ;ENSURE PIRQ CLEAR
2821 013752 000404 BR TST23 ;GO TO NEXT TEST
2822 013754 013767 177772 165234 1$: MOV @#PIRQ,$EPIRQ ;SAVE PIRQ FOR TYPEOUT
2823 013762 104163 ERROR 163 ;PIRQ REG. FAILED
2824 ;*****
2825 ;*TEST 23 PIR LEVEL 1 INTERRUPT
2826 ;*
2827 ;* IF BEN13 FAILS EXECUTION WOULD GO TO ONE OF THE FOLLOWING:
2828 ;* PUP.00, BRK.20, OR SER.00.
2829 ;* PUP.00 WOULD START THE POWER UP ROUTINE.
2830 ;* BRK.20 WOULD CAUSE A TRAP TO ZERO.
2831 ;* SER.00 WOULD PUT 4 IN THE SP AND PERFORM A RED ZONE TRAP.
2832 ;* IF TMCB PIRQ DOES NOT GET TO DAPE OR IF DAPE TV05*07 DOES NOT
2833 ;* GO HIGH OR DOES NOT GET THRU TO THE ALU A TRAP TO 4 WILL OCCUR.
2834 ;* IF THE INTERRUPT DOESN'T OCCUR AN ATTEMPT IS MADE TO ISOLATE THE
2835 ;* FAILURE.
2836 ;*****
2837 013764 000004 †ST23: SCOPE
2838 013766 012767 014334 165200 MOV #TST24,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
2839 013774 012767 014334 165266 MOV #TST24,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
2840 014002 012737 014272 000024 MOV #10,$,@#PWVREC ;SETUP
2841 014010 012737 014324 000000 MOV #12,$,@#0 ;SETUP LOCATION 0

```

MO7

```

2842 014016 012737 000340 000002      MOV      #PR7,@#2      ;PUT PRIORITY 7 IN 2
2843 014024 012737 014300 000004      MOV      #11$,@#4     ;FAILURE TRAP VECTORS
2844 014032 012737 014250 000240      MOV      #1$,@#PIRQVEC ;SETUP PIRQ VECTOR
2845 014040 032737 000020 177776      BIT      #BIT4,@#PSW  ;IS T BIT ON?
2846 014046 001404          BEQ      13$          ;BRANCH IF NO
2847 014050 012737 000360 000242      MOV      #360,@#PIRQVEC+2 ;SET T BIT IN PIRQ VECTOR
2848 014056 000403          BR      14$          ;
2849 014060 012737 000340 000242 13$: MOV      #PR7,@#PIRQVEC+2 ;SETUP PIRQ VECTOR PSW
2850 014066 012706 001100          14$: MOV      #STACK,SP   ;SETUP THE SP
2851 014072 000230          SPL      0           ;SET PRIORITY LEVEL AT ZERO
2852 014074 052737 001000 177772      BIS      #BIT9,@#PIRQ  ;SET LEVEL ONE
2853 014102 005037 177772          CLR      @#PIRQ      ;CLEAR LEVEL ONE
2854          ;TRY TO GET INTERRUPT AT FET.01 INSTEAD OF TST.10.
2855 014106 012737 014130 000240      MOV      #2$,@#PIRQVEC ;SETUP PIRQ VEC
2856 014114 012737 001000 177772      MOV      #BIT9,@#PIRQ  ;SET LEVEL ONE
2857 014122 005037 177772          CLR      @#PIRQ
2858 014126 000403          BR      3$          ;BRANCH IF NO INTERRUPT
2859 014130 005037 177772 2$: CLR      @#PIRQ    ;CLEAR LEVEL 1
2860 014134 104164          ERROR   164        ;STATE TST.10 HAS BAD BEN FIELD
2861          ;TRY PIR LEVEL 4
2862 014136 012737 014160 000240 3$: MOV      #4$,@#PIRQVEC ;SETUP PIRQ VECTOR
2863 014144 052737 010000 177772      BIS      #BIT12,@#PIRQ ;SET LEVEL 4
2864 014152 005037 177772          CLR      @#PIRQ
2865 014156 000403          BR      5$          ;BRANCH IF NO INTERRUPT
2866 014160 005037 177772 4$: CLR      @#PIRQ    ;CLEAR LEVEL 4
2867 014164 104165          ERROR   165        ;EITHER TMCB E62(1) IS BAD OR SOMETHING
2868          ;IN THE HONOR PIR 1 LOGIC IS BAD.
2869          ;TRY PIR 6
2870 014166 012737 014210 000240 5$: MOV      #6$,@#PIRQVEC ;SETUP PIRQ VECTOR
2871 014174 052737 040000 177772      BIS      #BIT14,@#PIRQ ;SET LEVEL 6
2872 014202 005037 177772          CLR      @#PIRQ
2873 014206 000403          BR      7$          ;BRANCH IF NO INTERRUPT
2874 014210 005037 177772 6$: CLR      @#PIRQ    ;
2875 014214 104166          ERROR   166        ;EITHER TMCB E51(9) OR E55(10-11) OR
2876          ;E62 IS BAD OR TMCA INH BELOW BR6 IS
2877          ;STUCK LOW
2878          ;TRY PIR 7
2879 014216 012737 014240 000240 7$: MOV      #8$,@#PIRQVEC ;SETUP PIRQ VECTOR
2880 014224 052737 100000 177772      BIS      #BIT15,@#PIRQ ;SET LEVEL 7
2881 014232 005037 177772          CLR      @#PIRQ
2882 014236 000403          BR      9$          ;BRANCH IF NO INTERRUPT
2883 014240 005037 177772 8$: CLR      @#PIRQ    ;
2884 014244 104167          ERROR   167        ;TMCA ABOVE BR7 MIGHT BE STUCK LOW
2885 014246 104170          9$: ERROR   170        ;TMCE BRQ CLOCK MIGHT BE STUCK LOW
2886 014250 005037 177772 1$: CLR      @#PIRQ    ;CLEAR LEVEL 1
2887 014254 012737 036242 000004      MOV      #CPUSPUR,@#ERRVEC ;RESTORE ERR VEC
2888 014262 012737 041646 000024      MOV      #SPWRDI,@#PWRVEC ;RESTORE THE POWER VECTOR
2889 014270 000421          BR      TST24       ;GO TO NEXT TEST
2890 014272 005037 177772 10$: CLR      @#PIRQ     ;CLEAR LEVEL 1
2891 014276 104171          ERROR   171        ;BEN 13 FAILED
2892 014300 005037 177772 11$: CLR      @#PIRQ
2893 014304 012737 036242 000004      MOV      #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
2894 014312 012706 001100          MOV      #STACK,SP   ;RESTORE THE SP
2895 014316 005037 177766          CLR      @#CPUERR    ;CLEAR ERROR REG
2896 014322 104172          ERROR   172        ;BEN 13 FAILED OR TRAP VECTOR FAILED
2897 014324 005037 177772 12$: CLR      @#PIRQ    ;CLEAR LEVEL 1
    
```

```

2898 014330 104377          ERROR 377          ;EITHER TMCB E53 IS NOT GOING HIGH
2899 014332 000454          454          ;OR TMCB PIRQ DOES NOT GO LOW. (BEN 13 FAILURE)
2900          ;*****
2901          ;*TEST 24          PIR LEVEL 2 INTERRUPT
2902          ;*
2903          ;*          IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
2904          ;*          THIS WILL ONLY HAPPEN IF TMCB E63(2) IS BAD.
2905          ;*
2906          ;*          IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(2)
2907          ;*          IS BAD OR TMCB HONOR PIR2 IS BEING HELD HIGH.
2908          ;*****
2909 014334 000004          TST24: SCOPE
2910 014336 012767 014472 164724          MOV      #TST25,NEXTTST ;SAVE ADDRESS OF NEXT TEST
2911 014344 012706 001100          MOV      #STACK,SP      ;SETUP THE SP
2912 014350 012737 014404 000000          MOV      #1$,@#0        ;SETUP LOC. 0 TO CATCH BEN 13 FAILURE
2913 014356 012737 014420 000240          MOV      #2$,@#PIRQVEC  ;SETUP PIRQ VECTOR
2914 014364 000231          SPL      1              ;SET CPU PRIORITY TO 1
2915 014366 052737 002000 177772          BIS      #BIT10,@#PIRQ  ;SET LEVEL 2
2916 014374 005037 177772          CLR      @#PIRQ         ;CLEAR LEVEL 2
2917 014400 104174          ERROR   174            ;PIR 2 DID NOT INTERRUPT
2918 014402 000406          BR      2$             ;
2919 014404 005037 177772          1$: CLR      @#PIRQ         ;CLEAR LEVEL 2
2920 014410 012737 036242 000004          MOV      #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
2921 014416 104175          ERROR   175            ;BEN 13 FAILED
2922          ;ENSURE PIR09 & 10 NOT SHORTED
2923 014420 012767 014426 164462          2$: MOV      #64$, $LPERR  ;SETUP ERROR LOOP
2924 014426 005037 177772          64$: CLR      @#PIRQ         ;CLEAR LEVEL 2
2925 014432 012737 014456 000240          MOV      #3$,@#PIRQVEC  ;SETUP VECTOR
2926 014440 000231          SPL      1              ;SET LEVEL 1
2927 014442 052737 001000 177772          BIS      #BIT9,@#PIRQ   ;SET PIR 1
2928 014450 005037 177772          CLR      @#PIRQ         ;CLEAR PIR 1
2929 014454 000406          BR      TST25          ;GO TO NEXT TEST
2930 014456 013767 177772 164532          3$: MOV      @#PIRQ,$EPIRQ ;SAVE PIRQ FOR TYPEOUT
2931 014464 005037 177772          CLR      @#PIRQ         ;CLEAR LEVEL 1
2932 014470 104176          ERROR   176            ;PIR09 & 10 SHORTED
2933          ;*****
2934          ;*TEST 25          PIR LEVEL 3 INTERRUPT
2935          ;*
2936          ;*          IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
2937          ;*          THIS WILL ONLY HAPPEN IF TMCB E63(3) IS BAD.
2938          ;*
2939          ;*          IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(3)
2940          ;*          IS BAD OR TMCB HONOR PIR3 IS BEING HELD HIGH.
2941          ;*****
2942 014472 000004          TST25: SCOPE
2943 014474 012767 014630 164566          MOV      #TST26,NEXTTST ;SAVE ADDRESS OF NEXT TEST
2944 014502 012706 001076          MOV      #1076,SP       ;INITIALIZE THE SP
2945 014506 012737 014556 000240          MOV      #1$,@#PIRQVEC  ;SETUP PIRQ VECTOR
2946 014514 012737 014542 000000          MOV      #2$,@#0        ;SETUP LOCATION 0
2947 014522 000232          SPL      2              ;SET CPU AT LEVEL 2
2948 014524 052737 004000 177772          BIS      #BIT11,@#PIRQ  ;SET LEVEL 3 PIR
2949 014532 005037 177772          CLR      @#PIRQ         ;CLEAR LEVEL 3
2950 014536 104177          ERROR   177            ;INTERRUPT FAILED
2951 014540 000406          BR      1$             ;
2952 014542 005037 177772          2$: CLR      @#PIRQ         ;CLEAR LEVEL 3
2953 014546 012737 036242 000004          MOV      #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4

```

```

2954 014554 104200          ERROR 200          ;BEN 13 FAILED
2955 014556 012767 014564 164324 15:  MOV  #64$,SLPERR ;SETUP ERROR LOOP
2956 014564 005037 177772          64$: CLR  @PIRQ      ;CLEAR LEVEL 3
2957 014570 012737 014614 000240  MOV  #3$,@PIRQVEC ;SETUP PIRQ VECTOR
2958 014576 000232          SPL  2          ;SET CPU PRIORITY AT 2
2959 014600 052737 002000 177772  BIS  #BIT10,@PIRQ ;ENABLE PIR2
2960 014606 005037 177772          CLR  @PIRQ      ;CLEAR LEVEL 2
2961 014612 000406          BR   TST26     ;GO TO NEXT TEST
2962 014614 013767 177772 164374 3$:  MOV  @PIRQ,SEPIRQ ;SAVE PIRQ
2963 014622 005037 177772          CLR  @PIRQ      ;CLEAR IT
2964 014626 104201          ERROR 201          ;LEVEL 2 INTERRUPT WHEN CPU LEVEL
2965                                     ;2 ENABLED.
2966                                     ;*****
2967                                     ;*TEST 26          PIR LEVEL 4 INTERRUPT
2968                                     ;*
2969                                     ;*   IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
2970                                     ;*   THIS WILL ONLY HAPPEN IF TMCB E63(5) IS BAD.
2971                                     ;*
2972                                     ;*   IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(5)
2973                                     ;*   IS BAD OR TMCA HONOR PIR 4 IS BEING HELD HIGH.
2974                                     ;*****
2975 014630 000004          †TST26: SCOPE
2976 014632 012767 015000 164430  MOV  #TST27,NEXTTST ;SAVE ADDRESS OF NEXT TEST
2977 014640 012706 001100          MOV  #STACK,SP    ;INITIALIZE THE SP
2978 014644 012737 014716 000240  MOV  #1$,@PIRQVEC ;SETUP PIRQ VEC
2979 014652 012737 014702 000000  MOV  #2$,@#0     ;SETUP LOCATION 0
2980 014660 000233          SPL  3          ;SET CPU AT 3
2981 014662 052737 010000 177772  BIS  #BIT12,@PIRQ ;ENABLE PIR 4
2982 014670 012737 036242 000004  MOV  #CPUSPUR,@ERRVEC
2983 014676 104202          ERROR 202          ;INTERRUPT DID NOT OCCUR
2984 014700 000406          BR   15
2985 014702 005037 177772          CLR  @PIRQ      ;CLEAR LEVEL 4
2986 014706 012737 036242 000004  MOV  #CPUSPUR,@ERRVEC
2987 014714 104203          ERROR 203          ;BEN 13 FAILED
2988 014716 012767 014724 164164 15:  MOV  #64$,SLPERR ;SETUP ERROR LOOP
2989 014724 005037 177772          64$: CLR  @PIRQ      ;CLEAR LEVEL 4
2990 014730 012737 036242 000004  MOV  #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
2991 014736 012737 014764 000240  MOV  #3$,@PIRQVEC ;SETUP PIRQ VECTOR
2992 014744 000233          SPL  3          ;SET CPU AT LEVEL 3
2993 014746 052737 004000 177772  BIS  #BIT11,@PIRQ ;SET LEVEL 3
2994 014754 005037 177772          CLR  @PIRQ      ;CLEAR LEVEL 3
2995 014760 000237          SPL  7
2996 014762 000406          BR   TST27     ;GO TO NEXT TEST
2997 014764 013767 177772 164224 3$:  MOV  @PIRQ,SEPIRQ ;SAVE PIRQ
2998 014772 005037 177772          CLR  @PIRQ      ;CLEAR IT
2999 014776 104204          ERROR 204          ;LEVEL 3 INTERRUPT WHEN CPU LEVEL 3 ENABLED
3000                                     ;*****
3001                                     ;*TEST 27          PIR LEVEL 5 INTERRUPT
3002                                     ;*
3003                                     ;*   IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
3004                                     ;*   THIS WILL ONLY HAPPEN IF TMCB E63(11) IS BAD.
3005                                     ;*
3006                                     ;*   IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(11)
3007                                     ;*   IS BAD OR TMCA HONOR PIR 5 IS BEING HELD HIGH.
3008                                     ;*****
3009 015000 000004          †TST27: SCOPE

```

```

3010 015002 012767 015154 164260      MOV      #TST30,NEXTTST ;SAVE ADDRESS OF NEXT TEST
3011 015010 012706 001100      MOV      #STACK,SP ;INITIALIZE THE SP
3012 015014 012737 015072 000240      MOV      #15,2#PIRQVEC ;SETUP THE PIRQ VECTOR
3013 015022 012737 015056 000000      MOV      #25,2#0 ;SETUP LOCATION 0
3014 015030 000234      SPL      4 ;SET CPU ST LEVEL 4
3015 015032 052737 020000 177772      BIS      #BIT13,2#PIRQ ;SET LEVEL 5
3016 015040 005037 177772      CLR      2#PIRQ ;CLEAR LEVEL 5
3017 015044 012737 036242 000004      MOV      #CPUSPUR,2#ERRVEC ;RESTORE LOCATION 4
3018 015052 104205      ERROR   205 ;INTERRUPT DID NOT OCCUR
3019 015054 000406      BR      15
3020 015056 005037 177772 25:      CLR      2#PIRQ ;CLEAR LEVEL 5
3021 015062 012737 036242 000004      MOV      #CPUSPUR,2#ERRVEC ;RESTORE LOCATION 4
3022 015070 104206      ERROR   206 ;BEN 13 FAILED
3023 015072 012767 015100 164010 15:      MOV      #645,$LPERR ;SETUP ERROR LOOP
3024 015100 012737 036242 000004 645:      MOV      #CPUSPUR,2#ERRVEC ;RESTORE LOCATION 4
3025 015106 005037 177772      CLR      2#PIRQ ;CLEAR LEVEL 5
3026 015112 012737 015140 000240      MOV      #35,2#PIRQVEC ;SETUP PIRQ VECTOR
3027 015120 000234      SPL      4 ;SET CPU AT LEVEL 4
3028 015122 012737 010000 177772      MOV      #BIT12,2#PIRQ ;SET LEVEL 4
3029 015130 005037 177772      CLR      2#PIRQ ;CLEAR LEVEL 4
3030 015134 000237      SPL      7
3031 015136 000406      BR      TST30 ;GO TO NEXT TEST
3032 015140 013767 177772 164050 35:      MOV      2#PIRQ,$EPIRQ ;SAVE PIRQ
3033 015146 005037 177772      CLR      2#PIRQ ;CLEAR LEVELS
3034 015152 104207      ERROR   207 ;LEVEL 4 INT. WHEN CPU LEVEL 4 ENABLED
3035 *****
3036 ;TEST 30 PIR LEVEL 6 INTERRUPT
3037 ;
3038 ; IF BEN13 FAILS EXECUTION WILL GO TO BRK.20.
3039 ;
3040 ; THIS WILL ONLY HAPPEN IF EITHER TMCB E63(12)
3041 ; IS BAD, OR E61(1) IS BAD.
3042 ;
3043 ; IF THE INTERRUPT DOES NOT OCCUR LEVEL 7 IS TRYED, TO TRY
3044 ; AND ISOLATE THE FAILURE BEFORE TMCB E55(9-8).
3045 *****
3046 015154 000004      TST30: SCOPE
3047 015156 012767 015354 164104      MOV      #TST31,NEXTTST ;SAVE ADDRESS OF NEXT TEST
3048 015164 012706 001100      MOV      #STACK,SP ;INITIALIZE THE SP
3049 015170 012737 015272 000240      MOV      #15,2#PIRQVEC ;SETUP THE PIRQ VECTOR
3050 015176 012737 015256 000000      MOV      #25,2#0 ;SETUP LOCATION 0
3051 015204 000235      SPL      5 ;SET THE CPU AT LEVEL 5
3052 015206 052737 040000 177772      BIS      #BIT14,2#PIRQ ;SET LEVEL 6
3053 015214 012737 036242 000004      MOV      #CPUSPUR,2#ERRVEC ;RESTORE LOCATION 4
3054 015222 012737 015246 000240      MOV      #35,2#PIRQVEC ;SETUP THE PIRQ VECTOR
3055 015230 012737 100000 177772      MOV      #BIT15,2#PIRQ ;SET LEVEL 7
3056 015236 005037 177772      CLR      2#PIRQ ;CLEAR LEVEL 7
3057 015242 104210      ERROR   210 ;FAILURE IS AFTER TMCB E70
3058 015244 000412      BR      15
3059 015246 005037 177772 35:      CLR      2#PIRQ ;CLEAR LEVEL 7
3060 015252 104211      ERROR   211 ;FAILURE IS IN TMCB E70 OR BEFORE
3061 015254 000406      BR      15
3062 015256 012737 036242 000004 25:      MOV      #CPUSPUR,2#ERRVEC ;RESTORE LOCATION 4
3063 015264 005037 177772      CLR      2#PIRQ ;CLEAR LEVEL 6
3064 015270 104212      ERROR   212 ;BEN 13 FAILED
3065 015272 012767 015300 163610 15:      MOV      #645,$LPERR ;SETUP ERROR LOOP

```



```

3066 015300 005037 177772      64$: CLR      2#PIRQ      ;CLEAR LEVEL 6
3067 015304 012737 036242 000004 MOV      #CPUSPUR,2#ERRVEC ;RESTORE LOCATION 4
3068 015312 012737 015340 000240 MOV      #4$,2#PIRQVEC ;SETUP PIRQ VECTOR
3069 015320 000235          SPL      5          ;SET PRIORITY AT 5
3070 015322 012737 020000 177772 MOV      #BIT13,2#PIRQ ;SET LEVEL 5
3071 015330 005037 177772          CLR      2#PIRQ      ;CLEAR LEVEL 5
3072 015334 000237          SPL      7
3073 015336 000406          BR      TST31          ;GO TO NEXT TEST
3074 015340 013767 177772 163650 4$: MOV      2#PIRQ,SEPIRQ ;SAVE PIRQ
3075 015346 005037 177772          CLR      2#PIRQ      ;CLEAR LEVEL 5
3076 015352 104213          ERROR   213          ;LEVEL 5 INTERRUPT WHEN CPU 5 ENABLED

```

```

3077 *****
3078 *TEST 31      PIR LEVEL 7 INTERRUPT
3079 *

```

```

3080 *      IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
3081 *      THIS WILL ONLY HAPPEN IF TMCB E63(6) IS BAD.
3082 *

```

```

3083 *      IF THE INTERRUPT DOES NOT OCCUR THEN EITHER TMCB E70(6)
3084 *      IS BAD OR TMCA HONOR PIR7 IS BEING HELD HIGH.
3085 *****

```

```

3086 TST31: SCOPE

```

```

3087 015354 000004          MOV      #TST32,NEXTTST ;SAVE ADDRESS OF NEXT TEST
3088 015356 012767 015526 163704 MOV      #STACK,SP      ;INITIALIZE THE SP
3089 015364 012706 001100          MOV      #1$,2#PIRQVEC ;SETUP THE PIRQ VECTOR
3090 015370 012737 015446 000240 MOV      #2$,2#0        ;SETUP LOCATION 0
3091 015404 000236          SPL      6          ;SET PRIORITY AT LEVEL 6
3092 015406 052737 100000 177772 BIS      #BIT15,2#PIRQ ;SET LEVEL 7
3093 015414 005037 177772          CLR      2#PIRQ      ;CLEAR LEVEL 7
3094 015420 012737 036242 000004 MOV      #CPUSPUR,2#ERRVEC ;RESTORE LOCATION 4
3095 015426 104214          ERROR   214          ;LEVEL 7 DID NOT INTERRUPT
3096 015430 000406          BR      1$

```

```

3097 015432 005037 177772      2$: CLR      2#PIRQ      ;CLEAR LEVEL 7
3098 015436 012737 036242 000004 MOV      #CPUSPUR,2#ERRVEC ;RESTORE LOCATION 4
3099 015444 104215          ERROR   215          ;BEN 13 FAILED

```

```

3100 015446 012767 015454 163434 1$: MOV      #64$,SLPERR ;SETUP ERROR LOOP
3101 015454 012737 036242 000004 64$: MOV      #CPUSPUR,2#ERRVEC ;RESTORE LOCATION 4
3102 015462 012737 015512 000240 MOV      #3$,2#PIRQVEC ;SETUP PIRQ VECTOR
3103 015470 005037 177772          CLR      2#PIRQ      ;CLEAR LEVEL 7

```

```

3104 015474 000236          SPL      6          ;SET LEVEL 6 IN CPU
3105 015476 012737 040000 177772 MOV      #BIT14,2#PIRQ ;SET PIR 6
3106 015504 005037 177772          CLR      2#PIRQ      ;CLEAR PIR 6

```

```

3107 015510 000406          BR      TST32          ;GO TO NEXT TEST
3108 015512 013767 177772 163476 3$: MOV      2#PIRQ,SEPIRQ
3109 015520 005037 177772          CLR      2#PIRQ      ;CLEAR LEVEL 6
3110 015524 104216          ERROR   216          ;LEVEL 6 INT. WHEN CPU AT 6

```

```

3111 *****
3112 *TEST 32      UNIBUS TIMEOUT
3113 *

```

```

3114 *      IF TMCC ABORT DOES NOT GO HIGH OR DOES NOT GET TO RACA
3115 *      OR IF RACA ZAP DOES NOT GO LOW THE PROCESSOR WILL NOT TRAP TO 4.
3116 *

```

```

3117 *      IF BEN06 FAILS THE STACKED PC WILL BE 160000 INSTEAD OF 1$-2.
3118 *

```

```

3119 *      IF BEN 13 FAILS EITHER TMCC AERF(1) L IS NOT GOING LOW
3120 *      OR TMCB E53(11) IS BAD.
3121 *

```



```

3122 ;* A TEST IS THEN MADE TO ENSURE THAT TMCC PRIORITY CLEAR GOES LOW.
3123 ;*****
3124 015526 000004 TST32: SCOPE
3125 015530 012767 016646 163532 MOV #TST33,NEXTTST ;SAVE ADDRESS OF NEXT TEST
3126 015536 012737 015630 000004 MOV #25,2#ERRVEC ;SETUP TIMEOUT VECTOR
3127 015544 032737 000020 177776 BIT #BIT4,2#PSW ;IS T BIT ON?
3128 015552 001404 BEQ 19$ ;BRANCH IF NO
3129 015554 012737 000360 000006 MOV #360,2#ERRVEC+2 ;SETUP ERROR VEC FOR T BIT
3130 015562 000403 BR 17$ ;
3131 015564 012737 000340 000006 19$: MOV #PR7,2#ERRVEC+2 ;SETUP ERROR VEC WITHOUT T BIT
3132 015572 012767 015600 163310 17$: MOV #12$,SLPERR ;SETUP ERROR LOOP
3133 015600 012706 001100 12$: MOV #STACK,SP ;INITIALIZE THE SP
3134 ;
3135 ;EXECUTE A TIMEOUT ON A DATI
3136 015604 013700 160000 MOV #160000,RO ;EXECUTE TIMEOUT ON DATI
3137 ;FAILURE-DID NOT TIMEOUT OR AERF DID NOT GO LOW
3138 015610 032737 000020 177766 BIT #BIT4,2#CPUERR ;DID TIMEOUT FLAG SET?
3139 015616 001002 BNE 15$ ;BRANCH IF YES
3140 015620 104217 ERROR 217 ;DID NOT TIMEOUT
3141 015622 000423 BR 7$ ;
3142 015624 104220 15: ERROR 220 ;BEN 13 FAILED
3143 015626 000421 BR 7$ ;
3144 015630 022716 160000 25: CMP #160000,(SP) ;DID 160000 GET STACKED?
3145 015634 001003 BNE 16$ ;BRANCH IF NO
3146 015636 104377 ERROR 377 ;EITHER PS RESTORE STUCK HIGH
3147 ;OR RACK E63(80) BAD
3148 015640 000455 455 ;
3149 015642 000413 BR 7$ ;
3150 015644 012737 015672 000004 16$: MOV #7$,2#ERRVEC ;SETUP TIMEOUT VECTOR
3151 015652 012767 015660 163230 MOV #13$,SLPERR ;SETUP ERROR LOOP
3152 015660 012706 001100 13$: MOV #STACK,SP ;INITIALIZE THE SP
3153 ;
3154 ;EXECUTE A TIMEOUT ON A DATO
3155 015664 010037 160000 MOV RO,2#160000 ;EXECUTE TIMEOUT ON DATO
3156 015670 104221 ERROR 221 ;DID NOT TIMEOUT
3157 ;PRIORITY CLEAR ON ABORT
3158 ;ENSURES PIR VECTOR DOES NOT COME IN ON ABORT
3159 015672 000237 7$: SPL 7 ;ENSURE CPU AT LEVEL 7
3160 015674 012767 015744 163206 MOV #15$,SLPERR ;SETUP THE ERROR LOOP
3161 015702 012737 015764 000004 MOV #11$,2#ERRVEC ;SETUP ERRVEC
3162 015710 012737 015772 000240 MOV #9$,2#PIRQVEC ;SETUP PIRQ VECTOR
3163 015716 032737 000020 177776 BIT #BIT4,2#PSW ;IS T BIT ON?
3164 015724 001404 BEQ 18$ ;BRANCH IF NO
3165 015726 012737 000360 000242 MOV #360,2#PIRQVEC+2 ;SETUP PIRQ PSW
3166 015734 000403 BR 15$ ;
3167 015736 012737 000340 000006 18$: MOV #PR7,2#ERRVEC+2 ;SETUP NEW PSW
3168 015744 012706 001100 15$: MOV #STACK,SP ;INITIALIZE THE SP
3169 015750 052737 100000 177772 BIS #BIT15,2#PIRQ ;SET PIR LEVEL 7
3170 015756 000236 SPL 6 ;
3171 015760 005237 160000 INC #160000 ;EXECUTE REFERENCE TO BAD ADDRESS
3172 015764 005037 177772 11$: CLR #PIRQ ;CLEAR PIRQ REGISTER
3173 015770 000404 BR 10$ ;SKIP OVER ERROR CALL
3174 015772 005037 177772 9$: CLR #PIRQ ;CLEAR PIRQ REG
3175 015776 104377 ERROR 377 ;TMCC PRIORITY CLEAR DID NOT GO LOW
3176 016000 000435 435 ;
3177 016002 012737 036242 000004 10$: MOV #CPUSPUR,2#ERRVEC ;RESTORE ERRVEC

```

```

3178 ;CPU ERROR REGISTER BIT 4 TEST
3179 016010 022737 000020 177766      CMP      #BIT4,2#CPUERR ;DID CPU ERROR REG TIMEOUT BIT SET?
3180 016016 001407                    BEQ      5$              ;BRANCH IF YES
3181 016020 013767 177766 163126      MOV      2#CPUERR,$REGO ;SAVE REG FOR TYPEOUT
3182 016026 012767 000020 163126      MOV      #BIT4,$TMPD    ;SAVE EXPECTED VALUE
3183 016034 104255                    ERROR    255            ;CPU ERROR REG FAILED TO SET
3184 016036 005037 177766      5$:    CLR      2#CPUERR      ;CLEAR OUT BIT 4
3185 016042 022737 000000 177766      CMP      #0,2#CPUERR   ;DID REGISTER CLEAR?
3186 016050 001401                    BEQ      PDINTE         ;BRANCH IF YES
3187 016052 104256                    ERROR    256            ;CPU ERROR REG DOES NOT CLEAR
3188
3189 ;*****
3189 ;SBTTL PERIPHERAL DETERMINATOR & INTERRUPT ENABLE ROUTINES
3190 ;*
3191 ;* THIS SECTION OF CODE TRYS TO FIND A DEVICE ON BR5 AND BR6.
3192 ;* WHEN IT FINDS A DEVICE IT PUTS THE ADDRESS OF THAT DEVICES
3193 ;* SUBROUTINE IN A LOCATION.
3194 ;*
3195 ;* THE CODE TO INITIATE AN INTERRUPT SEQUENCE ON CERTAIN
3196 ;* DEVICES IS ALSO HERE. WHEN A TEST REQUIRES AN INTERRUPT ON
3197 ;* A CERTAIN LEVEL IT LOOKS AT INTERX TO DETERMINE IF A
3198 ;* DEVICE IS AVAILABLE AND IF SO DOES A JSR TO THAT DEVICES
3199 ;* INTERRUPT ENABLE ROUTINE.
3199 ;*****
3200 PDINTE:
3201 016054 012767 016646 163112      MOV      #TST33,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
3202 016062 012767 016646 163200      MOV      #TST33,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3203 016070 005767 163004                    TST      $PASS          ;IS THIS FIRST PASS?
3204 016074 001004                    BNE     1$              ;BRANCH IF NO
3205 016076 032737 040000 177570      BIT      #SW14,2#SWR    ;IS SWITCH 14 ON?
3206 016104 001402                    BEQ     2$              ;BRANCH IF NO
3207 016106 000177 163156      1$:    JMP      2#NEXTTST      ;GO TO NEXT TEST
3208 016112 012706 001100      2$:    MOV      #STACK,SP    ;INITIALIZE THE SP
3209 016116 012737 016132 000004      MOV      #3$ ,2#ERRVEC ;SETUP ERROR VECTOR
3210 016124 005777 163110      TST     2#RSCS1        ;IS RS AVAILABLE?
3211 016130 000430                    BR      6$              ;YES
3212 016132 012737 016146 000004      3$:    MOV      #4$ ,2#ERRVEC ;SETUP ERROR VECTOR
3213 016140 005777 163100      TST     2#RPCS1        ;IS RP AVAILABLE?
3214 016144 000431                    BR      7$              ;YES
3215 016146 012737 016162 000004      4$:    MOV      #5$ ,2#ERRVEC ;SETUP ERROR VECTOR
3216 016154 005777 163074      TST     2#TMCS1        ;IS TM AVAILABLE?
3217 016160 000432                    BR      8$              ;YES
3218 016162 012737 016272 000004      5$:    MOV      #10$,2#ERRVEC ;SETUP ERROR VECTOR
3219 016170 005777 163054      TST     2#RKCS1        ;IS RK AVAILABLE?
3220 016174 016767 163050 163026      MOV      RKCS1,INTSST  ;YES,SAVE RK STATUS
3221 016202 016767 163044 163016      MOV      RKVEC,INTSVEC ;SAVE RK VECTOR
3222 016210 000424                    BR      9$              ;EXIT
3223 016212 016767 163022 163010      6$:    MOV      RSCS1,INTSST ;SAVE RS STATUS
3224 016220 016767 163016 163000      MOV      RSVEC,INTSVEC ;SAVE RS VECTOR
3225 016226 000415                    BR      9$              ;EXIT
3226 016230 016767 163010 162772      7$:    MOV      RPCS1,INTSST ;SAVE RP STATUS
3227 016236 016767 163004 162762      MOV      RPVEC,INTSVEC ;SAVE RP VECTOR
3228 016244 000406                    BR      9$              ;EXIT
3229 016246 016767 163002 162754      8$:    MOV      TMCS1,INTSST ;SAVE TM STATUS
3230 016254 016767 162776 162744      MOV      TMVEC,INTSVEC ;SAVE TM VECTOR
3231 016262 012767 016462 162734      9$:    MOV      #INTSSU,INTER5 ;SAVE ADDRESS OF INTER 5 SUBROUTINE
3232 016270 000406                    BR      LEVE6          ;GO CHECK LEVEL 6
3233 016272 032737 000440 177570      10$:   BIT      #440,2#SWR   ;SWITCH 8 OR 5 ON?

```

```

3234 016300 001002          BNE          LEVE6          ;BRANCH IF YES
3235 016302 104400 077030  TYPE          ,EM710
3236
3237 ;FIND OUT WHAT IS AVAILABLE ON LEVEL 6
3238 016306 012737 016356 000004 LEVE6: MOV #15,2#ERRVEC ;SETUP LOCATION 4
3239 016314 005777 162740          TST @LKSTAT ;IS LINE CLOCK AVAILABLE?
3240 016320 012767 016526 162704 MOV #3KW11L,INTER6 ;PUT ADDRESS OF KW11-L SUBROUTINE IN STORAGE
3241 016326 016767 162730 162700 MOV LKVEC,INT6VEC ;SAVE BR 6 INTERR VEC
3242 016334 016767 162720 162674 MOV LKSTAT,INT6ST ;SAVE ADDRESS OF BR 6 STATUS
3243 016342 012737 036242 000004 MOV #CPUSPUR,2#ERRVEC
3244 016350 005037 177766          CLR @#CPUERR ;ENSURE TIMEOUT BIT CLEAR
3245 016354 000534          BR TST33 ;;PERIPHERAL DETERMINATOR FINISHED
3246 016356 104400 077517 1$: TYPE ,EM725
3247
3248 ;LINE CLOCK NOT AVAILABLE, SEE IF PROGRAMMABLE CLOCK AVAILABLE
3249 016362 012737 016432 000004 MOV #25,2#ERRVEC ;SETUP LOCATION 4
3250 016370 005777 162670          TST @PLKSTAT ;IS PROGRAMMABLE AVAILABLE?
3251 016374 012767 016572 162630 MOV #3KW11P,INTER6 ;YES, PUT ADDRESS OF KW11-P SUBROUTINE IN STORAGE
3252 016402 016767 162660 162624 MOV PLKVEC,INT6VEC ;SAVE BR 6 INTERR VEC
3253 016410 016767 162650 162620 MOV PLKSTAT,INT6ST ;SAVE ADDRESS OF BR 6 STATUS
3254 016416 005037 177766          CLR @#CPUERR ;ENSURE TIMEOUT BIT CLEAR
3255 016422 012737 036242 000004 MOV #CPUSPUR,2#ERRVEC ;RESTORE ERROR VECTOR
3256 016430 000506          BR TST33 ;;PERIPHERAL DETERMINATOR FINISHED
3257 016432 012737 036242 000004 2$: MOV #CPUSPUR,2#ERRVEC ;RESTORE LOCATION 4
3258 016440 005037 177766          CLR @#CPUERR ;ENSURE TIMEOUT BIT CLEAR
3259 016444 032737 000500 177570 BIT #500,@#SWR ;SWITCH 6 OR 8 ON?
3260 016452 001002          BNE 3$ ;BRANCH IF YES
3261 016454 104400 077050 3$: TYPE ,EM711 ;TYPE MESSAGE(NO DEVICE AVAILABLE)
3262 016460
3263 016460 000472          BR TST33 ;;PERIPHERAL DETERMINATOR FINISHED
3264 ;THIS CODE SETS UP THE DEVICE ON LEVEL 5 TO INTERRUPT
3265 ;AND IS CALLED BY A JSR PC,@INTERS
3266 016462 011600 INT5SU: MOV (SP),R0 ;SAVE RETURN PC
3267 016464 012777 016516 162534 MOV #ENDBRS,2#INT5VEC ;SETUP LEVEL 5 VECTOR
3268 016472 005001          CLR R1 ;SETUP WAIT COUNTER
3269 016474 012777 000311 162526 MOV #311,2#INT5ST ;SET LEVEL 5 INTERRUPT
3270 016502 005201 2$: INC R1 ;WAIT FOR
3271 016504 001376          BNE 2$ ;INTERRUPT
3272 016506 005077 162516          CLR @INT5ST ;CLEAR INTERRUPT FLAG
3273 016512 062700 000002          ADD #2,R0 ;ADJUST RETURN PC
3274 016516 005077 162506 ENDBRS: CLR @INT5ST ;CLEAR LEVEL 5 STATUS
3275 016522 010046          MOV R0,-(SP) ;PUT RETURN PC ON STACK
3276 016524 000207          RTS PC ;RETURN
3277
3278 ;THIS CODE SETS UP THE KW11-L TO INTERRUPT AND IS CALLED BY A JSR PC,@INTER6
3279 016526 011600 $KW11L: MOV (SP),R0 ;SAVE THE RETURN PC
3280 016530 012777 016564 162476 MOV #15,2#INT6VEC ;SETUP INTERRUPT VECTOR
3281 016536 005001          CLR R1 ;SETUP COUNTER
3282 016540 012777 000100 162470 MOV #BIT6,2#INT6ST ;SET INTERRUPT ENABLE BIT & CLR MONITOR BIT
3283 016546 005201 2$: INC R1 ;WAIT FOR
3284 016550 001376          BNE 2$ ;INTERRUPT
3285 016552 005077 162460          CLR @INT6ST ;CLEAR INTERRUPT BIT
3286 016556 062700 000002          ADD #2,R0 ;ADJUST RETURN PC
3287 016562 000427          BR $ENDBR6 ;RETURN
3288 016564 005077 162446 1$: CLR @INT6ST ;CLEAR INTERRUPT FLAG
3289 016570 000424          BR $ENDBR6 ;RETURN
    
```

```

3290
3291 ; THIS CODE SETS UP THE KW11-P TO INTERRUPT AND IS CALLED BY A JSR PC, @INTER6
3292 016572 011600 $KW11P: MOV (SP),R0 ;SAVE THE RETURN PC
3293 016574 012777 016636 162432 MOV #15,@INT6VEC ;SETUP THE INTERRUPT VECTOR
3294 016602 012737 000001 172544 MOV #1,@PLKC ;SET THE COUNTER FOR ONE COUNT
3295 016610 005001 CLR R1 ;SETUP THE WAIT COUNTER
3296 016612 012777 000105 162416 MOV #105,@INT6ST ;START COUNTER
3297 016620 005201 2S: INC R1 ;WAIT FOR
3298 016622 001376 BNE 2S ;INTERRUPT
3299 016624 005077 162406 CLR @INT6ST ;CLEAR INTERRUPT BIT
3300 016630 062700 000002 ADD #2,R0 ;ADJUST R0 FOR RETURN
3301 016634 000402 BR $ENDBR6 ;RETURN
3302 016636 005077 162374 1S: CLR @INT6ST ;CLEAR INTERRUPT FLAG
3303 016642 010046 $ENDBR6:MOV R0,-(SP) ;PUT RETURN PC ON STACK.
3304 016644 000207 RTS PC ;RETURN
3305

```

```

3306 ;*****
3307 ;*TEST 33 BR LEVEL 4 INTERRUPT
3308 ;*
3309 ;* BEN 13 SHOULD NOT FAIL.
3310 ;* IF THE INTERRUPT DOESN'T OCCUR AN ATTEMPT IS MADE TO
3311 ;* ISOLATE THE FAILURE.
3312 ;*****

```

```

3313 016646 000004 TST33: SCOPE
3314 016650 012767 017136 162412 MOV #TST34,NEXTTST ;SAVE ADDRESS OF NEXT TEST
3315 016656 012767 003706 162220 MOV #D1990,$ICNT ;ADJUST ITERATION COUNT
3316 016664 012767 016672 162214 MOV #14,$SLPADR ;SETUP LOOP ADR
3317 016672 032737 000400 177570 14S: BIT #SW8,@SWR ;SWITCH 8 ON?
3318 016700 001012 BNE 8S ;BRANCH IF YES
3319 016702 032737 000020 177570 BIT #SW4,@SWR ;IS SWITCH 4 ON?
3320 016710 001406 BEQ 8S ;BRANCH IF NO
3321 016712 005767 162162 TST $PASS ;IS THIS FIRST PASS?
3322 016716 001002 BNE 9S ;BRANCH IF NO
3323 016720 104400 077070 TYPE ,EM712 ;TYPE MESSAGE(SKIPPING TEST)
3324 016724 9S:
3325 016724 000504 BR TST34 ;GO TO NEXT TEST
3326 016726 012706 001100 8S: MOV #STACK,SP ;INITIALIZE THE SP
3327 016732 012737 017004 000064 MOV #15,@TPVEC ;SETUP THE TERMINAL INTERRUPT VECTOR
3328 016740 032737 000020 177776 BIT #BIT4,@PSW ;IS T BIT ON?
3329 016746 001404 BEQ 10S ;BRANCH IF NO
3330 016750 012737 000360 000066 MOV #360,@TPVEC+2 ;SETUP TPVEC PSW
3331 016756 000403 BR 11S
3332 016760 012737 000340 000066 10S: MOV #PR7,@TPVEC+2 ;PUT PRIORITY 7 IN NEW PSW
3333 016766 000237 11S: SPL 7 ;SET CPU AT LEVEL 7
3334 016770 005000 CLR R0 ;SETUP R0
3335 016772 006077 162144 ROR @STPS ;GET INTERRUPT ON BR 4
3336 016776 005200 2S: INC R0 ;WAIT AND SEE
3337 017000 001376 BNE 2S ;IF INTERRUPT OCCURS
3338 017002 000403 BR 3S ;NO INTERRUPT
3339 017004 005077 162132 1S: CLR @STPS ;CLEAR INTERRUPT ENABLE
3340 017010 104222 ERROR 222 ;EITHER TMCB PS07(0) IS NOT GETTING TO
3341 ; TMCB E77 OR E77 IS BAD
3342 3S:

```

```

3343 017012 012767 017136 162154 MOV #TST34,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
3344 017020 012767 017136 162242 MOV #TST34,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3345 017026 012737 017132 000064 MOV #45,@TPVEC ;SETUP THE INTERRUPT VECTOR

```

```

3346 017034 005000          CLR      RO          ;SETUP RO
3347 017036 000233          SPL      3          ;SET CPU AT LEVEL 3
3348 017040 006077 162076   ROR      2$TPTS     ;GET A BR 4
3349 017044 005200          INC      RO          ;WAIT FOR
3350 017046 001376          BNE      5$         ;INTERRUPT
3351 017050 005077 162066   CLR      2$TPTS     ;CLEAR TP INTERRUPT BIT
3352          ;BR 4 INTERRUPT FAILED. IS THERE A BR 6 DEVICE AVAILABLE?
3353 017054 005767 162152   TST     INTER6     ;TEST LEVEL 6
3354 017060 001422          BEQ      7$         ;
3355 017062 016700 162146   MOV     INT6VEC,RO ;GET ADDR OF INTER 6 VECTOR
3356 017066 062700 000002   ADD     #2,RO      ;ADJUST TO PSW VEC
3357 017072 032737 000020 177776   BIT     #BIT4,2$PSW ;IS T BIT ON?
3358 017100 001403          BEQ     12$        ;BRANCH IF NO
3359 017102 012710 000360   MOV     #360,(RO)  ;SETUP PSW
3360 017106 000402          BR      13$        ;
3361 017110 012710 000340   12$: MOV     #PR7,(RO) ;SETUP PSW NO T BIT
3362 017114 000235          13$: SPL     5          ;SET CPU AT 5
3363 017116 004777 162110   JSR     PC,2$INTER6 ;EXECUTE INTERRUPT
3364 017122 000402          BR      6$         ;RETURN HERE IF 6 INTERRUPTS
3365 017124 104223          ERROR   223        ;BOTH BR 4 AND BR 6 FAILED
3366 017126 104224          7$:  ERROR   224        ;BR 4 FAILED
3367 017130 104225          6$:  ERROR   225        ;BR 4 FAILED BUT BR 6 OK
3368 017132 005077 162004   4$:  CLR      2$TPTS ;CLEAR PRINTER INTERRUPT FLAG
3369
3370          ;*****
3371          ;*TEST 34          BR LEVEL 5 INTERRUPT
3372          ;*
3373          ;*          THE ONLY POSSIBLE FAILURE IS THAT TMCA HONOR BR 5
3374          ;*          DOES NOT GO LOW OR TMCB E62(6) IS BAD.
3375          ;*****
3376 017136 000004          TST34: SCOPE
3377 017140 012767 017266 162026   MOV     #TST35,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
3378 017146 012767 017266 162114   MOV     #TST35,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3379 017154 032737 000400 177570   BIT     #SW8,2$SWR   ;SWITCH 8 ON?
3380 017162 001012          BNE     1$         ;BRANCH IF YES
3381 017164 032737 000040 177570   BIT     #SW5,2$SWR   ;IS SWITCH 5 UP?
3382 017172 001406          BEQ     1$         ;BRANCH IF NO
3383 017174 005767 161700   TST     $PASS      ;IS THIS FIRST PASS?
3384 017200 001002          BNE     3$         ;BRANCH IF NO
3385 017202 104400 077115   TYPE    ,EM713     ;TYPE MESSAGE(TEST BEING SKIPPED)
3386 017206          3$:
3387 017206 000427          BR      TST35      ;GO TO NEXT TEST
3388 017210 005767 162010   1$:  TST     INTER5   ;IS THERE A DEVICE AVAILABLE?
3389 017214 001424          BEQ     TST35      ;BRANCH IF NO
3390 017216 012706 001100   2$:  MOV     #STACK,SP ;INITIALIZE THE SP
3391 017222 016700 162000   MOV     INT5VEC,RO ;GET ADDR OF BR 5 VECTOR
3392 017226 062700 000002   ADD     #2,RO      ;ADJUST TO PSW ADDR
3393 017232 032737 000020 177776   BIT     #BIT4,2$PSW ;IS T BIT ON?
3394 017240 001403          BEQ     5$         ;BRANCH IF NO
3395 017242 012710 000360   MOV     #360,(RO)  ;PUT NEW PSW IN VECTOR
3396 017246 000402          BR      6$         ;
3397 017250 012710 000340   5$:  MOV     #PR7,(RO) ;PUT NEW PSW IN VEC NO T BIT
3398 017254 000234          6$:  SPL     4          ;SET CPU AT LEVEL 4
3399 017256 004777 161742   JSR     PC,2$INTERS ;EXECUTE LEVEL 5 INTERRUPT
3400 017262 000401          BR      TST35      ;GO TO NEXT TEST
3401 017264 104226          ERROR   226        ;BR 5 DID NOT INTERRUPT

```

```

3402
3403
3404
3405
3406
3407
3408
3409 017266 000004
3410 017270 012767 017416 161676
3411 017276 012767 017416 161764
3412 017304 032737 000400 177570
3413 017312 001012
3414 017314 032737 000100 177570
3415 017322 001406
3416 017324 005767 161550
3417 017330 001002
3418 017332 104400 077142
3419 017336
3420 017336 000427
3421 017340 005767 161666
3422 017344 001424
3423 017346 012706 001100
3424 017352 016700 161656
3425 017356 062700 000002
3426 017362 032737 000020 177776
3427 017370 001403
3428 017372 012710 000360
3429 017376 000402
3430 017400 012710 000340
3431 017404 000235
3432 017406 004777 161620
3433 017412 000401
3434 017414 104227
3435
3436
3437
3438
3439
3440
3441
3442
3443
3444
3445
3446
3447
3448
3449
3450
3451
3452
3453
3454
3455
3456 017416 000004
3457 017420 012767 017464 161460

```

```

*****
*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.
*****

```

```

TST35: SCOPE
MOV #TST36,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
MOV #TST36,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
BIT #SW8,@#SWR ;SWITCH 8 ON?
BNE 1$ ;BRANCH IF YES
BIT #SW6,@#SWR ;IS SWITCH 6 UP?
BEQ 1$ ;BRANCH IF NO
TST $PASS ;IS THIS FIRST PASS?
BNE 3$ ;BRANCH IF NO
TYPE ,EM715 ;TYPE MESSAGE(TEST BEING SKIPPED)

3$: BR TST36 ;GO TO NEXT TEST
1$: TST INTER6 ;IS THERE A DEVICE AVAILABLE?
BEQ TST36 ;BRANCH IF NO
2$: MOV #STACK,SP ;INITIALIZE THE SP
MOV INT6VEC,RO ;GET ADR OF BR 6 VECTOR
ADD #2,RO ;ADJUST TO PSW ADDR
BIT #BIT4,@#PSW ;IS T BIT ON?
BEQ 5$ ;BRANCH IF NO
MOV #360,(RO) ;SETUP NEW PSW
BR 6$

5$: MOV #PR7,(RO) ;SETUP NEW PSW
6$: SPL 5 ;SET CPU AT LEVEL 5
JSR PC,@INTER6 ;EXECUTE LEVEL 6 INTERRUPT
BR TST36 ;GO TO NEXT TEST
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.
*****

```

```

TST36: SCOPE
MOV #11$,$LPADR ;SETUP LP ADR

```

K08

PDP 11/70 CPU DIAGNOSTIC PART 2 MACY11 27(732) 21-DEC-76 16:00 PAGE 66
 DEKBBC.P11 T36 YELLOW ZONE TRAP

SEQ 0101

```

3458 017426 012767 017464 161454      MOV      #11$, $LPERR      ; SETUP ERROR LOOP
3459 017434 013767 177776 161526      MOV      @#PSW, $TMP3     ; SAVE PSW
3460 017442 032737 000020 177776      BIT      #BIT4, @#PSW    ; IS T BIT ON?
3461 017450 001405                BEQ      11$              ; BRANCH IF NO
3462 017452 012746 000340                MOV      #PR7, -(SP)     ; PUT NEW PSW ON STACK
3463 017456 012746 017464                MOV      #11$, -(SP)    ; PUT RETURN ADR ON STACK
3464 017462 000006                RTT                      ; TURN T BIT OFF
3465 017464 012737 000340 000006 11$:    MOV      #PR7, @#ERRVEC+2 ; RESTORE ERR VEC PSW
3466 017472 012737 017530 000004        MOV      #1$, @#ERRVEC   ; SETUP ERRVEC
3467 017500 012767 017506 161402        MOV      #2$, $LPERR     ; SETUP ERROR LOOP
3468 017506 012706 000376                MOV      #376, SP       ; SETUP THE SP
3469 017512 012700 177777                MOV      #-1, R0        ; SETUP R0
3470 017516 010016                MOV      R0, (SP)       ; EXECUTE INSTRUCTION UNDER TEST
3471 017520 012706 001100                MOV      #STACK, SP     ; REINITIALIZE THE SP
3472 017524 104230                ERROR    230            ; YELLOW ZONE DID NOT OCCUR
3473 017526 000407                BR       8$              ;
3474 017530 022737 177777 000376 1$:    CMP      #-1, @#376     ; DID RED ZONE OCCUR?
3475 017536 001403                BEQ      8$              ; BRANCH IF NO
3476 017540 012706 001100                MOV      #STACK, SP     ; RESTORE SP
3477 017544 104254                ERROR    254            ; RED ZONE IN YELLOW REGION
3478
3479                ; JSR WITH A BAD SP
3480 017546 012737 017606 000004 8$:    MOV      #6$, @#ERRVEC   ; SETUP ERRVEC
3481 017554 012767 017562 161326        MOV      #63$, $LPERR   ; SETUP THE ERROR LOOP
3482 017562 012706 000376                MOV      #376, SP       ; SETUP THE SP
3483 017566 004767 000000                JSR      PC, 7$         ; EXECUTE INSTRUCTION UNDER TEST
3484 017572 012706 001100                MOV      #STACK, SP     ; RESET THE SP
3485 017576 012737 036242 000004        MOV      #CPUSPUR, @#ERRVEC ; RESTORE ERRVEC
3486 017604 104232                ERROR    232            ; TMCC KERNAL R6 DID NOT GO HIGH ON JSR
3487
3488                ; DISABLE WITH STACK LIMIT REGISTER
3489 017606 012737 017634 000004 6$:    MOV      #3$, @#ERRVEC   ; SETUP ERRVEC
3490 017614 012767 017622 161266        MOV      #62$, $LPERR   ; SETUP ERROR LOOP
3491 017622 012706 000740                MOV      #740, SP       ; SETUP THE SP
3492 017626 013716 000742                MOV      @#742, (SP)    ; EXECUTE INSTRUCTION UNDER TEST
3493 017632 000406                BR       4$              ; GO TO NEXT SECTION
3494 017634 012737 036242 000004 3$:    MOV      #CPUSPUR, @#ERRVEC ; RESTORE ERRVEC
3495 017642 012706 001100                MOV      #STACK, SP     ; RESTORE THE SP
3496 017646 104233                ERROR    233            ; PDRC STACK LIMIT DID NOT DISABLE TRAP
3497
3498                ; DISABLE WITH TMCC KERNAL R6
3499                ; DISABLE KERNAL R6 WITH DATI
3500 017650 012737 017674 000004 4$:    MOV      #5$, @#ERRVEC   ; SETUP ERRVEC
3501 017656 012767 017664 161224        MOV      #61$, $LPERR   ; SETUP THE ERROR LOOP
3502 017664 012706 000376                MOV      #376, SP       ; SETUP THE SP
3503 017670 011606                MOV      (SP), SP       ; EXECUTE INSTRUCTION UNDER TEST
3504 017672 000415                BR       9$              ; GO TO NEXT SECTION
3505 017674 012706 001100                MOV      #STACK, SP     ; RESTORE THE SP
3506 017700 012737 036242 000004        MOV      #CPUSPUR, @#ERRVEC ; RESTORE ERRVEC
3507 017706 104234                ERROR    234            ; TMCC KERNAL R6 DID NOT DISABLE ON DATI
3508 017710 000406                BR       9$              ;
3509
3510                ; USED FOR ERROR LOOP IF BIT 3 IN ERROR REG
3511                ; DOES NOT SET
3512 017712 012706 000376                MOV      #376, SP       ; SETUP THE SP
3513 017716 012737 017726 000004        MOV      #9$, @#ERRVEC  ; SETUP ERROR VECTOR

```



```

3514 017724 005016          CLR      (SP)          ;CAUSE TRAP
3515 017726 012767 017712 161154 9$:  MOV      #60$, $LPERR ;SETUP ERROR LOOP
3516 017734 012706 001100          MOV      #STACK, SP   ;RESTORE THE SP
3517 017740 022737 000010 177766    CMP      #BIT3, @#CPUERR ;DID YEL ZONE BIT SET?
3518 017746 001407          BEQ      10$          ;BRANCH IF YES
3519 017750 013767 177766 161176    MOV      @#CPUERR, $REGO ;SAVE FOR TYPEOUT
3520 017756 012767 000010 161176    MOV      #BIT3, $TMPD   ;SAVE EXPECTED VALUE
3521 017764 104257          ERROR    257          ;YEL ZONE BIT DID NOT SET IN CPUERR
3522 017766 012767 017774 161114 10$:  MOV      #57$, $LPERR ;SETUP ERROR LOOP
3523 017774 005037 177766    57$:  CLR      @#CPUERR     ;CLEAR YEL ZONE BIT
3524 020000 005737 177766    TST      @#CPUERR     ;DID IT CLEAR?
3525 020004 001401          BEQ      TST37        ;BRANCH IF YES
3526 020006 104260          ERROR    260          ;BIT3 DID NOV CLEAR IN CPUERR
3527
3528 ;*****
3529 ;*TEST 37 ROM FIELD CHECK OF PC MANIPULATOR STATES
3530 ;*
3531 ;* THIS TEST EXECUTES THE MACHINE STATES THAT MANIPULATE
3532 ;* THE PC TO ENSURE THAT THE PCB ROM FIELD OR DRX ROM FIELD
3533 ;* OR SRX ROM FIELD OF THESE STATES IS FUNCTIONAL.
3534 ;* THESE STATES ARE S13.00, S45.00, MTP.10, D45.80, D45.90,
3535 ;* D45.00, AND D45.01.
3536 ;*****
3536 020010 000004          †ST37: SCOPE
3537 020012 012767 020042 161066    MOV      #18$, $LPADR  ;SETUP LOOP ADR
3538 020020 032767 000020 161142    BIT      #BIT4, $TMP3  ;WAS T BIT ON?
3539 020026 001405          BEQ      18$          ;BRANCH IF NO
3540 020030 012746 000360    MOV      #360, -(SP)   ;PUT NEW PSW ON STACK
3541 020034 012746 020042    MOV      #18$, -(SP)  ;PUT RETURN ADDR ON STACK
3542 020040 000006          RTT                    ;TURN T BIT ON
3543
3544 020042 012767 020050 161040    ;BIN*SM1*SF7*DM0(S13.00)
3545 020050 011700          18$:  MOV      #64$, $LPERR ;SETUP ERROR LOOP
3546 020052 020027 020027    64$:  MOV      (PC), RO    ;EXECUTE INSTRUCTION UNDER TEST
3547 020056 001402          CMP      RO, #20027   ;DID TEST WORK
3548 020060 104377          BEQ      1$          ;BRANCH IF YES
3549 020062 000443          ERROR    377          ;STATE S13.00 FAILED
3550          443
3551 020064 012767 020072 161016    ;BIN*SM4*SF7*DM2*DF7 (S45.00)
3552 020072 012767 024727 000000    1$:  MOV      #63$, $LPERR ;SETUP ERROR LOOP
3553 020100 024727          63$:  MOV      #24727, 2$ ;PUT INSTRUCTION IN 2$
3554 020102 000240          2$:  CMP      -(PC), (PC)+ ;EXECUTE INSTRUCTION UNDER TEST
3555 020104 001402          NOP                    ;USED TO BE SAFE
3556 020106 104377          BEQ      3$          ;BRANCH IF TEST OK
3557 020110 000444          ERROR    377          ;STATE S45.00 FAILED
3558          444
3559 020112 012767 020120 160770    ;MTP*DM2*DF7 (MTP.10)
3560 020120 012706 001100          3$:  MOV      #62$, $LPERR ;SETUP ERROR LOOP
3561 020124 012746 177777          62$:  MOV      #STACK, SP ;INITIALIZE THE SP
3562 020130 005067 000002          MOV      #-1, -(SP)  ;SET 1076 TO ALL ONES
3563 020134 006627          CLR      4$          ;ENSURE 4$ CLEAR
3564 020136 000000          MTPI    (PC)+       ;EXECUTE INSTRUCTION UNDER TEST
3565 020140 022767 177777 177770    4$:  .WORD    0
3566 020146 001402          CMP      #-1, 4$    ;DID INSTRUCTION WORK?
3567 020150 104377          BEQ      7$          ;BRANCH IF YES
3568 020152 000445          ERROR    377          ;STATE MTP.10 FAILED
3569          445
;BIN*SM2*SF7*DM4*DF7 (D45.80)

```


M08

```

3570 020154 012767 020162 160726 7$: MOV #61$, $LPERR ; SETUP ERROR LOOP
3571 020162 012747 61$: MOV (PC)+, -(PC) ; EXECUTE INSTRUCTION UNDER TEST
3572 020164 000402 BR 10$ ; WILL EXECUTE THIS IF INSTR. WORKS
3573 020166 104377 ERROR 377 ; STATE D45.80 FAILED
3574 020170 000447 447
3575 ; BIN*SM1*SFO*DM4*DF7 (D45.90)
3576 020172 012767 020200 160710 10$: MOV #50$, $LPERR ; SETUP ERROR LOOP
3577 020200 012767 141047 000024 60$: MOV #141047, 11$ ; SETUP INSTRUCTION TO EXECUTE
3578 020206 012700 001162 MOV #STMP0, R0 ; PUT ADDRESS OF STMP0 IN R0
3579 020212 005200 INC R0
3580 020214 112710 000002 MOVB #BIT1, (R0) ; SET BIT1 IN STMP1 HIGH BYTE
3581 020220 012705 001166 MOV #STMP2, R5 ; SETUP R5
3582 020224 012767 001000 160732 11$: MOV #BIT9, $TMP1 ; SETUP $TMP1
3583 020232 141047 BICB (R0), -(PC) ; EXECUTE INSTRUCTION UNDER TEST
3584 020234 005767 160724 TST $TMP1 ; DID $TMP1 GET CLEARED?
3585 020240 001402 BEQ 12$ ; BRANCH IF YES
3586 020242 104377 ERROR 377 ; STATE D45.00 FAILED
3587 020244 000450 450
3588 ; DAC*DM4*DF7 (D45.00)
3589 020246 012767 020254 160634 12$: MOV #57$, $LPERR ; SETUP ERROR LOOP
3590 020254 012767 140047 000016 57$: MOV #140047, 13$ ; SETUP INSTRUCTION TO EXECUTE
3591 020262 012700 000002 MOV #BIT1, R0 ; SETUP R0 TO CHANGE DR FROM 7 TO 5
3592 020266 012705 001165 MOV #STMP1+1, R5 ; PUT ADDRESS OF $TMP1 HIGH BYTE IN R5
3593 020272 012767 000002 160664 13$: MOV #BIT1, $TMP1 ; SET UP $TMP1 SO INSTRUCTION CLEARS IT
3594 020300 140047 BICB R0, -(PC) ; EXECUTE INSTRUCTION UNDER TEST
3595 020302 005767 160656 TST $TMP1 ; DID $TMP1 CLEAR?
3596 020306 001402 BEQ 14$ ; BRANCH IF YES
3597 020310 104377 ERROR 377 ; STATE D45.90 FAILED
3598 020312 000451 451
3599 ; DAC*DM5*DF7 (D45.01)
3600 020314 012767 020322 160566 14$: MOV #56$, $LPERR ; SETUP ERROR LOOP
3601 020322 012767 130057 000052 56$: MOV #130057, 15$ ; SETUP INSTRUCTION TO EXECUTE
3602 020330 032737 000020 177776 BIT #BIT4, @PSW ; IS T BIT ON?
3603 020336 001404 BEQ 20$ ; BRANCH IF NO
3604 020340 012737 000360 000066 MOV #360, @TPVEC+2 ; SETUP TP VEC PSW
3605 020346 000403 BR 17$
3606 020350 012737 000340 000066 20$: MOV #PR7, @TPVEC+2
3607 020356 012737 020416 000064 17$: MOV #16$, @TPVEC ; SETUP BR4 INTERRUPT VECTOR
3608 020364 000233 SPL 3 ; SET PROCESSOR PRIORITY BELOW BR4
3609 020366 152777 000100 160546 BISB #BIT6, @STPS ; SET INTERRUPT BIT
3610 020374 012777 000015 160542 MOV #15, @STPB ; SEND CHARACTER TO PRINTER
3611 020402 130057 000100 160530 15$: BITB R0, @-(PC) ; EXECUTE INSTRUCTION UNDER TEST
3612 020404 142777 000100 160530 BICB #BIT6, @STPS ; WILL EXECUTE IF STATE FAILS
3613 020412 104377 ERROR 377 ; STATE D45.01 FAILED
3614 020414 000452 452
3615 020416 142777 000100 160516 16$: BICB #BIT6, @STPS ; TEST OK, CLEAR INTERRUPT BIT
3616 ; CONTINUE
3617
3618 ; *****
3619 ; *TEST 40 RED ZONE TRAP
3620 ; *
3621 ; * A RED ZONE TRAP IS FIRST ATTEMPTED WITH THE SP AT 336.
3622 ; * IF BEN13 FAILS EXECUTION WILL GO TO EITHER BRK.80 OR BRK.20
3623 ; * OR PUP.00.
3624 ; * BRK.80 WILL CAUSE THE OLD PSW AND PC TO BE STACKED ON THE
3625 ; * OLD STACK INSTEAD OF LOCATIONS 2 AND 0.

```

N08

```

3626      ;*      BRK.20 WILL MAKE IT LOOK LIKE THE RED ZONE FAILED.
3627      ;*      PUP.00 WILL CAUSE A TRAP TO LOCATION 24.
3628      ;*
3629      ;*      IF THE PROCESSOR FAILS TO TRAP EITHER TMCD SL RED IS
3630      ;*      NOT GOING LOW OR TMCC ABORT IS NOT GOING LOW.
3631      ;*
3632      ;*      IF UCB ABORT RESTART FAILS TO GO LOW OR E10(13)
3633      ;*      IS BAD THE PROCESSOR WTL HANG IN THE PAUSE STATE.
3634      ;*      *****
3635      ;*      TST40: SCOPE
3636      020424 000004      MOV      #TST41,NEXTTST ;SAVE ADDRESS OF NEXT TEST
3637      020426 012767 021160 160634      MOV      #14$,SLPADR ;SETUP LOOP ADR
3638      020434 012767 020472 160444      MOV      @#PSW,$TMP3 ;SAVE PSW
3639      020442 013767 177776 160520      MOV      #BIT4,@#PSW ;IS T BIT ON?
3640      020450 032737 000020 177776      BIT      14$ ;BRANCH IF NO
3641      020456 001405      BEQ      14$ ;BRANCH IF NO
3642      020460 012746 000340      MOV      #PR7,-(SP) ;PUT NEW PSW ON STACK
3643      020464 012746 020472      MOV      #14$,-(SP) ;PUT RETURN ADR ON STACK
3644      020470 000006      RTT ;TURN T BIT OFF
3645      020472 012737 000340 000006 14$:      MOV      #PR7,@#ERRVEC+2 ;RESTORE ERRVEC PSW
3646      020500 012767 020514 160402      MOV      #64$,SLPERR ;SETUP ERROR LOOP
3647      020506 012737 020552 000024      MOV      #3$,@#PWRVEC ;SETUP LOCATION 24
3648      020514 012737 020570 000004 64$:      MOV      #1$,@#ERRVEC ;SETUP ERRVEC
3649      020522 012706 000336      MOV      #336,SP ;SET THE SP TO RED ZONE
3650      020526 012700 177777      MOV      #-1,RO ;SETUP RO
3651      020532 010016      MOV      RO,(SP) ;EXECUTE THE TRAP INSTRUCTION
3652      020534 012706 001100 6$:      MOV      #STACK,SP ;RESET THE SP
3653      020540 012737 036242 000004      MOV      #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
3654      020546 104235      ERROR 235 ;RED ZONE REFERENCE FAILED TO TRAP
3655      020550 000431      BR      8$
3656      020552 012706 001100 3$:      MOV      #STACK,SP ;RESET THE SP
3657      020556 012737 036242 000004      MOV      #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
3658      020564 104237      ERROR 237 ;BEN 13 FAILED TO PUP.00
3659      020566 000422      BR      8$
3660      020570 023727 000000 020534 1$:      CMP      @#0,#6$ ;DID BEN 13 FAIL?
3661      020576 001407      BEQ      7$ ;BRANCH IF NO
3662      020600 012706 001100      MOV      #STACK,SP ;RESET THE SP
3663      020604 012737 036242 000004      MOV      #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
3664      020612 104240      ERROR 240 ;BEN 13 FAILED TO BRK.80
3665      020614 000407      BR      8$
3666      020616 022737 177777 000336 7$:      CMP      #-1,@#336 ;:DID YEL ZONE OCCUR?
3667      020624 001003      BNE      8$ ;BRANCH IF NO
3668      020626 005037 000336      CLR      @#336 ;SETUP FOR LOOPING
3669      020632 104251      ERROR 251 ;YEL ZONE IN RED REGION
3670      ;*      ;TEST TO ENSURE PSW REFERENCE VIA THE SP CAUSES A RED ZONE TRAP
3671      020634 012767 020642 160246 8$:      MOV      #63$,SLPERR ;SETUP ERROR LOOP
3672      020642 012737 020672 000004 63$:      MOV      #4$,@#ERRVEC ;SETUP ERRVEC
3673      020650 012706 177776      MOV      #PSW,SP ;PUT ADDRESS OF PSW IN SP
3674      020654 005016      CLR      (SP) ;EXECUTE THE TRAP CAUSING INSTRUCTION
3675      020656 012706 001076      MOV      #1076,SP ;RESET THE SP
3676      020662 012737 036242 000004      MOV      #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
3677      020670 104241      ERROR 241 ;NO RED ZONE ON STACK OVERFLOW
3678      ;*
3679      ;*      ;TEST TO ENSURE SL REG GREATER THAN BADRR CAUSES A RED ZONE
3680      020672 012767 020700 160210 4$:      MOV      #62$,SLPERR ;SETUP ERROR LOOP
3681      020700 012737 020742 000004 62$:      MOV      #5$,@#ERRVEC ;SETUP RESVEC
  
```

