

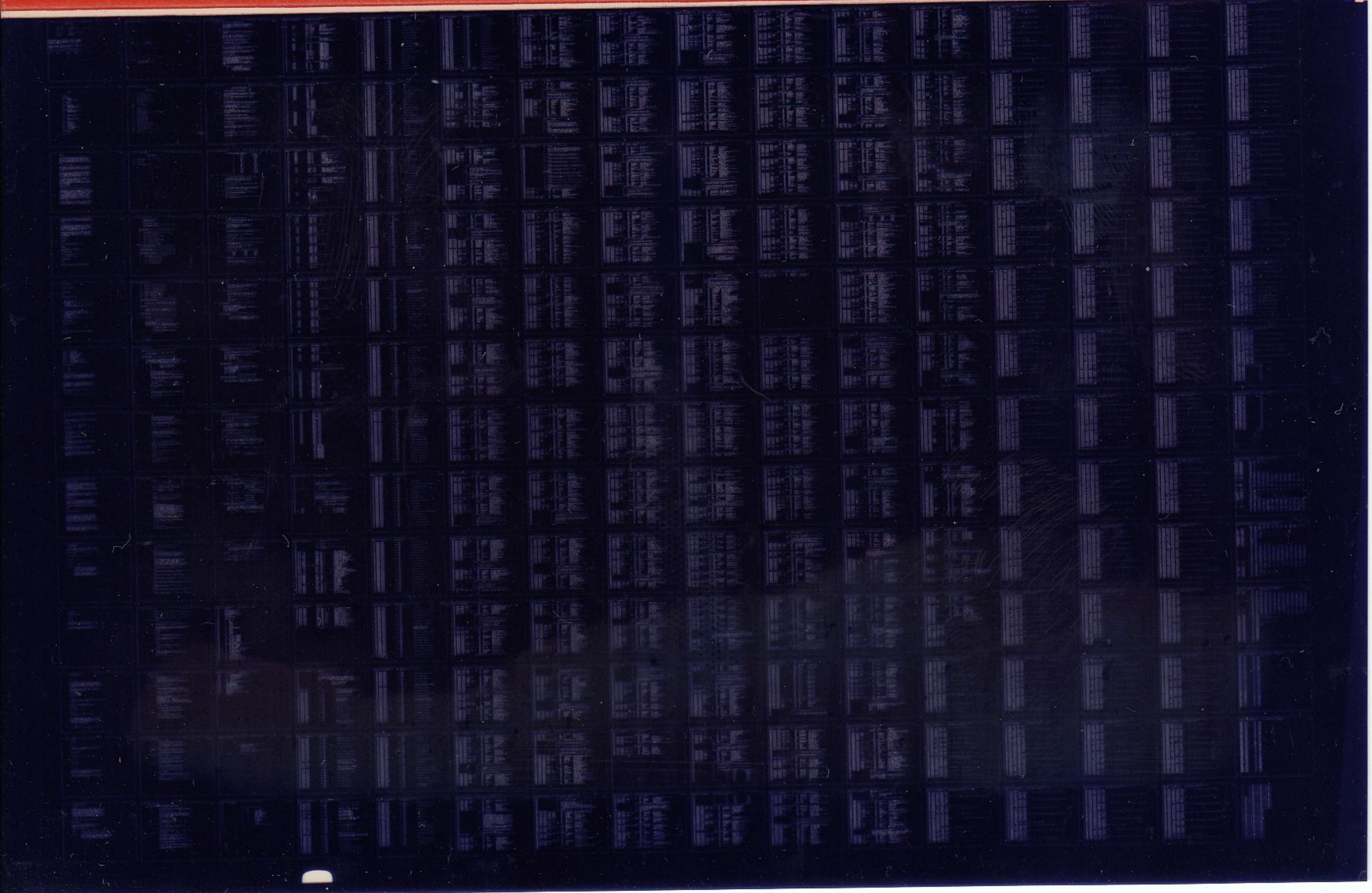
11/70-74

11/70 - 74 CPU # 2
CEKBBD0

AH-7968D-MC

COPYRIGHT 75-80
FICHE 1 OF 2

JAN 1980
digital
MADE IN USA



11/70-74

11/70 - 74 CPU # 2
CEKBBD0

AH-7968D-MC

COPYRIGHT ' 75-80

FICHE 2 OF 2

JAN 1980

digital

MADE IN USA

IDENTIFICATION

SEQ 0001

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

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.

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

COPYRIGHT (C) 1975,1979 BY DIGITAL EQUIPMENT CORPORATION

THE FOLLOWING ARE TRADEMARKS OF DIGITAL EQUIPMENT CORPORATION:

DIGITAL	PDP	UNIBUS	MASSBUS
DEC	DECUS	DECTAPE	DECX/11

C 1

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 AND HISTORY
 - 9.1 CEKBB
10. LISTINGS
 - 10.1 CEKBB

1. ABSTRACT

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

A. BASIC INSTRUCTION TESTS

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

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

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

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

B. ADVANCED INSTRUCTION AND MISCELLANEOUS LOGIC TESTS

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

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

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

C. IMPORTANT NOTE

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

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

2. REQUIREMENTS

2.1 EQUIPMENT

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

2.2 STORAGE

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

2.3 PRELIMINARY PROGRAMS

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

CEKBB REQUIRES THAT CEKBA RUN

3. LOADING PROCEDURE

3.1 METHOD

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

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE 5.1

4.2 STARTING ADDRESS

200

4.3 PROGRAM AND OPERATOR ACTION

A. CEKBA

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

B. CEK62

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

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

A. CEKBA

NONE

B. CEKBB

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

THE SWITCH SETTINGS ARE:

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

5.2 SUBROUTINE ABSTRACTS

A. CEKBA

SEE 5.2.4 AND 5.2.5

B. CEKBB

5.2.1 SPURIOUS ERROR HANDLER

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

5.2.2 SCOPE

THIS SUBROUTINE CALL (VIA AN IOT INSTRUCTION) IS PLACED BETWEEN EACH TEST. IT RECORDS THE STARTING ADDRESS OF EACH TEST IN LOCATION '\$LPADR' AND '\$LPERR' AS IT IS BEING ENTERED. IT ALSO CONTROLS TEST 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

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

H 1

SEG 0007

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

5.2.9.1 PERIPHERAL DETERMINATOR

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

5.2.9.2 INTERRUPT ENABLE ROUTINES

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

5.3 OPERATOR ACTION

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

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

6. ERRORS

6.1 ERROR HALTS AND DESCRIPTION

A. CEKBA

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

B. CEKBB

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

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

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

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

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

SEE SECTION 7.1 FOR NON-STANDARD CONFIGURATION.

6.2 ERROR RECOVERY

A. CEKBA

ERROR RECOVERY IS STRICTLY BY USER INTERVENTION.

B. CEKBB

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

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

7. RESTRICTIONS7.1 STARTING RESTRICTIONS

A. CEKBA

NONE

B. CEKBB

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

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

7.2 OPERATING RESTRICTIONS

A. CEKBA

NONE

B. CEKBB

SINCE THE PROGRAM COULD POSSIBLY DESTROY A MONITOR, IN PAGE 6, ALL LOCATIONS BETWEEN 152000 AND 157776 ARE SAVED AT THE BOTTOM OF THE PROGRAM. TO RESTORE THESE 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. MISCELLANEOUS8.1 EXECUTION TIME

A. CEKBA

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

B. CEKBB

THE FIRST PASS TAKES APPROXIMATELY 8 SECONDS. ALL SUBSEQUENT PASSES TAKE APPROXIMATELY 3 MINUTES.

8.2 STACK POINTER

STACK IS INITIALLY SET TO 1100.

8.3 PASS COUNT

A PROGRAM PASS THRU COUNT IS KEPT IN '\$PASS'.

A. CEKBA

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

B. CEKBB

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

8.4 ITERATIONS

A. CEKBA

NONE

B. CEKBB

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

8.5 SPECIAL REGISTERS

A. CEKBA

R0 IS RESERVED FOR THE TEST NUMBER.

B. CEKBB

NONE

8.6 T BIT TRAPPING

A. CEKBA

NONE

B. CEKBB

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

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

8.7 OSCILLOSCOPE SYNC POINTS

A. CEKBA

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

B. CEKBB

ONLY TESTS 1 THRU 20 CONTAIN SYNC INSTRUCTIONS.

8.8 CACHE CONTROL

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

9. PROGRAM DESCRIPTION AND HISTORY

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

DOCUMENT

PDP 11/70-74MP CPU DIAGNOSTIC PART 2

COPYRIGHT 1975, 1979
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 TIMEOUT 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) 1975,1978
DIGITAL EQUIPMENT CORP.
MAYNARD, MASS. 01754

PROGRAM BY DONALD W. MONROE

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

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

447

 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(AFIR04(1)*AFIR05(0)*(RTS:CCOP))
 IS BAD.
 D30.00 WILL CAUSE A TRAP TO LOCATION 10 AFTER STATE D10.60.
 THIS FAILURE CAN BE DIFFERENTIATED FROM THE FIRST BY TESTING
 THE REGISTER ASSOCIATED WITH BITS <2:0> OF THE OP CODE TO

1571

SEE IF IT WAS INCREMENTED.

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

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

 ROM FLOW-43,361

1614 TEST 2 RESET

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

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

 ROM FLOW-15,255,374

1672 TEST 3 MARK

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

1679

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

ROM FLOW-47,252,235,234

1733 TEST 4 ASH*DM0

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

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

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

1866 TEST 5 ASH*DM1

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

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

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

1902 TEST 6 ASH*DM2

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

ALL OTHER LOGIC HAS BEEN TESTED

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

1923 TEST 7 ASH*DM4

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

ALL OTHER LOGIC HAS BEEN TESTED.

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

1945 TEST 10 ASHC*DM0

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*DM0

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 MULTIPLIED 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 MULTIPLIED 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 MULTIPLIED BY BIT 0 OF THE MULTIPLIER.
IF RACK E50(C1) FAILS HIGH THE PROCESSOR WILL HANG UP IN
STATE MUL.20(266).
IF THE INSTRUCTION FAILS TO MULTIPLY CORRECTLY AND ONE
OF THE ABOVE CONDITIONS CANNOT BE DETERMINED THEN THE
FAILURE COULD BE IN THE INSTRUCTION DECODE ROM.

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

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

2222 TEST 13 MUL*DM1

FORK A SHOULD NOT FAIL.

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

IF STATE MUL.00 FAILS THE RESULT WILL BE BAD.

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

2254 TEST 14 DIV*DMO

FORK A SHOULD NOT FAIL.

SECTION 1

THE FIRST SECTION HAS A ZERO DIVISOR. IF STATE DIV.00 FAILS OR IRCF 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 R0 WILL END UP WITH 177777 AND R1 WILL HAVE 177774. IF BEN03 FAILS R0 WILL END UP WITH

2273

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

SECTION 3

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

SECTION 4

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

SECTION 5

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

SECTION 6

THE NEXT SECTION DIVIDES 1000000 B2 -2 TO TEST BEN05*DIV QUIT. THIS SECTION WILL ONLY FAIL IF GRAJ E5 (22(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 DVM.20

SECTION 12

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

2584 TEST 15 MTP*DM0

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

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

NOTE: THIS TEST ONLY TESTS THE CPU FUNCTIONS OF THIS INSTRUCTION.
THE MEMORY MANAGEMENT TEST VERIFIES THE INTERMODE TRANSFER.
AS FAR AS THE CPU IS CONCERNED THERE IS NO DIFFERENCE
BETWEEN MTP1 AND MTPD.

ROM FLOW-45,151,146,205

2627. TEST 16 MTP*DM1

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

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

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

2670 TEST 17 MFP*DM0

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

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

2710 TEST 20 MFP*DM2

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

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

2747 TEST 21 BPT

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

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

2781 TEST 22 BIT TEST OF PIRQ REGISTER

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

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

2824 TEST 23 PIR LEVEL 1 INTERRUPT

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

2900 TEST 24 PIR LEVEL 2 INTERRUPT

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

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

2933 TEST 25 PIR LEVEL 3 INTERRUPT

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

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

- 2966 TEST 26 PIR LEVEL 4 INTERRUPT
IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
THIS WILL ONLY HAPPEN IF TMCB E63(5) IS BAD.
IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(5)
IS BAD OR TMCA HONOR PIR 4 IS BEING HELD HIGH.
- 3000 TEST 27 PIR LEVEL 5 INTERRUPT
IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
THIS WILL ONLY HAPPEN IF TMCB E63(11) IS BAD.
IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(11)
IS BAD OR TMCA HONOR PIR 5 IS BEING HELD HIGH.
- 3035 TEST 30 PIR LEVEL 6 INTERRUPT
IF BEN13 FAILS EXECUTION WILL GO TO BRK.20.
THIS WILL ONLY HAPPEN IF EITHER TMCB E63(12)
IS BAD, OR E61(1) IS BAD.
IF THE INTERRUPT DOES NOT OCCUR LEVEL 7 IS TRYED, TO TRY
AND ISOLATE THE FAILURE BEFORE TMCB E55(9-8).
- 3077 TEST 31 PIR LEVEL 7 INTERRUPT
IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
THIS WILL ONLY HAPPEN IF TMCB E63(6) IS BAD.
IF THE INTERRUPT DOES NOT OCCUR THEN EITHER TMCB E70(6)
IS BAD OR TMCA HONOR PIR7 IS BEING HELD HIGH.
- 3083
- 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 1S-2.
IF BEN 13 FAILS EITHER TMCC AERF(1) L IS NOT GOING LOW
OR TMCB E53(11) IS BAD.
A TEST IS THEN MADE TO ENSURE THAT TMCC PRIORITY CLEAR GOES LOW.

3188

PERIPHERAL DETERMINATOR & INTERRUPT ENABLE ROUTINES

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

3191 SUBROUTINE IN A LOCATION.

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

3306 TEST 33 BR LEVEL 4 INTERRUPT

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

3370 TEST 34 BR LEVEL 5 INTERRUPT

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

3403 TEST 35 BR LEVEL 6 INTERRUPT

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

3436 TEST 36 YELLOW ZONE TRAP

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

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

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

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

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

3527 TEST 37 ROM FIELD CHECK OF PC MANIPULATOR STATES

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

3618 TEST 40 RED ZONE TRAP

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

3623

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

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

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

3726 TEST 41 BIT TEST OF STACK LIMIT REGISTER

FIRST A 125252 AND 52525 PATTERN IS PUT IN THE REGISTER TO ENSURE THAT THE REGISTER DOESN'T HAVE ANY STUCK BITS AND THAT THE DMUX SELECT AND INPUT LINES WORK.

3731

IF SCCE SL ADRS DOES NOT GET TO TMCD OR IF TMCD E28 OR F14 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

3785

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

THE FOLLOWING ARE THE TYPES OF ERRORS THAT CAN OCCUR IN THIS TEST:

TYPE	DESCRIPTION
0	RED ZONE TRAP ON YELLOW ZONE ADDRESS
2	RED ZONE TRAP ON LEGAL ADDRESS
4	YELLOW ZONE TRAP ON RED ZONE ADDRESS
6	YELLOW ZONE TRAP ON LEGAL ADDRESS

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

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

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 DATA' CANNOT BE TESTED.
THIS SIGNAL COMES UP WHEN A TRAP VECTOR IS READ IN FROM THE BUS.

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

4039 TEST 45 T BIT TRAP

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

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

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

4058 TEST 46 T BIT TRAP AND RTT

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

4174 TEST 47 PRIORITY ARBITRATION

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

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

THE FOLLOWING IS A TABLE OF CONTENTS OF THIS TEST:

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

4646 TEST 50 GPR SET 1 SELECT TEST

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

4649

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

4748 TEST 51 REGISTER SET 1 STUCK BIT TEST

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

4807 TEST 52 PSW HIGH BYTE BIT TEST

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

4837 TEST 53 SP SELECTION TEST IN SUPER AND USER MODE

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

4895 TEST 54 SUPER AND USER SP BIT TEST

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

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

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

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

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

5023 TEST 57 UPAD 7 IN USER MODE

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

5027

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

5074 TEST 60 SPL*SUPERVISOR MODE

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

5090 TEST 61 PSW CLOCKING TEST

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

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

FOLLOWING IS A TABLE TO DESCRIBE THE PSW FOR EACH SECTION

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

5196 TEST 62 ILLEGAL HALT

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

IF BEN6 FAILS EXECUTION WOULD GO TO FET.04 WHICH WOULD CAUSE THE HALT TO LOOK LIKE A NOP.