```

3682 020706 012737 000400 177774      MOV      #400,2#STKLMT      ;SET STACK LIMIT REGISTER TO 400
3683 020714 012706 000336              MOV      #336,SP           ;SET THE SP
3684 020720 005016              CLR      (SP)              ;EXECUTE THE TRAP CAUSING INSTRUCTION
3685 020722 012706 001100      MOV      #STACK,SP        ;RESET THE SP
3686 020726 005037 177774      CLR      2#STKLMT         ;ENSURE SL REG. CLEAR
3687 020732 012737 036242 000004      MOV      #CPUSPUR,2#ERRVEC ;RESTORE ERRVEC
3688 020740 104242              ERROR    242              ;NO RED ZONE WHEN SL REG>BUS ADDR

```

```

3689
3690      ;TEST OF TMCD YEL ZONE (E15)
3691 020742 012767 020754 160140 5$: MOV      #61$, $LPERR      ;SETUP ERROR LOOP
3692 020750 005037 177774              CLR      2#STKLMT         ;ENSURE SL CLEAR
3693 020754 012737 020774 000004 61$: MOV      #9$, 2#ERRVEC    ;SETUP ERRVEC
3694 020762 012706 000240              MOV      #240,SP          ;SETUP THE SP
3695 020766 012700 177777              MOV      #-1,RO           ;SETUP RO
3696 020772 010016              MOV      RO,(SP)          ;EXECUTE THE TRAP CAUSING INSTRUCTION
3697 020774 022737 177777 000240 9$: CMP      #-1,2#240        ;DID YEL ZONE OCCUR?
3698 021002 001010              BNE     10$              ;BRANCH IF NO
3699 021004 012706 001100      MOV      #STACK,SP        ;RESTORE THE SP
3700 021010 012737 036242 000004      MOV      #CPUSPUR,2#ERRVEC ;RESTORE ERRVEC
3701 021016 005037 000240              CLR      2#240            ;FOR LOOPING
3702 021022 104252              ERROR    252            ;TMCD YEL ZONE DID NOT GO LOW
3703 021024 012767 021032 160056 10$: MOV      #60$, $LPERR      ;SETUP ERROR LOOP
3704 021032 012737 021046 000004 60$: MOV      #11$, 2#ERRVEC   ;SETUP ERRVEC
3705 021040 012706 000140              MOV      #140,SP          ;SETUP THE SP
3706 021044 010016              MOV      RO,(SP)          ;EXECUTE THE TRAP CAUSING INSTR.
3707 021046 022737 177777 000140 11$: CMP      #-1,2#140        ;DID YEL ZONE OCCUR?
3708 021054 001010              BNE     12$              ;BRANCH IF NO
3709 021056 012706 001100      MOV      #STACK,SP        ;RESTORE SP
3710 021062 012737 036242 000004      MOV      #CPUSPUR,2#ERRVEC ;RESTORE ERRVEC
3711 021070 005037 000140              CLR      2#140            ;FOR LOOPING
3712 021074 104253              ERROR    253            ;TMCD YEL ZONE DID NOT GO LOW
3713 021076 012706 001100      MOV      #STACK,SP        ;RESTORE SP
3714 021102 012737 036242 000004 12$: MOV      #CPUSPUR,2#ERRVEC ;RESTORE ERRVEC
3715 021110 022737 000004 177766      CMP      #BIT2,2#CPUERR   ;DID RED ZONE BIT IN CPU ERROR SET?
3716 021116 001407              BEQ     13$              ;BRANCH IF YES
3717 021120 013767 177766 160026      MOV      2#CPUERR,$REGO   ;SAVE FOR TYPEOUT
3718 021126 012767 000004 160026      MOV      #BIT2,$TMP0      ;SAVE EXPECTED VALUE
3719 021134 104261              ERROR    261            ;RED ZONE BIT IN CPU ERROR DID NOT SET
3720 021136 012767 021144 157744 13$: MOV      #57$, $LPERR      ;SETUP ERROR LOOP
3721 021144 005037 177766 57$: CLR      2#CPUERR        ;CLEAR RED ZONE BIT
3722 021150 005737 177766      TST     2#CPUERR          ;DID REG CLEAR?
3723 021154 001401              BEQ     TST41            ;BRANCH IF YES
3724 021156 104262              ERROR    262            ;RED ZONE BIT DID NOT CLEAR

```

```

3725
3726      ;*****
3727      ;TEST 41      BIT TEST OF STACK LIMIT REGISTER
3728      ;
3729      ;      FIRST A 125252 AND 52525 PATTERN IS PUT IN THE REGISTER TO ENSURE
3730      ;      THAT THE REGISTER DOESN'T HAVE ANY STUCK BITS AND THAT
3731      ;      THE DMUX SELECT AND INPUT LINES WORK.
3732      ;
3733      ;      IF SCCE SL ADRS DOES NOT GET TO TMCD OR IF TMCD E28 OR E14
3734      ;      IS BAD THE BR WILL BE SELECTED.  THE PB REGISTER IS LOADED
3735      ;      WITH 200 SO IF TMCD LO BYTE EN DOES NOT GO LOW AN
3736      ;      ERROR WILL BE DETECTED.
3737      ;*****

```

```

3738 021160 000004 TST41: SCOPE
3739 021162 012767 021424 160004 MOV #TST42,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
3740 021170 012767 021424 160072 MOV #TST42,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3741 021176 012737 041646 000024 MOV #SPWRDN,#PWAVEC ;RESTORE POWER VECTOR
3742 021204 005005 CLR R5 ;CLEAR R5
3743 021206 012737 125252 177774 MOV #125252,#STKLMT ;PUT PATTERN IN STACK LIMIT REG
3744 021214 012737 000200 177770 MOV #200,#177770 ;PUT 200 IN PB REGISTER
3745 021222 012700 125000 MOV #125000,R0 ;SETUP R0 TO LOOK LIKE STK LIMIT
3746 021226 000237 SPL 7 ;PUT PRIORITY BITS IN KNOWN CONFIGURATION
3747 021230 020037 177774 CMP R0,#STKLMT ;EXECUTE TEST ON STKLMT REG.
3748 021234 001404 BEQ 1$ ;BRANCH IF TEST OK
3749 021236 005205 INC R5 ;INCREMENT TEST FAIL INDICATOR
3750 021240 013767 177774 157752 MOV #STKLMT,E1STKLM ;SAVE ERROR VALUE
3751 021246 012737 052400 177774 1$: MOV #52400,#STKLMT ;PUT COMPLEMENT PATTERN REG.
3752 021254 012700 052400 MOV #52400,R0 ;PUT IN R0
3753 021260 020037 177774 CMP R0,#STKLMT ;EXECUTE TEST ON REG.
3754 021264 001415 BEQ 2$ ;BRANCH IF TEST OK
3755 021266 005705 TST R5 ;DID FIRST TEST FAIL?
3756 021270 001023 BNE 3$ ;BRANCH IF YES
3757 021272 013767 177774 157722 MOV #STKLMT,E2STKLM ;SAVE ERROR VALUE
3758 021300 012767 052400 157654 MOV #52400,$TMP0 ;SAVE EXPECTED VALUE
3759 021306 005037 177774 CLR #STKLMT ;CLEAR THE REG
3760 021312 012706 001100 MOV #STACK,SP ;RESTORE THE SP
3761 021316 104243 ERROR 243 ;52400 PATTERN FAILED
3762 021320 005705 2$: TST R5 ;DID FIRST TEST FAIL?
3763 021322 001436 BEQ 4$ ;BRANCH IF NO
3764 021324 012767 125000 157630 MOV #125000,$TMP0 ;SAVE EXPECTED VALUE
3765 021332 005037 177774 CLR #STKLMT ;CLEAR REG.
3766 021336 104244 ERROR 244 ;125252 PATTERN FAILED
3767 021340 005037 177774 3$: CLR #STKLMT ;CLEAR STACK LIMIT REG
3768 021344 026727 157650 013767 CMP E1STKLM,#13767 ;DID BR GET SELECTED ON STACK LIMIT REF.?
3769 021352 001001 BNE 5$ ;BRANCH IF NO
3770 021354 104245 ERROR 245 ;BR SELECTED BY DMUX
3771 021356 026727 157636 125200 5$: CMP E1STKLM,#125200 ;DID PB GET GATED ALSO?
3772 021364 001001 BNE 6$ ;BRANCH IF NO
3773 021366 104246 ERROR 246 ;TMCD LO BYTE EN DOES NOT GO LOW
3774 021370 022767 000340 157622 6$: CMP #340,E1STKLM ;DID PS GET SELECTED?
3775 021376 001001 BNE 7$ ;BRANCH IF NO
3776 021400 104247 ERROR 247 ;D MUX SELECTED PSW
3777 021402 012767 125000 157552 7$: MOV #125000,$TMP0
3778 021410 012767 052400 157546 MOV #52400,$TMP1
3779 021416 104250 ERROR 250 ;BOTH PATTERNS FAILED BUT DON'T KNOW WHY
3780 021420 005037 177774 4$: CLR #STKLMT ;CLEAR THE STACK LIMIT REG.

```

```

3781
3782 ;*****
3783 ;TEST 42 SL REGISTER COMPARATOR TEST 1
3784 ;
3785 ;
3786 ; THIS TEST RUNS A HIGH BYTE COUNT PATTERN THRU THE BUS
3787 ; ADDRESS MUX FOR EACH PATTERN OF THE STACK LIMIT REGISTER.
3788 ; FOR EACH PATTERN OF ADDRESSES THERE WILL BE ONE YEL TRAP
3789 ; AT THE ADDRESS CORRESPONDING TO THE SL REG+340 AND A RED
3790 ; TRAP AT EVERY ADDRESS BELOW THIS.
3791 ; THIS TEST ONLY TESTS ADDRESSES UP TO THE I/O PAGE.
3792 ; THE I/O PAGE ADDRESSES WILL BE TESTED SEPARATELY WITH
3793 ; MEMORY MANAGEMENT ENABLED AND THE I/O PAGE MAPPED INTO RESIDENT
; MEMORY

```

```

3794
3795
3796
3797
3798
3799
3800
3801
3802
3803
3804
3805
3806
3807 021424 000004
3808
3809
3810
3811
3812
3813
3814
3815
3816 021426 012767 022034 157540
3817 021434 012767 022034 157626
3818 021442 012767 003706 157434
3819 021450 012767 021456 157430
3820 021456 032737 000010 177570 13$:
3821 021464 001002
3822 021466 000177 157502
3823 021472 012737 021602 000004 11$:
3824 021500 005067 157462
3825 021504 012703 000340
3826 021510 012700 120000
3827 021514 005060 000002
3828 021520 012701 000340
3829 021524 012704 000344
3830 021530 012702 000340 2$:
3831 021534 012705 000344
3832 021540 016567 177776 157414 3$:
3833 021546 016567 177774 157410
3834 021554 010506 4$:
3835 021556 011616
3836
3837 021560 020604
3838 021562 101066
3839 021564 001403
3840
3841 021566 052710 000010
3842 021572 000442
3843
3844 021574 052710 000012
3845 021600 000437
3846
3847
3848 021602 016765 157354 177776 1$:
3849 021610 016765 157350 177774

```

; * 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.
; *
; * *****
; * ST42: 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.
; * *****
; * MOV #TST43, \$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
; * MOV #TST43, NEXTTST ;SAVE START ADDRESS OF NEXT TEST
; * MOV #101990, \$ICNT ;SETUP ITERATION COUNT
; * MOV #13\$, \$LPAOR ;SETUP LOOP ADDRESS
; * BIT #BIT3, 2\$SWR ;IS SWITCH 3 ON?
; * BNE 11\$;BRANCH IF YES
; * JMP 2\$ESCAPE ;GO TO NEXT TEST
; * MOV #1\$, 2\$ERRVEC ;SETUP ERRVEC
; * CLR \$TMP2 ;INITIALIZE ERROR OVERFLOW FLAG
; * MOV #340, R3 ;SET SOB COUNT FOR SL REGISTER
; * MOV #120000, R0 ;INITIALIZE ERROR DATA POINTER
; * CLR 2(R0) ;INITIALIZE ERROR DATA BUFFER
; * MOV #340, R1 ;INITIALIZE YELLOW ZONE ADDRESS (TRAP CASE)
; * MOV #344, R4 ;SETUP YELLOW ZONE ADDR (NO TRAP)
; * MOV #340, R2 ;SET SOB COUNT FOR SP
; * MOV #344, R5 ;INITIALIZE SECONDARY STORAGE FOR SP
; * MOV -2(R5), \$TMP0 ;SAVE WORDS AT
; * MOV -4(R5), \$TMP1 ;STACK UNDER TEST
; * MOV R5, R6 ;SET THE SP
; * MOV (R6), (R6) ;EXECUTE TEST INSTRUCTION
; * NO TRAP. DETERMINE IF THIS IS CORRECT.
; * CMP R6, R4 ;IS ADDRESS > YELL ZONE BOUNDARY?
; * BHI 5\$;BRANCH IF YES
; * BEQ 6\$;BRANCH IF ADDRESS = YELL ZONE BOUNDARY
; * NO TRAP ADDRESS IS LESS THAN YELLOW ZONE BOUNDARY
; * BIS #BIT3, (R0) ;SET ERROR TYPE IN DATA BUFFER
; * BR 10\$;GO RECORD DATA
; * NO TRAP EQUALS YELLOW ZONE BOUNDARY
; * BIS #12, (R0) ;SET ERROR TYPE IN DATA BUFFER
; * BR 10\$;GO RECORD DATA
; * GOT A TRAP. RESTORE AND DETERMINE IF IT IS CORRECT.
; * MOV \$TMP0, -2(R5) ;RESTORE WORDS AT
; * MOV \$TMP1, -4(R5) ;OLD STACK UNDER TEST

```

3850 021616 032737 000004 177766 BIT #BIT2,@#CPUERR ;WAS IT A RED ZONE?
3851 021624 001406 BEQ 8$ ;BRANCH IF NO
3852 021626 020106 CMP R1,R6 ;IS ADDRESS < YELL ZONE BOUNDARY?
3853 021630 101043 BHI 5$ ;BRANCH IF YES
3854 021632 001422 BEQ 10$ ;BRANCH IF ADDRESS = YELL ZONE BOUNDRY
3855 ;RED ZONE TRAP ON LEGAL ADDRESS
3856 021634 052710 000002 BIS #BIT1,(R0) ;SET ERROR TYPE IN DATA BUFFER
3857 021640 000417 BR 10$ ;GO RECORD DATA
3858 ;NOT A RED ZONE. IS IT A YELLOW ZONE?
3859 021642 032737 000010 177766 8$: BIT #BIT3,@#CPUERR ;IS THIS A YELLOW ZONE TRAP?
3860 021650 001002 BNE 12$ ;BRANCH IF NO
3861 021652 000167 014364 JMP CPUSPUR ;GO TO SPURIOUS ROUTINE
3862 021656 020106 12$: CMP R1,R6 ;IS ADDRESS = YELL ZONE BOUNDARY?
3863 021660 001427 BEQ 5$ ;BRANCH IF YES
3864 021662 101003 BHI 9$ ;BRANCH IF ADDRESS IS < YELL ZONE BOUNDRY
3865 ;YELLOW ZONE TRAP ON LEGAL ADDRESS
3866 021664 052710 000006 BIS #6,(R0) ;SET ERROR TYPE IN DATA BUFFER
3867 021670 000403 BR 10$ ;GO RECORD DATA
3868 ;YELLOW ZONE TRAP ON RED ZONE ADDRESS
3869 021672 052710 000004 9$: BIS #BIT2,(R0) ;SET ERROR TYPE IN DATA BUFFER
3870 021676 000400 BR 10$ ;GO RECORD DATA
3871 ;
3872 ;RECORD ERROR DATA
3873 021700 032737 001000 177570 10$: BIT #BIT9,@#SWR ;IS LOOP ON ERROR ENABLED?
3874 021706 001322 BNE 4$ ;BRANCH IF YES
3875 021710 005767 157252 TST $TMP2 ;HAS ERROR BUFFER OVERFLOWED?
3876 021714 001011 BNE 5$ ;BRANCH IF YES
3877 021716 005200 INC R0 ;SET POINTER TO HIGH BYTE
3878 021720 113720 177775 MOVB @#STKLMT+1,(R0)+ ;SAVE ERROR STACK LIMIT
3879 021724 010620 MOV R6,(R0)+ ;SAVE ERROR SP
3880 021726 020027 157774 CMP R0,#157774 ;IS BUFFER AT PAGE 7?
3881 021732 001002 BNE 5$ ;BRANCH IF NO
3882 021734 005267 157226 INC $TMP2 ;SET BUFFER OVERFLOW FLAG
3883 ;
3884 021740 062705 000400 5$: ADD #400,R5 ;GO TO NEXT STACK ADDRESS
3885 021744 005037 177766 CLR @#CPUERR ;CLEAR ERROR REG
3886 021750 000240 NOP
3887 021752 005302 DEC R2 ;REPLACES A
3888 021754 001271 BNE 3$ ;SOB INSTRUCTION
3889 021756 062737 000400 177774 ADD #400,@#STKLMT ;GO TO NEXT SL ADDRESS
3890 021764 062701 000400 ADD #400,R1 ;SET NEXT YELLOW ZONE ADDRESS
3891 021770 000240 NOP
3892 021772 062704 000400 ADD #400,R4 ;SET NEXT YELLOW ZONE ADDR(NO TRAP)
3893 021776 005303 DEC R3 ;THIS REPLACES
3894 022000 001253 BNE 2$ ;A SOB
3895 ;
3896 ;DONE WITH TEST. WAS THERE AN ERROR?
3897 022002 005037 177774 CLR @#STKLMT ;RESET THE SL REG
3898 022006 012706 001100 MOV #STACK,SP ;AND SP
3899 022012 012737 036242 000004 MOV #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
3900 022020 005737 120002 TST @#120002 ;WAS THERE AN ERROR?
3901 022024 001403 BEQ TST43 ;BRANCH IF NO
3902 022026 010067 157122 MOV R0,$REGO ;SAVE ERROR DATA POINTER
3903 022032 104263 ERROR 263 ;STACK LIMIT COMPARATORS FAILED
3904 ;*****
3905 ;*TEST 43 ODD ADDRESS ERROR

```

```

3906
3907
3908
3909
3910
3911
3912
3913
3914
3915
3916 022034 000004
3917 022036 012767 022470 157224
3918 022044 005005
3919
3920 022046 012706 001100
3921 022052 012737 022132 000004
3922 022060 012700 001163
3923 022064 012767 177777 157070
3924 022072 005210
3925
3926 022074 012737 022120 000004
3927 022102 110030
3928 022104 012737 036242 000004
3929 022112 005205
3930 022114 104265
3931 022116 000423
3932 022120 012737 036242 000004
3933 022126 104266
3934 022130 000416
3935
3936
3937 022132 005037 177766
3938 022136 012706 001100
3939 022142 012767 022132 156740
3940 022150 012737 022166 000004
3941 022156 012700 001163
3942 022162 110030
3943 022164 000404
3944 022166 032737 000100 177766
3945 022174 001037
3946
3947 022176 005037 177766
3948 022202 012706 001077
3949 022206 012737 022242 000004
3950 022214 012737 022224 000030
3951 022222 104000
3952 022224 012706 001100
3953 022230 012737 036242 000004
3954 022236 104267
3955 022240 000446
3956 022242 012706 001100
3957 022246 012737 036242 000004
3958 022254 032737 000100 177766
3959 022262 001401
3960 022264 104270
3961 022266 104377

```

```

; *
; * 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.
; * *****
; * ST43: SCOPE
; * MOV #ST44, NEXTTST ; SAVE ADDRESS OF NEXT TEST
; * CLR R5 ; INITIALIZE ERROR COUNT
; * ; NON BYTE REFERENCE ON DATI
; * MOV #STACK, SP ; INITIALIZE THE SP
; * MOV #1$, @ERRVEC ; SETUP ERRVEC
; * MOV #STMP0+1, RO ; PUT ODD ADDRESS IN RO
; * MOV #-1, STMP0 ; PUT -1 IN STMP0 TO SEE IF TEST CHANGES IT
; * INC (RO) ; EXECUTE ODD ADDRESS INSTRUCTION
; * ; TRAP DID NOT OCCUR. TRY A DATI.
; * MOV #2$, @ERRVEC ; SETUP ERRVEC
; * MOV RO, @ (RO)+ ; EXECUTE DATI TO CAUSE ODD ADDR
; * MOV #CPUSPUR, @ERRVEC ; RESTORE ERRVEC
; * INC R5 ; SET ERROR COUNT
; * ERROR 265 ; NEITHER - BYIN OR DATI CAUSE ODD ADDR
; * BR 3$
; * 2$: MOV #CPUSPUR, @ERRVEC ; RESTORE ERRVEC
; * ERROR 266 ; DATI TRAPPED BUT -BYIN DIDN'T
; * BR 3$
; * ; NON BYTE REFERENCE WORKED. NOW TRY DATI.
; * 1$: CLR @CPUERR ; CLEAR ERROR REG
; * MOV #STACK, SP ; INITIALIZE THE SP
; * MOV #1$, @LPERR ; CHANGE LPERR ADDRESS TO THIS SECTION
; * MOV #3$, @ERRVEC ; SETUP ERRVEC
; * MOV #STMP0+1, RO ; PUT ODD ADDR IN RO
; * MOV RO, @ (RO)+ ; EXECUTE ODD ADDRESS INSTRUCTION
; * BR 13$
; * 3$: BIT #BIT6, @CPUERR ; DID ODD ADDR BIT SET?
; * BNE 12$ ; BRANCH IF YES
; * ; TRAP FAILED OR ERROR REG FAILED. TRY A DATO. (REVERSES INPUTS TO TMCC E5(12,13))
; * 13$: CLR @CPUERR ; CLEAR ODD ADDR BIT
; * MOV #STACK-1, SP ; MAKE SP AN ODD ADDRESS
; * MOV #4$, @ERRVEC ; SETUP ERRVEC
; * MOV #5$, @EMTVEC ; SETUP EMT VECTOR
; * EMT 0 ; EXECUTE DATO TO SP
; * 5$: MOV #STACK, SP ; RESTORE SP
; * MOV #CPUSPUR, @ERRVEC ; RESTORE ERRVEC
; * ERROR 267 ; BOTH DATI AND DATO FAILED
; * BR 6$
; * 4$: MOV #STACK, SP ; RESTORE THE SP
; * MOV #CPUSPUR, @ERRVEC ; RESTORE ERRVEC
; * BIT #BIT6, @CPUERR ; DID ODD ADDR BIT SET?
; * BEQ 14$ ; BRANCH IF NO
; * ERROR 270 ; DATO WORKS BUT DATI FAILED
; * 14$: ERROR 377

```



```

3962 022270 000456          456
3963 022272 000431          BR      65
3964
3965 ;EITHER DATI WORKED OR -BYIN AND DATI FAILED. NOW TRY DATO.
3966 022274 012767 022166 156606 125:  MOV    #3$, $LPERR      ;CHANGE LPERR ADDRESS TO THIS SECTION
3967 022302 012706 001077          MOV    #STACK-1, SP      ;MAKE SP AN ODD ADDRESS
3968 022306 012737 022356 000004          MOV    #6$, @#ERRVEC    ;SETUP ERRVEC
3969 022314 012737 022324 000030          MOV    #7$, @#EMTVEC    ;SETUP EMTVEC TO CATCH A FAILURE
3970 022322 104000          EMT    0                ;EXECUTE DATO TO ODD ADDRESS
3971 ;TRAP FAILED
3972 022324 012706 001076          75:  MOV    #1076, SP        ;RESET SP
3973 022330 012737 036242 000004          MOV    #CPUSPUR, @#ERRVEC ;RESTORE ERRVEC
3974 022336 012737 037124 000030          MOV    #ERROR, @#EMTVEC  ;RESTORE EMTVEC
3975 022344 005705          TST    R5
3976 022346 001002          BNE    115
3977 022350 104271          ERROR  271             ;NO TRAP ON DATO BUT DATI & -BYIN OK
3978 022352 000401          BR     65
3979 022354 104274          115: ERROR  274             ;NO TRAPS
3980
3981 ;EITHER DATO WORKED OR -BYIN AND DATI AND DATO FAILED OR DATO
3982 ;FAILED BUT -BYIN AND DATI OK. TRY SM357 AND SRC1 DATI.
3983 022356 012737 037124 000030          65:  MOV    #ERROR, @#EMTVEC  ;RESTORE EMT VEC
3984 022364 012767 022372 156516          MOV    #8$, $LPERR      ;CHANGE LOOP ERROR ADDR TO THIS SECT.
3985 022372 012706 001076          85:  MOV    #1076, SP        ;RESET SP
3986 022376 012737 022444 000004          MOV    #9$, @#ERRVEC    ;SETUP ERRVEC
3987 022404 012700 001163          MOV    #STMP0+1, R0      ;PUT ODD ADDRESS IN R0
3988 022410 012767 001164 156544          MOV    #STMP1, $TMP0     ;PUT EVEN ADDRESS IN $TMP0
3989 022416 113001          MOVB   @(R0)+, R1        ;EXECUTE INSTRUCTION TO CAUSE TRAP
3990 022420 012737 036242 000004          MOV    #CPUSPUR, @#ERRVEC ;RESTORE ERRVEC
3991 022426 012767 022470 156540          MOV    #TST44, $ESCAPE   ;SAVE START ADDRESS OF NEXT TEST
3992 022434 012767 022470 156626          MOV    #TST44, NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
3993 022442 104272          ERROR  272             ;SM357*SRC1 DATI FAILED TO TRAP
3994
3995 ;SM357*SRC1 DATI WORKS CHECK CPU ERROR REG.
3996 022444 012737 036242 000004          95:  MOV    #CPUSPUR, @#ERRVEC ;RESTORE ERRVEC
3997 022452 032737 000100 177766          BIT    #BIT6, @#CPUERR  ;IS ODD ADDRESS BIT SET?
3998 022460 001001          BNE    105             ;BRANCH IF YES
3999 022462 104273          ERROR  273             ;CPU ERROR REG BIT DOES NOT SET
4000 022464 005037 177766          105: CLR    @#CPUERR     ;CLEAR ERROR REG.
4001 ;CONTINUE
4002 ;*****
4003 ;*TEST 44      T BIT TRAP
4004 ;*
4005 ;* IF BEN 13 FAILS EXECUTION WOULD GO TO BRK.20.
4006 ;* THIS WOULD LOOK LIKE THE TRAP DIDN'T OCCUR.
4007 ;*
4008 ;* IF THE TRAP DOESN'T OCCUR THEN EITHER PDRD P504(1) DOES NOT GET
4009 ;* TO TMCB AS A HIGH OR IT DOES NOT GET TO TMCB E51(10)
4010 ;* AS A LOW OR E51 IS BAD OR IRCD RTT DOES NOT GET TO TMCB AS A HIGH.
4011 ;*****
4012 022470 000004          TST44: SCOPE
4013 022472 012767 022624 156474          MOV    #TST45, $ESCAPE  ;SAVE START ADDRESS OF NEXT TEST
4014 022500 012767 022624 156562          MOV    #TST45, NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
4015 ;*****
4016 ;THIS CODE TURNS THE T BIT OFF IF IT IS ON.
4017 022506 012746 000340          MOV    #PR7, -(SP)     ;PUT PRIORITY LEVEL 7 ON SP

```



```

4018 022512 012746 022526      MOV      #1$, -(SP)      ; PUT RETURN ADDRESS ON SP
4019 022516 013767 177776 156444  MOV      @#PSW, $TMP3    ; SAVE PSW
4020 022524 000006      RTT                    ; TURN OFF T BIT
4021                                     ; *****
4022 022526 012767 022534 156354 1$:  MOV      #2$, $LPERR    ; SETUP LOOP ADDRESS
4023 022534 012706 001100 2$:  MOV      #1100, SP      ; INITIALIZE THE SP
4024 022540 012737 022624 000014  MOV      #3$, @#TBITVEC  ; SETUP T BIT VECTOR
4025 022546 012737 022622 000010  MOV      #8$, @#RESVEC   ; SETUP RESVEC
4026 022554 012746 000360      MOV      #360, -(SP)    ; PUT NEW PSW ON STACK (ENABLE T BIT)
4027 022560 012746 022566      MOV      #5$, -(SP)    ; PUT RETURN LOCATION ON STACK
4028 022564 000002      RTI                    ; TURN ON T BIT
4029 022566 012746 000340 5$:  MOV      #PR7, -(SP)    ; SETUP STACK TO
4030 022572 012746 022606      MOV      #6$, -(SP)    ; TURN OFF T BIT
4031 022576 013767 177776 156402  MOV      @#PSW, $ERPSW  ; SAVE PSW
4032 022604 000006      RTT                    ; TURN T BIT OFF
4033 022606 032767 000020 156372 6$:  BIT      #BIT4, $ERPSW  ; DID T BIT SET?
4034 022614 001401      BEQ      7$            ; BRANCH IF NO
4035 022616 104275      ERROR    27$          ; T BIT TRAP FAILED
4036 022620 104276      ERROR    27$          ; T BIT NEVER SET
4037 022622 104300      ERROR    30$          ; TRAP VECTOR DECODE FAILED
4038 022624      3$:  CONTINUE
4039                                     ; *****
4040 ;*TEST 45      T BIT TRAP AND RTT
4041 ;*
4042 ;* IF THE INSTRUCTION AFTER THE RTT DOES NOT GET EXECUTED THEN
4043 ;* EITHER IRCD RTT DOES NOT GO LOW OR IT DOES NOT GET TO
4044 ;* TMCB E74(11) OR TMCB E74 IS BAD.
4045 ;* *****
4046 022624 000004  †ST45: SCOPE
4047 022626 012767 022676 156434  MOV      #TST46, NEXTTST ; SAVE ADDRESS OF NEXT TEST
4048 022634 005000      CLR      R0            ; ENSURE R0 CLEAR
4049 022636 012706 001100      MOV      #STACK, SP    ; INITIALIZE THE SP
4050 022642 012746 000360      MOV      #360, -(SP)   ; PUT NEW PSW ON STACK (ENABLE T BIT)
4051 022646 012746 022662      MOV      #1$, -(SP)    ; PUT PC ON STACK
4052 022652 012737 022666 000014  MOV      #2$, @#TBITVEC ; SETUP T BIT VECTOR
4053 022660 000006      RTT                    ; TURN T BIT ON WITH RTT
4054 022662 012700 000001 1$:  MOV      #1, R0        ; THIS SHOULD EXECUTE
4055 022666 022700 000001 2$:  CMP      #1, R0        ; DID MOV INSTRUCTION EXECUTE?
4056 022672 001401      BEQ      TST46        ; BRANCH IF YES
4057 022674 104301      ERROR    30$          ; RTT DID NOT DISABLE T BIT
4058 ;* *****
4059 ;*TEST 46      ILLEGAL INSTRUCTIONS
4060 ;* THIS TEST ENSURES THAT ILLEGAL OP CODES TRAP TO LOCATION 10.
4061 ;* ONLY THOSE OP CODES THAT HAVE A SINGLE BIT THAT
4062 ;* DISTINGUISHES THEM FROM A LEGAL INSTRUCTION WILL BE TESTED.
4063 ;* *****
4064 022676 000004  †ST46: SCOPE
4065 022700 012767 023360 156362  MOV      #TST47, NEXTTST ; SAVE ADDRESS OF NEXT TEST
4066 022706 012737 036226 000014  MOV      #$RTN, @#TBITVEC ; RESTORE T BIT VEC
4067 022714 012767 022760 156164  MOV      #31$, $LPADR   ; SETUP LOOP ADR
4068 022722 032767 000020 156240  BIT      #BIT4, $TMP3   ; WAS T BIT ON?
4069 022730 001410      BEQ      30$          ; BRANCH IF NO
4070 022732 012737 000360 000012  MOV      #360, @#RESVEC+2 ; SETUP RESVEC PSW
4071 022740 012746 000360      MOV      #360, -(SP)   ; SETUP STACK PSW
4072 022744 012746 022760      MOV      #31$, -(SP)   ; PUT RETURN ADR ON SATCK
4073 022750 000006      RTT                    ; TURN T BIT ON

```

```

4074      ;SECTION 1-07 THRU 77
4075 022752 012737 000340 000012 30$: MOV #PR7,2#RESVEC+2 ;RESTORE RESVEC PSW
4076 022760 012767 022774 156122 31$: MOV #1$, $LPERR ;SETUP ERROR LOOP
4077 022766 012737 023012 000010      MOV #2$,2#RESVEC ;SETUP RESVEC
4078 022774 012706 001100      1$: MOV #STACK, SP ;INITIALIZE THE SP
4079 023000 012746 023006      MOV #3$, -(SP) ;PUT ADDRESS OF 3$ ON STACK
4080 023004 000007      7 ;EXECUTE OP CODE 7
4081      ;FAILURE-RTT OCCURED
4082 023006 104377      3$: ERROR 377 ;OP CODE 7 FAILED TO TRAP
4083 023010 000453      453
4084 023012 012767 023026 156070 2$: MOV #4$, $LPERR ;SETUP ERROR LOOP
4085 023020 012737 023034 000010      MOV #5$,2#RESVEC ;SETUP RESVEC
4086 023026 000015      4$: 15 ;EXECUTE OP CODE 15
4087      ;FAILURE
4088 023030 104377      ERROR 377 ;OP CODE 15 FAILED TO TRAP
4089 023032 000453      453
4090 023034 012767 023050 156046 5$: MOV #6$, $LPERR ;SETUP ERROR LOOP
4091 023042 012737 023056 000010      MOV #7$,2#RESVEC ;SETUP RESVEC
4092 023050 000025      6$: 25 ;EXECUTE OP CODE 25
4093      ;FAILURE
4094 023052 104377      ERROR 377 ;OP CODE 25 FAILED TO TRAP
4095 023054 000453      453
4096 023056 012767 023072 156024 7$: MOV #8$, $LPERR ;SETUP ERROR LOOP
4097 023064 012737 023100 000010      MOV #9$,2#RESVEC ;SETUP RESVEC
4098 023072 000045      8$: 45 ;EXECUTE OP CODE 45
4099      ;FAILURE
4100 023074 104377      ERROR 377 ;OP CODE 45 FAILED TO TRAP
4101 023076 000453      453
4102      ;
4103      ;*****
4104      ;SECTION 2-210 THRU 227
4105 023100 012767 023114 156002 9$: MOV #10$, $LPERR ;SETUP ERROR LOOP
4106 023106 012737 023132 000010      MOV #11$,2#RESVEC ;SETUP RESVEC
4107 023114 012706 001100      10$: MOV #STACK, SP ;INITIALIZE THE SP
4108 023120 012700 023126      MOV #12$,R0 ;SETUP R0 INCASE RTS
4109 023124 000210      210 ;EXECUTE OP CODE 210
4110      ;FAILURE-RTS EXECUTED
4111 023126 104377      12$: ERROR 377 ;OP CODE 210 FAILED TO TRAP
4112 023130 000453      453
4113 023132 012767 023146 155750 11$: MOV #13$, $LPERR ;SETUP ERROR LOOP
4114 023140 012737 023164 000010      MOV #14$,2#RESVEC ;SETUP RESVEC
4115 023146 012706 001100      13$: MOV #STACK, SP ;INITIALIZE THE SP
4116 023152 012700 023160      MOV #15$,R0 ;SETUP R0 TO CATCH RTS
4117 023156 000220      220 ;EXECUTE OP CODE 220
4118      ;FAILURE-RTS EXECUTED
4119 023160 104377      15$: ERROR 377 ;OP CODE 220 FAILED TO TRAP
4120 023162 000453      453
4121      ;
4122      ;*****
4123      ;SECTION 3-7000 THRU 7777
4124 023164 012767 023200 155716 14$: MOV #27$, $LPERR ;SETUP ERROR LOOP
4125 023172 012737 023206 000010      MOV #16$,2#RESVEC ;SETUP RESVEC
4126 023200 007000      27$: 7000 ;EXECUTE OP CODE 7000
4127      ;FAILURE
4128 023202 104377      ERROR 377 ;OP CODE 7000 FAILED TO TRAP
4129 023204 000453      453

```

```

4130
4131
4132
4133 023206 012767 023222 155674
4134 023214 012737 023230 000010
4135 023222 075000
4136
4137 023224 104377
4138 023226 000453
4139 023230 012767 023244 155652
4140 023236 012737 023252 000010
4141 023244 076000
4142
4143 023246 104377
4144 023250 000453
4145
4146
4147
4148 023252 012767 023266 155630
4149 023260 012737 023274 000010
4150 023266 106400
4151
4152 023270 104377
4153 023272 000453
4154
4155
4156
4157 023274 012767 023310 155606
4158 023302 012737 023322 000010
4159 023310 012706 001100
4160 023314 106700
4161
4162 023316 104377
4163 023320 000453
4164 023322 012767 023336 155560
4165 023330 012737 023344 000010
4166 023336 107000
4167
4168 023340 104377
4169 023342 000453
4170 023344 012737 000012 000010
4171 023352 012737 000340 000012
4172
4173
4174
4175
4176
4177
4178
4179
4180
4181
4182
4183
4184
4185

```

```

*****
SECTION 4-75000 THRU 76777
16$: MOV #17$, $LPERR ; SETUP ERROR LOOP
      MOV #18$, @RESVEC ; SETUP RESVEC
17$: 75000 ; EXECUTE OP CODE 75000
; FAILURE
      ERROR 377 ; OP CODE 75000 FAILED TO TRAP
      453
18$: MOV #19$, $LPERR ; SETUP ERROR LOOP
      MOV #20$, @RESVEC ; SETUP RESVEC
19$: 76000 ; EXECUTE OP CODE 76000
; FAILURE
      ERROR 377 ; OP CODE 76000 FAILED TO TRAP
      453
*****
SECTION 5-106400 THRU 106477
20$: MOV #21$, $LPERR ; SETUP ERROR LOOP
      MOV #22$, @RESVEC ; SETUP RESVEC
21$: 106400 ; EXECUTE OP CODE 106400
; FAILURE
      ERROR 377 ; OP CODE 106400 FAILED TO TRAP
      453
*****
SECTION 6-106700 THRU 107777
22$: MOV #23$, $LPERR ; SETUP ERROR LOOP
      MOV #24$, @RESVEC ; SETUP RESVEC
23$: MOV #STACK, SP ; INITIALIZE THE SP
      106700 ; EXECUTE OP CODE 106700
; FAILURE
      ERROR 377 ; OP CODE 106700 FAILED TO TRAP
      453
24$: MOV #25$, $LPERR ; SETUP ERROR LOOP
      MOV #26$, @RESVEC ; SETUP RESVEC
25$: 107000 ; EXECUTE OP CODE 107000
; FAILURE
      ERROR 377 ; OP CODE 107000 FAILED TO TRAP
      453
26$: MOV #12, @RESVEC ; RESTORE RESVEC
      MOV #PR7, @RESVEC+2 ; RESTORE RESVEC PSW
; CONTINUE

```

```

*****
*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
*
*****

```

4186	..*	2	PIR 1	SL YELLOW
4187	..*	3	PIR 2	SL YELLOW
4188	..*	4	PIR 3	SL YELLOW
4189	..*	5	BR 4	PIR 4
4190	..*	6	BR 4	PIR 5
4191	..*	7	BR 4	BR 5
4192	..*	8	BR 4	PIR 6
4193	..*	9	BR 4	PIR 7
4194	..*	10	PIR 4	BR 5
4195	..*	11	PIR 4	BR 6
4196	..*	12	PIR 4	SL YELLOW
4197	..*	13	BR 5	PIR 5
4198	..*	14	BR 5	PIR 6
4199	..*	15	BR 5	PIR 7
4200	..*	16	PIR 5	BR 6
4201	..*	17	PIR 5	SL YELLOW
4202	..*	18	BR 6	PIR 6
4203	..*	19	BR 6	PIR 7
4204	..*	20	PIR 6	SL YELLOW
4205	..*	21	PIR 7	SL YELLOW

```

4207 023360 000004 ST47: SCOPE
4208 023362 012767 026146 155700 MOV #TST50,NEXTTST ;SAVE ADDRESS OF NEXT TEST
4209 023370 012767 003706 155506 MOV #+D1990,$ICNT ;SETUP ITERATION COUNT
4210 023376 012767 023404 155502 MOV #103$,$LPADR ;SETUP LOOP ADDRESS
4211 023404 005005 103$: CLR R5 ;CLEAR BR5 DISABLE FLAG
4212 023406 005004 CLR R4 ;CLEAR BR6 DISABLE FLAG
4213 023410 005003 CLR R3 ;CLEAR BR4 DISABLE FLAG
4214 023412 032737 000020 177776 BIT #BIT4,@#PSW ;IS T BIT ON?
4215 023420 001412 BEQ 71$ ;BRANCH IF NO
4216 023422 012737 000360 000066 MOV #360,@#TPVEC+2
4217 023430 012737 000360 000242 MOV #360,@#PIRQVEC+2
4218 023436 012737 000360 000006 MOV #360,@#ERRVEC+2
4219 023444 000411 BR 72$
4220 023446 012737 000340 000066 71$: MOV #PR7,@#TPVEC+2 ;PUT PRIORITY 7 IN PRINTER VECTOR
4221 023454 012737 000340 000006 MOV #PR7,@#ERRVEC+2 ;RESTORE ERRVEC PSW
4222 023462 012737 000340 000242 MOV #PR7,@#PIRQVEC+2 ;PUT PRIORITY 7 IN PIRQ VECTOR
4223 023470 032737 000400 177570 72$: BIT #SW8,@#SWR ;SWITCH 8 ON?
4224 023476 001006 BNE 100$ ;BRANCH IF YES
4225 023500 032737 000040 177570 BIT #SW5,@#SWR ;IS BR 5 TESTING DISABLED?
4226 023506 001402 BEQ 100$ ;BRANCH IF NO
4227 023510 005205 102$: INC R5 ;SET BR 5 DISABLE FLAG
4228 023512 000403 BR 101$ ;CONTINUE
4229 023514 005767 155504 100$: TST INTER5 ;IS THERE A BR 5 DEVICE?
4230 023520 001773 BEQ 102$ ;BRANCH IF NO

```

```

4232 SECTION 1 - BR4 AND PIR1
4233 023522 032737 000400 177570 101$: BIT #SW8,@#SWR ;SWITCH 8 ON?
4234 023530 001006 BNE 68$ ;BRANCH IF YES
4235 023532 032737 000020 177570 BIT #SW4,@#SWR ;IS BR4 TESTING DISABLED?
4236 023540 001402 BEQ 68$ ;BRANCH IF NO
4237 023542 005203 INC R3 ;SET BR4 FLAG
4238 023544 000430 BR 2$ ;GO TO NEXT SECTION
4239 023546 012767 023570 155334 68$: MOV #1$, $LPERR ;SETUP ERROR LOOP
4240 023554 012737 023626 000064 MOV #2$, @#TPVEC ;SETUP PRINTER VECTOR
4241 023562 012737 023614 000240 MOV #3$, @#PIRQVEC ;SETUP PIRQ VECTOR

```

```

4242 023570 012706 001076      1$:  MOV      #1076,SP      ;INITIALIZE THE SP
4243 023574 000237              SPL      7              ;ENSURE CPU AT LEVEL 7
4244 023576 052737 001000 177772  BIS      #BIT9,@#PIRQ    ;SET PIR LEVEL 1
4245 023604 004767 002204      JSR      PC,LEVEL4      ;GO GET BR4
4246 023610 000230              SPL      0              ;SET CPU AT ZERO
4247 023612 000240              NOP                    ;USED WITH SPL
4248
4249 023614 005037 177772      3$:  CLR      @#PIRQ        ;CLEAR THE PIRQ
4250 023620 005077 155316      CLR      @STPS         ;CLEAR THE PRINTER FLAG
4251 023624 104302      ERROR   302          ;PIR 1 DID NOT DISABLE ON BR
4252
;*****
;SECTION 2 - PIR 1 AND SL YELLOW
4253
4254 023626 005077 155310      2$:  CLR      @STPS         ;CLEAR PRINTER INT FLAG
4255 023632 012767 023660 155250  MOV      #4$, $LPERR    ;SETUP ERROR LOOP
4256 023640 012737 023716 000004  MOV      #5$, @#ERRVEC  ;SETUP LOCATION 4
4257 023646 012737 023676 000240  MOV      #6$, @#PIRQVEC ;SETUP PIRQ VECTOR
4258 023654 005037 177772      CLR      @#PIRQ        ;CLEAR LEVEL 1
4259 023660 012706 000376      4$:  MOV      #376, SP      ;SETUP THE SP
4260 023664 052737 001000 177772  BIS      #BIT9,@#PIRQ    ;SET LEVEL 1
4261 023672 000230              SPL      0              ;SET CPU AT ZERO
4262 023674 011616      MOV      (SP), (SP)    ;EXECUTE YELL ZONE INSTR
4263
;FAILURE, PIR CAME THRU
4264 023676 012706 001100      6$:  MOV      #STACK, SP   ;RESET SP
4265 023702 012737 036242 000004  MOV      #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
4266 023710 005037 177772      CLR      @#PIRQ        ;CLEAR LEVEL 1
4267 023714 104303      ERROR   303          ;PIR CAME THRU ON YELLOW ZONE
4268
;*****
;SECTION 3 - PIR 2 AND SL YELLOW
4269
4270
4271 023716 005037 177772      5$:  CLR      @#PIRQ        ;CLEAR LEVEL 1
4272 023722 012767 023744 155160  MOV      #7$, $LPERR    ;SETUP ERROR LOOP
4273 023730 012737 024002 000004  MOV      #8$, @#ERRVEC  ;SETUP LOCATION 4
4274 023736 012737 023762 000240  MOV      #9$, @#PIRQVEC ;SETUP PIRQ VECTOR
4275 023744 012706 000376      7$:  MOV      #376, SP      ;SETUP THE SP
4276 023750 052737 002000 177772  BIS      #BIT10,@#PIRQ  ;SET LEVEL 2
4277 023756 000231      SPL      1              ;SET CPU AT LEVEL 1
4278 023760 011616      MOV      (SP), (SP)    ;EXECUTE TRAP CAUSING INSTR
4279
;FAILURE, PIR CAME THRU
4280 023762 012706 001100      9$:  MOV      #STACK, SP   ;RESET SP
4281 023766 012737 036242 000004  MOV      #CPUSPUR,@#ERRVEC ;RESET ERRVEC
4282 023774 005037 177772      CLR      @#PIRQ        ;CLEAR LEVEL 2
4283 024000 104304      ERROR   304          ;PIR 2 CAME THRU ON YELLOW ZONE
4284
;*****
;SECTION 4 - PIR 3 AND YEL ZONE
4285
4286
4287 024002 005037 177772      8$:  CLR      @#PIRQ        ;CLEAR LEVEL 2
4288 024006 012767 024030 155074  MOV      #10$, $LPERR   ;SETUP ERROR LOOP
4289 024014 012737 024066 000004  MOV      #11$, @#ERRVEC ;SETUP ERR VECTOR
4290 024022 012737 024046 000240  MOV      #12$, @#PIRQVEC ;SETUP PIRQ VECTOR
4291 024030 012706 000376      10$: MOV      #376, SP      ;SETUP THE SP
4292 024034 052737 004000 177772  BIS      #BIT11,@#PIRQ  ;SET LEVEL 3
4293 024042 000232      SPL      2              ;SET CPU AT LEVEL 2
4294 024044 011616      MOV      (SP), (SP)    ;EXECUTE TRAP CAUSING INSTR
4295
;FAILURE, PIR CAME THRU
4296 024046 012706 001100      12$: MOV      #STACK, SP   ;RESET SP
4297 024052 012737 036242 000004  MOV      #CPUSPUR,@#ERRVEC ;RESET LOCATION 4

```

```

4298 024060 005037 177772          CLR    @#PIRQ          ;CLEAR LEVEL 3
4299 024064 104305          ERROR  305           ;PIR 3 CAME THRU ON YELLOW ZONE
4300                                     ;*****
4301                                     ;SECTION 5- PIR4 AND BR4
4302 024066 012706 001100 11$:  MOV    #STACK,SP      ;RESTORE THE SP
4303 024072 005703          TST    R3             ;IS BR4 TESTING DISABLED?
4304 024074 001402          BEQ    70$           ;BRANCH IF NO
4305 024076 000167 000410          JMP    29$           ;SKIP BR4 SECTIONS
4306 024102 012767 024124 155000 70$:  MOV    #13$, $LPERR    ;SETUP ERROR LOOP
4307 024110 012737 024160 000240  MOV    #15$, @#PIRQVEC ;SETUP PIRQ VEC
4308 024116 012737 024146 000064  MOV    #14$, @#TPVEC   ;SETUP TPVEC
4309 024124 012706 001100 13$:  MOV    #STACK,SP      ;INITIALIZE THE SP
4310 024130 012737 010000 177772  MOV    #BIT12, @#PIRQ  ;SET PIR LEVEL 4
4311 024136 004767 001652          JSR    PC, LEVEL4    ;GO GET BR 4
4312 024142 000233          SPL                    ;LOWER CPU
4313 024144 000240          NOP                    ;REQUIRED BECAUSE OF SPL
4314                                     ;FAILURE BR4 CAME THRU
4315 024146 005077 154770 14$:  CLR    @#STPS        ;CLEAR PRINTER INT FLAG
4316 024152 005037 177772          CLR    @#PIRQ        ;CLEAR LEVEL 4
4317 024156 104306          ERROR  306           ;BR4 CAME THRU ON PIR4
4318                                     ;
4319                                     ;*****
4320                                     ;SECTION 6- PIR 5 AND BR4
4321 024160 012767 024202 154722 15$:  MOV    #16$, $LPERR    ;SETUP ERROR LOOP
4322 024166 012737 024224 000064  MOV    #17$, @#TPVEC   ;SETUP BR4 VEC
4323 024174 012737 024236 000240  MOV    #18$, @#PIRQVEC ;SETUP PIRQ VEC
4324 024202 012706 001100 16$:  MOV    #STACK,SP      ;INITIALIZE THE SP
4325 024206 012737 020000 177772  MOV    #BIT13, @#PIRQ  ;SET PIR LEVEL 5
4326 024214 004767 001574          JSR    PC, LEVEL4    ;GO GET BR4
4327 024220 000233          SPL                    ;SET CPU AT 3
4328 024222 000240          NOP                    ;REQUIRED BECAUSE OF SPL
4329                                     ;FAILURE BR4 CAME THRU
4330 024224 005077 154712 17$:  CLR    @#STPS        ;CLEAR BR4 INT FLAG
4331 024230 005037 177772          CLR    @#PIRQ        ;CLEAR PIR5
4332 024234 104307          ERROR  307           ;BR4 CAME THRU ON PIR5
4333                                     ;
4334                                     ;*****
4335                                     ;SECTION 7- BR5 AND BR4
4336                                     ;
4337 024236 005037 177772 18$:  CLR    @#PIRQ        ;CLEAR PIR LEVEL 5
4338 024242 005077 154674          CLR    @#STPS        ;CLEAR PRINTER INT FLAG
4339 024246 005705          TST    R5             ;IS BR 5 TESTING DISABLED?
4340 024250 001040          BNE    23$           ;BRANCH IF YES
4341 024252 012767 024320 154630 20$:  MOV    #21$, $LPERR    ;SETUP ERROR LOOP
4342 024260 012737 024340 000064  MOV    #22$, @#TPVEC   ;SETUP BR4 VECTOR
4343 024266 016700 154734          MOV    INT5VEC, R0    ;GET BR5 VECTOR
4344 024272 012720 024352          MOV    #23$, (R0)+    ;PUT ADDRESS OF 23$ IN VECTOR
4345 024276 032737 000020 177776  BIT    #BIT4, @#PSW    ;IS T BIT ON?
4346 024304 001403          BEQ    73$           ;BRANCH IF NO
4347 024306 012710 000360          MOV    #360, (R0)     ;SETUP VECTOR PSW
4348 024312 000402          BR    21$            ;
4349 024314 012710 000340 73$:  MOV    #PR7, (R0)     ;PUT PR7 IN VECTOR+2
4350 024320 012706 001100 21$:  MOV    #STACK,SP      ;INITIALIZE THE SP
4351 024324 004767 001472          JSR    PC, LEVEL5    ;GO GET BR5
4352 024330 004767 001460          JSR    PC, LEVEL4    ;GO GET BR4
4353 024334 000233          SPL                    ;SET CPU AT LEVEL 3

```

```

4354 024336 000240      NOP ;REQUIRED BECAUSE OF SPL
4355      ;FAILURE, BR4 CAME IN
4356 024340 004767 001524 22$: JSR PC,KILBR5 ;GET RID OF BR 5
4357 024344 005077 154572 CLR @STPS ;CLEAR BR4 INT ENABLE
4358 024350 104310 ERROR 310 ;BR4 CAME THRU ON BR5
4359
4360 ;*****
4361 ;SECTION 8- PIR6 AND BR4
4362 024352 004767 001512 23$: JSR PC,KILBR5 ;GET RID OF BR 5
4363 024356 012767 024400 154524 MOV #24$,SLPERR ;SETUP ERROR LOOP
4364 024364 012737 024422 000064 MOV #25$,@#TPVEC ;SETUP BR4 INTERRUPT VECTOR.
4365 024372 012737 024434 000240 MOV #26$,@#PIRQVEC ;SETUP PIRQ VECTOR
4366 024400 012706 001076 24$: MOV #1076,SP ;INITIALIZE THE SP
4367 024404 012737 040000 177772 MOV #BIT14,@#PIRQ ;SET PIR LEVEL 6
4368 024412 004767 001376 JSR PC,LEVEL4 ;GO GET BR4
4369 024416 000233 SPL 3 ;SET CPU AT LEVEL 3
4370 024420 000240 NOP ;REQUIRED BECAUSE OF SPL
4371      ;FAILURE, BR4 CAME IN
4372 024422 005037 177772 25$: CLR @#PIRQ ;CLEAR PIR LEVEL 6
4373 024426 005077 154510 CLR @STPS ;CLEAR PRINTER INTER FLAG
4374 024432 104311 ERROR 311 ;BR4 CAME IN ON PIR6
4375
4376 ;*****
4377 ;SECTION 9- PIR 7 AND BR4
4378 024434 012767 024456 154446 26$: MOV #27$,SLPERR ;SETUP ERROR LOOP
4379 024442 012737 024500 000064 MOV #28$,@#TPVEC ;SETUP BR4 VECTOR
4380 024450 012737 024512 000240 MOV #29$,@#PIRQVEC ;SETUP PIRQ VECTOR
4381 024456 012706 001076 27$: MOV #1076,SP ;SETUP THE SP
4382 024462 012737 100000 177772 MOV #BIT15,@#PIRQ ;SET PIR LEVEL 7
4383 024470 004767 001320 JSR PC,LEVEL4 ;GO GET BR4
4384 024474 000233 SPL 3 ;LOWER CPU TO 3
4385 024476 000240 NOP ;REQUIRED BECAUSE OF SPL
4386      ;FAILURE, BR4 CAME IN
4387 024500 005077 154436 28$: CLR @STPS ;CLEAR PRINTER INTERR FLAG
4388 024504 005037 177772 CLR @#PIRQ ;CLEAR PIR LEVEL 7
4389 024510 104312 ERROR 312 ;BR4 CAME IN ON PIR7
4390
4391 ;*****
4392 ;SECTION 10- BR5 AND PIR4
4393 024512 005077 154424 29$: CLR @STPS ;CLEAR BR4 INTERR FLAG
4394 024516 005037 177772 CLR @#PIRQ ;CLEAR PIRQ LEVEL 7
4395 024522 005705 TST R5 ;IS THERE A BR5 DEVICE?
4396 024524 001041 BNE 32$ ;BRANCH TO SECTION 11 IF NO
4397 024526 012767 024574 154354 MOV #30$,SLPERR ;SETUP ERROR LOOP
4398 024534 012737 024616 000240 MOV #31$,@#PIRQVEC ;SETUP PIRQ VECTOR
4399 024542 016700 154460 MOV INTEVEC,RO ;GET ADDR OF BR 5 VECTOR
4400 024546 012720 024630 MOV #32$, (RO)+ ;SETUP BR5 VECTOR
4401 024552 032737 000020 177776 BIT #BIT4,@#PSW ;IS T BIT ON?
4402 024560 001403 BEQ 74$ ;BRANCH IF NO
4403 024562 012710 000360 MOV #360, (RO)
4404 024566 000462 BR 75$
4405 024570 012710 000340 74$: MOV #PR7, (RO)
4406 024574 012706 001100 30$: MOV #STACK, SP ;SETUP THE SP
4407 024600 012737 010000 177772 MOV #BIT12,@#PIRQ ;SET PIR4
4408 024606 004767 001210 JSR PC,LEVEL5 ;GO GET BR5
4409 024612 000233 SPL 3 ;LOWER CPU TO LEVEL 3

```



```

4410 024614 000240 NOP ;REQUIRED BECAUSE OF SPL
4411 :FAILURE, PIR4 CAME IN
4412 024616 004767 001246 31$: JSP PC,KILBRS ;GET RID OF BR 5
4413 024622 005037 177772 CLR 2#PIRQ ;CLEAR PIR LEVEL 4
4414 024626 104313 ERROR 313 ;PIR4 CAME IN ON BR5
4415
4416 :*****
4417 :SECTION 11- BR6 AND PIR4
4418 024630 005037 177772 32$: CLR 2#PIRQ ;CLEAR PIR LEVEL 4
4419 024634 004767 001230 JSR PC,KILBRS ;GET RID OF BR 5
4420 024640 032737 000400 177570 BIT #SW8,2#SWR ;SWITCH 8 ON?
4421 024646 001004 BNE 80$ ;BRANCH IF YES
4422 024650 032737 000100 177570 BIT #SW6,2#SWR ;IS BR6 TESTING DISABLED?
4423 024656 001003 BNE 33$ ;BRANCH IF YES
4424 024660 005767 154346 80$: TST INTER6 ;IS THERE A BR6 DEVICE?
4425 024664 001002 BNE 34$ ;BRANCH IF YES
4426 024666 005204 33$: INC R4 ;SET BR 6 DISABLE FLAG
4427 024670 000441 BR 37$ ;GO TO SECTION 12
4428 024672 012767 024740 154210 34$: MOV #35$, $LPERR ;SETUP ERROR LOOP
4429 024700 012737 024762 000240 MOV #36$, 2#PIRQVEC ;SETUP PIR VECTOR
4430 024706 016701 154322 MOV INT6VEC, R1 ;GET BR 6 VECTOR
4431 024712 012721 024774 MOV #37$, (R1)+ ;PU ADDRESS OF 37$ IN VECTOR
4432 024716 032737 000020 177776 BIT #BIT4, 2#PSW ;IS T BIT ON?
4433 024724 001403 BEQ 75$ ;BRANCH IF NO
4434 024726 012711 000360 MOV #360, (R1)
4435 024732 000402 BR 35$
4436 024734 012711 000340 75$: MOV #PR7, (R1) ;PUT PRIORITY OF 7 IN VECTOR+2
4437 024740 012706 001100 35$: MOV #STACK, SP ;INITIALIZE THE SP
4438 024744 012737 010000 177772 MOV #BIT12, 2#PIRQ ;SET PIR LEVEL 4
4439 024752 004767 001054 JSR PC, LEVEL6 ;GO GET BR6
4440 024756 000233 SPL 3 ;LOWER CPU TO LEVEL 3
4441 024760 000240 NOP ;REQUIRED BECAUSE OF SPL
4442 :FAILURE, PIR4 CAME IN
4443 024762 005077 154250 36$: CLR 2#INT6ST ;CLEAR BR6 INTERR FLAG
4444 024766 005037 177772 CLR 2#PIRQ ;CLEAR PIR LEVEL 4
4445 024772 104314 ERROR 314 ;PIR4 CAME IN ON BR6
4446
4447 :*****
4448 :SECTION 12- PIR4 AND STACK LIMIT YELLOW
4449 024774 005077 154236 37$: CLR 2#INT6ST ;CLEAR BR6 INTERR. FLAG
4450 025000 005037 177772 CLR 2#PIRQ ;CLEAR LEVEL 4
4451 025004 012767 025026 154076 MOV #38$, $LPERR ;SETUP ERROR LOOP
4452 025012 012737 025044 000240 MOV #39$, 2#PIRQVEC ;SETUP PIRQ VECTOR
4453 025020 012737 025064 000004 MOV #40$, 2#ERRVEC ;SETUP YELL ZONE VECTOR
4454 025026 012706 000376 38$: MOV #376, SP ;SETUP THE SP
4455 025032 052737 010000 177772 BIS #BIT12, 2#PIRQ ;SET LEVEL 4
4456 025040 000233 SPL 3 ;SET CPU AT LEVEL 3
4457 025042 011616 MOV (SP), (SP) ;EXECUTE TRAP CAUSING INSTR
4458 :FAILURE, PIR 4 CAME THRU
4459 025044 012706 001100 39$: MOV #STACK, SP ;RESET THE SP
4460 025050 012737 036242 000004 MOV #CPUSPUR, 2#ERRVEC ;RESTORE ERR VEC
4461 025056 005037 177772 CLR 2#PIRQ ;CLEAR PIR LEVEL 4
4462 025062 104315 ERROR 315 ;PIR4 CAME IN ON SL YELLOW
4463
4464 :*****
4465 :SECTION 13- BR5 AND PIR5

```



```

4466 025064 012706 001100 40$: MOV #STACK, SP ;RESTORE THE SP
4467 025070 012737 036242 000004 MOV #CPUSPR, @ERRVEC ;RESTORE LOCATION 4
4468 025076 005037 177772 CLR @PIRQ ;CLEAR PIR 4
4469 025102 005705 TST R5 ;IS BR5 DISABLED?
4470 025104 001105 BNE 49$ ;BRANCH IF YES TO SECTION 16
4471 025106 012767 025130 153774 MOV #41$, $LPERR ;SETUP ERROR LOOP
4472 025114 012777 025152 154104 MOV #42$, @INTSVEC ;SETUP BR5 VECTOR
4473 025122 012737 025164 000240 MOV #43$, @PIRQVEC ;SETUP PIRQ VECTOR
4474 025130 012706 001076 41$: MOV #1076, SP ;INITIALIZE THE SP
4475 025134 012737 020000 177772 MOV #BIT13, @PIRQ ;SET PIR LEVEL 5
4476 025142 004767 000654 JSR PC, LEVELS ;GO GET BR5
4477 025146 000234 SPL 4 ;LOWER CPU TO LEVEL 4
4478 025150 000240 NOP ;REQUIRED BECAUSE OF SPL
4479 ;FAILURE BR5 CAME IN
4480 025152 004767 000712 42$: JSR PC, KILBR5 ;GET RID OF BR 5
4481 025156 005037 177772 CLR @PIRQ ;CLEAR PIR5
4482 025162 104316 ERROR 316 ;BR5 CAME IN ON PIR5
4483
4484 ;*****
4485 ;SECTION 14- BR5 AND PIR6
4486 025164 012767 025206 153716 43$: MOV #44$, $LPERR ;SETUP ERROR LOOP
4487 025172 012777 025230 154026 MOV #45$, @INTSVEC ;SETUP BR5 VEC
4488 025200 012737 025242 000240 MOV #46$, @PIRQVEC ;SETUP PIRQ VECTOR
4489 025206 012706 001076 44$: MOV #1076, SP ;INITIALIZE THE SP
4490 025212 012737 040000 177772 MOV #BIT14, @PIRQ ;SET PIR LEVEL 6
4491 025220 004767 000576 JSR PC, LEVELS ;GO GET BR5
4492 025224 000234 SPL 4 ;SET CPU AT LEVEL 4
4493 025226 000240 NOP ;REQUIRED BECAUSE OF SPL
4494 ;FAILURE BR5 CAME IN
4495 025230 004767 000634 45$: JSR PC, KILBR5 ;GET RID OF BR5
4496 025234 005037 177772 CLR @PIRQ ;CLEAR PIR LEVEL 6
4497 025240 104317 ERROR 317 ;BR5 CAME IN ON PIR6
4498
4499 ;*****
4500 ;SECTION 15- BR5 AND PIR7
4501 025242 012767 025264 153640 46$: MOV #47$, $LPERR ;SETUP ERROR LOOP
4502 025250 012777 025306 153750 MOV #48$, @INTSVEC ;SETUP BR5 VECTOR
4503 025256 012737 025320 000240 MOV #49$, @PIRQVEC ;SETUP PIRQ VECTOR
4504 025264 012706 001076 47$: MOV #1076, SP ;INITIALIZE THE SP
4505 025270 012737 100000 177772 MOV #BIT15, @PIRQ ;SET PIR LEVEL 7
4506 025276 004767 000520 JSR PC, LEVELS ;GO GET BR5
4507 025302 000234 SPL 4 ;ALLOW BRQ
4508 025304 000240 NOP ;REQUIRED BECAUSE OF SPL
4509 ;FAILURE BR5 CAME IN
4510 025306 004767 000556 48$: JSR PC, KILBR5 ;GET RID OF BR5
4511 025312 005037 177772 CLR @PIRQ ;CLEAR PIR LEVEL 7
4512 025316 104320 ERROR 320 ;BR5 CAME IN ON PIR7
4513
4514 ;*****
4515 ;SECTION 16- PIR5 AND BR6
4516 025320 004767 000544 49$: JSR PC, KILBR5 ;GET RID OF BR 5
4517 025324 005704 TST R4 ;IS BR6 TESTING DISABLED?
4518 025326 001030 BNE 52$ ;BRANCH IF YES TO SECTION 17
4519 025330 012767 025352 153552 MOV #50$, $LPERR ;SETUP ERROR LOOP
4520 025336 012737 025376 000240 MOV #51$, @PIRQVEC ;SETUP PIR VECTOR
4521 025344 012777 025410 153662 MOV #52$, @INT6VEC ;SETUP BR6 VECTOR

```

```

4522 025352 012706 001076 50$: MOV #1076,SP ;INITIALIZE THE SP
4523 025356 012737 020000 177772 MOV #BIT13,2#PIRQ ;SET IR LEVEL 5
4524 025364 004767 000442 JSR PC,LEVEL6 ;GO GET BR6
4525 025370 000234 SPL 4 ;ALLOW BRQ
4526 025372 000240 NOP ;REQUIRED BECAUSE OF SPL
4527 ;FAILURE, PIR 5 CAME IN
4528 025374 000000 HALT ;DEBUG ONLY
4529 025376 005077 153634 51$: CLR 2#INT6ST ;CLEAR BR6 INTERR FLAG
4530 025402 005037 177772 CLR 2#PIRQ ;CLEAR PIR LEVEL 5
4531 025406 104321 ERROR 321 ;PIR 5 CAME IN ON BR6
4532 ;
4533 ;*****
4534 ;SECTION 17- PIR 5 AND STACK LIMIT YELLOW
4535 025410 005077 153622 52$: CLR 2#INT6ST ;CLEAR BR6 INTERR FLAG
4536 025414 012767 025436 153466 MOV #53$,2#SLPERR ;SETUP ERROR LOOP
4537 025422 012737 025454 000240 MOV #54$,2#PIRQVEC ;SETUP PIRQVEC
4538 025430 012737 025474 000004 MOV #55$,2#ERRVEC ;SETUP LOCATION 4
4539 025436 012706 000376 53$: MOV #376,SP ;SETUP THE SP
4540 025442 052737 020000 177772 BIS #BIT13,2#PIRQ ;SET LEVEL 5
4541 025450 000234 SPL 4 ;SET CPU AT 4
4542 025452 011616 MOV (SP), (SP) ;EXECUTE YEL ZONE INSTR
4543 ;FAILURE, PIRS CAME IN ON SL YELLOW
4544 025454 005037 177772 54$: CLR 2#PIRQ ;CLEAR LEVEL 5
4545 025460 012737 036242 000004 MOV #CPUSPUR,2#ERRVEC ;RESTORE ERRVEC
4546 025466 012706 001100 MOV #STACK,SP ;RESET THE SP
4547 025472 104322 ERROR 322 ;PIRS CAME IN ON SL YELLOW
4548 ;
4549 ;*****
4550 ;SECTION 18- BR6 AND PIR6
4551 025474 012706 001100 55$: MOV #STACK,SP ;RESTORE THE SP
4552 025500 005704 TST R4 ;IS BR6 TESTING DISABLED?
4553 025502 001056 BNE 61$ ;BRANCH IF YES TO SECTION 20
4554 025504 012767 025526 153376 MOV #56$,2#SLPERR ;SETUP ERROR LOOP
4555 025512 012777 025550 153514 MOV #57$,2#INT6VEC ;SETUP BR6 INTERR VECTOR
4556 025520 012737 025562 000240 MOV #58$,2#PIRQVEC ;SETUP PIR VECTOR
4557 025526 012706 001076 56$: MOV #1076,SP ;SETUP THE SP
4558 025532 012737 040000 177772 MOV #BIT14,2#PIRQ ;SET PIR LEVEL 6
4559 025540 004767 000266 JSR PC,LEVEL6 ;GO GET BR6
4560 025544 000235 SPL 5 ;LOWER CPU TO 5
4561 025546 000240 NOP ;REQUIRED BECAUSE OF SPL
4562 ;FAILURE, BR6 CAME IN ON PIR6
4563 025550 005077 153462 57$: CLR 2#INT6ST ;CLEAR BR6 INTERR FLAG
4564 025554 005037 177772 CLR 2#PIRQ ;CLEAR LEVEL 6
4565 025560 104323 ERROR 323 ;BR6 CAME IN ON PIR6
4566 ;
4567 ;*****
4568 ;SECTION 19- BR6 AND PIR7
4569 025562 012767 025604 153320 58$: MOV #59$,2#SLPERR ;SETUP ERROR LOOP
4570 025570 012777 025626 153436 MOV #60$,2#INT6VEC ;SETUP BR6 VECTOR
4571 025576 012737 025640 000240 MOV #61$,2#PIRQVEC ;SETUP PIRQ VECTOR
4572 025604 012706 001076 59$: MOV #1076,SP ;SETUP THE SP
4573 025610 012737 100000 177772 MOV #BIT15,2#PIRQ ;SET PIR 7
4574 025616 004767 000210 JSR PC,LEVEL6 ;GO GET BR6
4575 025622 000235 SPL 5 ;LOWER CPU TO 5
4576 025624 000240 NOP ;REQUIRED BECAUSE OF SPL
4577 ;FAILURE, BR6 CAME IN ON PIR7

```

```

4578 025626 005077 153404      60$: CLR      @INT6ST      ;CLEAR BR6 INTERR FLAG
4579 025632 005037 177772      CLR      @PIRQ        ;CLEAR LEVEL 7
4580 025636 104324      ERROR    324          ;BR6 CAME IN ON PIR7
4581
4582 ;*****
4583 ;SECTION 20- PIR6 AND SL YELLOW
4584 025640 012767 025662 153242 61$: MOV      @62$,@SLPERR ;SETUP LOOP ADDRESS
4585 025646 012737 025700 000240      MOV      @63$,@PIRQVEC ;SETUP PIRQ VECTOR
4586 025654 012737 025712 000004      MOV      @64$,@ERRVEC  ;SETUP LOCATION 4
4587 025662 012706 000376      MOV      @376,SP       ;SETUP THE SP
4588 025666 052737 040000 177772 62$: BIS      @BIT14,@PIRQ ;SET LEVEL 6
4589 025674 000235      SPL      5             ;SET CPU AT LEVEL 5
4590 025676 011616      MOV      (SP),(SP)     ;EXECUTE YELL ZONE INSTR
4591 ;FAILURE PIR 6 CAME IN ON SL YELLOW
4592 025700 005037 177772      63$: CLR      @PIRQ        ;CLEAR PIR 6
4593 025704 012706 001100      MOV      @STACK,SP    ;RESTORE THE SP
4594 025710 104325      ERROR    325          ;PIR 6 CAME IN ON SL YELLOW
4595
4596 ;*****
4597 ;SECTION 21-PIR7 AND STACK LIMIT YELLOW
4598 025712 005077 153320      64$: CLR      @INT6ST      ;ENSURE BR6 FLAG CLEAR
4599 025716 012767 025740 153164      MOV      @65$,@SLPERR  ;SETUP ERROR LOOP
4600 025724 012737 025756 000240      MOV      @66$,@PIRQVEC ;SETUP PIRQ VECTOR
4601 025732 012737 025770 000004      MOV      @67$,@ERRVEC  ;SETUP THE ERROR VECTOR
4602 025740 012706 000376      MOV      @376,SP       ;SETUP THE SP
4603 025744 052737 100000 177772 65$: BIS      @BIT15,@PIRQ ;SET LEVEL 7
4604 025752 000236      SPL      6             ;SET UP AT 6
4605 025754 011616      MOV      (SP),(SP)     ;CAUSE YELL ZONE
4606 ;FAILURE PIR 7 CAME IN ON SL YELLOW
4607 025756 005037 177772      66$: CLR      @PIRQ        ;CLEAR LEVEL 7
4608 025762 012706 001100      MOV      @STACK,SP    ;RESTORE THE SP
4609 025766 104326      ERROR    326          ;PIR7 CAME IN ON SL YELLOW
4610
4611 ;END OF TEST
4612 025770 012706 001100      67$: MOV      @STACK,SP  ;RESET THE SP
4613 025774 005037 177772      CLR      @PIRQ        ;CLEAR LEVEL 7
4614 026000 012737 036242 000004      MOV      @CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
4615 026006 005037 177766      CLR      @CPUERR      ;CLEAR OUT ERROR REG
4616 026012 000455      BR       TST50         ;CONTINUE
4617
4618 ;ROUTINES TO GET BUS REQUESTS
4619 026014 006077 153122      LEVEL4: ROR     @STPS    ;GET A BR 4
4620 026020 000445      BR       WAIT         ;GO WAIT
4621 026022 012777 000311 153200      LEVEL5: MOV     @311,@INTSST ;START BR 5 INTERRUPT
4622 026030 000441      BR       WAIT         ;GO WAIT
4623 026032 026727 153176 000100      LEVEL6: CMP     INT6VEC,@100 ;IS BR6 DEVICE KW11-L?
4624 026040 001004      BNE     IS           ;BRANCH IF NO
4625 026042 012777 000100 153166      MOV     @BIT6,@INT6ST ;SET INTERR BIT
4626 026050 000431      BR       WAIT         ;GO WAIT
4627 026052 012737 000001 172544      IS:     MOV     @1,@PLKC  ;SET KW11-P COUNTER
4628 026060 012777 000105 153150      MOV     @105,@INT6ST ;SET INTERR BIT
4629 026066 000422      BR       WAIT
4630 026070 012777 026126 153130      KILBR5: MOV    @25,@INT5VEC ;SET UP INTER 5 VEC
4631 026076 042777 000100 153124      BIC    @BIT6,@INT5ST ;ENSURE INTERRUPT FLAG CLEAR
4632 026104 005037 177772      CLR    @PIRQ        ;ENSURE PIRQ REG CLEAR
4633 026110 005077 153026      CLR    @STPS

```

F10

```

4634 026114 000230          SPL      0          ;LET BR 5 COME IN
4635 026116 005000          CLR      RO
4636 026120 005200          3$: INC      RO
4637 026122 001376          BNE     3$
4638 026124 000406          BR      ENOGET      ;DON'T CHANGE STACK
4639 026126 062706 000004  2$: ADD     #4,SP      ;RESTORE THE SP
4640 026132 000403          BR      ENOGET
4641 026134 005000          WAIT:  CLR      RO      ;WAIT FOR
4642 026136 005200          2$: INC      RO      ;THE INTERRUPT
4643 026140 001376          BNE     2$          ;TO COME IN
4644 026142 000237          ENOGET: SPL     7
4645 026144 000207          RTS      PC          ;RETURN
4646
4647 ;*****
4648 ;*TEST 50          GPR SET 1 SELECT TEST
4649 ;*
4650 ;* THIS TEST FIRST ENSURES THAT PSW BIT 11 SETS AND CLEARS.
4651 ;* IT THEN ENSURES THAT GRAC GOREG SET 1 AND GSREG SET 1 GOES
4652 ;* HIGH FOR THE MUX SELECTS LL, LH, AND HL.
4653 ;* MUX SELECT HH WILL BE TESTED IN SUPERVISOR MODE.
4654 ;*****
4654 026146 000004          †TST50: SCOPE
4655 026150 012767 026576 153016  MOV     #TST51,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
4656 026156 012767 026576 153104  MOV     #TST51,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
4657 026164 052737 004000 177776  BIS     #BIT11,@#PSW    ;SET REG SET SELECT BIT
4658 026172 032737 004000 177776  BIT     #BIT11,@#PSW    ;DID BIT 11 SET?
4659 026200 001004          BNE     20$          ;BRANCH IF YES
4660 026202 042737 004000 177776  BIC     #BIT11,@#PSW
4661 026210 104327          ERROR   327          ;EITHER PSW BIT 11 DOES NOT SET OR TMCF
4662 ;*
4663 ;* CLK HI PS DOES NOT GO LOW OR PSW BIT
4664 ;* 11 DOES NOT GET TO OR THRU THE DMUX.
4664 026212 105037 177777 20$: CLR     @#PSW+1      ;CLEAR BIT 11
4665 026216 032737 004000 177776  BIT     #BIT11,@#PSW    ;DID BIT 11 CLEAR?
4666 026224 001402          BEQ     1$          ;BRANCH IF YES
4667 026226 104377          ERROR   377
4668 026230 000457          ;PSW BIT 11 DID NOT CLEAR
4669 ;*****
4670 ;UPAD 5 TEST
4671 026232 005067 152736 1$: CLR     $ESCAPE
4672 026236 012767 026244 152644  MOV     #10$,SLPERR
4673 026244 012702 177777 10$: MOV     #-1,R2      ;SETUP R2
4674 026250 052737 004000 177776  BIS     #BIT11,@#PSW    ;SELECT REG SET 1
4675 026256 005002          CLR     R12          ;CLEAR R12
4676 026260 042737 004000 177776  BIC     #BIT11,@#PSW    ;GO BACK TO REG SET 0
4677 026266 005702          TST     R2          ;DID R2 CLEAR?
4678 026270 001002          BNE     2$          ;BRANCH IF NO
4679 026272 104330          ERROR   330          ;CLEAR R10 ACTUALLY CLEARED R2
4680 026274 000403          BR      3$
4681 026276 020202 2$: CMP     R2,R2      ;WAS R2 SOURCE AFFECTED BY R12?
4682 026300 001401          BEQ     3$          ;BRANCH IF NO
4683 026302 104331          ERROR   331          ;R2 SRC WAS AFFECTED BY CLR R12
4684 ;*****
4685 ;UPAD 0 TEST
4686 026304 012767 026312 152576 3$: MOV     #11$,SLPERR
4687 026312 005002 11$: CLR     R2          ;SETUP R2
4688 026314 052737 004000 177776  BIS     #BIT11,@#PSW    ;GO TO REG SET 1
4689 026322 012202          MOV     (R12)+,R12    ;EXECUTE A UPAD 0 INSTRUCTION
    
```

G10

```

4690 026324 042737 004000 177776      BIC      #BIT11,2#PSW      ;COME BACK TO SET 0.
4691 026332 005702                    TST      R2                ;DID DESTINATION CHANGE
4692 026334 001402                    BEQ      4$                ;BRANCH IF NO
4693 026336 104332                    ERROR   332                ;R2 DST GOT CHANGED ON (R12)+
4694 026340 000403                    BR       5$
4695 026342 020202                    4$:    CMP      R2,R2        ;DID SRC R2 GET CHANGED?
4696 026344 001401                    BEQ      5$                ;BRANCH IF NO
4697 026346 104333                    ERROR   333                ;R2 SRC GOT CHANGED ON (R12)+
4698                                     ;:*****
4699                                     ;UPAD 2 TEST
4700 026350 012767 026356 152532    5$:    MOV      #12$,SLPERR
4701 026356 012705 026440            12$:   MOV      #9$,R5          ;SETUP R5
4702 026362 052737 004000 177776    BIS      #BIT11,2#PSW      ;GO TO SET 1
4703 026370 012705 026414            MOV      #6$,R15          ;SETUP R15
4704 026374 020527 026414            CMP      R15,#6$          ;IS THERE STUCK BITS IN R15 SRC?
4705 026400 001401                    BEQ      7$                ;BRANCH IF NO
4706 026402 000475                    BR       TST51            ;CAN'T DO THIS TEST, GO TO STUCK BIT TEST
4707 026404 020505                    7$:    CMP      R15,R15      ;IS THERE STUCK BITS IN R15 DST?
4708 026406 001401                    BEQ      8$                ;BRANCH IF NO
4709 026410 000472                    BR       TST51            ;CAN'T DO THIS TEST, GO TO STUCK BIT TEST
4710 026412 006400                    8$:    MARK     0            ;EXECUTE A UPAD 2 INSTRUCTION
4711 026414 026705 177774            6$:    CMP      6$,R15        ;DID R15 DST GET LOADED?
4712 026420 001425                    BEQ      16$              ;BRANCH IF YES
4713 026422 042737 004000 177776    BIC      #BIT11,2#PSW      ;GO BACK TO SET 0
4714 026430 012706 001100            MOV      #STACK,SP        ;RESTORE THE SP
4715 026434 104334                    ERROR   334                ;R15 DST BAD AFTER UPAD2
4716 026436 000416                    BR       16$
4717 026440 022705 026440            9$:    CMP      #9$,R15        ;IS DST ALSO BAD?
4718 026444 001407                    BEQ      15$              ;BRANCH IF YES
4719 026446 042737 004000 177776    BIC      #BIT11,2#PSW      ;RESTORE THE SP
4720 026454 012706 001100            MOV      #STACK,SP        ;R15 SRC DOES NOT SELECT ON UPAD2
4721 026460 104335                    ERROR   335
4722 026462 000404                    BR       16$
4723 026464 042737 004000 177776    15$:   BIC      #BIT11,2#PSW      ;PDRD PS11 DOES NOT GET TO GRAC
4724 026472 104336                    ERROR   336
4725                                     ;:*****
4726                                     ;TEST GRAB E19(2 & 3)
4727 026474 012706 001100            16$:   MOV      #STACK,SP        ;RESET THE SP
4728 026500 012767 026506 152402    MOV      #14$,SLPERR
4729 026506 042737 004000 177776    14$:   BIC      #BIT11,2#PSW      ;GO BACK TO SET 0
4730 026514 005004                    CLR      R4                ;ENSURE R4 CLEAR
4731 026516 052737 004000 177776    BIS      #BIT11,2#PSW      ;GO TO SET 1
4732 026524 005204                    INC      R14              ;INCREMENT R14
4733 026526 042737 004000 177776    BIC      #BIT11,2#PSW      ;GO BACK TO SET 0
4734 026534 005704                    TST      R4                ;WAS R4 AFFECTED?
4735 026536 001401                    BEQ      17$              ;BRANCH IF NO
4736 026540 104341                    ERROR   341                ;GRAB DST SET 1 DID NOT GO LOW ON R14
4737                                     ;:*****
4738                                     ;TEST GRAB E18(2 & 3)
4739 026542 012767 026550 152340    17$:   MOV      #13$,SLPERR
4740 026550 052737 004000 177776    13$:   BIS      #BIT11,2#PSW      ;GO TO SET 1
4741 026556 005004                    CLR      R14              ;ENSURE EVEN ADDRESS IN R14
4742 026560 012404                    MOV      (R14)+,R14        ;EXECUTE A UPAD 0
4743 026562 042737 004000 177776    BIC      #BIT11,2#PSW      ;GO BACK TO SET 0
4744 026570 005704                    TST      R4                ;WAS R4 AFFECTED?
4745 026572 001401                    BEQ      TST51            ;BRANCH IF NO
    
```

H10

```
4746 026574 104342          ERROR 342          ;GRAB SRC SET 1 DID NOT GO LOW ON R14
4747
4748
4749 ;*****
4750 ;*TEST 51 REGISTER SET 1 STUCK BIT TEST
4751 ;*
4752 ;* THIS TEST ENSURES THAT ALL BITS IN GPR'S R10 THRU R15 WORK
4753 ;*****
4753 026576 000004          †TST51: SCOPE
4754 026600 012767 027012 152366      MOV #TST52,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
4755 026606 012767 027012 152454      MOV #TST52,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
4756 026614 005067 152342             CLR $TMP0 ;INITIALIZE PASS COUNT
4757 026620 052737 004000 177776      BIS #BIT11,@#PSW ;GO TO REG SET 1
4758 026626 012737          4$: MOV (PC)+,@(PC)+ ;SAVE EXPECTED VALUE
4759 026630 125252          5$: .WORD 125252
4760 026632 001164          .WORD $TMP1 ;ADDRESS OF $TEMP1
4761 026634 012700          MOV (PC)+,R10 ;PUT PATTERN IN R10
4762 026636 125252          6$: .WORD 125252
4763 026640 022700          CMP (PC)+,R10 ;IS R10 DST OK?
4764 026642 125252          7$: .WORD 125252
4765 026644 001040          BNE 1$ ;BRANCH IF NO
4766 026646 020027          CMP R10,(PC)+ ;IS R10 SRC OK?
4767 026650 125252          8$: .WORD 125252
4768 026652 001034          BNE 2$ ;BRANCH IF NO
4769 026654 010001          MOV R10,R11
4770 026656 020001          CMP R10,R11 ;IS R11 DST OK?
4771 026660 001032          BNE 1$ ;BRANCH IF NO
4772 026662 020100          CMP R11,R10 ;IS R11 SRC OK?
4773 026664 001027          BNE 2$ ;BRANCH IF NO
4774 026666 010102          MOV R11,R12
4775 026670 020002          CMP R10,R12 ;IS R12 DST OK?
4776 026672 001025          BNE 1$ ;BRANCH IF NO
4777 026674 020200          CMP R12,R10 ;IS R12 SRC OK?
4778 026676 001022          BNE 2$ ;BRANCH IF NO
4779 026700 010003          MOV R10,R13
4780 026702 020003          CMP R10,R13 ;IS R13 DST OK
4781 026704 001020          BNE 1$ ;BRANCH IF NO
4782 026706 020300          CMP R13,R10 ;IS R13 SRC OK?
4783 026710 001015          BNE 2$ ;BRANCH IF NO
4784 026712 010004          MOV R10,R14
4785 026714 020004          CMP R10,R14 ;IS R14 DST OK?
4786 026716 001013          BNE 1$ ;BRANCH IF NO
4787 026720 020400          CMP R14,R10 ;IS R14 SRC OK?
4788 026722 001010          BNE 2$ ;BRANCH IF NO
4789 026724 010005          MOV R10,R15
4790 026726 020005          CMP R10,R15 ;IS R15 DST OK
4791 026730 001006          BNE 1$ ;BRANCH IF NO
4792 026732 020500          CMP R15,R10 ;IS R15 SRC OK
4793 026734 001410          BEQ 3$ ;BRANCH IF YES
4794 026736 042737 004000 177776      BIC #BIT11,@#PSW
4795 026744 104337          2$: ERROR 337 ;BAD BITS IN GPR SET 1 SRC
4796 026746 042737 004000 177776      1$: BIC #BIT11,@#PSW
4797 026754 104340          ERROR 340 ;BAD BITS IN GPR SET 1 DST
4798 026756 005767 152200          3$: TST $TMP0 ;IS THIS FIRST PASS?
4799 026762 001013          BNE TST52 ;BRANCH IF NO
4800 026764 005267 152172          INC $TMP0 ;SET PASS COUNT
4801 026770 005167 177634          COM 5$ ;GO TO
```

4802	026774	005167	177636	COM	6\$;COMPLIMENT
4803	027000	005167	177636	COM	7\$;PATTERN
4804	027004	005167	177640	COM	8\$	
4805	027010	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.
*****

```

```

TST52: SCOPE
MOV      #TST53,NEXTTST ;SAVE ADDRESS OF NEXT TEST
MOV      #120,@#PSW+1  ;SET TO SUPER & PREVIOUS SUPER
MOV      #120,@#PSW+1  ;IS IT OK?
CMP      #120,@#PSW+1  ;BRANCH IF YES
BEQ      1$            ;SAVE EXPECTED VALUE
MOV      #50000,$TMP0  ;SAVE ERROR VALUE
MOV      @#PSW,$ERPSW  ;MASK OFF HIGH BYTE
BIC      #377,$ERPSW   ;PATTERN FAILED
ERROR    343           ;SET COMPLIMENT PATTERN
1$:      MOV      #350,@#PSW+1 ;IS IT OK?
CMP      #350,@#PSW+1  ;BRANCH IF YES
BEQ      2$            ;SAVE EXPECTED VALUE
MOV      #164000,$TMP0 ;SAVE ERROR VALUE
MOV      @#PSW,$ERPSW  ;MASK OFF HIGH BYTE
BIC      #377,$ERPSW   ;PATTERN FAILED
ERROR    344           ;CLEAR ALL BITS
2$:      CLRB     @#PSW+1  ;DID THEY ALL CLEAR?
TST      @#PSW+1
BEQ      TST53        ;BRANCH IF YES
CLR      $TMP0        ;SAVE EXPECTED VALUE
MOV      @#PSW,$ERPSW  ;SAVE ERROR VALUE
BIC      #377,$ERPSW  ;MASK OFF HIGH BYTE
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
*****

```

```

TST53: SCOPE
MOV      #TST54,NEXTTST ;SAVE ADDRESS OF NEXT TEST
;UPAD 5-SSP
MOV      #5$,$LPERR    ;SETUP ERROR LOOP
5$:      MOV      #STACK,KSP ;SETUP THE KERNAL SP
BIS      #BIT14,@#PSW  ;GO TO SUPER MODE
CLR      SSP           ;CLEAR THE SUPER SP (UPAD 5)
BIC      #BIT14,@#PSW  ;GO BACK TO KERNEL MODE
TST      KSP          ;DID THE KSP CHANGE?
BNE      1$           ;BRANCH IF NO
MOV      #STACK,KSP   ;RESTORE THE KSP
ERROR    346           ;SUPER SP DOES NOT SELECT ON UPAD 5
;UPAD 0-SSP
1$:      MOV      #6$,$LPERR ;SETUP ERROR LOOP

```



```

4858 027242 012706 001100 6$: MOV #STACK,KSP ;INITIALIZE THE SP
4859 027246 052737 040000 177776 BIS #BIT14,@#PSW ;GO TO SUPER MODE
4860 027254 012706 001776 MOV #1776,SSP ;SETUP SUPER SP
4861 027260 012606 MOV (SSP)+,SSP ;EXECUTE A UPAD 0 TYPE OPERATION
4862 027262 042737 040000 177776 BIC #BIT14,@#PSW ;GO BACK TO KERNEL
4863 027270 020627 001100 CMP KSP,#STACK ;DID KSP CHANGE?
4864 027274 001403 BEQ 2$ ;BRANCH IF NO
4865 027276 012706 001100 MOV #STACK,KSP ;RESTORE THE KSP
4866 027302 104347 ERROR 347 ;SUPER SP DOES NOT SELECTON UPAD 0

```

```

4867 ;UPAD 5-USP
4868 2$: MOV #7$,SLPERR ;SETUP ERROR LOOP
4869 027304 012767 027312 151576 7$: BIS #BIT14,@#PSW ;GO TO SUPER MODE
4870 027312 052737 040000 177776 MOV #-1,SSP ;SET THE SSP TO A NON-ZERO NUMBER
4871 027320 012706 177777 BIS #BIT15,@#PSW ;GO TO USER MODE
4872 027324 052737 100000 177776 CLR USP ;CLEAR R17
4873 027332 005006 BIC #BIT15,@#PSW ;GO BACK TO SUPER MODE
4874 027334 042737 100000 177776 TST SSP ;DID SSP CLEAR?
4875 027342 005706 BNE 3$ ;BRANCH IF NO
4876 027344 001003 CLRB @#PSW+1 ;GO BACK TO KERNEL
4877 027346 105037 177777 ERROR 350 ;USER SP DOES NOT SELECT ON UPAD 5
4878 027352 104350

```

```

4879 ;UPAD 0-USP
4880 3$: MOV #8$,SLPERR ;SETUP ERROR LOOP
4881 027354 012767 027362 151526 8$: BIS #BIT14,@#PSW ;GO TO SUPER MODE
4882 027362 052737 040000 177776 CLR SSP ;ENSURE SSP CLEAR
4883 027370 005006 BIS #BIT15,@#PSW ;GO TO USER MODE
4884 027372 052737 100000 177776 MOV #1776,USP ;SET USP TO EVEN NUMBER
4885 027400 012706 001776 MOV (USP)+,USP ;EXECUTE A UPAD 0 TYPE INSTURCTION
4886 027404 012606 BIC #BIT15,@#PSW ;GO BACK TO SUPER
4887 027406 042737 100000 177776 TST SSP ;DID SSP CHANGE?
4888 027414 005706 BEQ 4$ ;BRANCH IF NO
4889 027416 001403 CLRB @#PSW+1 ;GO BACK TO KERNEL
4890 027420 105037 177777 ERROR 351 ;USER SP DOES NOT SELECT ON UPAD 0
4891 027424 104351 4$: CLRB @#PSW+1 ;GO BACK TO KERNAL
4892 027426 105037 177777 ;CONTINUE
4893
4894
4895

```

```

4896 ;*****
4897 ;*TEST 54 SUPER AND USER SP BIT TEST
4898 ;*
4899 ;* THIS TEST ENSURES THAT THE SUPERVISOR AND USER STACK POINTERS
4900 ;* DON'T HAVE ANY STUCK BITS.
4901 ;*****

```

```

4901 027432 000004 TST54: SCOPE
4902 027434 012767 027576 151532 MOV #TST55,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
4903 027442 012767 027576 151620 MOV #TST55,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
4904 027450 005067 151510 CLR $TMP1 ;CLEAR LOOP COUNT
4905 027454 052737 040000 177776 BIS #BIT14,@#PSW ;GO TO SUPER MODE
4906 027462 012706 4$: MOV (PC)+,SP ;PUT PATTERN IN SP
4907 027464 052525 6$: .WORD 52525 ;DATA PATTERN
4908 027466 022706 CMP (PC)+,SP ;IS DST OK?
4909 027470 052525 7$: .WORD 52525 ;DATA PATTERN
4910 027472 001403 BEQ 1$ ;BRANCH IF YES
4911 027474 105037 177777 CLRB @#PSW+1 ;GO BACK TO KERNEL
4912 027500 104352 ERROR 352 ;DST FAILED
4913 027502 020627 1$: CMP SP,(PC)+ ;IS SRC OK?

```



```

4914 027504 052525      8$:      .WORD      52525      ;DATA PATTERN
4915 027506 001403      BEQ          2$          ;BRANCH IF YES
4916 027510 105037 177777      CLRB        2#PSW+1    ;GO BACK TO KERNEL
4917 027514 104353      ERROR       353        ;SRC FAILED
4918 027516 005737 177776      2$:      TST          2#PSW    ;IS BIT 15 SET?
4919 027522 100404      BMI          3$          ;BRANCH IF YES
4920 027524 052737 100000 177776      BIS        #BIT15,2#PSW ;GO TO USER MODE
4921 027532 000753      BR          4$          ;GO CHECK USER STACK POINTER
4922 027534 005767 151424      3$:      TST          $TMP1    ;IS THIS SECOND PASS?
4923 027540 001014      BNE          5$          ;BRANCH IF YES
4924 027542 005167 177716      COM         6$          ;CHANGE TEST
4925 027546 005167 177716      COM         7$          ;TO COMPLIMENT
4926 027552 005167 177726      COM         8$          ;PATTERN
4927 027556 042737 100000 177776      BIC        #BIT15,2#PSW ;GO BACK TO SUPER MODE
4928 027564 005267 151374      INC         $TMP1    ;SET PASS COUNT
4929 027570 000734      BR          4$          ;GO TEST COMPLIMENT PATTERN
4930 027572 105037 177777      5$:      CLRB        2#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.
;*****
    
```

```

4939 027576 000004      TST55:  SCOPE
4940 027600 012767 030032 151462      MOV        #TST56,NEXTTST ;SAVE ADDRESS OF NEXT TEST
;SECTION 1-POP THE KSP AND PUT IT IN THE SSP
4943 027606 012706 001100      MOV        #STACK,KSP    ;SETUP THE KERNEL SP
4944 027612 012746 177777      MOV        #-1,-(KSP)    ;PUT -1 ON THE STACK
4945 027616 005000      CLR        RO            ;CLEAR ERROR COUNT
4946 027620 052737 010000 177776      BIS        #BIT12,2#PSW  ;SET PREVIOUS MODE SUPER
4947 027626 106606      MTPD       SP            ;EXECUTE INSTRUCTION UNDER TEST
4948 027630 022706 001100      CMP        #STACK,KSP    ;DID THE KSP DST COME OUT OK?
4949 027634 001401      BEQ        2$          ;BRANCH IF YES
4950 027636 005200      INC        RO            ;SET THE ERROR COUNT
4951 027640 020627 001100      2$:      CMP        KSP,#STACK    ;DID THE KSP SRC COME OUT OK?
4952 027644 001410      BEQ        3$          ;BRANCH IF YES
4953 027646 012706 001100      MOV        #STACK,KSP    ;RESTORE THE SP
4954 027652 005700      TST        RO            ;DID DST ALSO FAIL?
4955 027654 001002      BNE        4$          ;BRANCH IF YES
4956 027656 104354      ERROR     354          ;KSP SRC WAS CHANGED
4957 027660 000431      BR         9$
4958 027662 104355      4$:      ERROR     355          ;BOTH KSP SRC AND DST CHANGED
4959 027664 000427      BR         9$
4960 027666 005700      3$:      TST        RO            ;DID KSP DST CHANGE
4961 027670 001404      BEQ        5$          ;BRANCH IF NO
4962 027672 012706 001100      MOV        #STACK,KSP    ;RESTORE THE SP
4963 027676 104356      ERROR     356          ;KSP DST WAS CHANGED
4964 027700 000421      BR         9$
4965 027702 052737 040000 177776      5$:      BIS        #BIT14,2#PSW  ;GO TO SUPER MODE
4966 027710 022706 177777      CMP        #-1,SSP      ;DID SSP LOAD OK?
4967 027714 001412      BEQ        6$          ;BRANCH IF YES
4968 027716 010667 151232      MOV        SSP,$REG0    ;SAVE THE SSP
4969 027722 005006      CLR        SSP          ;FOR LOOPING
    
```

```

4970 027724 042737 040000 177776      BIC    #BIT14,2#PSW    ;GO BACK TO KERNEL
4971 027732 012767 177777 151216      MOV    #-1,$REG1      ;SAVE EXPECTED VALUE
4972 027740 104357          ERROR   357           ;SSP DID NOT LOAD PROPERLY
4973
4974          ;SECTION 2-POP THE KSP AND PUT IT IN THE USP
4975 027742 005006      6$:    CLR    SSP           ;ENSURE SSP CLEAR FOR ITERATIONS
4976 027744 042737 040000 177776      9$:    BIC    #BIT14,2#PSW ;GO BACK TO KERNEL
4977 027752 012767 027760 151130      MOV    #7,$,SLPERR    ;SET UP ERROR LOOP
4978 027760 012706 001100      7$:    MOV    #STACK,KSP  ;SETUP THE SP
4979 027764 012746 177777      MOV    #-1,-(KSP)     ;PUT -1 ON THE KERNEL STACK
4980 027770 052737 030000 177776      BIS    #30000,2#PSW   ;SET PREVIOUS MODE USER
4981 027776 106606      MTPD   SP             ;EXECUTE INSTRUCTION UNDER TEST
4982 030000 052737 140000 177776      BIS    #140000,2#PSW  ;GO TO USER MODE
4983 030006 020627 177777      CMP    USP,#-1        ;DID USER SP GET LOADED?
4984 030012 001404      BEQ    8$             ;BRANCH IF YES
4985 030014 005006      CLR    USP           ;CLEAR USER SP
4986 030016 105037 177777      CLRB  2#PSW+1        ;GO BACK TO KERNEL
4987 030022 104360      ERROR   360           ;USER SP DID NOT LOAD ON MTP
4988 030024 005006      8$:    CLR    USP           ;ENSURE USP CLEAR, IF ITERATING
4989 030026 105037 177777      CLRB  2#PSW+1        ;GO BACK TO KERNEL
4990
4991          ;CONTINUE
4992          ;*****
4993          ;*TEST 56      MFP*DMO*DF6*PREVIOUS MODE SUPER
4994          ;*
4995          ;* THE ONLY POSSIBLE WAY THIS TEST CAN FAIL IS THAT
4996          ;* IRCC DMO (MFP+MTP) DOES NOT GO HIGH ON MFP.
4997          ;* THIS WILL ONLY HAPPEN IF THE IR DECODE ROM HAS
4998          ;* A BAD FIELD (R(MFP+MTP)).
4999          ;*****
5000 030032 000004      †ST56: SCOPE
5001 030034 012767 030160 151132      MOV    #TST57,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
5002 030042 012767 030160 151220      MOV    #TST57,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
5003
5004          ;PUSH THE SSP ONTO THE KERNEL STACK
5005 030050 012767 030114 151030      MOV    #1$,SLPADR    ;SETUP LOOP ADR
5006 030056 012767 030114 151024      MOV    #1$,SLPERR    ;SETUP ERROR LOOP
5007 030064 013767 177776 151076      MOV    2#PSW,$TMP3   ;SAVE PSW
5008 030072 032737 000020 177776      BIT    #BIT4,2#PSW   ;IS T BIT ON?
5009 030100 001405      BEQ    1$            ;BRANCH IF NO
5010 030102 012746 000340      MOV    #PR7,-(SP)    ;PUT NEW PSW ON STACK
5011 030106 012746 030114      MOV    #1$,-(SP)     ;PUT RETURN ADDR ON STACK
5012 030112 000006      RTT
5013 030114 052737 040000 177776      1$:    BIS    #BIT14,2#PSW ;GO TO SUPER MODE
5014 030122 005006      CLR    SSP           ;CLEAR THE SUPER SP
5015 030124 105037 177777      CLRB  2#PSW+1        ;GO BACK TO KERNEL
5016 030130 012706 001100      MOV    #STACK,KSP   ;SETUP THE KSP
5017 030134 012766 177777 177776      MOV    #-1,-2(KSP)  ;PUT KNOWN VALUE ON STACK
5018 030142 052737 010000 177776      BIS    #BIT12,2#PSW ;SET PREVIOUS MODE SUPER
5019 030150 106506      MFPD   SP             ;EXECUTE INSTRUCTION UNDER TEST
5020 030152 005716      TST    (KSP)         ;DID STACK GET SUPER SP?
5021 030154 001401      BEQ    TST57         ;BRANCH IF YES
5022 030156 104361      ERROR   361           ;IR DECODE ROM BAD
5023
5024          ;*****
5025          ;*TEST 57      UPAD 7 IN USER MODE
5026          ;*

```

M10

```

5026      ;*      THIS TEST ENSURES THAT A UPAD 7(OCCURS IN RTI) CAUSES THE USER
5027      ;*      STACK POINTER TO BE USED TO FETCH THE NEW PS AND PC.
5028      ;*
5029      ;*      IF PORD PS14(1) DOES NOT GET TO THE GSAM(ON GRAC)
5030      ;*      THE TEST WILL BLOW UP.
5031      ;*****
5032 030160 000004          †TST57: SCOPE
5033 030162 012767 030370 151100      MOV      #TST60,NEXTTST ;SAVE ADDRESS OF NEXT TEST
5034 030170 012706 001100              MOV      #STACK,KSP      ;SETUP THE KERNAL SP
5035 030174 052737 040000 177776      BIS      #BIT14,@#PSW    ;GO TO SUPER MODE
5036 030202 012706 001064              MOV      #1064,SSP      ;SETUP THE SSP
5037 030206 012716 030246              MOV      #1$,($SSP)     ;PUT ADDRESS OF 1$ ON SUPER STACK
5038 030212 012766 140340 000002      MOV      #140340,2(SSP) ;PUT NEW PSW ON SUPER STACK
5039 030220 052737 100000 177776      BIS      #BIT15,@#PSW    ;GO TO USER MODE
5040 030226 012706 001060              MOV      #1060,USP      ;SETUP THE USP
5041 030232 012716 030316              MOV      #2$,($USP)     ;PUT ADDRESS OF 2$ ON USER STACK
5042 030236 012766 140340 000002      MOV      #140340,2(USP) ;PUT NEW PSW ON USER STACK
5043 030244 000002              RTI                    ;EXECUTE INSTRUCTION THAT USES UPAD 7
5044      ;FAILURE, SUPER STACK POINTER WAS READ
5045 030246 022706 001070 1$:          CMP      #1070,USP      ;DID USP DST GET WRITTEN?
5046 030252 001004              BNE      3$            ;BRANCH IF NO
5047 030254 105037 177777              CLRB    @#PSW+1        ;GO BACK TO KERNEL
5048 030260 104362              ERROR   362           ;SSP WAS READ AND USP WAS WRITTEN
5049 030262 000415              BR      2$            ;
5050 030264 042737 100000 177776 3$:    BIC      #BIT15,@#PSW    ;GO TO SUPER MODE
5051 030272 022706 001070              CMP      #1070,SSP      ;WAS THE SSP WRITTEN?
5052 030276 001004              BNE      4$            ;BRANCH IF NO
5053 030300 105037 177777              CLRB    @#PSW+1        ;GO BACK TO KERNEL
5054 030304 104363              ERROR   363           ;SSP WAS READ AND WRITTEN
5055 030306 000403              BR      2$            ;
5056 030310 105037 177777 4$:          CLRB    @#PSW+1        ;GO BACK TO KERNEL
5057 030314 104364              ERROR   364           ;DON'T KNOW WHAT HAPPENED
5058      ;READ OK, NOW CHECK WRITE
5059 030316 052737 140000 177 2$:      BIS      #140000,@#PSW  ;SETUP PSW
5060 030324 022706 001064              CMP      #1064,USP      ;DID THE USP GET WRITTEN?
5061 030330 001415              BEQ      5$            ;BRANCH IF YES
5062 030332 042737 100000 177776      BIC      #BIT15,@#PSW    ;GO TO SUPER MODE
5063 030340 022706 001064              CMP      #1064,SSP      ;DID THE SSP GET WRITTEN?
5064 030344 001004              BNE      6$            ;BRANCH IF NO
5065 030346 105037 177777              CLRB    @#PSW+1        ;GO TO KERNEL MODE
5066 030352 104365              ERROR   365           ;USP WAS READ BUT SSP WAS WRITTEN
5067 030354 000403              BR      5$            ;
5068 030356 105037 177777 6$:          CLRB    @#PSW+1        ;GO TO KERNEL MODE
5069 030362 104366              ERROR   366           ;USP WAS READ BUT REG 7 WAS WRITTEN
5070 030364 105037 177777 5$:          CLRB    @#PSW+1        ;GO TO KERNEL
5071      ;CONTINUE
5072
5073
5074      ;*****
5075      ;*TEST 60      SPL*SUPERVISOR MODE
5076      ;*
5077      ;*      THIS TEST ENSURES THAT SPL DOES NOT LOAD THE PSW IN SUPER+USER MODE.
5078      ;*****
5079 030370 000004          †TST60: SCOPE
5080 030372 000237              SPL      7              ;SET CPU AT LEVEL 7
5081 030374 052737 040000 177776      BIS      #BIT14,@#PSW    ;GO TO SUPER MODE

```

N10

```

5082 030402 000230
5083 030404 042737 040000 177776
5084 030412 013767 177776 150566
5085 030420 042767 177437 150560
5086 030426 022767 000340 150552
5087 030434 001404
5088 030436 012767 000340 150516
5089 030444 104367
5090
5091
5092
5093
5094
5095
5096
5097
5098
5099
5100
5101
5102
5103
5104
5105
5106
5107
5108
5109 030446 000004
5110 030450 013767 177776 150506
5111 030456 012767 174000 150476
5112 030464 012767 031150 150576
5113 030472 005067 150510
5114
5115
5116 030476 012767 030504 150404
5117 030504 012706 001072
5118 030510 012716 030524
5119 030514 012766 177757 000002
5120 030522 000002
5121 030524 122737 000370 177777
5122 030532 001406
5123 030534 113767 177777 150445
5124 030542 105037 177777
5125 030546 104370
5126
5127
5128 030550 012767 030556 150332
5129 030556 152737 000370 177777
5130 030564 012706 001072
5131 030570 012716 030602
5132 030574 005066 000002
5133 030600 000002
5134 030602 122737 000370 177777
5135 030610 001406
5136 030612 113767 177777 150367
5137 030620 105037 177777

```

```

SPL 0 ;TRY AND CLEAR PRIORITIES
BIC #BIT14,2#PSW ;GO BACK TO KERNEL
MOV 2#PSW,$ERPSW ;SAVE PSW
BIC #1C<PR7>,$ERPSW ;MASK OFF PRIORITIES
CMP #PR7,$ERPSW ;DID SPL CLR PRIORITIES
BEQ TST61 ;BRANCH IF NO
MOV #PR7,$TMP0 ;SAVE EXPECTED VALUE
LRROR 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
* 2 174XXX 000XXX 174XXX
* 3 040XXX 134XXX 174XXX
* 4 144XXX 000XXX 030XXX
* 5 030XXX 000XXX 000XXX
*****
TST61: SCOPE
MOV 2#PSW,$TMP1 ;SAVE PSW
MOV #174000,$TMP0 ;SAVE EXPECTED VALUE OF PSW
MOV #TST62,NEXTTST ;SAVE ADDRESS OF NEXT TEST
CLR $ERPSW ;ENSURE $ERPSW CLEAR
*****
;RTI IN KERNEL MODE WITH NEW PSW<15:11>174 AND OLD PSW<15:11>000
11$: MOV #11,$SLPERR ;SETUP ERROR LOOP
MOV #1072,SP ;SETUP THE SP
MOV #1$(,SP) ;PUT ADDRESS OF 1$ ON THE STACK
MOV #177757,2(SP) ;SET ALL BITS IN NEW PSW EXCEPT T
RTI ;EXECUTE INSTRUCTION
1$: CMPB #370,2#PSW+1 ;DID NEW PSW LOAD OK?
BEQ 2$ ;BRANCH IF YES
MOVB 2#PSW+1,$ERPSW+1 ;SAVE ERROR PSW
CLRB 2#PSW+1 ;GO BACK TO KERNEL
ERROR: 370 ;A HIGH BYTE BIT DID NOT SET IN PSW
*****
;RTI IN USER MODE WITH NEW PSW<15:11>000 AND OLD PSW<15:11>174
2$: MOV #12,$SLPERR ;SETUP ERROR LOOP
BISB #370,2#PSW+1 ;SETUP THE PSW
MOV #1072,USP ;SETUP USER SP
MOV #3$(,USP) ;PUT ADDRESS OF 3$ ON STACK
CLR 2(USP) ;PUT NEW PSW ON STACK
RTI ;EXECUTE INSTRUCTION
3$: CMPB #370,2#PSW+1 ;DID PSW STAY THE SAME?
BEQ 4$ ;BRANCH IF YES
MOVB 2#PSW+1,$ERPSW+1 ;SAVE PSW
CLRB 2#PSW+1 ;GO BACK TO KERNEL

```

```

5138 030624 104371          ERROR 371          ;A HIGH BYTE BIT CLEARED IN PSW
5139          ;*****
5140          ;RTI IN SUPERVISOR MODE WITH NEW PSW<15:11>134 AND OLD PSW<15:11>040
5141          ;CHECKS PRESETS ON BITS 11, 12, 13, AND 15
5142 030626 012767 030634 150254 4$:  MOV  #13$, $LPERR      ;SETUP ERROR LOOP
5143 030634 012737 040340 177776 13$: MOV  #40340, 2#PSW    ;GO TO SUPER AND CLEAR PREVIOUS MODE AND REG BITS
5144 030642 012706 001072          MOV  #1072, $SP        ;SET THE SSP
5145 030646 012716 030662          MOV  #5$, ($SP)       ;PUT ADDRESS OF 5$ ON STACK
5146 030652 012766 134000 000002  MOV  #134000, 2($SP)  ;PUT NEW PSW ON STACK
5147 030660 000002          RTI                   ;EXECUTE INSTRUCTION
5148 030662 122737 000370 177777 5$:  CMPB #370, 2#PSW+1    ;DID ALL BITS SET?
5149 030670 001406          BEQ  10$              ;BRANCH IF YES
5150 030672 113767 177777 150307  MOVB 2#PSW+1, $ERPSW+1 ;SAVE PSW HIGH BYTE
5151 030700 105037 177777          CLRB 2#PSW+1        ;GO BACK TO KERNEL
5152 030704 104372          ERROR 372          ;BITS <15,13:11> DID NOT PRESET
5153          ;*****
5154          ;IOT IN USER MODE WITH NEW PSW<15:11>000 AND OLD PSW<15:11>144
5155 030706 012767 030714 150174 10$: MOV  #14$, $LPERR    ;SETUP ERROR LOOP
5156 030714 112737 000310 177777 14$: MOVB #310, 2#PSW+1    ;SETUP PSW
5157 030722 012737 030740 000020  MOV  #6$, 2#IOTVEC    ;PUT ADDRESS OF 6$ IN IOT VECTOR
5158 030730 012737 000340 000022  MOV  #PR7, 2#IOTVEC+2 ;PUT NEW PSW IN IOT VECTOR+2
5159 030736 000004          IOT                  ;EXECUTE INSTRUCTION
5160 030740 122737 000060 177777 6$:  CMPB #60, 2#PSW+1    ;DID NEW PSW COME UP OK?
5161 030746 001411          BEQ  7$              ;BRANCH IF YES
5162 030750 113767 177777 150231  MOVB 2#PSW+1, $ERPSW+1 ;SAVE PSW
5163 030756 012767 030000 150176  MOV  #30000, $TMP0    ;SAVE EXPECTED VALUE
5164 030764 105037 177777          CLRB 2#PSW+1        ;MAKE SURE KERNEL MODE
5165 030770 104373          ERROR 373          ;PSW HIGH BYTE WRONG
5166          ;*****
5167          ;IOT IN KERNEL MODE WITH NEW PSW<15:11>000 AND OLD PSW<15:11>030
5168 030772 012767 031000 150110 7$:  MOV  #15$, $LPERR    ;SETUP ERROR LOOP
5169 031000 112737 000060 177777 15$: MOVB #60, 2#PSW+1    ;SETUP PSW
5170 031006 012737 031024 000020  MOV  #8$, 2#IOTVEC    ;SETUP IOT VECTOR
5171 031014 012737 030340 000022  MOV  #30340, 2#IOTVEC+2 ;SETUP NEW PSW
5172 031022 000004          IOT                  ;EXECUTE INSTRUCTION
5173 031024 105737 177777          8$:  TSTB 2#PSW+1      ;IS PSW HIGH BYTE CLEAR?
5174 031030 001430          BEQ  9$              ;BRANCH IF YES
5175 031032 113767 177777 150147  MOVB 2#PSW+1, $ERPSW+1 ;SAVE PSW
5176 031040 005067 150116  CLR  $TMP0            ;SAVE EXPECTED VALUE
5177 031044 104374          ERROR 374          ;PREVIOUS MODE BITS DID NOT CLEAR
5178          ;*****
5179          ;RESET IN SUPER MODE
5180 031046 012737 074357 177776  MOV  #74357, 2#PSW    ;SETUP PSW
5181 031054 000005          RESET              ;EXECUTE INSTRUCTION
5182 031056 013767 177776 150122  MOV  2#PSW, $ERPSW    ;SAVE PSW
5183 031064 026727 150116 074340  CMP  $ERPSW, #74340   ;WAS PSW OK?
5184 031072 001412          BEQ  16$            ;BRANCH IF YES
5185 031074 012767 074340 150060  MOV  #74340, $TMP0    ;SAVE EXPECTED VALUE
5186 031102 005037 177776  CLR  2#PSW
5187 031106 104377          ERROR 377          ;RESET AFFECTED BITS IN PSW
5188 031110 000461          461
5189 031112 012737 036656 000020 9$:  MOV  #5$SCOPE, 2#IOTVEC ;RESTORE IOTVEC
5190 031120 032767 000020 150036 16$: BIT  #BIT4, $TMP1    ;WAS T BIT ON?
5191 031126 001410          BEQ  TST62          ;BRANCH IF NO
5192 031130 012706 001074          MOV  #1074, SP       ;SETUP THE STACK
5193 031134 016716 150130          MOV  NEXTTST, (SP)   ;PUT ADDRESS OF NEXT TEST ON STACK

```

```

5194 031140 012766 000360 000002
5195 031146 000002
5196
5197
5198
5199
5200
5201
5202
5203
5204
5205
5206
5207 031150 000004
5208 031152 012767 031264 150110
5209 031160 012706 001100
5210 031164 012737 031212 000004
5211 031172 052737 040000 177776
5212 031200 000000
5213
5214 031202 105037 177777
5215 031206 104377
5216 031210 000417
5217 031212 032737 000200 177766 15:
5218 031220 001010
5219 031222 013767 177766 147724
5220 031230 012767 000200 147724
5221 031236 104377
5222 031240 000420
5223 031242 005037 177766 25:
5224 031246 005737 177766
5225 031252 001401
5226 031254 104256
5227 031256 012737 036242 000004 35:
5228
5229
5230
5231
5232
5233
5234
5235
5236
5237 031264 000004
5238 031266 012767 031760 147774
5239 031274 012767 003706 147602
5240 031302 012767 031310 147576
5241 031310 012737 031360 000064 105:
5242 031316 012767 031324 147564
5243 031324 012706 001100 15:
5244 031330 000233
5245 031332 012777 000100 147602
5246 031340 012777 000015 147576
5247 031346 000001
5248
5249 031350 005077 147566

```

```

MOV #360,2(SP) ;SET T BIT ON STACK
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 BENS 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 LOCALLOCATION 15.
*
* THE CPU ERROR REGISTER BIT 7 IS ALSO TESTED HERE.
*****
TST62: SCOPE
MOV #TST63,NEXTTST ;SAVE ADDRESS OF NEXT TEST
MOV #STACK,SP ;SETUP THE SP
MOV #15,ERRVEC ;SETUP ERRVEC
BIS #BIT14,PSW ;GO TO SUPER MODE
HALT ;EXECUTE INSTRUCTION UNDER TEST
;FAILURE, NO TRAP
CLRB #PSW+1 ;GO BACK TO KERNEL
ERROR 377 ;ILLEGAL HALT DID NOT
417 ;TRAP TO 4
15: BIT #BIT7,CPUERR ;IS ERROR REG OK?
BNE 25 ;BRANCH IF YES
MOV #CPUERR,$REGO ;SAVE FOR TYP0UT
MOV #200,$TMP0 ;SAVE EXPECTED VALUE
ERROR 377 ;BIT 7 DID NOT SET
25: CLR #CPUERR ;CLEAR BIT 7
TST #CPUERR ;DID BIT 7 CLEAR?
BEQ 35 ;BRANCH IF YES
ERROR 256 ;BIT 7 DID NOT CLEAR
35: MOV #CPUSUR,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.
*****
TST63: SCOPE
MOV #TST64,NEXTTST ;SAVE ADDRESS OF NEXT TEST
MOV #101990,$ICNT ;ADJUST ITERATION COUNT
MOV #105,$LPADR ;SETUP LOOP ADR
105: MOV #25,TPVEC ;SETUP TELEPRINTER VECTOR
MOV #15,$LPERR ;SETUP ERROR LOOP
15: MOV #STACK,SP ;INITIALIZE THE SP
SPL 3
MOV #BIT6,$STPS ;SET PRINTER INTERRUPT FLAG
MOV #15,$STPB ;SEND CHARACTER
WAIT ;EXECUTE INSTRUCTION UNDER TEST
;FAILURE
CLR $STPS ;CLEAR INTERR ENABLE BIT

```

```

5250 031354 104377          ERROR 377          ;WAIT INSTRUCTION DID NOT
5251 031356 000440          440          ;WAIT FOR INTERRUPT
5252          :TEST OK, NO TRY WITH T BIT
5253 031360 012767 031402 147522 2$: MOV #3$,SLPERR ;SETUP ERROR LOOP
5254 031366 012737 031436 000014  MOV #4$,Q#TBITVEC ;SETUP T BIT VECTOR
5255 031374 012737 031460 000064  MOV #5$,Q#TPVEC ;SETUP TELEPRINTER VECTOR
5256 031402 012706 001100          3$: MOV #STACK,SP ;INITIALIZE THE SP
5257 031406 012746 000020          MOV #20,-(SP) ;PUT SP ON STACK TO TURN ON T BIT
5258 031412 012746 031434          MOV #6$,-(SP) ;PUT RETURN ADDRESS ON STACK
5259 031416 052777 000100 147516  BIS #BIT6,Q#STPS ;SET PRINTER INTERRUPT BIT
5260 031424 012777 000015 147512  MOV #15,Q#STPB ;SEND CHARACTER
5261 031432 000006          RTT          ;TURN T BIT ON
5262 031434 000001          6$: WAIT          ;EXECUTE INSTRUCTION UNDER TEST
5263          :FAILURE, T BIT CAME IN
5264 031436 005077 147500          4$: CLR Q#STPS ;CLEAR PRINTER STATUS
5265 031442 012706 001100          MOV #STACK,SP ;RESTORE THE SP
5266 031446 012737 036226 000014  MOV #SRTN,Q#TBITVEC ;RESTORE T BIT VECTOR
5267 031454 104377          ERROR 377          ;T BIT CAUSED WAIT TO
5268 031456 000441          441          ;QUIT
5269          :TEST OK, NOW TRY PIRQ
5270 031460 012737 036226 000014 5$: MOV #SRTN,Q#TBITVEC ;RESTORE T BIT VECTOR
5271 031466 012767 031510 147414  MOV #7$,SLPERR ;SETUP ERROR LOOP
5272 031474 012737 031534 000064  MOV #8$,Q#TPVEC ;SETUP TP VECTOR
5273 031502 012737 031550 000240  MOV #9$,Q#PIRQVEC ;SETUP PIRQ VECTOR
5274 031510 012706 001100          7$: MOV #STACK,SP ;SETUP THE SP
5275 031514 012737 100000 177772  MOV #BIT15,Q#PIRQ ;SET PIR LEVEL 7
5276 031522 012777 000015 147414  MOV #15,Q#STPB ;SEND CHARACTER
5277 031530 000233          SPL 3 ;LOWER CPU
5278 031532 000001          WAIT          ;EXECUTE INSTRUCTION UNDER TEST
5279          :FAILURE, PIRQ DID NOT INTERRUPT
5280 031534 005077 147402          8$: CLR Q#STPS ;CLEAR TP STATUS
5281 031540 005037 177772          CLR Q#PIRQ ;CLEAR PIR LEVEL 7
5282 031544 104377          ERROR 377          ;PIRQ DID NOT CAUSE
5283 031546 000442          442          ;INTERRUPT
5284          :TEST OK
5285 031550 005077 147366          9$: CLR Q#STPS ;CLEAR INTERRUPT FLAG
5286 031554 005037 177772          CLR Q#PIRQ ;CLEAR PIR LEVEL 7
5287          ;CONTINUE
5288
5289          ;*****
5290          ;* THE NEXT 6 TESTS USE MEMORY MANAGEMENT. IF THESE TESTS ARE NOT TO
5291          ;* BE EXECUTED, THEN SWITCH 3 SHOULD BE PUT ON.
5292          ;* IT SHOULD BE REMEMBERED THAT A FAILURE IN THESE TESTS
5293          ;* COULD BE DUE TO MEMORY MANAGEMENT.
5294          ;*****
5295
5296          ;SBTTL MEMORY MANAGEMENT SETUP
5297          ;*
5298          ;* THIS ROUTINE SETS UP THE KERNEL AND SUPERVISOR PAR'S AND
5299          ;* PDR'S TO MAP VIRTUAL ADDRESSES TO THE SAME PHYSICAL ADDRESSES.
5300
5301 031560 032737 000400 177570 MMSET: BIT #SW8,Q#SWR ;SWITCH 8 ON?
5302 031566 001027          BNE 1$ ;BRANCH IF YES
5303 031570 032737 000010 177570  BIT #BIT3,Q#SWR ;IS SWITCH 3 ON?
5304 031576 001423          BEQ 1$ ;BRANCH IF NO
5305 031600 005767 147274          TST $PASS ;IS THIS FIRST PASS?

```



```

5306 031604 001007 BNE 3$ ;BRANCH IF NO
5307 031606 032737 040000 177570 BIT #SW14,2#SWR ;IS TEST BEING LOOPED ON?
5308 031614 001012 BNE 5$ ;BRANCH IF YES
5309 031616 104400 076721 TYPE ,EM703 ;TYPE MESSAGE
5310 031622 000404 BR 6$ ;EXIT
5311 031624 032737 004000 177570 3$: BIT #SW11,2#SWR ;IS TEST ITERATING?
5312 031632 001403 BEQ 5$ ;BRANCH IF YES
5313 031634 062767 000006 147240 6$: ADD #6,STSTNM ;ADJUST TEST NUMBER
5314 031642 000167 002012 5$: JMP CACHE ;SKIP OVER MEMORY MANAGEMENT TESTS
5315 031646 012703 172340 1$: MOV #KIPARO,R3 ;GET ADDRESS OF KIPARO
5316 031652 012704 172300 MOV #KIPDRO,R4
5317 031656 005005 CLR R5
5318 031660 010300 4$: MOV R3,R0 ;GET ADDRESS OF KIPARO OR SIPARO
5319 031662 012720 000000 MOV #0,(R0)+ ;SETUP
5320 031666 012720 000200 MOV #200,(R0)+ ;THE
5321 031672 012720 000400 MOV #400,(R0)+ ;KERNEL OR SUPERVISOR
5322 031676 012720 000600 MOV #600,(R0)+ ;PAR'S
5323 031702 012720 001000 MOV #1000,(R0)+ ;TO MAP
5324 031706 012720 001200 MOV #1200,(R0)+ ;THE PROGRAM
5325 031712 012720 001400 MOV #1400,(R0)+ ;TO
5326 031716 012710 177600 MOV #177600,(R0) ;ITSELF
5327 031722 010400 MOV R4,R0 ;GET ADDRESS OF KIPDRO OR SIPDRO
5328 031724 012701 077406 MOV #77406,R1 ;GET PDR DATA
5329 031730 012702 000010 MOV #10,R2 ;SETUP SOB COUNT
5330 031734 010120 2$: MOV R1,(R0)+ ;INSTRUCTION TO SETUP PDR'S
5331 031736 077202 SOB R2,2$ ;EXECUTE EIGHT TIMES
5332 031740 005705 TST R5 ;IS SETUP COMPLETE?
5333 031742 001006 BNE TST64 ;BRANCH IF YES
5334 031744 012703 172240 MOV #SIPARO,R3 ;GET ADDRESS OF SUPER PARO
5335 031750 012704 172200 MOV #SIPDRO,R4 ;GET ADDRESS OF SUPER PDRO
5336 031754 005205 INC R5 ;SET PASS COUNT
5337 031756 000740 BR 4$ ;GO SETUP SUPER PAR'S AND PDR'S

```

```

*****
*TEST 64 MEMORY MANAGEMENT ABORT

```

```

* THIS TEST SETS UP PDR6 TO CAUSE A MEMORY MANAGEMENT ABORT.
* IF TMCC AERF(1) DOES NOT GO HIGH IT WILL LOOK LIKE THE TRAP FAILED.
*
* IF TMCC ABORT DOES NOT GO HIGH THE TEST WILL NOT TRAP AT ALL.
* IF TMCB SEGT DOES NOT GO LOW A TRAP TO 4 WILL OCCUR.
*
* IF TMCE CACHE BEND DOES NOT GO HIGH A TRAP TO 350 WILL OCCUR.

```

```

5349 031760 000004 TST64: SCOPE
5350 031762 012767 032164 147204 MOV #TST65,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
5351 031770 012767 032164 147272 MOV #TST65,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
5352 031776 012737 032112 000004 MOV #3$,2#ERRVEC ;SETUP LOCATION 4
5353 032004 012737 000340 000006 MOV #PR7,2#ERRVEC+2 ;RESTOR ERROR VEC PSW
5354 032012 012737 032152 000250 MOV #4$,2#MMVEC ;SETUP ABORT VECTOR
5355 032020 012737 032122 000010 MOV #5$,2#RESVEC ;SETUP LOCATION 10
5356 032026 012737 000340 000012 MOV #PR7,2#RESVEC+2 ;RESTORE RESVEC PSW
5357 032034 012737 032132 000240 MOV #6$,2#240 ;SETUP LOCATION 240
5358 032042 012737 032142 000350 MOV #7$,2#350 ;SETUP LOCATION 350
5359 032050 012767 032064 147032 MOV #1$, $LPERR ;SETUP ERROR LOOP
5360 032056 012737 077401 172314 MOV #77401,2#KIPDR6 ;MAKE PAGE 6 CAUSE ABORT
5361 032064 012706 001100 1$: MOV #STACK,SP ;SETUP THE SP

```


F11

```

5362 032070 012737 000001 177572 MOV #BIT0,2#MMRO ;TURN RELOCATION ON
5363 032076 012737 177777 140000 MOV #-1,2#140000 ;EXECUTE ABORT INSTRUCTION
5364 ;ABORT FAILED
5365 032104 005037 177572 CLR 2#MMRO ;TURN RELOCATION OFF
5366 032110 104375 ERROR 375 ;NO KT ABORT
5367 ;TRAPPED TO LOCATION 4
5368 032112 005037 177572 3$: CLR 2#MMRO ;TURN OFF MM
5369 032116 104377 ERROR 377 ;TRAPPED TO 4
5370 032120 000400 400
5371 032122 005037 177572 5$: CLR 2#MMRO ;TURN OFF MM
5372 032126 104377 ERROR 377 ;TRAPPED TO 10
5373 032130 000401 401
5374 032132 005037 177572 6$: CLR 2#MMRO ;TURN OFF MM
5375 032136 104377 ERROR 377 ;TRAPPED TO 240
5376 032140 000402 402
5377 032142 005037 177572 7$: CLR 2#MMRO ;TURN RELOCATION OFF
5378 032146 104377 ERROR 377
5379 032150 000416 416 ;TMCE CACHE BEND DID NOT GO HIGH
5380 032152 005037 177572 4$: CLR 2#MMRO ;TURN MM OFF
5381 032156 012737 000012 000010 MOV #12,2#RESVEC

```

;CONTINUE

```

*****
;TEST 65 MEMORY MANAGEMENT TRAP

```

```

;
; THIS TEST ENSURES THAT THE MEMORY MANAGEMENT TRAP LOGIC WORKS.
; IF TMCA HONOR SEGTF DOES NOT GO LOW OR DOES NOT GET THRU
; TO TMCB BRQ TRUE THE TRAP WILL NOT OCCUR.

```

```

;
; IF TMCB SEGT DOES NOT GO LOW BEN13 WILL FAIL TO BRK.20.
; THE FLOW WILL THEN GO TO RTI.60 WHICH MEANS AN ACKNOWLEDGE
; IS NEVER GIVEN AND THE PROCESSOR WILL HANG UP.

```

```

;
; AN INSTRUCTION IS THEN EXECUTED THAT CAUSES A MEMORY MANAGEMENT
; TRAP ON THE SOURCE OPERAND BUT NOT ON THE DESTINATION.