IF TMCE SET CONF GOES LOW THE PROCESSOR WILL HALT AT 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.

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

601* TEST 65 OPERATOR INTERVENTION TEST

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

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

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

6074

END OF PASS ROUTINE

6076

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

6146

SPURIOUS ERROR HANDLER

6147

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

621*

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

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

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

6277

```
*****  
ERROR HANDLER ROUTINE  
*****
```

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

6328

```
*****  
ERROR MESSAGE TYPEOUT ROUTINE  
*****
```

6329 THIS ROUTINE USES THE "ITEM CONTROL BYTE" (\$ITEMB) TO DETERMINE WHICH
ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" (\$ERRTR),
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   ;;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 JP ROUTINES

24	OPERATIONAL SWITCH SETTINGS
45	BASIC DEFINITIONS
170	CACHE REGISTER DEFINITIONS
181	CPU REGISTER DEFINITIONS
195	MEMORY MANAGEMENT DEFINITIONS
344	UNIBUS MAP REGISTER DEFINITIONS
438	TRAP CATCHER
445	STARTING ADDRESS(ES)
451	ACT11 HOOKS
477	COMMON TAGS
556	ERROR POINTER TABLE
1573	T1 SPL
1628	T2 RESET
1687	T3 MARK
1749	T4 ASH*DM0
1883	T5 ASH*DM1
1920	T6 ASH*DM2
1942	T7 ASH*DM4
1965	T10 ASHC*DM0
2099	T11 ASHC*DM1
2137	T12 MUL*DM0
2245	T13 MUL*DM1
2278	T14 DIV*DM0
2609	T15 MTP*DM0
2653	T16 MTP*DM1
2697	T17 MFP*DM0
2738	T20 MFP*DM2
2776	T21 BPT
2811	T22 BIT TEST OF PIRQ REGISTER
2855	T23 PIR LEVEL 1 INTERRUPT
2932	T24 PIR LEVEL 2 INTERRUPT
2966	T25 PIR LEVEL 3 INTERRUPT
3000	T26 PIR LEVEL 4 INTERRUPT
3035	T27 PIR LEVEL 5 INTERRUPT
3071	T30 PIR LEVEL 6 INTERRUPT
3114	T31 PIR LEVEL 7 INTERRUPT
3150	T32 UNIBUS TIMEOUT
3228	PERIPHERAL DETERMINATOR & INTERRUPT ENABLE ROUTINES
3345	T33 BR LEVEL 4 INTERRUPT
3410	T34 BR LEVEL 5 INTERRUPT
3444	T35 BR LEVEL 6 INTERRUPT
3478	T36 YELLOW ZONE TRAP
3570	T37 ROM FIELD CHECK OF PC MANIPULATOR STATES
3662	T40 RED ZONE TRAP
3771	T41 BIT TEST OF STACK LIMIT REGISTER
3828	T42 SL REGISTER COMPARATOR TEST 1
3951	T43 ODD ADDRESS ERROR
4050	T44 ILLEGAL INSTRUCTIONS
4194	T45 T BIT TRAP
4257	T46 T BIT TRAP AND RTT
4277	T47 PRIORITY ARBITRATION
4754	T50 GPR SET 1 SELECT TEST
4857	T51 REGISTER SET 1 STUCK BIT TEST
4917	T52 PSW HIGH BYTE BIT TEST
4948	T53 SP SELECTION TEST IN SUPER AND USER MODE
5007	T54 SUPER AND USER SP BIT TEST

5046	T55	MTP*DMO*DF6*PREVIOUS MODE(SUPER*USER)
5105	T56	MFP*DMO*DF6*PREVIOUS MODE SUPER
5138	T57	UPAD 7 IN USER MODE
5190	T60	SPL*SUPERVISOR MODE
5207	T61	PSW CLOCKING TEST
5324	T62	ILLEGAL HALT
5358	T63	WAIT
5419	T64	CHECK MFPT INSTRUCTION (KB-11E/EM ONLY)
5449	T65	OPERATOR INTERVENTION TEST
5513		END OF PASS ROUTINE
5586		SPURIOUS ERROR HANDLER
5650		SCOPE HANDLER ROUTINE
5714		ERROR HANDLER ROUTINE
5762		ERROR MESSAGE TIMEOUT ROUTINE
5806		STACK LIMIT TEST TYPE OUT ROUTINE
5924		MONITOR RESTORE ROUTINE
5962		CHECK TEST SEQUENCE ROUTINE
6023		TYPE ROUTINE
6096		BINARY TO OCTAL (ASCII) AND TYPE
6174		CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
6242		TRAP DECODER
6257		TRAP TABLE
6272		POWER DOWN AND JP ROUTINES

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

000001
177400

.TITLE PDP 11/70-74MP CPU DIAGNOSTIC PART 2
:*COPYRIGHT (C) 1975,1979
:*DIGITAL EQUIPMENT CORP.
:*MAYNARD, MASS. 01754
:*
:*PROGRAM BY DONALD W. MONROE
:*
:*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
:*PACKAGE (MAINDEC-11-DZQAC-A5).
:*
\$TN 1 \$SWR-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	NOT USED
2	NOT USED
1	NOT USED
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      000006  SP=%6           ;;STACK POINTER
80      .EQUIV  SP,KSP          ;;KERNEL STACK POINTER
81      .EQUIV  SP,SSP          ;;SUPERVISOR STACK POINTER
82      .EQUIV  SP,USP          ;;USER STACK POINTER
83      000007  PC %7           ;;PROGRAM COUNTER
84
85      ;*PRIORITY LEVEL DEFINITIONS
86      000000  PRO 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

;*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

;*BASIC 'CPU' TRAP VECTOR ADDRESSES
ERRVEC= 4 ;:TIME OUT AND OTHER ERRORS

150
151
152 000004

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	TPVEC= 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	177740	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

208	177600	JIPDR0-	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	SDPDR2= 172224
266	172226	SDPDR3= 172226
267	172230	SDPDR4= 172230
268	172232	SDPDR5= 172232
269	172234	SDPDR6= 172234
270	172236	SDPDR7= 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

326		
327		
328		
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

;*RTL 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'

337		
338		
339		
340		
341		
342		
343		
344		
345		
346		
347		
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  
436  
437          000600  
438  
439  
440  
441  
442          000200  
443  
444          000200 000137 004742  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466          000204  
467          000046  
468 000046 032572  
469          000052  
470 000052 000000  
471          000204  
  
          .SBTTL TRAP CATCHER  
          .-0  
          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"  
          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS  
          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS  
  
          .SBTTL STARTING ADDRESS(ES)  
          .-200  
          JMP @#START          ;; JUMP TO STARTING ADDRESS OF PROGRAM  
          ;*****  
  
          .SBTTL ACT11 HOOKS  
  
          ;*THE FOLLOWING LOCATIONS ARE SETUP TO BE USED WITH ACT11  
          ;*  
          ;*LOCATION 46 WILL CONTAIN THE ADDRESS OF THE 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  
  
          $SVPC=.          ;;SAVE LOCATION COUNTER  
          . 46          ;;SET LOCATION COUNTER  
          .WORD $ENDAD          ;;SET LOC.46 TO ADDRESS $ENDAD  
          .-52          ;;SET LOCATION COUNTER  
          .WORD 0          ;;SET LOC.52 TO ZERO  
          .-$SVPC          ;; 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 $STNM: .BYTE 0          ::CONTAINS THE TEST NUMBER
484 001103 000 $ERFLG: .BYTE 0          ::CONTAINS ERROR FLAG
485 001104 000000 $ICNT: .WORD 0          ::CONTAINS SUBTEST ITERATION COUNT
486 001106 000000 $LPADP: .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 C          ::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 $TMP0: .WORD 0          ::USER DFFINED
511 001164 000000 $TMP1: .WORD 0          ::USER DEFINED
512 001166 000000 $TMP2: .WORD 0          ::USER DFFINED
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: .ASCII <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: .ASCII <12>          ::LINE FEED
520 001206 000000 $ERPSW: .WORD 0          ::ERROR PSW
521 001210 000000 $$STNM: .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	INT5VEC: .WORD	0	:BR 5 INTERRUPT VECTOR
529	001230	000000	INT5ST: .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	PLKSTA: .WORD	172540	:ADDRESS OF PROG LINE CLOCK STATUS REG
544	001266	000104	PLKVEC: .WORD	104	:ADDRESS OF PROG LINE CLOCK VECTOR
545	001270	000000	NEXTTGT: .WORD	0	:ADDRESS OF NEXT TEST
546	001272	000	KB11E: .BYTE	0	:KB-11E/EM WITHOUT MP CACHE
547	001273	000	KB11EM: .BYTE	0	:KB-11E/EM WITH MP MODS
548					
549			:OPCODE FOR MFPT INSTRUCTION (AVAILABLE ON KB11-E AND KB11-EM ONLY)		
550	000007		MFPT-7		

```

551      ;:*****
552
553      .SBTTL  ERROR POINTER TABLE
554
555      ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
556      ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
557      ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
558      ;*NOTE1:      IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
559      ;*NOTE2:      EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
560
561      ;*      EM      ;;POINTS TO THE ERROR MESSAGE
562      ;*      DH      ;;POINTS TO THE DATA HEADER
563      ;*      DT      ;;POINTS TO THE DATA
564      ;*      DF      ;;POINTS TO THE DATA FORMAT
565
566
567      001274      $ERRTB:
568
569      001274      036546      ITEM1:  EM1      ;:EITHER SPL FAILED OR BITS STUCK
570      001276      036614      DH1      ;:ERRORPC SPL 5 PSW   SPL 2 PSW  TEST NUMBER
571      ;:      EXPECT ACTUAL   EXPECT ACTUAL
572      001300      036734      DT1      ;:$ERRPC,$SPR5,$SERPSW,$SPR2,$TMPO
573      001302      036752      ITEM2:  EM2      ;:PR5 LOADS OK BUT PR2 DOESN'T
574      001304      036614      DH1
575      001306      036734      DT1
576      001310      036773      ITEM3:  EM3      ;:PR2 LOADS OK BUT PR5 DOESN'T
577      001312      036614      DH1
578      001314      036734      DT1
579      001316      037020      ITEM4:  EM4      ;:FORK A FAILED TO D30.00
580      001320      037063      DH4      ;:ERRORPC TEST NUMBER
581      001322      037110      DT4      ;:$ERRPC,$STSTNM
582      001324      037116      ITEM5:  EM5      ;:FORK A FAILED TO RSD.02
583      001326      037063      DH4
584      001330      037110      DT4
585      001332      037204      ITEM6:  EM6      ;:RESET DID NOT WORK
586      001334      037063      DH4
587      001336      037110      DT4
588      001340      037237      ITEM7:  EM7      ;:FORK A FAILED INTO TRP.02
589      001342      037063      DH4
590      001344      037110      DT4
591      001346      037253      ITEM10: EM10     ;:FORK A FAILED INTO WAT.00
592      001350      037063      DH4
593      001352      037110      DT4
594      001354      037267      ITEM11: EM11     ;:FORK A FAILED TO MTP.00
595      001356      037063      DH4
596      001360      037110      DT4
597      001362      037326      ITEM12: EM12     ;:FORK A FAILED TO MFP.80
598      001364      037063      DH4
599      001366      037110      DT4
600      001370      037365      ITEM13: EM13     ;:PCB DID NOT LOAD FROM R5
601      001372      037063      DH4
602      001374      037110      DT4
603      001376      037415      ITEM14: EM14     ;:MARK DID NOT LOAD SP PROPERLY
604      001400      037445      DH14     ;:ERRORPC      SP      TEST NUMBER
605      ;:      EXPECT ACTUAL
606      001402      037526      DT14     ;:$ERRPC,$REG1,$REG0,$STSTNM

```

607	001404	037540	ITEM15:	EM15	:R5 DID NOT LOAD PROPERLY
608	001406	037570		DH15	:ERRORPC R5 TEST NUMBER
609	001410	037526		DT14	
610	001412	037651	ITEM16:	EM16	:FORK A FAILED TO RSD.00
611	001414	037063		DH4	
612	001416	037110		DT4	
613	001420	037705	ITEM17:	EM17	:FORK A FAILED TO D67.01
614	001422	037063		DH4	
615	001424	037110		DT4	
616	001426	037762	ITEM20:	EM20	:R1 SHIFTED RIGHT INSTEAD OF LEFT
617	001430	037110		DH4	
618	001432	037110		DT4	
619	001434	040053	ITEM21:	EM21	:R1 DID NOT SHIFT
620	001436	037063		DH4	
621	001440	037110		DT4	
622	001442	040073	ITEM22:	EM22	:R1 SHIFTED BUT CARRY DID NOT SET
623	001444	037063		DH4	
624	001446	037110		DT4	
625	001450	040170	ITEM23:	EM23	:R1 SHIFTED LEFT INSTEAD OF RIGHT
626	001452	037063		DH4	
627	001454	037110		DT4	
628	001456	040260	ITEM24:	EM24	:SHIFT RIGHT DID NOT SIGN FILL
629	001460	037063		DH4	
630	001462	037110		DT4	
631	001464	040313	ITEM25:	EM25	:R1 SHIFTED BUT DON'T KNOW WHERE
632	001466	040345		DH25	:ERRORPC R5 TEST NUMBER
633					: EXPECT ACTUAL
634	001470	040420		DT25	:SERRPC,\$TMP1,\$REG1,\$STSTNM
635	001472	040432	ITEM26:	EM26	:SHIFT OK BUT CARRY DID NOT LOAD
636	001474	037063		DH4	
637	001476	037110		DT4	
638	001500	040472	ITEM27:	EM27	:ASH.20 DID NOT LOAD CC'S CORRECTLY
639	001502	041456		DH42	:ERRORPC PSW TEST NUMBER
640					: EXPECT ACTUAL
641	001504	041534		DT42	:SERRPC,\$TMP0,\$ERPSW,\$STSTNM
642	001506	040534	ITEM30:	EM30	:R1 SHIFTED WITH A SHIFT COUNT OF 0
643	001510	037063		DH4	
644	001512	037110		DT4	
645	001514	040572	ITEM31:	EM31	:ASH.40 DID NOT LOAD CC'S CORRECTLY
646	001516	041456		DH42	
647	001520	041534		DT42	
648	001522	040634	ITEM32:	EM32	:FORK A FAILED TO RSD.00
649	001524	037063		DH4	
650	001526	037110		DT4	
651	001530	040700	ITEM33:	EM33	:STATE ASH.00 FAILED
652	001532	037063		DH4	
653	001534	037110		DT4	
654	001536	040716	ITEM34:	EM34	:FORK B FAILED INTO MUL.00
655	001540	037063		DH4	
656	001542	037110		DT4	
657	001544	040751	ITEM35:	EM35	:FORK B FAILED TO RSD.00
658	001546	037063		DH4	
659	001550	037110		DT4	
660	001552	041067	ITEM36:	EM36	:FORK A FAILED TO RSD.00
661	001554	037063		DH4	
662	001556	037110		DT4	