```

```

*****
;ST65: SCOPE

```

```

5397 032164 000004
5398 032166 012767 032330 147000 MOV #TST66,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
5399 032174 012767 032330 147066 MOV #TST66,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
5400 032202 012737 036242 000004 MOV #CPUSP0R,2#ERRVEC ;RESTORE LOCATION 4
5401 032210 012737 032260 000250 MOV #2$,2#MMVEC ;SETUP LOCATION 250
5402 032216 012767 032232 146664 MOV #3$,SLPERR ;SETUP ERROR LOOP
5403 032224 012737 077404 172314 MOV #77404,2#KIPDR6 ;SET ACF 4 IN PDR6
5404 032232 012706 001100 3$: MOV #STACK,SP ;INITIALIZE THE SP
5405 032236 012737 001001 177572 MOV #1001,2#MMRO ;TURN RELOCATION ON
5406 032244 005037 140000 CLR 2#140000 ;EXECUTE TRAP TYPE INSTRUCTION
5407 ;NO TRAP
5408 032250 005037 177572 CLR 2#MMRO ;TURN RELOCATION OFF
5409 032254 104377 ERROR 377 ;NO KT TRAP
5410 032256 000403 403
5411 ;TRAP OK, NOW TRY TRAP ON SOURCE NO TRAP ON DESTINATION
5412 032260 012737 032324 000250 2$: MOV #4$,2#MMVEC ;SETUP VECTOR
5413 032266 012767 032274 146614 MOV #5$,SLPERR ;SETUP ERROR LOOP
5414 032274 012706 001100 5$: MOV #STACK,SP ;INITIALIZE THE SP
5415 032300 012737 001001 177572 MOV #1001,2#MMRO ;INITIALIZE MMRO
5416 032306 013767 140000 146646 MOV 2#140000,$TMPD ;EXECUTE TRAP ON SOURCE
5417 ;NO TRAP

```

G11

5418 032314 005037 177572
5419 032320 104377
5420 032322 000405
5421
5422 032324 005037 177572
5423
5424
5425
5426
5427
5428
5429
5430
5431
5432
5433
5434 032330 000004
5435 032332 012767 032512 146730
5436 032340 022737 157777 177760
5437 032346 001461
5438 032350 012767 032400 146532
5439 032356 012737 032434 000004
5440 032364 013737 177760 172354
5441 032372 012737 077406 172314
5442 032400 012706 001100
5443 032404 012737 000020 172516
5444 032412 012737 000001 177572
5445 032420 005037 140100
5446
5447 032424 005037 177572
5448 032430 104377
5449 032432 000406
5450
5451 032434 005037 177572
5452 032440 022737 000040 177766
5453 032446 001412
5454 032450 013767 177766 146476
5455 032456 012767 000040 146476
5456 032464 005037 177766
5457 032470 104377
5458 032472 000410
5459 032474 005037 177766
5460 032500 005737 177766
5461 032504 001402
5462 032506 104377
5463 032510 000411
5464
5465
5466
5467
5468
5469
5470
5471
5472 032512 000004
5473 032514 012767 033010 146546

```

CLR      2#MMRO      ;TURN RELOCATION OFF
ERROR    377        ;NO TRAP ON SOURCE
405
:TRAP OK
45: CLR      2#MMRO      ;TURN RELOCATION OFF
;CONTINUE
:*****
:TEST 66      NON EXISTANT MEMORY ABORT
:
: THIS TEST ENSURES THAT A NON EXISTANT MEMORY
: REFERENCE FUNCTIONS PROPERLY.
:
: IF TMCC AERF(1) DOES NOT GO HIGH IT WILL LOOK LIKE THE ABORT FAILED.
: IF TMCC ABORT DOES NOT GO HIGH THE ABORT WILL STILL OCCUR,
: BUT THE ERROR REGISTER WILL NOT BE LOADED.
:*****
TST66: SCOPE
MOV      #TST67,NEXTTST ;SAVE ADDRESS OF NEXT TEST
CMP      #157777,2#SIZELO;2 MILLION WORDS ON SYSTEM?
BEQ      TST67          ;BRANCH IF YES
MOV      #15,$LPERR     ;SETUP ERROR LOOP
MOV      #35,2#ERRVEC  ;SETUP LOCATION 4
MOV      2#SIZELO,2#KIPAR6 ;SETUP PAR6
MOV      #77406,2#KIPDR6 ;PUT 6 IN PDR6 ACF FIELD
15: MOV      #STACK,SP   ;SETUP THE SP
MOV      #20,2#MMR3    ;SETUP FOR 22 BIT MODE
MOV      #BIT0,2#MMRO  ;TURN RELOCATION ON
CLR      2#140100     ;MAKE REFERENCE TO NEXM
;NO ABORT
CLR      2#MMRO      ;TURN RELOCATION OFF
ERROR    377        ;NO ABORT ON NEXM
406
:ABORT OK
35: CLR      2#MMRO      ;TURN RELOCATION OFF
CMP      #BITS,2#CPUERR ;IS ERROR REGISTER OK?
BEQ      45          ;BRANCH IF YES
MOV      2#CPUERR,$REGO ;SAVE ERROR REG FOR TYPEOUT
MOV      #BITS,$TMP0  ;SAVE EXPECTED VALUE
CLR      2#CPUERR    ;CLEAR ERROR REG
ERROR    377        ;NEXM BIT DID NOT SET IN CPU ERROR
410
45: CLR      2#CPUERR   ;CLEAR ERROR REG
TST      2#CPUERR    ;DID REGISTER CLEAR?
BEQ      TST67      ;BRANCH IF YES
ERROR    377        ;NEXM BIT DID NOT CLEAR
411
:*****
:TEST 67      KT BEND
:
: THIS TEST ENSURES THAT TMCE KT BEND GOES LOW ON AN ODD
: ADDRESS ERROR, SL RED, AND NEXM. THIS IS DONE BY EXECUTING
: AN INSTRUCTION FOR EACH OF THESE THREE CASES THAT ALSO
: CAUSES A KT ABORT. THE ABORT SHOULD NOT BE HONORED.
:*****
TST67: SCOPE
MOV      #TST70,NEXTTST ;SAVE ADDRESS OF NEXT TEST

```

H11

```

5474 032522 022737 157777 177760      CMP      #157777,2#SIZELO;2 MILLION WORDS ON SYSTEM?
5475 032530 001432                      BEQ      3$;BRANCH IF YES
5476 032532 012767 032570 146350      MOV      #1$, $LPERR;SETUP ERROR LOOP
5477 032540 012737 032606 000250      MOV      #2$,2#MMVEC;SETUP MEMORY VECTOR
5478 032546 012737 032616 000004      MOV      #3$,2#ERRVEC;SETUP LOCATION 4
5479 032554 052737 000007 172314      BIS      #7,2#KIPDR6;SET ACF FIELD TO 7 IN PDR6
5480 032562 012737 000060 172516      MOV      #60,2#MMR3;SETUP MMR3
5481 032570 012706 001100      1$: MOV      #STACK,SP;INITIALIZE THE SP
5482 032574 012737 000001 177572      MOV      #BIT0,2#MMR0;TURN RELOCATION ON
5483 032602 005037 140100      CLR      2#140100;MAKE A REFERENCE TO NEXM
5484                      ;FAILURE, KT ABORT CAME IN
5485 032606 005037 177572      2$: CLR      2#MMR0;TURN RELOCATION OFF
5486 032612 104377                      ERROR   377;KT ABORT OCCURRED ON NEXM
5487 032614 000412                      412
5488                      ;NEXM OK, TRY SL RED
5489 032616 005037 172516      3$: CLR      2#MMR3;GO BACK TO 18 BIT MODE
5490 032622 052737 000007 172314      BIS      #7,2#KIPDR6;MAKE KERNEL PAGE 6 NON RESIDENT
5491 032630 012737 032674 000250      MOV      #5$,2#MMVEC;SETUP MEM VECTOR
5492 032636 012737 032674 000004      MOV      #5$,2#ERRVEC;SETUP LOCATION 4
5493 032644 012767 032652 146236      MOV      #6$, $LPERR;SETUP ERROR LOOP
5494 032652 012737 000001 177572      6$: MOV      #BIT0,2#MMR0;TURN RELOCATION ON
5495 032660 012706 140336      MOV      #140336,KSP;PUT SP IN RED ZONE PAGE 6
5496 032664 012737 140000 177774      MOV      #140000,2#STKLMT;SET STACK BOUNDARY AT 24K + 400
5497 032672 005016                      CLR      (SP);EXECUT INSTRUCTION TO RED ZONE NON-RESIDENT
5498 032674 032737 100000 177572      5$: BIT      #BIT15,2#MMR0;DID KT ABORT FLAG COME ON?
5499 032702 001410                      BEQ      10$;BRANCH IF NO
5500 032704 005037 177572      CLR      2#MMR0;TURN RELOCATION OFF
5501 032710 005037 177774      CLR      2#STKLMT
5502 032714 012706 001100      MOV      #STACK,SP;RESTORE THE SP
5503 032720 104377                      ERROR   377
5504 032722 000413                      413;KT ABORT OCCURED ON SL RED
5505                      ;SL RED OK, TRY ODD ADDRESS
5506 032724 005037 177774      10$: CLR      2#STKLMT;CLEAR THE SL REG
5507 032730 012737 032770 000250      MOV      #7$,2#MMVEC;SETUP MM VECTOR
5508 032736 012737 033000 000004      MOV      #8$,2#ERRVEC;SETUP LOCATION 4
5509 032744 012767 032752 146136      MOV      #9$, $LPERR;SETUP ERROR LOOP
5510 032752 012737 000001 177572      9$: MOV      #BIT0,2#MMR0;TURN RELOCATION ON
5511 032760 012706 001100      MOV      #STACK,SP;INITIALIZE THE SP
5512 032764 005037 140001      CLR      2#140001;EXECUTE ODD ADDRESS AND KT ABORT
5513                      ;FAILURE, KT ABORT CAME IN
5514 032770 005037 177572      7$: CLR      2#MMR0;TURN RELOCATION OFF
5515 032774 104377                      ERROR   377;KT ABORT OCCURRED ON ODD ADDRESS
5516 032776 000414                      414
5517                      ;ODD ADDRESS OK
5518 033000 005037 177572      8$: CLR      2#MMR0;TURN RELOCATION OFF
5519 033004 005037 177766      CLR      2#CPUERR;ENSURE ERROR REG CLEAR
5520                      ;CONTINUE
5521                      ;*****
5522                      ;*TEST 70      SL REGISTER COMPARATOR TEST 2
5523                      ;*
5524                      ;*      THIS TEST IS THE SAME AS TEST 153 EXCEPT IT TESTS THE ADDRESSES
5525                      ;*      ON EVERY PAGE. THIS IS DONE BY MAPPING I/O PAGE ADDRESSES TO
5526                      ;*      MEMORY IN KERNEL MODE. THIS MAKES THE I/O PAGE INACCESSABLE
5527                      ;*      IN KERNEL MODE SO AN IOT INSTRUCTION IS USED TO RETURN TO
5528                      ;*      SUPERVISOR MODE WHEN THE I/O PAGE IS NEEDED.
5529                      ;*****

```

```

5530 033010 000004 TST70: SCOPE
5531 033012 012767 033552 146250 MOV #TST71,NEXTTST ;SAVE ADDRESS OF NEXT TEST
5532 033020 005067 146142 CLR $TMP2 ;CLEAR BUFFER OVERFLOW FLAG
5533 033024 012767 003706 146052 MOV #1D1990,$ICNT ;SETUP ITERATION COUNT
5534 033032 012767 033040 146046 MOV #23,$SLPADR ;SETUP LOOP ADDRESS
5535 033040 012767 033524 146126 23$: MOV #22,$$ESCAPE ;SETUP ESCAPE
5536 033046 012737 033520 000020 MOV #12,$@#IOTVEC ;SETUP THE IOT VECTOR
5537 033054 012737 040340 000022 MOV #40340,@#IOTVEC+2 ;SETUP THE IOT VEC PSW
5538 033062 012737 077406 172314 MOV #77406,@#KIPDR6 ;ENSURE PDR6 OK
5539 033070 012737 001400 172354 MOV #1400,@#KIPAR6 ;ENSURE PAR6 OK
5540 033076 012737 033222 000004 MOV #1,$@#ERRVEC ;SETUP ERROR VECTOR
5541 033104 012737 040340 000006 MOV #40340,@#ERRVEC+2 ;SETUP ERRVEC PSW
5542 033112 012737 001600 172356 MOV #1600,@#KIPAR7 ;MAP KERNEL PAGE 7 TO 28K
5543 033120 052737 040000 177776 BIS #BIT14,@#PSW ;GO TO SUPER MODE
5544 033126 012737 000001 177572 MOV #BIT0,@#MMRO ;TURN RELOCATION ON
5545 033134 012706 001100 MOV #STACK,SSP ;INITIALIZE THE SSP
5546 033140 105037 177777 CLR @#PSW+1 ;GO BACK TO KERNEL
5547 033144 012703 000400 MOV #400,R3 ;SET SOB COUNT FOR SL REGISTER
5548 033150 012700 120000 MOV #120000,R0 ;INITIALIZE ERROR DATA POINTER
5549 033154 005060 000002 CLR 2(R0) ;INITIALIZE ERROR DATA BUFFER
5550 033160 012701 000340 MOV #340,R1 ;INITIALIZE YELLOW ZONE ADDRESS
5551 033164 012702 000400 2$: MOV #400,R2 ;SET SOB COUNT FOR SP
5552 033170 012705 000340 MOV #340,R5 ;INITIALIZE SECONDARY STORAGE FOR SP
5553 033174 010506 3$: MOV R5,K$P ;SET THE SSP
5554 033176 011616 MOV (K$P),(K$P) ;EXECUTE TEST INSTRUCTION
5555 ;NO TRAP. DETERMINE IF THIS IS CORRECT.
5556 033200 020601 CMP K$P,R1 ;IS ADDRESS > YELL ZONE BOUNDARY?
5557 033202 101075 BHI 5$ ;BRANCH IF YES
5558 033204 001403 BEQ 6$ ;BRANCH IF ADDRESS = YELL ZONE BOUNDARY
5559 ;NO TRAP ADDRESS IS LESS THAN YELLOW ZONE BOUNDARY
5560 033206 052710 000010 BIS #BIT3,(R0) ;SET ERROR TYPE IN DATA BUFFER
5561 033212 000443 BR 10$ ;GO RECORD DATA
5562 ;NO TRAP EQUALS YELLOW ZONE BOUNDARY
5563 033214 052710 000012 6$: BIS #12,(R0) ;SET ERROR TYPE IN DATA BUFFER
5564 033220 000440 BR 10$ ;GO RECORD DATA
5565 ;
5566 ;GOT A TRAP. NOW DETERMINE IF IT IS CORRECT.
5567 ; NOW IN SUPER MODE
5568 033222 032737 000004 177766 1$: BIT #BIT2,@#CPUERR ;WAS IT A RED ZONE?
5569 033230 001406 BEQ 8$ ;BRANCH IF NO
5570 033232 020105 CMP R1,R5 ;IS ADDRESS < YELL ZONE BOUNDARY?
5571 033234 101060 BHI 5$ ;BRANCH IF YES
5572 033236 001431 BEQ 10$ ;BRANCH IF ADDRESS = YELL ZONE BOUNDARY
5573 ;RED ZONE TRAP ON LEGAL ADDRESS
5574 033240 052710 000002 BIS #BIT1,(R0) ;SET ERROR TYPE IN DATA BUFFER
5575 033244 000426 BR 10$ ;GO RECORD DATA
5576 ;NOT A RED ZONE. IS IT A YELLOW ZONE?
5577 033246 032737 000010 177766 8$: BIT #BIT3,@#CPUERR ;IS THIS A YELLOW ZONE TRAP?
5578 033254 001011 BNE 31$ ;BRANCH IF YES
5579 033256 012737 177600 172356 MOV #177600,@#KIPAR7 ;REMAP KERNEL I/O PAGE
5580 033264 005037 177777 CLR @#PSW+1 ;GO BACK TO KERNEL
5581 033270 005037 177572 CLR @#MMRO ;TURN RELOCATION OFF
5582 033274 000167 002742 JMP CPUSPUR
5583 033300 020105 31$: CMP R1,R5 ;IS ADDRESS = YELL ZONE BOUNDARY?
5584 033302 001435 BEQ 5$ ;BRANCH IF YES
5585 033304 101003 BHI 9$ ;BRANCH IF ADDRESS IS < YELL ZONE BOUNDARY
    
```

```

5586 ;YELLOW ZONE TRAP ON LEGAL ADDRESS
5587 033306 052710 000006 BIS #6,(R0) ;SET ERROR TYPE IN DATA BUFFER
5588 033312 000403 BR 10$ ;GO RECORD DATA
5589 ;YELLOW ZONE TRAP ON RED ZONE ADDRESS
5590 033314 052710 000004 9$: BIS #BIT2,(R0) ;SET ERROR TYPE IN DATA BUFFER
5591 033320 000400 BR 10$ ;GO RECORD DATA
5592 ;
5593 ;RECORD ERROR DATA
5594 033322 000004 10$: IOT ;GO TO SUPER
5595 033324 032737 001000 177570 BIT #SW9,@#SWR ;IS LOOP ON ERROR ENABLED?
5596 033332 001405 BEQ 20$ ;BRANCH IF NO
5597 033334 012706 001100 MOV #STACK,SSP ;INITIALIZE THE SSP
5598 033340 105037 177777 CLRB @#PSW+1 ;GO BACK TO KERNEL
5599 033344 000713 BR 3$ ;LOOP
5600 033346 005767 145614 20$: TST $TMP2 ;HAS BUFFER OVERFLOWED?
5601 033352 001011 BNE 5$ ;BRANCH IF YES
5602 033354 005200 INC R0 ;SET POINTER TO HIGH BYTE
5603 033356 113720 177775 MOVB @#STKLM+1,(R0)+ ;SAVE ERROR STACK LIMIT
5604 033362 010520 MOV R5,(R0)+ ;SAVE ERROR SP
5605 033364 020027 157774 CMP R0,#157774 ;HAS BUFFER REACHED PAGE 7?
5606 033370 001002 BNE 5$ ;BRANCH IF NO
5607 033372 005267 145570 INC $TMP2 ;SET BUFFER OVERFLOW FLAG
5608 ;
5609 ;CONTINUE TEST
5610 033376 062705 000400 5$: ADD #400,R5 ;GO TO NEXT STACK ADDRESS
5611 033402 000004 21$: IOT ;GO TO SUPERVISOR MODE
5612 033404 012706 001100 MOV #STACK,SSP ;RESET THE SSP
5613 033410 005037 177766 CLR @#CPUERR ;CLEAR ERROR REGISTER
5614 033414 105037 177777 CLRB @#PSW+1 ;GO BACK TO KERNEL
5615 033420 005302 DEC R2 ;REPLACES A
5616 033422 001264 BNE 3$ ;SOB
5617 033424 000004 IOT ;GO TO SUPERVISOR MODE
5618 033426 062701 000400 16$: ADD #400,R1 ;SET NEXT YELLOW ZONE ADDRESS
5619 033432 062737 000400 177774 ADD #400,@#STKLMT ;GO TO NEXT SL ADDRESS
5620 033440 105037 177777 CLRB @#PSW+1 ;GO BACK TO KERNEL
5621 033444 005303 DEC R3 ;THIS REPLACES
5622 033446 001246 BNE 2$ ;A SOB
5623 ;
5624 ;DONE WITH TEST. WAS THERE AN ERROR?
5625 033450 000004 IOT ;GO TO SUPERVISOR MODE
5626 033452 012737 177600 172356 MOV #177600,@#KIPAR7 ;RESTORE KERNEL I/O PAGE
5627 033460 105037 177777 CLRB @#PSW+1 ;GO TO KERNEL MODE
5628 033464 005037 177774 18$: CLR @#STKLMT ;RESET THE SL REG
5629 033470 012706 001100 MOV #STACK,SP ;AND SP
5630 033474 012737 036242 000004 MOV #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5631 033502 005737 120002 TST @#120002 ;WAS THERE AN ERROR?
5632 033506 001406 BEQ 22$ ;BRANCH IF NO
5633 033510 010067 145440 MOV R0,$REGO ;SAVE ERROR DATA POINTER
5634 033514 104263 ERROR 263 ;STACK LIMIT COMPARATORS FAILED
5635 033516 000402 BR 22$
5636 ;IOT ROUTINE
5637 033520 000176 000000 12$: JMP @($SP) ;RETURN IN SUPER MODE
5638 ;
5639 ;TEST FINISHED, CLEAN UP VECTORS
5640 033524 005037 177572 22$: CLR @#MMRO ;TURN RELOCATION OFF
5641 033530 012737 036656 000020 MOV #$$SCOPE,@#IOTVEC ;RESTORE IOT VECTOR
    
```

K11

```

5642 033536 012737 000340 000022
5643 033544 012737 000340 000006
5644
5645
5646
5647
5648
5649
5650
5651
5652
5653
5654
5655
5656
5657
5658 033552 000004
5659 033554 012767 033746 145506
5660 033562 012737 033634 000004
5661 033570 012737 000340 000036
5662 033576 042737 000007 172314
5663 033604 012767 033612 145276
5664 033612 005037 000002 2$:
5665 033616 012737 000001 177572
5666 033624 012706 150004
5667 033630 000236
5668 033632 104400
5669 033634 005037 177572 1$:
5670 033640 012706 001100
5671 033644 022737 000310 000002
5672 033652 001402
5673 033654 104377
5674 033656 000421
5675
5676
5677
5678
5679
5680
5681
5682
5683
5684
5685 033660 032737 000400 177570
5686 033666 001027
5687 033670 032737 000004 177570
5688 033676 001423
5689 033700 005767 145174
5690 033704 001007
5691 033706 032737 040000 177570
5692 033714 001012
5693 033716 104400 076751
5694 033722 000404
5695 033724 032737 004000 177570 1$:
5696 033732 001403
5697 033734 062767 000004 145140 2$:

```

```

MOV #PR7,2#IOTVEC+2
MOV #PR7,2#ERRVEC+2
:CONTINUE
*****
*TEST 71 PS RESTORE
*
* THIS TEST ENSURES THAT BEN6 WORKS ON A PS RESTORE.
* THIS IS DONE BY SETTING THE STACK TO A NON RESIDENT PAGE
* AND DOING A TRAP INSTRUCTION. WHEN THE PROCESSOR TRYS TO
* PUSH THE OLD PSW ON THE STACK A KT ABORT WILL OCCUR.
* SINCE IT WAS A KERNEL R6 OPERATION THIS WILL CAUSE BEN13
* TO GO TO STATE SER.00 WHICH WILL PUSH THE PSW AND PC INTO
* LOCATIONS 2 AND 0, AND THEN TRAP TO LOCATION 4.
* THE PSW IN LOCATION 2 SHOULD BE THE PSW BEFORE THE
* TRAP INSTRUCTION AND NOT THE PSW IN THE TRAP VECTOR.
*****
TST71: SCOPE
MOV #TST72,NEXTTST ;SAVE ADDRESS OF NEXT TEST
MOV #1$,2#ERRVEC ;SETUP ERROR VECTOR
MOV #PR7,2#TRAPVEC+2 ;ENSURE PR7 IN LOCATION 36
BIC #7,2#KIPDR6 ;MAKE PAGE 6 NON-RESIDENT
MOV #2$,2#SLPERR ;SETUP ERROR LOOP
2$: CLR #2 ;ENSURE LOCATION 2 CLEAR
MOV #BIT0,2#MMRO ;TURN RELOCATION ON
MOV #150004,SP ;SETUP THE SP
SPL 6 ;SET CPU PRIORITY AT 6
TRAP ;EXECUTE TRAP INSTRUCTION
1$: CLR 2#MMRO ;TURN RELOCATION OFF
MOV #STACK,SP ;RETSORE SP
CMP #310,2#2 ;DID CORRECT PSW GET STACKED?
BEQ CACHE ;BRANCH IF YES
ERROR 377 ;BEN6 FAILED ON PS RESTORE
421
;
*****
* THE NEXT 4 TESTS USE CERTAIN FUNCTIONS IN THE CACHE.
* IF THESE TESTS ARE NOT TO BE EXECUTED, THEN SWITCH 2 SHOULD
* BE PUT ON.
*
* IT SHOULD BE REMEMBERED THAT A FAILURE IN THESE
* TESTS COULD BE DUE TO THE CACHE.
*****
CACHE: BIT #SW8,2#SWR ;IS SWITCH 8 ON?
BNE TST72 ;BRANCH IF YES
BIT #SW2,2#SWR ;IS SWITCH 2 ON?
BEQ TST72 ;BRANCH IF NO
TST $PASS ;IS THIS FIRST PASS?
BNE 1$ ;BRANCH IF NO
BIT #SW14,2#SWR ;IS TEST BEING LOOPED ON?
BNE 3$ ;BRANCH IF YES
TYPE EM704 ;TYPE MESSAGE
BR 2$ ;EXIT
1$: BIT #SW11,2#SWR ;IS TEST ITERATING?
BEQ 3$ ;BRANCH IF YES
2$: ADD #4,$STSTM ;ADJUST TEST NUMBER

```

```

5698 033742 000167 001534 3$: JMP TST76 ;SKIP CACHE TESTS
5699 ;*****
5700 ;*TEST 72 PARITY ERROR ABORT
5701 ;*
5702 ;* THIS TEST ENSURES THAT A CACHE PARITY ERROR FLAG CAUSES AN ABORT.
5703 ;* THIS IS DONE BY FORCING A PARITY ERROR ON AN EVEN WORD.
5704 ;*****
5705 033746 000004 TST72: SCOPE
5706 033750 012767 034166 145312 MOV #TST73,NEXTTST ;SAVE ADDRESS OF NEXT TEST
5707 033756 012767 034140 145210 MOV #2$, $ESCAPE ;SETUP ESCAPE ADDRESS
5708 033764 012737 000014 177746 MOV #14, @#CONTRL ;ENSURE MISSES TO BOTH GROUPS
5709 033772 012767 034040 145110 MOV #7$, $LPERR ;SETUP ERROR LOOP
5710 034000 012737 034140 000114 MOV #2$, @#CACHVEC ;SETUP CACHE VECTOR
5711 034006 012737 034066 000004 MOV #3$, @#ERRVEC ;SETUP LOCATION 4
5712 034014 012737 034104 000014 MOV #4$, @#14 ;SETUP LOCATION 14
5713 034022 012737 034122 000104 MOV #5$, @#104 ;SETUP LOCATION 104
5714 034030 012704 170000 MOV #170000, R4 ;PUT MAINTENANCE DATA IN R4
5715 034034 012702 177750 MOV #MAINT, R2 ;PUT ADDRESS OF MAIN REG IN R2
5716 034040 012706 001100 7$: MOV #STACK, SP ;INITIALIZE THE SP
5717 034044 000401 BR 1$ ;GO TO NEXT INSTRUCTION
5718 034046 LOC= ;THIS IS
5719 034044 LOC=-3&LOC ;USED TO MAKE
5720 034050 LOC=LOC+4 ;1$ FALL ON
5721 034050 .=LOC ;AND EVEN WORD
5722 034050 000240 1$: NOP ;USED TO MAKE BAD PARITY INSTR ON EVEN WORD
5723 034052 010412 MOV R4, (R2) ;SET BITS IN MAINT REG
5724 034054 005701 TST R1 ;EXECUTE INSTR TO CAUSE PE ABORT
5725 ;FAILURE, NO ABORT
5726 034056 005012 CLR (R2) ;CLEAR MAINT REG
5727 034060 000240 NOP
5728 034062 104377 ERROR 377 ;NO PE ABORT
5729 034064 000422 422
5730 ;FAILURE, ABORTED TO WRONG VECTOR
5731 034066 005012 3$: CLR (R2) ;ENSURE MAINT REG CLEAR
5732 034070 000240 NOP
5733 034072 012737 177777 177744 MOV #-1, @#MEMERR
5734 034100 104377 ERROR 377 ;ABORTED TO LOCATION 4
5735 034102 000423 423
5736 034104 005012 4$: CLR (R2) ;ENSURE MAINT REG CLEAR
5737 034106 000240 NOP
5738 034110 012737 177777 177744 MOV #-1, @#MEMERR
5739 034116 104377 ERROR 377 ;ABORTED TO 14
5740 034120 000424 424
5741 034122 005012 5$: CLR (R2) ;ENSURE MAINT REG CLEAR
5742 034124 000240 NOP
5743 034126 012737 177777 177744 MOV #-1, @#MEMERR
5744 034134 104377 ERROR 377 ;ABORTED TO 104
5745 034136 000425 425
5746 ;TEST OK
5747 034140 005012 2$: CLR (R2) ;ENSURE MAINT REG CLEAR
5748 034142 000240 NOP
5749 034144 012737 177777 177744 MOV #-1, @#MEMERR ;CLEAR MEMORY ERROR REG
5750 034152 012737 036242 000004 MOV #CPUSPUR, @#ERRVEC ;RESET LOCATION 4
5751 034160 012737 036226 000014 MOV #SRTRN, @#TBITVEC ;RESTORE T BIT VECTOR
5752 ;CONTINUE
5753 ;*****

```



```

5754
5755
5756
5757
5758
5759
5760
5761
5762 034166 000004
5763 034170 012767 034334 145072
5764 034176 012767 034306 144770
5765 034204 012767 034236 144676
5766 034212 012737 034270 000004
5767 034220 012737 034306 000114
5768 034226 012704 170000
5769 034232 012702 177750
5770 034236 012706 001100
5771 034242 000402
5772 034244
5773 034244
5774 034250
5775 034250
5776 034250 000240
5777 034252 010412
5778 034254 000240
5779 034256 005701
5780
5781 034260 005012
5782 034262 000240
5783 034264 104377
5784 034266 000426
5785
5786 034270 005012
5787 034272 000240
5788 034274 012737 177777 177744
5789 034302 104377
5790 034304 000427
5791
5792 034306 005012
5793 034310 000240
5794 034312 012737 177777 177744
5795 034320 012737 036242 000004
5796 034326 012737 036446 000250
5797
5798
5799
5800
5801
5802
5803
5804
5805
5806
5807 034334 000004
5808 034336 012767 035166 144724
5809 034344 032737 000400 177570

```

```

;*TEST 73      PARITY ERROR TRAP
;*
;* THIS TEST ENSURES THAT A PARITY TRAP FUNCTIONS PROPERLY.
;* THIS IS DONE BY MAKING THE ODD WORD HAVE BAD PARITY.
;* IF THE TRAP DOESN'T OCCUR THEN THE PROBLEM IS ON TMCA.
;* IF A TRAP OCCURS TO THE WRONG VECTOR THE PROBLEM COULD BE
;* ON TMCA OR UCB.
;*****
†ST73:  SCOPE
        MOV      #TST74,NEXTTST ;SAVE ADDRESS OF NEXT TEST
        MOV      #3$, $ESCAPE   ;SETUP ESCAPE ADDRESS
        MOV      #1$, $LPERR    ;SETUP ERROR LOOP
        MOV      #2$, @#ERRVEC  ;SETUP THE ERROR VECTOR
        MOV      #3$, @#CACHVEC ;SETUP THE CACHE VECTOR
        MOV      #170000, R4    ;PUT MAINT DATA IN R4
        MOV      #MAINT, R2    ;PUT MAINT REG ADDR IN R2
1$:     MOV      #STACK, SP    ;INITIALIZE THE SP
        BR       4$           ;GO TO NEXT INSTRUCTION
        LOC=     ;THIS IS USED
        LOC=-3&LOC ;TO MAKE
        LOC=LOC+4 ;1$ FALL ON
        .=LOC    ;AN EVEN WORD
4$:     NOP                ;GOOD PARITY ON EVEN WORD
        MOV      R4, (R2)    ;SET BITS IN MAINT REG
        NOP
        TST      R1
;FAILURE, NO TRAP
        CLR      (R2)        ;ENSURE MAINT REG CLEAR
        NOP
        ERROR   377         ;NO PE TRAP
        426
;FAILURE, TRAPPED TO WRONG VECTOR
2$:     CLR      (R2)        ;ENSURE MAINT REG CLEAR
        NOP
        MOV      #-1, @#MEMERR ;CLEAR MEM ERROR REG
        ERROR   377         ;PE TRAP, TRAPPED TO
        427                ;WRONG VECTOR
;TEST OK
3$:     CLR      (R2)        ;ENSURE MAINT REG CLEAR
        NOP
        MOV      #-1, @#MEMERR ;CLEAR MEM ERROR REG
        MOV      #CPUSPUR, @#ERRVEC ;RESTORE LOCATION 4
        MOV      #CACHSPU, @#MMVEC ;RESTORE MEM VEC
        ;CONTINUE
;*****
;*TEST 74      MEM MGT AND PE TRAP PRIORITY ARBITRATION
;*
;* THIS TEST ENSURES THAT THE ARBITRATION LOGIC WORKS FOR MEMORY
;* MANAGEMENT AND PARITY ERROR TRAPS.
;*
;* NOTE: BOTH SWITCHES 3 AND 2 MUST BE DOWN TO EXECUTE THIS TEST AND
;* THE NEXT TEST.
;*****
†ST74:  SCOPE
        MOV      #TST75,NEXTTST ;SAVE ADDRESS OF NEXT TEST
        BIT      #SW8, @#SWR    ;SWITCH 8 ON?

```



```

5810 034352 001010          BNE      1$          ;BRANCH IF YES
5811 034354 032737 000010 177570  BIT      #BIT3,2#SWR ;IS SWITCH 3 ON?
5812 034362 001404          BEQ      1$          ;BRANCH IF NO
5813 034364 005267 144512    2$:      INC      $TSTNM
5814 034370 000167 001106    JMP      TST76      ;GO TO NEXT TEST
5815 034374 012737 001400 172354 1$:      MOV      #1400,2#KIPAR6 ;RESTORE PAR6
5816 034402 112737 000004 172314  MOVB     #4,2#KIPDR6 ;SETUP PAGE 6 TO TRAP ON ALL ACCESSES
5817 034410 012704 170000    MOV      #170000,R4 ;PUT MAINT REG DATA IN R4
5818 034414 012702 177750    MOV      #MAINT,R2  ;PUT ADDRESS OF MAINT REG IN R2
5819
5820 ;:*****
;PIR6 DISABLED BY MGMT
5821 034420 012737 040000 140000  MOV      #BIT14,2#140000 ;PUT PIR6 ENABLE BIT IN PAGE 6
5822 034426 012737 034500 000240  MOV      #3$,2#PIRQVEC ;SETUP PIRQ VECTOR
5823 034434 012737 000340 000252  MOV      #PR7,2#MMVEC+2 ;SET UP MMVEC PSW
5824 034442 012737 034514 000250  MOV      #4$,2#MMVEC   ;SETUP MEM MGMT VECTOR
5825 034450 012767 034456 144432  MOV      #5$, $LPERR  ;SETUP ERROR LOOP
5826 034456 012706 001100    5$:      MOV      #STACK,SP   ;INITIALIZE THE SP
5827 034462 012737 001001 177572  MOV      #1001,2#MMRO ;TURN RELOCATION ON
5828 034470 000235          SPL      5           ;SET PROCESSOR AT LEVEL 5
5829 034472 013737 140000 177772  MOV      2#140000,2#PIRQ ;SET PIR6 AND MEM MGT TRAP
5830 ;FAILURE, PIR6 CAME THRU
5831 034500 005037 177572    3$:      CLR      2#MMRO    ;TURN RELOCATION OFF
5832 034504 005037 177772    CLR      2#PIRQ    ;CLEAR PIR6
5833 034510 104377          ERROR   377       ;PIR6 CAME IN ON
5834 034512 000430          430          ;MGMT TRAP
5835
5836 ;:*****
5837 ;PIR3 DISABLED BY MGMT
5838 034514 005037 177572    4$:      CLR      2#MMRO    ;TURN RELOCATION OFF
5839 034520 005037 177772    CLR      2#PIRQ    ;CLEAR PIR LEVEL 6
5840 034524 012737 034576 000240  MOV      #6$,2#PIRQVEC ;SETUP PIRQ VECTOR
5841 034532 012737 034612 000250  MOV      #7$,2#MMVEC   ;SETUP MEM MGMT VECTOR
5842 034540 012767 034554 144342  MOV      #8$, $LPERR  ;SETUP ERROR LOOP
5843 034546 012737 004000 140000  MOV      #BIT11,2#140000 ;PUT PIR3 ENABLE BIT IN PAGE 6
5844 034554 012706 001100    8$:      MOV      #STACK,SP   ;INITIALIZE THE SP
5845 034560 012737 001001 177572  MOV      #1001,2#MMRO ;TURN ON RELOCATION
5846 034566 000232          SPL      2         ;LOWER CPU TO LEVEL 2
5847 034570 013737 140000 177772  MOV      2#140000,2#PIRQ ;SET PIR3 & MGMT
5848 ;FAILURE, PIR3 CAME THRU
5849 034576 005037 177572    6$:      CLR      2#MMRO    ;TURN OFF RELOCATION
5850 034602 005037 177772    CLR      2#PIRQ    ;CLEAR PIR3
5851 034606 104377          ERROR   377       ;PIR3 CAME IN ON
5852 034610 000431          431          ;MEM MGT TRAP
5853
5854 ;:*****
5855 ;STACK LIMIT YELLOW DISABLED BY PARITY ERROR
5856 034612 005037 177572    7$:      CLR      2#MMRO    ;TURN RELOCATION OFF
5857 034616 005037 177772    CLR      2#PIRQ    ;CLEAR PIR LEVEL 3
5858 034622 012767 034662 144260  MOV      #9$, $LPERR  ;SETUP ERROR LOOP
5859 034630 012737 034710 000004  MOV      #10$,2#ERRVEC ;SETUP THE ERROR VECTOR
5860 034636 012737 034710 000114  MOV      #10$,2#CACHEVEC ;SETUP CACHEVEC
5861 034644 012737 000240 000116  MOV      #PR5,2#CACHEVEC+2 ;PUT PRIORITY 5 IN CACHE VECTOR PSW
5862 034652 012704 170000    MOV      #170000,R4 ;PUT MAINT REG DATA IN R4
5863 034656 012702 177750    MOV      #MAINT,R2  ;PUT ADDRESS OF MAINT ON R2
5864 034662 005037 000370    9$:      CLR      2#370     ;ENSURE LOCATION 370 CLEAR
5865 034666 012706 000376    MOV      #376,SP   ;SETUP THE SP TO YELLOW ZONE

```

```

5866 034672 000402 BR 115 ;GO TO 125
5867 034674 LOC=. ;THIS MPKES
5868 034674 LOC=-3&LOC ;THE NEXT INSTRUCTION
5869 034700 LOC=LOC+4 ;FALL ON
5870 034700 .=LOC ;AN EVEN WORD
5871 034700 010412 115: MOV R4,(R2) ;SET MAINT REG
5872 034702 000240 NOP ;ODD WORD GOOD PARITY
5873 034704 005216 INC (SP) ;CAUSE YEL ZONE (GOOD PARITY)
5874 034706 005701 TST R1 ;ODD WORD BAD PARITY
5875 ;SHOULD TAKE PE TRAP THEN YEL ZONE TRAP
5876 034710 005012 105: CLR (R2) ;CLEAR MAINTENANCE REGISTER
5877 034712 000240 NOP
5878 034714 022737 000240 000370 CMP #PR5,#370 ;DID CACHVEC PSW GET STACKER?
5879 034722 001404 BEQ 125 ;BRANCH IF YES
5880 034724 012706 001100 MOV #STACK,SP ;RESTORE THE SP
5881 034730 104377 ERROR 377 ;YEL ZONE CAME THRU ON PE TRAP
5882 034732 000432 432
5883 ;*****
5884 ;MEMORY MANAGEMENT TRAP DISABLED BY PARITY TRAP
5885 034734 012767 035002 144146 125: MOV #135,$LPERR ;SETUP ERROR LOOP
5886 034742 012737 035030 000250 MOV #155,#MMVEC ;SETUP MEM MGT VECTOR
5887 034750 012737 035030 000114 MOV #155,#CACHVEC ;SETUP CACHVEC
5888 034756 012737 000340 000116 MOV #PR7,#CACHVEC+2 ;RESTORE EACH VEC PSW
5889 034764 012704 170000 MOV #170000,R4 ;PUT MAINT DATA IN R4
5890 034770 012702 177750 MOV #MAINT,R2 ;PUT ADDRESS OF MAINT REG IN R2
5891 034774 112737 000004 172314 MOV #4,#KIPDR6 ;ENSURE PAGE 6 TRAPS
5892 035002 012706 001100 135: MOV #STACK,SP ;INITIALIZE THE SP
5893 035006 012737 001001 177572 MOV #1001,#MMRO ;TURN RELOCATION ON
5894 035014 000401 BR 165
5895 035016 LOC=.
5896 035014 LOC=-3&LOC
5897 035020 LOC=LOC+4
5898 035020 .=LOC
5899 035020 010412 165: MOV R4,(R2) ;SET MAINT REG (PARITY GOOD)
5900 035022 000240 NOP ;ODD WORD PARITY GOOD
5901 035024 005237 140402 INC #140402 ;INC HAS GOOD PARITY BUT ADDRESS
5902 ;HAS BAD PARITY. CAUSES MM TRAP
5903 ;AND PE TRAP
5904 ;TEST OK
5905 035030 005012 155: CLR (R2) ;CLEAR MAINT REG
5906 035032 000240 NOP
5907 035034 005037 177572 CLR #MMRO ;TURN RELOCATION OFF
5908 035040 026627 000002 000340 CMP 2(SP),#PR7 ;DID PE TRAP OCCUR FIRST?
5909 035046 001402 BEQ 145 ;BRANCH IF YES
5910 035050 104377 ERROR 377 ;MEM MGT TRAP CAME
5911 035052 000433 433 ;THRU ON PARITY ERROR
5912 035054 012737 036242 000004 145: MOV #CPUSPU,#ERRVEC ;RESTORE LOCATION 4
5913 035062 012737 036446 000114 MOV #CACHSPU,#CACHVEC ;RESTORE LOCATION 114
5914 035070 012737 177777 177744 MOV #-1,#MEMERR ;CLEAR MEM ERROR REG
5915 035076 005037 177766 CLR #CPUERR ;ENSURE CPUERROR CLEAR
5916 ;CONTINUE
5917 ;
5918 ;*****
5919 ;* THE NEXT TEST USES THE MAPPING BOX AND THE CACHE TO
5920 ;* GENERATE A PARITY ERROR ON THE UNIBUS. SWITCHES 3, 2 AND
5921 ;* 1 MUST BE OFF TO EXECUTE THIS TEST.

```

```

5922 ;*****
5923 ;
5924 035102 032737 000400 177570 BIT #SW8,#SWR ;IS SWITCH 8 ON?
5925 035110 001026 BNE TST75 ;BRANCH IF YES
5926 035112 032737 000016 177570 BIT #16,#SWR ;ARE SWITCHES 3 AND 2 AND 1 OFF?
5927 035120 001422 BEQ TST75 ;BRANCH IF YES
5928 035122 005767 143752 TST $PASS ;IS THIS PASS1?
5929 035126 001007 BNE 17$ ;BRANCH IF NO
5930 035130 032737 040000 177570 BIT #SW14,#SWR ;IS TEST BEING LOOPED ON?
5931 035136 001011 BNE 18$ ;BRANCH IF YES
5932 035140 104400 076777 TYPE EM706 ;TYPE MESSAGE
5933 035144 000404 BR 19$ ;EXIT
5934 035146 032737 004000 177570 17$: BIT #SW11,#SWR ;IS TEST BEING ITERATED?
5935 035154 001402 BEQ 18$ ;BRANCH IF YES
5936 035156 005267 143720 19$: INC $STNM
5937 035162 000167 000314 18$: JMP TST76 ;SKIP THIS TEST
5938 ;*****
5939 ;TEST 75 UNIBUS PARITY ERROR
5940 ;
5941 ; THIS TEST MAKES A REFERENCE TO MEMORY THRU THE MAPPING
5942 ; BOX THAT WILL CAUSE A PARITY ERROR. IF THE ABORT DOESN'T
5943 ; HAPPEN THEN THE PROBLEM IS ON UCB.
5944 ;
5945 ; NOTE: MAP REGISTER 0 AND 1 ARE NOT USED INCASE THE PROGRAM
5946 ; IS RUNNING UNDER ACT11.
5947 ;*****
5948 035166 000004 †TST75: SCOPE
5949 035170 012767 035502 143776 MOV #TST76,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
5950 035176 012767 035502 144064 MOV #TST76,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
5951 035204 012737 077406 172314 MOV #77406,#KIPDR6 ;SETUP PDR6
5952 035212 012737 000060 172516 MOV #60,#MMR3 ;SETUP MMR3
5953 035220 012706 001100 MOV #STACK,SP ;INITIALIZE THE SP
5954 035224 012700 170210 MOV #MAPL2,R0 ;GET ADDRESS OF MAP REG 2
5955 035230 012701 000036 MOV #36,R1 ;SETUP SOB COUNT
5956 035234 012737 035246 000004 MOV #5$,#ERRVEC ;SETUP ERROR VECTOR
5957 035242 005720 8$: TST (R0)+ ;SEE IF MAP REG IS ENABLED
5958 035244 000417 BR 6$ ;BRANCH IF YES
5959 035246 062700 000002 5$: ADD #2,R0 ;ADJUST R0 TO NEXT REGISTER
5960 035252 077105 SOB R1,8$ ;TEST NEXT REGISTER
5961 035254 012706 001100 7$: MOV #STACK,SP ;RESTORE THE SP
5962 035260 005767 143614 TST $PASS ;FIRST PASS?
5963 035264 001106 BNE TST76 ;BRANCH IF NO
5964 035266 032737 040000 177570 BIT #SW14,#SWR ;IS TEST BEING LOOPED ON?
5965 035274 001102 BNE TST76 ;BRANCH IF YES
5966 035276 104400 077446 TYPE EM724 ;TYPE MESSAGE
5967 035302 000477 BR †TST76 ;GO TO NEXT TEST
5968 035304 005010 6$: CLR (R0) ;ENSURE MAP REG HIGH CLEAR
5969 035306 162700 000002 SUB #2,R0 ;GET ADDR OF MAP REG LOW
5970 035312 012710 140000 MOV #140000,(R0) ;PUT ADDR OF PAGE 6 IN MAP REG
5971 035316 072027 000005 ASH #5,R0 ;ADJUST ADDR FOR PAR6
5972 035322 052700 170000 BIS #170000,R0 ;SET UNIBUS ADDR BITS
5973 035326 010037 172354 MOV R0,#KIPAR6 ;PUT IN PAGE 6 PAR
5974 035332 012737 005701 140000 MOV #5701,#140000 ;PUT WORD WITH PAD PARITY IN 140000
5975 035340 012704 170000 MOV #170000,R4 ;PUT MAINT REG DATA IN R4
5976 035344 012702 177750 MOV #MAINT,R2 ;PUT ADDRESS OF MAINT REG IN R2
5977 035350 012767 035372 143532 MOV #1$, $LPERR ;SETUP ERROR LOOP

```

```

5978 035356 012737 035436 000000      MOV      #4$,2#0      ;SETUP LOCATION ZERO
5979 035364 012737 035460 000114      MOV      #2$,2#CACHVEC ;SETUP CACH VECTOR
5980 035372 012706 001100      MOV      #STACK,SP    ;INITIALIZE THE SP
5981 035376 052737 000001 177572 1$:  BIS      #BIT0,2#MMR0  ;TURN RELOCATION ON
5982 035404 000401      BR       3$          ;GO TO TEST
5983      035406      LOC=      ;
5984      035404      LOC=-38LOC
5985      035410      LOC=LOC+4
5986      035410      .=LOC
5987 035410 010412      3$:  MOV      R4,(R2)    ;SET BITS IN MAINT REG
5988 035412 000240      NOP      ;GOOD PARITY ON ODD WORD
5989 035414 005037 140000      CLR      2#140000    ;EXECUTE A DATIP THRU THE
;MAP THAT CAUSES A PE
5990
5991      ;FAILURE, NO ABORT
5992 035420 005012      CLR      (R2)        ;CLEAR MAINT REG
5993 035422 000240      NOP
5994 035424 005037 177572      CLR      2#MMR0      ;TURN RELOCATION OFF
5995 035430 104377      ERROR   377         ;NO UNIBUS PE ABORT
5996 035432 000436      436
5997 035434 000411      BR       2$
5998      ;TRAPPED TO WRONG VECTOR
5999 035436 005012      4$:  CLR      (R2)        ;ENSURE MAINT REG CLEAR
6000 035440 000240      NOP
6001 035442 005037 177572      CLR      2#MMR0      ;TURN OFF RELOCATION
6002 035446 012737 177777 000004      MOV      #-1,2#ERRVEC ;CLEAR ERROR REGISTER
6003 035454 104377      ERROR   377         ;TRAPPED TO ZERO
6004 035456 000437      437
6005      ;TEST OK
6006 035460 005012      2$:  CLR      (R2)        ;ENSURE MAINT REG CLEAR
6007 035462 000240      NOP
6008 035464 005037 177572      CLR      2#MMR0      ;TURN RELOCATION OFF
6009 035470 012737 177777 177744      MOV      #-1,2#MEMERR ;CLEAR ERROR REG
6010 035476 005037 172516      CLR      2#MMR3     ;ENSURE MAP TURNED OFF
6011
6012      ;*****
6013      ;TEST 76 OPERATOR INTERVENTION TEST
6014      ;
6015      ; THIS TEST ENSURES THAT THE RESET AND WAIT FLOWS PUT RO
6016      ; AND THE PC IN THE LIGHTS. THE TEST IS ONLY EXECUTED ON
6017      ; PASS 1 AND CAN BE DISABLED ALTOGETHER WITH SWITCH 0.
6018      ;
6019      ; THE RESET LOOP IS STOPED BY CHANGING THE POSITION OF
6020      ; SWITCH 7.
6021      ;
6022      ; THE WAIT IS EXITED BY TYPING A CHARACTER
6023      ; ON THE TERMINAL.
6024      ;*****
6024 035502 000004      †ST76: SCOPE
6025 035504 012767 035742 143556      MOV      #SEOP,NEXTTST ;SAVE ADDRESS OF SEOP
6026 035512 005767 143362      TST      $PASS       ;IS THIS FIRST PASS?
6027 035516 001006      BNE     1$          ;BRANCH IF NO
6028 035520 032737 000001 177570      BIT      #SW0,2#SWR   ;IS SWITCH 0 ON?
6029 035526 001403      BEQ     2$          ;BRANCH IF NO
6030 035530 104400 077167      TYPE    EM717       ;TYPE MESSAGE(SKIPPING TEST)
6031 035534 000502      1$:  BR       $EOP      ;EXIT
6032 035536 005737 000042      2$:  TST      2#42     ;IS MONITOR ACT11?
6033 035542 001077      BNE     $EOP        ;BRANCH IF YES

```

```

6034 035544 012700 166667      MOV      #166667,RO      ;SETUP RO
6035 035550 104400 077212      TYPE     ,EM720        ;TYPE MESSAGE
6036 035554 032737 000200 177570  BIT      #SW7,#SWR      ;IS SWITCH 7 ON?
6037 035562 001416          BEQ      3$            ;BRANCH IF NO
6038 035564 012767 035606 143370  MOV      #4$+2,$TMP0    ;SAVE PC
6039 035572 016746 143364      MOV      $TMP0,-(SP)   ;SAVE $TMP0 FOR TYPEOUT
6040                                ;TYPE ADDRESS
6041 035576 104402          TYPOC     ;GO TYPE--OCTAL ASCII(ALL DIGITS)
6042 035600 104400 077351      TYPE     ,EM721        ;TYPE MESSAGE
6043 035604 000005          RESET    ;EXECUTE INSTRUCTION
6044 035606 032737 000200 177570  BIT      #SW7,#SWR      ;HAS SWITCH 7 CHANGED?
6045 035614 001416          BEQ      5$            ;BRANCH IF YES
6046 035616 000772          BR       4$            ;LOOP
6047 035620 012767 035642 143334  MOV      #6$+2,$TMP0    ;SAVE PC
6048 035626 016746 143330      MOV      $TMP0,-(SP)   ;SAVE $TMP0 FOR TYPEOUT
6049                                ;TYPE ADDRESS
6050 035632 104402          TYPOC     ;GO TYPE--OCTAL ASCII(ALL DIGITS)
6051 035634 104400 077351      TYPE     ,EM721        ;TYPE MESSAGE
6052 035640 000005          RESET    ;EXECUTE INSTRUCTION
6053 035642 032737 000200 177570  BIT      #SW7,#SWR      ;HAS SWITCH 7 CHANGED?
6054 035650 001773          BEQ      6$            ;BRANCH IF NO
6055                                ;
6056                                ;WAIT TEST
6057 035652 104400 077212      5$:     TYPE     ,EM720        ;TYPE MESSAGE
6058 035656 012767 035722 143276  MOV      #7$,$TMP0    ;SAVE PC
6059 035664 016746 143272      MOV      $TMP0,-(SP)   ;SAVE $TMP0 FOR TYPEOUT
6060                                ;TYPE ADDRESS
6061 035670 104402          TYPOC     ;GO TYPE--OCTAL ASCII(ALL DIGITS)
6062 035672 104400 077407      TYPE     ,EM722        ;TYPE MESSAGE
6063 035676 005077 143236      CLR      @TKB          ;CLEAR KEYBOARD BUFFER
6064 035702 012737 035722 000060  MOV      #7$,@TKVEC    ;SETUP KEYBOARD VECTOR
6065 035710 052777 000100 143220  BIS      #BIT6,@TKS    ;SET INTERRUPT ENABLE BIT
6066 035716 000233          SPL      3          ;LOWER PRIORITY FOR INTERRUPT
6067 035720 000001          WAIT    ;EXECUTE INSTRUCTION
6068 035722 012737 040224 000060  7$:     MOV      #QUIT,@TKVEC ;RESTORE TKVECTOR
6069 035730 005077 143204          CLR      @TKB          ;ENSURE BUFFER CLEAR
6070 035734 152777 000100 143174  BISB    #BIT6,@TKS    ;SET INTERRUPT ENABLE FLAG
6071                                ;CONTINUE
6072                                ;.MCALL .SEOP
6073                                ;*****
6074                                ;
6075                                ;.SBTTL END OF PASS ROUTINE
6076                                ;
6077                                ;*INCREMENT THE PASS NUMBER ($PASS)
6078                                ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
6079                                ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
6080                                ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
6081                                ;*IF SW12=1 INHIBIT TRACE TRAP
6082                                ;*IF THERES A MONITOR GO TO IT
6083                                ;*IF THERE ISN'T JUMP TO LOOP
6084                                ;
6085                                ;SEOP:
6086                                ;
6087                                ;SCOPE
6088                                ;CLR      $STNM      ;;ZERO THE TEST NUMBER
6089                                ;CLR      $TIMES     ;;ZERO THE NUMBER OF ITERATIONS
6090                                ;INC      $PASS      ;;INCREMENT THE PASS NUMBER

```

6090	035760	042767	100000	143112	BIC	#100000,\$PASS	::DON'T ALLOW A NEG. NUMBER
6091	035766	005327			DEC	(PC)+	::LOOP?
6092	035770	000001			\$EOPCT:	.WORD 1	
6093	035772	003076			BGT	\$DOAGN	::YES
6094	035774	012737			MOV	(PC)+,2(PC)+	::RESTORE COUNTER
6095	035776	000001			\$ENDCT:	.WORD 1	
6096	036000	035770			\$EOPCT		
6097	036002	104400	036010		TYPE	65\$::TYPE ASCIZ STRING
6098	036006	000407			BR	64\$::GET OVER THE ASCIZ
6099					::65\$:	.ASCIZ <12><15>/END PASS #/	
6100	036026				64\$:		
6101	036026	016746	143046		MOV	\$PASS,-(SP)	::SAVE \$PASS FOR TYPEOUT
6102							::TYPE PASS NUMBER
6103	036032	104410			TYPDS		::GO TYPE--DECIMAL ASCII WITH SIGN
6104	036034	104400	036042		TYPE	67\$::TYPE ASCIZ STRING
6105	036040	000421			BR	66\$::GET OVER THE ASCIZ
6106					::67\$:	.ASCIZ / TOTAL ERRORS SINCE LAST REPORT /	
6107	036104				66\$:		
6108	036104	016746	143002		MOV	\$ERTTL,-(SP)	::SAVE \$ERTTL FOR TYPEOUT
6109							::TOTAL NUMBER OF ERRORS
6110	036110	104410			TYPDS		::GO TYPE--DECIMAL ASCII WITH SIGN
6111	036112	104400	001203		TYPE	\$SCLF	::TYPE CARRIAGE RETURN, LINE FEED
6112	036116	005067	142770		CLR	\$ERTTL	::CLEAR ERROR TOTAL
6113	036122	013700	000042		\$GET42:	MOV 2#42,RO	::GET MONITOR ADDRESS
6114	036126	001420			BEQ	\$DOAGN	::BRANCH IF NO MONITOR
6115	036130	005046			CLR	-(SP)	::INSURE THE "T" BIT IS CLEAR
6116	036132	012746	036140		MOV	#SCLR.T,-(SP)	::SETUP FOR AN RTI OR RTT
6117	036136	000433			BR	\$RTTN	::GO DO AN RTI OR RTT TO LOAD THE PSW
6118							::WITH A CLEARED "T" BIT
6119	036140				\$CLR.T:		
6120	036140	012703	000001		MOV	#1,R3	::MAKE R3=1
6121	036144	004767	002112		JSR	PC,CHAINQ	::GO RESTORE MONITOR
6122	036150	013700	000042		MOV	2#42,RO	::INSURE RO CONTAINS THE MONITORS
6123	036154	001405			BEQ	\$DOAGN	::RETURN ADDRESS
6124	036156	000005			RESET		::CLEAR THE WORLD
6125	036160	004710			\$ENDAD:	JSR PC,(RO)	::GO TO MONITOR
6126	036162	000240			NOP		::SAVE ROOM
6127	036164	000240			NOP		::FOR
6128	036166	000240			NOP		::ACT11
6129	036170				\$DOAGN:		
6130	036170	013746	177776		MOV	2#PS,-(SP)	::PUT THE PS ON THE STACK AND
6131	036174	042716	000020		BIC	#20,(SP)	::CLEAR THE "T" BIT
6132	036200	032737	010000	177570	BIT	#BIT12,2#SWR	::RUN WITH TRACE TRAP?
6133	036206	001005			BNE	1\$::BR IF NO
6134	036210	005167	000020		COM	\$TBIT	::IS IT TIME FOR TRACE TRAP
6135	036214	100402			BMI	1\$::BR IF NO
6136	036216	052716	000020		BIS	#20,(SP)	::SET TRACE TRAP
6137	036222	012746	036230		1\$:	MOV #SLOOP,-(SP)	::JUMP TO START OF TEST
6138	036226	000002			\$RTTN:	RTI	::RETURN--THIS IS CHANGED TO
6139							::AN "RTT" IF "RTT" IS A LEGAL
6140							::INSTRUCTION
6141	036230				\$LOOP:		
6142	036230	000137	005336		JMP	2#LOOP	::RETURN
6143	036234	000000			\$TBIT:	.WORD 0	::"T" BIT STATE INDICATOR
6144	036236	377	377	000	\$ENULL:	.BYTE -1,-1,0	::NULL CHARACTER STRING
6145	036242				.EVEN		

```

6146                                     ;*****
6147 ;SBTTL SPURIOUS ERROR HANDLER
6148 ;* THIS ROUTINE IS ENTERED BY AN UNEXPECTED TRAP TO 4 OR 114.
6149 ;* IF SWITCH 13 IS OFF, AN ERROR MESSAGE, THE ERROR REGISTER,
6150 ;* THE ERROR PC, AND THE TEST NUMBER ARE TYPED OUT.
6151 ;* IF SWITCH 13 IS ON, ONLY THE ERROR MESSAGE WILL BE TYPED.
6152 ;*****
6153 036242 011667 142650 CPUSPUR:MOV (SP), $ERRPC ;SAVE THE ERROR PC
6154 036246 012706 001100 MOV #STACK, SP ;RESTORE THE SP
6155 036252 104400 036260 TYPE ,65$ ;TYPE ASCIZ STRING
6156 036256 000414 BR ,64$ ;GET OVER THE ASCIZ
6157 ;:65$: .ASCIZ /UNEXPECTED TRAP TO 4/<15><12>
6158 036310 ;64$:
6159 036310 032737 020000 177570 BIT #BIT13, @SWR ;IS INHIBIT ERROR TYPEOUT ON?
6160 036316 001045 BNE ,1$ ;BRANCH IF YES
6161 036320 104400 036326 TYPE ,67$ ;TYPE ASCIZ STRING
6162 036324 000417 BR ,66$ ;GET OVER THE ASCIZ
6163 ;:67$: .ASCIZ /ERRORPC TSTNUM CPUERR REG/<15><12>
6164 036364 ;66$:
6165 036364 013767 177766 142570 MOV @CPUERR, $TMPD ;GET CPU ERROR REG
6166 036372 116767 142504 142610 MOVB $TSTNM, $$TSTNM ;GET TEST NUMBER
6167 036400 016746 142512 MOV $ERRPC, -(SP) ;SAVE $ERRPC FOR TYPEOUT
6168 ;TYPE ERROR PC
6169 036404 104402 TYPDC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
6170 036406 104400 040220 TYPE TWOSP
6171 036412 016746 142572 MOV $$TSTNM, -(SP) ;SAVE $$TSTNM FOR TYPEOUT
6172 ;TYPE TEST NUMBER
6173 036416 104402 TYPDC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
6174 036420 104400 040220 TYPE TWOSP
6175 036424 016746 142532 MOV $TMPD, -(SP) ;SAVE $TMPD FOR TYPEOUT
6176 ;TYPE ERROR REGISTER
6177 036430 104402 TYPDC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
6178 036432 005037 177766 1$: CLR @CPUERR ;CLEAR CPU ERROR REG
6179 036436 016767 142626 142530 MOV NEXTTST, $ESCAPE ;SETUP ESCAPE ADDRESS
6180 036444 104264 ERROR 264 ;MAKE THE ERROR CALL TO SYSMAC
6181 036446 011667 142444 CACHSPU:MOV (SP), $ERRPC ;SAVE ERROR PC
6182 036452 012706 001100 MOV #STACK, SP ;RESTORE STACK
6183 036456 104400 036464 TYPE ,65$ ;TYPE ASCIZ STRING
6184 036462 000415 BR ,64$ ;GET OVER THE ASCIZ
6185 ;:65$: .ASCIZ /UNEXPECTED TRAP TO 114/<15><12>
6186 036516 ;64$:
6187 036516 032737 020000 177570 BIT #BIT13, @SWR ;IS SWITCH 13 ON?
6188 036524 001045 BNE ,1$ ;BRANCH IF YES
6189 036526 104400 036534 TYPE ,67$ ;TYPE ASCIZ STRING
6190 036532 000417 BR ,66$ ;GET OVER THE ASCIZ
6191 ;:67$: .ASCIZ /ERRORPC TSTNUM MEMERR REG/<15><12>
6192 036572 ;66$:
6193 036572 013767 177744 142362 MOV @MEMERR, $TMPD ;SAVE MEMORY ERROR REG
6194 036600 116767 142276 142402 MOVB $TSTNM, $$TSTNM ;SAVE TEST NUMBER
6195 036606 016746 142304 MOV $ERRPC, -(SP) ;SAVE $ERRPC FOR TYPEOUT
6196 ;TYPE ERROR PC
6197 036612 104402 TYPDC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
6198 036614 104400 040220 TYPE TWOSP
6199 036620 016746 142364 MOV $$TSTNM, -(SP) ;SAVE $$TSTNM FOR TYPEOUT
6200 ;TYPE TEST NUMBER
6201 036624 104402 TYPDC ;GO TYPE--OCTAL ASCII(ALL DIGITS)

```



```

6202 036626 104400 040220          TYPE      TWOSP
6203 036632 016746 142324          MOV      $TMPD,-(SP)      ;;SAVE $TMPD FOR TYPEOUT
6204                                ;;TYPE MEM ERROR REG
6205 036636 104402                                ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
6206 036640 013737 177744 177744 1$:  MOV      @#MEMERR,@#MEMERR  ;CLEAR MEMERR REG.
6207 036646 016767 142416 142320  MOV      NEXTTST,$ESCAPE ;SETUP ESCAPE ADDRESS
6208 036654 104264          ERROR      264
6209                                .MCALL   .$TYPDEC,.$STRAP,.$SPOWER,.$STYPOCT
6210                                ;*****
6211
6212                                .SBTTL  SCOPE HANDLER ROUTINE
6213
6214                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
6215                                ;*AND LOAD THE TEST NUMBER($STNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
6216                                ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
6217                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
6218                                ;*SW14=1      LOOP ON TEST
6219                                ;*SW11=1      INHIBIT ITERATIONS
6220                                ;*SW09=1      LOOP ON ERROR
6221                                ;*SW08=1      LOOP ON TEST IN SWR<7:0>
6222                                ;*CALL
6223                                ;*      SCOPE          ;;SCOPE=IOT
6224
6225                                $SCOPE:
6226 036656 006137 177570          ROL      @#SWR          ;;LOOP ON PRESENT TEST?
6227 036662 100511          BMI      $OVER          ;;YES IF SW14=1
6228                                ;*****START OF CODE FOR THE XOR TESTER*****
6229 036664 000416          $XTSTR: BR      6$          ;; IF RUNNING ON THE "XOR" TESTER CHANGE
6230                                ;; THIS INSTRUCTION TO A "NOP" (NOP=240)
6231 036666 013746 000004          MOV      @#ERRVEC, -(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
6232 036672 012737 036712 000004  MOV      #5,@#ERRVEC    ;;SET FOR TIMEOUT
6233 036700 005737 177060          TST      @#177060      ;;TIME OUT ON XOR?
6234 036704 012637 000004          MOV      (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
6235 036710 000463          BR      $SVLAD        ;;GO TO THE NEXT TEST
6236 036712 022626          5$:  CMP      (SP)+,(SP)+ ;;CLEAR THE STACK AFTER A TIME OUT
6237 036714 012637 000004          MOV      (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
6238 036720 000423          BR      7$          ;;LOOP ON THE PRESENT TEST
6239 036722          6$:;*****END OF CODE FOR THE XOR TESTER*****
6240 036722 032737 000400 177570  BIT      #BIT08,@#SWR   ;;LOOP ON SPEC. TEST?
6241 036730 001404          BEQ      2$          ;;BR IF NO
6242 036732 123767 177570 142142  CMPB     @#SWR,$STNM   ;;ON THE RIGHT TEST?  SWR<7:0>
6243 036740 001462          BEQ      $OVER       ;;BR IF YES
6244 036742 105767 142135          2$:  TSTB     $ERFLG      ;;HAS AN ERROR OCCURRED?
6245 036746 001421          BEQ      3$          ;;BR IF NO
6246 036750 126767 142141 142125  CMPB     $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
6247 036756 101015          BHI      3$          ;;BR IF NO
6248 036760 032737 001000 177570  BIT      #BIT09,@#SWR   ;;LOOP ON ERROR?
6249 036766 001404          BEQ      4$          ;;BR IF NO
6250 036770 016767 142114 142110  7$:  MOV      $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
6251 036776 000443          BR      $OVER
6252 037000 105067 142077          4$:  CLRB     $ERFLG      ;;ZERO THE ERROR FLAG
6253 037004 005067 142162          CLR      $TIMES      ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
6254 037010 000415          BR      1$          ;;ESCAPE TO THE NEXT TEST
6255 037012 032737 004000 177570  3$:  BIT      #BIT11,@#SWR  ;;INHIBIT ITERATIONS?
6256 037020 001011          BNE     1$          ;;BR IF YES
6257 037022 005767 142052          TST     $PASS        ;;IF FIRST PASS OF PROGRAM

```



```

6258 037026 001406          BEQ      1$          ;; INHIBIT ITERATIONS
6259 037030 005267 142050    INC      $ICNT      ;; INCREMENT ITERATION COUNT
6260 037034 026767 142132 142042  CMP     $TIMES,$ICNT ;; CHECK THE NUMBER OF ITERATIONS MADE
6261 037042 002021          BGE     $OVER      ;; BR IF MORE ITERATION REQUIRED
6262 037044 012767 000001 142032 1$:  MOV     #1,$ICNT    ;; REINITIALIZE THE ITERATION COUNTER
6263 037052 016767 000044 142112  MOV     $MXCNT,$TIMES ;; SET NUMBER OF ITERATIONS TO DO
6264 037060 105267 142016    $SVLAD: INCB    $STNM    ;; COUNT TEST NUMBERS
6265 037064 011667 142016    MOV     (SP),$LPADR  ;; SAVE SCOPE LOOP ADDRESS
6266 037070 011667 142014    MOV     (SP),$LPERR  ;; SAVE ERROR LOOP ADDRESS
6267 037074 005067 142074    CLR     $ESCAPE     ;; CLEAR THE ESCAPE FROM ERROR ADDRESS
6268 037100 112767 000001 142007  MOV     #1,$ERMAX   ;; ONLY ALLOW ONE(1) ERROR ON NEXT TEST
6269 037106 016737 141770 177570 $OVER: MOV     $STNM,$#DISPLAY ;; DISPLAY TEST NUMBER
6270 037114 016716 141766    MOV     $LPADR,(SP) ;; FUDGE RETURN ADDRESS
6271 037120 000002          RTI             ;; FIXES PS
6272 037122 003720    $MXCNT: 2000.    ;; MAX. NUMBER OF ITERATIONS
6273 ;*****
6274 ;*****
6275 .SBTTL  ERROR HANDLER ROUTINE
6276
6277 ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
6278 ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
6279 ;*AND GO TO ETYPDM ON ERROR
6280 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
6281 ;*SW15=1      HALT ON ERROR
6282 ;*           HALT CAN OCCUR BEFORE AND AFTER THE ERROR TYPEOUT
6283 ;*SW13=1      INHIBIT ERROR TYPEOUTS
6284 ;*SW10=1     BELL ON ERROR
6285 ;*SW09=1     LOOP ON ERROR
6286 ;*CALL
6287 ;*          ERROR  N      ;;ERROR=EMT AND N=ERROR ITEM NUMBER
6288
6289 037124          $ERROR:
6290 037124 116767 141752 142056  MOV     $STNM,$$STNM ;; GET TEST NUMBER FOR TYPE OUT
6291 037132 105267 141745    7$:  INCB    $ERFLG    ;; SET THE ERROR FLAG
6292 037136 001775          BEQ      7$          ;; DON'T LET THE FLAG GO TO ZERO
6293 037140 016737 141736 177570  MOV     $STNM,$#DISPLAY ;; DISPLAY TEST NUMBER AND ERROR FLAG
6294 037146 005737 177570    TST     $#SWR      ;; HALT ON ERROR = 1?
6295 037152 100001          BPL     8$          ;; BRANCH IF NO
6296 037154 000000          HALT          ;; YES--HALT
6297 037156 032737 002000 177570  8$:  BIT     #BIT10,$#SWR ;; BELL ON ERROR?
6298 037164 001402          BEQ      1$          ;; NO - SKIP
6299 037166 104400 001176    TYPE    $BELL      ;; RING BELL
6300 037172 005267 141714    1$:  INC     $ERTTL    ;; COUNT THE NUMBER OF ERRORS
6301 037176 011667 141714    MOV     (SP),$ERRPC ;; GET ADDRESS OF ERROR INSTRUCTION
6302 037202 162767 000002 141706  SUB     #2,$ERRPC
6303 037210 117767 141702 141676  MOV     $#ERRPC,$ITEMB ;; STRIP AND SAVE THE ERROR ITEM CODE
6304 037216 032737 020000 177570  BIT     #BIT13,$#SWR ;; SKIP TYPEOUT IF SET
6305 037224 001004          BNE     2$          ;; SKIP TYPEOUTS
6306 037226 004767 000056    JSR     PC,ETYPDM  ;; GO TO USER ERROR ROUTINE
6307 037232 104400 001203    TYPE    $CRLF
6308 037236 005737 177570    2$:  TST     $#SWR      ;; HALT ON ERROR
6309 037242 100001          BPL     9$          ;; SKIP IF CONTINUE
6310 037244 000000          HALT          ;; HALT ON ERROR!
6311 037246 022767 036160 140566  9$:  CMP     $#ENDAD,42 ;; ACT-11?
6312 037254 001001          BNE     3$          ;; BRANCH IF NO
6313 037256 000000          HALT          ;; YES