663	001560	041152	ITEM37: EM37	:RACE E45 BAD (AFIRO4(1)*MUL:ASHC+MFP))
664	001562	037063	DH4	
665	001564	037110	DT4	
666	001566	041222	ITEM40: EM40	:RACE E33 BAD (AFIRO5(1)*(MUL:ASHC+MFP))
667	001570	037063	DH4	
668	001572	037110	DT4	
669	001574	041272	ITEM41: EM41	:R0 DID NOT SIGN FILL ON RIGHT SHIFT
670	001576	041335	DH41	:ERROR PC RO TEST NUMBER
671				: EXPECT ACTUAL
672	001600	041414	DT41	:SERRPC,\$TMP0,\$REG0,\$STSTNM
673	001602	041426	ITEM42: EM42	:BAD CC'S ON RIGHT SHIF?
674	001604	041456	DH42	:ERRORPC PSW TEST NUMBER
675				: EXPECT ACTUAL
676	001606	041534	DT42	:SERRPC,\$TMP0,\$ERPSW,\$STSTNM
677	001610	041546	ITEM43: EM43	:R0<0> DID NOT GO TO R1<15>
678	001612	041600	DH43	:ERRORPC RO R1 TEST NUMBER
679				: EXPECT ACTUAL EXPECT ACTUAL
680	001614	041710	DT43	:SERRPC,\$TMP0,\$REG0,\$TMP1,\$REG1,\$STSTNM
681	001616	041726	ITEM44: EM44	:R0 DID NOT SHIFT LEFT PROPERLY
682	001620	041600	DH43	
683	001622	041710	DT43	
684	001624	041772	ITEM45: EM45	:BAD CC'S ON LEFT SHIFT
685	001626	041456	DH42	
686	001630	041534	DT42	
687	001632	040313	ITEM46: EM25	:R1 DID NOT SHIFT LEFT PROPERLY
688	001634	041600	DH43	
689	001636	041710	DT43	
690	001640	042150	ITEM47: EM47	:BAD CC'S ON NO SHIFT
691	001642	041456	DH42	
692	001644	041534	DT42	
693	001646	042175	ITEM50: EM50	:R1 DID NOT ROTATE PROPERLY
694	001650	041600	DH43	
695	001652	041710	DT43	
696	001654	042227	ITEM51: EM51	:BITS STUCK IN SC (52 PATTERN)
697	001656	042267	DH51	:ERRORPC RO R1 C BIT
698				: EXPECT ACTUAL EXPECT ACTUAL EXPECT ACTUAL
699	001660	042426	DT51	:SERRPC,\$TMP0,\$REG0,\$TMP1,\$REG1,\$TMP2,\$ERPSW,\$STSTNM
700	001662	042450	ITEM52: EM52	:BITS STUCK IN SC (25 PATTERN)
701	001664	042267	DH51	
702	001666	042426	DT51	
703	001670	042523	ITEM53: EM53	:IRCB B FORK MUX INPUT B3 NOT GOING LOW
704	001672	042267	DH51	
705	001674	042426	DT51	
706	001676	042556	ITEM54: EM54	:STATE ASC.00 FAILED
707	001700	041600	DH43	
708	001702	041710	DT43	
709	001704	042574	ITEM55: EM55	:FORK A FAILED TO RSD.00
710	001706	037063	DH4	
711	001710	037110	DT4	
712	001712	042652	ITEM56: EM56	:EITHER GRAD DR00 STUCK H OR RACK E64 BAD
713	001714	041600	DH43	
714	001716	041710	DT43	
715	001720	042723	ITEM57: EM57	:EITHER GRAD DR00 STUCK LOW OR RACK E64 BAD
716	001722	041600	DH43	
717	001724	041710	DT43	
718	001726	042773	ITEM60: EM60	:EITHER GRAJ SC=0 NOT GETTING TO RACK E50

719	001730	041600		DH43	,OR RACK E50 BAD
720	001732	041710		DT43	
721	001734	043074	ITEM61:	EM61	;INSTRUCTION FAILED TO LOAD R0 & R1 CORRECTLY
722	001736	041600		DH43	;ON POSITIVE
723	001740	041710		DT43	
724	001742	043152	ITEM62:	EM62	;BAD CC'S
725	001744	041456		DH42	
726	001746	041534		DT42	
727	001750	043214	ITEM63:	EM63	;R0 DID NOT LOAD ON NEG.
728	001752	041600		DH43	
729	001754	041710		DT43	
730	001756	043254	ITEM64:	EM64	;R1 DID NOT LOAD ON NEG.
731	001760	041600		DH43	
732	001762	041710		DT43	
733	001764	043314	ITEM65:	EM65	;BAD CC'S DUE TO STATE MUL.50
734	001766	041456		DH42	
735	001770	041534		DT42	
736	001772	043351	ITEM66:	EM66	;C DID NOT SET ON OVERFLOW
737	001774	037063		DH4	
738	001776	037110		DT4	
739	002000	043402	ITEM67:	EM67	;C DID NOT SET ON UNDERFLOW
740	002002	037063		DH4	
741	002004	037110		DT4	
742	002006	043434	ITEM70:	EM70	;STATE MUL.00 FAILED
743	002010	041600		DH43	
744	002012	041710		DT43	
745	002014	042574	ITEM71:	EM55	;BAD FIELD IN IR DECODE ROM
746	002016	037063		DH4	
747	002020	037110		DT4	
748	002022	043452	ITEM72:	EM72	;STATE ASH.30 DID NOT LOAD CC'S CORRECTLY
749	002024	041456		DH42	
750	002026	041534		DT42	
751	002030	043514	ITEM73:	EM73	;STATE ASH.41 FAILED
752	002032	040345		DH25	
753	002034	040420		DT25	
754	002036	043532	ITEM74:	EM74	;BAD CC'S DUE TO STATE ASH.41
755	002040	041456		DH42	
756	002042	041534		DT42	
757	002044	043574	ITEM75:	EM75	;BEN2 FAILED
758	002046	037063		DH4	
759	002050	037110		DT4	
760	002052	043661	ITEM76:	EM76	;BAD CC'S DUE TO DVE.00
761	002054	041456		DH42	
762	002056	041534		DT42	
763	002060	043704	ITEM77:	EM77	;BAD CC'S DUE TO DVC.70
764	002062	041456		DH42	
765	002064	041534		DT42	
766	002066	043727	ITE100:	EM100	;BEN16 FAILED
767	002070	037063		DH4	
768	002072	037110		DT4	
769	002074	043757	ITE101:	EM101	;BEN4 FAILED
770	002076	037063		DH4	
771	002100	037110		DT4	
772	002102	044103	ITE102:	EM102	;QUOTIENT OK REMAINDER BAD
773	002104	041600		DH43	
774	002106	041710		DT43	

775	002110	044155	ITE103:	EM103	;BEN3 FAILED
776	002112	037063		DH4	
777	002114	037110		DT4	
778	002116	044204	ITE104:	EM104	;BEN04 STUCK TO DIV SUB
779	002120	037063		DH4	
780	002122	037110		DT4	
781	002124	044274	ITE105:	EM105	;CAN'T DETERMINE CAUSE OF FAILURE
782	002126	041600		DH43	
783	002130	041710		DT43	
784	002132	044325	ITE106:	EM106	;BEN16 FAILED TO DIV.50
785	002134	037063		DH4	
786	002136	037110		DT4	
787	002140	044274	ITE107:	EM105	;EM107 SAME AS EM105
788	002142	041600		DH43	
789	002144	041710		DT43	
790	002146	044415	ITE110:	EM110	;BEN04*-N FAILED
791	002150	037063		DH4	
792	002152	037110		DT4	
793	002154	044505	ITE111:	EM111	;BEN02 FAILED
794	002156	037063		DH4	
795	002160	037110		DT4	
796	002162	044535	ITE112:	EM112	;BEN05 FAILED
797	002164	037063		DH4	
798	002166	037110		DT4	
799	002170	044655	ITE113:	EM113	;BEN16 FAILED (RACK E50(B0))
800	002172	037063		DH4	
801	002174	037110		DT4	
802	002176	044704	ITE114:	EM114	;BEN16 FAILED (RACK E64(B0))
803	002200	037063		DH4	
804	002202	037110		DT4	
805	002204	044733	ITE115:	EM115	;ROM STATE FAILED
806	002206	041600		DH43	
807	002210	041710		DT43	
808	002212	044103	ITE116:	EM102	;QUOTIENT OK, REMAINDER BAD
809	002214	041600		DH43	
810	002216	041710		DT43	
811	002220	045001	ITE117:	EM117	;BAD CC'S IN DVC.90 OR RACK E63 BAD
812	002222	041456		DH42	
813	002224	041534		DT42	
814	002226	045056	ITE120:	EM120	;BEN05 DIV QUIT (N(0)*SR15(0)) DID NOT
815	002230	037063		DH4	;GO LOW OR RACK E63(CC) STUCK HIGH
816	002232	037110		DT4	
817	002234	045145	ITE121:	EM121	;CC'S BAD DUE TO DIV.30 OR DVE.20
818	002236	041456		DH42	
819	002240	041534		DT42	
820	002242	045215	ITE122:	EM122	;GRAJ E5 BAD
821	002244	037063		DH4	
822	002246	037110		DT4	
823	002250	045255	ITE123:	EM123	;RACK E63(D0) STUCK LOW
824	002252	037063		DH4	
825	002254	037110		DT4	
826	002256	045304	ITE124:	EM124	;CC'S DID NOT LOAD PROPERLY
827	002260	041456		DH42	
828	002262	041534		DT42	
829	002264	045336	ITE125:	EM125	;EITHER GRAJ E5(N(1)*SR15(1)) BAD
830	002266	041600		DH43	;OR ROM STATE BAD

831	002270	041710		
832	002272	044103	ITE126:	DT43
833	002274	041600		EM102
834	002276	041710		DH43
835	002300	045411	ITE127:	DT43
836	002302	041600		EM127
837	002304	041710		DH43
838	002306	045463	ITE130:	DT43
839	002310	041600		EM130
840	002312	041710		DH43
841	002314	044103	ITE131:	DT43
842	002316	041600		EM102
843	002320	041710		DH43
844	002322	045463	ITE132:	DT43
845	002324	041600		EM130
846	002326	041710		DH43
847	002330	044103	ITE133:	DT43
848	002332	041600		EM102
849	002334	041710		DH43
850	002336	045524	ITE134:	DT43
851	002340	041456		EM134
852	002342	041534		DH42
853	002344	045573	ITE135:	DT42
854	002346	041335		EM135
855	002350	041414		DH41
856	002352	045602	ITE136:	DT41
857	002354	037063		EM136
858	002356	037110		DH4
859	002360	045304	ITE137:	DT4
860	002362	041456		EM124
861	002364	041534		DH42
862	002366	045635	ITE140:	DT42
863	002370	037063		EM140
864	002372	037110		DH4
865	002374	045672	ITE141:	DT4
866	002376	037063		EM141
867	002400	037110		DH4
868	002402	045731	ITE142:	DT4
869	002404	037445		EM142
870	002406	037526		DH14
871	002410	045757	ITE143:	DT14
872	002412	037063		EM143
873	002414	037110		DH4
874	002416	046013	ITE144:	DT4
875	002420	037063		EM144
876	002422	037110		DH4
877	002424	046042	ITE145:	DT4
878	002426	037063		EM145
879	002430	037110		DH4
880	002432	046101	ITE146:	DT4
881	002434	037063		EM146
882	002436	037110		DH4
883	002440	046140	ITE147:	DT4
884	002442	041456		EM147
885	002444	041534		DH42
886	002446	045635	ITE150:	DT42
				EM140

;QUOT. OK REMAINDER BAD
;QUOT. BAD, REMAINDER OK
;QUOTIENT BAD
;QUOT. OK, REMAINDER BAD
;QUOT. BAD
;QUOT. OK, REMAIN. BAD
;BAD CC'S ON DIVISION OVERFLOW
;RO DID NOT LOAD CORRECTLY
;THE SP DID NOT INCREMENT
;CC'S DID NOT LOAD CORRECTLY
;FORK A FAILED
;STATE MTF.10 FAILED
;SP LOADED INCORRECTLY
;STATE MTF.00 DID NOT PUT PCB IN DR
;FORK A FAILED
;STATE MFP.10 DID NOT DEC. THE SP
;RO DID NOT GET PUT ON THE STACK
;BAD CC'S
;RACF X/CLASS DOES NOT GO HIGH

887	002450	037063	DH4	
888	002452	037110	DT4	
889	002454	046171	ITE151:	EM151 ;STATE MFP.00 IS BAD
890	002456	037063	DH4	
891	002460	037110	DT4	
892	002462	046204	ITE152:	EM152 ;BAD CC'S
893	002464	041456	DH42	
894	002466	041534	DT42	
895	002470	037651	ITE153:	EM16 ;RACE X/CLASS DOES NOT GO HIGH
896	002472	037063	DH4	
897	002474	037110	DT4	
898	002476	042574	ITE154:	EM55 ;R(NUL:ASHC+MFP) FIELD OF IR ROM BAD
899	002500	037063	DH4	
900	002502	037110	DT4	
901	002504	046233	ITE155:	EM155 ;STATE TRP.01 FAILED
902	002506	037063	DH4	
903	002510	037110	DT4	
904	002512	046264	ITE156:	EM156 ;TRAP VECTOR DECODE FAILED
905	002514	037063	DH4	
906	002516	037110	DT4	
907	002520	046404	ITE157:	EM157 ;STATE TRP.01 FAILED
908	002522	037063	DH4	
909	002524	037110	DT4	
910	002526	046435	ITE160:	EM160 ;STATE TRP.01 FAILED
911	002530	037063	DH4	
912	002532	037110	DT4	
913	002534	046466	ITE161:	EM161 ;TRAP VECTOR DECODE FAILED
914	002536	037063	DH4	
915	002540	037110	DT4	
916	002542	046563	ITE162:	EM162 ;STATE TRP.01 FAILED
917	002544	037063	DH4	
918	002546	037110	DT4	
919	002550	046614	ITE163:	EM163 ;BIT FAILED IN PIRQ REG
920	002552	046643	DH163	;ERRORPC PIRQ TEST NUMBER
921				; EXPECT ACTUAL
922	002554	046722	DT163	;SERRPC,\$TMP0,\$EPIRQ,\$\$STSTM
923	002556	046734	ITE164:	EM164 ;STATE TST.10 HAS BAD BEN FIELD
924	002560	037063	DH4	
925	002562	037110	DT4	
926	002564	046765	ITE165:	EM165 ;HONOR PIR 1 DOES NOT GO LOW
927	002566	037063	DH4	
928	002570	037110	DT4	
929	002572	047047	ITE166:	EM166 ;PIR 6 WORKS BUT 4 & 1 DON'T
930	002574	037063	DH4	
931	002576	037110	DT4	
932	002600	047154	ITE167:	EM167 ;TMCA ABOVE BR7 MIGHT BE STUCK LOW
933	002602	037063	DH4	
934	002604	037110	DT4	
935	002606	047216	ITE170:	EM170 ;TMCE BRQ CLDCK MIGH BE STUCH LOW
936	002610	037063	DH4	
937	002612	037110	DT4	
938	002614	047260	ITE171:	EM171 ;BEN 13 FAILED TO PUP.00
939	002616	037063	DH4	
940	002620	037110	DT4	
941	002622	047401	ITE172:	EM172 ;BEN 13 FAILED TO SFR.00
942	002624	037063	DH4	

943	002626	037110		
944	002630	042021	ITE173:	DT4 EM46 ;MFPT FAILED TO LOAD R0 CORRECTLY
945	002632	042054		DH46
946	002634	042136		DT46
947	002636	047654	ITE174:	EM174 ;PIR 2 DID NOT INTERRUPT
948	002640	037063		DH4
949	002642	037110		DT4
950	002644	047736	ITE175:	EM175 ;BEN 13 FAILED
951	002646	037063		DH4
952	002650	037110		DT4
953	002652	047753	ITE176:	EM176 ;LEVEL 1 INT. WHEN CPU LEVEL 1 ENABLED
954	002654	050145		DH201 ;ERRORPC PIRQ TEST NUMBER
955	002656	050176		DT201 ;SERRPC,\$EPIRQ,\$\$TSTNM
956	002660	050015	ITE177:	EM177 ;PIR 3 DID NOT INTERRUPT
957	002662	037063		DH4
958	002664	037110		DT4
959	002666	047736	ITE200:	EM175 ;BEN 13 FAILED
960	002670	037063		DH4
961	002672	037110		DT4
962	002674	050077	ITE201:	EM201 ;LEVEL 2 WHEN CPU LEVEL 2 ENABLED
963	002676	050145		DH201
964	002700	050176		DT201
965	002702	050206	ITE202:	EM202 ;PIR 4 DID NOT INTERRUPT
966	002704	037063		DH4
967	002706	037110		DT4
968	002710	047736	ITE203:	EM175 ;BEN 13 FAILED
969	002712	037063		DH4
970	002714	037110		DT4
971	002716	050270	ITE204:	EM204 ;LEVEL 3 WHEN CPU LEVEL 3 ENABLED
972	002720	050145		DH201
973	002722	050176		DT201
974	002724	050336	ITE205:	EM205 ;PIR 5 DID NOT INTERRUPT
975	002726	037063		DH4
976	002730	037110		DT4
977	002732	047736	ITE206:	EM175 ;BEN 13 FAILED
978	002734	037063		DH4
979	002736	037110		DT4
980	002740	050421	ITE207:	EM207 ;LEVEL 4 INTERRUPT WHEN NOT SUPPOSE TO
981	002742	050145		DH201
982	002744	050176		DT201
983	002746	050467	ITE210:	EM210 ;FAILURE AFTER *MCB E70
984	002750	037063		DH4
985	002752	037110		DT4
986	002754	050535	ITE211:	EM211 ;FAILURE IN TMCB E70 OR BEFORE
987	002756	037063		DH4
988	002760	037110		DT4
989	002762	050616	ITE212:	EM212 ;BEN 13 FAILED
990	002764	037063		DH4
991	002766	037110		DT4
992	002770	050662	ITE213:	EM213 ;LEVEL 5 INTERRUPT WHEN NOT SUPPOSE TO
993	002772	050145		DH201
994	002774	050176		DT201
995	002776	050730	ITE214:	EM214 ;LEVEL 7 DID NOT INTERRUPT
996	003000	037063		DH4
997	003002	037110		DT4
998	003004	047736	ITE215:	EM175 ;BEN 13 FAILED

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

1055	003166	052754	ITE240: EM240	:BEN 13 FAILED TO BRK.80
1056	003170	037063	DH4	
1057	003172	037110	DT4	
1058	003174	053056	ITE241: EM241	:NO RED ZONE ON STACK OVERFLOW
1059	003176	037063	DH4	
1060	003200	037110	DT4	
1061	003202	053167	ITE242: EM242	:NO RED ZONE WHEN SL REG>BADDR
1062	003204	037063	DH4	
1063	003206	037110	DT4	
1064	003210	053323	ITE243: EM243	:52400 PATTERN FAILED IN SL REG
1065	003212	053373	DH243	:ERRORPC SL REG TEST NUMBER
1066				: EXPECT ACTUAL
1067	003214	053456	DT243	:SERRPC,\$TMP0,E2STKLMT,\$\$STNM
1068	003216	053470	ITE244: EM244	:125000 PATTERN FAILED IN SL REG
1069	003220	053373	DH243	
1070	003222	053540	DT244	
1071	003224	053552	ITE245: EM245	:SERRPC,\$TMP0,E1STKLMT,\$\$STNM
1072	003226	037063	DH4	:BR SELECTED INSTEAD OF SL
1073	003230	037110	DT4	
1074	003232	053645	ITE246: EM246	:SL AND PB BOTH SELECTED
1075	003234	037063	DH4	
1076	003236	037110	DT4	
1077	003240	053766	ITE247: EM247	:PSW SELECTED INSTEAD OF SL
1078	003242	037063	DH4	
1079	003244	037110	DT4	
1080	003246	054077	ITE250: EM250	:WHAT HAPPENED?
1081	003250	054124	DH250	:BOTH PATTERNS FAILED
1082	003252	054242	DT250	
1083	003254	054260	ITE251: EM251	:YEL ZONE IN RED REGION (SP-330)
1084	003256	037063	DH4	
1085	003260	037110	DT4	
1086	003262	054425	ITE252: EM252	:YEL ZONE IN RED REGION (SP=240)
1087	003264	037063	DH4	
1088	003266	037110	DT4	
1089	003270	054475	ITE253: EM253	:YEL ZONE IN RED REGION (SP-140)
1090	003272	037063	DH4	
1091	003274	037110	DT4	
1092	003276	054545	ITE254: EM254	:RED ZONE IN YELLOW REGION (SP-376)
1093	003300	037063	DH4	
1094	003302	037110	DT4	
1095	003304	054635	ITE255: EM255	:CPU ERR REG BIT 4 DOES NOT SET
1096	003306	054714	DH255	
1097	003310	041414	DT41	
1098	003312	054775	ITE256: EM256	:CPU ERR REG DOES NOT CLEAR
1099	003314	037063	DH4	
1100	003316	037110	DT4	
1101	003320	055032	ITE257: EM257	:CPU ERROR BIT 3 DOES NOT SET
1102	003322	054714	DH255	
1103	003324	041414	DT41	
1104	003326	055103	ITE260: EM260	:CPU ERROR BIT 3 DOES NOT CLEAR
1105	003330	037063	DH4	
1106	003332	037110	DT4	
1107	003334	055146	ITE261: EM261	:CPU ERROR BIT 2 DOES NOT SET
1108	003336	054714	DH255	
1109	003340	041414	DT41	
1110	003342	055220	ITE262: EM262	:CPU ERROR BIT 2 DOES NOT CLEAR

1111	003344	037063	DH4	
1112	003346	037110	DT4	
1113	003350	055262	ITE263: EM263	
1114	003352	055451	DH263	
1115	003354	000000	0	
1116	003356	056074	ITE264: EM264	;GOING TO NEXT TEST
1117	003360	000000	0	
1118	003362	000000	0	
1119	003364	056117	ITE265: EM265	;NEITHER - BYIN NOR DATI CAUSED ODD ADDR.
1120	003366	037063	DH4	
1121	003370	037110	DT4	
1122	003372	056435	ITE266: EM266	;DATI TRAPPED BUT - BYIN DIDN'T
1123	003374	037063	DH4	
1124	003376	037110	DT4	
1125	003400	056553	ITE267: EM267	;BOTH DATI & DATO FAILED
1126	003402	037063	DH4	
1127	003404	037110	DT4	
1128	003406	056633	ITE270: EM270	;DATO WORKS BUT DATI FAILED
1129	003410	037063	DH4	
1130	003412	037110	DT4	
1131	003414	056731	ITE271: EM271	;NO TRAP ON DATO
1132	003416	037063	DH4	
1133	003420	037110	DT4	
1134	003422	057061	ITE272: EM272	;NO TRAP ON SM357*SRC1 DATI
1135	003424	037063	DH4	
1136	003426	037110	DT4	
1137	003430	057120	ITE273: EM273	;ODD ADDR BIT IN CPU ERROR FAILED
1138	003432	037063	DH4	
1139	003434	037110	DT4	
1140	003436	057166	ITE274: EM274	;NO TRAP ON DATO
1141	003440	037063	DH4	
1142	003442	037110	DT4	
1143	003444	057206	ITE275: EM275	;T BIT TRAP FAILED
1144	003446	037063	DH4	
1145	003450	037110	DT4	
1146	003452	057556	ITE276: EM276	;T BIT NEVER SET
1147	003454	037063	DH4	
1148	003456	037110	DT4	
1149	003460	057607	ITE277: EM277	;PS<08> DID NOT DISABLE T TRAP (KB-11E/EM ONLY)
1150	003462	037063	DH4	
1151	003464	037110	DT4	
1152	003466	057663	ITE300: EM300	;TRAP VECTOR DECODE FAILED
1153	003470	037063	DH4	
1154	003472	037110	DT4	
1155	003474	057752	ITE301: EM301	;RTT DID NOT DISABLE T BIT
1156	003476	037063	DH4	
1157	003500	037110	DT4	
1158	003502	060037	ITE302: EM302	;PIR 1 DID NOT DISABLE ON BR4
1159	003504	037063	DH4	
1160	003506	037110	DT4	
1161	003510	060104	ITE303: EM303	;PIR 1 CAME THRU ON YELLOW ZONE
1162	003512	037063	DH4	
1163	003514	037110	DT4	
1164	003516	060235	ITE304: EM304	;PIR 2 CAME THRU ON YELLOW ZONE
1165	003520	037063	DH4	
1166	003522	037110	DT4	

1167	003524	060322	ITE305: EM305	;PIR 3 CAME THRU ON YELLOW ZONE
1168	003526	037063	DH4	
1169	003530	037110	DT4	
1170	003532	060410	ITE306: EM306	;BR4 CAME THRU ON PIR 5
1171	003534	037063	DH4	
1172	003536	037110	DT4	
1173	003540	060532	ITE307: EM307	;BR4 CAME THRU ON PIR 5
1174	003542	037063	DH4	
1175	003544	037110	DT4	
1176	003546	060605	ITE310: EM310	;BR4 CAME THRU ON BR 5
1177	003550	037063	DH4	
1178	003552	037110	DT4	
1179	003554	060657	ITE311: EM311	;BR4 CAME IN ON PIR 6
1180	003556	037063	DH4	
1181	003560	037110	DT4	
1182	003562	060730	ITE312: EM312	;BR4 CAME IN ON PIR 7
1183	003564	037063	DH4	
1184	003566	037110	DT4	
1185	003570	061054	ITE313: EM313	;PIR 4 CAME IN ON BR5
1186	003572	037063	DH4	
1187	003574	037110	DT4	
1188	003576	061124	ITE314: EM314	;PIR 4 CAME IN ON BR6
1189	003600	037063	DH4	
1190	003602	037110	DT4	
1191	003604	061235	ITE315: EM315	;PIR 4 CAME THRU ON SL YELLOW
1192	003606	037063	DH4	
1193	003610	037110	DT4	
1194	003612	061357	ITE316: EM316	;BR5 CAME IN ON PIR 5
1195	003614	037063	DH4	
1196	003616	037110	DT4	
1197	003620	061472	ITE317: EM317	;BR5 CAME IN ON PIR 6
1198	003622	037063	DH4	
1199	003624	037110	DT4	
1200	003626	061512	ITE320: EM320	;BR5 CAME IN ON PIR 7
1201	003630	037063	DH4	
1202	003632	037110	DT4	
1203	003634	061630	ITE321: EM321	;PIR 5 CAME IN ON BR6
1204	003636	037063	DH4	
1205	003640	037110	DT4	
1206	003642	061700	ITE322: EM322	;PIR 5 CAME IN ON SL YELLOW
1207	003644	037063	DH4	
1208	003646	037110	DT4	
1209	003650	061755	ITE323: EM323	;BR6 CAME IN ON PIR 6
1210	003652	037063	DH4	
1211	003654	037110	DT4	
1212	003656	062065	ITE324: EM324	;BR6 CAME IN ON PIR 7
1213	003660	037063	DH4	
1214	003662	037110	DT4	
1215	003664	062146	ITE325: EM325	;PIR 6 CAME IN ON SL YELLOW
1216	003666	037063	DH4	
1217	003670	037110	DT4	
1218	003672	062223	ITE326: EM326	;PIR 7 CAME IN ON SL YELLOW
1219	003674	037063	DH4	
1220	003676	037110	DT4	
1221	003700	062304	ITE327: EM327	;PSW BIT 11 DID NOT SET
1222	003702	037063	DH4	

1223	003704	037110		
1224	003706	062447	ITE330:	EM330 ;R12 CLEARED R2
1225	003710	037063		DH4
1226	003712	037110		DT4
1227	003714	062560	ITE331:	EM331 ;R2 SRC WAS AFFECTED BY CLR R12
1228	003716	037063		DH4
1229	003720	037110		DT4
1230	003722	062673	ITE332:	EM332 ;R2 DST WAS AFFECTED BY (R12)+
1231	003724	037063		DH4
1232	003726	037110		DT4
1233	003730	062754	ITE333:	EM333 ;R2 SRC WAS AFFECTED BY (R12)+
1234	003732	037063		DH4
1235	003734	037110		DT4
1236	003736	063007	ITE334:	EM334 ;R15 DST AFTER UPAD 2
1237	003740	037063		DH4
1238	003742	037110		DT4
1239	003744	063042	ITE335:	EM335 ;R15 SRC DOES NOT SELECT ON UPAD 2
1240	003746	037063		DH4
1241	003750	037110		DT4
1242	003752	063077	ITE336:	EM336 ;PDRD PS11 DOES NOT GET TO GRAC
1243	003754	037063		DH4
1244	003756	037110		DT4
1245	003760	063146	ITE337:	EM337 ;BAD BITS IN GPR SET 1 SRC
1246	003762	063213		DH337 ;ERRORPC PATTERN TESTNUMBER
1247	003764	063250		DT337 ;\$ERRPC,\$TMP1,\$STSTNM
1248	003766	063260	ITE340:	EM340 ;BAD BITS IN GPR SET 1 DST
1249	003770	063213		DH337
1250	003772	063250		DT337
1251	003774	063332	ITE341:	EM341 ;GRAB DST SET 1 DOES NOT GO LOW ON R14
1252	003776	037063		DH4
1253	004000	037110		DT4
1254	004002	063377	ITE342:	EM342 ;GRAB SRC SET 1 DOES NOT GO LOW ON R14
1255	004004	037063		DH4
1256	004006	037110		DT4
1257	004010	063444	ITE343:	EM343 ;50000 PATTERN FAILED IN PSW
1258	004012	041456		DH42
1259	004014	041534		DT42
1260	004016	063500	ITE344:	EM344 ;164000 PATTERN FAILED IN PSW
1261	004020	041456		DH42
1262	004022	041534		DT42
1263	004024	063535	IRE345:	EM345 ;PSW HIGH BYTE DID NOT CLEAR
1264	004026	041456		DH42
1265	004030	041534		DT42
1266	004032	063570	ITE346:	EM346 ;SUPER SP DOES NOT SELECT ON UPAD 5
1267	004034	037063		DH4
1268	004036	037110		DT4
1269	004040	063650	ITE347:	EM347 ;SUPER SP DOES NOT SELECT ON UPAD 0
1270	004042	037063		DH4
1271	004044	037110		DT4
1272	004046	063730	ITE350:	EM350 ;USER SP DOES NOT SELECT ON UPAD 5
1273	004050	037063		DH4
1274	004052	037110		DT4
1275	004054	064005	ITE351:	EM351 ;USER SP DOES NOT SELECT ON UPAD 0
1276	004056	037063		DH4
1277	004060	037110		DT4
1278	004062	064062	ITE352:	EM352 ;EITHER USER OR SUPER SP DST FAILED

1279	004064	063213	DH337	
1280	004066	063250	DT337	
1281	004070	064127	ITE353: EM353	;EITHER USER OR SUPER SP SRC FAILED
1282	004072	063213	DH337	
1283	004074	063250	DT337	
1284	004076	064174	ITE354: EM354	;KSP SRC CHANGED ON MTP
1285	004100	037063	DH4	
1286	004102	037110	DT4	
1287	004104	064226	ITE355: EM355	;KSP SRC & DST CHANGED ON MTP
1288	004106	037063	DH4	
1289	004110	037110	DT4	
1290	004112	064305	ITE356: EM356	;KSP DST CHANGED ON MTP
1291	004114	037063	DH4	
1292	004116	037110	DT4	
1293	004120	064340	ITE357: EM357	;SSP DID NOT LOAD PROPERLY ON MTPD
1294	004122	064431	DH357	;ERRORPC SSP TEST NUMBER
1295				; EXPECT ACTUAL
1296	004124	037526	DT14	
1297	004126	064503	ITE360: EM360	;USER SP DID NOT LOAD ON MTP
1298	004130	037063	DH4	
1299	004132	037110	DT4	
1300	004134	064562	ITE361: EM361	;BAD FIELD IN IR DECODE ROM
1301	004136	037063	DH4	
1302	004140	037110	DT4	
1303	004142	064632	ITE362: EM362	;SSP WAS READ AND USP WAS WRITTEN
1304	004144	037063	DH4	
1305	004146	037110	DT4	
1306	004150	064667	ITE363: EM363	;SSP WAS READ AND WRITTEN
1307	004152	037063	DH4	
1308	004154	037110	DT4	
1309	004156	064737	ITE364: EM364	;CAN'T DETERMINE WHAT HAPPENED
1310	004160	037063	DH4	;SSP WAS READ BUT THE WRITE FAILED
1311	004162	037110	DT4	
1312	004164	065021	ITE365: EM365	;USP WAS READ BUT SSP WAS WRITTEN
1313	004166	037063	DH4	
1314	004170	037110	DT4	
1315	004172	065056	ITE366: EM366	;USP WAS READ BUT REG 7 WAS WRITTEN
1316	004174	037063	DH4	
1317	004176	037110	DT4	
1318	004200	065112	ITE367: EM367	;SPL WORKED IN SUPERVISOR MODE
1319	004202	041456	DH42	
1320	004204	041534	DT42	
1321	004206	065226	ITE370: EM370	;BIT FAILED TO SET IN PSW
1322	004210	041456	DH42	
1323	004212	041534	DT42	
1324	004214	065353	ITE371: EM371	;BIT FAILED TO CLEAR IN PSW
1325	004216	041456	DH42	
1326	004220	041534	DT42	
1327	004222	065460	ITE372: EM372	;BITS <13:11> DID NOT PRESET
1328	004224	041456	DH42	
1329	004226	041534	DT42	
1330	004230	065610	ITE373: EM373	;IOT DID NOT CHANGE PSW CORRECTLY
1331	004232	041456	DH42	
1332	004234	041534	DT42	
1333	004236	065722	ITE374: EM374	;PREVIOUS MODE BITS DID NOT CLEAR
1334	004240	041456	DH42	