```

```

6314 037260 032737 001000 177570 3$: BIT #BIT09,2#SWR ;;LOOP ON ERROR SWITCH SET?
6315 037266 001402 BEQ 4$ ;;BR IF NO
6316 037270 016716 141614 MOV $LPERR,(SP) ;;FUDGE RETURN FOR LOOPING
6317 037274 005767 141674 4$: TST $ESCAPE ;;CHECK FOR AN ESCAPE ADDRESS
6318 037300 001402 BEQ 5$ ;;BR IF NONE
6319 037302 016716 141666 MOV $ESCAPE,(SP) ;;FUDGE RETURN ADDRESS FOR ESCAPE
6320 037306 5$:
6321 037306 000002 RTI ;;RETURN
6322 ;:*****
6323 ;SBTTL ERROR MESSAGE TIMEOUT ROUTINE
6324 ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
6325 ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
6326 ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
6327 ;:*****
6328 037310 026727 141674 000042 ETYPDM: CMP $STSTNM,#42 ;IS THIS TEST 42?
6329 037316 001461 BEQ TYP5LE ;BRANCH IF YES
6330 037320 026727 141664 000070 CMP $STSTNM,#70 ;IS THIS TEST 70?
6331 037326 001455 BEQ TYP5LE ;BRANCH IF YES
6332 037330 104400 001203 TYPE $SCRLF ;; "CARRIAGE RETURN" & "LINE FEED"
6333 037334 005000 CLR R0 ;; PICKUP THE ITEM INDEX
6334 037336 153700 001114 BISB 2#$ITEMB,R0
6335 037342 126727 141546 000377 CMPB $ITEMB,#377 ;IS MESSAGE NUMBER OVER 377
6336 037350 001003 BNE 1$ ;BRANCH IF NO
6337 037352 016600 000002 MOV 2(SP),R0 ;GET RETURN PC FROM STACK
6338 037356 014000 MOV -(R0),R0 ;GET MESSAGE NUMBER
6339 037360 005300 1$: DEC R0 ;;ADJUST THE INDEX SO THAT IT WILL
6340 037362 010001 MOV R0,R1
6341 037364 070127 000006 MUL #6,R1 ;;WORK FOR THE ERROR TABLE
6342 037370 010100 MOV R1,R0
6343 037372 062700 001272 ADD #$ERRTB,R0 ;;FORM TABLE POINTER
6344 037376 012067 000004 MOV (R0)+,2$ ;;PICKUP "ERROR MESSAGE" POINTER
6345 037402 001404 BEQ 3$ ;;SKIP TIMEOUT IF NO POINTER
6346 037404 104400 TYPE ;;TYPE THE "ERROR MESSAGE"
6347 037406 000000 2$: .WORD 0 ;; "ERROR MESSAGE" POINTER GOES HERE
6348 037410 104400 001203 TYPE $SCRLF ;; "CARRIAGE RETURN" & "LINE FEED"
6349 037414 012067 000004 3$: MOV (R0)+,4$ ;;PICKUP "DATA HEADER" POINTER
6350 037420 001404 BEQ 5$ ;;SKIP TIMEOUT IF 0
6351 037422 104400 TYPE ;;TYPE THE "DATA HEADER"
6352 037424 000000 4$: .WORD 0 ;; "DATA HEADER" POINTER GOES HERE
6353 037426 104400 001203 TYPE $SCRLF ;; "CARRIAGE RETURN" & "LINE FEED"
6354 037432 011000 5$: MOV (R0),R0 ;;PICKUP "DATA TABLE" POINTER
6355 037434 001003 BNE 7$ ;;GO TYPE THE DATA
6356 037436 104400 001203 6$: TYPE $SCRLF ;; "CARRIAGE RETURN" & "LINE FEED"
6357 037442 000207 RTS PC ;;RETURN
6358 037444 7$:
6359 037444 013046 MOV 2(R0)+,-(SP) ;;SAVE 2(R0)+ FOR TIMEOUT
6360 037446 104402 TYPOC ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
6361 037450 005710 TST (R0) - ;;IS THERE ANOTHER NUMBER?
6362 037452 001771 BEQ 6$ ;;BR IF NO
6363 037454 104400 040220 TYPE $TWOSP ;;TYPE TWO(2) SPACES
6364 037460 000771 BR 7$ ;;LOOP
6365 ;:*****
6366 ;SBTTL STACK LIMIT TEST TYPE OUT ROUTINE
6367 ;* THIS ROUTINE TYPES THE ADDRESS AND STACK LIMIT REGISTER
6368 ;*VALUES THAT FAILED IN TEST 153 OR 201 IN OCTAL AND BINARY.
6369 ;:*****

```

6370	037462	104400				TYPESLE: TYPE	;TYPE
6371	037464	061227				EM263	;ERROR MESSAGE
6372	037466	104400				TYPE	;TYPE
6373	037470	061416				DH263	;DATA HEADER
6374	037472	016727	141420			MOV \$ERRPC,(PC)+	;GET ERROR PC
6375	037476	000000			1\$:	.WORD	;ERROR PC
6376	037500	016746	177772			MOV 1\$,-(SP)	;SAVE 1\$ FOR TYPEOUT
6377							;TYPE IT
6378	037504	104402				TYP0C	;GO TYPE--OCTAL ASCII(ALL DIGITS)
6379	037506	104400				TYPE	;TYPE TWO SPACES
6380	037510	040220				TWOSP	
6381	037512	016727	141472			MOV \$\$STSTNM,(PC)+	;GET TEST NUMBER
6382	037516	000000			2\$:	.WORD 0	;TEST NUMBER
6383	037520	016746	177772			MOV 2\$,-(SP)	;SAVE 2\$ FOR TYPEOUT
6384							;TYPE IT
6385	037524	104402				TYP0C	;GO TYPE--OCTAL ASCII(ALL DIGITS)
6386	037526	104400				TYPE	;TYPE CR LF
6387	037530	001203				\$CRLF	
6388	037532	104400				TYPE	;TYPE DATA HEADER
6389	037534	061443				DH263A	
6390	037536	005067	000020			CLR \$TYPEE	;INITIALIZE \$TYPEE
6391	037542	005001			STYPE:	CLR R1	;ENSURE R1 CLEAR
6392	037544	005067	000134			CLR SLDATA	
6393	037550	005067	000174			CLR SPDATA	
6394	037554	012700	120000			MOV #120000,R0	;GET ERROR DATA BUFFER POINTER
6395	037560	122710			BUFFDP:	CMPB (PC)+,(R0)	;IS THIS CORRECT TYPE?
6396	037562	000001			\$TYPEE:	.WORD 1	
6397	037564	001417				BEQ TYPEHE	;BRANCH IF YES, GO TYPE IT
6398	037566	062700	000004			ADD #4,R0	;STEP R0 TO NEXT ENTRY
6399	037572	020067	141356		NEXTEN:	CMP R0,\$REGO	;LAST ENTRY?
6400	037576	001401				BEQ 2\$;BRANCH IF YES
6401	037600	000767				BR BUFFDP	;GO CHECK NEXT ENTRY
6402	037602	000241			2\$:	CLC	;ENSURE C CLEAR
6403	037604	062767	000002	177750		ADD #2,\$TYPEE	;SELECT NEXT ERROR TYPE
6404	037612	026727	177744	000014		CMP \$TYPEE,#14	;IS TYPE ROUTINE DONE?
6405	037620	001350				BNE STYPE	;BRANCH IF NO
6406	037622	000467				BR \$\$DONE	;RETURN
6407							ERROR OF THIS TYPE
6408	037624	005701			:OUTPUT	THE TYPE HEADER IF FIRST	ERROR OF THIS TYPE?
6409	037626	001014			TYPEHE:	TST R1	
6410	037630	016701	177726			BNE 4\$;BRANCH IF NO
6411	037634	062701	062024			MOV \$TYPEE,R1	;GET TYPE NUMBER
6412	037640	011167	000006			ADD #INDEX,R1	;GET ADDRESS OF TYPE HEADER ADDRESS
6413	037644	104400	040202			MOV (R1),3\$;GET ADDRESS OF HEADER
6414	037650	104400				TYPE ,SP16	;TYPE SPACES
6415	037652	000000			3\$:	.WORD	;GO TYPE THE HEADER
6416	037654	104400				TYPE	;ADDRESS OF HEADER
6417	037656	001203				\$CRLF	;TYPE A CRLF
6418							
6419	037660	126067	000001	000017	:OUTPUT	THE DATA	
6420	037666	001005			4\$:	CMPB 1(R0),SLDATA+1	;IS SL DATA SAME AS LAST SL DATA?
6421	037670	104400				BNE 8\$;BRANCH IF NO
6422	037672	040160				TYPE SP38	;TYPE 38 SPACES
6423	037674	062700	000002			ADD #2,R0	;STEP R0 TO SP DATA
6424	037700	000422				BR SPDATA-2	;GO TO SP DATA OUTPUT
6425	037702	012027			8\$:	MOV (R0)+,(PC)+	;GET SL DATA

```

6426 037704 000000          SLDATA: .WORD
6427 037706 042767 000377 177770 BIC #377,SLDATA ;MASK HIGH BYTE
6428 037714 016746 177764 MOV SLDATA,-(SP) ;SAVE SLDATA FOR TYPEOUT
6429 ;TYPE IT
6430 037720 104402          TYP0C ;GO TYPE--OCTAL ASCII(ALL DIGITS)
6431 037722 016746 177756 MOV SLDATA,-(SP) ;PUT DATA ON STACK
6432 037726 004767 000150 JSR PC,TYPEBN ;GO TYPE BINARY
6433 037732 021067 000012 CMP (R0),SPDATA ;IS SP DATA SAME AS LAST SP DATA?
6434 037736 001003 BNE 9$ ;BRANCH IF NO
6435 037740 062700 000002 ADD #2,R0 ;STEP RO TO NEXT SL DATA
6436 037744 000413 BR TERM ;GO TERMINATE LINE
6437 037746 012027 9$: MOV (R0)+,(PC)+ ;GET SP ERROR DATA
6438 037750          SPDATA: .WORD
6439 037752 104400 040216 TYPE FOURSP ;TYPE FOUR SPACES
6440 037756 016746 177766 MOV SPDATA,-(SP) ;SAVE SPDATA FOR TYPEOUT
6441 ;GO TYPE IT IN OCTAL
6442 037762 104402          TYP0C ;GO TYPE--OCTAL ASCII(ALL DIGITS)
6443 037764 016746 177760 MOV SPDATA,-(SP) ;PUT DATA ON STACK
6444 037770 004767 000106 JSR PC,TYPEBN ;GO TYPE BINARY
6445 037774 104400 TERM: TYPE ;TYPE A CR LF
6446 037776 001203 SCRLF
6447 040000 000674 BR NEXTEIN ;GO CHECK NEXT ENTRY
6448 040002 026727 141146 157774 $$DONE: CMP $REGO,#157774 ;DID BUFFER OVERFLOW?
6449 040010 001033 BNE 1$ ;BRANCH IF NO
6450 040012 104400 040020 TYPE ,65$ ;TYPE ASCII STRING
6451 040016 000430 BR 64$ ;GET OVER THE ASCII
6452 ;65$: .ASCIIZ <15><12>/PROBABLY MORE ERRORS BUT BUFFER OVERFLOWED/<15><12>
6453 040100 64$:
6454 040100 000207 1$: RTS PC ;RETURN TO $ERROR
6455
6456 040102 016602 000002 TYPEBN: MOV 2(SP),R2 ;GET NUMBER TO BE TYPED
6457 040106 104400 TYPE ;TYPE FOUR SPACES
6458 040110 040216 FOURSP
6459 040112 012703 000010 MOV #10,R3 ;SETUP SOB COUNT
6460 040116 112767 000060 000030 3$: MOV# #60,BINARY ;PUT ASCII 0 IN LOCATION BINARY
6461 040124 006102 ROL R2 ;GET BIT TO BE TYPED
6462 040126 103005 BCC 1$ ;BRANCH IF IT IS 0
6463 040130 005567 000020 ADC BINARY ;MAKE IT ASCII
6464
6465 040134 104400 040154 TYPE ,BINARY ;GO TYPE IT
6466 040140 000402 BR 2$ ;GO TO END OF LOOP
6467 040142 104400 040217 1$: TYPE ,THRESP ;BIT WAS 0 TYPE 3 SPACES
6468 040146 077315 2$: SOB R3,3$
6469 040150 012616 MOV (SP)+,(SP) ;READ JUST STACK
6470 040152 000207 RTS PC ;RETURN
6471 040154 000 040 040 BINARY: .BYTE 0,40,40,0 ;STORAGE FOR ASCII CHARACTERS & TERMINATOR
6472 040157 000
6473 040160 020040 020040 020040 SP38: .ASCII / /
6474 040166 020040 020040 020040
6475 040174 020040 020040 020040
6476 040202 020040 020040 020040 SP16: .ASCII / /
6477 040210 020040 020040 040
6478 040215 040 FIVESP: .ASCII // /
6479 040216 040 FOURSP: .ASCII // /
6480 040217 040 THRESP: .ASCII // /
6481 040220 020040 000 TWOSP: .ASCIIZ // /
    
```

```

6482          040224          .EVEN
6483          ;*****
6484          .SBTTL  MONITOR RESTORE ROUTINE
6485          ;*THIS ROUTINE IS ENTERED BY TYPING A CHARACTER ON THE KEYBOARD
6486          ;*IF THE CHARACTER IS NOT A CTRL C EXECUTION IS RETURNED TO THE
6487          ;*TEST FOLLOWING THE ONE THAT WAS INTERRUPTED.
6488          ;*IF IT IS A CTRL C THE MONITOR IS RESTORED AND THE
6489          ;*PROCESSOR HALTS.
6490          QUIT:  CLR      R3              ;NOT ENTERED THROUGH XXDP CHAIN
6491          MOV      2$TKB,R0             ;GET CHARACTER
6492          BIC      #BIT7,R0             ;GET RID OF PARITY BIT
6493          CMP      #3,R0                ;WAS IT A CONTROL C?
6494          BNE     DUMMY                 ;BRANCH IF NO
6495          RESET                      ;CLEAR THE WORLD
6496          TYPE     ,65$                 ;TYPE ASCIZ STRING
6497          BR      64$                   ;GET OVER THE ASCIZ
6498          ;:65$: .ASCIZ  /↑C/<15><12>
6499          64$:
6500          CHAINQ: MOV     #↑D1500,R0      ;SETUP SOB COUNT
6501          MOV     #SUBTAB+2,R1          ;GET END ADDRESS OF PROGRAM
6502          MOV     #160000,R2           ;GET TOP ADDRESS OF MONITOR
6503          1$:   MOV     (R1)+,-(R2)     ;RESTORE THE MONITOR
6504          SOB    R0,1$
6505          DEC    R3
6506          BNE   2$
6507          RTS   PC
6508          2$:
6509          TYPE   ,65$                   ;:TYPE ASCIZ STRING
6510          BR    64$                     ;:GET OVER THE ASCIZ
6511          ;:65$: .ASCIZ  /MONITOR RESTORED/<15><12><15><12>
6512          64$:
6513          HALT
6514          DUMMY: CLR     2$TKB           ;CLEAR KEYBOARD BUFFER
6515          BISB   #BIT6,2$TKS          ;RESET INTERRUPT ENABLE BIT
6516          TYPE   ,65$                 ;:TYPE ASCIZ STRING
6517          BR    64$                     ;:GET OVER THE ASCIZ
6518          ;:65$: .ASCIZ  <15><12>/TYPE A CTRL C TO QUIT!!!!/<15><12>
6519          64$:
6520          JMP    2$NEXTTST             ;RETURN
6521          ;*****
6522          .SBTTL  CHECK TEST SEQUENCE ROUTINE
6523          ;*THIS ROUTINE IS CALLED IN THE SCOPE ROUTINE. IT VERIFYS
6524          ;*THAT A TEST HAS NOT BEEN SKIPPED.
6525          SEQENC: MOV    2(SP),R5        ;GET ADDRESS OF SCOPE+2
6526          SUB     #2,R5                 ;GET ADDRESS OF SCOPE CALL
6527          CLR     R4                    ;ENSURE R4 CLEAR
6528          MOVB   $TSTNM,R4             ;GET TEST NUMBER
6529          ROL    R4                      ;ADJUST
6530          CMP    ADRTAB(R4),R5         ;HAS TEST BEEN SKIPPED?
6531          BEQ    1$                      ;BRANCH IF NO
6532          ROR    R4
6533          DEC    R4
6534          MOV    R4,$REGO               ;SAVE PREVIOUS TST NUM FOR TYP0UT
6535          ;
6536          ;FIND OUT WHICH TEST THIS IS.
6537          MOV    #2,$TMP0               ;SET BUFFER HEADER POINTER

```

```

6538 040474 012767 000176 140462      MOV      #176,$TMP1      ;SET BUFFER END POINTER
6539 040502 012703 000100              MOV      #100,R3        ;SET R3 AT MIDDLE OF BUFFER
6540 040506 026305 077534      4$:    CMP      ADRTAB(R3),R5 ;IS IT THIS TEST?
6541 040512 001426              BEQ      2$             ;BRANCH IF YES
6542 040514 101014              BHI      3$             ;MOVE UP TABLE
6543                                ;CORRECT TEST IS FURTHER DOWN TABLE
6544 040516 010367 140440      MOV      R3,$TMP0      ;UPDATE HEADER POINTER
6545 040522 016702 140436      MOV      $TMP1,R2      ;GET END POINTER
6546 040526 166702 140430      SUB      $TMP0,R2      ;FIND RANGE OF REMAINING TABLE
6547 040532 000241              CLC                       ;
6548 040534 006002              ROR      R2              ;FIND MIDPOINT OF RANGE
6549 040536 060203              ADD      R2,R3          ;MAKE R3 MID POINT OF REMAINING TABLE
6550 040540 042703 000001      5$:    BIC      #BIT0,R3     ;ENSURE EVEN ADDRESS
6551 040544 000760              BR       4$             ;GO CHECK AGAIN
6552                                ;
6553                                ;CORRECT TEST IS FURTHER UP THE TABLE
6554 040546 010367 140412      3$:    MOV      R3,$TMP1      ;UPDATE END POINTER
6555 040552 010302              MOV      R3,R2          ;GET END POINTER
6556 040554 166702 140402      SUB      $TMP0,R2      ;FIND RANGE OF REMAINING TABLE
6557 040560 000241              CLC                       ;
6558 040562 006002              ROR      R2              ;FIND MID POINT OF RANGE
6559 040564 160203              SUB      R2,R3          ;MAKE R3 MIDPOINT OF REMAINING TABLE
6560 040566 000764              BR       5$             ;MAKE EVEN ADDRESS AND CHECK AGAIN
6561                                ;
6562                                ;FOUND THE CORRECT TEST
6563 040570 000241      2$:    CLC                       ;
6564 040572 006003              ROR      R3              ;GET TEST NUMBER
6565 040574 110367 140302      MOVB     R3,$STSTM      ;UPDATE TEST NUMBER
6566 040600 010367 140352      MOV      R3,$REG1      ;SAVE FOR TYPEOUT
6567 040604 104400 040612      TYPE     ,65$          ;TYPE ASCIZ STRING
6568 040610 000404              BR       64$           ;GET OVER THE ASCIZ
6569                                ;
6570 040622      65$: .ASCIZ /TEST /
6571 040622 016746 140326      64$:   MOV      $REG0,-(SP) ;SAVE $REG0 FOR TYPEOUT
6572 040626 104402              TYPOC     ;GO TYPE--OCTAL ASCII(ALL DIGITS)
6573 040630 104400 040636      TYPE     ,67$          ;TYPE ASCIZ STRING
6574 040634 000426              BR       66$           ;GET OVER THE ASCIZ
6575                                ;
6576 040712      67$: .ASCIZ / FAILED, CAUSING ECECUTION TO GO TO TEST /
6577 040712 016746 140240      66$:   MOV      $REG1,-(SP) ;SAVE $REG1 FOR TYPEOUT
6578 040716 104402              TYPOC     ;GO TYPE--OCTAL ASCII(ALL DIGITS)
6579 040720 104400 001203      TYPE     ,SCLRF
6580 040724 000207      1$:    RTS      PC        ;RETURN
6581                                ;*****
6582                                ;
6583                                .SBTTL TYPE ROUTINE
6584                                ;
6585                                ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
6586                                ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
6587                                ;*NOTE1:          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
6588                                ;*NOTE2:          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
6589                                ;*NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
6590                                ;*
6591                                ;*CALL:
6592                                ;*1) USING A TRAP INSTRUCTION
6593                                ;*      TYPE      ,MESADR      ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING

```

```

6594      ;*OR
6595      ;*   TYPE
6596      ;*   MESADR
6597      ;*
6598      ;*2) USING A JSR INSTRUCTION
6599      ;*   MOV     PS,-(SP)      ;; PUSH PROCESSOR STATUS WORD ON THE STACK
6600      ;*   JSR     PC,$TYPE     ;; CALL TYPE ROUTINE
6601      ;*   MESADDR                ;; FIRST ADDRESS OF MESSAGE
6602
6603 040726 105767 140217 $TYPE: TSTB  $TPFLG      ;; IS THERE A TERMINAL?
6604 040732 100002      BPL     1$          ;; BR IF YES
6605 040734 000000      HALT                    ;; HALT HERE IF NO TERMINAL
6606 040736 000407      BR     3$          ;; LEAVE
6607 040740 010046 1$:  MOV     RO,-(SP)      ;; SAVE RO
6608 040742 017600 000002 MOV     22(SP),RO      ;; GET ADDRESS OF ASCIZ STRING
6609 040746 112046 2$:  MOVB   (RO)+,-(SP)      ;; PUSH CHARACTER TO BE TYPED ONTO STACK
6610 040750 001005      BNE     4$          ;; BR IF IT ISN'T THE TERMINATOR
6611 040752 005726      TST     (SP)+          ;; IF TERMINATOR POP IT OFF THE STACK
6612 040754 012600      MOV     (SP)+,RO      ;; RESTORE RO
6613 040756 062716 000002 3$:  ADD     #2,(SP)          ;; ADJUST RETURN PC
6614 040762 000002      RTI                    ;; RETURN
6615 040764 122716 000011 4$:  CMPB   #HT,(SP)          ;; BRANCH IF <HT>
6616 040770 001426      BEQ     8$          ;;
6617 040772 122716 000200      CMPB   #CRLF,(SP)      ;; BRANCH IF NOT
6618 040776 001004      BNE     5$          ;;
6619 041000 005726      TST     (SP)+          ;; POP <CR><LF> EQUIV
6620 041002 104400 001203      TYPE   $CRLF          ;;
6621 041006 000757      BR     2$          ;; GET NEXT CHARACTER
6622 041010 004767 000056 5$:  JSR     PC,$TYPEC      ;; GO TYPE THIS CHARACTER
6623 041014 126726 140130 6$:  CMPB   $FILLC,(SP)+  ;; IS IT TIME FOR FILLER CHARS.?
6624 041020 001352      BNE     2$          ;; IF NO GO GET NEXT CHAR.
6625 041022 016746 140120      MOV     $NULL,-(SP)   ;; GET # OF FILLER CHARS. NEEDED
6626      ;; AND THE NULL CHAR.
6627 041026 105366 000001 7$:  DECB   1(SP)          ;; DOES A NULL NEED TO BE TYPED?
6628 041032 002770      BLT     6$          ;; BR IF NO--GO POP THE NULL OFF OF STACK
6629 041034 004767 000032      JSR     PC,$TYPEC      ;; GO TYPE A NULL
6630 041040 105367 000072      DECB   $CHARCNT      ;; DON'T COUNT THE NULL AS A CHARACTER
6631 041044 000770      BR     7$          ;; LOOP
6632
6633      ;; HORIZONTAL TAB PROCESSOR
6634
6635 041046 112716 000040 8$:  MOVB   #' (SP)          ;; REPLACE TAB WITH SPACE
6636 041052 004767 000014 9$:  JSR     PC,$TYPEC      ;; TYPE A SPACE
6637 041056 132767 000007 000052 BITB   #7,$CHARCNT      ;; BRANCH IF NOT AT
6638 041064 001372      BNE     9$          ;; TAB STOP
6639 041066 005726      TST     (SP)+          ;; POP SPACE OFF STACK
6640 041070 000726      BR     2$          ;; GET NEXT CHARACTER
6641 041072 105777 140044 $TYPEC: TSTB  2$TPS          ;; WAIT UNTIL PRINTER IS READY
6642 041076 100375      BPL     $TYPEC          ;;
6643 041100 116677 000002 140036 MOVB   2(SP),2$TPB      ;; LOAD CHAR TO BE TYPED INTO DATA REG.
6644 041106 122766 000015 000002 CMPB   #CR,2(SP)        ;; BRANCH IF
6645 041114 001003      BNE     1$          ;; NOT <CR>
6646 041116 105067 000014      CLRB   $CHARCNT      ;;
6647 041122 000406      BR     $TYPEX          ;; EXIT
6648 041124 122766 000012 000002 1$:  CMPB   #LF,2(SP)      ;; BRANCH IF
6649 041132 001402      BEQ     $TYPEX          ;; <LF>

```

```

6650 041134 105227          INCB      (PC)+      ;; INC SPACE
6651 041136 000000          $CHARCNT: .WORD 0      ;; COUNT
6652 041140 000207          $TYPEX: RTS      PC
6653
6654 ;;*****
6655
6656 .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
6657
6658 ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
6659 ;*OCTAL (ASCII) NUMBER AND TYPE IT.
6660 ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
6661 ;*CALL:
6662 ;*      MOV      NUM,-(SP)      ;; NUMBER TO BE TYPED
6663 ;*      TYPOS      ;; CALL FOR TYPEOUT
6664 ;*      .BYTE  N      ;; N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
6665 ;*      .BYTE  M      ;; M=1 OR 0
6666 ;*                               ;; 1=TYPE LEADING ZEROS
6667 ;*                               ;; 0=SUPPRESS LEADING ZEROS
6668
6669 ;*$STYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
6670 ;*$TYPOS OR $TYPOC
6671 ;*CALL:
6672 ;*      MOV      NUM,-(SP)      ;; NUMBER TO BE TYPED
6673 ;*      TYPON      ;; CALL FOR TYPEOUT
6674
6675 ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
6676 ;*CALL:
6677 ;*      MOV      NUM,-(SP)      ;; NUMBER TO BE TYPED
6678 ;*      TYPOC      ;; CALL FOR TYPEOUT
6679
6680 041142 017646 000000          $TYPOS: MOV      2(SP),-(SP)      ;; PICKUP THE MODE
6681 041146 116667 000001 000211  MOVB      1(SP), $OFILL      ;; LOAD ZERO FILL SWITCH
6682 041154 112667 000207          MOVB      (SP)+, $SOMODE+1      ;; NUMBER OF DIGITS TO TYPE
6683 041160 062716 000002          ADD      #2, (SP)      ;; ADJUST RETURN ADDRESS
6684 041164 000406          BR      $TYPON
6685 041166 112767 000001 000171  $TYPOC: MOVB      #1, $OFILL      ;; SET THE ZERO FILL SWITCH
6686 041174 112767 000006 000165  MOVB      #6, $SOMODE+1      ;; SET FOR SIX(6) DIGITS
6687 041202 112767 000005 000154  $TYPON: MOVB      #5, $OCNT      ;; SET THE ITERATION COUNT
6688 041210 010346          MOV      R3, -(SP)      ;; SAVE R3
6689 041212 010446          MOV      R4, -(SP)      ;; SAVE R4
6690 041214 010546          MOV      R5, -(SP)      ;; SAVE R5
6691 041216 116704 000145          MOVB      $SOMODE+1, R4      ;; GET THE NUMBER OF DIGITS TO TYPE
6692 041222 005404          NEG      R4
6693 041224 062704 000006          ADD      #6, R4      ;; SUBTRACT IT FOR MAX. ALLOWED
6694 041230 110467 000132          MOVB      R4, $SOMODE      ;; SAVE IT FOR USE
6695 041234 116704 000125          MOVB      $OFILL, R4      ;; GET THE ZERO FILL SWITCH
6696 041240 016605 000012          MOV      12(SP), R5      ;; PICKUP THE INPUT NUMBER
6697 041244 005003          CLR      R3      ;; CLEAR THE OUTPUT WORD
6698 041246 006105          1$: ROL      R5      ;; ROTATE MSB INTO "C"
6699 041250 000404          BR      3$      ;; GO DO MSB
6700 041252 006105          2$: ROL      R5      ;; FORM THIS DIGIT
6701 041254 006105          ROL      R5
6702 041256 006105          ROL      R5
6703 041260 010503          MOV      R5, R3
6704 041262 006103          3$: ROL      R3      ;; GET LSB OF THIS DIGIT
6705 041264 105367 000076          DECB      $SOMODE      ;; TYPE THIS DIGIT?

```



```

6706 041270 100016          BPL      7$          ;; BR IF NO
6707 041272 042703 177770  BIC      #177770,R3 ;; GET RID OF JUNK
6708 041276 001002          BNE      4$          ;; TEST FOR 0
6709 041300 005704          TST      R4          ;; SUPPRESS THIS 0?
6710 041302 001403          BEQ      5$          ;; BR IF YES
6711 041304 005204          4$: INC      R4          ;; DON'T SUPPRESS ANYMORE 0'S
6712 041306 052703 000060  BIS      #'0,R3     ;; MAKE THIS DIGIT ASCII
6713 041312 052703 000040  5$: BIS      #' ,R3   ;; MAKE ASCII IF NOT ALREADY
6714 041316 110367 000040  MOVB     R3,8$      ;; SAVE FOR TYPING
6715 041322 104400 041362  TYPE     8$        ;; GO TYPE THIS DIGIT
6716 041326 105367 000032  7$: DECB    $OCNT   ;; COUNT BY 1
6717 041332 003347          BGT      2$          ;; BR IF MORE TO DO
6718 041334 002402          BLT      6$          ;; BR IF DONE
6719 041336 005204          INC      R4          ;; INSURE LAST DIGIT ISN'T A BLANK
6720 041340 000744          BR       2$          ;; GO DO THE LAST DIGIT
6721 041342 012605          6$: MOV      (SP)+,R5  ;; RESTORE R5
6722 041344 012604          MOV      (SP)+,R4  ;; RESTORE R4
6723 041346 012603          MOV      (SP)+,R3  ;; RESTORE R3
6724 041350 016666 000002 000004 MOV      2(SP),4(SP) ;; SET THE STACK FOR RETURNING
6725 041356 012616          MOV      (SP)+,(SP)
6726 041360 000002          RTI          ;; RETURN
6727 041362 000          8$: .BYTE    0      ;; STORAGE FOR ASCII DIGIT
6728 041363 000          .BYTE    0      ;; TERMINATOR FOR TYPE ROUTINE
6729 041364 000          $OCNT: .BYTE  0    ;; OCTAL DIGIT COUNTER
6730 041365 000          $OFILL: .BYTE 0    ;; ZERO FILL SWITCH
6731 041366 000000          $OMODE: .WORD  0   ;; NUMBER OF DIGITS TO TYPE
6732          ;;*****
6733          .SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
6734
6735          ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
6736          ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
6737          ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
6738          ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
6739          ;*REPLACED WITH SPACES.
6740          ;*CALL:
6741          ;*
6742          ;* MOV      NUM,-(SP)      ;; PUT THE BINARY NUMBER ON THE STACK
6743          ;* TYPDS          ;; GO TO THE ROUTINE
6744
6745          $TYPDS:
6746          MOV      R0,-(SP)      ;; PUSH R0 ON STACK
6747          MOV      R1,-(SP)      ;; PUSH R1 ON STACK
6748          MOV      R2,-(SP)      ;; PUSH R2 ON STACK
6749          MOV      R3,-(SP)      ;; PUSH R3 ON STACK
6750          MOV      R5,-(SP)      ;; PUSH R5 ON STACK
6751          MOV      #20200,-(SP)   ;; SET BLANK SWITCH AND SIGN
6752          MOV      20(SP),R5     ;; GET THE INPUT NUMBER
6753          BPL      1$          ;; BR IF INPUT IS POS.
6754          NEG      R5          ;; MAKE THE BINARY NUMBER POS.
6755          MOVB     #'-,1(SP)     ;; MAKE THE ASCII NUMBER NEG.
6756          1$: CLR      R0          ;; ZERO THE CONSTANTS INDEX
6757          MOV      #0BLK,R3      ;; SETUP THE OUTPUT POINTER
6758          MOVB     #' ,(R3)+     ;; SET THE FIRST CHARACTER TO A BLANK
6759          2$: CLR      R2          ;; CLEAR THE BCD NUMBER
6760          MOV      $DTBL(R0),R1  ;; GET THE CONSTANT
6761          3$: SUB      R1,R5     ;; FORM THIS BCD DIGIT

```

```

6762 041446 002402          BLT      4$          ;; BR IF DONE
6763 041450 005202          INC      R2          ;; INCREASE THE BCD DIGIT BY 1
6764 041452 000774          BR       3$
6765 041454 060105          4$: ADD     R1,R5      ;; ADD BACK THE CONSTANT
6766 041456 005702          TST     R2          ;; CHECK IF BCD DIGIT=0
6767 041460 001002          BNE     5$          ;; FALL THROUGH IF 0
6768 041462 105716          TSTB   (SP)        ;; STILL DOING LEADING 0'S?
6769 041464 100407          BMI     7$          ;; BR IF YES
6770 041466 106316          5$: ASLB   (SP)        ;; MSD?
6771 041470 103003          BCC     6$          ;; BR IF NO
6772 041472 116663 000001 177777  MOVB   1(SP),-1(R3) ;; YES--SET THE SIGN
6773 041500 052702 000060 6$: BIS   #'0,R2      ;; MAKE THE BCD DIGIT ASCII
6774 041504 052702 000040 7$: BIS   #' ,R2      ;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
6775 041510 110223          MOVB   R2,(R3)+    ;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
6776 041512 005720          TST   (R0)+        ;; JUST INCREMENTING
6777 041514 020027 000210  CMP    RO,#10      ;; CHECK THE TABLE INDEX
6778 041520 002746          BLT    2$          ;; GO DO THE NEXT DIGIT
6779 041522 003002          BGT    8$          ;; GO TO EXIT
6780 041524 010502          MOV    R5,R2      ;; GET THE LSD
6781 041526 000764          BR     6$          ;; GO CHANGE TO ASCII
6782 041530 105726          8$: TSTB (SP)+      ;; WAS THE LSD THE FIRST NON-ZERO?
6783 041532 100003          BPL    9$          ;; BR IF NO
6784 041534 116663 177777 177776 9$: MOVB -1(SP),-2(R3) ;; YES--SET THE SIGN FOR TYPING
6785 041542 105013          CLRB  (R3)        ;; SET THE TERMINATOR
6786 041544 012605          MOV   (SP)+,R5    ;; POP STACK INTO R5
6787 041546 012603          MOV   (SP)+,R3    ;; POP STACK INTO R3
6788 041550 012602          MOV   (SP)+,R2    ;; POP STACK INTO R2
6789 041552 012601          MOV   (SP)+,R1    ;; POP STACK INTO R1
6790 041554 012600          MOV   (SP)+,R0    ;; POP STACK INTO R0
6791 041556 104400 041604  TYPE   $DBLK      ;; NOW TYPE THE NUMBER
6792 041562 016666 000002 000004  MOV   2(SP),4(SP) ;; ADJUST THE STACK
6793 041570 012616          MOV   (SP)+,(SP)
6794 041572 000002          RTI
6795 041574 023420          SDBL: 10000.      ;; RETURN TO USER
6796 041576 001750          1000.
6797 041600 000144          100.
6798 041602 000012          10.
6799 041604 000004          SDBLK: .BLKW 4
6800          ;;*****
6801          .SBTTL TRAP DECODER
6802          ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
6803          ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
6804          ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
6805          ;*GO TO THAT ROUTINE.
6806
6807
6808
6809 041614 010046          STRAP: MOV    RO,-(SP) ;; SAVE RO
6810 041616 016600 000002  MOV    2(SP),RO      ;; GET TRAP ADDRESS
6811 041622 005740          TST   -(RO)         ;; BACKUP BY 2
6812 041624 111000          MOVB  (RO),RO       ;; GET RIGHT BYTE OF TRAP
6813 041626 016000 041634  MOV   $TRPAD(RO),RO ;; INDEX TO TABLE
6814 041632 000200          RTS   RO            ;; GO TO ROUTINE
6815
6816          .SBTTL TRAP TABLE
6817

```

6818
6819
6820
6821
6822
6823
6824
6825
6826
6827
6828
6829
6830
6831
6832
6833
6834
6835
6836
6837
6838
6839
6840
6841
6842
6843
6844
6845
6846
6847
6848
6849
6850
6851
6852
6853
6854
6855
6856
6857
6858
6859
6860
6861
6862
6863
6864
6865
6866
6867
6868
6869
6870
6871
6872
6873

041634
041634 040726
041636 041166
041640 041142
041642 041202
041644 041370

041646 012737 041770 000024
041654 012737 000340 000026
041662 010046
041664 010146
041666 010246
041670 010346
041672 010446
041674 010546
041676 010667 000072
041702 012737 041714 000024
041710 000000
041712 000776

041714 016706 000054
041720 005067 000050
041724 005267 000044
041730 001375
041732 012605
041734 012604
041736 012603
041740 012602
041742 012601
041744 012600
041746 012737 041646 000024
041754 012737 000340 000026
041762 104400
041764 041776
041766 000002
041770 000000
041772 000776
041774 000000
041776 005015 047520 042527
042004 000122

042006 044505 044124 051105
042014 051440 046120 043040
042022 044501 042514 020104
042030 051117 050040 053523

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

ROUTINE

\$TRPAD:

```

$TYPE    ;;CALL=TYPE    TRAP+0(104400)  TTY TYPEOUT ROUTINE
$TYPOC   ;;CALL=TYPOC   TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
$TYPOS   ;;CALL=TYPOS   TRAP+4(104404)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
$TYPON   ;;CALL=TYPON   TRAP+6(104406)  TYPE OCTAL NUMBER (AS PER LAST CALL)
$TYPDS   ;;CALL=TYPDS   TRAP+10(104410) TYPE DECIMAL NUMBER (WITH SIGN)
    
```

;;*****

.SBTTL POWER DOWN AND UP ROUTINES

:POWER DOWN ROUTINE

```

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

:POWER UP ROUTINE

```

$PWRUP: MOV    $SAVR6,SP            ;;GET SP
        CLR    $SAVR6              ;;WAIT LOOP FOR THE TTY
        IS:   INC    $SAVR6         ;;WAIT FOR THE INC
        BNE   IS                    ;;OF WORD
        MOV   (SP)+,R5              ;;POP STACK INTO R5
        MOV   (SP)+,R4              ;;POP STACK INTO R4
        MOV   (SP)+,R3              ;;POP STACK INTO R3
        MOV   (SP)+,R2              ;;POP STACK INTO R2
        MOV   (SP)+,R1              ;;POP STACK INTO R1
        MOV   (SP)+,R0              ;;POP STACK INTO R0
        MOV   #$PWRDN,2#$PWRVEC    ;;SET UP THE POWER DOWN VECTOR
        MOV   #340,2#$PWRVEC+2    ;;PRIO:7
        TYPE  $POWER               ;;REPORT THE POWER FAILURE
        $PWRMG: .WORD $POWER        ;;POWER FAIL MESSAGE POINTER
        RTI
        $SILLUP: HALT              ;;THE POWER UP SEQUENCE WAS STARTED
        BR     .-2                  ;;BEFORE THE POWER DOWN WAS COMPLETE
        $SAVR6: 0                   ;;PUT THE SP HERE
        $POWER: .ASCIZ <15><12>"POWER"
    