1335	004242	041534		DT42	
1336	004244	066012	ITE375:	EM375	;NO KT ABORT
1337	004246	037063		DH4	
1338	004250	037110		DT4	
1339	004252	066165	ITE376:	EM376	;PS<08> FAILED TO SET
1340	004254	037063		DH4	
1341	004256	037110		DT4	
1342	004260	000000	ITE377:	0	
1343	004262	000000		0	
1344	004264	000000		0	
1345	004266	066215	ITE400:	EM400	;KT ABORT TRAPPED TO 4
1346	004270	037063		DH4	
1347	004272	037110		DT4	
1348	004274	066316	ITE401:	EM401	;KT ABORT TRAPPED TO 10
1349	004276	037063		DH4	
1350	004300	037110		DT4	
1351	004302	066363	ITE402:	EM402	;KT ABORT TRAPPED TO 240
1352	004304	037063		DH4	
1353	004306	037110		DT4	
1354	004310	066425	ITE403:	EM403	;KT TRAP DID NOT WORK
1355	004312	037063		DH4	
1356	004314	037110		DT4	
1357	004316	000000	ITE404:	0	;DELETED
1358	004320	037063		DH4	
1359	004322	037110		DT4	
1360	004324	066531	ITE405:	EM405	;NO KT TRAP ON SOURCE OPERAND
1361	004326	037063		DH4	
1362	004330	037110		DT4	
1363	004332	066647	ITE406:	EM406	;NO ABORT ON NEXM
1364	004334	037063		DH4	
1365	004336	037110		DT4	
1366	004340	000000	ITE407:	0	;DELETED
1367	004342	037063		DH4	
1368	004344	037110		DT4	
1369	004346	066777	ITE410:	EM410	;NEXM BIT DID NOT SET IN CPUERR REG
1370	004350	054714		DH255	
1371	004352	041414		DT41	
1372	004354	067102	ITE411:	EM411	;NEXM BIT DID NOT CLEAR IN CPUERR REG
1373	004356	037063		DH4	
1374	004360	037110		DT4	
1375	004362	067151	ITE412:	EM412	;KT ABORT ON NEXM (KB11-B/C)
1376	004364	037063		DH4	
1377	004366	037110		DT4	
1378	004370	067226	ITE413:	EM413	;KT ABORT ON SL RED
1379	004372	037063		DH4	
1380	004374	037110		DT4	
1381	004376	067301	ITE414:	EM414	;KT ABORT ON ODD ADDRESS
1382	004400	037063		DH4	
1383	004402	037110		DT4	
1384	004404	067362	ITE415:	EM415	;KT ABORT FAILED TO OVER-RIDE NEXM (KB11-E/EM)
1385	004406	037063		DH4	
1386	004410	037110		DT4	
1387	004412	067445	ITE416:	EM416	;TMCE CACHE BEND DID NOT GO
1388	004414	037063		DH4	;HIGH ON KT ABORT
1389	004416	037110		DT4	
1390	004420	067520	ITE417:	EM417	;ILLEGAL HALT DID NOT TRAP

1391	004422	037063	DH4	
1392	004424	037110	DT4	
1393	004426	067611	ITE420: EM420	;CPU ERROR REG BIT 5 DID NOT SET
1394	004430	054714	DH255	
1395	004432	041414	DT41	
1396	004434	067666	ITE421: EM4 1	;BEN 6 FAILED ON PS RESTORE
1397	004436	037063	DH4	
1398	004440	037110	DT4	
1399	004442	067757	ITE422: EM422	;NO PE ABORT
1400	004444	037063	DH4	
1401	004446	037110	DT4	
1402	004450	070167	ITE423: EM423	;PE ABORTED TO 4
1403	004452	037063	DH4	
1404	004454	037110	DT4	
1405	004456	070261	ITE424: EM424	;PE ABORTED TO 14
1406	004460	037063	DH4	
1407	004462	037110	DT4	
1408	004464	070340	ITE425: EM425	;PE ABORTED TO 104
1409	004466	037063	DH4	
1410	004470	037110	DT4	
1411	004472	070357	ITE426: EM426	;NO PE TRAP
1412	004474	037063	DH4	
1413	004476	037110	DT4	
1414	004500	070443	ITE427: EM427	;PE TRAP, TRAPPED TO
1415	004502	037063	DH4	;WRONG VECTOR
1416	004504	037110	DT4	
1417	004506	070547	ITE430: EM430	;PIR 6 CAME IN ON MEM MGT TRAP
1418	004510	037063	DH4	
1419	004512	037110	DT4	
1420	004514	070612	ITE431: EM431	;PIR 3 CAME IN ON MEM MGT TRAP
1421	004516	037063	DH4	
1422	004520	037110	DT4	
1423	004522	070656	ITE432: EM432	;YEL ZONE CAME IN ON PE TRAP
1424	004524	037063	DH4	
1425	004526	03 110	DT4	
1426	004530	070742	ITE433: EM433	;MEM MGT TRAP CAME IN ON PE
1427	004532	037063	DH4	
1428	004534	037110	DT4	
1429	004536	000000	ITE434: 0	;DELETED
1430	004540	037063	DH4	
1431	004542	037110	DT4	
1432	004544	071010	ITE435: EM435	;TMCC PRIORITY CLEAR DID NOT WORK
1433	004546	037063	DH4	
1434	004550	037110	DT4	
1435	004552	071130	ITE436: EM436	;UNIBUS PE ABORT DID NOT HAPPEN
1436	004554	037063	DH4	
1437	004556	037110	DT4	
1438	004560	071173	ITE437: EM437	;UNIBUS PE TRAPPED TO 0
1439	004562	037063	DH4	
1440	004564	037110	DT4	

1441	004566	071243	ITE440:	EM440		;WAIT INSTRUCTION FAILED
1442	004570	037063		DH4		
1443	004572	037110		DT4		
1444	004574	071316	ITE441:	EM441		;I BIT INTERRUPTED WAIT
1445	004576	037063		DH4		
1446	004600	037110		DT4		
1447	004602	071367	ITE442:	EM442		;PIRQ DID NOT INTERRUPT WAIT
1448	004604	037063		DH4		
1449	004606	037110		DT4		
1450	004610	071444	ITE443:	EM443		;STATE S13.00 FAILED
1451	004612	037063		DH4		
1452	004614	037110		DT4		
1453	004616	071462	ITE444:	EM444		;STATE S45.00 FAILED
1454	004620	037063		DH4		
1455	004622	037110		DT4		
1456	004624	071500	ITE445:	EM445		;STATE MTP.10 FAILED
1457	004626	037063		DH4		
1458	004630	037110		DT4		
1459	004632	071516	ITE446:	EM446		;STATE NEG.00 FAILED
1460	004634	037063		DH4		
1461	004636	037110		DT4		
1462	004640	071534	ITE447:	EM447		;STATE D45.00 FAILED
1463	004642	037063		DH4		
1464	004644	037110		DT4		
1465	004646	071552	ITE450:	EM450		;STATE D45.90 FAILED
1466	004650	037063		DH4		
1467	004652	037110		DT4		
1468	004654	071570	ITE451:	EM451		;STATE D45.00 FAILED
1469	004656	037063		DH4		
1470	004660	037110		DT4		
1471	004662	071606	ITE452:	EM452		;STATE D45.01 FAILED
1472	004664	037063		DH4		
1473	004666	037110		DT4		
1474	004670	071624	ITE453:	EM453		;RESERVED INSTRUCTION TRAP FAILED
1475	004672	037063		DH4		
1476	004674	037110		DT4		
1477	004676	071672	ITE454:	EM454		;TMCB PIRQ DOES NOT GO LOW
1478	004700	037063		DH4		
1479	004702	037110		DT4		
1480	004704	071757	ITE455:	EM455		;BEN06 FAILED
1481	004706	037063		DH4		
1482	004710	037110		DT4		
1483	004712	072075	ITE456:	EM456		;ODD ADDRESS FAILED ON DATI & DATO
1484	004714	037063		DH4		
1485	004716	037110		DT4		
1486	004720	072151	ITE457:	EM457		;PSW BIT 11 DOES NOT CLEAR
1487	004722	037063		DH4		
1488	004724	037110		DT4		
1489	004726	072202	ITE460:	EM460		;PSW CHANGED ON RESET
1490	004730	041456		DH42		
1491	004732	041534		DT42		
1492	004734	072246	ITE461:	EM461		;PSW CHANGES ON RESET IN SUPER MODE
1493	004736	041456		DH42		
1494	004740	041534		DT42		
1495	004742	012737	000014 177746 START:	MOV	#14,@CONTRL	;FORCE MISSES IN CACHE
1496	004750	005037	170200	CLR	@MAPLO	;SETUP MAP 0 AND 1 TO PHYSICAL CORE

```
1497 004754 005037 170202 CLR @MAPH0
1498 004760 012737 020000 170204 MOV #20000,@MAPL1
1499 004766 005037 170206 CLR @MAPH1
1500 004772 012737 000340 177776 MOV #340,@PS ;:LOCK OUT ALL INTERRUPTS
1501 005000 012706 001100 MOV #SCMTAG,R6 ;:FIRST LOCATION TO BE CLEARED
1502 005004 005026 CLR (R6)+ ;:CLEAR MEMORY LOCATION
1503 005006 022706 001136 CMP #STKS,R6 ;:DONE?
1504 005012 001374 BNE -6 ;:LOOP BACK IF NO
1505 005014 012706 001100 MOV #STACK,SP ;:SETUP THE STACK POINTER
1506 005020 012737 033270 000020 MOV #SCOPE,@IOTVEC ;:IOT VECTOR FOR SCOPE ROUTINE
1507 005026 012737 000340 000022 MOV #340,@IOTVEC+2 ;:LEVEL 7
1508 005034 012737 033536 000030 MOV #ERROR,@EMTVEC ;:EMT VECTOR FOR ERROR ROUTINE
1509 005042 012737 000340 000032 MOV #340,@EMTVEC+2 ;:LEVEL 7
1510 005050 012737 036234 000034 MOV #STRAP,@TRAPVEC ;:TRAP VECTOR FOR TRAP CALLS
1511 005056 012737 000340 000036 MOV2 ;:LEVEL 7
1520 005140 012767 000002 025472 MOV #RTI,$RTRN ;:SET $RTRN TO A RTI
1521 005146 012737 005174 000010 MOV #65$,@RESVEC ;:TRY TO DO A RTT
1522 005154 005046 CLR -(SP) ;:DUMMY PS
1523 005156 012746 005164 MOV #64$,-(SP) ;:AND PC
1524 005162 000006 RTT ;:TRY THE RTT
1525 005164 012767 000006 025446 64$: MOV #RTT,$RTRN ;:RTT IS LEGAL--SET $RTRN TO A RTT
1526 005172 000402 BR 66$
1527 005174 062706 000010 65$: ADD #10,SP ;:RTT ILLEGAL--CLEAN OFF THE STACK
1528 005200 012737 000012 000010 66$: MOV #RESVEC+2,@RESVEC ;:RESTORE TRAP CATCHER
1529 005206 005067 025434 CLR $TBIT ;:CLEAR 'T' BIT SWITCH
1530 005212 012767 005212 173666 MOV #,$LPADR ;:INITIALIZE THE LOOP ADDRESS FOR SCOPE
1531 005220 012767 005220 173662 MOV #,$LPERR ;:SETUP THE ERROR LOOP ADDRESS
1532
1533 ;:*****
1534 005226 005227 177777 INC #-1 ;:FIRST TIME?
1535 005232 001037 BNE 67$ ;:BRANCH IF NO
1536 005234 022737 032572 000042 CMP #SENDAD,@#42 ;:ACT-11?
1537 005242 001433 BEQ 67$ ;:BRANCH IF YES
1538 005244 104400 005252 TYPE ,68$ ;:TYPE ASCIZ STRING
1539 005250 000430 BR 67$ ;:GET OVER THE ASCIZ
1540 ;:68$: .ASCIZ <CRLF>'CEKBB-D PDP11/70-74MP CPU DIAGNOSTIC PART 2'<CRLF>
1541 005332 67$:
1542 005332 005227 177777 INC #-1 ;:FIRST TIME?
1543 005336 001000 BNE 100$ ;:BR IF NO
1544 005340 100$:
1545 ;:*****
1546 ;:SAVE THE MONITOR IF INITIAL LOAD
1547 005340 005737 000042 LOOP: TST @#42 ;:RUNNING UNDER XXDP CHAIN?
1548 005344 001003 BNE 3$ ;:BRANCH IF YES
1549 005346 005227 177777 INC #-1 ;:INITIAL LOAD?
1550 005352 001010 BNE 2$ ;:BRANCH IF NO
1551 005354 012700 002734 3$: MOV #^D1500,R0 ;:SETUP SOB COUNT
1552 005360 012701 073300 MOV #SUBTAB+2,R1 ;:GET END ADDRESS OF PROGRAM
```

1553 005364 012702 160000
1554 005370 014221
1555 005372 077002
1556 005374 012737 034644 000060
1557 005402 012737 000340 000062
1558 005410 005077 173524
1559 005414 152777 000100 173514
1560 005422 012737 032654 000004
1561 005430 012737 000340 000006
1562 005436 012737 033060 000114
1563 005444 012737 000340 000116
1564 005452 005037 177766
1565 005456 012737 177777 177744
1566 005464 005767 173410
1567 005470 001402
1568 005472 005037 177746
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1589
1590
1591
1592 005476 000004
1593 005500 012767 005660 173466
1594 005506 012767 005660 173554
1595 005514 012706 001100
1596 005520 005005
1597 005522 012737 005642 000010
1598 005530 012737 000113 177770
1599 005536 000235
1600 005540 013767 177776 173440
1601 005546 042767 177437 173432
1602 005554 022767 000240 173424
1603 005562 001401
1604 005564 005205
1605 005566 012737 000044 177770
1606 005574 000240
1607 005576 000232
1608 005600 013767 177776 173354

```
MOV #160000,R2 ;GET END ADDRESS OF MONITOR
1$: MOV -(R2),(R1)+ ;SAVE 1500 DECIMAL WORDS
SOB R0,1$
2$: MOV #QUIT,@TKVEC ;SETUP KEYBOARD VECTOR
MOV #PR7,@TKVEC+2 ;SETUP KEYBOARD PSW
CLR @STKB ;ENSURE BUFFER CLEAR
BISB #BIT6,@STKS ;SET INTERRUPT ENABLE BIT
MOV #CPUSPUR,@ERRVEC
MOV #PR7,@ERRVEC+2
MOV #CACHSPU,@CACHVEC
MOV #PR7,@CACHVEC+2
CLR @CPUERR ;ENSURE ERROR REGISTER CLEAR
MOV #-1,@MEMERR ;ENSURE MEMORY ERROR REG CLEAR
TST $PASS ;FIRST PASS?
BEQ TST1 ;BRANCH IF YES
CLR @CONTRL ;ENABLE CACHE

*****
*TEST 1 SPL
*
* IF FORK A FAILS EXECUTION WILL GO TO ONE OF 3 STATES:
* RSD.02,SVC.70, OR D30.00.
* RSD.02 WILL CAUSE A TRAP TO LOCATION 10.
* SVC.70 WILL HANG THE PROCESSOR IN THE PAUSE STATE. THIS
* WILL ONLY OCCUR IF RACF E17(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,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
MOV #TST2,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
MOV #STACK,SP ;INITIALIZE THE SP
CLR R5 ;INITIALIZE ERROR RECORD
MOV #1$,@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,$ERPSW ;DID PRIORITY LEVEL 5 GET SET?
BEQ 2$ ;BRANCH IF YES
INC R5 ;SET ERROR RECORD
2$: MOV #44,@177770 ;SETUP MICROPROCESSOR BREAK REG
NOP ;SYNC INSTRUCTION
SPL 2 ;TEST TO SEE IF BITS 5&7 CLEAR & BIT 6 SETS
MOV @PSW,$TMP0 ;SAVE PSW
```

```
1609 005606 042767 177437 173346 BIC #177437,$TMP0 ;MASK OFF THE PRIORITY
1610 005614 022767 000100 173340 CMP #PR2,$TMP0 ;DID BIT 6 SET & 5&7 CLEAR?
1611 005622 001404 BEQ 3$ ;BRANCH IF YES
1612 005624 005705 TST R5 ;DID SPL 5 FAIL?
1613 005626 001401 BEQ 4$ ;BRANCH IF NO
1614 005630 104001 ERROR 1 ;EITHER SPL DOES NOT LOAD PSW OR BITS STUCK
1615 005632 104002 4$: ERROR 2 ;PR5 OK BUT PR2 BAD
1616 005634 005705 3$: TST R5 ;DID SPL 5 FAIL?
1617 005636 001410 BEQ TST2 ;;BRANCH IF NO
1618 005640 104003 ERROR 3 ;PR2 OK BUT PR5 FAILED
1619 005642 012737 000012 000010 1$: MOV #12,@#RESVEC ;RESTORE RESERVEC VECTOR ADDRESS
1620 005650 005705 TST R5 ;DID R5 GET INCREMENTED?
1621 005652 001401 BEQ 5$ ;BRANCH OF NO
1622 005654 104004 ERROR 4 ;FORK A FAILED TO D30.00
1623 005656 104005 5$: ERROR 5 ;FORK A FAILED TO RSD.02
1624 .....
1625 *TEST 2 RESET
1626 *
1627 * IF FORK A FAILS EXECUTION WILL EITHER GO TO WAT.00 OR TRP.02.
1628 * THE LA30 PRINTER WILL BE STARTED SUCH THAT A FAILURE INTO
1629 * WAT.00 WILL RECOVER.
1630 *
1631 * IF TRP.01 IS ENTERED A TRAP SEQUENCE WILL EXECUTE
1632 * WITH A TRAP VECTOR OF 4.
1633 *
1634 * ROM FLOW-15,255,374
1635 .....
1636 005660 000004 TST2: SCOPE
1637 005662 012767 003674 173214 MOV #^D1980,$ICNT ;SET ITERATION COUNT
1638 005670 012767 006154 173276 MOV #TST3,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
1639 005676 012767 006154 173364 MOV #TST3,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
1640 005704 013767 177776 173256 MOV @#PSW,$TMP3 ;SAVE PSW
1641 005712 012767 005742 173166 MOV #4$,$LPADR ;SETUP LOOP ADR
1642 005720 012767 005742 173162 MOV #4$,$LPERR ;SETUP ERROR LOOP
1643 005726 012737 006144 000064 MOV #2$,@#TPVEC ;PUT ADDRESS OF 2$ IN PRINTER INTERRUPT VEC
1644 005734 012737 006134 000004 MOV #1$,@#ERRVEC ;PUT ADDRESS OF 1$ IN LOCATION 4
1645 005742 012706 001100 4$: MOV #STACK,SP ;INITIALIZE THE SP
1646 ;TO CATCH A FAILURE TO THE TRP.02 STATE
1647 005746 000230 SPL 0 ;SET CPU AT 0
1648 005750 012777 000015 173166 MOV #15,@#STPB ;SEND A CR TO THE PRINTER
1649 005756 105777 173160 7$: TSTB @#STPS ;WAIT FOR CR TO FINISH
1650 005762 100375 BPL 7$ ;INCASE TP DOUBLE BUFFERED
1651 005764 012777 000101 173152 MOV #101,@#STPB ;SEND AN 'A'
1652 005772 052777 000100 173142 BIS #BIT6,@#STPS ;SET THE INTERRUPT FLAG
1653 006000 012737 034157 177776 MOV #34157,@#PSW ;SET AS MANY BITS AS POSSIBLE IN PSW
1654 006006 000240 NOP ;SYNC POINT
1655 006010 000005 RESET ;EXECUTE INSTRUCTION UNDER TEST
1656 006012 013767 177776 173166 MOV @#PSW,$ERPSW ;SAVE PSW
1657 006020 017700 173116 MOV @#STPS,R0 ;GET THE PRINTER STATUS
1658 006024 042777 000100 173112 BIC #BIT6,@#STPB ;CLEAR INTERRUPT FLAG INCASE RESET FAILED
1659 006032 032700 000100 BIT #BIT6,R0 ;DID INTERRUPT FLAG CLEAR?
1660 006036 001401 BEQ 3$ ;BRANCH IF YES
1661 006040 104006 ERROR 6 ;RESET DID NOT WORK
1662 006042 105777 173074 3$: TSTB @#STPS ;WAIT FOR CHARACTER TO FINISH
1663 006046 100375 BPL :$
1664 006050 012737 032654 000004 MOV #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
```

```
1665 006056 012767 034157 173076      MOV      #34157,$TMP0      ;SAVE EXPECTED VALUE
1666 006064 032737 000020 177776      BIT      #BIT4,@#PSW      ;IS T BIT ON?
1667 006072 001407                BEQ      5$                ;BRANCH IF NO
1668 006074 012767 034177 173060      MOV      #34177,$TMP0      ;SAVE EXPECTED VALUE
1669 006102 022767 034177 173076      CMP      #34177,$ERPSW     ;DID PSW COME OUT OK?
1670 006110 000403                BR       6$                ;
1671 006112 022767 034157 173066      5$:     CMP      #34157,$ERPSW     ;DID PSW CHANGE?
1672 006120 001415                6$:
1673 006120 001415                BEQ      TST3              ;:BRANCH IF NO
1674 006122 012767 034157 173032      MOV      #34157,$TMP0      ;SAVE EXPECTED VALUE
1675 006130 104377                ERROR   377                ;PSW CHANGED ON RESET
1676 006132 000460                460
1677 006134 042777 000100 173000      1$:     BIC      #BIT6,@$TPS      ;CLEAR PRINTER INTERRUPT FLAG
1678 006142 104007                ERROR   7                  ;FORK A FAILED INTO TRP.02
1679 006144 042777 000100 172770      2$:     BIC      #BIT6,@$TPS      ;CLEAR PRINTER INTERRUPT FLAG
1680 006152 104010                ERROR   10                 ;FORK A FAILED TO WAT.00
1681
1682
1683
1684
1685
1686
1687
1688
1689
1690
1691
1692
1693
1694
1695
1696 006154 000004                TST3:  SCOPE
1697 006156 152777 000100 172752      BISB    #BIT6,@$TKS      ;RESTORE INTER FLAG AFTER RESET
1698 006164 012767 006252 172714      MOV     #9$,$LPADR       ;SETUP LOOP ADDRESS
1699 006172 012767 006252 172710      MOV     #9$,$LPERR       ;SETUP ERROR LOOP
1700 006200 013767 177776 172762      MOV     @#PSW,$TMP3      ;SAVE PSW
1701 006206 032737 000020 177776      BIT     #BIT4,@#PSW      ;IS T BIT ON?
1702 006214 001416                BEQ     9$                ;BRANCH IF NO
1703 006216 012746 000340                MOV     #PR7,-(SP)       ;PUT NEW PSW ON STACK
1704 006222 012746 006252                MOV     #9$,-(SP)       ;PUT RETURN ADDR ON STACK
1705 006226 000006                RTT
1706 006230 012767 006436 172736      MOV     #TST4,$ESCAPE    ;SAVE START ADDRESS OF NEXT TEST
1707 006236 012767 006436 173024      MOV     #TST4,NEXTTST    ;SAVE SIART ADDRESS OF NEXT TEST
1708 006244 012737 006406 000010      MOV     #2$,@#RESVEC     ;SETUP LOC 10 TO TRAP TO THIS ROUTINE
1709 006252 012706 001100      9$:     MOV     #STACK,SP        ;INITIALIZE STACK
1710 006256 012746 100000                MOV     #BIT15,-(SP)     ;SET SIGN BIT ON STACK
1711 006262 012766 100000 177776      MOV     #BIT15,-2(SP)    ;SET SIGN BIT AT LOCATION 1074
1712 006270 012705 006326                MOV     #1$,$R5         ;SETUP R5 FOR MARK INSTRUCTION
1713 006274 000240                NOP
1714 006276 006401                MARK   1                  ;EXECUTE INSTRUCTION UNDER TEST
1715 006300 022701 006300      8$:     CMP     #8$,$R1         ;DID MTP OCCUR?
1716 006304 001401                BEQ     3$                ;BRANCH IF NO
1717 006306 104011                5$:     ERROR  11             ;FORK A FAILED TO MTP
1718 006310 013701 001074      3$:     MOV     @#1074,$R1      ;DID MFP OCCUR?
1719 006314 100401                BMI     4$                ;BRANCH IF NO
1720 006316 104012                ERROR  12                 ;FORK A FAILED TO MFP
```

```

1721 006320 012706 001076      4$:  MOV    #1076,SP      ;INITIALIZE SP INCASE MARK CHANGED IT
1722 006324 104013              ERROR 13              ;PCB DID NOT LOAD FROM R5
1723 006326 020627 006304      1$:  CMP    SP,#5$-2      ;DID SP GET LOADED PROPERLY?
1724 006332 001410              BEQ    6$              ;BRANCH IF YES
1725 006334 010667 172614      MOV    SP,$REG0        ;SAVE SP FOR TYPEOUT
1726 006340 012767 006306 172610  MOV    #5$,$REG1        ;STORE ADDRESS OF 5$ FOR TYPEOUT
1727 006346 012706 001100      MOV    #STACK,SP      ;REINITIALIZE SP
1728 006352 104014              ERROR 14              ;MARK DID NOT LOAD SP PROPERLY
1729 006354 012706 001100      6$:  MOV    #STACK,SP      ;RESTORE THE SP
1730 006360 020527 006300      CMP    R5,#8$         ;DID R5 GET LOADED PROPERLY?
1731 006364 001424              BEQ    TST4           ;:BRANCH IF YES
1732 006366 010567 172562      MOV    R5,$REG0        ;SAVE R5 FOR TYPEOUT
1733 006372 013767 006304 172556  MOV    @#5$-2,$REG1    ;STORE ADDRESS OF 5$-2 FOR TYPEOUT
1734 006400 012706 001076      MOV    #1076,SP      ;RESTORE THE SP
1735 006404 104015              ERROR 15              ;R5 DID NOT LOAD PROPERLY
1736 006406 012737 000012 000010 2$:  MOV    #12,@#RESVEC    ;RESTORE RESERVED INSTR VECTOR
1737 006414 021627 006300      CMP    (SP),#8$       ;DID PCB GET INCREMENTED BY D67.01?
1738 006420 001003              BNE    7$             ;BRANCH IF NO
1739 006422 012706 001100      MOV    #STACK,SP      ;RESTORE THE SP
1740 006426 104016              ERROR 16              ;FORK A FAILED TO RSD.00
1741 006430 012706 001100      7$:  MOV    #STACK,SP      ;RESTORE THE SP
1742 006434 104017              ERROR 17              ;FORK A FAILED TO D67.01
1743
1744      ;*****
1745      ;*TEST 4      ASH*DMO
1746      ;*
1747      ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
1748      ;*
1749      ;*      IF GRAJ SC05 L DOES NOT GO LOW OR DOES NOT GET THRU TO
1750      ;*      RACK BRCA04 L A SHIFT RIGHT WILL OCCUR.
1751      ;*      IF THE SHIFT COUNTER DOES NOT SHIFT OR GRAJ SC 0 DOES NOT GO
1752      ;*      LOW THE PROCESSOR WILL HANG UP IN STATE ASH.41.
1753      ;*
1754      ;*      ROM FLOW-52,305,257,166 LEFT SHIFT
1755      ;*      52,305,277      RIGHT SHIFT
1756      ;*
1756 006436 000004      TST4: SCOPE
1757 006440 012767 007200 172622      MOV    #TST5,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
1758 006446 012767 006512 172432      MOV    #19$,$LPADR    ;SETUP LOOP ADR
1759 006454 012767 006512 172426      MOV    #19$,$LPERR    ;SETUP ERROR LOOP
1760 006462 032767 000020 172500      BIT    #BIT4,$TMP3    ;WAS T BIT ON?
1761 006470 001405              BEQ    18$            ;BRANCH IF NO
1762 006472 012746 000360      MOV    #360,-(SP)     ;PUT NEW PSW ON STACK
1763 006476 012746 006504      MOV    #18$,-(SP)     ;PUT RETURN ADR ON STACK
1764 006502 000006              RTT                   ;TURN T BIT ON
1765
1766      ;ARITHMETIC LEFT SHIFT
1767 006504 012737 007066 000010 18$:  MOV    #1$,@#RESVEC    ;SET UP FOR FORK A FAILURE
1768 006512 012706 001100      19$:  MOV    #STACK,SP      ;INITIALIZE THE SP
1769 006516 012700 177701      MOV    #177701,R0     ;PUT SHIFT COUNT IN R0 (+1)
1770 006522 012701 100000      MOV    #BIT15,R1      ;SET BIT 15 IN REGISTER TO BE SHIFTED
1771 006526 000240              NOP                   ;SYNC POINT
1772 006530 072100              ASH    R0,R1          ;EXECUTE INSTRUCTION UNDER TEST
1773 006532 103414              BCS    2$             ;BRANCH IF SHIFT WORKED
1774 006534 020127 140000      CMP    R1,#140000     ;DID SHIFT GO IN WRONG DIRECTION?
1775 006540 001002              BNE    3$             ;BRANCH IF NO
1776 006542 104020              ERROR 20              ;SHIFTED RIGHT INSTEAD OF LEFT