```

.EVEN

EM1: .ASCIZ /EITHER SPL FAILED OR PSW PRIORITY BITS STUCK/

6874	042036	050040	044522	051117					
6875	042044	052111	020131	044502					
6876	042052	051524	051440	052524					
6877	042060	045503	000						
6878	042063	105	051122	051117	DH1:	.ASCII	/ERROR PC	SPL 5 PSW	SPL 2 PSW TST NUM/<CRLF>
6879	042070	050040	020103	020040					
6880	042076	050123	020114	020065					
6881	042104	051520	020127	020040					
6882	042112	020040	020040	050123					
6883	042120	020114	020062	051520					
6884	042126	020127	020040	051524					
6885	042134	020124	052516	100115					
6886	042142	020011	054105	042520		.ASCIZ	/	EXPECT ACTUAL	EXPECT ACTUAL/
6887	042150	052103	040440	052103					
6888	042156	040525	020114	020040					
6889	042164	054105	042520	052103					
6890	042172	040440	052103	040525					
6891	042200	000114							
6892						.EVEN			
6893	042202	001116	001214	001206	DT1:	.WORD	\$ERRPC,\$SPR5,\$SERPSW,\$SPR2,\$TMP0,\$\$STSTNM,0		
6894	042210	001212	001162	001210					
6895	042216	000000							
6896	042220	051120	032440	047440	EM2:	.ASCIZ	/PR 5 OK PR 2 BAD/		
6897	042226	020113	051120	031040					
6898	042234	041040	042101	000					
6899	042241	120	020122	020062	EM3:	.ASCIZ	/PR 2 OK BUT PR 5 BAD/		
6900	042246	045517	041040	052125					
6901	042254	050040	020122	020065					
6902	042262	040502	000104						
6903	042266	040522	043103	042440	EM4:	.ASCIZ	/RACF E3 BAD(NOT GOING HIGH ON SPL)/		
6904	042274	020063	040502	024104					
6905	042302	047516	020124	047507					
6906	042310	047111	020107	044510					
6907	042316	044107	047440	020116					
6908	042324	050123	024514	000					
6909	042331	105	051122	051117	DH4:	.ASCIZ	/ERRORPC TEST NUMBER/		
6910	042336	041520	052040	051505					
6911	042344	020124	052516	041115					
6912	042352	051105	000						
6913		042356				.EVEN			
6914	042356	001116	001210	000000	DT4:	.WORD	\$ERRPC,\$\$STSTNM,0		
6915	042364	040522	043103	042440	EM5:	.ASCIZ	/RACF E5 BAD(AFIRO4(1)*AFIRO3(1)*AFIRO5(1)*(RTS:CCOP))/		
6916	042372	020065	040502	024104					
6917	042400	043101	051111	032060					
6918	042406	030450	025051	043101					
6919	042414	051111	031460	030450					
6920	042422	025051	043101	051111					
6921	042430	032460	030450	025051					
6922	042436	051050	051524	041472					
6923	042444	047503	024520	000051					
6924	042452	042522	042523	020124	EM6:	.ASCIZ	/RESET DID NOT SEND OUT INIT/		
6925	042460	044504	020104	047516					
6926	042466	020124	042523	042116					
6927	042474	047440	052125	044440					
6928	042502	044516	000124						
6929	042506	040522	043103	042440	EM7:	.ASCIZ	/RACF E5 BAD/		

6930	042514	020065	040502	000104				
6931	042522	040522	043103	042440	EM10:	.ASCIZ	/RACF EB BAD/	
6932	042530	020070	040502	000104				
6933	042536	040522	043103	042440	EM11:	.ASCIZ	"RACF E17 BAD(AFIRO7(0)*X/CLASS"	
6934	042544	033461	041040	042101				
6935	042552	040450	044506	030122				
6936	042560	024067	024460	054052				
6937	042566	041457	040514	051523				
6938	042574	000						
6939	042575	122	041501	020106	EM12:	.ASCIZ	"RACF E5 BAD(AFIRO6(0)*X/CLASS"	
6940	042602	032505	041040	042101				
6941	042610	040450	044506	030122				
6942	042616	024066	024460	054052				
6943	042624	041457	040514	051523				
6944	042632	000051						
6945	042634	041520	020102	044504	EM13:	.ASCIZ	/PCB DID NOT LOAD FROM R5/	
6946	042642	020104	047516	020124				
6947	042650	047514	042101	043040				
6948	042656	047522	020115	032522				
6949	042664	000						
6950	042665	123	020120	044504	EM14:	.ASCIZ	/SP DID NOT LOAD PROPERLY/	
6951	042672	020104	047516	020124				
6952	042700	047514	042101	050040				
6953	042706	047522	042520	046122				
6954	042714	000131						
6955	042716	051105	047522	050122	DH14:	.ASCII	/ERRORPC SP TST NUM/<CRLF>	
6956	042724	020103	020040	020040				
6957	042732	020040	050123	020011				
6958	042740	051524	020124	052516				
6959	042746	100115						
6960	042750	020040	020040	020040		.ASCIZ	/ EXPECT ACTUAL/	
6961	042756	020040	054105	042520				
6962	042764	052103	020040	041501				
6963	042772	052524	046101	000				
6964		043000				.EVEN		
6965	043000	001116	001156	001154	DT14:	.WORD	\$ERRPC,\$REG1,\$REG0,\$\$TSTNM,0	
6966	043006	001210	000000					
6967	043012	032522	042040	042111	EM15:	.ASCIZ	/R5 DID NOT LOAD PROPERLY/	
6968	043020	047040	052117	046040				
6969	043026	040517	020104	051120				
6970	043034	050117	051105	054514				
6971	043042	000						
6972	043043	105	051122	051117	DH15:	.ASCII	/ERRORPC R5 TST NUM/<CRLF>	
6973	043050	041520	020040	020040				
6974	043056	020040	051040	020065				
6975	043064	020040	020040	020040				
6976	043072	052040	052123	047040				
6977	043100	046525	200					
6978	043103	011	042440	050130		.ASCIZ	/ EXPECT ACTUAL/	
6979	043110	041505	020124	040440				
6980	043116	052103	040525	000114				
6981	043124	040522	043103	054040	EM16:	.ASCIZ	"RACF X/CLASS DID NOT GO HIGH"	
6982	043132	041457	040514	051523				
6983	043140	042040	042111	047040				
6984	043146	052117	043440	020117				
6985	043154	044510	044107	000				

6986	043161	105	052111	042510	EM17:	.ASCIZ	/EITHER RACF E34 BAD OR NOT GETTING THRU RACL RADROS/
6987	043166	020122	040522	043103			
6988	043174	042440	032063	041040			
6989	043202	042101	047440	020122			
6990	043210	047516	020124	042507			
6991	043216	052124	047111	020107			
6992	043224	044124	052522	051040			
6993	043232	041501	020114	040522			
6994	043240	051104	032460	000			
6995	043245	107	040522	020112	EM20:	.ASCIZ	/GRAJ SC05 L DOES NOT GET THRU TO RACK BRCAB04 L AS A HIGH/
6996	043252	041523	032460	046040			
6997	043260	042040	042517	020123			
6998	043266	047516	020124	042507			
6999	043274	020124	044124	052522			
7000	043302	052040	020117	040522			
7001	043310	045503	041040	041522			
7002	043316	041101	032060	046040			
7003	043324	040440	020123	020101			
7004	043332	044510	044107	000			
7005	043337	122	020061	044504	EM21:	.ASCIZ	/R1 DID NOT SHIFT/
7006	043344	020104	047516	020124			
7007	043352	044123	043111	000124			
7008	043360	030522	051440	044510	EM22:	.ASCII	/R1 SHIFTED BUT CARRY DID NOT SET/<CRLF>
7009	043366	052106	042105	041040			
7010	043374	052125	041440	051101			
7011	043402	054522	042040	042111			
7012	043410	047040	052117	051440			
7013	043416	052105	200				
7014	043421	123	044510	052106		.ASCIZ	/SHIFT COUNTER COULD BE STUCK/
7015	043426	041440	052517	052116			
7016	043434	051105	041440	052517			
7017	043442	042114	041040	020105			
7018	043450	052123	041525	000113			
7019	043456	051107	045101	051440	EM23:	.ASCIZ	/GRAJ SC05 L DOES NOT GET THRU TO RACK BRCAB04 L AS A LOW/
7020	043464	030103	020065	020114			
7021	043472	047504	051505	047040			
7022	043500	052117	043440	052105			
7023	043506	052040	051110	020125			
7024	043514	047524	051040	041501			
7025	043522	020113	051102	040503			
7026	043530	030102	020064	020114			
7027	043536	051501	040440	046040			
7028	043544	053517	000				
7029	043547	101	044123	051040	EM24:	.ASCIZ	/ASH RIGHT DID NOT SIGN FILL/
7030	043554	043511	052110	042040			
7031	043562	042111	047040	052117			
7032	043570	051440	043511	020116			
7033	043576	044506	046114	000			
7034	043603	122	020061	044504	EM25:	.ASCIZ	/R1 DID NOT SHIFT CORRECTLY/
7035	043610	020104	047516	020124			
7036	043616	044123	043111	020124			
7037	043624	047503	051122	041505			
7038	043632	046124	000131				
7039	043636	051105	047522	050122	DH25:	.ASCII	/ERRORPC R1 TST NUM/<CRLF>
7040	043644	020103	020040	020040			
7041	043652	020040	030522	020011			

7042	043660	051524	020124	052516		
7043	043666	100115				
7044	043670	020011	054105	042520	.ASCIZ /	EXPECT ACTUAL/
7045	043676	052103	020040	041501		
7046	043704	052524	046101	000		
7047		043712				
7048	043712	001116	001164	001156	DT25:	.EVEN
7049	043720	001210	000000			.WORD \$ERRPC,\$TMP1,\$REG1,\$STSTNM,0
7050	043724	030522	051440	044510	EM26:	.ASCIZ /R1 SHIFTED BUT CARRY DID NOT SET/
7051	043732	052106	042105	041040		
7052	043740	052125	041440	051101		
7053	043746	054522	042040	042111		
7054	043754	047040	052117	051440		
7055	043762	052105	000			
7056	043765	101	044123	031056	EM27:	.ASCIZ /ASH.20 DID NOT LOAD CC'S CORRECTLY/
7057	043772	020060	044504	020104		
7058	044000	047516	020124	047514		
7059	044006	042101	041440	023503		
7060	044014	020123	047503	051122		
7061	044022	041505	046124	000131		
7062	044030	030522	051440	044510	EM30:	.ASCIZ /R1 SHIFTED WHEN SHIFT COUNT=0/
7063	044036	052106	042105	053440		
7064	044044	042510	020116	044123		
7065	044052	043111	020124	047503		
7066	044060	047125	036524	000060		
7067	044066	051501	027110	030064	EM31:	.ASCIZ /ASH.40 DID NOT LOAD CC'S CORRECTLY/
7068	044074	042040	042111	047040		
7069	044102	052117	046040	040517		
7070	044110	020104	041503	051447		
7071	044116	041440	051117	042522		
7072	044124	052103	054514	000		
7073	044131	122	041501	020106	EM32:	.ASCIZ "RACF U/CLASS DOES NOT GO HIGH ON ASH"
7074	044136	027525	046103	051501		
7075	044144	020123	047504	051505		
7076	044152	047040	052117	043440		
7077	044160	020117	044510	044107		
7078	044166	047440	020116	051501		
7079	044174	000110				
7080	044176	051501	027110	030060	EM33:	.ASCIZ /ASH.00 FAILED/
7081	044204	043040	044501	042514		
7082	044212	000104				
7083	044214	051111	041103	042440	EM34:	.ASCIZ /IRCB E35(B2) NOT GOING LOW/
7084	044222	032463	041050	024462		
7085	044230	047040	052117	043440		
7086	044236	044517	043516	046040		
7087	044244	053517	000			
7088	044247	105	052111	042510	EM35:	.ASCII /EITHER IRCB (MUL:ASHC)+MFP L DID NOT GO LOW OR IRCB E37 BAD/<<CRLF>
7089	044254	020122	051111	041103		
7090	044262	024040	052515	035114		
7091	044270	051501	041510	025451		
7092	044276	043115	020120	020114		
7093	044304	044504	020104	047516		
7094	044312	020124	047507	046040		
7095	044320	053517	047440	020122		
7096	044326	051111	041103	042440		
7097	044334	033463	041040	042101		

K13

7098	044342	200						
7099	044343	117	020122	051111		.ASCIZ	/OR IRCB E35(B1) STUCK LOW/	
7100	044350	041103	042440	032463				
7101	044356	041050	024461	051440				
7102	044364	052524	045503	046040				
7103	044372	053517	000					
7104	044375	105	052111	042510	EM36:	.ASCIZ	/EITHER RACE (MUL:ASHC+MFP) DID NOT GO HIGH OR RACE E44 BAD/	
7105	044402	020122	040522	042503				
7106	044410	024040	052515	035114				
7107	044416	051501	041510	046453				
7108	044424	050106	020051	044504				
7109	044432	020104	047516	020124				
7110	044440	047507	044040	043511				
7111	044446	020110	051117	051040				
7112	044454	041501	020105	032105				
7113	044462	020064	040502	000104				
7114	044470	040522	042503	042440	EM37:	.ASCIZ	/RACE E45 BAD (AFIRO4(1)*(MUL:ASHC+MFP))/	
7115	044476	032464	041040	042101				
7116	044504	024040	043101	051111				
7117	044512	032060	030450	025051				
7118	044520	046450	046125	040472				
7119	044526	044123	025503	043115				
7120	044534	024520	000051					
7121	044540	040522	042503	042440	EM40:	.ASCIZ	/RACE E33 BAD (AFIRO5(1)*(MUL:ASHC+MFP))/	
7122	044546	031463	041040	042101				
7123	044554	024040	043101	051111				
7124	044562	032460	030450	025051				
7125	044570	046450	046125	040472				
7126	044576	044123	025503	043115				
7127	044604	024520	000051					
7128	044610	030122	042040	042111	EM41:	.ASCIZ	/RO DID NOT SIGN FILL ON RIGHT SHIFT/	
7129	044616	047040	052117	051440				
7130	044624	043511	020116	044506				
7131	044632	046114	047440	020116				
7132	044640	044522	044107	020124				
7133	044646	044123	043111	000124				
7134	044654	051105	047522	050122	DH41:	.ASCII	/ERRORPC RO TST NUM/<CRLF>	
7135	044662	020103	020040	020040				
7136	044670	020040	051040	004460				
7137	044676	020040	051524	020124				
7138	044704	052516	100115					
7139	044710	020011	042440	050130		.ASCIZ	/ EXPECT ACTUAL/	
7140	044716	041505	020124	040440				
7141	044724	052103	040525	000114				
7142								
7143	044732	001116	001162	001154	DT41:	.EVEN		
7144	044740	001210	000000			.WORD	\$ERRPC,\$TMPD,\$REGD,\$STSTNM,0	
7145	044744	040502	020104	041503	EM42:	.ASCIZ	/BAD CC'S ON RIGHT SHIFT/	
7146	044752	051447	047440	020116				
7147	044760	044522	044107	020124				
7148	044766	044123	043111	000124				
7149	044774	051105	047522	050122	DH42:	.ASCII	/ERRORPC PSW TST NUM/<CRLF>	
7150	045002	020103	020040	020040				
7151	045010	020040	051520	004527				
7152	045016	020040	051524	020124				
7153	045024	052516	100115					

7154	045030	020011	054105	042520		.ASCIZ /	EXPECT	ACTUAL/		
7155	045036	052103	020040	041501						
7156	045044	052524	046101	000						
7157		045052								
7158	045052	001116	001162	001206	DT42:	.EVEN	\$ERRPC,\$TMPD,\$ERPSW,\$STSTNM,0			
7159	045060	001210	000000			.WORD				
7160	045064	030122	030074	020076	EM43:	.ASCIZ	/RO<D> DID NOT GO TO R1<15>/			
7161	045072	044504	020104	047516						
7162	045100	020124	047507	052040						
7163	045106	020117	030522	030474						
7164	045114	037065	000							
7165	045117	0105	051122	051117	DH43:	.ASCII	/ERRORPC	RO	R1	TST NUM/<<CRLF>
7166	045124	041520	020040	020040						
7167	045132	020040	051040	004460						
7168	045140	020040	020040	020040						
7169	045146	020040	030522	020011						
7170	045154	052040	052123	047040						
7171	045162	046525	200							
7172	045165	011	042440	050130		.ASCIZ /	EXPECT	ACTUAL	EXPECT	ACTUAL/
7173	045172	041505	020124	040440						
7174	045200	052103	040525	020114						
7175	045206	020040	054105	042520						
7176	045214	052103	020040	041501						
7177	045222	052524	046101	000						
7178		045230								
7179	045230	001116	001162	001154	DT43:	.EVEN	\$ERRPC,\$TMPD,\$REGD,\$TMP1,\$REG1,\$STSTNM,0			
7180	045236	001164	001156	001210		.WORD				
7181	045244	000000								
7182	045246	030122	042040	042111	EM44:	.ASCIZ	/RO DID NOT GET SHIFTED LEFT PROPERLY/			
7183	045254	047040	052117	043440						
7184	045262	052105	051440	044510						
7185	045270	052106	042105	046040						
7186	045276	043105	020124	051120						
7187	045304	050117	051105	054514						
7188	045312	000								
7189	045313	0102	042101	041440	EM45:	.ASCIZ	/BAD CC'S ON LEFT SHIFT/			
7190	045320	023503	020123	047117						
7191	045326	046040	043105	020124						
7192	045334	044123	043111	000124						
7193	045342	040502	020104	041503	EM47:	.ASCIZ	/BAD CC'S ON NO SHIFT/			
7194	045350	051447	047440	020116						
7195	045356	047516	051440	044510						
7196	045364	052106	000							
7197	045367	0122	020061	044504	EM50:	.ASCIZ	/R1 DID NOT ROTATE PROPERLY/			
7198	045374	020104	047516	020124						
7199	045402	047522	040524	042524						
7200	045410	050040	047522	042520						
7201	045416	046122	000131							
7202	045422	044502	020124	052123	EM51:	.ASCIZ	/BIT STUCK IN SC WITH 52 PATTERN/			
7203	045430	041525	020113	047111						
7204	045436	051440	020103	044527						
7205	045444	044124	032440	020062						
7206	045452	040520	052124	051105						
7207	045460	000116								
7208	045462	051105	047522	050122	DH51:	.ASCII	/ERRORPC	RO	R1	C BIT TST NUM/<<CRLF>
7209	045470	020103	020040	020040						

M13

7210	045476	020040	030122	020011		
7211	045504	020040	020040	020040		
7212	045512	030522	020011	020040		
7213	045520	041440	041040	052111		
7214	045526	020011	051524	020124		
7215	045534	052516	100115			
7216	045540	020011	054105	042520	.ASCIZ /	EXPECT ACTUAL EXPECT ACTUAL EXPECT ACTUAL/
7217	045546	052103	020040	041501		
7218	045554	052524	046101	020040		
7219	045562	054105	042520	052103		
7220	045570	020040	041501	052524		
7221	045576	046101	020040	054105		
7222	045604	042520	052103	020040		
7223	045612	041501	052524	046101		
7224	045620	000				
7225		045622				
7226	045622	001116	001162	001154	DT51: .EVEN	
7227	045630	001164	001156	001166	.WORD	SERRPC, STMP0, SREG0, STMP1, SREG1, STMP2, SERPSW, SSTSTNM, 0
7228	045636	001206	001210	000000		
7229	045644	044502	020124	052123	EMS2: .ASCIZ	/BIT STUCK IN SHIFT COUNTER WITH 25 PATTERN/
7230	045652	041525	020113	047111		
7231	045660	051440	044510	052106		
7232	045666	041440	052517	052116		
7233	045674	051105	053440	052111		
7234	045702	020110	032462	050040		
7235	045710	052101	042524	047122		
7236	045716	000				
7237	045717	111	041522	020102	EMS3: .ASCIZ	/IRCB E35(B3) NOT GOING LOW/
7238	045724	031505	024065	031502		
7239	045732	020051	047516	020124		
7240	045740	047507	047111	020107		
7241	045746	047514	000127			
7242	045752	051501	027103	030060	EMS4: .ASCIZ	/ASC.00 FAILED/
7243	045760	043040	044501	042514		
7244	045766	000104				
7245	045770	024122	052515	035114	EMS5: .ASCIZ	/R(MUL:ASHC+MFP) FIELD IN INSTR DECODE ROM BAD/
7246	045776	051501	041510	046453		
7247	046004	050106	020051	044506		
7248	046012	046105	020104	047111		
7249	046020	044440	051516	051124		
7250	046026	042040	041505	042117		
7251	046034	020105	047522	020115		
7252	046042	040502	000104			
7253	046046	044505	044124	051105	EMS6: .ASCIZ	/EITHER GRAD DROO STUCK HIGH OR RACK E64(C1) BAD/
7254	046054	043440	040522	020104		
7255	046062	051104	030060	051440		
7256	046070	052524	045503	044040		
7257	046076	043511	020110	051117		
7258	046104	051040	041501	020113		
7259	046112	033105	024064	030503		
7260	046120	020051	040502	000104		
7261	046126	044505	044124	051105	EMS7: .ASCIZ	/EITHER GRAD DROO STUCK LOW OR RACK E64(C1) BAD/
7262	046134	043440	040522	020104		
7263	046142	051104	030060	051440		
7264	046150	052524	045503	046040		
7265	046156	053517	047440	020122		

7266	046164	040522	045503	042440	
7267	046172	032066	041450	024461	
7268	046200	041040	042101	000	
7269	046205	105	052111	042510	EM60: .ASCII /EITHER GRAJ SC=0 L NOT GETTING TO RACK ESO(C1) OR/<CRLF>
7270	046212	020122	051107	045101	
7271	046220	051440	036503	020060	
7272	046226	020114	047516	020124	
7273	046234	042507	052124	047111	
7274	046242	027107	047524	051040	
7275	046250	041501	020113	032505	
7276	046256	024060	030503	020051	
7277	046264	051117	200		
7278	046267	040	032505	020060	.ASCIZ / ESO BAD (FAILED LOW)/
7279	046274	040502	020104	043050	
7280	046302	044501	042514	020104	
7281	046310	047514	024527	000	
7282	046315	105	052111	042510	EM61: .ASCIZ /EITHER RO OR R1 OR BOTH BAD ON POSITIVE MULTIPLICAND/
7283	046322	020122	030122	047440	
7284	046330	020122	030522	047440	
7285	046336	020122	047502	044124	
7286	046344	041040	042101	047440	
7287	046352	020116	047520	044523	
7288	046360	044524	042526	046440	
7289	046366	046125	044524	046120	
7290	046374	041511	047101	000104	
7291	046402	040502	020104	041503	EM62: .ASCIZ /BAD CC'S ON POSITIVE MULTIPLICAND/
7292	046410	051447	047440	020116	
7293	046416	047520	044523	044524	
7294	046424	042526	046440	046125	
7295	046432	044524	046120	041511	
7296	046440	047101	000104		
7297	046444	030122	041040	042101	EM63: .ASCIZ /RO BAD ON NEGATIVE MULTIPLICAND/
7298	046452	047440	020116	042516	
7299	046460	040507	044524	042526	
7300	046466	046440	046125	044524	
7301	046474	046120	041511	047101	
7302	046502	000104			
7303	046504	030522	041040	042101	EM64: .ASCIZ /R1 BAD ON NEGATIVE MULTIPLICAND/
7304	046512	047440	020116	042516	
7305	046520	040507	044524	042526	
7306	046526	046440	046125	044524	
7307	046534	046120	041511	047101	
7308	046542	000104			
7309	046544	040502	020104	041503	EM65: .ASCIZ /BAD CC'S DUE TO STATE MUL.50/
7310	046552	051447	042040	042525	
7311	046560	052040	020117	052123	
7312	046566	052101	020105	052515	
7313	046574	027114	030065	000	
7314	046601	103	042040	042111	EM66: .ASCIZ /C DID NOT SET ON OVERFLOW/
7315	046606	047040	052117	051440	
7316	046614	052105	047440	020116	
7317	046622	053117	051105	046106	
7318	046630	053517	000		
7319	046633	103	042040	042111	EM67: .ASCIZ /C DID NOT SET ON UNDERFLOW/
7320	046640	047040	052117	051440	
7321	046646	052105	047440	020116	

7322	046654	047125	042504	043122		
7323	046662	047514	000127			
7324	046666	052515	027114	030060	EM70:	.ASCIZ /MUL.00 FAILED/
7325	046674	043040	044501	042514		
7326	046702	000104				
7327	046704	051501	027110	030063	EM72:	.ASCIZ /ASH.30 DID NOT LOAD CC'S CORRECTLY/
7328	046712	042040	042111	047040		
7329	046720	052117	046040	040517		
7330	046726	020104	041503	051447		
7331	046734	041440	051117	042522		
7332	046742	052103	054514	000		
7333	046747	101	044123	032056	EM73:	.ASCIZ /ASH.41 FAILED/
7334	046754	020061	040506	046111		
7335	046762	042105	000			
7336	046765	101	044123	032056	EM74:	.ASCIZ /ASH.41 DID NOT LOAD CC'S CORRECTLY/
7337	046772	020061	044504	020104		
7338	047000	047516	020124	047514		
7339	047006	042101	041440	023503		
7340	047014	020123	047503	051122		
7341	047022	041505	046124	000131		
7342	047030	044505	044124	051105	EM75:	.ASCIZ /EITHER IRCF Z2(1) DOES NOT GO HIGH OR RACK E49(B1) STUCK LOW/
7343	047036	044440	041522	020105		
7344	047044	031132	030450	020051		
7345	047052	047504	051505	047040		
7346	047060	052117	043440	020117		
7347	047066	044510	044107	047440		
7348	047074	020122	040522	045503		
7349	047102	042440	034464	041050		
7350	047110	024461	051440	052524		
7351	047116	045503	046040	053517		
7352	047124	000				
7353	047125	103	023503	020123	EM76:	.ASCIZ /CC'S BAD IN DVE.00/
7354	047132	040502	020104	047111		
7355	047140	042040	042526	030056		
7356	047146	000060				
7357	047150	041503	051447	041040	EM77:	.ASCIZ /CC'S BAD IN DVC.70/
7358	047156	042101	044440	020116		
7359	047164	053104	027103	030067		
7360	047172	000				
7361	047173	122	041501	020113	EM100:	.ASCIZ /RACK E50(B0) STUCK HIGH/
7362	047200	032505	024060	030102		
7363	047206	020051	052123	041525		
7364	047214	020113	044510	044107		
7365	047222	000				
7366	047223	105	052111	042510	EM101:	.ASCIZ /EITHER GRAJ DIV SUB L NOT GOING LOW OR NOT GETTING THRU/<<CRLF>
7367	047230	020122	051107	045101		
7368	047236	042040	053111	051440		
7369	047244	041125	046040	047040		
7370	047252	052117	043440	044517		
7371	047260	043516	046040	053517		
7372	047266	047440	020122	047516		
7373	047274	020124	042507	052124		
7374	047302	047111	020107	044124		
7375	047310	052522	000200			
7376	047314	047524	051040	041501		.ASCIZ /TO RACK E49 OR E49(D0) STUCK HIGH/
7377	047322	020113	032105	020071		

7378	047330	051117	042440	034464	
7379	047336	042050	024460	051440	
7380	047344	052524	045503	044040	
7381	047352	043511	000110		
7382	047356	052521	052117	047440	EM102: .ASCIZ /QUOT OK REMAINDER BAD (ROM STATE FAILURE)/
7383	047364	020113	042522	040515	
7384	047372	047111	042504	020122	
7385	047400	040502	020104	051050	
7386	047406	046517	051440	040524	
7387	047414	042524	043040	044501	
7388	047422	052514	042522	000051	
7389	047430	040522	045503	042440	EM103: .ASCIZ /RACK E49(A1) STUCK LOW/
7390	047436	034464	040450	024461	
7391	047444	051440	052524	045503	
7392	047452	046040	053517	000	
7393	047457	105	052111	042510	EM104: .ASCIZ /EITHER GRAJ DIV SUB L NOT GOING HIGH OR RACK E49(D0) STUCK LOW/
7394	047464	020122	051107	045101	
7395	047472	042040	053111	051440	
7396	047500	041125	046040	047040	
7397	047506	052117	043440	044517	
7398	047514	043516	044040	043511	
7399	047522	020110	051117	051040	
7400	047530	041501	020113	032105	
7401	047536	024071	030104	020051	
7402	047544	052123	041525	020113	
7403	047552	047514	000127		
7404	047556	052521	047524	042511	EM105: .ASCIZ /QUOTIENT & REMAINDER BAD/
7405	047564	052116	023040	051040	
7406	047572	046505	044501	042116	
7407	047600	051105	041040	042101	
7408	047606	000			
7409	047607	105	052111	042510	EM106: .ASCIZ /EITHER GRAH SR15 NOT GETTING TO RACK E64 OR E64(B0) STUCK HIGH/
7410	047614	020122	051107	044101	
7411	047622	051440	030522	020065	
7412	047630	047516	020124	042507	
7413	047636	052124	047111	020107	
7414	047644	047524	051040	041501	
7415	047652	020113	033105	020064	
7416	047660	051117	042440	032066	
7417	047666	041050	024460	051440	
7418	047674	052524	045503	044040	
7419	047702	043511	000110		
7420	047706	044505	044124	051105	EM110: .ASCIZ /EITHER IRCH N(1) NOT GETTING TO RACK E63 OR E63(D0) STUCK HIGH/
7421	047714	044440	041522	020110	
7422	047722	024116	024461	047040	
7423	047730	052117	043440	052105	
7424	047736	044524	043516	052040	
7425	047744	020117	040522	045503	
7426	047752	042440	031466	047440	
7427	047760	020122	033105	024063	
7428	047766	030104	020051	052123	
7429	047774	041525	020113	044510	
7430	050002	044107	000		
7431	050005	122	041501	020113	EM111: .ASCIZ /RACK E49(B1) STUCK HIGH/
7432	050012	032105	024071	030502	
7433	050020	020051	052123	041525	

7434	050026	020113	044510	044107	
7435	050034	000			
7436	050035	105	052111	042510	EM112: .ASCIZ /EITHER GRAJ DIV QUIT L NOT GOING HIGH OR NOT GETTING/<CRLF>
7437	050042	020122	051107	045101	
7438	050050	042040	053111	050440	
7439	050056	044525	020124	020114	
7440	050064	047516	020124	047507	
7441	050072	047111	020107	044510	
7442	050100	044107	047440	020122	
7443	050106	047516	020124	042507	
7444	050114	052124	047111	100107	
7445	050122	000			
7446	050123	124	020117	040522	.ASCIZ /TO RACK E63 OR E63(CO) STUCK LOW/
7447	050130	045503	042440	031466	
7448	050136	047440	020122	033105	
7449	050144	024063	030103	020051	
7450	050152	052123	041525	020113	
7451	050160	047514	000127		
7452	050164	040522	045503	042440	EM113: .ASCIZ /RACK E50(B0) STUCK LOW/
7453	050172	030065	041050	024460	
7454	050200	051440	052524	045503	
7455	050206	046040	053517	000	
7456	050213	122	041501	020113	EM114: .ASCIZ /RACK E64(B0) STUCK LOW/
7457	050220	033105	024064	030102	
7458	050226	020051	052123	041525	
7459	050234	020113	047514	000127	
7460	050242	044505	044124	051105	EM115: .ASCIZ /EITHER DVC.20,DVC.40,DVC.80,OR DVC.90 FAILED/
7461	050250	042040	041526	031056	
7462	050256	026060	053104	027103	
746	050264	030064	042054	041526	
7461	050272	034056	026060	051117	
7465	050300	042040	041526	034456	
7466	050306	020060	040506	046111	
7467	050314	042105	000		
7468	050317	105	052111	042510	EM117: .ASCIZ /EITHER BAD CC'S IN DVC.90 OR RACK E63(D0) STUCK LOW/
7469	050324	020122	040502	020104	
7470	050332	041503	051447	044440	
7471	050340	020116	053104	027103	
7472	050346	030071	047440	020122	
7473	050354	040522	045503	042440	
7474	050362	031466	042050	024460	
7475	050370	051440	052524	045503	
7476	050376	046040	053517	000	
7477	050403	105	052111	042510	EM120: .ASCIZ /EITHER GRAJ DIV QUIT DID NOT GO LOW OR RACK E63(CO) STUCK HIGH/
7478	050410	020122	051107	045101	
7479	050416	042040	053111	050440	
7480	050424	044525	020124	044504	
7481	050432	020104	047516	020124	
7482	050440	047507	046040	053517	
7483	050446	047440	020122	040522	
7484	050454	045503	042440	031466	
7485	050462	041450	024460	051440	
7486	050470	052524	045503	044040	
7487	050476	043511	000110		
7488	050502	041503	051447	041040	EM121: .ASCIZ /CC'S BAD DUE TO EITHER DIV.30 OR DVE.20/
7489	050510	042101	042040	042525	

7490	050516	052010	020117	044505	
7491	050524	044124	051105	042040	
7492	050532	053111	031456	020060	
7493	050540	051117	042040	042526	
7494	050546	031056	000060		
7495	050552	051107	045101	042440	EM122: .ASCIZ /GRAJ ES BAD(Z2(0)*LEFT SAVE(1))/
7496	050560	020065	040502	024104	
7497	050566	031132	030050	025051	
7498	050574	042514	052106	051440	
7499	050602	053101	024105	024461	
7500	050610	000051			
7501	050612	040522	045503	042440	EM123: .ASCIZ /RACK E63(00) STUCK LOW/
7502	050620	031466	042050	024460	
7503	050626	051440	052524	045503	
7504	050634	046040	053517	000	
7505	050641	103	023503	020123	EM124: .ASCIZ /CC'S DID NOT LOAD PROPERLY/
7506	050646	044504	020104	047516	
7507	050654	020124	047514	042101	
7508	050662	050040	047522	042520	
7509	050670	046122	000131		
7510	050674	044505	044124	051105	EM125: .ASCIZ /EITHER GRAJ ES(N(1)*SR15(1)) BAD OR ROM STATE BAD/
7511	050702	043440	040522	020112	
7512	050710	032505	047050	030450	
7513	050716	025051	051123	032461	
7514	050724	030450	024451	041040	
7515	050732	042101	047440	020122	
7516	050740	047522	020115	052123	
7517	050746	052101	020105	040502	
7518	050754	000104			
7519	050756	052521	052117	041040	EM127: .ASCIZ /QUOT BAD REMAINDER OK (ROM STATE FAILURE)/
7520	050764	042101	051040	046505	
7521	050772	044501	042116	051105	
7522	051000	047440	020113	051050	
7523	051006	046517	051440	040524	
7524	051014	042524	043040	044501	
7525	051022	052514	042522	000051	
7526	051030	052521	052117	042511	EM130: .ASCIZ /QUOTIENT BAD (ROM STATE FAILURE)/
7527	051036	052116	041040	042101	
7528	051044	024040	047522	020115	
7529	051052	052123	052101	020105	
7530	051060	040506	046111	051125	
7531	051066	024505	000		
7532	051071	102	042101	041440	EM134: .ASCIZ /BAD CC'S ON DIV OVERFLOW, STATE DVD.10/
7533	051076	023503	020123	047117	
7534	051104	042040	053111	047440	
7535	051112	042526	043122	047514	
7536	051120	026127	051440	040524	
7537	051126	042524	042040	042126	
7538	051134	030456	000060		
7539	051140	030122	041040	042101	EM135: .ASCIZ /RO BAD/
7540	051146	000			
7541	051147	123	040524	042524	EM136: .ASCIZ /STATE M17.00 DID NOT INC SP/
7542	051154	046440	050124	030056	
7543	051162	020060	044504	020104	
7544	051170	047516	020124	047111	
7545	051176	020103	050123	000	

7546	051203	122	041501	020106	EM140:	.ASCIZ	"RACF X/CLASS DOES NOT GO HIGH"
7547	051210	027530	046103	051501			
7548	051216	020123	047504	051505			
7549	051224	047040	052117	043440			
7550	051232	020117	044510	044107			
7551	051240	000					
7552	051241	115	050124	030456	EM141:	.ASCIZ	/MTP.10 FAILED TO RELOAD THE DR/
7553	051246	020060	040506	046111			
7554	051254	042105	052040	020117			
7555	051262	042522	047514	042101			
7556	051270	052040	042510	042040			
7557	051276	000122					
7558	051300	050123	046040	040517	EM142:	.ASCIZ	/SP LOADED INCORRECTLY/
7559	051306	042504	020104	047111			
7560	051314	047503	051122	041505			
7561	051322	046124	000131				
7562	051326	052115	027120	030061	EM143:	.ASCIZ	/MTP.10 DID NOT PUT PCB IN DR/
7563	051334	042040	042111	047040			
7564	051342	052117	050040	052125			
7565	051350	050040	041103	044440			
7566	051356	020116	051104	000			
7567	051363	122	041501	020106	EM144:	.ASCIZ	/RACF E20(4) STUCK HIGH/
7568	051370	031105	024060	024464			
7569	051376	051440	052524	045503			
7570	051404	044040	043511	000110			
7571	051412	043115	027120	030061	EM145:	.ASCIZ	/MFP.10 DID NOT DECREMENT THE SP/
7572	051420	042040	042111	047040			
7573	051426	052117	042040	041505			
7574	051434	042522	042515	052116			
7575	051442	052040	042510	051440			
7576	051450	000120					
7577	051452	030122	042040	042111	EM146:	.ASCIZ	/RD DID NOT GET PUT ON THE STACK/
7578	051460	047040	052117	043440			
7579	051466	052105	050040	052125			
7580	051474	047440	020116	044124			
7581	051502	020105	052123	041501			
7582	051510	000113					
7583	051512	040502	020104	041503	EM147:	.ASCIZ	/BAD CC'S, CC CONTROL ROM/
7584	051520	051447	020054	041503			
7585	051526	041440	047117	051124			
7586	051534	046117	051040	046517			
7587	051542	000					
7588	051543	115	050106	030056	EM151:	.ASCIZ	/MFP.00 BAD/
7589	051550	020060	040502	000104			
7590	051556	040502	020104	041503	EM152:	.ASCIZ	/BAD CC'S DUE TO MFP.00/
7591	051564	051447	042040	042525			
7592	051572	052040	020117	043115			
7593	051600	027120	030060	000			
7594	051605	124	050122	030056	EM155:	.ASCIZ	/TRP.01 FAILED TO LOAD BR/
7595	051612	020061	040506	046111			
7596	051620	042105	052040	020117			
7597	051626	047514	042101	041040			
7598	051634	000122					
7599	051636	044505	044124	051105	EM156:	.ASCII	/EITHER IRCD IOT DOES NOT GO LOW OR DAPE TV04 DOES<<CRLF>
7600	051644	044440	041522	020104			
7601	051652	047511	020124	047504			

7602	051660	051505	047040	052117				
7603	051666	043440	020117	047514				
7604	051674	020127	051117	042040				
7605	051702	050101	020105	053124				
7606	051710	032060	042040	042517				
7607	051716	100123						
7608	051720	047516	020124	047507		.ASCIZ	/NOT GO HIGH OR DOES NOT GET TO THE ALU/	
7609	051726	044040	043511	020110				
7610	051734	051117	042040	042517				
7611	051742	020123	047516	020124				
7612	051750	042507	020124	047524				
7613	051756	052040	042510	040440				
7614	051764	052514	000					
7615	051767	124	050122	030056	EM157:	.ASCIZ	/TRP.01 FAILED TO LOAD DR/	
7616	051774	020061	040506	046111				
7617	052002	042105	052040	020117				
7618	052010	047514	042101	042040				
7619	052016	000122						
7620	052020	051124	027120	030060	EM160:	.ASCIZ	/TRP.00 FAILED TO LOAD BR/	
7621	052026	043040	044501	042514				
7622	052034	020104	047524	046040				
7623	052042	040517	020104	051102				
7624	052050	000						
7625	052051	105	052111	042510	EM161:	.ASCII	/EITHER IRCD OPCODE3 DOES NOT GO LOW OR DOES NOT/<CRLF>	
7626	052056	020122	051111	042103				
7627	052064	047440	041520	042117				
7628	052072	031505	042040	042517				
7629	052100	020123	047516	020124				
7630	052106	047507	046040	053517				
7631	052114	047440	020122	047504				
7632	052122	051505	047040	052117				
7633	052130	200						
7634	052131	107	052105	052040		.ASCIZ	/GET THRU TO DAPE TV03/	
7635	052136	051110	020125	047524				
7636	052144	042040	050101	020105				
7637	052152	053124	031460	000				
7638	052157	124	050122	030056	EM162:	.ASCIZ	/TRP.00 FAILED TO LOAD DR/	
7639	052164	020060	040506	046111				
7640	052172	042105	052040	020117				
7641	052200	047514	042101	042040				
7642	052206	000122						
7643	052210	044502	020124	040506	EM163:	.ASCIZ	/BIT FAILED IN PIRQ REG/	
7644	052216	046111	042105	044440				
7645	052224	020116	044520	050522				
7646	052232	051040	043505	000				
7647	052237	105	051122	051117	DH163:	.ASCII	/ERRORPC PIRQ TST NUM/<CRLF>	
7648	052244	041520	020040	020040				
7649	052252	020040	050040	051111				
7650	052260	004521	020040	051524				
7651	052266	020124	052516	100115				
7652	052274	020011	054105	042520		.ASCIZ	/ EXPECT ACTUAL/	
7653	052302	052103	020040	041501				
7654	052310	052524	046101	000				
7655		052316				.EVEN		
7656	052316	001116	001162	001216	DT163:	.WORD	\$ERRPC,\$TMPO,\$EPIRQ,\$\$TSTNM,0	
7657	052324	001210	000000					

7658	052330	042506	027124	030060	EM164:	.ASCIZ /FET.00 HAD BAD BEN FIELD/
7659	052336	044040	042101	041040		
7660	052344	042101	041040	047105		
7661	052352	043040	042511	042114		
7662	052360	000				
7663	052361	105	052111	042510	EM165:	.ASCIZ /EITHER TMCB E62(1) BAD OR TMCB HONOR PIR 1 NOT GOING LOW/
7664	052366	020122	046524	041103		
7665	052374	042440	031066	030450		
7666	052402	020051	040502	020104		
7667	052410	051117	052040	041515		
7668	052416	020102	047510	047516		
7669	052424	020122	044520	020122		
7670	052432	020061	047516	020124		
7671	052440	047507	047111	020107		
7672	052446	047514	000127			
7673	052452	044505	044124	051105	EM166:	.ASCII /EITHER TMCB E51(9) OR E55(10,11) OR E62 BAD OR/<CRLF>
7674	052460	052040	041515	020102		
7675	052466	032505	024061	024471		
7676	052474	047440	020122	032505		
7677	052502	024065	030061	030454		
7678	052510	024461	047440	020122		
7679	052516	033105	020062	040502		
7680	052524	020104	051117	200		
7681	052531	124	041515	020101		.ASCIZ /TMCA INH BELOW BR6 STUCK LOW/
7682	052536	047111	020110	042502		
7683	052544	047514	020127	051102		
7684	052552	020066	052123	041525		
7685	052560	020113	047514	000127		
7686	052566	046524	040503	040440	EM167:	.ASCIZ /TMCA ABOVE BR7 MIGHT BE STUCK LOW/
7687	052574	047502	042526	041040		
7688	052602	033522	046440	043511		
7689	052610	052110	041040	020105		
7690	052616	052123	041525	020113		
7691	052624	047514	000127			
7692	052630	046524	042503	041040	EM170:	.ASCIZ /TMCE BRQ CLOCK MIGHT BE STUCK LOW/
7693	052636	050522	041440	047514		
7694	052644	045503	046440	043511		
7695	052652	052110	041040	020105		
7696	052660	052123	041525	020113		
7697	052666	047514	000127			
7698	052672	044505	044124	051105	EM171:	.ASCII /EITHER TMCB PF(0)*(SF+TF) NOT GOING HIGH OR NOT/<CRLF>
7699	052700	052040	041515	020102		
7700	052706	043120	030050	025051		
7701	052714	051450	025506	043124		
7702	052722	020051	047516	020124		
7703	052730	047507	047111	020107		
7704	052736	044510	044107	047440		
7705	052744	020122	047516	100124		
7706	052752	042507	052124	047111		.ASCIZ /GETTING TO RACK E50 OR RACK E50(A1) BAD/
7707	052760	020107	047524	051040		
7708	052766	041501	020113	032505		
7709	052774	020060	051117	051040		
7710	053002	041501	020113	032505		
7711	053010	024060	030501	020051		
7712	053016	040502	000104			
7713	053022	044505	044124	051105	EM172:	.ASCII /EITHER TMCB PF(0)*(SF+-TF) NOT GOING LOW OR/<CRLF>

7714	053030	052040	041515	020102	
7715	053036	043120	030050	025051	
7716	053044	051450	025506	052055	
7717	053052	024506	047040	052117	
7718	053060	043440	044517	043516	
7719	053066	046040	053517	047440	
7720	053074	100122			
7721	053076	047516	020124	042507	.ASCII /NOT GETTING TO RACK E64 OR RACK E64(A1) BAD/<<CRLF>
7722	053104	052124	047111	020107	
7723	053112	047524	051040	041501	
7724	053120	020113	033105	020064	
7725	053126	051117	051040	041501	
7726	053134	020113	033105	024064	
7727	053142	030501	020051	040502	
7728	053150	100104			
7729	053152	051117	052040	041515	.ASCII /OR TMCB PIRQ NOT GETTING TO DAPE OR DAPE TV05*07/<<CRLF>
7730	053160	020102	044520	050522	
7731	053166	047040	062117	043440	
7732	053174	052105	044524	043516	
7733	053202	052040	020117	040504	
7734	053210	042520	047440	020122	
7735	053216	040504	042520	052040	
7736	053224	030126	025065	033460	
7737	053232	200			
7738	053233	116	052117	043440	.ASCIZ /NOT GOING HIGH OR NOT GETTING TO THE ALU/
7739	053240	044517	043516	044040	
7740	053246	043511	020110	051117	
7741	053254	047040	052117	043440	
7742	053262	052105	044524	043516	
7743	053270	052040	020117	044124	
7744	053276	020105	046101	000125	
7745	053304	044505	044124	051105	EM174: .ASCIZ /EITHER TMCB E62(2) BAD OR TMCB HONOR PIR 2 NOT GOING LOW/
7746	053312	052040	041515	020102	
7747	053320	033105	024062	024462	
7748	053326	041040	042101	047440	
7749	053334	020122	046524	041103	
7750	053342	044040	047117	051117	
7751	053350	050040	051111	031040	
7752	053356	047040	052117	043440	
7753	053364	044517	043516	046040	
7754	053372	053517	000		
7755	053375	124	041515	020102	EM175: .ASCIZ /TMCB E63 BAD/
7756	053402	033105	020063	040502	
7757	053410	000104			
7758	053412	042514	042526	020114	EM176: .ASCIZ /LEVEL 2 INTERRUPT WHEN LEVEL 1 ON/
7759	053420	020062	047111	042524	
7760	053426	051122	050125	020124	
7761	053434	044127	047105	046040	
7762	053442	053105	046105	030440	
7763	053450	047440	000116		
7764	053454	044505	044124	051105	EM177: .ASCIZ /EITHER TMCB E62(3) BAD OR TMCB HONOR PIR 3 NOT GOING LOW/
7765	053462	052040	041515	020102	
7766	053470	033105	024062	024463	
7767	053476	041040	042101	047440	
7768	053504	020122	046524	041103	
7769	053512	044040	047117	051117	

7770	053520	050040	051111	031440	
7771	053526	047040	052117	043440	
7772	053534	044517	043516	046040	
7773	053542	053517	000		
7774	053545	114	053105	046105	EM201: .ASCIZ /LEVEL 2 INTERRUPT WHEN CPU LEVEL 2 ON/
7775	053552	031040	044440	052116	
7776	053560	051105	052522	052120	
7777	053566	053440	042510	020116	
7778	053574	050103	020125	042514	
7779	053602	042526	020114	020062	
7780	053610	047117	000		
7781	053613	105	051122	051117	DH201: .ASCIZ /ERRORPC PIRQ TST NUM/
7782	053620	041520	020040	044520	
7783	053626	050522	020040	020040	
7784	053634	051524	020124	052516	
7785	053542	000115			
7786					
7787	053644	001116	001216	001210	DT201: .EVEN .WORD SERRPC,SEPIRQ,\$\$TSTNM,0
7788	053652	000000			
7789	053654	044505	044124	051105	EM202: .ASCIZ /EITHER TMCB E62(5) BAD OR TMCA HONOR PIR 4 NOT GOING LOW/
7790	053662	052040	041515	020102	
7791	053670	033105	024062	024465	
7792	053676	041040	042101	047440	
7793	053704	020122	046524	040503	
7794	053712	044040	047117	051117	
7795	053720	050040	051111	032040	
7796	053726	047040	052117	043440	
7797	053734	044517	043516	046040	
7798	053742	053517	000		
7799	053745	114	053105	046105	EM204: .ASCIZ /LEVEL 3 INTERRUPT WHEN CPU LEVEL 3 ON/
7800	053752	031440	044440	052116	
7801	053760	051105	052522	052120	
7802	053766	053440	042510	020116	
7803	053774	050103	020125	042514	
7804	054002	042526	020114	020063	
7805	054010	047117	000		
7806	054013	105	052111	042510	EM205: .ASCIZ /EITHER TMCB E62(11) BAD OR TMCA HONOR PIR 5 NOT GOING LOW/
7807	054020	020122	046524	041103	
7808	054026	042440	031066	030450	
7809	054034	024461	041040	042101	
7810	054042	047440	020122	046524	
7811	054050	040503	044040	047117	
7812	054056	051117	050040	051111	
7813	054064	032440	047040	052117	
7814	054072	043440	044517	043516	
7815	054100	046040	053517	000	
7816	054105	114	053105	046105	EM207: .ASCIZ /LEVEL 4 INTERRUPT WHEN CPU LEVEL 4 ON/
7817	054112	032040	044440	052116	
7818	054120	051105	052522	052120	
7819	054126	053440	042510	020116	
7820	054134	050103	020125	042514	
7821	054142	042526	020114	020064	
7822	054150	047117	000		
7823	054153	105	052111	042510	EM210: .ASCIZ /EITHER TMCB E51(11) BAD OR TMCB E55(8-9) BAD/
7824	054160	020122	046524	041103	
7825	054166	042440	030465	030450	

7826	054174	024461	041040	042101	
7827	054202	047440	020122	046524	
7828	054210	041103	042440	032465	
7829	054216	034050	034455	020051	
7830	054224	040502	000104		
7831	054230	044505	044124	051105	EM211: .ASCIZ /EITHER TMCB E70(1) BAD OR TMCA HONOR PIR6 NOT GOING LOW/
7832	054236	052040	041515	020102	
7833	054244	033505	024060	024461	
7834	054252	041040	042101	047440	
7835	054260	020122	046524	040503	
7836	054266	044040	047117	051117	
7837	054274	050040	051111	C?0066	
7838	054302	047516	020124	047507	
7839	054310	047111	020107	047514	
7840	054316	000127			
7841	054320	044505	044124	051105	EM212: .ASCIZ /EITHER TMCB E63(12) BAD OR TMCB E61(1) BAD/
7842	054326	052040	041515	020102	
7843	054334	033105	024063	031061	
7844	054342	020051	040502	020104	
7845	054350	051117	052040	041515	
7846	054356	020102	033105	024061	
7847	054364	024461	041040	042101	
7848	054372	000			
7849	054373	114	053105	046105	EM213: .ASCIZ /LEVEL 5 INTERRUPT WHEN CPU LEVEL 5 ON/
7850	054400	032440	044440	052116	
7851	054406	051105	052522	052120	
7852	054414	053440	042510	020116	
7853	054422	050103	020125	042514	
7854	054430	042526	020114	020065	
7855	054436	047117	000		
7856	054441	105	052111	042510	EM214: .ASCIZ /EITHER TMCB E70(6) BAD OR TMCA HONOR PIR 7 NOT GOING LOW/
7857	054446	020122	046524	041103	
7858	054454	042440	030067	033050	
7859	054462	020051	040502	020104	
7860	054470	051117	052040	041515	
7861	054476	020101	047510	047516	
7862	054504	020122	044520	020122	
7863	054512	020067	047516	020124	
7864	054520	047507	047111	020107	
7865	054526	047514	000127		
7866	054532	042514	042526	020114	EM216: .ASCIZ /LEVEL 6 INTERRUPT WHEN CPU LEVEL 6 ON/
7867	054540	020066	047111	042524	
7868	054546	051122	050125	020124	
7869	054554	044127	047105	041440	
7870	054562	052520	046040	053105	
7871	054570	046105	033040	047440	
7872	054576	000116			
7873	054600	044524	042515	052517	EM217: .ASCIZ /TIMEOUT 0 ATI DID NOT WORK/
7874	054606	020124	047117	042040	
7875	054614	052101	020111	044504	
7876	054622	020104	047516	020124	
7877	054630	047527	045522	000	
7878	054635	105	052111	042510	EM220: .ASCII /EITHER TMCC AERF(1) L NOT GOING LOW/<CRLF>
7879	054642	020122	046524	041503	
7880	054650	040440	051105	024106	
7881	054656	024461	046040	047040	

7882	054664	052117	043440	044517	
7883	054672	043516	046040	053517	
7884	054700	200			
7885	054701	117	020122	046524	.ASCIZ /OR TMCB E53(11) BAD/
7886	054706	041103	042440	031465	
7887	054714	030450	024461	041040	
7888	054722	042101	000		
7889	054725	124	046511	047505	EM221: .ASCIZ /TIMEOUT ON DATO DID NOT WORK/
7890	054732	052125	047440	020116	
7891	054740	040504	047524	042040	
7892	054746	042111	047040	052117	
7893	054754	053440	051117	000113	
7894	054762	044505	044124	051105	EM222: .ASCIZ /EITHER TMCB PS07(0) NOT GETTING TO TMCB E77 OR E77 BAD/
7895	054770	052040	041515	020102	
7896	054776	051520	033460	030050	
7897	055004	020051	047516	020124	
7898	055012	042507	052124	047111	
7899	055020	020107	047524	052040	
7900	055026	041515	020102	033505	
7901	055034	020067	051117	042440	
7902	055042	033467	041040	042101	
7903	055050	000			
7904	055051	102	032122	023040	EM223: .ASCIZ /BR4 & BR6 FAILED/
7905	055056	041040	033122	043040	
7906	055064	044501	042514	000104	
7907	055072	051102	020064	040506	EM224: .ASCII /BR4 FAILED. EITHER TMCB HONOR BR4 NOT GOING LOW OR/<CRLF>
7908	055100	046111	042105	020056	
7909	055106	044505	044124	051105	
7910	055114	052040	041515	020102	
7911	055122	047510	047516	020122	
7912	055130	051102	020064	047516	
7913	055136	020124	047507	047111	
7914	055144	020107	047514	020127	
7915	055152	051117	200		
7916	055155	124	041515	020102	.ASCIZ /TMCB E45(4) BAD OR INTERRUPT OR BG LOGIC ON UE^ BAD/
7917	055162	032105	024065	024464	
7918	055170	041040	042101	047440	
7919	055176	020122	047111	042524	
7920	055204	051122	050125	020124	
7921	055212	051117	041040	020107	
7922	055220	047514	044507	020103	
7923	055226	047117	052440	041502	
7924	055234	041040	042101	000	
7925	055241	102	032122	043040	EM225: .ASCII /BR4 FAILED BR6 OK EITHER TMCB HONOR BR4 NOT GOING/<CRLF>
7926	055246	044501	042514	020104	
7927	055254	051102	020066	045517	
7928	055262	042440	052111	042510	
7929	055270	020122	046524	041103	
7930	055276	044040	047117	051117	
7931	055304	041040	032122	047040	
7932	055312	052117	043440	044517	
7933	055320	043516	200		
7934	055323	114	053517	047440	.ASCIZ /LOW OR TMCB E62(4) BAD/
7935	055330	020122	046524	041103	
7936	055336	042440	031066	032050	
7937	055344	020051	040502	000104	

M14

7938	055352	044505	044124	051105	EM226: .ASCIZ /EITHER TMCA HONOR BR5 NOT GOING LOW OR TMCB E62(6) BAD/
7939	055360	052040	041515	020101	
7940	055366	047510	047516	020122	
7941	055374	051102	020065	047516	
7942	055402	020124	047507	047111	
7943	055410	020107	047514	020127	
7944	055416	051117	052040	041515	
7945	055424	020102	033105	024062	
7946	055432	024466	041040	042101	
7947	055440	000			
7948	055441	105	052111	042510	EM227: .ASCIZ /EITHER TMCA HONOR BR6 NOT GOING LOW OR TMCB E62(12) BAD/
7949	055446	020122	046524	040503	
7950	055454	044040	047117	051117	
7951	055462	041040	033122	047040	
7952	055470	052117	043440	044517	
7953	055476	043516	046040	053517	
7954	055504	047440	020122	046524	
7955	055512	041103	042440	031066	
7956	055520	030450	024462	041040	
7957	055526	042101	000		
7958	055531	131	046105	055040	EM230: .ASCII /YEL ZONE FAILED EITHER TMCD SL YEL NOT GOING HIGH OR/<CRLF>
7959	055536	047117	020105	040506	
7960	055544	046111	042105	042440	
7961	055552	052111	042510	020122	
7962	055560	046524	042103	051440	
7963	055566	020114	042531	020114	
7964	055574	047516	020124	047507	
7965	055602	047111	020107	044510	
7966	055610	044107	047440	100122	
7967	055616	051117	052040	041515	.ASCII /OR TMCA HONOR SLY NOT GOING LOW OR TMCB E70(3) BAD/<CRLF>
7968	055624	020101	047510	047516	
7969	055632	020122	046123	020131	
7970	055640	047516	020124	047507	
7971	055646	047111	020107	047514	
7972	055654	020127	051117	052040	
7973	055662	041515	020102	033505	
7974	055670	024060	024463	041040	
7975	055676	042101	200		
7976	055701	117	020122	042502	.ASCII /OR BEN13 FAILED- EITHER TMCA HONOR SLY NOT GETTING/<CRLF>
7977	055706	030516	020063	040506	
7978	055714	046111	042105	020055	
7979	055722	044505	044124	051105	
7980	055730	052040	041515	020101	
7981	055736	047510	047516	020122	
7982	055744	046123	020131	047516	
7983	055752	020124	042507	052124	
7984	055760	047111	100107		
7985	055764	047524	052040	041515	.ASCIZ /TO TMCB E53 OR E53(3) BAD/
7986	055772	020102	032505	020063	
7987	056000	051117	042440	031465	
7988	056006	031450	020051	040502	
7989	056014	000104			
7990	056016	044505	044124	051105	EM232: .ASCIZ /EITHER TMCC E16(8) NOT GOING LOW OR TMCC E36 BAD/
7991	056024	052040	041515	020103	
7992	056032	030505	024066	024470	
7993	056040	047040	052117	043440	

7994	056046	044517	043516	046040	
7995	056054	053517	047440	020122	
7996	056062	046524	041503	042440	
7997	056070	033063	041040	042101	
7998	056076	000			
7999	056077	105	052111	042510	EM233: .ASCII /EITHER PDRC STACK LIMIT NOT GETTING TO TMCD AS A LOW OR/<CRLF>
8000	056104	020122	042120	041522	
8001	056112	051440	040524	045503	
8002	056120	046040	046511	052111	
8003	056126	047040	052117	043440	
8004	056134	052105	044524	043516	
8005	056142	052040	020117	046524	
8006	056150	042103	040440	020123	
8007	056156	020101	047514	020127	
8008	056164	051117	200		
8009	056167	124	041515	020104	.ASCIZ /TMCD E8 BAD/
8010	056174	034105	041040	042101	
8011	056202	000			
8012	056203	105	052111	042510	EM234: .ASCII /EITHER UBCC DATI NOT GETTING TO TMCC AS A LOW OR EITHER/<CRLF>
8013	056210	020122	041125	041503	
8014	056216	042040	052101	020111	
8015	056224	047516	020124	042507	
8016	056232	052124	047111	020107	
8017	056240	047524	052040	041515	
8018	056246	020103	051501	040440	
8019	056254	046040	053517	047440	
8020	056262	020122	044505	044124	
8021	056270	051105	200		
8022	056273	124	041515	020103	.ASCIZ /TMCC E30 OR E36 BAD/
8023	056300	031505	020060	051117	
8024	056306	042440	033063	041040	
8025	056314	042101	000		
8026	056317	105	052111	042510	EM235: .ASCII /EITHER TMCD SL RED NOT GOING LOW OR TMCC ABORT/<CRLF>
8027	056324	020122	046524	042103	
8028	056332	051440	020114	042522	
8029	056340	020104	047516	020124	
8030	056346	047507	047111	020107	
8031	056354	047514	020127	051117	
8032	056362	052040	041515	020103	
8033	056370	041101	051117	100124	
8034	056376	047516	020124	047507	.ASCII /NOT GOING LOW /
8035	056404	047111	020107	047514	
8036	056412	020127			
8037	056414	051117	052040	041515	.ASCIZ /OR TMCB E50(6) DID NOT GO HIGH ON TMCC SERF(1)L/
8038	056422	020102	032505	024060	
8039	056430	024466	042040	042111	
8040	056436	047040	052117	043440	
8041	056444	020117	044510	044107	
8042	056452	047440	020116	046524	
8043	056460	041503	051440	051105	
8044	056466	024106	024461	000114	
8045	056474	044505	044124	051105	EM237: .ASCII /EITHER TMCC SERF(1) NOT GOING LOW OR/<CRLF>
8046	056502	052040	041515	020103	
8047	056510	042523	043122	030450	
8048	056516	020051	047516	020124	
8049	056524	047507	047111	020107	

8050	056532	047514	020127	051117	
8051	056540	200			
8052	056541	116	052117	043440	.ASCIZ /NOT GETTING TO TMCB E50(2&1)/
8053	056546	052105	044524	043516	
8054	056554	052040	020117	046524	
8055	056562	041103	042440	030065	
8056	056570	031050	030446	000051	
8057	056576	044505	044124	051105	EM240: .ASCII /EITHER TMCB PF(0)*(SF+-TF) NOT GOING HIGH OR/<CRLF>
8058	056604	052040	041515	020102	
8059	056612	043120	030050	025051	
8060	056620	051450	025506	052055	
8061	056626	024506	047040	052117	
8062	056634	043440	044517	043516	
8063	056642	044040	043511	020110	
8064	056650	051117	200		
8065	056653	116	052117	043440	.ASCIZ /NOT GETTING TO RACK BRCAB04/
8066	056660	052105	044524	043516	
8067	056666	052040	020117	040522	
8068	056674	045503	041040	041522	
8069	056702	041101	032060	000	
8070	056707	105	052111	042510	EM241: .ASCII /EITHER SCCE STACK OVERFLOW NOT GOING HIGH OR/<CRLF>
8071	056714	020122	041523	042503	
8072	056722	051440	040524	045503	
8073	056730	047440	042526	043122	
8074	056736	047514	020127	047516	
8075	056744	020124	047507	047111	
8076	056752	020107	044510	044107	
8077	056760	047440	100122		
8078	056764	047516	020124	042507	.ASCIZ /NOT GETTING TO TMCD E31 OR E31 BAD/
8079	056772	052124	047111	020107	
8080	057000	047524	052040	041515	
8081	057006	020104	031505	020061	
8082	057014	051117	042440	030463	
8083	057022	041040	042101	000	
8084	057027	105	052111	042510	EM242: .ASCII /EITHER PDRC RED ZONE NOT GOING HIGH OR/<CRLF>
8085	057034	020122	042120	041522	
8086	057042	051040	042105	055040	
8087	057050	047117	020105	047516	
8088	057056	020124	047507	047111	
8089	057064	020107	044510	044107	
8090	057072	047440	100122		
8091	057076	047516	020124	042507	.ASCIZ /NOT GETTING TO TMCD E31 OR TMCD E31 BAD OR SL REG BIT 0 BAD/
8092	057104	052124	047111	020107	
8093	057112	047524	052040	041515	
8094	057120	020104	031505	020061	
8095	057126	051117	052040	041515	
8096	057134	020104	031505	020061	
8097	057142	040502	020104	051117	
8098	057150	051440	020114	042522	
8099	057156	020107	044502	020124	
8100	057164	020060	040502	000104	
8101	057172	031065	030064	020060	EM243: .ASCIZ /52400 PATTERN FAILED, 125000 PATTERN OK/
8102	057200	040520	052124	051105	
8103	057206	020116	040506	046111	
8104	057214	042105	020054	031061	
8105	057222	030065	030060	050040	

8106	057230	052101	042524	047122					
8107	057236	047440	000113						
8108	057242	051105	047522	050122	DH243:	.ASCII	/ERRORPC	SL REG	TST NUM/<CRLF>
8109	057250	020103	020040	020040					
8110	057256	051440	020114	042522					
8111	057264	020107	020040	020040					
8112	057272	052040	052123	047040					
8113	057300	046525	200						
8114	057303	011	042440	050130		.ASCIZ	/	EXPECT	ACTUAL/
8115	057310	041505	020124	040440					
8116	057316	052103	040525	000114					
8117						.EVEN			
8118	057324	001116	001162	001222	DT243:	.WORD	SERRPC, \$TMPO, E2STKLM, \$STSTNM, 0		
8119	057332	001210	000000						
8120	057336	031061	030065	030060	EM244:	.ASCIZ	/125000 PATTERN FAILED 52400 PATTERN OK/		
8121	057344	050040	052101	042524					
8122	057352	047122	043040	044501					
8123	057360	042514	020104	031065					
8124	057366	030064	020060	040520					
8125	057374	052124	051105	020116					
8126	057402	045517	000						
8127		057406				.EVEN			
8128	057406	001116	001162	001220	DT244:	.WORD	SERRPC, \$TMPO, E1STKLM, \$STSTNM, 0		
8129	057414	001210	000000						
8130	057420	044505	044124	051105	EM245:	.ASCII	/EITHER SCCE SL ADDRESS NOT GETTING TO TMCD OR/<CRLF>		
8131	057426	051440	041503	020105					
8132	057434	046123	040440	042104					
8133	057442	042522	051523	047040					
8134	057450	052117	043440	052105					
8135	057456	044524	043516	052040					
8136	057464	020117	046524	042103					
8137	057472	047440	100122						
8138	057476	046524	042103	042440		.ASCIZ	/TMCD E28 OR E14 BAD/		
8139	057504	034062	047440	020122					
8140	057512	030505	020064	040502					
8141	057520	000104							
8142	057522	044505	044124	051105	EM246:	.ASCII	/EITHER TMCD LOW BYTE EN DOES NOT GO LOW OR/<CRLF>		
8143	057530	052040	041515	020104					
8144	057536	047514	020127	054502					
8145	057544	042524	042440	020116					
8146	057552	047504	051505	047040					
8147	057560	052117	043440	020117					
8148	057566	047514	020127	051117					
8149	057574	200							
8150	057575	116	052117	043440		.ASCIZ	/NOT GETTING THRU TO THE DMUX (PDRE) AS A LOW /		
8151	057602	052105	044524	043516					
8152	057610	052040	051110	020125					
8153	057616	047524	052040	042510					
8154	057624	042040	052515	020130					
8155	057632	050050	051104	024505					
8156	057640	040440	020123	020101					
8157	057646	047514	020127	000					
8158	057653	105	052111	042510	EM247:	.ASCII	/EITHER TMCD DMX S1 STUCK HIGH OR IT DOES NOT/<CRLF>		
8159	057660	020122	046524	042103					
8160	057666	042040	054115	051440					
8161	057674	020061	052123	041525					

8162	057702	020113	044510	044107	
8163	057710	047440	020122	052111	
8164	057716	042040	042517	020123	
8165	057724	047516	100124		
8166	057730	042507	020124	044124	.ASCIZ /GET THRU TO THE DMUX(PDRE) AS A LOW/
8167	057736	052522	052040	020117	
8168	057744	044124	020105	046504	
8169	057752	054125	050050	051104	
8170	057760	024505	040440	020123	
8171	057766	020101	047514	000127	
8172	057774	047502	044124	050040	EM250: .ASCIZ /BOTH PATTERNS FAILED/
8173	060002	052101	042524	047122	
8174	060010	020123	040506	046111	
8175	060016	042105	000		
8176	060021	105	051122	051117	DH250: .ASCII /ERRORPC SL REG SL REG TST NUM/<CRLF>
8177	060026	041520	020040	020040	
8178	060034	020040	046123	051040	
8179	060042	043505	020011	020040	
8180	060050	020040	051440	020114	
8181	060056	042522	020107	020040	
8182	060064	020040	051524	020124	
8183	060072	052516	100115		
8184	060076	020011	054105	042520	.ASCIZ / EXPECT ACTUAL EXPECT ACTUAL/
8185	060104	052103	020040	041501	
8186	060112	052524	046101	020040	
8187	060120	054105	042520	052103	
8188	060126	020040	041501	052524	
8189	060134	046101	000		
8190		060140			
8191	060140	001116	001162	001220	DT250: .EVEN \$ERRPC,\$TMPO,E1STKLM,\$TMP1,E2STKLM,\$STSTNM,0
8192	060146	001164	001222	001210	
8193	060154	000000			
8194	060156	044505	044124	051105	EM251: .ASCII /EITHER TMCD YEL ZONE DID NOT GO LOW OR IT DID NOT GET THRU TO E31/<CRLF>
8195	060164	052040	041515	020104	
8196	060172	042531	020114	047532	
8197	060200	042516	042040	042111	
8198	060206	047040	052117	043440	
8199	060214	020117	047514	020127	
8200	060222	051117	044440	020124	
8201	060230	044504	020104	047516	
8202	060236	020124	042507	020124	
8203	060244	044124	052522	052040	
8204	060252	020117	031505	100061	
8205	060260	051117	052040	041515	.ASCIZ /OR TMCE CACHE BEND DID NOT GO HIGH ON SL RED/
8206	060266	020105	040503	044103	
8207	060274	020105	042502	042116	
8208	060302	042040	042111	047040	
8209	060310	052117	043440	020117	
8210	060316	044510	044107	047440	
8211	060324	020116	046123	051040	
8212	060332	042105	000		
8213	060335	124	041515	020104	EM252: .ASCIZ /TMCD YEL ZONE DOES NOT GO LOW ON ADR 240/
8214	060342	042531	020114	047532	
8215	060350	042516	042040	042517	
8216	060356	020123	047516	020124	
8217	060364	047507	046040	053517	

E15

8218	060372	047440	020116	042101	
8219	060400	020122	032062	000060	
8220	060406	046524	042103	054440	EM253: .ASCIZ /TMCD YEL ZONE DOES NOT GO LOW ON ADR 140/
8221	060414	046105	055040	047117	
8222	060422	020105	047504	051505	
8223	060430	047040	052117	043440	
8224	060436	020117	047514	020127	
8225	060444	047117	040440	051104	
8226	060452	030440	030064	000	
8227	060457	105	052111	042510	EM254: .ASCIZ /EITHER TMCD YEL ZONE DOES NOT GO HIGH OR DID NOT GET THRU TO E31/
8228	060464	020122	046524	042103	
8229	060472	054440	046105	055040	
8230	060500	047117	020105	047504	
8231	060506	051505	047040	052117	
8232	060514	043440	020117	044510	
8233	060522	044107	047440	020122	
8234	060530	044504	020104	047516	
8235	060536	020124	042507	020124	
8236	060544	044124	052522	052040	
8237	060552	020117	031505	000061	
8238	060560	047125	041111	051525	EM255: .ASCIZ /UNIBUS TIMEOUT BIT IN CPU ERROR REG DID NOT SET/
8239	060566	052040	046511	047505	
8240	060574	052125	041040	052111	
8241	060602	044440	020116	050103	
8242	060610	020125	051105	047522	
8243	060616	020122	042522	020107	
8244	060624	044504	020104	047516	
8245	060632	020124	042523	000124	
8246	060640	051105	047522	050122	DH255: .ASCII /ERRORPC CPUERR REG TST NUM/<CRLF>
8247	060646	020103	020040	041440	
8248	060654	052520	051105	020122	
8249	060662	042522	020107	020040	
8250	060670	051524	020124	052516	
8251	060676	100115			
8252	060700	020011	054105	042520	.ASCIZ / EXPECT ACTUAL/
8253	060706	052103	020040	041501	
8254	060714	052524	046101	000	
8255	060721	103	052520	042440	EM256: .ASCIZ /CPU ERRPROR REG DID NOT CLEAR/
8256	060726	051122	051120	051117	
8257	060734	051040	043505	042040	
8258	060742	042111	047040	052117	
8259	060750	041440	042514	051101	
8260	060756	000			
8261	060757	131	046105	055040	EM257: .ASCIZ /YEL ZONE BIT IN CPU ERROR REG DID NOT SET/
8262	060764	047117	020105	044502	
8263	060772	020124	047111	041440	
8264	061000	052520	042440	051122	
8265	061006	051117	051040	043505	
8266	061014	042040	042111	047040	
8267	061022	052117	051440	052105	
8268	061030	000			
8269	061031	105	052111	042510	EM260: .ASCIZ /EITHER TMCD E4(4) NOT GOING LOW OR E4 BAD/
8270	061036	020122	046524	042103	
8271	061044	042440	024064	024464	
8272	061052	047040	052117	043440	
8273	061060	044517	043516	046040	

8274	061066	053517	047440	020122	
8275	061074	032105	041040	042101	
8276	061102	000			
8277	061103	122	040505	020104	EM261: .ASCIZ /READ ZONE BIT IN CPU ERROR REG DID NOT SET/
8278	061110	047532	042516	041040	
8279	061116	052111	044440	020116	
8280	061124	050103	020125	051105	
8281	061132	047522	020122	042522	
8282	061140	020107	044504	020104	
8283	061146	047516	020124	042523	
8284	061154	000124			
8285	061156	044505	044124	051105	EM262: .ASCIZ /EITHER TMCD E15(6) OR E18(14) OR E18 BAD/
8286	061164	052040	041515	020104	
8287	061172	030505	024065	024466	
8288	061200	047440	020122	030505	
8289	061206	024070	032061	020051	