```

```
1777 006544 000425          BR      9$
1778 006546 020127 100000 3$:  CMP    R1,#BIT15      ;DID R1 SHIFT AT ALL?
1779 006552 001002          BNE    4$                ;BRANCH IF YES
1780 006554 104021          ERROR  21                ;R1 DID NOT SHIFT
1781 006556 000420          BR     9$
1782 006560 104022          4$:  ERROR  22                ;R1 SHIFTED BUT CARRY DID NOT SET
1783                                     ;SHIFT COUNTER COULD BE STUCK
1784 006562 000416          BR     9$
1785 006564 102003          2$:  BVC    10$            ;BRANCH IF V DID NOT SET
1786 006566 100402          BMI    10$            ;BRANCH IF N DID NOT CLEAR
1787 006570 001001          BNE    10$            ;BRANCH IF Z DID NOT SET
1788 006572 000412          BR     9$                ;CC'S OK
1789 006574 013767 177776 172404 10$:  MOV    @#PSW,$ERPSW      ;SAVE PSW
1790 006602 042767 177760 172376  BIC    #177760,$ERPSW   ;MASK OFF CC'S
1791 006610 012767 000007 172344  MOV    #7,$TMP0         ;PUT EXPECTED CC'S IN STORAGE
1792 006616 104031          ERROR  31                ;STATE ASH.40 DID NOT LOAD CC'S CORRECTLY
1793                                     ;*****
1794                                     ;ARITHMETIC RIGHT SHIFT TEST
1795 006620 012767 006626 172262 9$:  MOV    #64,$LPERR      ;SETUP ERROR LOOP
1796 006626 012701 100001 64$:  MOV    #100001,R1     ;SETUP R1 FOR RIGH SHIFT
1797 006632 012700 000077          MOV    #77,R0          ;PUT SHIFT COUNT IN R0 (-1)
1798 006636 005002          CLR    R2              ;ENSURE R2 CLEAR
1799 006640 000240          NOP                    ;SYNC POINT
1800 006642 072100          ASH   R0,R1            ;EXECUTE RIGHT SHIFT
1801 006644 005502          ADC   R2              ;SAVE CARRY IN R2
1802 006646 020127 140000  CMP    R1,#140000      ;DID R1 SHIFT IN RIGHT DIRECTION?
1803 006652 001417          BEQ   5$                ;BRANCH IF YES
1804 006654 005701          TST   R1              ;DID R1 SHIFT LEFT INSTEAD OF RIGHT?
1805 006656 001002          BNE   6$                ;BRANCH IF NO
1806 006660 104023          ERROR  23                ;R1 SHIFTED WRONG DIRECTION
1807 006662 000416          BR     8$
1808 006664 022701 040000 6$:  CMP    #40000,R1      ;DID SHIFT FAIL TO SIGN FILL?
1809 006670 001001          BNE   7$                ;BRANCH IF NO
1810 006672 104024          ERROR  24                ;SHIFT RIGHT DID NOT SIGN FILL
1811 006674 010167 172256 7$:  MOV    R1,$REG1        ;SAVE R1 FOR TYPE OUT
1812 006700 012767 140000 172256  MOV    #140000,$TMP1   ;PUT EXPECTED VALUE IN STORAGE
1813 006706 104025          ERROR  25                ;R1 SHIFTED, PUT DON'T KNOW WHERE
1814 006710 000403          BR     8$
1815 006712 005702          5$:  TST   R2              ;DID CARRY GET SET ON SHIFT?
1816 006714 001001          BNE   8$                ;BRANCH IF YES
1817 006716 104026          ERROR  26                ;SHIFT OK BUT CARRY DID NOT LOAD
1818                                     ;*****
1819                                     ;CONDITION CODE LOAD TEST(STATE ASH.30)
1820 006720 012767 006726 172162 8$:  MOV    #63,$LPERR      ;SETUP ERROR LOOP
1821 006726 012701 100000 63$:  MOV    #BIT15,R1      ;SETUP R0 TO TEST CC'S IN STATE ASH.30
1822                                     ;R0 HAS SHIFT COUNT (-1)
1823 006732 000250          CLN                    ;SET UP CC'S TO
1824 006734 000266          +SEV!SEZ              ;COMPLIMENT OF EXPECTED RESULT
1825                                     ;AND OSCILLOSCOPE SYNC POINT
1826 006736 072100          ASH   R0,R1            ;EXECUTE THE SHIFT
1827 006740 013767 177776 172240  MOV    @#PSW,$ERPSW      ;SAVE PSW
1828 006746 042767 177760 172232  BIC    #177760,$ERPSW   ;MASK OFF THE CC'S
1829 006754 022767 000010 172224  CMP    #10,$ERPSW      ;DID CC'S COME OUT OK?
1830 006762 001404          BEQ   12$              ;BRANCH IF YES
1831 006764 012767 000010 172170  MOV    #10,$TMP0        ;PUT EXPECTED VALUE IN STORAGE
1832 006772 104072          EPROR  72              ;STATE ASH.30 DID NOT LOAD CC'S CORRECTLY
```



```
1833  
1834  
1835 006774 012767 007002 172106  
1836 007002 012701 100000  
1837 007006 005000  
1838 007010 000263  
1839 007012 000250  
1840  
1841 007014 072100  
1842 007016 013767 177776 172162  
1843 007024 042767 177760 172154  
1844 007032 022767 000010 172146  
1845 007040 001416  
1846 007042 022701 100000  
1847 007046 001005  
1848 007050 012767 000010 172104  
1849 007056 104027  
1850 007060 000406  
1851 007062 104030  
1852 007064 000404  
1853 007066 012737 000012 000010  
1854 007074 104032  
1855  
1856  
1857 007076 012767 007104 172004  
1858 007104 012700 000002  
1859  
1860 007110 012701 060000  
1861 007114 000274  
1862 007116 000243  
1863  
1864 007120 072100  
1865 007122 013767 177776 172056  
1866 007130 020127 100000  
1867 007134 001406  
1868 007136 010167 172014  
1869 007142 012767 100000 172014  
1870 007150 104073  
1871 007152 042767 177760 172026  
1872 007160 022767 000013 172020  
1873 007166 001404  
1874 007170 012767 000013 171764  
1875 007176 104074  
1876  
1877  
1878  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888 007200 000004
```

```
*****  
:SHIFT COUNT=0  
12$: MOV #62,$LPERR ;SETUP ERROR LOOP  
62$: MOV #BIT15,R1 ;SET SIGN BIT IN REGISTER TO BE SHIFTED  
CLR R0 ;SET SHIFT COUNT TO ZERO  
+SEC!SEV ;SETUP CC'S TO COMPLIMENT  
CLN ;OF EXPECTED VALUE  
AND OSCILLOSCOPE SYNC POINT  
ASH R0,R1 ;EXECUTE INSTRUCTION  
MOV @#PSW,$ERPSW ;SAVE PSW  
BIC #177760,$ERPSW ;MASK OFF THE CC'S  
CMP #10,$ERPSW ;DID CC'S COME OUT OK?  
BEQ 16$ ;BRANCH IF YES  
13$: CMP #BIT15,R1 ;DID R1 SHIFT?  
BNE 15$ ;BRANCH IF YES  
MOV #10,$TMP0 ;SAVE EXPECTED VALUE  
ERROR 27 ;STATE ASH.20 DID NOT LOAD CC'S CORRECTLY  
BR 16$  
15$: ERROR 30 ;R1 SHIFTED WHEN NOT SUPPOSE TO  
BR 16$  
1$: MOV #12,@#RESVEC ;RESTORE RESERVED VECTOR  
ERROR 32 ;FORK A FAILED  
*****  
:CONDITION CODE TEST OF STATE ASH.41  
16$: MOV #61,$LPERR ;SETUP ERROR LOOP  
61$: MOV #2,R0 ;PUT SHIFT COUNT IN R0  
TO TEST STATE ASH.41  
MOV #60000,R1 ;SETUP R1 FOR SHIFT LEFT  
+SEN!SEZ ;SETUP CC'S TO COMPLEMENT  
+CLV!CLC ;OF EXPECTED VALUE  
AND OSCILLOSCOPE SYNC POINT  
ASH R0,R1 ;EXECUTE INSTRUCTION UNDER TEST  
MOV @#PSW,$ERPSW ;SAVE PSW  
CMP R1,#BIT15 ;DID R1 SHIFT CORRECTLY?  
BEQ 17$ ;BRANCH IF YES  
MOV R1,$REG1 ;SAVE R1 FOR TYPEOUT  
MOV #BIT15,$TMP1 ;STORE EXPECTED VALUE  
ERROR 73 ;STATE ASH.41 FAILED  
17$: BIC #177760,$ERPSW ;MASK OFF CC'S  
CMP #13,$ERPSW ;DID CC'S LOAD CORRECTLY?  
BEQ TST5 ;BRANCH IF YES  
MOV #13,$TMP0 ;STORE EXPECTED VALUE  
ERROR 74 ;BAD CC'S DUE TO ASH.41  
*****  
:TEST 5 ASH*DM1  
: IF FORK A FAILS EXECUTION WILL GO TO RSD.00.  
: IF FORK B FAILS EXECUTION WILL EITHER GO TO RSD.00 OR  
: MUL.00.  
: MUL.00 WILL ONLY BE ENTERED IF EITHER B FORK MUX INPUT B2  
: DOES NOT GO HIGH OR THE MUX IS BAD.  
: ROM FLOW-1,175,62,52,305,257  
:*****  
TST5: SCOPE
```

```
1889 007202 012767 007314 171764 MOV #TST6,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
1890 007210 012767 007314 172052 MOV #TST6,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
1891 007216 012706 001100 MOV #STACK,SP ;INITIALIZE THE STACK
1892 007222 012737 007262 000010 MOV #1$,@RESVEC ;SETUP RESERVED VECTOR
1893 007230 012700 001164 MOV #STMP1,R0 ;PUT ADDRESS OF SHIFT COUNT IN R0
1894 007234 012710 000001 MOV #1,(R0) ;SET SHIFT COUNT TO ONE
1895 007240 012701 100000 MOV #BIT15,R1 ;SETUP REGISTER TO BE SHIFTED
1896 007244 000240 NOP ;OSCILLOSCOPE SYNC POINT
1897 007246 072110 ASH (R0),R1 ;EXECUTE INSTRUCTION UNDER TEST
1898 007250 103421 BCS TST6 ;:TEST OK, GO TO NEXT TEST
1899 007252 005701 TST R1 ;DID MULTIPLY OCCUR?
1900 007254 100401 BMI 2$ ;BRANCH IF YES
1901 007256 104033 ERROR 33 ;STATE ASH.00 FAILED
1902 007260 104034 2$: ERROR 34 ;FORK B FAILED INTO MUL.00
1903 007262 012737 007276 000010 1$: MOV #3$,@RESVEC ;SETUP RESVEC
1904 007270 005000 CLR R0 ;PUT EVEN ADDRESS IN R0
1905 007272 000240 NOP ;OSCILLOSCOPE SYNC POINT
1906 007274 072120 ASH (R0)+,R1 ;EXECUTE DM2
1907 007276 012737 000012 000010 3$: MOV #12,@RESVEC ;RESTORE RESERVED VECTOR
1908 007304 005700 TST R0 ;DID R0 AUTO INCREMENT?
1909 007306 001401 BEQ 4$ ;BRANCH IF NO
1910 007310 104035 ERROR 35 ;FORK B FAILED
1911 007312 104036 4$: ERROR 36 ;FORK A FAILED
1912 *****
1913 *TEST 6 ASH*DM2
1914 *
1915 * IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
1916 *
1917 * ALL OTHER LOGIC HAS BEEN TESTED
1918 *
1919 * ROM FLOW-2,175,62,52,305
1920 *****
1921 TST6: SCOPE
1922 007314 000004 MOV #TST7,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
1923 007316 012767 007374 171650 MOV #TST7,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
1924 007324 012767 007374 171736 MOV #STACK,SP ;INITIALIZE THE SP
1925 007332 012706 001100 MOV #1$,@RESVEC ;PUT ADDRESS OF 1$ IN RESERVED VECTOR
1926 007336 012737 007364 000010 MOV #STMP1,R0 ;PUT ADDRESS OF SHIFT COUNT IN R0
1927 007344 012700 001164 CLR (R0) ;PUT SHIFT COUNT OF ZERO IN $TMP1
1928 007350 005010 ASH (R0)+,R1 ;EXECUTE INSTRUCTION UNDER TEST
1929 007352 072120 MOV #12,@RESVEC ;RESTORE RESVEC
1930 007354 012737 000012 000010 BR TST7 ;GO TO NEXT TEST
1931 007362 000404 1$: MOV #12,@RESVEC ;RESTORE RESERVED VECTOR
1932 007364 012737 000012 000010 ERROR 37 ;FORK A FAILED
1933 *****
1934 *TEST 7 ASH*DM4
1935 *
1936 * IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
1937 *
1938 * ALL OTHER LOGIC HAS BEEN TESTED.
1939 *
1940 * ROM FLOW-4,122,177,62,52,305
1941 *****
1942 TST7: SCOPE
1943 007374 000004 MOV #TST10,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
1944 007376 012767 007460 171570 MOV #TST10,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
1945 007404 012767 007460 171650
```

```

1945 007412 012706 001100      MOV      #STACK,SP      ;INITIALIZE THE SP
1946 007416 012737 007450 000010  MOV      #1$,@PRESVEC   ;PUT ADDRESS OF 1$ IN RESERVED VECTOR
1947 007424 012700 001166      MOV      #STMP2,R0      ;PUT ADDRESS OF SHIFT COUNT+2 IN R0
1948 007430 005060 177776      CLR      -2(R0)         ;PUT SHIFT COUNT IN STMP1
1949 007434 000240      NOP                     ;OSCILLOSCOPE SYNC POINT
1950 007436 072140      ASH      -(R0),R1       ;EXECUTE INSTRUCTION UNDER TEST
1951 007440 012737 000012 000010  MOV      #12,@PRESVEC   ;RESTORE RESVEC
1952 007446 000404      BR      TST10           ;GO TO NEXT TEST
1953 007450 012737 000012 000010 1$: MOV      #12,@PRESVEC   ;RESTORE RESVEC
1954 007456 104040      ERROR   40             ;FORK A FAILED
1955
1956 .....
1957 *TEST 10      ASHC*DMO
1958
1959 *          NEITHER FORK A NOR BEN03 SHOULD FAIL.
1960
1961 *          IF THE INSTRUCTION FAILS, ONE OF THE ASC STATES IS BAD.
1962
1963 *          ONCE IT IS DETERMINED THAT THE INSTRUCTION WORKS,
1964 *          A BIT STUCK TEST IS PERFORMED ON THE SHIFT COUNTER.
1965
1966 *          ROM FLOW-53,306,267,227      RIGHT SHIFT
1967 *                               53,306,247,176,136 LEFT SHIFT
1968 *                               53,306,207      NO SHIFT
1969 .....
1969 007460 000004      TST10: SCOPE
1970 007462 012767 010302 171600  MOV      #TST11,NEXTTST ;SAVE ADDRESS OF NEXT TEST
1971
1972 *ARITHMETIC RIGHT SHIFT COMBINED
1973 007470 012700 100001      MOV      #100001,R0     ;SETUP R0 FOR RIGHT SHIFT
1974 007474 012701 000001      MOV      #BIT0,R1       ;SETUP R1 FOR RIGH SHIFT
1975 007500 012702 000077      MOV      #77,R2         ;PUT SHIFT COUNT (-1) IN R2
1976 007504 000262      SEV                     ;ENSURE V SET
1977
1978 *AND OSCILLOSCOPE SYNC POINT
1978 007506 073002      ASHC     R2,R0          ;EXECUTE INSTRUCTION UNDER TEST
1979 007510 013767 177776 171470  MOV      @PSW,$ERPSW    ;SAVE PSW
1980 007516 042767 177760 171462  BIC      #177760,$ERPSW ;MASK OFF THE CC'S
1981 007524 022767 000011 171454  CMP      #11,$ERPSW     ;DID CC'S LOAD PROPERLY?
1982 007532 001014      BNE     1$             ;BRANCH IF NO
1983 007534 005701      TST     R1             ;DID R1 GET SIGN BIT SET
1984 007536 100017      BPL     2$             ;BRANCH IF NO
1985 007540 022700 140000      CMP      #140000,R0     ;DID R0 SIGN FILL?
1986 007544 001427      BEQ     3$             ;BRANCH IF YES
1987 007546 012767 140000 171406  MOV      #140000,$TMP0   ;EXPECTED VALUE
1988 007554 010067 171374      MOV      R0,$REG0       ;SAVE R0 FOR TYPEOUT
1989 007560 104041      ERROR   41             ;R0 DID NOT SIGN FILL
1990 007562 000420      BR      3$
1991 007564 012767 000011 171370 1$: MOV      #11,$TMP0      ;EXPECTED VALUE
1992 007572 104042      ERROR   42             ;BAD CC'S
1993 007574 000413      BR      3$
1994 007576 012767 140000 171356 2$: MOV      #140000,$TMP0   ;GET EXPECTED VALUES
1995 007604 012767 100000 171352  MOV      #100000,$TMP1   ;FOR TYPEOUT
1996 007612 010067 171336      MOV      R0,$REG0       ;SAVE R0 & R1 FOR TYPEOUT
1997 007616 010167 171334      MOV      R1,$REG1
1998 007622 104043      ERROR   43             ;R0<0> DID NOT GO TO R1<15>
1999 .....
2000 *ARITHMETIC LEFT SHIFT