8290	061214	051117	042440	034061	
8291	061222	041040	042101	000	
8292	061227	106	046117	047514	EM263: .ASCII /FOLLOWING IS A LIST OF THE STACK LIMIT REG/<CRLF>
8293	061234	044527	043516	044440	
8294	061242	020123	020101	044514	
8295	061250	052123	047440	020106	
8296	061256	044124	020105	052123	
8297	061264	041501	020113	044514	
8298	061272	044515	020124	042522	
8299	061300	100107			
8300	061302	020046	050123	053040	.ASCII /8 SP VALUES THAT CAUSED AN ERROR. THEY ARE/<CRLF>
8301	061310	046101	042525	020123	
8302	061316	044124	052101	041440	
8303	061324	052501	042523	020104	
8304	061332	047101	042440	051122	
8305	061340	051117	020056	044124	
8306	061346	054505	040440	042522	
8307	061354	200			
8308	061355	107	047522	050125	.ASCIZ /GROUPED ACCORDING TO ERROR TYPES/
8309	061362	042105	040440	041503	
8310	061370	051117	044504	043516	
8311	061376	052040	020117	051105	
8312	061404	047522	020122	054524	
8313	061412	042520	000123		
8314	061416	051105	047522	050122	DH263: .ASCIZ /ERRORPC TEST NUMBER/<CRLF>
8315	061424	020103	042524	052123	
8316	061432	047040	046525	042502	
8317	061440	100122	000		
8318	061443	011	052123	041501	DH263A: .ASCII /
8319	061450	020113	044514	044515	STACK LIMIT REGISTER
8320	061456	020124	042522	044507	STACK POINTER/<CRLF>
8321	061464	052123	051105	004411	
8322	061472	020040	020040	020040	
8323	061500	051440	040524	045503	
8324	061506	050040	044517	052116	
8325	061514	051105	200		
8326	061517	117	052103	046101	.ASCII /OCTAL 15 14 13 12 11 10 9 8 OCTAL/
8327	061524	020040	020040	032461	
8328	061532	030440	020064	031461	
8329	061540	030440	020062	030461	

8330	061546	030440	020060	034440	
8331	061554	020040	020070	020040	
8332	061562	020040	047440	052103	
8333	061570	046101			
8334	061572	020040	020040	032461	.ASCIZ / 15 14 13 12 11 10 9 8/<CRLF>
8335	061600	030440	020064	031461	
8336	061606	030440	020062	030461	
8337	061614	030440	020060	034440	
8338	061622	020040	100070	000	
8339	061627	122	042105	052040	DH263B: .ASCIZ /RED TRAP ON YEL ADR/
8340	061634	040522	020120	047117	
8341	061642	054440	046105	040440	
8342	061650	051104	000		
8343	061653	122	042105	052040	DH263C: .ASCIZ /RED TRAP ON LEGAL ADR/
8344	061660	040522	020120	047117	
8345	061666	046040	043505	046101	
8346	061674	040440	051104	000	
8347	061701	131	046105	047514	DH263D: .ASCIZ /YELLOW TRAP ON RED ADR/
8348	061706	020127	051124	050101	
8349	061714	047440	020116	042522	
8350	061722	020104	042101	000122	
8351	061730	042531	020114	051124	DH263E: .ASCIZ /YEL TRAP ON LEGAL ADR/
8352	061736	050101	047440	020116	
8353	061744	042514	040507	020114	
8354	061752	042101	000122		
8355	061756	047516	052040	040522	DH263F: .ASCIZ /NO TRAP ON RED ADR/
8356	061764	020120	047117	051040	
8357	061772	042105	040440	051104	
8358	062000	000			
8359	062001	116	020117	051124	DH263G: .ASCIZ /NO TRAP ON YEL ADR/
8360	062006	050101	047440	020116	
8361	062014	042531	020114	042101	
8362	062022	000122			
8363					.EVEN
8364	062024	061627			INDEX: DH263B
8365	062026	061653			DH263C
8366	062030	061701			DH263D
8367	062032	061730			DH263E
8368	062034	061756			DH263F
8369	062036	062001			DH263G
8370	062040	047507	047111	020107	EM264: .ASCIZ /GOING TO NEXT TEST/
8371	062046	047524	047040	054105	
8372	062054	020124	042524	052123	
8373	062062	000			
8374					
8375	062063	116	052117	035105	EM265: .ASCII /NOTE: IF NONE OF THE ODD ADR ERRORS TRAP/<CRLF>
8376	062070	044440	020106	047516	
8377	062076	042516	047440	020106	
8378	062104	044124	020105	042117	
8379	062112	020104	042101	020122	
8380	062120	051105	047522	051522	
8381	062126	052040	040522	100120	
8382	062134	020040	020040	020040	.ASCII / THEN EITHER TMCC ODD ADRS ERR NOT GETTING TO/<CRLF>
8383	062142	044124	047105	042440	
8384	062150	052111	042510	020122	
8385	062156	046524	041503	047440	

8386	062164	042104	040440	051104	
8387	062172	020123	051105	020122	
8388	062200	047516	020124	042507	
8389	062206	052124	047111	020107	
8390	062214	047524	200		
8391	062217	040	020040	020040	.ASCII / TMCC BUS ERROR OR DAPB BAMX00 NOT/<CRLF>
8392	062224	052040	041515	020103	
8393	062232	052502	020123	051105	
8394	062240	047522	020122	051117	
8395	062246	042040	050101	020102	
8396	062254	040502	054115	030060	
8397	062262	047040	052117	200	
8398	062267	040	020040	020040	.ASCII / GETTING TO TMCC E7 AS A HIGH/<CRLF><CRLF>
8399	062274	043440	052105	044524	
8400	062302	043516	052040	020117	
8401	062310	046524	041503	042440	
8402	062316	020067	051301	040440	
8403	062324	044040	043511	100110	
8404	062332	200			
8405	062333	116	044505	044124	.ASCIZ /NEITHER -BYIN NOR DATI CAUSED A TRAP/<CRLF>
8406	062340	051105	026440	034502	
8407	062346	047111	047040	051117	
8408	062354	042040	052101	020111	
8409	062362	040503	051525	042105	
8410	062370	040440	052040	040522	
8411	062376	100120	000		
8412	062401	105	052111	042510	EM266: .ASCII /EITHER IRCD BYIN DOES NOT GET TO TMCC E12 AS A HIGH/<CRLF>
8413	062406	020122	051111	042103	
8414	062414	041040	044531	020116	
8415	062422	047504	051505	047040	
8416	062430	052117	043440	052105	
8417	062436	052040	020117	046524	
8418	062444	041503	042440	031061	
8419	062452	040440	020123	020101	
8420	062460	044510	044107	200	
8421	062465	117	020122	046524	.ASCIZ /OR TMCC E12(9,8) BAD OR E7(2) BAD/
8422	062472	041503	042440	031061	
8423	062500	034450	034054	020051	
8424	062506	040502	020104	051117	
8425	062514	042440	024067	024462	
8426	062522	041040	042101	000	
8427	062527	105	052111	042510	EM267: .ASCIZ /EITHER DAPB BAMX00 DOES NOT GET TO TMCC E7(4) OR E7 BAD/
8428	062534	020122	040504	041120	
8429	062542	041040	046501	030130	
8430	062550	020060	047504	051505	
8431	062556	047040	052117	043440	
8432	062564	052105	052040	020117	
8433	062572	046524	041503	042440	
8434	062600	024067	024464	047440	
8435	062606	020122	033505	041040	
8436	062614	042101	000		
8437	062617	105	052111	042510	EM270: .ASCII /EITHER RACC UBSC00 DOES NOT GET TO TMCC E5(13) AS/<CRLF>
8438	062624	020122	040522	041503	
8439	062632	052440	051502	030103	
8440	062640	020060	047504	051505	
8441	062646	047040	052117	043440	

8442	062654	052105	052040	020117	
8443	062662	046524	041503	042440	
8444	062670	024065	031461	020051	
8445	062676	051501	200		
8446	062701	101	046040	053517	.ASCIZ /A LOW OR E5(13) BAD/
8447	062706	047440	020122	032505	
8448	062714	030450	024463	041040	
8449	062722	042101	000		
8450	062725	105	052111	042510	EM271: .ASCII /EITHER RACC UBSC02 DOES NOT GET TO TMCC E12(5) AS/<CRLF>
8451	062732	020122	040522	041503	
8452	062740	052440	051502	030103	
8453	062746	020062	047504	051505	
8454	062754	047040	052117	043440	
8455	062762	052105	052040	020117	
8456	062770	046524	041503	042440	
8457	062776	031061	032450	020051	
8458	063004	051501	200		
8459	063007	101	044040	043511	.ASCIZ /A HIGH OR E12(6) DOES NOT GO LOW OR E5(12) BAD/
8460	063014	020110	051117	042440	
8461	063022	031061	033050	020051	
8462	063030	047504	051505	047040	
8463	063036	052117	043440	020117	
8464	063044	047514	020127	051117	
8465	063052	042440	024065	031061	
8466	063060	020051	040502	000104	
8467	063066	046523	032463	025067	EM272: .ASCIZ /SM357*SRC1 DATI FAILED TO TRAP/
8468	063074	051123	030503	042040	
8469	063102	052101	020111	040506	
8470	063110	046111	042105	052040	
8471	063116	020117	051124	050101	
8472	063124	000			
8473	063125	117	042104	040440	EM273: .ASCIZ /ODD ADR BIT IN CPUERR REG DOES NOT SET/
8474	063132	051104	041040	052111	
8475	063140	044440	020116	050103	
8476	063146	042525	051122	051040	
8477	063154	043505	042040	042517	
8478	063162	020123	047516	020124	
8479	063170	042523	000124		
8480	063174	047516	052040	040522	EM274: .ASCIZ /NO TRAP ON DATO/
8481	063202	020120	047117	042040	
8482	063210	052101	000117		
8483	063214	044505	044124	051105	EM275: .ASCII /EITHER PS04(1) DOES NOT GET TO TMCB E74(9) AS A HIGH/<CRLF>
8484	063222	050040	030123	024064	
8485	063230	024461	042040	042517	
8486	063236	020123	047516	020124	
8487	063244	042507	020124	047524	
8488	063252	052040	041515	020102	
8489	063260	033505	024064	024471	
8490	063266	040440	020123	020101	
8491	063274	044510	044107	200	
8492	063301	117	020122	052111	.ASCII /OR IT DOES NOT GET TO E51(10) AS A LOW OR E51 BAD/<CRLF>
8493	063306	042040	042517	020123	
8494	063314	047516	020124	042507	
8495	063322	020124	047524	042440	
8496	063330	030465	030450	024460	
8497	063336	040440	020123	020101	

8498	063344	047514	020127	051117	
8499	063352	042440	030465	041040	
8500	063360	042101	200		
8501	063363	117	020122	051111	.ASCII /OR IRCD RTT DOES NOT GET TO TMCB E74(11) AS A HIGH/<CRLF>
8502	063370	042103	051040	052124	
8503	063376	042040	042517	020123	
8504	063404	047516	020124	042507	
8505	063412	020124	047524	052040	
8506	063420	041515	020102	033505	
8507	063426	024064	030461	020051	
8508	063434	051501	040440	044040	
8509	063442	043511	100110		
8510	063446	051117	052040	041515	.ASCII /OR TMCB HONOR T DID NOT GO LOW OR TMCB TOK DID/<CRLF>
8511	063454	020102	047510	047516	
8512	063462	020122	020124	044504	
8513	063470	020104	047516	020124	
8514	063476	047507	046040	053517	
8515	063504	047440	020122	046524	
8516	063512	041103	052040	045517	
8517	063520	042040	042111	200	
8518	063525	116	052117	043440	.ASCIZ /NOT GO LOW OR IT DID NOT GET THRU TMCB E53/
8519	063532	020117	047514	020127	
8520	063540	051117	044440	020124	
8521	063546	044504	020104	047516	
8522	063554	020124	042507	020124	
8523	063562	044124	052522	052040	
8524	063570	041515	020102	032505	
8525	063576	000063			
8526	063600	044502	020124	020064	EM276: .ASCIZ /BIT 4 IN PSW DOES NOT SET/
8527	063606	047111	050040	053523	
8528	063614	042040	042517	020123	
8529	063622	047516	020124	042523	
8530	063630	000124			
8531	063632	044505	044124	051105	EM300: .ASCIZ /EITHER TMCB TOK DOES NOT GET TO DAPE E7(11) AS A LOW OR E7 BAD/
8532	063640	052040	041515	020102	
8533	063646	047524	020113	047504	
8534	063654	051505	047040	052117	
8535	063662	043440	052105	052040	
8536	063670	020117	040504	042520	
8537	063676	042440	024067	030461	
8538	063704	020051	051501	040440	
8539	063712	046040	053517	047440	
8540	063720	020122	033505	041040	
8541	063726	042101	000		
8542	063731	105	052111	042510	EM301: .ASCII /EITHER IRCD RTT DOES NOT GET TO TMCB E74 AS A LOW/<CRLF>
8543	063736	020122	051111	042103	
8544	063744	051040	052124	042040	
8545	063752	042517	020123	047516	
8546	063760	020124	042507	020124	
8547	063766	047524	052040	041515	
8548	063774	020102	033505	020064	
8549	064002	051501	040440	046040	
8550	064010	053517	200		
8551	064013	117	020122	033505	.ASCIZ /OR E74 BAD/
8552	064020	020064	040502	000104	
8553	064026	044505	044124	051105	EM302: .ASCIZ /EITHER TMCB E58(5) DID NOT GO LOW OR E64 BAD/

8554	064034	052040	041515	020102	
8555	064042	032505	024070	024465	
8556	064050	042040	042111	047040	
8557	064056	052117	043440	020117	
8558	064064	047514	020127	051117	
8559	064072	042440	032066	041040	
8560	064100	042101	000		
8561	064103	105	052111	042510	EM303: .ASCII /EITHER TMCA ABOVE BR7 DOES NOT GO LOW ON A YEL ZONE OR/<CRLF>
8562	064110	020122	046524	040503	
8563	064116	040440	047502	042526	
8564	064124	041040	033522	042040	
8565	064132	042517	020123	047516	
8566	064140	020124	047507	046040	
8567	064146	053517	047440	020116	
8568	064154	020101	042531	020114	
8569	064162	047532	042516	047440	
8570	064170	100122			
8571	064172	052111	042040	042517	.ASCIZ /IT DOES NOT GET TO TMCB E84(13) OR E84 BAD/
8572	064200	020123	047516	020124	
8573	064206	042507	020124	047524	
8574	064214	052040	041515	020102	
8575	064222	034105	024064	031461	
8576	064230	020051	051117	042440	
8577	064236	032070	041040	042101	
8578	064244	000			
8579	064245	105	052111	042510	EM304: .ASCIZ /EITHER TMCA ABOVE BR7 DOES NOT GET TO TMCB E84(1) OR E84 BAD/
8580	064252	020122	046524	040503	
8581	064260	040440	047502	042526	
8582	064266	041040	033522	042040	
8583	064274	042517	020123	047516	
8584	064302	020124	042507	020124	
8585	064310	047524	052040	041515	
8586	064316	020102	034105	024064	
8587	064324	024461	047140	020122	
8588	064332	034105	020064	040502	
8589	064340	000104			
8590	064342	044505	044124	051105	EM305: .ASCIZ /EITHER TMCA ABOVE BR7 DOES NOT GET TO TMCB E77(13) OR E77 BAD/
8591	064350	052040	041515	020101	
8592	064356	041101	053117	020105	
8593	064364	051102	020067	047504	
8594	064372	051505	047040	052117	
8595	064400	043440	052105	052040	
8596	064406	020117	046524	041103	
8597	064414	042440	033467	030450	
8598	064422	024463	047440	020122	
8599	064430	033505	020067	040502	
8600	064436	000104			
8601	064440	044505	044124	051105	EM306: .ASCII /EITHER TMCA BLOCK BR4 DOES NOT GO LOW ON PIR4 OR/<CRLF>
8602	064446	052040	041515	020101	
8603	064454	046102	041517	020113	
8604	064462	051102	020064	047504	
8605	064470	051505	047040	052117	
8606	064476	043440	020117	047514	
8607	064504	020127	047117	050040	
8608	064512	051111	020064	051117	
8609	064520	200			

8610	064521	111	020124	047504		.ASCIZ /IT DOES NOT GET TO TMCB E77(5) OR E77 BAD/
8611	064526	051505	047040	052117		
8612	064534	043440	052105	052040		
8613	064542	020117	046524	041103		
8614	064550	042440	033467	032450		
8615	064556	020051	051117	042440		
8616	064564	033467	041040	042101		
8617	064572	000				
8618	064573	124	041515	020101	EM307:	.ASCIZ /TMCA INH BELOW PIR4 DOES NOT GO LOW ON PIR5/
8619	064600	047111	020110	042502		
8620	064606	047514	020127	044520		
8621	064614	032122	042040	042517		
8622	064622	020123	047516	020124		
8623	064630	047507	046040	053517		
8624	064636	047440	020116	044520		
8625	064644	032522	000			
8626	064647	124	041515	020101	EM310:	.ASCIZ /TMCA INH BELOW PIR4 DOES NOT GO LOW ON BR5/
8627	064654	047111	020110	042502		
8628	064662	047514	020127	044520		
8629	064670	032122	042040	042517		
8630	064676	020123	047516	020124		
8631	064704	047507	046040	053517		
8632	064712	047440	020116	051102		
8633	064720	000065				
8634	064722	046524	040503	041040	EM311:	.ASCIZ /TMCA BLOCK LEVEL 4 DID NOT GO LOW ON PIR6/
8635	064730	047514	045503	046040		
8636	064736	053105	046105	032040		
8637	064744	042040	042111	047040		
8638	064752	052117	043440	020117		
8639	064760	047514	020127	047117		
8640	064766	050040	051111	000066		
8641	064774	044505	044124	051105	EM312:	.ASCII /EITHER TMCA ABOVE BR7 DOES NOT GO LOW(ON PIR 7) OR/<CRLF>
8642	065002	052040	041515	020101		
8643	065010	041101	053117	020105		
8644	065016	051102	020067	047504		
8645	065024	051505	047040	052117		
8646	065032	043440	020117	047514		
8647	065040	024127	047117	050040		
8648	065046	051111	033440	020051		
8649	065054	051117	200			
8650	065057	111	020124	047504		.ASCIZ /IT DOES NOT GET TO TMCB E77(1) OR E77 BAD/
8651	065064	051505	047040	052117		
8652	065072	043440	052105	052040		
8653	065100	020117	046524	041103		
8654	065106	042440	033467	030450		
8655	065114	020051	051117	042440		
8656	065122	033467	041040	042101		
8657	065130	000				
8658	065131	125	041502	020104	EM313:	.ASCIZ /UBCD EXT BRQ DID NOT GO LOW ON HONOR BR5/
8659	065136	054105	020124	051102		
8660	065144	020121	044504	020104		
8661	065152	047516	020124	047507		
8662	065160	046040	053517	047440		
8663	065166	020116	047510	047516		
8664	065174	020122	051102	000065		
8665	065202	044505	044124	051105	EM314:	.ASCII /EITHER INH BELOW BR6 DOES NOT GO LOW(ON BR6) OR IT DOES/<CRLF>

M15

PDP 11/70 CPU DIAGNOSTIC PART 2 MACY11 27(732) 21-DEC-76 16:00 PAGE 159
 DEKBBC.P11 POWER DOWN AND UP ROUTINES

SEQ 0194

8666	065210	044440	044116	041040	
8667	065216	046105	053517	041040	
8668	065224	033122	042040	042517	
8669	065232	020123	047516	020124	
8670	065240	047507	046040	053517	
8671	065246	047450	020116	051102	
8672	065254	024466	047440	020122	
8673	065262	052111	042040	042517	
8674	065270	100123			
8675	065272	047516	020124	042507	.ASCIZ /NOT GET THRU TMCA E82(6)/
8676	065300	020124	044124	052522	
8677	065306	052040	041515	020101	
8678	065314	034105	024062	024466	
8679	065322	000			
8680	065323	105	052111	042510	EM315: .ASCII /EITHER TMCA E67(8) DOES NOT GO LOW (ON SL YEL) OR IT IS NOT/<CRLF>
8681	065330	020122	046524	040503	
8682	065336	042440	033466	034050	
8683	065344	020051	047504	051505	
8684	065352	047040	052117	043440	
8685	065360	020117	047514	020127	
8686	065366	047450	020116	046123	
8687	065374	054440	046105	020051	
8688	065402	051117	044440	020124	
8689	065410	051511	047040	052117	
8690	065416	200			
8691	065417	107	052105	044524	.ASCIZ /GETTING TO E60(12) OR E60 BAD/
8692	065424	043516	052040	020117	
8693	065432	033105	024060	031061	
8694	065440	020051	051117	042440	
8695	065446	030066	041040	042101	
8696	065454	000			
8697	065455	105	052111	042510	EM316: .ASCIZ /EITHER TMCA E81(12) DOES NOT GO LOW(ON PIR5) OR IT DOES/<CRLF>
8698	065462	020122	046524	040503	
8699	065470	042440	030470	030450	
8700	065476	024462	042040	042517	
8701	065504	020123	047516	020124	
8702	065512	047507	046040	053517	
8703	065520	047450	020116	044520	
8704	065526	032522	020051	051117	
8705	065534	044440	020124	047504	
8706	065542	051505	000200		
8707	065546	042507	020124	047524	.ASCIZ /GET TO E83(10) OR E83 BAD/
8708	065554	042440	031470	030450	
8709	065562	024460	047440	020122	
8710	065570	034105	020063	040502	
8711	065576	000104			
8712	065600	046524	040503	042440	EM317: .ASCIZ /TMCA E81(2) BAD/
8713	065606	030470	031050	020051	
8714	065614	040502	000104		
8715	065620	044505	044124	051105	EM320: .ASCII /EITHER TMCA E67(8) DOES NOT GO LOW(ON PIR 7) OR IT DOES/<CRLF>
8716	065626	052040	041515	020101	
8717	065634	033105	024067	024400	
8718	065642	042040	042517	020123	
8719	065650	047516	020124	047507	
8720	065656	046040	053517	047450	
8721	065664	020116	044520	020122	

8722	065672	024467	047440	020122	
8723	065700	052111	042040	042517	
8724	065706	100123			
8725	065710	047516	020124	042507	.ASCIZ /NOT GET TO E83(13) OR E83 BAD/
8726	065716	020124	047524	042440	
8727	065724	031470	030450	024463	
8728	065732	047440	020122	034105	
8729	065740	020063	040502	000104	
8730	065746	041125	042103	042440	EM321: .ASCIZ /UBCD EXT BRQ DID NOT GO LOW ON HONOR BR6/
8731	065754	052130	041040	050522	
8732	065762	042040	042111	047040	
8733	065770	052117	043440	020117	
8734	065776	047514	020127	047117	
8735	066004	044040	047117	051117	
8736	066012	041040	033122	000	
8737	066017	105	052111	042510	EM322: .ASCIZ /EITHER TMCA E67(8) DOES NOT GET TO E83(1) OR E83 BAD/
8738	066024	020122	046524	040503	
8739	066032	042440	033466	034050	
8740	066040	020051	047504	051505	
8741	066046	047040	052117	043440	
8742	066054	052105	052040	020117	
8743	066062	034105	024063	024461	
8744	066070	047440	020122	034105	
8745	066076	020063	040502	000104	
8746	066104	044505	044124	051105	EM323: .ASCII /EITHER TMCA E81(8) IS NOT GOING LOW OR IT IS NOT/<CRLF>
8747	066112	052040	041515	020101	
8748	066120	034105	024061	024470	
8749	066126	044440	020123	047516	
8750	066134	020124	047507	047111	
8751	066142	020107	047514	020127	
8752	066150	051117	044440	020124	
8753	066156	051511	047040	052117	
8754	066164	200			
8755	066165	107	052105	044524	.ASCIZ /GETTING TO E76(12) OR E76 BAD/
8756	066172	043516	052040	020117	
8757	066200	033505	024066	031061	
8758	066206	020051	051117	042440	
8759	066214	033067	041040	042101	
8760	066222	000			
8761	066223	105	052111	042510	EM324: .ASCIZ /EITHER TMCA E67(8) NOT GETTING TO E76(13) OR E76 IS BAD/
8762	066230	020122	046524	040503	
8763	066236	042440	033466	034050	
8764	066244	020051	047516	020124	
8765	066252	042507	052124	047111	
8766	066260	020107	047524	042440	
8767	066266	033067	030450	024463	
8768	066274	047440	020122	033505	
8769	066302	020066	051511	041040	
8770	066310	042101	000		
8771	066313	105	052111	042510	EM325: .ASCIZ /EITHER TMCA E67(8) DOES NOT GET TO E76(1) OR E76 BAD/
8772	066320	020122	046524	040503	
8773	066326	042440	033466	034050	
8774	066334	020051	047504	051505	
8775	066342	047040	052117	043440	
8776	066350	052105	052040	020117	
8777	066356	033505	024066	024461	

8778	066364	047440	020122	033505	
8779	066372	020066	040502	000104	
8780	066400	044505	044124	051105	EM326: .ASCIZ /EITHER TMCA E67(6) NOT GETTING TO E69(12) OR E69 IS BAD/
8781	066406	052040	041515	020101	
8782	066414	033105	024067	024466	
8783	066422	047040	052117	043440	
8784	066430	052105	044524	043516	
8785	066436	052040	020117	033105	
8786	066444	024071	031061	020051	
8787	066452	051117	042440	034466	
8788	066460	044440	020123	040502	
8789	066466	000104			
8790	066470	044505	044124	051105	EM327: .ASCII /EITHER PSW BIT 11 DOES NOT SET OR TMCF CLK HI PS DOES/<CRLF>
8791	066476	050040	053523	041040	
8792	066504	052111	030440	020061	
8793	066512	047504	051505	047040	
8794	066520	052117	051440	052105	
8795	066526	047440	020122	046524	
8796	066534	043103	041440	045514	
8797	066542	044040	020111	051520	
8798	066550	042040	042517	100123	
8799	066556	047516	020124	047507	.ASCIZ /NOT GO LOW OR BIT 11 DOES NOT GET TO OR THRU THE DMUX/
8800	066564	046040	053517	047440	
8801	066572	020122	044502	020124	
8802	066600	030461	042040	042517	
8803	066606	020123	047516	020124	
8804	066614	042507	020124	047524	
8805	066622	047440	020122	044124	
8806	066630	052522	052040	042510	
8807	066636	042040	052515	000130	
8808	066644	044505	044124	051105	EM330: .ASCII /EITHER GRAC GDREG SET 1 DOES NOT GO HIGH (ON PAD 5) OR/<CRLF>
8809	066652	043440	040522	020103	
8810	066660	042107	042522	020107	
8811	066666	042523	020124	020061	
8812	066674	047504	051505	047040	
8813	066702	052117	043440	020117	
8814	066710	044510	044107	024040	
8815	066716	047117	050040	042101	
8816	066724	032440	020051	051117	
8817	066732	200			
8818	066733	104	042517	020123	.ASCIZ /DOES NOT GET TO THE GD REG/
8819	066740	047516	020124	042507	
8820	066746	020124	047524	052040	
8821	066754	042510	043440	020104	
8822	066762	042522	000107		
8823	066766	044505	044124	051105	EM331: .ASCII /EITHER GRAC GSREG SET 1 DOES NOT GO HIGH(ON PAD 5) OR/<CRLF>
8824	066774	043440	040522	020103	
8825	067002	051507	042522	020107	
8826	067010	042523	020124	020061	
8827	067016	047504	051505	047040	
8828	067024	052117	043440	020117	
8829	067032	044510	044107	047450	
8830	067040	020116	040520	020104	
8831	067046	024465	047440	100122	
8832	067054	052111	042040	042517	.ASCIZ /IT DOES NOT GET TO THE GS REG/
8833	067062	020123	047516	020124	

8834	067070	042507	020124	047524	
8835	067076	052040	042510	043440	
8836	067104	020123	042522	000107	
8837	067112	044505	044124	051105	EM332: .ASCIZ /EITHER GRAB SRC SET 1 DOES NOT G^ JW OR GRAC E23(6) BAD/
8838	067120	043440	040522	020102	
8839	067126	051123	020103	042523	
8840	067134	020124	020061	047504	
8841	067142	051505	047040	052117	
8842	067150	043440	020117	047514	
8843	067156	020127	051117	043440	
8844	067164	040522	020103	031105	
8845	067172	024063	024466	041040	
8846	067200	042101	000		
8847	067203	107	040522	020103	EM333: .ASCIZ /GRAC E24(6) DOES NOT GO LOW/
8848	067210	031105	024064	024466	
8849	067216	042040	042517	020123	
8850	067224	047516	020124	047507	
8851	067232	046040	053517	000	
8852	067237	107	040522	020103	EM334: .ASCIZ /GRAC E23(4) DOES NOT GO LOW/
8853	067244	031105	024063	024464	
8854	067252	042040	042517	020123	
8855	067260	047516	020124	047507	
8856	067266	046040	053517	000	
8857	067273	107	040522	020103	EM335: .ASCIZ /GRAC E23(4) IS NOT GOING LOW/
8858	067300	031105	024063	024464	
8859	067306	044440	020123	047516	
8860	067314	020124	047507	047111	
8861	067322	020107	047514	000127	
8862	067330	042120	042122	050040	EM336: .ASCIZ /PDRD PS11 DOES NOT GET TO GRAC AS A LOW/
8863	067336	030523	020061	047504	
8864	067344	051505	047040	052117	
8865	067352	043440	052105	052040	
8866	067360	020117	051107	041501	
8867	067366	040440	020123	020101	
8868	067374	047514	0001		
8869	067400	042522	044507	052123	EM337: .ASCIZ /REGISTER SET 1 SOURCE HAS STUCK BITS/
8870	067406	051105	051440	052105	
8871	067414	030440	051440	052517	
8872	067422	041522	020105	040510	
8873	067430	020123	052123	041525	
8874	067436	020113	044502	051524	
8875	067444	000			
8876	067445	105	051122	051117	DH337: .ASCIZ /ERRORPC PATTERN TEST NUMBER/
8877	067452	041520	050040	052101	
8878	067460	042524	047122	052040	
8879	067466	051505	020124	052516	
8880	067474	041115	051105	000	
8881		067502			
8882	067502	001116	001164	001210	DT337: .EVEN .WORD \$ERRPC,\$STMP1,\$STSTNM,0
8883	067510	000000			
8884	067512	042522	044507	052123	EM340: .ASCIZ /REGISTER SET 1 DESTINATION HAS STUCK BITS/
8885	067520	051105	051440	052105	
8886	067526	030440	042040	051505	
8887	067534	044524	040516	044524	
8888	067542	047117	044040	051501	
8889	067550	051440	052524	045503	

8890	067556	041040	052111	000123	
8891	067564	051107	041101	042040	EM341: .ASCIZ /GRAB DST SET 1 DOES NOT GO LOW ON R14/
8892	067572	052123	051440	052105	
8893	067600	030440	042040	042517	
8894	067606	020123	047516	020124	
8895	067614	047507	046040	053517	
8896	067622	047440	020116	030522	
8897	067630	000064			
8898	067632	051107	041101	051440	EM342: .ASCIZ /GRAB SRC SET 1 DOES NOT GO LOW ON R14/
8899	067640	041522	051440	052105	
8900	067646	030440	042040	042517	
8901	067654	020123	047516	020124	
8902	067662	047507	046040	053517	
8903	067670	047440	020116	030522	
8904	067678	000064			
8905	067700	030065	030060	020060	EM343: .ASCIZ /50000 PATTERN FAILED IN PSW/
8906	067706	040520	052124	051105	
8907	067714	020116	040506	046111	
8908	067722	042105	044440	020116	
8909	067730	051520	000127		
8910	067734	033061	030064	030060	EM344: .ASCIZ /164000 PATTERN FAILED IN PSW/
8911	067742	050040	052101	042524	
8912	067750	047122	043040	044501	
8913	067756	042514	020104	047111	
8914	067764	050040	053523	000	
8915	067771	120	053523	044040	EM345: .ASCIZ /PSW HIGH BYTE DID NOT CLEAR/
8916	067776	043511	020110	054502	
8917	070004	042524	042040	042111	
8918	070012	047040	052117	041440	
8919	070020	042514	051101	000	
8920	070025	107	040522	020102	EM346: .ASCIZ /GRAB DST SET 1 DOES NOT GO LOW ON DF6*SUPER MODE/
8921	070032	051504	020124	042523	
8922	070040	020124	020061	047504	
8923	070046	051505	047040	052117	
8924	070054	043440	020117	047514	
8925	070062	020127	047117	042040	
8926	070070	033106	051452	050125	
8927	070076	051105	046440	042117	
8928	070104	000105			
8929	070106	051107	041101	051440	EM347: .ASCIZ /GRAB SRC SET 1 DOES NOT GO LOW ON SF6*SUPER MODE/
8930	070114	041522	051440	052105	
8931	070122	030440	042040	042517	
8932	070130	020123	047516	020124	
8933	070136	047507	046040	053517	
8934	070144	047440	020116	043123	
8935	070152	025066	052523	042520	
8936	070160	020122	047515	042504	
8937	070166	000			
8938	070167	107	040522	020103	EM350: .ASCIZ /GRAB E35(8) DOES NOT GO HIGH ON DF6*USER MODE/
8939	070174	031505	024065	024470	
8940	070202	042040	042517	020123	
8941	070210	047516	020124	047507	
8942	070216	044040	043511	020110	
8943	070224	047117	042040	033106	
8944	070232	052452	042523	020122	
8945	070240	047515	042504	000	

E16

PDP 11/70 CPU DIAGNOSTIC PART 2 MACY11 27(732) 21-DEC-76 16:00 PAGE 164
 DEKBBC.P11 POWER DOWN AND UP ROUTINES

SEQ 0199

8946	070245	107	040522	020103	EM351:	.ASCIZ /GRAC E15(6) DOES NOT GO HIGH ON SF6*USER MODE/
8947	070252	030505	024065	024466		
8948	070260	042040	042517	020123		
8949	070266	047516	020124	047507		
8950	070274	044040	043511	020110		
8951	070302	047117	051440	033106		
8952	070310	052452	042523	020122		
8953	070316	047515	042504	000		
8954	070323	105	052111	042510	EM352:	.ASCIZ /EITHER USER OR SUPER SP DST FAILED BIT TEST/
8955	070330	020122	051525	051105		
8956	070336	047440	020122	052523		
8957	070344	042520	020122	050123		
8958	070352	042040	052123	043040		
8959	070360	044501	042514	020104		
8960	070366	044502	020124	042524		
8961	070374	052123	000			
8962	070377	105	052111	042510	EM353:	.ASCIZ /EITHER USER OR SUPER SP SRC FAILED BIT TEST/
8963	070404	020122	051525	051105		
8964	070412	047440	020122	052523		
8965	070420	042520	020122	050123		
8966	070426	051440	041522	043040		
8967	070434	044501	042514	020104		
8968	070442	044502	020124	042524		
8969	070450	052123	000			
8970	070453	107	040522	020103	EM354:	.ASCIZ /GRAC E20(4) NOT GOING LOW/
8971	070460	031105	024060	024464		
8972	070466	047040	052117	043440		
8973	070474	044517	043516	046040		
8974	070502	053517	000			
8975	070505	107	040522	020103	EM355:	.ASCIZ /GRAC E20(12) NOT GOING LOW ON MTP*DMO*DF6*PS12/
8976	070512	031105	024060	031061		
8977	070520	020051	047516	020124		
8978	070526	047507	047111	020107		
8979	070534	047514	020127	047117		
8980	070542	046440	050124	042052		
8981	070550	030115	042052	033106		
8982	070556	050052	030523	000062		
8983	070564	051107	041501	042440	EM356:	.ASCIZ /GRAC E20(11) NOT GOING LOW/
8984	070572	030062	030450	024461		
8985	070600	047040	052117	043440		
8986	070606	044517	043516	046040		
8987	070614	053517	000			
8988	070617	123	050123	041040	EM357:	.ASCII /SSP BAD ON MTP EITHER GRAC GRWE/<<CRLF>
8989	070624	042101	047440	020116		
8990	070632	052115	020120	044505		
8991	070640	044124	051105	043440		
8992	070646	040522	020103	051107		
8993	070654	042527	200			
8994	070657	110	041111	047440		.ASCIZ /HIB OR LOB NOT GOING LOW/
8995	070664	020122	047514	020102		
8996	070672	047516	020124	047507		
8997	070700	047111	020107	047514		
8998	070706	000127				
8999	070710	051105	047522	050122	DH357:	.ASCII /ERRORPC SSP TST NUM/<<CRLF>
9000	070716	020103	020040	020040		
9001	070724	051440	050123	052011		

DEKBBC.P11

POWER DOWN AND UP ROUTINES

9002	070732	052123	047040	046525	
9003	070740	200			
9004	070741	011	042440	050130	.ASCIZ / EXPECT ACTUAL/
9005	070746	041505	020124	040440	
9006	070754	052103	040525	000114	
9007	070762	051107	041501	042440	EM360: .ASCIZ /GRAC E35(8) DID NOT GO HIGH ON MTP*DMO*DF6*PS13/
9008	070770	032463	034050	020051	
9009	070776	044504	020104	047516	
9010	071004	020124	047507	044040	
9011	071012	043511	020110	047117	
9012	071020	046440	050124	042052	
9013	071026	030115	042052	033106	
9014	071034	050052	030523	000063	
9015	071042	051111	041503	042040	EM361: .ASCIZ /IRCC DMO(MFP+MTP) NOT GOING HIGH ON MFP/
9016	071050	030115	046450	050106	
9017	071056	046453	050124	020051	
9018	071064	047516	020124	047507	
9019	071072	047111	020107	044510	
9020	071100	044107	047440	020116	
9021	071106	043115	000120		
9022	071112	051107	041501	042440	EM362: .ASCIZ /GRAC E24(13) DOES NOT GO HIGH/
9023	071120	032062	030450	024463	
9024	071126	042040	042517	020123	
9025	071134	047516	020124	047507	
9026	071142	044040	043511	000110	
9027	071150	042120	042122	050040	EM363: .ASCIZ /PDRD PS15 DOES NOT GET TO GRAC AS A HIGH/
9028	071156	030523	020065	047504	
9029	071164	051505	047040	052117	
9030	071172	043440	052105	052040	
9031	071200	020117	051107	041501	
9032	071206	040440	020123	020101	
9033	071214	044510	044107	000	
9034	071221	123	050123	053440	EM364: .ASCIZ /SSP WAS READ BUT NEITHER USP NOR SSP WERE WRITTEN/
9035	071226	051501	051040	040505	
9036	071234	020104	052502	020124	
9037	071242	042516	052111	042110	
9038	071250	020122	051525	020120	
9039	071256	047516	020122	051523	
9040	071264	020120	042527	042522	
9041	071272	053440	044522	052124	
9042	071300	047105	000		
9043	071303	107	040522	020103	EM365: .ASCIZ /GRAC E23(13) DOES NOT GO HIGH/
9044	071310	031105	024063	031461	
9045	071316	020051	047504	051505	
9046	071324	047040	052117	043440	
9047	071332	020117	044510	044107	
9048	071340	000			
9049	071341	107	040522	020103	EM366: .ASCIZ /GRAC E23(3) DOES NOT GO HIGH/
9050	071346	031105	024063	024463	
9051	071354	042040	042517	020123	
9052	071362	047516	020124	047507	
9053	071370	044040	043511	000110	
9054	071376	044505	044124	051105	EM367: .ASCII /EITHER PDRD PS14(0) DOES NOT GET TO PDRD E73(13)/<CRLF>
9055	071404	050040	051104	020104	
9056	071412	051520	032061	030050	
9057	071420	020051	047504	051505	

9058	071426	047040	052117	043440	
9059	071434	052105	052040	020117	
9060	071442	042120	042122	042440	
9061	071450	031467	030450	024463	
9062	071456	200			
9063	071457	101	020123	020101	.ASCIZ /AS A HIGH OR E73(12) NOT GOING LOW/
9064	071464	044510	044107	047440	
9065	071472	020122	033505	024063	
9066	071500	031061	020051	047516	
9067	071506	020124	047507	047111	
9068	071514	020107	047514	000127	
9069	071522	020101	051520	020127	EM370: .ASCII /A PSW HIGH BYTE BIT(S) DID NOT SET ON LOAD PS &/<CRLF>
9070	071530	044510	044107	041040	
9071	071536	052131	020105	044502	
9072	071544	024124	024523	042040	
9073	071552	042111	047040	052117	
9074	071560	051440	052105	047440	
9075	071566	020116	047514	042101	
9076	071574	050040	020123	100046	
9077	071602	042513	047122	046105	.ASCIZ /KERNEL MODE WITH A BR VALUE OF 174000/
9078	071610	046440	042117	020105	
9079	071616	044527	044124	040440	
9080	071624	041040	020122	040526	
9081	071632	052514	020105	043117	
9082	071640	030440	032067	030060	
9083	071646	000060			
9084	071650	020101	044502	024124	EM371: .ASCII /A BIT(S) IN THE PSW CLEARED ON AN RTI IN USER MODE/<CRLF>
9085	071656	024523	044440	020116	
9086	071664	044124	020105	051520	
9087	071672	020127	046103	040505	
9088	071700	042522	020104	047117	
9089	071706	040440	020116	052122	
9090	071714	020111	047111	052440	
9091	071722	042523	020122	047515	
9092	071730	042504	200		
9093	071733	046	040440	041040	.ASCIZ /& A BR VALUE OF 0/
9094	071740	020122	040526	052514	
9095	071746	020105	043117	030040	
9096	071754	000			
9097	071755	101	041040	052111	EM372: .ASCII /A BIT(S) OF PSW <15,13:11> DID NOT PRESET ON AN RTI/<CRLF>
9098	071762	051450	020051	043117	
9099	071770	050040	053523	036040	
9100	071776	032461	030454	035063	
9101	072004	030461	020076	044504	
9102	072012	020104	047516	020124	
9103	072020	051120	051505	052105	
9104	072026	047440	020116	047101	
9105	072034	051040	044524	200	
9106	072041	111	020116	052523	.ASCIZ /IN SUPER MODE & A BR VALUE OF 134000/
9107	072046	042520	020122	047515	
9108	072054	042504	023040	040440	
9109	072062	041040	020122	040526	
9110	072070	052514	020105	043117	
9111	072076	030440	032063	030060	
9112	072104	000060			
9113	072106	020101	044502	024124	EM373: .ASCII /A BIT(S) IN PSW HIGH BYTE FAILED ON AN IOT IN USER/<CRLF>

9114	072114	024523	044440	020116
9115	072122	051520	020127	044510
9116	072130	044107	041040	052131
9117	072136	020105	040506	046111
9118	072144	042105	047440	020116
9119	072152	047101	044440	052117
9120	072160	044440	020116	051525
9121	072166	051105	200	
9122	072171	115	042117	020105
9123	072176	020046	020101	051102
9124	072204	053040	046101	042525
9125	072212	047440	020106	000060
9126	072220	020101	051120	053105
9127	072226	047511	051525	046440
9128	072234	042117	020105	044502
9129	072242	024124	024523	042040
9130	072250	042111	047040	052117
9131	072256	041440	042514	051101
9132	072264	047440	020116	047101
9133	072272	044440	052117	044440
9134	072300	020116	042513	047122
9135	072306	046105	000	
9136	072311	105	052111	042510
9137	072316	020122	051523	041522
9138	072324	045440	020124	041101
9139	072332	051117	020124	046106
9140	072340	020107	047504	051505
9141	072346	047040	052117	043440
9142	072354	052105	052040	020117
9143	072362	046524	041503	042440
9144	072370	032063	030450	024460
9145	072376	047440	100122	
9146	072402	031505	024064	030061
9147	072410	020051	040502	020104
9148	072416	051117	052040	041515
9149	072424	020103	042501	043122
9150	072432	030450	020051	047504
9151	072440	051505	047040	052117
9152	072446	043440	020117	044510
9153	072454	044107	047440	020116
9154	072462	020101	052113	040440
9155	072470	047502	052122	000
9156	072475	105	052111	042510
9157	072502	020122	046524	041103
9158	072510	051440	043505	020124
9159	072516	047504	051505	047040
9160	072524	052117	043440	020117
9161	072532	047514	020127	047117
9162	072540	040440	045440	020124
9163	072546	041101	051117	020124
9164	072554	051117	200	
9165	072557	111	020124	047504
9166	072564	051505	047040	052117
9167	072572	043440	052105	052040
9168	072600	020117	040504	042520
9169	072606	000		

.ASCIZ /MODE & A BR VALUE OF 0/

EM374: .ASCIZ /A PREVIOUS MODE BIT(S) DID NOT CLEAR ON AN IOT IN KERNEL/

EM375: .ASCII /EITHER SSRC KT ABORT FLG DOES NOT GET TO TMCC E34(10) OR/<CRLF>

.ASCIZ /E34(10) BAD OR TMCC AERF(1) DOES NOT GO HIGH ON A KT ABORT/

EM400: .ASCII /EITHER TMCB SEGT DOES NOT GO LOW ON A KT ABORT OR/<CRLF>

.ASCIZ /IT DOES NOT GET TO DAPE/

9170	072607	104	050101	020105	EM401:	.ASCIZ /DAPE TV05*07 DOES NOT GO HIGH ON SEG7/
9171	072614	053124	032460	030052		
9172	072622	020067	047504	051505		
9173	072630	047040	052117	043440		
9174	072636	020117	044510	044107		
9175	072644	047440	020116	042523		
9176	072652	052107	000			
9177	072655	104	050101	020105	EM402:	.ASCIZ /DAPE TV03 DOES NOT GO HIGH ON SEG7/
9178	072662	053124	031460	042040		
9179	072670	042517	020123	047516		
9180	072676	020124	047507	044040		
9181	072704	043511	020110	047117		
9182	072712	051440	043505	000124		
9183	072720	044505	044124	051105	EM403:	.ASCII /EITHER TMCA SEG+CON+PAR DOES NOT GO LOW OR IT DOES/<CRLF>
9184	072726	052040	041515	020101		
9185	072734	042523	025507	047503		
9186	072742	025516	040520	020122		
9187	072750	047504	051505	047040		
9188	072756	052117	043440	020117		
9189	072764	047514	020127	051117		
9190	072772	044440	020124	047504		
9191	073000	051505	200			
9192	073003	116	052117	043440		.ASCIZ /NOT GET THRU TMCB E70(2)/
9193	073010	052105	052040	051110		
9194	073016	020125	046524	041103		
9195	073024	042440	030067	031050		
9196	073032	000051				
9197	073034	044505	044124	051105	EM405:	.ASCII /EITHER TMCA SEGTF DOES NOT GET TO E44 OR TMCE PAUSES H/<15><12>
9198	073042	052040	041515	020101		
9199	073050	042523	052107	020106		
9200	073056	047504	051505	047040		
9201	073064	052117	043440	052105		
9202	073072	052040	020117	032105		
9203	073100	020064	051117	052040		
9204	073106	041515	020105	040520		
9205	073114	051525	051505	044040		
9206	073122	005015				
9207	073124	047504	051505	047040		.ASCIZ /DOES NOT GET TO E44 OR E44 BAD/
9208	073132	052117	043440	052105		
9209	073140	052040	020117	032105		
9210	073146	020064	051117	042440		
9211	073154	032064	041040	042101		
9212	073162	000				
9213	073163	105	052111	042510	EM406:	.ASCII /EITHER TMCC NEXM DOES NOT GO LOW OR IT DOES NOT GET THRU E34/<CRLF>
9214	073170	020122	046524	041503		
9215	073176	047040	054105	020115		
9216	073204	047504	051505	047040		
9217	073212	052117	043440	020117		
9218	073220	047514	020127	051117		
9219	073226	044440	020124	047504		
9220	073234	051505	047040	052117		
9221	073242	043440	052105	052040		
9222	073250	051110	020125	031505		
9223	073256	100064				
9224	073260	051117	044440	020124		.ASCIZ /OR IT DOES NOT GET TO E14 OR E40 BAD/
9225	073266	047504	051505	047040		

9226	073274	052117	043440	052105	
9227	073302	052040	020117	030505	
9228	073310	020064	051117	042440	
9229	073316	030064	041040	042101	
9230	073324	000			
9231	073325	116	054105	020115	EM410: .ASCII /NEXM BIT DID NOT SET IN CPU ERROR REG/⟨CRLF⟩
9232	073332	044502	020124	044504	
9233	073340	020104	047516	020124	
9234	073346	042523	020124	047111	
9235	073354	041440	052520	042440	
9236	073362	051122	051117	051040	
9237	073370	043505	200		
9238	073373	117	020122	046524	.ASCIZ /OR TMCC ABORT DOES NOT GO HIGH/
9239	073400	041503	040440	047502	
9240	073406	052122	042040	042517	
9241	073414	020123	047516	020124	
9242	073422	047507	044040	043511	
9243	073430	000110			
9244	073432	042516	046530	041040	EM411: .ASCIZ /NEXM BIT DID NOT CLEAR IN CPU ERROR REG/
9245	073440	052111	042040	042111	
9246	073446	047040	052117	041440	
9247	073454	042514	051101	044440	
9248	073462	020116	050103	020125	
9249	073470	051105	047522	020122	
9250	073476	042522	000107		
9251	073502	046524	042503	045440	EM412: .ASCIZ /TMCE KT BEND DOES NOT GO LOW ON TMCC NEXM LOW/
9252	073510	020124	042502	042116	
9253	073516	042040	042517	020123	
9254	073524	047516	020124	047507	
9255	073532	046040	053517	047440	
9256	073540	020116	046524	041503	
9257	073546	047040	054105	020115	
9258	073554	047514	000127		
9259	073560	046524	042503	045440	EM413: .ASCIZ /TMCE KT BEND DOES NOT GO LOW ON TMCC SL RED/
9260	073566	020124	042502	042116	
9261	073574	042040	042517	020123	
9262	073602	047516	020124	047507	
9263	073610	046040	053517	047440	
9264	073616	020116	046524	042103	
9265	073624	051440	020114	042522	
9266	073632	000104			
9267	073634	046524	042503	045440	EM414: .ASCIZ /TMCE KT BEND DOES NOT GO LOW ON TMCC ODD ADRS ERR/
9268	073642	020124	042502	042116	
9269	073650	042040	042517	020123	
9270	073656	047516	020124	047507	
9271	073664	046040	053517	047440	
9272	073672	020116	046524	041503	
9273	073700	047440	042104	040440	
9274	073706	051104	020123	051105	
9275	073714	000122			
9276	073716	046524	042503	041440	EM416: .ASCIZ /TMCE CACHE BEND DID NOT GO HIGH ON KT ABORT/
9277	073724	041501	042510	041040	
9278	073732	047105	020104	044504	
9279	073740	020104	047516	020124	
9280	073746	047507	044040	043511	
9281	073754	020110	047117	045440	

K16

PDP 11/70 CPU DIAGNOSTIC PART 2 MACY11 27(732) 21-DEC-76 16:00 PAGE 170
 DEKBBC.P11 POWER DOWN AND UP ROUTINES

SEQ 0205

9282	073762	020124	041101	051117	
9283	073770	030124			
9284	073772	044505	044124	051105	EM417: .ASCII /EITHER DAPA BR14 DOES NOT GET TO RACK E49 AS A LOW/<CRLF>
9285	074000	042040	050101	020101	
9286	074006	051102	032061	042040	
9287	074014	042517	020123	047516	
9288	074022	020124	042507	020124	
9289	074030	047524	051040	041501	
9290	074036	020113	032105	020071	
9291	074044	051501	040440	046040	
9292	074052	053517	200		
9293	074055	117	020122	032105	..ASCIZ /OR E49(5) BAD/
9294	074062	024071	024465	041040	
9295	074070	042101	800		
9296	074073	111	046114	043505	EM420: .ASCIZ /ILLEGAL HALT BIT IN CPU ERROR REG DID NOT SET/
9297	074100	046101	044040	046101	
9298	074106	020124	044502	020124	
9299	074114	047111	041440	052520	
9300	074122	042440	051122	051117	
9301	074130	051040	043505	042040	
9302	074136	042111	047040	052117	
9303	074144	051440	052105	000	
9304	074151	105	052111	042510	EM421: .ASCII /EITHER SSPA PS RESTORE(1) DOES NOT GET TO RACK E63/<CRLF>
9305	074156	020122	051523	040522	
9306	074164	050040	020123	042522	
9307	074172	052123	051117	024105	
9308	074200	024461	042040	042517	
9309	074206	020123	047516	020124	
9310	074214	042507	020124	047524	
9311	074222	051040	041501	020113	
9312	074230	033105	100063		
9313	074234	051117	042440	031466	.ASCIZ /OR E63(5) BAD/
9314	074242	032450	020051	040502	
9315	074250	000104			
9316	074252	044505	044124	051105	EM422: .ASCII /EITHER UBCB PE ABORT DOES NOT GO LOW OR/<CRLF>
9317	074260	052440	041502	020102	
9318	074266	042520	040440	047502	
9319	074274	052122	042040	042517	
9320	074302	020123	047516	020124	
9321	074310	047507	046040	053517	
9322	074316	047440	100122		
9323	074322	052111	042040	042517	.ASCII /IT DOES NOT GET TO TMCC E33 OR E33 BAD/<CRLF>
9324	074330	020123	047516	020124	
9325	074336	042507	020124	047524	
9326	074344	052040	041515	020103	
9327	074352	031505	020063	051117	
9328	074360	042440	031463	041040	
9329	074366	042101	200		
9330	074371	117	020122	041125	.ASCII /OR UBCB PARITY ERR DOES NOT GET TO TMCB E53/<CRLF>
9331	074376	041103	050040	051101	
9332	074404	052111	020131	051105	
9333	074412	020122	047504	051505	
9334	074420	047040	052117	043440	
9335	074426	052105	052040	020117	
9336	074434	046524	041103	042440	
9337	074442	031465	200		

9338	074445	101	020123	020101		.ASCIZ /AS A LOW OR E53(5) BAD/
9339	074452	047514	020127	051117		
9340	074460	042440	031465	032450		
9341	074466	020051	040502	000104		
9342	074474	044505	044124	051105	EM423:	.ASCII /EITHER UBCB PARITY ERR DOES NOT GO LOW OR IT DOES/<CRLF>
9343	074502	052440	041502	020102		
9344	074510	040520	044522	054524		
9345	074516	042440	051122	042040		
9346	074524	042517	020123	047516		
9347	074532	020124	047507	046040		
9348	074540	053517	047440	020122		
9349	074546	052111	042040	042517		
9350	074554	100123				
9351	074556	047516	020124	042507		.ASCIZ /NOT GET TO DAPE/
9352	074564	020124	047524	042040		
9353	074572	050101	000105			
9354	074576	044505	044124	051105	EM424:	.ASCIZ /EITHER DAPE E11(4) BAD OR TV06 DOES NOT GET TO THE ALU/
9355	074604	042040	050101	020105		
9356	074612	030505	024061	024464		
9357	074620	041040	042101	047440		
9358	074626	020122	053124	033060		
9359	074634	042040	042517	020123		
9360	074642	047516	020124	042507		
9361	074650	020124	047524	052040		
9362	074656	042510	040440	052514		
9363	074664	000				
9364	074665	104	050101	020105	EM425:	.ASCIZ /DAPE E7(1) BAD/
9365	074672	033505	030450	020051		
9366	074700	040502	000104			
9367	074704	046524	040503	051440	EM426:	.ASCIZ /TMCA SEG+CON+PAR DOES NOT GO LOW ON CCBJ PARITY TRAP/
9368	074712	043505	041453	047117		
9369	074720	050053	051101	042040		
9370	074726	042517	020123	047516		
9371	074734	020124	047507	046040		
9372	074742	053517	047440	020116		
9373	074750	041503	045102	050040		
9374	074756	051101	052111	020131		
9375	074764	051124	050101	000		
9376	074771	105	052111	042510	EM427:	.ASCII /EITHER TMCB PART DOES NOT GO LOW OR DOES/<CRLF>
9377	074776	020122	046524	041103		
9378	075004	050040	051101	020124		
9379	075012	047504	051505	047040		
9380	075020	052117	043440	020117		
9381	075026	047514	020127	051117		
9382	075034	042040	042517	100123		
9383	075042	047516	020124	042507		.ASCIZ /NOT GET TO UBCB OR UBCB E18(1) BAD/
9384	075050	020124	047524	052440		
9385	075056	041502	020102	051117		
9386	075064	052440	041502	020102		
9387	075072	030505	024070	024461		
9388	075100	041040	042101	000		
9389	075105	124	041515	020101	EM430:	.ASCIZ /TMCA E67(8) DOES NOT GO LOW ON MGMT/
9390	075112	033105	024067	024470		
9391	075120	042040	042517	020123		
9392	075126	047516	020124	047507		
9393	075134	046040	053517	047440		

M16

9394	075142	020116	043515	052115	
9395	075150	000			
9396	075151	124	041515	020101	EM431: .ASCIZ /TMCA E67(12) DOES NOT GO LOW ON MGMT/
9397	075156	033105	024067	031061	
9398	075164	020051	047504	051505	
9399	075172	047040	052117	043440	
9400	075200	020117	047514	020127	
9401	075206	047117	046440	046507	
9402	075214	000124			
9403	075216	044505	044124	051105	EM432: .ASCII /EITHER TMCA E68(6) DOES NOT GO LOW ON PAR TRP/<CRLF>
9404	075224	052040	041515	020101	
9405	075232	033105	024070	024466	
9406	075240	042040	042517	020123	
9407	075246	047516	020124	047507	
9408	075254	046040	053517	047440	
9409	075262	020116	040520	020122	
9410	075270	051124	100120		
9411	075274	051117	042440	032464	.ASCIZ /OR E45(4) BAD/
9412	075302	032050	020051	040502	
9413	075310	000104			
9414	075312	046524	040503	042440	EM433: .ASCIZ /TMCA E68(8) DOES NOT GO LOW ON PAR TRP/
9415	075320	034066	034050	020051	
9416	075326	047504	051505	047040	
9417	075334	052117	043440	020117	
9418	075342	047514	020127	047117	
9419	075350	050040	051101	052040	
9420	075356	050122	000		
9421	075361	105	052111	042510	EM435: .ASCII /EITHER TMCC PRIORITY CLEAR DID NOT GO LOW OR DID NOT/<CRLF>
9422	075366	020122	046524	041503	
9423	075374	050040	044522	051117	
9424	075402	052111	020131	046103	
9425	075410	040505	020122	044504	
9426	075416	020104	047516	020124	
9427	075424	047507	046040	053517	
9428	075432	047440	020122	044504	
9429	075440	020104	047516	100124	
9430	075446	042507	020124	044124	.ASCIZ /GET THRU TMCA E43(2) ON ABORT CLEAR/
9431	075454	052522	052040	041515	
9432	075462	020101	032105	024063	
9433	075470	024462	047440	020116	
9434	075476	041101	051117	020124	
9435	075504	046103	040505	000122	
9436	075512	052502	020123	041120	EM436: .ASCIZ /BUS PB DID NOT GET TO UBCB PE ABORT/
9437	075520	042040	042111	047040	
9438	075526	02117	043440	052105	
9439	075534	052040	020117	041125	
9440	075542	041103	050040	020105	
9441	075550	041101	051117	000124	
9442	075556	041125	041103	050040	EM437: .ASCIZ /UBCB PARITY ERR DID NOT GO LOW ON BUS PB/
9443	075564	051101	052111	020131	
9444	075572	051105	020122	044504	
9445	075600	020104	047516	020124	
9446	075606	047507	046040	053517	
9447	075614	047440	020116	052502	
9448	075622	020123	041120	000	
9449	075627	127	044501	020124	EM440: .ASCIZ /WAIT INSTRUCTION DID NOT WAIT FOR INTERRUPT/

9450	075634	047111	052123	052522	
9451	075642	052103	047511	020116	
9452	075650	044504	020104	047516	
9453	075656	020124	040527	052111	
9454	075664	043040	051117	044440	
9455	075672	052116	051105	052522	
9456	075700	052120	000		
9457	075703	127	044501	020124	EM441: .ASCIZ /WAIT INSTRUCTION FELL THRU ON T BIT TRAP/
9458	075710	047111	052123	052522	
9459	075716	052103	047511	020116	
9460	075724	042506	046114	052040	
9461	075732	051110	020125	047117	
9462	075740	052040	041040	052111	
9463	075746	052040	040522	000120	
9464	075754	041125	041503	024040	EM442: .ASCIZ /UBCC (PWR+INTR) DOES NOT GO LOW ON TMCB PIRQ/
9465	075762	053520	043122	044453	
9466	075770	052116	024522	042040	
9467	075776	042517	020123	047516	
9468	076004	020124	047507	046040	
9469	076012	053517	047440	020116	
9470	076020	046524	041103	050040	
9471	076026	051111	000121		
9472	076032	030523	027063	030060	EM443: .ASCIZ /S13.00 FAILED/
9473	076040	043040	044501	042514	
9474	076046	000104			
9475	076050	032123	027065	030060	EM444: .ASCIZ /S45.00 FAILED/
9476	076056	043040	044501	042514	
9477	076064	000104			
9478	076066	052115	027120	030061	EM445: .ASCIZ /MTP.10 FAILED/
9479	076074	043040	044501	042514	
9480	076102	000104			
9481	076104	042516	027107	030071	EM446: .ASCIZ /NEG.90 FAILED/
9482	076112	043040	044501	042514	
9483	076120	000104			
9484	076122	032104	027065	030070	EM447: .ASCIZ /D45.80 FAILED/
9485	076130	043040	044501	042514	
9486	076136	000104			
9487	076140	032104	027065	030071	EM450: .ASCIZ /D45.90 FAILED/
9488	076146	043040	044501	042514	
9489	076154	000104			
9490	076156	032104	027065	030060	EM451: .ASCIZ /D45.00 FAILED/
9491	076164	043040	044501	042514	
9492	076172	000104			
9493	076174	032104	027065	030460	EM452: .ASCIZ /D45.01 FAILED/
9494	076202	043040	044501	042514	
9495	076210	000104			
9496	076212	047101	044440	046114	EM453: .ASCIZ /AN ILLEGAL INSTRUCTION FAILED TO TRAP/
9497	076220	043505	046101	044440	
9498	076226	051516	051124	041525	
9499	076234	044524	047117	043040	
9500	076242	044501	042514	020104	
9501	076250	047524	052040	040522	
9502	076256	000120			
9503	076260	044505	044124	051105	EM454: .ASCIZ /EITHER TMCB E53 DOES NOT GO HIGH OR TMCB PIRQ DOES NOT GO LOW/
9504	076266	052040	041515	020102	
9505	076274	032505	020063	047504	

9506	076302	051505	047040	052117	
9507	076310	043440	020117	044510	
9508	076316	044107	047440	020122	
9509	076324	046524	041103	050040	
9510	076332	051111	020121	047504	
9511	076340	051505	047040	052117	
9512	076346	043440	020117	047514	
9513	076354	000127			
9514	076356	044505	044124	051105	EM455: .ASCII /EITHER SSRA PS RESTORE IS STUCK HIGH OR IT IS NOT/<CRLF>
9515	076364	051440	051123	020101	
9516	076372	051520	051040	051505	
9517	076400	047524	042522	044440	
9518	076406	020123	052123	041525	
9519	076414	020113	044510	044107	
9520	076422	047440	020122	052111	
9521	076430	044440	020123	047516	
9522	076436	100124			
9523	076440	042507	052124	047111	.ASCIIZ /GETTING THRU RACK E63(BD) AS A LOW/
9524	076446	020107	044124	052522	
9525	076454	051040	041501	020113	
9526	076462	033105	024063	030102	
9527	076470	020051	051501	040440	
9528	076476	046040	053517	000	
9529	076503	124	041515	020103	EM456: .ASCIIZ /TMCC E6(11) DOES NOT GO HIGH ON DATI OR DATO/
9530	076510	032505	030450	024461	
9531	076516	042040	042517	020123	
9532	076524	047516	020124	047507	
9533	076532	044040	043511	020110	
9534	076540	047117	042040	052101	
9535	076546	020111	051117	042040	
9536	076554	052.01	000117		
9537	076560	051520	020127	044502	EM457: .ASCIIZ /PSW BIT 11 DOES NOT CLEAR/
9538	076566	020124	030461	042040	
9539	076574	042517	020123	047516	
9540	076602	020124	046103	040505	
9541	076610	000122			
9542	076612	051520	020127	044103	EM460: .ASCIIZ /PSW CHANGED ON RESET IN KERNEL MODE/
9543	076620	047101	042507	020104	
9544	076626	047117	051040	051505	
9545	076634	052105	044440	020116	
9546	076642	042513	047122	046105	
9547	076650	046440	042117	000105	
9548	076656	051520	020127	044103	EM461: .ASCIIZ /PSW CHANGES ON RESET IN SUPER MODE/
9549	076664	047101	042507	020123	
9550	076672	047117	051040	051505	
9551	076700	052105	044440	020116	
9552	076706	052523	042520	020122	
9553	076714	047515	042504	000	
9554	076721	115	046505	046440	EM703: .ASCIIZ /MEM MGT TESTS DISABLED/<CRLF>
9555	076726	052107	052040	051505	
9556	076734	051524	042040	051511	
9557	076742	041101	042514	100104	
9558	076750	000			
9559	076751	103	041501	042510	EM704: .ASCIIZ /CACHE TESTS DISABLED/<CRLF>
9560	076756	052040	051505	051524	
9561	076764	042040	051511	041101	

D01

9562	076772	042514	100104	000	
9563	076777	125	044516	052502	EM706: .ASCIZ /UNIBUS PE TEST DISABLED/<CRLF>
9564	077004	020123	042520	052040	
9565	077012	051505	020124	044504	
9566	077020	040523	046102	042105	
9567	077026	000200			
9568	077030	047516	041040	020122	EM710: .ASCIZ /NO BR 5 DEVICE/<CRLF>
9569	077036	020065	042504	044526	
9570	077044	042503	000200		
9571	077050	047516	041040	020122	EM711: .ASCIZ /NO BR 6 DEVICE/<CRLF>
9572	077056	020066	042504	044526	
9573	077064	042503	000200		
9574	077070	051102	032040	052040	EM712: .ASCIZ /BR 4 TESTS DISABLED/<CRLF>
9575	077076	051505	051524	042040	
9576	077104	051511	041101	042514	
9577	077112	100104	000		
9578	077115	102	020122	020065	EM713: .ASCIZ /BR 5 TESTS DISABLED/<CRLF>
9579	077122	042524	052123	020123	
9580	077130	044504	040523	046102	
9581	077136	042105	000200		
9582	077142	051102	033040	052040	EM715: .ASCIZ /BR 6 TESTS DISABLED/<CRLF>
9583	077150	051505	051524	042040	
9584	077156	051511	041101	042514	
9585	077164	100104	000		
9586	077167	117	051120	052040	EM717: .ASCIZ /OPR TEST DISABLED/<CRLF>
9587	077174	051505	020124	044504	
9588	077202	040523	046102	042105	
9589	077210	000200			
9590	077212	047514	045517	040440	EM720: .ASCII /LOOK AT THE CONSOLE LIGHTS/<CRLF>
9591	077220	020124	044124	020105	
9592	077226	047503	051516	046117	
9593	077234	020105	044514	044107	
9594	077242	051524	200		
9595	077245	124	042510	042040	.ASCII /THE DATA LIGHTS SHOULD READ 166667/<CRLF>
9596	077252	052101	020101	044514	
9597	077260	044107	051524	051440	
9598	077266	047510	046125	020104	
9599	077274	042522	042101	030440	
9600	077302	033066	033066	100067	
9601	077310	044124	020105	042101	.ASCIZ /THE ADDRESS LIGHTS SHOULD READ /
9602	077316	051104	051505	020123	
9603	077324	044514	044107	051524	
9604	077332	051440	047510	046125	
9605	077340	020104	042522	042101	
9606	077346	020040	000		
9607	077351	200	044103	047101	EM721: .ASCIZ <CRLF>/CHANGE SWITCH 7 TO CONTINUE/<CRLF>
9608	077356	042507	051440	044527	
9609	077364	041524	020110	020067	
9610	077372	047524	041440	047117	
9611	077400	044524	052516	100105	
9612	077406	000			
9613	077407	200	054524	042520	EM722: .ASCIZ <CRLF>/TYPE A CHARACTER TO CONTINUE/<CRLF>
9614	077414	040440	041440	040510	
9615	077422	040522	052103	051105	
9616	077430	052040	020117	047503	
9617	077436	052116	047111	042525	

E01

9618	077444	000200				
9619	077446	047200	020117	040515	EM724:	.ASCIZ <CRLF>/NO MAP REGISTERS AVAILABLE FOR TEST 75/<CRLF>
9620	077454	020120	042522	044507		
9621	077462	052123	051105	020123		
9622	077470	053101	044501	040514		
9623	077476	046102	020105	047506		
9624	077504	020122	042524	052123		
9625	077512	033440	100065	000		
9626	077517	116	020117	020114	EM725:	.ASCIZ /NO L CLOCK/<CRLF>
9627	077524	046103	041517	100113		
9628	077532	000				
9629		077534				
9630	077534	077534			AORTAB:	.EVEN
9631	077536	005474				.WORD
9632	077540	005656				TST1
9633	077542	006152				TST2
9634	077544	006434				TST3
9635	077546	007176				TST4
9636	077550	007312				TST5
9637	077552	007372				TST6
9638	077554	007456				TST7
9639	077556	010300				TST10
9640	077560	010440				TST11
9641	077562	011002				TST12
9642	077564	011116				TST13
9643	077566	012576				TST14
9644	077570	012744				TST15
9645	077572	013116				TST16
9646	077574	013304				TST17
9647	077576	013444				TST20
9648	077600	013560				TST21
9649	077602	013764				TST22
9650	077604	014334				TST23
9651	077606	014472				TST24
9652	077610	014630				TST25
9653	077612	015000				TST26
9654	077614	015154				TST27
9655	077616	015354				TST30
9656	077620	015526				TST31
9657	077622	016646				TST32
9658	077624	017136				TST33
9659	077626	017266				TST34
9660	077630	017416				TST35
9661	077632	020010				TST36
9662	077634	020424				TST37
9663	077636	021160				TST40
9664	077640	021424				TST41
9665	077642	022034				TST42
9666	077644	022470				TST43
9667	077646	022624				TST44
9668	077650	022676				TST45
9669	077652	023360				TST46
9670	077654	026146				TST47
9671	077656	026576				TST50
9672	077660	027012				TST51
9673	077662	027162				TST52
						TST53

F01

PDP 11/70 CPU DIAGNOSTIC PART 2 MACY11 27(732) 21-DEC-76 16:00 PAGE 177
DEKBBC.P11 POWER DOWN AND UP ROUTINES

SEQ 0212

9674	077664	027432	TST54
9675	077666	027576	TST55
9676	077670	030032	TST56
9677	077672	030160	TST57
9678	077674	030370	TST60
9679	077676	030446	TST61
9680	077700	031150	TST62
9681	077702	031264	TST63
9682	077704	031760	TST64
9683	077706	032164	TST65
9684	077710	032330	TST66
9685	077712	032512	TST67
9686	077714	033010	TST70
9687	077716	033552	TST71
9688	077720	033746	TST72
9689	077722	034166	TST73
9690	077724	034334	TST74
9691	077726	035166	TST75
9692	077730	035502	TST76
9693	077732	035742	
9694		000001	

SUBTAB: \$EOP
.END

GO1

PDP 11/70 CPU DIAGNOSTIC PART 2 MACY11 27(732) 21-DEC-76 16:00 PAGE 179
 DEKB8C.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0213

ADRTAB	077534	6530	6540	9630#										
BINARY	040154	6460*	6463*	6465	6471#									
BIT0 =	000001	149#	1964	1994	2396	5362	5444	5482	5494	5510	5544	5665	5981	6550
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	6240										
BIT09 =	001000	130#	140	6248	6314									
BIT1 =	000002	148#	3580	3591	3593	3856	5574							
BIT10 =	002000	129#	2915	2959	4276	6297								
BIT11 =	004000	128#	2948	2993	4292	4657	4658	4660	4665	4674	4676	4688	4690	4702
		4713	4719	4723	4729	4731	4733	4740	4743	4757	4794	4796	5843	6255
BIT12 =	010000	127#	2863	2981	3028	4310	4407	4438	4455	4946	5017	6132		
BIT13 =	020000	126#	3015	3070	4325	4475	4523	4540	6159	6187	6304			
BIT14 =	040000	125#	2871	3052	3105	4367	4490	4558	4588	4848	4850	4859	4862	4870
		4882	4905	4965	4970	4976	5012	5035	5081	5083	5211	5543	5821	
BIT15 =	100000	124#	1700	1701	1760	1768	1811	1826	1836	1856	1859	1885	1992	2021
		2035	2099	2170	2217	2478	2485	2605	2613	2615	2643	2691	2728	2880
		3055	3092	3169	4382	4505	4573	4603	4872	4874	4884	4887	4920	4927
		5039	5050	5062	5275	5498								
BIT2 =	000004	147#	3715	3718	3850	3869	5568	5590						
BIT3 =	000010	146#	3517	3520	3820	3841	3859	5303	5560	5577	5811			
BIT4 =	000020	145#	1656	1691	1750	2797	2845	3127	3138	3163	3179	3182	3328	3357
		3393	3426	3460	3538	3602	3639	4033	4068	4214	4345	4401	4432	5007
		5190												
BIT5 =	000040	144#	5452	5455										
BIT6 =	000100	143#	1549	1642	1648	1649	1667	1669	1687	3282	3609	3612	3615	3944
		3958	3997	4625	4631	5245	5259	6065	6070	6515				
BIT7 =	000200	142#	5217	6492										
BIT8 =	000400	141#												
BIT9 =	001000	140#	2852	2856	2927	3582	3873	4244	4260					
BPTVEC=	000014	156#	2761*											
BUFFDP	037560	6395#	6401											
CACHE	033660	5314	5672	5685#										
CACHSP	036446	1552	5796	5913	6181#									
CACHVE=	000114	163#	1552*	1553*	5710*	5767*	5860*	5861*	5887*	5888*	5913*	5979*		
CHAINQ	040262	6121	6500#											
CONTRL=	177746	173#	1490*	1558*	5708*									
CPUERR=	177766	186#	1554*	2895*	3138	3179	3181	3184*	3185	3244*	3254*	3258*	3517	3519
		3523*	3524	3715	3717	3721*	3722	3850	3859	3885*	3937*	3944	3947*	3958
		3997	4000*	4615*	5217	5219	5223*	5224	5452	5454	5456*	5459*	5460	5519*
		5568	5577	5613*	5915*	6165	6178*							
CPUSPU	036242	1550	1654	2774	2887	2893	2920	2953	2982	2986	2990	3017	3021	3024
		3053	3062	3067	3094	3098	3101	3177	3243	3255	3257	3485	3494	3506
		3652	3656	3662	3676	3687	3700	3710	3714	3861	3899	3928	3932	3953
		3957	3973	3990	3996	4265	4281	4297	4460	4467	4545	4614	5227	5400
		5582	5630	5750	5795	5912	6153#							
CR =	000015	61#	6644	6654										
CRLF =	000200	62#	1534	6617	6654	6878	6955	6972	7008	7039	7088	7134	7149	7165
		7208	7269	7366	7436	7599	7625	7647	7673	7698	7713	7721	7729	7878
		7907	7925	7958	7967	7976	7999	8012	8026	8045	8057	8070	8084	8108

EM134	051071	845	7532#					
EM135	051140	848	7539#					
EM136	051147	851	7541#					
EM14	042665	598	6950#					
EM140	051203	857	881	7546#				
EM141	051241	860	7552#					
EM142	051300	863	7558#					
EM143	051326	866	7562#					
EM144	051363	869	7567#					
EM145	051412	872	7571#					
EM146	051452	875	7577#					
EM147	051512	878	7583#					
EM15	043012	602	6967#					
EM151	051543	884	7588#					
EM152	051556	887	7590#					
EM155	051605	896	7594#					
EM156	051636	899	7599#					
EM157	051767	902	7615#					
EM16	043124	605	890	6981#				
EM160	052020	905	7620#					
EM161	052051	908	7625#					
EM162	052157	911	7638#					
EM163	052210	914	7643#					
EM164	052330	918	7658#					
EM165	052361	921	7663#					
EM166	052452	924	7673#					
EM167	052566	927	7686#					
EM17	043161	608	6986#					
EM170	052630	930	7692#					
EM171	052672	933	7698#					
EM172	053022	936	7713#					
EM174	053304	942	7745#					
EM175	053375	945	954	963	972	993	7755#	
EM176	053412	948	7758#					
EM177	053454	951	7764#					
EM2	042220	568	6896#					
EM20	043245	611	6995#					
EM201	053545	957	7774#					
EM202	053654	960	7789#					
EM204	053745	966	7799#					
EM205	054013	969	7806#					
EM207	054105	975	7816#					
EM21	043337	614	7005#					
EM210	054153	978	7823#					
EM211	054230	981	7831#					
EM212	054320	984	7841#					
EM213	054373	987	7849#					
EM214	054441	990	7856#					
EM216	054532	996	7866#					
EM217	054600	999	7873#					
EM22	043360	617	7008#					
EM220	054635	1002	7878#					
EM221	054725	1005	7889#					
EM222	054762	1008	7894#					
EM223	055051	1011	7904#					
EM224	055072	1014	7907#					

K01

SEQ 0217

EM225	055241	1017	7925#	
EM226	055352	1020	7938#	
EM227	055441	1023	7948#	
EM23	043456	620	7019#	
EM230	055531	1026	7958#	
EM232	056016	1032	7990#	
EM233	056077	1035	7999#	
EM234	056203	1038	8012#	
EM235	056317	1041	8026#	
EM237	056474	1047	8045#	
EM24	043547	623	7029#	
EM240	056576	1050	8057#	
EM241	056707	1053	8070#	
EM242	057027	1056	8084#	
EM243	057172	1059	8101#	
EM244	057336	1063	8120#	
EM245	057420	1066	8130#	
EM246	057522	1069	8142#	
EM247	057653	1072	8158#	
EM25	043603	626	682	7034#
EM250	057774	1075	8172#	
EM251	060156	1078	8194#	
EM252	060335	1081	8213#	
EM253	060406	1084	8220#	
EM254	060457	1087	8227#	
EM255	060560	1090	8238#	
EM256	060721	1093	8255#	
EM257	060757	1096	8261#	
EM26	043724	630	7050#	
EM260	061031	1099	8269#	
EM261	061103	1102	8277#	
EM262	061156	1105	8285#	
EM263	061227	1108	6371	8292#
EM264	062040	1111	8370#	
EM265	062063	1114	8375#	
EM266	062401	1117	8412#	
EM267	062527	1120	8427#	
EM27	043765	633	7056#	
EM270	062617	1123	8437#	
EM271	062725	1126	8450#	
EM272	063066	1129	8467#	
EM273	063125	1132	8473#	
EM274	063174	1135	8480#	
EM275	063214	1138	8483#	
EM276	063500	1141	8526#	
EM3	042241	571	6899#	
EM30	044030	637	7062#	
EM300	063632	1147	8531#	
EM301	063731	1150	8542#	
EM302	064026	1153	8553#	
EM303	064103	1156	8561#	
EM304	064245	1159	8579#	
EM305	064342	1162	8590#	
EM306	064440	1165	8601#	
EM307	064573	1168	8618#	
EM31	044066	640	7067#	

EM310	064647	1171	8626#
EM311	064722	1174	8634#
EM312	064774	1177	8641#
EM313	065131	1180	8658#
EM314	065202	1183	8665#
EM315	065323	1186	8680#
EM316	065455	1189	8697#
EM317	065600	1192	8712#
EM32	044131	643	7073#
EM320	065620	1195	8715#
EM321	065746	1198	8730#
EM322	066017	1201	8737#
EM323	066104	1204	8746#
EM324	066223	1207	8761#
EM325	066313	1210	8771#
EM326	066400	1213	8780#
EM327	066470	1216	8790#
EM33	044176	646	7080#
EM330	066644	1219	8808#
EM331	066766	1222	8823#
EM332	067112	1225	8837#
EM333	067203	1228	8847#
EM334	067237	1231	8852#
EM335	067273	1234	8857#
EM336	067330	1237	8862#
EM337	067400	1240	8869#
EM34	044214	649	7083#
EM340	067512	1243	8884#
EM341	067564	1246	8891#
EM342	067632	1249	8898#
EM343	067700	1252	8905#
EM344	067734	1255	8910#
EM345	067771	1258	8915#
EM346	070025	1261	8920#
EM347	070106	1264	8929#
EM35	044247	652	7088#
EM350	070167	1267	8938#
EM351	070245	1270	8946#
EM352	070323	1273	8954#
EM353	070377	1276	8962#
EM354	070453	1279	8970#
EM355	070505	1282	8975#
EM356	070564	1285	8983#
EM357	070617	1288	8988#
EM36	044375	655	7104#
EM360	070762	1292	9007#
EM361	071042	1295	9015#
EM362	071112	1298	9022#
EM363	071150	1301	9027#
EM364	071221	1304	9034#
EM365	071303	1307	9043#
EM366	071341	1310	9049#
EM367	071376	1313	9054#
EM37	044470	658	7114#
EM370	071522	1316	9069#
EM371	071650	1319	9084#

MO1

SEQ 0219

EM372	071755	1322	9097#
EM373	072106	1325	9113#
EM374	072220	1328	9126#
EM375	072311	1331	9136#
EM4	042266	574	6903#
EM40	044540	661	7121#
EM400	072475	1340	9156#
EM401	072607	1343	9170#
EM402	072655	1346	9177#
EM403	072720	1349	9183#
EM405	073034	1355	9197#
EM406	073163	1358	9213#
EM41	044610	664	7128#
EM410	073325	1364	9231#
EM411	073432	1367	9244#
EM412	073502	1370	9251#
EM413	073560	1373	9259#
EM414	073634	1376	9267#
EM416	073716	1382	9276#
EM417	073772	1385	9284#
EM42	044744	668	7145#
EM420	074073	1388	9296#
EM421	074151	1391	9304#
EM422	074252	1394	9316#
EM423	074474	1397	9342#
EM424	074576	1400	9354#
EM425	074665	1403	9364#
EM426	074704	1406	9367#
EM427	074771	1409	9376#
EM43	045064	672	7160#
EM430	075105	1412	9389#
EM431	075151	1415	9396#
EM432	075216	1418	9403#
EM433	075312	1421	9414#
EM435	075361	1427	9421#
EM436	075512	1430	9436#
EM437	075556	1433	9442#
EM44	045246	676	7182#
EM440	075627	1436	9449#
EM441	075703	1439	9457#
EM442	075754	1442	9464#
EM443	076032	1445	9472#
EM444	076050	1448	9475#
EM445	076066	1451	9478#
EM446	076104	1454	9481#
EM447	076122	1457	9484#
EM45	045313	679	7189#
EM450	076140	1460	9487#
EM451	076156	1463	9490#
EM452	076174	1466	9493#
EM453	076212	1469	9496#
EM454	076260	1472	9503#
EM455	076356	1475	9514#
EM456	076503	1478	9529#
EM457	076560	1481	9537#
EM460	076612	1484	9542#

ITEM50	001644	688#
ITEM51	001652	691#
ITEM52	001660	695#
ITEM53	001666	698#
ITEM54	001674	701#
ITEM55	001702	704#
ITEM56	001710	707#
ITEM57	001716	710#
ITEM6	001330	580#
ITEM60	001724	713#
ITEM61	001732	716#
ITEM62	001740	719#
ITEM63	001746	722#
ITEM64	001754	725#
ITEM65	001762	728#
ITEM66	001770	731#
ITEM67	001776	734#
ITEM7	001336	583#
ITEM70	002004	737#
ITEM71	002012	740#
ITEM72	002020	743#
ITEM73	002026	746#
ITEM74	002034	749#
ITEM75	002042	752#
ITEM76	002050	755#
ITEM77	002056	758#
ITE100	002064	761#
ITE101	002072	764#
ITE102	002100	767#
ITE103	002106	770#
ITE104	002114	773#
ITE105	002122	776#
ITE106	002130	779#
ITE107	002136	782#
ITE110	002144	785#
ITE111	002152	788#
ITE112	002160	791#
ITE113	002166	794#
ITE114	002174	797#
ITE115	002202	800#
ITE116	002210	803#
ITE117	002216	806#
ITE120	002224	809#
ITE121	002232	812#
ITE122	002240	815#
ITE123	002246	818#
ITE124	002254	821#
ITE125	002262	824#
ITE126	002270	827#
ITE127	002276	830#
ITE130	002304	833#
ITE131	002312	836#
ITE132	002320	839#
ITE133	002326	842#
ITE134	002334	845#
ITE135	002342	848#

ITE136	002350	851#
ITE137	002356	854#
ITE140	002364	857#
ITE141	002372	860#
ITE142	002400	863#
ITE143	002406	866#
ITE144	002414	869#
ITE145	002422	872#
ITE146	002430	875#
ITE147	002436	878#
ITE150	002444	881#
ITE151	002452	884#
ITE152	002460	887#
ITE153	002466	890#
ITE154	002474	893#
ITE155	002502	896#
ITE156	002510	899#
ITE157	002516	902#
ITE160	002524	905#
ITE161	002532	908#
ITE162	002540	911#
ITE163	002546	914#
ITE164	002554	918#
ITE165	002562	921#
ITE166	002570	924#
ITE167	002576	927#
ITE170	002604	930#
ITE171	002612	933#
ITE172	002620	936#
ITE173	002626	939#
ITE174	002634	942#
ITE175	002642	945#
ITE176	002650	948#
ITE177	002656	951#
ITE200	002664	954#
ITE201	002672	957#
ITE202	002700	960#
ITE203	002706	963#
ITE204	002714	966#
ITE205	002722	969#
ITE206	002730	972#
ITE207	002736	975#
ITE210	002744	978#
ITE211	002752	981#
ITE212	002760	984#
ITE213	002766	987#
ITE214	002774	990#
ITE215	003002	993#
ITE216	003010	996#
ITE217	003016	999#
ITE220	003024	1002#
ITE221	003032	1005#
ITE222	003040	1008#
ITE223	003046	1011#
ITE224	003054	1014#
ITE225	003062	1017#

E02

ITE226	003070	1020#
ITE227	003076	1023#
ITE230	003104	1026#
ITE231	003112	1029#
ITE232	003120	1032#
ITE233	003126	1035#
ITE234	003134	1038#
ITE235	003142	1041#
ITE236	003150	1044#
ITE237	003156	1047#
ITE240	003164	1050#
ITE241	003172	1053#
ITE242	003200	1056#
ITE243	003206	1059#
ITE244	003214	1063#
ITE245	003222	1066#
ITE246	003230	1069#
ITE247	003236	1072#
ITE250	003244	1075#
ITE251	003252	1078#
ITE252	003260	1081#
ITE253	003266	1084#
ITE254	003274	1087#
ITE255	003302	1090#
ITE256	003310	1093#
ITE257	003316	1096#
ITE260	003324	1099#
ITE261	003332	1102#
ITE262	003340	1105#
ITE263	003346	1108#
ITE264	003354	1111#
ITE265	003362	1114#
ITE266	003370	1117#
ITE267	003376	1120#
ITE270	003404	1123#
ITE271	003412	1126#
ITE272	003420	1129#
ITE273	003426	1132#
ITE274	003434	1135#
ITE275	003442	1138#
ITE276	003450	1141#
ITE277	003456	1144#
ITE300	003464	1147#
ITE301	003472	1150#
ITE302	003500	1153#
ITE303	003506	1156#
ITE304	003514	1159#
ITE305	003522	1162#
ITE306	003530	1165#
ITE307	003536	1168#
ITE310	003544	1171#
ITE311	003552	1174#
ITE312	003560	1177#
ITE313	003566	1180#
ITE314	003574	1183#
ITE315	003602	1186#

ITE316	003610	1189#
ITE317	003616	1192#
ITE320	003624	1195#
ITE321	003632	1198#
ITE322	003640	1201#
ITE323	003646	1204#
ITE324	003654	1207#
ITE325	003662	1210#
ITE326	003670	1213#
ITE327	003676	1216#
ITE330	003704	1219#
ITE331	003712	1222#
ITE332	003720	1225#
ITE333	003726	1228#
ITE334	003734	1231#
ITE335	003742	1234#
ITE336	003750	1237#
ITE337	003756	1240#
ITE340	003764	1243#
ITE341	003772	1246#
ITE342	004000	1249#
ITE343	004006	1252#
ITE344	004014	1255#
ITE346	004030	1261#
ITE347	004036	1264#
ITE350	004044	1267#
ITE351	004052	1270#
ITE352	004060	1273#
ITE353	004066	1276#
ITE354	004074	1279#
ITE355	004102	1282#
ITE356	004110	1285#
ITE357	004116	1288#
ITE360	004124	1292#
ITE361	004132	1295#
ITE362	004140	1298#
ITE363	004146	1301#
ITE364	004154	1304#
ITE365	004162	1307#
ITE366	004170	1310#
ITE367	004176	1313#
ITE370	004204	1316#
ITE371	004212	1319#
ITE372	004220	1322#
ITE373	004226	1325#
ITE374	004234	1328#
ITE375	004242	1331#
ITE376	004250	1334#
ITE377	004256	1337#
ITE400	004264	1340#
ITE401	004272	1343#
ITE402	004300	1346#
ITE403	004306	1349#
ITE404	004314	1352#
ITE405	004322	1355#
ITE406	004330	1358#

ITE407	004336	1361#
ITE410	004344	1364#
ITE411	004352	1367#
ITE412	004360	1370#
ITE413	004366	1373#
ITE414	004374	1376#
ITE415	004402	1379#
ITE416	004410	1382#
ITE417	004416	1385#
ITE420	004424	1388#
ITE421	004432	1391#
ITE422	004440	1394#
ITE423	004446	1397#
ITE424	004454	1400#
ITE425	004462	1403#
ITE426	004470	1406#
ITE427	004476	1409#
ITE430	004504	1412#
ITE431	004512	1415#
ITE432	004520	1418#
ITE433	004526	1421#
ITE434	004534	1424#
ITE435	004542	1427#
ITE436	004550	1430#
ITE437	004556	1433#
ITE440	004564	1436#
ITE441	004572	1439#
ITE442	004600	1442#
ITE443	004606	1445#
ITE444	004614	1448#
ITE445	004622	1451#
ITE446	004630	1454#
ITE447	004636	1457#
ITE450	004644	1460#
ITE451	004652	1463#
ITE452	004660	1466#
ITE453	004666	1469#
ITE454	004674	1472#
ITE455	004702	1475#
ITE456	004710	1478#
ITE457	004716	1481#
ITE460	004724	1484#
ITE461	004732	1487#
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#

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#	1491#
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#	1493#
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#	5954
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#	

K02

PDP 11/70 CPU DIAGNOSTIC PART 2 MACY11 27(732) 21-DEC-76 16:00 PAGE 196

DEKBBC.P11 CROSS REFERENCE TABLE -- USER SYMBOLS

SEQ 0230

PS = 177776
PSM = 177776

PWRVEC= 000024
QUIT 040224
RESVEC= 000010

RKCS1 001250
RKVEC 001252
RPCS1 001244
RPVEC 001246
RSCS1 001240
RSVEC 001242
RO =%000000

RI =%000001

51#	52	1495*	6130										
52#	1590	1598	1630	1643*	1646	1656	1690	1691	1779	1817	1832	1855	
1969	1998	2026	2058	2070	2103	2152	2190	2324	2341	2421	2454	2484	
2578	2603*	2611	2683	2696	2733	2845	3127	3163	3328	3357	3393	3426	
3459	3460	3602	3638	3639	3673	4019	4031	4214	4345	4401	4432	4657*	
4658	4660*	4664*	4665	4674*	4676*	4688*	4690*	4702*	4713*	4719*	4723*	4729*	
4731*	4733*	4740*	4743*	4757*	4794*	4796*	4815*	4816	4819	4822*	4823	4826	
4829*	4830	4833	4848*	4850*	4859*	4862*	4870*	4872*	4874*	4877*	4882*	4884*	
4887*	4890*	4892*	4905*	4911*	4916*	4918	4920*	4927*	4930*	4946*	4965*	4970*	
4976*	4980*	4982*	4986*	4989*	5006	5007	5012*	5014*	5017*	5035*	5039*	5047*	
5050*	5053*	5056*	5059*	5062*	5065*	5068*	5070*	5081*	5083*	5084	5110	5121	
5123	5124*	5129*	5134	5136	5137*	5143*	5148	5150	5151*	5156*	5160	5162	
5164*	5169*	5173	5175	5180*	5182	5186*	5211*	5214*	5543*	5546*	5580*	5598*	
5614*	5620*	5627*											
158#	1507*	1508*	2840*	2888*	3646*	3741*	6835*	6836*	6844*	6859*	6860*		
1546	6068	6490#											
153#	1516*	1523*	1587*	1609*	1698*	1726*	1757*	1843*	1882*	1893*	1897*	1915*	
1919*	1921*	1936*	1941*	1943*	2093*	2113*	2237*	2251*	2602*	2625*	2641*	2666*	
2668*	2687*	2708*	2723*	2742*	4025*	4070*	4075*	4077*	4085*	4091*	4097*	4106*	
4114*	4125*	4134*	4140*	4149*	4158*	4165*	4170*	4171*	5355*	5356*	5381*		
537#	3219	3220											
538#	3221												
535#	3213	3226											
536#	3227												
533#	3210	3223											
534#	3224												
65#	73	1541*	1545*	1647*	1649	1759*	1762	1787*	1790	1816	1827*	1831	
1848*	1854	1883*	1884*	1887	1894*	1896	1898	1916*	1917*	1918	1937*	1938*	
1940	1963*	1968*	1975	1978	1986	1992*	1997*	2000	2008	2021*	2025*	2036*	
2037	2044	2050*	2057*	2061	2066*	2069*	2073	2095*	2101*	2105	2110	2144*	
2151*	2153	2157	2166	2184*	2189*	2191	2195	2211*	2214*	2217*	2219*	2241*	
2248	2318*	2323*	2340*	2342	2362	2368	2372	2376	2380	2382	2392	2404*	
2407*	2414*	2420*	2422	2424	2428	2435	2447*	2453*	2455	2467*	2471*	2472	
2478*	2483*	2485	2497*	2501*	2502	2509	2515*	2521*	2522	2528	2534*	2540*	
2541	2543	2553*	2559*	2560	2562	2572*	2577*	2606*	2610*	2612	2615*	2691*	
2695	2727*	2728*	2732	2743	2807*	2819*	3136*	3155	3266*	3273*	3275	3279*	
3286*	3292*	3300*	3303	3334*	3336*	3346*	3349*	3355*	3356*	3359*	3361*	3391*	
3392*	3395*	3397*	3424*	3425*	3428*	3430*	3469*	3470	3545*	3546	3578*	3579*	
3580*	3583	3591*	3594	3611	3649*	3650	3695*	3696	3706	3745*	3747	3752*	
3753	3826*	3827*	3841*	3844*	3856*	3866*	3869*	3877*	3878*	3879*	3880	3902	
3922*	3924*	3927*	3941*	3942*	3987*	3989	4048*	4054*	4055	4108*	4116*	4343*	
4344*	4347*	4349*	4399*	4400*	4403*	4405*	4635*	4636*	4641*	4642*	4945*	4950*	
4954	4960	5318*	5319*	5320*	5321*	5322*	5323*	5324*	5325*	5326*	5327*	5330*	
5548*	5549*	5560*	5563*	5574*	5587*	5590*	5602*	5603*	5604*	5605	5633	5954*	
5957	5959*	5968*	5969*	5970*	5971*	5972*	5973	6034*	6113*	6122*	6125	6333*	
6334*	6337*	6338*	6339*	6340	6342*	6343*	6344	6349	6354*	6359	6361	6394*	
6395	6398*	6399	6419	6423*	6425	6433	6435*	6437	6491*	6492*	6493	6500*	
6504*	6607	6608*	6609	6612*	6746	6756*	6760	6776	6777	6790*	6809	6810*	
6811	6812*	6813*	6814*	6837	6858*								
66#	74	1542*	1544*	1705	1708*	1760*	1762*	1764	1768	1786*	1790*	1792	
1794	1798	1801	1811*	1816*	1826*	1831*	1836	1850*	1854*	1856	1858	1885*	
1887*	1889	1896*	1898*	1940*	1964*	1973	1987	1993*	2001	2006	2019*	2035*	
2041*	2042	2045	2052*	2062	2065*	2074	2096*	2106	2108	2111	2146*	2154	
2155	2159	2163	2170	2192	2199	2238*	2245*	2246	2249	2319*	2344	2354	
2358	2363	2383	2386	2390	2393	2405*	2408	2415*	2429	2433	2436	2448*	
2468*	2479*	2498*	2505	2510	2516*	2523	2524	2535*	2542	2547	2554*	2561	

	4309	4324	4350	4406	4437	4459	4466	4546	4551	4593	4608	4612	4714	
	4720	4727	4847	4853	4858	4863	4865	4943	4948	4951	4953	4962	4978	
	5015	5034	5209	5243	5256	5265	5274	5361	5404	5414	5442	5481	5502	
	5511	5545	5597	5612	5629	5670	5716	5770	5826	5844	5880	5892	5957	
	5961	5980	6154	6182										
START	004740	445	1490*											
STKLMT=	177774	53*	3682*	3686*	3692*	3743*	3747	3750	3751*	3753	3757	3759*	3765*	3767*
		3780*	3878	3889*	3897*	5496*	5501*	5506*	5603	5619*	5628*			
STYPE	037542	6391*	6405											
SUBTAB	077732	1542	6501	9693*										
SUPSTK=	000700	47*												
SWR	= 177570	55*	56	3205	3233	3259	3317	3319	3379	3381	3412	3414	3920	3973
		4223	4225	4233	4235	4420	4422	5301	5303	5307	5311	5595	5685	5687
		5601	5695	5809	5811	5924	5926	5930	5934	5964	6028	6036	6044	6053
		6132	6159	6187	6226*	6240	6242	6248	6255	6294	6297	6304	6308	6314
SW0	= 000001	121*	6028											
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*	5311	5695	5934									
SW12	= 010000	99*												
SW13	= 020000	98*												
SW14	= 040000	97*	3205	5307	5691	5930	5964							
SW15	= 100000	96*												
SW2	= 000004	119*	5687											
SW3	= 000010	118*												
SW4	= 000020	117*	3319	4235										
SW5	= 000040	116*	3381	4225										
SW6	= 000100	115*	3414	4422										
SW7	= 000200	114*	6036	6044	6053									
SW8	= 000400	113*	3317	3379	3412	4223	4233	4420	5301	5685	5809	5924		
SW9	= 001000	112*	5595											
SYSTID=	177764	185*												
TAPE1	= *****	U	6483											
TAPE2	= *****	U	6483											
TBITVE=	000014	154*	1513*	1514*	2775*	4024*	4052*	4066*	5254*	5266*	5270*	5751*		
TERM	037774	6436	6445*											
THRESP	040217	6467	6480*											
TKVEC	= 000060	161*	1546*	1547*	6064*	6068*								
TMCS1	001254	539*	3216	3229										
TMVEC	001256	540*	3230											
TPVEC	= 000064	162*	1633*	3327*	3330*	3332*	3345*	3604*	3606*	3607*	4216*	4220*	4240*	4308*
		4322*	4342*	4364*	4379*	5241*	5255*	5272*						
TRAPVE=	000034	160*	1505*	1506*	5661*									
TR7VEC=	000014	155*												
TST1	005474	1557	1582*	9631										
TST10	007456	1933	1934	1942	1959*	9638								

TST11	010300	1960	2072	2089#	9639				
TST12	010440	2090	2091	2109	2140#	9640			
TST13	011002	2141	2220	2233#	9641				
TST14	011116	2234	2235	2247	2315#	9642			
TST15	012576	2316	2581	2599#	9643				
TST16	012744	2600	2601	2622	2639#	9644			
TST17	013116	2640	2667	2678#	9645				
TST2	005656	1583	1584	1607	1626#	9632			
TST20	013304	2679	2680	2705	2720#	9646			
TST21	013444	2721	2722	2739	2758#	9647			
TST22	013560	2759	2760	2791#	9648				
TST23	013764	2792	2793	2821	2837#	9649			
TST24	014334	2838	2839	2889	2909#	9650			
TST25	014472	2910	2929	2942#	9651				
TST26	014630	2943	2961	2975#	9652				
TST27	015000	2976	2996	3009#	9653				
TST3	006152	1628	1629	1663	1686#	9633			
TST30	015154	3010	3031	3046#	9654				
TST31	015354	3047	3073	3086#	9655				
TST32	015526	3087	3107	3124#	9656				
TST33	016646	3125	3201	3202	3245	3256	3263	3313#	9657
TST34	017136	3314	3325	3343	3344	3376#	9658		
TST35	017266	3377	3378	3387	3389	3400	3409#	9659	
TST36	017416	3410	3411	3420	3422	3433	3456#	9660	
TST37	020010	3525	3536#	9661					
TST4	006434	1696	1697	1721	1746#	9634			
TST40	020424	3635#	9662						
TST41	021160	3636	3723	3738#	9663				
TST42	021424	3739	3740	3807#	9664				
TST43	022034	3816	3817	3901	3916#	9665			
TST44	022470	3917	3991	3992	4012#	9666			
TST45	022624	4013	4014	4046#	9667				
TST46	022676	4047	4056	4064#	9668				
TST47	023360	4065	4207#	9669					
TST5	007176	1747	1863	1878#	9635				
TST50	026146	4208	4616	4654#	9670				
TST51	026576	4655	4656	4706	4709	4745	4753#	9671	
TST52	027012	4754	4755	4799	4813#	9672			
TST53	027162	4814	4831	4843#	9673				
TST54	027432	4844	4901#	9674					
TST55	027576	4902	4903	4939#	9675				
TST56	030032	4940	4999#	9676					
TST57	030160	5000	5001	5020	5032#	9677			
TST6	007312	1879	1880	1888	1911#	9636			
TST60	030370	5033	5079#	9678					
TST61	030446	5087	5109#	9679					
TST62	031150	5112	5191	5207#	9680				
TST63	031264	5208	5237#	9681					
TST64	031760	5238	5333	5349#	9682				
TST65	032164	5350	5351	5397#	9683				
TST66	032330	5398	5399	5434#	9684				
TST67	032512	5435	5437	5461	5472#	9685			
TST7	007372	1912	1913	1920	1932#	9637			
TST70	033010	5473	5530#	9686					
TST71	033552	5531	5658#	9687					
TST72	033746	5659	5686	5688	5705#	9688			

SBELL	001176	516#	6299	6322															
SCHARC	041136	6630*	6637	6646*	6651#														
SCLR.T	036140	6116	6119#																
SCMTAG	001100	481#	1495	1496	1503	1509	1510	1511											
SCM1 =	000003	507#	508#	509#	510#														
SCM2 =	000006	507#	508#	509#	510#														
SCM3 =	000003	505#	507																
SCM4 =	000004	510#	511#	512#	513#	514#													
SCRLF	001203	518#	6111	6307	6322	6332	6348	6353	6356	6387	6417	6446	6579	6620					
		6654																	
SDBLK	041604	6757	6791	6799#															
SDDAGN	036170	6093	6114	6123	6129#														
SOTBL	041574	6760	6795#																
SENDAD	036160	468	1529	6125#	6311														
SENDBR	016642	3287	3289	3301	3303#														
SENDCT	035776	1509	6095#																
SENULL	036236	6144#																	
SEOP	035742	6025	6031	6033	6085#	9693													
SEOPCT	035770	1509#	6092#	6096															
SEPIRQ	001216	524#	2822#	2930#	2962*	2997*	3032*	3074*	3108*	7656	7787								
SERFLG	001103	484#	6216	6244	6246	6252*	6273	6291*	6322										
SERMAX	001115	490#	1512*	6246	6268*	6273													
SERPSW	001206	520#	1590#	1591*	1592	1646*	1659	1661	1779*	1780*	1817*	1818*	1819	1832*					
		1833#	1834	1855*	1861*	1862	1969*	1970*	1971	1998*	1999*	2004	2026*	2028*					
		2029	2058#	2060*	2070*	2071*	2103*	2152*	2176*	2177	2190*	2203*	2204	2324*					
		2325*	2326	2328	2341*	2346*	2347	2396	2421*	2439*	2440	2454*	2459*	2460					
		2484*	2489*	2490	2578*	2579*	2580	2611*	2620*	2621	2696*	2703*	2704	2733*					
		2737*	2738	4031*	4033	4819*	4820*	4826*	4827*	4833*	4834*	5084*	5085*	5086					
		5113*	5123*	5136*	5150*	5162*	5175*	5182*	5183	6893	7158	7226							
SERROR	037124	1503	3974	3983	6289#														
SERRPC	001116	491#	6153*	6167	6181*	6195	6301*	6302*	6303	6322	6374	6893	6914	6965					
		7048	7143	7158	7179	7226	7656	7787	8118	8128	8191	8882							
SERRTB	001272	562#	6343																
SERTTL	001112	488#	6108	6112*	6300*	6322													
SESCAP	001174	515#	1511*	1583*	1623*	1696*	1879*	1912*	1933*	2090*	2234*	2600*	2679*	2721*					
		2759*	2792*	2838*	3201*	3343*	3377*	3410*	3739*	3816*	3822	3991*	4013*	4655*					
		4671*	4754*	4902*	5000*	5350*	5398*	5535*	5707*	5764*	5949*	6179*	6207*	6267*					
		6317	6319	6322															
SFILLC	001150	503#	6623	6654															
SFILLS	001147	502#	6654																
SGDADR	001120	492#																	
SGDOAT	001124	494#																	
SGET42	036122	6113#																	
SHD =	000001	1#	16	17															
STCNT	001104	485#	1627*	3315*	3818*	4209*	5239*	5533*	6259*	6260	6262*	6272							
SLLUP	041770	6835	6864#																
SSTEMB	001114	489#	6303*	6322	6334	6335													
SHW11L	016526	3240	3279#																
SHW11P	016572	3251	3292#																
SLF	001204	519#	6322	6654															
SLOOP	036230	6137	6141#																
SLPADR	001106	486#	1525*	1631*	1688*	1748*	2681*	2795*	3316*	3457*	3537*	3637*	3819*	4067*					
		4210*	5004*	5240*	5534*	6250*	6265*	6270	6272										
SLPERR	001110	487#	1526*	1632*	1689*	1749*	1785*	1810*	1825*	1847*	1991*	2018*	2034*	2049*					
		2064*	2183*	2210*	2336*	2403*	2413*	2446*	2466*	2477*	2496*	2514*	2533*	2552*					
		2571*	2656*	2682*	2796*	2923*	2955*	2988*	3023*	3065*	3100*	3132*	3151*	3160*					

MSG135	3370#	3372													
MSG136	3403#	3405													
MSG137	3436#	3438													
MSG140	3618#	3620													
MSG141	3726#	3728													
MSG142	3782#	3784													
MSG143	3904#	3906													
MSG144	4002#	4004													
MSG145	4039#	4041													
MSG146	4174#	4176													
MSG147	4646#	4648													
MSG150	4746#	4750													
MSG151	4807#	4809													
MSG152	4837#	4839													
MSG153	4895#	4897													
MSG154	4933#	4935													
MSG155	4991#	4993													
MSG156	5023#	5025													
MSG157	5074#	5076													
MSG160	5090#	5092													
MSG161	5338#	5340													
MSG162	5383#	5385													
MSG163	5424#	5426													
MSG164	5464#	5466													
MSG165	5521#	5523													
MSG166	5645#	5647													
MSG167	5699#	5701													
MSG170	5753#	5755													
MSG171	5798#	5800													
MSG172	5938#	5940													
MSG173	6011#	6013													
MS137A	3527#	3529													
MS145A	4058#	4060													
MS160A	5196#	5198													
MS160B	5229#	5231													
MULT	1#	431#													
MYTAGS	1#	520													
NEWTST	1#	4#	431#	1560	1614	1672	1733	1866	1902	1923	1945	2078	2115	2222	2254
	2584	2627	2670	2710	2747	2781	2824	2900	2933	2966	3000	3035	3077	3112	3306
	3370	3403	3436	3527	3618	3726	3782	3904	4002	4039	4058	4174	4646	4748	4807
	4837	4895	4933	4991	5023	5074	5090	5196	5229	5338	5383	5424	5464	5521	5645
	5699	5753	5798	5938	6011										
POP	1#	431#	6786	6853											
PUSH	1#	431#	6745	6837											
SAVEAD	1#	1583	1628	1696	1879	1912	1933	2090	2234	2600	2679	2721	2759	2792	2838
	3200	3342	3377	3410	3739	3816	3991	4013	4655	4754	4902	5000	5350	5398	5949
SCOPE	50#	1582	1626	1686	1746	1878	1911	1932	1959	2089	2140	2233	2315	2599	2639
	2678	2720	2758	2791	2837	2909	2942	2975	3009	3046	3086	3124	3313	3376	3409
	3456	3536	3635	3738	3807	3916	4012	4046	4064	4207	4654	4753	4813	4843	4901
	4939	4999	5032	5079	5109	5207	5237	5349	5397	5434	5472	5530	5658	5705	5762
	5807	5948	6024	6086											
SETTRA	6816#	6826	6827	6828	6829										
SETUP	1#	431#	1495												
SKIP	1#	431#	1557	1607	1662	1721	1863	1888	1920	1942	2072	2109	2220	2247	2581
	2622	2667	2705	2739	2821	2889	2929	2961	2996	3031	3073	3107	3186	3245	3256
	3262	3324	3386	3389	3400	3419	3422	3433	3525	3723	3901	4056	4616	4706	4709

K03

SEQ 0243

.\$DB20	1#			
.\$DB20	1#			
.\$DIV	1#			
.\$EOP	1#	4#	6072#	6073
.\$ERRO	1#	4#	6273	
.\$ERRT	1#	4#		
.\$MULT	1#			
.\$POME	1#	4#	6209#	6830
.\$RAND	1#			
.\$RDDE	1#			
.\$RDOC	1#			
.\$READ	1#			
.\$SAVE	1#			
.\$SB20	1#			
.\$SB20	1#			
.\$SCOP	1#	4#	6210	
.\$SIZE	1#			
.\$SUPR	1#			
.\$STRAP	1#	4#	6209#	6800
.\$STYPB	1#			
.\$STYPD	1#	4#	6209#	6732
.\$STYPE	1#	4#	6581	
.\$STYPO	1#	4#	6209#	6654
.\$1170	1#	4#	41	

ADC	1791	6463													
ADD	1522	2810	2815	2818	3273	3286	3300	3356	3392	3425	3884	3889	3890	3892	4639
	5313	5610	5618	5619	5697	5959	6343	6398	6403	6411	6423	6435	6549	6613	6683
	6693	6765													
ASH	1762	1790	1816	1831	1854	1887	1896	1918	1940	5971					
ASHC	1968	1997	2025	2041	2057	2069	2101								
ASLB	6770														
BCC	2102	6462	6771												
BCS	1763	1888	2059	2072	2215	2220									
BEQ	1530	1557	1593	1601	1603	1607	1611	1650	1657	1663	1692	1706	1714	1721	1751
	1793	1820	1835	1857	1863	1899	1976	2009	2030	2043	2158	2164	2178	2196	2200
	2205	2247	2327	2348	2381	2391	2397	2423	2434	2441	2456	2461	2473	2486	2491
	2503	2506	2525	2529	2544	2548	2563	2567	2581	2618	2622	2698	2705	2735	2739
	2744	2770	2798	2846	3128	3164	3180	3186	3206	3320	3329	3354	3358	3382	3389
	3394	3415	3422	3427	3461	3475	3518	3525	3539	3547	3555	3566	3585	3596	3603
	3640	3660	3716	3723	3748	3754	3763	3839	3851	3854	3863	3901	3959	4034	4056
	4069	4215	4226	4230	4236	4304	4346	4402	4433	4666	4682	4692	4696	4705	4708
	4712	4718	4735	4745	4793	4817	4824	4831	4864	4889	4910	4915	4949	4952	4961
	4967	4984	5008	5020	5061	5087	5122	5135	5149	5161	5174	5184	5191	5225	5304
	5312	5437	5453	5461	5475	5499	5558	5569	5572	5584	5596	5632	5672	5688	5696
	5812	5879	5909	5927	5935	6029	6037	6045	6054	6114	6123	6241	6243	6245	6249
	6258	6292	6298	6315	6318	6329	6331	6345	6350	6362	6397	6400	6531	6541	6616
	6649	6710													
BGE	6261														
BGT	6093	6717	6779												
BHI	3838	3853	3864	5557	5571	5585	6247	6542							
BIC	1591	1599	1648	1667	1669	1780	1818	1833	1861	1970	1999	2028	2060	2071	2176
	2203	2325	2346	2439	2459	2489	2579	2620	2703	2737	2768	4631	4660	4676	4690
	4713	4719	4723	4729	4733	4743	4794	4796	4820	4827	4834	4850	4862	4874	4887
	4927	4970	4976	5050	5062	5083	5085	5662	6090	6131	6427	6492	6550	6707	
BICB	3583	3594	3612	3615											
BIS	1642	2852	2863	2871	2880	2915	2927	2948	2959	2981	2993	3015	3052	3092	3169
	3841	3844	3856	3866	3869	4244	4260	4276	4292	4455	4540	4588	4603	4657	4674
	4688	4702	4731	4740	4757	4848	4859	4870	4872	4882	4884	4905	4920	4946	4965
	4980	4982	5012	5017	5035	5039	5059	5081	5211	5259	5479	5490	5543	5560	5563
	5574	5587	5590	5972	5981	6065	6136	6712	6713	6773	6774				
BISB	1549	1687	3609	5129	6070	6334	6515								
BIT	1649	1656	1691	1750	2396	2797	2845	3127	3138	3163	3205	3233	3259	3317	3319
	3328	3357	3379	3381	3393	3412	3414	3426	3460	3538	3602	3639	3820	3850	3859
	3873	3944	3958	3997	4033	4068	4214	4223	4225	4233	4235	4345	4401	4420	4422
	4432	4658	4665	5007	5190	5217	5301	5303	5307	5311	5498	5568	5577	5595	5685
	5687	5691	5695	5809	5811	5824	5926	5930	5934	5964	6028	6036	6044	6053	6132
	6159	6187	6240	6248	6255	6297	6304	6314							
BITB	3611	6637													
BLT	6628	6718	6762	6778											
BMI	1709	1776	1890	2109	2614	2648	2662	2701	4919	6135	6227	6769			
BNE	1499	1528	1538	1540	1728	1765	1769	1777	1795	1799	1806	1837	1972	2005	2007
	2156	2160	2167	2171	2329	2343	2345	2355	2359	2369	2373	2377	2337	2409	2425
	2650	2806	2812	2814	3139	3145	3204	3234	3260	3271	3284	3298	3318	3322	3337
	3350	3380	3384	3413	3417	3666	3698	3708	3756	3769	3772	3775	3821	3860	3874
	3876	3881	3888	3894	3945	3976	3998	4224	4234	4340	4396	4421	4423	4425	4470
	4518	4553	4624	4637	4643	4659	4678	4765	4768	4771	4773	4776	4778	4781	4783
	4786	4788	4791	4799	4852	4876	4923	4955	5046	5052	5064	5218	5302	5306	5308
	5333	5578	5601	5606	5616	5622	5686	5690	5692	5810	5925	5929	5931	5963	5965
	6027	6033	6133	6160	6188	6256	6305	6312	6336	6355	6405	6409	6420	6434	6449
	6494	6506	6610	6618	6624	6638	6645	6708	6767	6852					

M03

PDP 11/70 CPU DIAGNOSTIC PART 2 MACY11 27(732) 21-DEC-76 16:00 PAGE 213
 DEKBBC.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

SEQ 0245

BPL	1640	1653	1974	6295	6309	6604	6642	6706	6753	6783					
BPT	2767														
BR	1521	1532	1660	1767	1771	1774	1778	1797	1804	1840	1842	1920	1942	1980	1983
	2011	2014	2162	2165	2169	2173	2175	2198	2202	2331	2351	2357	2361	2365	2371
	2375	2379	2385	2389	2395	2399	2427	2432	2438	2458	2488	2504	2527	2546	2565
	2652	2667	2821	2848	2858	2865	2873	2882	2889	2918	2929	2951	2961	2984	2996
	3019	3031	3058	3061	3073	3096	3107	3130	3141	3143	3149	3166	3173	3211	3214
	3217	3222	3225	3228	3232	3245	3256	3263	3287	3289	3301	3325	3331	3338	3360
	3364	3387	3396	3400	3420	3429	3433	3473	3493	3504	3508	3572	3605	3654	3658
	3664	3842	3845	3857	3867	3870	3931	3934	3943	3955	3963	3978	4219	4228	4238
	4348	4404	4427	4435	4616	4620	4622	4626	4629	4638	4640	4680	4694	4706	4709
	4716	4722	4805	4921	4929	4957	4959	4964	5049	5055	5067	5310	5337	5561	5564
	5575	5588	5591	5599	5635	5694	5717	5771	5866	5894	5933	5958	5967	5982	5997
	6031	6046	6098	6105	6117	6156	6162	6184	6190	6229	6235	6238	6251	6254	6364
	6401	6406	6424	6436	6447	6451	6466	6497	6510	6517	6551	6560	6568	6574	6606
	6621	6631	6640	6647	6684	6699	6720	6764	6781	6846	6865				
BVC	1775														
CCC	2481														
CLC	1852	6402	6547	6557	6563										
CLN	1813	1829	2023	2187	2608	2693	2730								
CLR	1491	1492	1494	1497	1510	1511	1517	1524	1548	1554	1558	1586	1788	1827	1894
	1917	1938	2020	2052	2053	2054	2066	2075	2096	2098	2104	2146	2147	2179	2241
	2242	2318	2320	2349	2353	2367	2404	2414	2418	2448	2468	2479	2554	2557	2603
	2606	2644	2663	2665	2689	2690	2725	2726	2803	2804	2820	2853	2857	2859	2864
	2866	2872	2874	2881	2883	2886	2890	2892	2895	2897	2916	2919	2924	2928	2931
	2949	2952	2956	2960	2963	2985	2989	2994	2998	3016	3020	3025	3029	3033	3056
	3059	3063	3066	3071	3075	3093	3097	3103	3106	3109	3172	3174	3184	3244	3254
	3258	3268	3272	3274	3281	3285	3288	3295	3299	3302	3334	3339	3346	3351	3368
	3514	3523	3562	3667	3674	3684	3686	3692	3701	3711	3721	3742	3759	3755	3767
	3780	3824	3827	3885	3897	3918	3937	3947	4000	4048	4211	4212	4213	4249	4250
	4254	4258	4266	4271	4282	4287	4298	4315	4316	4330	4331	4337	4338	4357	4372
	4373	4387	4388	4393	4394	4413	4418	4443	4444	4449	4450	4461	4468	4481	4496
	4511	4529	4530	4535	4544	4563	4564	4578	4579	4592	4598	4607	4613	4615	4632
	4633	4635	4641	4671	4675	4687	4730	4741	4756	4832	4849	4873	4883	4904	4945
	4969	4975	4985	4988	5013	5113	5132	5176	5186	5223	5249	5264	5280	5281	5285
	5286	5317	5365	5368	5371	5374	5377	5380	5406	5408	5418	5422	5445	5447	5451
	5456	5459	5483	5485	5489	5497	5500	5501	5506	5512	5514	5518	5519	5532	5549
	5580	5581	5613	5628	5640	5664	5669	5726	5731	5736	5741	5747	5781	5786	5792
	5831	5832	5838	5839	5849	5850	5856	5857	5864	5876	5905	5907	5915	5968	5989
	5992	5994	5999	6001	6006	6008	6010	6063	6069	6087	6088	6112	6115	6178	6253
	6267	6333	6390	6391	6392	6393	6490	6514	6527	6697	6756	6759	6850	6850	6855
CLRB	4664	4829	4877	4890	4892	4911	4916	4930	4986	4989	5014	5047	5053	5056	5065
	5068	5070	5124	5137	5151	5164	5214	5546	5598	5614	5620	5627	6252	6646	6785
CLV	1852	2451	2576												
CMP	1498	1529	1592	1600	1659	1661	1705	1713	1720	1727	1764	1768	1792	1798	1819
	1834	1836	1856	1862	1971	1975	2004	2006	2008	2029	2042	2155	2159	2170	2177
	2195	2199	2204	2246	2326	2328	2342	2347	2354	2358	2368	2372	2376	2380	2386
	2390	2408	2422	2424	2433	2440	2455	2460	2472	2485	2490	2502	2505	2524	2528
	2543	2547	2562	2566	2580	2617	2621	2649	2697	2704	2734	2738	2743	2769	2805
	2811	3144	3179	3185	3474	3517	3546	3553	3565	3659	3665	3697	3707	3715	3747
	3753	3768	3771	3774	3837	3852	3862	3880	4055	4623	4681	4695	4704	4707	4711
	4717	4763	4766	4770	4772	4775	4777	4780	4782	4785	4787	4790	4792	4863	4908
	4913	4948	4951	4966	4983	5045	5051	5060	5063	5086	5183	5436	5452	5474	5556
	5570	5583	5605	5671	5878	5908	6236	6260	6311	6328	6330	6399	6404	6433	6448
	6493	6530	6540	6777											
CMPB	2813	4816	4823	5121	5134	5148	5160	6242	6246	6335	6395	6419	6615	6617	6623

NO3

COM	6644	6648													
DEC	4801	4802	4803	4804	4924	4925	4926	6134							
DECB	3887	3893	5615	5621	6091	6339	6505	6533							
DIV	6627	6630	6705	6716											
EMT	2323	2340	2407	2420	2453	2471	2483	2501	2521	2540	2559	2577			
HALT	49	3951	3970												
INC	441	4528	5212	6296	6310	6313	6513	6605	6845	6864					
INCB	1527	1539	1594	2816	3171	3270	3283	3297	3336	3349	3579	3749	3877	3882	3924
IOT	3929	4227	4237	4426	4636	4642	4732	4800	4928	4950	5336	5602	5607	5813	5873
JMP	5901	5936	6049	6259	6300	6711	6719	6763	6851						
MOV	6264	6291	6650												
MFFC	50	5159	5172	5594	5611	5617	5625								
MFFI	445	3207	3822	3861	4305	5314	5582	5637	5698	5814	5937	6142	6520		
MFFO	3363	3399	3432	3483	4245	4311	4326	4351	4352	4356	4362	4368	4383	4408	4412
MFFS	4419	4439	4476	4480	4491	4495	4506	4510	4516	4524	4559	4574	6121	6125	6306
MFFT	6432	6444	6622	6629	6636										
MFFV	1704	4710													
MFFX	5018														
MFFY	2695	2732													
MFFZ	1490	1493	1495	1496	1500	1501	1502	1503	1504	1505	1506	1507	1508	1509	1513
MFFA	1514	1515	1516	1518	1520	1523	1525	1526	1541	1542	1543	1544	1546	1547	1550
MFFB	1551	1552	1553	1555	1583	1584	1585	1587	1588	1590	1595	1598	1609	1627	1628
MFFC	1629	1630	1631	1632	1633	1634	1635	1638	1641	1643	1646	1647	1654	1655	1658
MFFD	1664	1688	1689	1690	1693	1694	1696	1697	1698	1699	1700	1701	1702	1708	1711
MFFE	1715	1716	1717	1719	1722	1723	1724	1726	1729	1731	1747	1748	1749	1752	1753
MFFG	1757	1758	1759	1760	1779	1781	1785	1786	1787	1801	1802	1810	1811	1817	1821
MFFH	1825	1826	1832	1838	1843	1847	1848	1850	1855	1858	1859	1864	1879	1880	1881
MFFI	1882	1883	1884	1885	1893	1897	1912	1913	1914	1915	1916	1919	1921	1933	1934
MFFJ	1935	1936	1937	1941	1943	1960	1963	1964	1965	1969	1977	1978	1981	1984	1985
MFFK	1986	1987	1991	1992	1993	1994	1998	2000	2001	2002	2003	2012	2018	2019	2021
MFFL	2026	2027	2034	2035	2036	2037	2038	2039	2044	2045	2049	2050	2051	2055	2058
MFFM	2061	2062	2064	2065	2067	2070	2073	2074	2076	2090	2091	2092	2093	2094	2095
MFFN	2097	2099	2103	2105	2106	2110	2111	2113	2141	2144	2145	2148	2152	2153	2154
MFFO	2183	2184	2190	2191	2192	2193	2194	2206	2210	2211	2217	2234	2235	2236	2237
MFFP	2238	2239	2240	2243	2248	2249	2251	2316	2319	2324	2332	2336	2337	2341	2352
MFFQ	2362	2363	2366	2382	2383	2392	2393	2403	2405	2413	2415	2416	2417	2421	2428
MFFR	2429	2435	2436	2442	2446	2447	2449	2454	2462	2466	2467	2469	2477	2478	2480
MFFS	2484	2452	2496	2497	2498	2499	2507	2508	2509	2510	2514	2515	2516	2517	2518
MFFT	2519	2522	2523	2533	2534	2535	2536	2537	2538	2541	2542	2552	2553	2555	2556
MFFU	2560	2561	2571	2572	2573	2574	2578	2582	2600	2601	2602	2604	2605	2611	2612
MFFV	2613	2615	2623	2625	2640	2641	2642	2643	2653	2654	2656	2657	2666	2668	2679
MFFW	2680	2681	2682	2683	2684	2685	2687	2688	2691	2696	2706	2708	2721	2722	2723
MFFX	2724	2727	2728	2733	2740	2742	2759	2760	2761	2762	2763	2764	2765	2774	2775
MFFY	2792	2793	2794	2795	2796	2799	2800	2807	2808	2809	2822	2838	2839	2840	2841
MFFZ	2842	2843	2844	2847	2849	2850	2855	2856	2862	2870	2879	2887	2898	2899	2894
MFAA	2910	2911	2912	2913	2920	2923	2925	2930	2943	2944	2945	2946	2953	2955	2957
MFAB	2962	2976	2977	2978	2979	2982	2986	2988	2990	2991	2997	3010	3011	3012	3013
MFAC	3017	3021	3023	3024	3026	3028	3032	3047	3048	3049	3050	3053	3054	3055	3062
MFAD	3065	3067	3068	3070	3074	3087	3088	3089	3090	3094	3098	3100	3101	3102	3105
MFAE	3108	3125	3126	3129	3131	3132	3133	3136	3150	3151	3152	3155	3160	3161	3162
MFAF	3165	3167	3168	3177	3181	3182	3201	3202	3208	3209	3212	3215	3218	3220	3221
MFAH	3223	3224	3226	3227	3229	3230	3231	3238	3240	3241	3242	3243	3249	3251	3252
MFAI	3253	3255	3257	3266	3267	3269	3275	3279	3280	3282	3292	3293	3294	3296	3303
MFAJ	3314	3315	3316	3326	3327	3330	3332	3343	3344	3345	3355	3359	3361	3377	3378
MFAK	3390	3391	3395	3397	3410	3411	3423	3424	3428	3430	3457	3458	3459	3462	3463
MFAL	3465	3466	3467	3468	3469	3470	3471	3476	3480	3481	3482	3484	3485	3489	3490

3491	3492	3494	3495	3500	3501	3502	3503	3505	3506	3512	3513	3515	3516	3519
3520	3522	3537	3540	3541	3544	3545	3551	3552	3559	3560	3561	3570	3571	3576
3577	3578	3581	3582	3589	3590	3591	3592	3593	3600	3601	3604	3606	3607	3610
3636	3637	3638	3641	3642	3644	3645	3646	3647	3648	3649	3650	3651	3652	3655
3656	3661	3662	3671	3672	3673	3675	3676	3680	3681	3682	3683	3685	3687	3691
3693	3694	3695	3696	3699	3700	3703	3704	3705	3706	3709	3710	3713	3714	3717
3718	3720	3739	3740	3741	3743	3744	3745	3750	3751	3752	3757	3758	3760	3764
3777	3778	3816	3817	3818	3819	3823	3825	3826	3828	3829	3830	3831	3832	3833
3834	3835	3848	3849	3879	3898	3899	3902	3917	3920	3921	3922	3923	3926	3928
3932	3938	3939	3940	3941	3948	3949	3950	3952	3953	3956	3957	3966	3967	3958
3969	3972	3973	3974	3983	3984	3985	3986	3987	3988	3990	3991	3992	3996	4013
4014	4017	4018	4019	4022	4023	4024	4025	4026	4027	4029	4030	4031	4047	4049
4050	4051	4052	4054	4065	4066	4067	4070	4071	4072	4075	4075	4077	4078	4079
4084	4085	4090	4091	4096	4097	4105	4106	4107	4108	4113	4114	4115	4116	4124
4125	4133	4134	4139	4140	4148	4149	4157	4158	4159	4164	4165	4170	4171	4208
4209	4210	4216	4217	4218	4220	4221	4222	4239	4240	4241	4242	4255	4256	4257
4259	4262	4264	4265	4272	4273	4274	4275	4278	4280	4281	4288	4289	4290	4291
4294	4296	4297	4302	4306	4307	4308	4309	4310	4321	4322	4323	4324	4325	4341
4342	4343	4344	4347	4349	4350	4363	4364	4365	4366	4367	4378	4379	4380	4381
4382	4397	4398	4399	4400	4403	4405	4406	4407	4428	4429	4430	4431	4434	4436
4427	4438	4451	4452	4453	4454	4457	4459	4460	4466	4467	4471	4472	4473	4474
4475	4486	4487	4488	4489	4490	4491	4501	4502	4504	4505	4519	4520	4521	4522
4523	4536	4537	4538	4539	4542	4545	4546	4551	4554	4555	4556	4557	4558	4569
4570	4571	4572	4573	4584	4585	4586	4587	4590	4593	4599	4600	4601	4602	4605
4608	4612	4614	4621	4625	4627	4628	4630	4655	4656	4672	4673	4686	4689	4700
4701	4703	4714	4720	4727	4728	4739	4742	4754	4755	4758	4761	4769	4774	4779
4784	4789	4814	4818	4819	4825	4826	4833	4844	4846	4847	4853	4857	4858	4860
4861	4865	4869	4871	4881	4885	4886	4902	4903	4906	4940	4943	4944	4953	4962
4968	4971	4977	4978	4979	5000	5001	5004	5005	5006	5009	5010	5015	5016	5033
5034	5036	5037	5038	5040	5041	5042	5084	5088	5110	5111	5112	5116	5117	5118
5119	5128	5130	5131	5142	5143	5144	5145	5146	5153	5157	5158	5163	5168	5170
5171	5180	5182	5185	5189	5192	5193	5194	5208	5209	5210	5219	5220	5227	5238
5239	5240	5241	5242	5243	5245	5246	5253	5254	5255	5256	5257	5258	5260	5265
5266	5270	5271	5272	5273	5274	5275	5276	5315	5316	5318	5319	5320	5321	5322
5323	5324	5325	5326	5327	5328	5329	5330	5334	5335	5350	5351	5352	5353	5354
5355	5356	5357	5358	5359	5360	5361	5362	5363	5381	5398	5399	5400	5401	5402
5403	5404	5405	5412	5413	5414	5415	5416	5435	5438	5439	5440	5441	5442	5443
5444	5454	5455	5473	5476	5477	5478	5480	5481	5482	5491	5492	5493	5494	5495
5496	5502	5507	5508	5509	5510	5511	5531	5533	5534	5535	5536	5537	5538	5539
5540	5541	5542	5544	5545	5547	5548	5550	5551	5552	5553	5554	5579	5597	5604
5612	5626	5629	5630	5633	5641	5642	5643	5659	5660	5661	5663	5665	5666	5670
5706	5707	5708	5709	5710	5711	5712	5713	5714	5715	5716	5723	5733	5738	5743
5749	5750	5751	5763	5764	5765	5766	5767	5768	5769	5770	5777	5788	5794	5795
5796	5808	5815	5817	5818	5821	5822	5823	5824	5825	5826	5827	5829	5840	5841
5842	5843	5844	5845	5847	5858	5859	5860	5861	5862	5863	5865	5871	5880	5885
5886	5887	5888	5889	5890	5892	5893	5899	5912	5913	5914	5949	5950	5951	5952
5953	5954	5955	5956	5961	5970	5973	5974	5975	5976	5977	5978	5979	5980	5987
6002	6009	6025	6034	6038	6039	6047	6048	6058	6059	6064	6068	6094	6101	6108
6113	6116	6120	6122	6130	6137	6153	6154	6165	6167	6171	6175	6179	6181	6182
6193	6195	6199	6203	6206	6207	6231	6232	6234	6237	6250	6262	6263	6265	6266
6269	6270	6293	6301	6316	6319	6337	6338	6340	6342	6344	6349	6354	6359	6374
6376	6381	6383	6394	6410	6412	6425	6428	6431	6437	6440	6443	6456	6459	6469
6491	6500	6501	6502	6503	6525	6534	6537	6538	6539	6544	6545	6554	6555	6566
6571	6577	6607	6608	6612	6625	6680	6688	6689	6690	6696	6703	6721	6722	6723
6724	6725	6746	6747	6748	6749	6750	6751	6752	6757	6760	6780	6786	6787	6789
6789	6790	6792	6793	6809	6810	6813	6835	6836	6837	6838	6839	6840	6841	6842

MOV8	6843	6844	6849	6853	6854	6855	6856	6857	6858	6859	6860	5156	5162	5169	5175
	1512	3580	3878	3927	3942	3989	4815	4822	5123	5136	5150	6609	6635	6643	6681
	5603	5816	5891	6166	6194	6268	6290	6303	6460	6528	6565	6609	6635	6643	6681
	6682	6685	6686	6687	6691	6694	6695	6714	6755	6758	6772	6775	6784	6812	
MTPO	4947	4981													
MTPI	2610	2646	2659	3563											
MUL	2151	2189	2214	2219	2245	6341									
NEG	6692	6754													
NOP	1596	1644	1703	1761	1789	1886	1895	1939	2040	2056	2068	2100	2213	2244	2406
	2419	2470	2500	2520	2539	2558	2645	2658	3554	3896	3891	4247	4313	4328	4354
	4370	4385	4410	4441	4478	4493	4508	4526	4561	4576	5722	5727	5732	5737	5742
	5748	5776	5778	5782	5787	5793	5872	5877	5900	5906	5988	5993	6000	6007	6126
	6127	6128													
RESET	1645	5181	6043	6052	6124	6495									
ROL	2817	6226	6461	6529	6698	6700	6701	6702	6704						
ROR	3335	3348	4619	6532	6548	6558	6564								
RTI	1515	4028	5043	5120	5133	5147	5195	6138	6271	6321	6614	6726	6794	6863	
RTS	3276	3304	4645	6357	6454	6470	6507	6580	6652	6814					
RTT	1519	1520	1695	1754	2686	2801	3464	3542	3643	4020	4032	4053	4073	5011	5261
SCC	2022	2149	2186	2338	2450	2575	2607	2692	2729						
SEC	1828														
SEN	1851	1995	2321												
SEV	1814	1828	1966												
SEZ	1814	1851	1995	2482											
SOB	1545	2819	5331	5960	6460	6504									
SPL	1589	1597	1637	2766	2802	2851	2914	2926	2947	2958	2980	2992	2995	3014	3027
	3030	3051	3069	3072	3091	3104	3159	3170	3333	3347	3362	3398	3431	3608	3746
	4243	4246	4261	4277	4293	4312	4327	4353	4369	4384	4409	4440	4456	4477	4492
	4507	4525	4541	4560	4575	4589	4604	4634	4644	5080	5082	5244	5277	5667	5828
	5846	6066													
SUB	5969	6302	6526	6546	6556	6559	6761								
TRAP	5668	6816	6826	6827	6828	6829									
TST	1537	1556	1602	1606	1610	1794	1805	1889	1898	1973	2108	2157	2163	2166	2344
	2647	2661	2700	3203	3210	3213	3216	3219	3239	3250	3321	3353	3383	3388	3416
	3421	3524	3584	3595	3722	3755	3762	3875	3900	3975	4229	4303	4339	4395	4424
	4469	4517	4552	4677	4691	4734	4744	4798	4851	4875	4888	4918	4922	4954	4960
	5019	5224	5305	5332	5460	5600	5631	5689	5724	5779	5874	5928	5957	5962	6026
	6032	6233	6257	6294	6308	6317	6361	6408	6611	6619	6639	6709	6766	6776	6811
TSTB	1639	1652	4830	5173	6244	6603	6641	6768	6782						
WAIT	5247	5262	5278	6067											
.ASCII	517	518	6473	6476	6478	6479	6480	6878	6955	6972	7008	7039	7088	7134	7149
	7165	7208	7269	7599	7625	7647	7673	7698	7713	7721	7729	7878	7907	7925	7958
	7967	7976	7999	8012	8026	8034	8045	8057	8070	8084	8108	8130	8142	8158	8176
	8194	8246	8292	8300	8318	8326	8375	8382	8391	8398	8412	8437	8450	8483	8492
	8501	8510	8542	8561	8601	8641	8665	8680	8715	8746	8790	8808	8823	8988	8999
	9054	9069	9084	9097	9113	9136	9156	9183	9197	9213	9231	9284	9304	9316	9323
	9330	9342	9376	9403	9421	9514	9590	9595							
.ASCIZ	516	519	1534	6100	6107	6158	6164	6186	6192	6453	6481	6499	6512	6519	6570
	6576	6867	6870	6886	6896	6899	6903	6909	6915	6924	6929	6931	6933	6939	6945
	6950	6960	6967	6978	6981	6986	6995	7005	7014	7019	7029	7034	7044	7050	7056
	7062	7067	7073	7080	7083	7099	7104	7114	7121	7128	7139	7145	7154	7160	7172
	7182	7189	7193	7197	7202	7216	7229	7237	7242	7245	7253	7261	7278	7282	7291
	7297	7303	7309	7314	7319	7324	7327	7333	7336	7342	7353	7357	7361	7366	7376
	7382	7389	7393	7404	7409	7420	7431	7436	7446	7452	7456	7460	7468	7477	7488
	7495	7511	7505	7510	7519	7526	7532	7539	7541	7546	7552	7558	7562	7567	7571
	7577	7583	7588	7590	7594	7608	7615	7620	7634	7638	7643	7652	7658	7663	7681

	7686	7692	7706	7738	7745	7755	7758	7764	7774	7781	7789	7799	7806	7816	7823
	7831	7841	7849	7856	7866	7873	7885	7889	7894	7904	7916	7934	7938	7948	7985
	7990	8009	8022	8037	8052	8065	8078	8091	8101	8114	8120	8138	8150	8166	8172
	8184	8205	8213	8220	8227	8238	8252	8255	8261	8269	8277	8285	8308	8314	8334
	8339	8343	8347	8351	8355	8359	8370	8405	8421	8427	8446	8459	8467	8473	8480
	8518	8526	8531	8551	8553	8571	8579	8590	8610	8618	8626	8634	8650	8658	8675
	8691	8697	8707	8712	8725	8730	8737	8755	8761	8771	8780	8799	8818	8832	8837
	8847	8852	8857	8862	8869	8876	8884	8891	8898	8905	8910	8915	8920	8929	8938
	8946	8954	8962	8970	8975	8983	8994	9004	9007	9015	9022	9027	9034	9043	9049
	9063	9077	9093	9106	9122	9126	9146	9165	9170	9177	9192	9207	9224	9238	9244
	9251	9259	9267	9276	9293	9296	9313	9338	9351	9354	9364	9367	9383	9389	9396
	9411	9414	9430	9436	9442	9449	9457	9464	9472	9475	9478	9481	9484	9487	9490
	9493	9496	9503	9523	9529	9537	9542	9548	9554	9559	9563	9568	9571	9574	9578
	9582	9586	9601	9607	9613	9619	9626								
.BLKW	6799														
.BYTE	483	484	489	490	501	502	503	504	6144	6471	6727	6728	6729	6730	
.ENABL	1	4													
.END	9694														
.ENDC	11	29	31	32	33	41	430	434	446	447	469	471	473	480	505
	510	514	515	516	517	546	547	1500	1501	1503	1505	1507	1509	1510	1511
	1513	1525	1527	1529	1531	1534	1536	1558	1561	1562	1581	1582	1583	1608	1615
	1616	1625	1626	1627	1664	1673	1674	1685	1686	1687	1722	1734	1735	1745	1746
	1747	1784	1809	1824	1846	1864	1867	1878	1877	1878	1879	1889	1903	1904	1910
	1911	1912	1921	1924	1925	1931	1932	1933	1943	1946	1947	1958	1959	1960	1990
	2017	2033	2048	2073	2079	2080	2088	2089	2090	2110	2116	2117	2139	2140	2141
	2182	2209	2221	2223	2224	2232	2233	2234	2248	2255	2256	2314	2315	2316	2335
	2402	2412	2445	2465	2476	2495	2513	2532	2551	2570	2582	2585	2586	2598	2599
	2600	2623	2628	2629	2638	2639	2640	2668	2671	2672	2677	2678	2679	2706	2711
	2712	2719	2720	2721	2740	2748	2749	2757	2758	2759	2778	2780	2782	2783	2790
	2791	2792	2822	2825	2826	2836	2837	2838	2890	2901	2902	2908	2909	2910	2930
	2934	2935	2941	2942	2943	2962	2967	2968	2974	2975	2976	2997	3001	3002	3008
	3009	3010	3032	3036	3037	3045	3046	3047	3074	3078	3079	3085	3086	3087	3108
	3113	3114	3123	3124	3125	3187	3189	3200	3246	3257	3264	3307	3308	3312	3313
	3314	3326	3371	3372	3375	3376	3377	3388	3390	3401	3404	3405	3408	3409	3410
	3421	3423	3434	3437	3438	3455	3456	3457	3526	3528	3529	3535	3536	3537	3619
	3620	3634	3635	3636	3724	3727	3728	3737	3738	3739	3783	3784	3806	3807	3808
	3809	3816	3902	3905	3906	3915	3916	3917	4003	4004	4011	4012	4013	4016	4022
	4040	4041	4045	4046	4047	4057	4059	4060	4063	4064	4065	4104	4123	4132	4147
	4156	4175	4176	4206	4207	4208	4232	4253	4270	4286	4301	4320	4336	4361	4377
	4392	4417	4448	4465	4485	4500	4515	4534	4550	4568	4583	4597	4617	4647	4648
	4653	4654	4655	4670	4685	4699	4707	4710	4726	4738	4746	4749	4750	4752	4753
	4754	4800	4808	4809	4812	4813	4814	4832	4838	4839	4842	4843	4844	4896	4897
	4900	4901	4902	4934	4935	4938	4939	4940	4992	4993	4998	4999	5000	5021	5024
	5025	5031	5032	5033	5075	5076	5078	5079	5080	5088	5091	5092	5108	5109	5110
	5115	5127	5140	5154	5167	5179	5192	5197	5198	5206	5207	5208	5230	5231	5236
	5237	5238	5290	5295	5334	5339	5340	5348	5349	5350	5384	5385	5396	5397	5398
	5425	5426	5433	5434	5435	5438	5462	5465	5466	5471	5472	5473	5522	5523	5529
	5530	5531	5646	5647	5657	5658	5659	5673	5677	5684	5687	5689	5700	5701	5704
	5705	5706	5754	5755	5761	5762	5763	5799	5800	5806	5807	5808	5820	5837	5855
	5884	5919	5923	5926	5928	5939	5940	5947	5948	5949	5964	5966	5968	6012	6013
	6023	6024	6025	6074	6077	6079	6081	6082	6084	6087	6093	6096	6097	6100	6107
	6113	6120	6122	6124	6142	6143	6144	6145	6147	6153	6158	6164	6186	6192	6211
	6217	6222	6226	6228	6239	6242	6243	6244	6246	6248	6255	6259	6264	6269	6272
	6273	6274	6280	6291	6297	6301	6307	6308	6314	6321	6322	6323	6328	6366	6370
	6453	6484	6499	6512	6519	6522	6570	6576	6582	6655	6733	6801	6810	6813	6815
	6825	6826	6827	6828	6829	6830	6831	6843	6853	6863	6870				

E04

PDP 11/70 CPU DIAGNOSTIC PART 2 MACY11 27(732) 21-DEC-76 16:00 PAGE 218
 DEKBBC.P11 CROSS REFERENCE TABLE -- PERMANENT SYMBOLS

SEQ 0250

	49	50	52	73	74	75	76	77	78	79	80	81	82	83	112
.EQUIV	113	114	115	116	117	118	119	120	121	140	141	142	143	144	145
	146	147	148	149	201	202	203	204	412	413	414	415	416	417	418
	419	420	421	422	423	424	425	426	427						
.EVEN	1534	6100	6107	6145	6158	6164	6186	6192	6453	6482	6499	6512	6519	6570	6576
	6869	6892	6913	6964	7047	7142	7157	7178	7225	7655	7786	8117	8127	8190	8363
	8881	9629													
.IF	7	28	29	30	31	32	33	41	430	434	441	446	468	470	472
	479	505	510	514	515	516	520	546	1495	1500	1501	1503	1505	1507	1509
	1510	1511	1513	1525	1528	1529	1530	1533	1535	1557	1560	1562	1581	1583	1607
	1614	1616	1625	1627	1663	1672	1674	1685	1687	1721	1733	1735	1745	1747	1783
	1808	1823	1845	1863	1866	1868	1877	1879	1888	1902	1904	1910	1912	1920	1923
	1925	1931	1933	1942	1945	1947	1958	1960	1989	2016	2032	2047	2072	2078	2080
	2088	2090	2109	2115	2117	2139	2141	2181	2208	2220	2222	2224	2232	2234	2247
	2254	2256	2314	2316	2334	2401	2411	2444	2464	2475	2494	2512	2531	2550	2569
	2581	2584	2586	2598	2600	2622	2627	2629	2638	2640	2667	2670	2672	2677	2679
	2705	2710	2712	2719	2721	2739	2747	2749	2757	2759	2777	2779	2781	2783	2790
	2792	2821	2824	2826	2836	2838	2889	2900	2902	2908	2910	2929	2933	2935	2941
	2943	2961	2966	2968	2974	2976	2996	3000	3002	3008	3010	3031	3035	3037	3045
	3047	3073	3077	3079	3085	3087	3107	3112	3114	3123	3125	3126	3198	3199	3245
	3256	3263	3306	3308	3312	3314	3325	3370	3372	3375	3377	3387	3389	3400	3403
	3405	3408	3410	3420	3422	3433	3436	3438	3455	3457	3525	3527	3529	3535	3537
	3618	3620	3634	3636	3723	3726	3728	3737	3739	3782	3784	3806	3808	3815	3901
	3904	3906	3915	3917	4002	4004	4011	4013	4015	4021	4039	4041	4045	4047	4056
	4058	4060	4063	4065	4103	4122	4131	4146	4155	4174	4176	4206	4208	4231	4252
	4269	4285	4300	4319	4335	4360	4376	4391	4416	4447	4464	4494	4499	4514	4533
	4549	4567	4582	4596	4616	4646	4648	4653	4655	4669	4684	4698	4706	4709	4725
	4737	4745	4748	4750	4752	4754	4799	4807	4809	4812	4814	4831	4837	4839	4843
	4844	4895	4897	4900	4902	4933	4935	4938	4940	4991	4993	4998	5000	5020	5023
	5025	5031	5033	5074	5076	5078	5090	5087	5090	5092	5108	5110	5114	5126	5139
	5153	5166	5178	5191	5196	5198	5206	5208	5229	5231	5236	5238	5289	5294	5333
	5338	5340	5348	5350	5383	5385	5396	5398	5424	5426	5433	5435	5437	5461	5464
	5466	5471	5473	5521	5523	5529	5531	5645	5647	5657	5659	5672	5676	5683	5686
	5688	5699	5701	5704	5706	5753	5755	5761	5763	5798	5800	5806	5808	5819	5836
	5854	5883	5918	5922	5925	5927	5938	5940	5947	5949	5963	5965	5967	6011	6013
	6023	6025	6073	6077	6078	6079	6081	6083	6084	6086	6092	6095	6097	6099	6106
	6115	6120	6122	6130	6142	6143	6144	6146	6152	6157	6163	6185	6191	6210	6216
	6221	6226	6238	6240	6241	6242	6244	6245	6246	6255	6257	6266	6271	6272	6273
	6279	6290	6294	6297	6304	6306	6307	6308	6314	6321	6322	6327	6365	6369	6452
	6483	6498	6511	6518	6521	6569	6575	6581	6654	6732	6800	6809	6813	6815	6816
	6826	6827	6828	6829	6830	6843	6853	6861	6863	6867					
.IFF	29	31	32	33	447	469	471	473	479	505	547	1500	1528	1530	1536
	1558	1561	1562	1582	1583	1608	1615	1616	1626	1627	1664	1673	1674	1686	1687
	1722	1734	1735	1746	1747	1784	1809	1824	1846	1864	1867	1868	1878	1879	1881
	1903	1904	1911	1912	1921	1924	1925	1932	1933	1943	1946	1947	1959	1960	1990
	2017	2033	2048	2073	2079	2080	2089	2090	2110	2116	2117	2140	2141	2192	2209
	2221	2223	2224	2233	2234	2248	2255	2256	2315	2316	2335	2402	2412	2445	2465
	2476	2495	2513	2532	2551	2570	2582	2585	2586	2599	2600	2623	2628	2629	2639
	2640	2668	2671	2672	2678	2679	2706	2711	2712	2720	2721	2740	2748	2749	2758
	2759	2778	2780	2782	2783	2791	2792	2822	2825	2826	2837	2838	2890	2901	2902
	2909	2910	2930	2934	2935	2942	2943	2962	2967	2968	2975	2976	2997	3001	3002
	3009	3010	3032	3036	3037	3046	3047	3074	3078	3079	3086	3087	3108	3113	3114
	3124	3125	3186	3189	3200	3246	3257	3264	3307	3308	3313	3314	3326	3371	3372
	3376	3377	3388	3390	3401	3404	3405	3409	3410	3421	3423	3434	3437	3438	3456
	3457	3526	3528	3529	3536	3537	3619	3620	3635	3636	3724	3727	3728	3738	3739
	3783	3784	3807	3808	3809	3816	3902	3905	3906	3916	3917	4003	4004	4012	4013

	4016	4022	4040	4041	4046	4047	4057	4059	4060	4064	4065	4104	4123	4132	4147
	4156	4175	4176	4207	4208	4232	4253	4270	4286	4301	4320	4336	4361	4377	4392
	4417	4448	4465	4485	4500	4515	4534	4550	4568	4583	4597	4617	4647	4648	4654
	4655	4670	4685	4699	4707	4710	4726	4738	4746	4749	4750	4753	4754	4800	4808
	4809	4813	4814	4832	4838	4839	4843	4844	4896	4897	4901	4902	4934	4935	4939
	4940	4992	4993	4999	5000	5021	5024	5025	5032	5033	5075	5076	5079	5080	5088
	5091	5092	5109	5110	5115	5127	5140	5154	5167	5179	5192	5197	5198	5207	5208
	5230	5231	5237	5238	5290	5295	5334	5339	5340	5349	5350	5384	5385	5397	5398
	5425	5426	5434	5435	5438	5462	5465	5466	5472	5473	5522	5523	5530	5531	5646
	5647	5658	5659	5672	5677	5684	5687	5689	5700	5701	5705	5706	5754	5755	5762
	5763	5799	5800	5807	5808	5820	5837	5855	5884	5919	5923	5926	5928	5939	5940
	5948	5949	5964	5966	5968	6012	6013	6024	6025	6074	6083	6087	6092	6095	6143
	6147	6153	6211	6239	6242	6243	6246	6272	6273	6274	6279	6297	6308	6321	6322
	6323	6328	6366	6370	6484	6522	6582	6655	6733	6801	6810	6831	6863		
.IFT	1534	6100	6107	6158	6164	6186	6192	6254	6307	6453	6499	6512	6519	6570	6576
.IFTF	1534	6100	6107	6158	6164	6186	6192	6252	6306	6453	6499	6512	6519	6570	6576
.IIF	6	11	16	17	25	26	27	29	32	33	34	35	36	37	38
	39	40	441	520	1501	1503	1509	1510	1511	1525	1526	1529	6040	6049	6060
	6079	6087	6088	6102	6109	6143	6145	6168	6172	6176	6196	6200	6204	6217	6218
	6219	6220	6221	6222	6253	6254	6269	6272	6273	6280	6281	6282	6283	6284	6285
	6308	6322	6360	6377	6384	6429	6441	6483	6572	6578	6654	6825	6826	6827	6828
	6829														
.IRP	434	520	1560	1614	1672	1733	1866	1902	1923	1945	2078	2115	2222	2254	2584
	2627	2670	2710	2747	2781	2824	2900	2933	2966	3000	3035	3077	3112	3306	3370
	3403	3436	3527	3618	3726	3782	3904	4002	4039	4058	4174	4646	4748	4807	4837
	4895	4933	4991	5023	5074	5090	5196	5229	5338	5383	5424	5464	5521	5645	5699
	5753	5798	5938	6011	6120	6290	6746	6786	6837	6853					
.LIST	1	4	32	431	434	441	505	507	508	509	510	511	512	513	514
	1529	1534	1560	1583	1614	1627	1672	1687	1733	1747	1866	1879	1902	1912	1923
	1933	1945	1960	2078	2090	2115	2141	2222	2234	2254	2316	2584	2600	2627	2640
	2670	2679	2710	2721	2747	2759	2781	2792	2824	2838	2900	2910	2933	2943	2966
	2976	3000	3010	3035	3047	3077	3087	3112	3125	3306	3314	3370	3377	3403	3410
	3436	3457	3527	3537	3618	3636	3726	3739	3782	3808	3904	3917	4002	4013	4039
	4047	4058	4065	4174	4208	4646	4655	4748	4754	4807	4814	4837	4844	4895	4902
	4933	4940	4991	5000	5023	5033	5074	5080	5090	5110	5196	5208	5229	5238	5338
	5350	5383	5398	5424	5435	5464	5473	5521	5531	5645	5659	5699	5706	5753	5763
	5798	5808	5938	5949	6011	6025	6087	6100	6107	6115	6122	6158	6164	6186	6192
	6221	6308	6453	6499	6512	6519	6570	6576	6815	6816	6825	6826	6827	6828	6829
	6830														
.MACRO	1	33	472	1513	1560	1614	1672	1733	1866	1902	1923	1945	2078	2115	2222
	2253	2584	2627	2670	2710	2747	2781	2824	2900	2933	2966	3000	3035	3077	3112
	3306	3370	3403	3436	3527	3618	3726	3782	3904	4002	4039	4058	4174	4646	4748
	4807	4837	4895	4933	4991	5023	5074	5090	5196	5229	5338	5383	5424	5464	5521
	5645	5699	5753	5798	5938	6011	6073	6816							
.MCALL	4	431	6072	6209											
.MLIST	1	4	32	431	434	441	505	507	508	509	510	511	512	513	514
	1529	1534	1560	1583	1614	1627	1672	1687	1733	1747	1866	1879	1902	1912	1923
	1933	1945	1960	2078	2090	2115	2141	2222	2234	2254	2316	2584	2600	2627	2640
	2670	2679	2710	2721	2747	2759	2781	2792	2824	2838	2900	2910	2933	2943	2966
	2976	3000	3010	3035	3047	3077	3087	3112	3125	3306	3314	3370	3377	3403	3410
	3436	3457	3527	3537	3618	3636	3726	3739	3782	3808	3904	3917	4002	4013	4039
	4047	4058	4065	4174	4208	4646	4655	4748	4754	4807	4814	4837	4844	4895	4902
	4933	4940	4991	5000	5023	5033	5074	5080	5090	5110	5196	5208	5229	5238	5338
	5350	5383	5398	5424	5435	5464	5473	5521	5531	5645	5659	5699	5706	5753	5763
	5798	5808	5938	5949	6011	6025	6087	6100	6107	6115	6122	6158	6164	6186	6192
	6221	6308	6453	6499	6512	6519	6570	6576	6815	6816	6825	6826	6827	6828	6829

.PAGE	6830														
.REPT	18	41	472	546											
.SBTTL	411	507	510												
	21	42	167	178	192	341	435	442	448	474	548	1560	1614	1672	1733
	1866	1902	1923	1945	2078	2115	2222	2254	2584	2627	2670	2710	2747	2781	2824
	2900	2933	2966	3000	3035	3077	3112	3189	3306	3370	3403	3436	3527	3618	3726
	3782	3904	4002	4039	4058	4174	4646	4748	4807	4837	4895	4933	4991	5023	5074
	5090	5196	5229	5296	5338	5383	5424	5464	5521	5645	5699	5753	5798	5938	6011
	6075	6147	6212	6275	6323	6366	6484	6522	6583	6656	6734	6802	6817	6832	
.TITLE	6														
.WORD	441	468	470	482	485	486	487	488	491	492	493	494	495	496	505
	507	508	509	510	511	512	513	520	521	522	523	524	525	526	527
	528	529	530	531	532	533	534	535	536	537	538	539	540	541	542
	543	544	545	2660	3564	4759	4760	4762	4764	4767	4907	4909	4914	6092	6095
	6143	6347	6352	6375	6382	6396	6415	6426	6438	6651	6731	6862	6893	6914	6965
	7048	7143	7158	7179	7226	7656	7787	8118	8128	8191	8882	9630			

ERRORS DETECTED: 0
 DEFAULT GLOBALS GENERATED: 0

*, DEKBBC.SEQ/CRF/SOL/NL:TOC=DSKZ:DEKBBC.SML, DEKBBC.P11/DS:ERFZ
 RUN-TIME: 65 99 18 SECONDS
 RUN-TIME RATIO: 621/183=3.3
 CORE USED: 37K (73 PAGES)