```

```
2001 007624 012767 007632 171256 3$: MOV #64$, $LPERR ;SETUP ERROR LOOP
2002 007632 012700 100000 64$: MOV #BIT15,R0 ;SETUP R0 FOR LEFT SHIFT
2003 007636 012701 100001 MOV #100001,R1 ;SETUP R1 FOR LEFT SHIFT
2004 007642 012702 000001 MOV #BIT0,R2 ;PUT SHIFT COUNT (+1) IN R2
2005 007646 000274 +SEZ!SEN ;ENSURE Z AND N SET
2006 ;AND OSCILLOSCOPE SYNC POINT
2007 007650 073002 ASHC R2,R0 ;EXECUTE INSTRUCTION UNDER TEST
2008 007652 013767 177776 171326 MOV @#PSW,$ERPSW ;SAVE PSW
2009 007660 042767 177760 171320 BIC #177760,$ERPSW ;MASK OFF THE CC'S
2010 007666 010067 171262 MOV R0,$REG0 ;SAVE REG0
2011 007672 010167 171260 MOV R1,$REG1 ;SAVE REG1
2012 007676 012767 000001 171256 MOV #1,$TMP0 ;GET EXPECTED VALUES
2013 007704 012767 000002 171252 MOV #2,$TMP1 ;OF REGISTERS
2014 007712 022767 000003 171266 CMP #3,$ERPSW ;ARE CC'S CORRECT?
2015 007720 001010 BNE 7$ ;BRANCH IF NO
2016 007722 022701 000002 CMP #2,R1 ;DID R1 SHIFT PROPERLY?
2017 007726 001012 BNE 11$ ;BRANCH IF NO
2018 007730 022700 000001 CMP #1,R0 ;DID R0 SHIFT PROPERLY?
2019 007734 001410 BEQ 12$ ;BRANCH IF YES
2020 007736 104044 ERROR 44 ;R0 DID NOT GET SHIFTED PROPERLY
2021 007740 000406 BR 12$
2022 007742 012767 000003 171212 7$: MOV #3,$TMP0 ;SAVE EXPECTED VALUE
2023 007750 104045 ERROR 45 ;BAD CC'S
2024 007752 000401 BR 12$
2025 007754 104046 11$: ERROR 46 ;R1 DID NOT SHIFT PROPERLY
2026 ;:*****
2027 ;:NO SHIFT
2028 007756 012767 007764 171124 12$: MOV #63$, $LPERR ;SETUP ERROR LOOP
2029 007764 012701 040000 63$: MOV #40000,R1 ;SETUP R1 FOR NO SHIFT
2030 007770 005002 CLR R2 ;PUT SHIFT COUNT (0) IN R2
2031 007772 012700 100000 MOV #BIT15,R0 ;SET SIGN BIT IN R0 & SET N
2032 007776 000277 SCC ;ENSURE ALL CC'S SET
2033 010000 000250 CLN ;ENSURE N CLEAR
2034 ;AND OSCILLOSCOPE SYNC POINT
2035 010002 073002 ASHC R2,R0 ;EXECUTE INSTRUCTION UNDER TEST
2036 010004 013767 177776 171174 MOV @#PSW,$ERPSW ;SAVE PSW
2037 010012 012767 000010 171142 MOV #10,$TMP0 ;SAVE EXPECTED VALUE
2038 010020 042767 177760 171160 BIC #177760,$ERPSW ;MASK OFF THE CC'S
2039 010026 022767 000010 171152 CMP #10,$ERPSW ;DID THE CC'S COME OUT CORRECT?
2040 010034 001401 BEQ 14$ ;BRANCH IF YES
2041 010036 104047 13$: ERROR 47 ;BAD CC'S ON NO SHIFT
2042 ;:*****
2043 ;:CHECK ROTATE
2044 010040 012767 010046 171042 14$: MOV #62$, $LPERR ;SETUP ERROR LOOP
2045 010046 012701 100000 62$: MOV #BIT15,R1 ;SETUP R1 FOR A ROTATE
2046 010052 010700 MOV PC,R0 ;PUT RANDOM NUMBER IN R0
2047 010054 010067 171102 MOV R0,$TMP0 ;SAVE R0
2048 010060 012702 177761 MOV #-17,R2 ;PUT SHIFT COUNT (-17) IN R2
2049 010064 012767 000001 171072 MOV #1,$TMP1 ;EXPECTED VALUE OF R1 IN $TMP1
2050 010072 000240 NOP ;OSCILLOSCOPE SYNC POINT
2051 010074 073102 ASHC R2,R1 ;EXECUTE INSTRUCTION UNDER TEST
2052 010076 022701 000001 CMP #1,R1 ;DID R1 ROTATE CORRECTLY?
2053 010102 001405 BEQ 15$ ;BRANCH IF YES
2054 010104 010067 171044 MOV R0,$REG0 ;SAVE R0 FOR TYPEOUT
2055 010110 010167 171042 MOV R1,$REG1 ;SAVE R1 FOR TYPEOUT
2056 010114 104050 ERROR 50 ;R1 DID NOT ROTATE PROPERLY
```

```
2057 :*****
2058 :THE FOLLOWING CODE CHECKS THE BITS IN THE SC
2059 010116 012767 010124 170764 15$: MOV #61$, $LPERR ;SETUP ERROR LOOP
2060 010124 012700 000040 61$: MOV #40, R0 ;SETUP R1 FOR RIGHT SHIFT
2061 010130 012702 000052 MOV #52, R2 ;PUT SHIFT COUNT (-26) IN R2
2062 010134 005001 CLR R1 ;ENSURE R0 CLEAR
2063 010136 005067 171020 CLR $TMP0 ;PUT EXPECTED VALUES OF
2064 010142 005067 171016 CLR $TMP1 ;R0 & R1 IN STORAGE
2065 010146 012767 000001 171012 MOV #1, $TMP2 ;C BIT EXPECTED
2066 010154 000240 NOP ;OSCILLOSCOPE SYNC POINT
2067 010156 073002 ASHC R2, R0 ;EXECUTE SHIFT
2068 010160 013767 177776 171020 MOV @RPSW, $ERPSW ;SAVE PSW
2069 010166 103410 BCS 16$ ;BRANCH IF CORRECT SHIFT
2070 010170 042767 177776 171010 BIC #177776, $ERPSW ;MASK OFF C BIT
2071 010176 010067 170752 MOV R0, $REG0
2072 010202 010167 170750 MOV R1, $REG1
2073 010206 104051 ERROR 51 ;STUCK BITS IN SC
2074 010210 012767 010216 170672 16$: MOV #60$, $LPERR ;SETUP ERROR LOOP
2075 010216 012701 004000 60$: MOV #4000, R1 ;SET UP R1 FOR LEFT SHIFT
2076 010222 005000 CLR R0 ;ENSURE R0 CLEAR
2077 010224 012702 000025 MOV #25, R2 ;PUT SHIFT COUNT (+25) IN R2
2078 010230 000240 NOP ;OSCILLOSCOPE SYNC POINT
2079 010232 073002 ASHC R2, R0 ;EXECUTE SHIFT
2080 010234 013767 177776 170744 MOV @RPSW, $ERPSW
2081 010242 042767 177776 170736 BIC #177776, $ERPSW ;MASK OFF C BIT
2082 010250 103414 BCS TST11 ;GO TO NEXT TEST IF CORRECT SHIFT
2083 010252 012767 000000 170674 MOV #R0, $REG0 ;SAVE R0 FOR TYPEOUT
2084 010260 012767 000001 170670 MOV #R1, $REG1 ;SAVE R1 FOR TYPEOUT
2085 010266 005067 170670 CLR $TMP0 ;SAVE EXPECTED
2086 010272 012767 000001 170664 MOV #1, $TMP1 ;VALUES
2087 010300 104052 ERROR 52 ;STUCK BITS IN SC
2088 :*****
2089 :*TEST 11 ASHC*DM1
2090 :
2091 : THE ONLY POSSIBLE FAILURES ARE FORK B OR STATE ASC.00.
2092 :
2093 : IF FORK B FAILS EXECUTION WILL EITHER GO TO RSD.00 OR
2094 : ASH.00.
2095 : ASH.00 WOULD PERFORM AN ASH INSTRUCTION INSTEAD OF AN ASHC.
2096 :
2097 : ROM FLOW-1,175,63,53,306,267,227
2098 :*****
2099 010302 000004 TST11: SCOPE
2100 010304 012767 010442 170662 MOV #TST12, $ESCAPE ;SAVE START ADDRESS OF NEXT TEST
2101 010312 012767 010442 170750 MOV #TST12, NEXTTST ;SAVE START ADDRESS OF NEXT TEST
2102 010320 012706 001100 MOV #STACK, SP ;INITIALIZE THE SP
2103 010324 012737 010432 000010 MOV #1$, @RESVEC ;SETUP RESERVED VECTOR
2104 010332 012702 001166 MOV # $TMP2, R2 ;PUT ADDRESS OF SHIFT COUNT IN R2
2105 010336 012700 000001 MOV #1, R0 ;SETUP R0 FOR RIGHT SHIFT
2106 010342 005001 CLR R1 ;SETUP R1 FOR RIGHT SHIFT
2107 010344 012712 000077 MOV #77, (R2) ;PUT SHIFT COUNT (-1) IN $TMP2
2108 010350 005067 170606 CLR $TMP0 ;PUT EXPECTED VALUES OF
2109 010354 012767 100000 170602 MOV #BIT15, $TMP1 ;R0 & R1 IN STORAGE
2110 010362 000240 NOP ;OSCILLOSCOPE SYNC POINT
2111 010364 073012 ASHC (R2), R0 ;EXECUTE INSTRUCTION UNDER TEST
2112 010366 103012 BCC 2$ ;BRANCH IF FORK B DID NOT FAIL
```

```
2113 010370 013767 177776 170610      MOV    @#PSW,$ERPSW    ;SAVE PSW FOR TYPEOUT
2114 010376 005067 170560              CLR    $TMP0          ;SAVE EXPECTED VALUE
2115 010402 010067 170546              MOV    R0,$REG0
2116 010406 010167 170544              MOV    R1,$REG1
2117 010412 104053                      ERROR  53              ;FORK B FAILED TO ASH.00
2118 010414 005701                      2$:  TST    R1          ;DID R1 SIGN BIT SET?
2119 010416 100411                      BMI    TST12          ;:BRANCH IF YES
2120 010420 010067 170530              MOV    R0,$REG0      ;SAVE R0 & R1
2121 010424 010167 170526              MOV    R1,$REG1      ;FOR TYPEOUT
2122 010430 104054                      ERROR  54              ;STATE ASC.00 FAILED
2123 010432 012737 000012 000010 1$:  MOV    #12,@#RESVEC   ;RESTORE RESVFC
2124 010440 104055                      ERROR  55              ;FORK A FAILED TO RSD.00
2125                                     ;*****
2126                                     ;*TEST 12          MUL*DMO
2127                                     ;*
2128                                     ;*
2129                                     ;*   FORK A SHOULD NOT FAIL.
2130                                     ;*   THE FOLLOWING WOULD BE BEN11 FAILURES:
2131                                     ;*   IF EITHER GRAD DROO IS STUCK HIGH OR NOT GETTING THRU TO
2132                                     ;*   RACK E64(C1) OR RACK E64 IS BAD (INPUT C1 FAILED HIGH)
2133                                     ;*   THE MULTPLICAND WILL BE MULTITPLIED BY 177777.
2134                                     ;*   IF EITHER GRAD DROO IS STUCK LOW OR NOT GETTING THRU TO
2135                                     ;*   RACK E64(C1) OR RACK E64 IS BAD (INPUT C1 FAILED LOW)
2136                                     ;*   THE MULTIPLICAND WILL BE MULTIPLIED BY ZERO.
2137                                     ;*   IF GRAJ SC=0 IS NOT GETTING TO RACK E50(C1) AND RACK E50
2138                                     ;*   IS BAD (INPUT C1 FAILED LOW) THE MULTIPLICAND WILL ONLY
2139                                     ;*   BE MULTIPLIED BY BIT 0 OF THE MULTIPLIER.
2140                                     ;*   IF RACK E50(C1) FAILS HIGH THE PROCESSOR WILL HANG UP IN
2141                                     ;*   STATE MUL.20(266).
2142                                     ;*   IF THE INSTRUCTION FAILS TO MULTIPLY CORRECTLY AND ONE
2143                                     ;*   OF THE ABOVE CONDITIONS CANNOT BE DETERMINED THEN THE
2144                                     ;*   FAILURE COULD BE IN THE INSTRUCTION DECODE ROM.
2145                                     ;*
2146                                     ;*   IF THE CC'S COME UP BAD THEN THE FAILURE COULD BE IN
2147                                     ;*   STATES MUL.40,50, OR 60, OR IN THE CONDITION CODE ROM.
2148                                     ;*
2149                                     ;*   ROM FLOW-50,102,266/246,226/206,310
2150                                     ;******
2150 010442 000004      TS112: SCOPE
2151 010444 012767 011004 170616      MOV    #TST13,NEXTTST ;SAVE ADDRESS OF NEXT TEST
2152                                     ;
2153                                     ;MULTIPLY 3 TIMES 2
2154 010452 012700 000002      MOV    #2,R0          ;PUT MULTIPLICAND IN R0
2155 010456 012702 000003      MOV    #3,R2          ;PUT MULTIPLIER IN R2
2156 010462 005001          CLR    R1             ;ENSURE R1 CLEAR
2157 010464 005067 170472      CLR    $TMP0          ;PUT EXPECTED VALUE OF
2158 010470 012767 000006 170466      MOV    #6,$TMP1       ;R0 & R1 IN STORAGE
2159 010476 000277          SCC                  ;MULTIPLY SHOULD CLEAR ALL OF THEM
2160                                     ;AND OSCILLOSCOPE SYNC POINT
2161 010500 070002          MUL    R2,R0          ;EXECUTE INSTRUCTION UNDER TEST
2162 010502 013767 177776 170476      MOV    @#PSW,$ERPSW   ;SAVE PSW
2163 010510 010067 170440          MOV    R0,$REG0      ;SAVE R0 & R1
2164 010514 010167 170436          MOV    R1,$REG1      ;FOR TYPEOUT
2165 010520 020127 000006          CMP    R1,#6          ;DID R1 GET LOW PRODUCT
2166 010524 001002          BNE   1$             ;BRANCH IF NO
2167 010526 005700          TST   R0             ;DID R0 GET UPPER PRODUCT?
2168 010530 001423          BEQ   2$             ;BRANCH IF YES
```

```
2169 010532 020127 177776 1$: CMP R1,#177776 ;DID R0 GET MULTIPLIED BY 177777?
2170 010536 001002 BNE 3$ ;BRANCH IF NO
2171 010540 104056 ERROR 56 ;EITHER GRAD DROO STUCK H OR RACK E64 BAD
2172 010542 000430 BR 8$
2173 010544 005701 3$: TST R1 ;DID R1 STAY ZERO?
2174 010546 001401 BEQ 4$ ;BRANCH IF YES
2175 010550 000404 BR 5$ ;CONTINUE ERROR ANALYSIS
2176 010552 005700 4$: TST R0 ;DID R1 & R0 GO TO ZERO?
2177 010554 001002 BNE 5$ ;BRANCH IF NO
2178 010556 104057 ERROR 57 ;EITHER GRAD DROO STUCK L OR RACK E64 BAD
2179 010560 000421 BR 8$
2180 010562 020127 100000 5$: CMP R1,#BIT15 ;DID R0 ONLY GET MULTIPLIED BY BIT 0?
2181 010566 001002 BNE 6$ ;BRANCH IF NO
2182 010570 104060 ERROR 60 ;RACH E50(C1) FAILED LOW
2183 010572 000414 BR 8$
2184 010574 104061 6$: ERROR 61 ;CAN'T DETERMINE WHAT HAPPENED
2185 010576 000412 BR 8$
2186 010600 042767 177760 170400 2$: BIC #177760,$ERPSW ;MASK OFF THE CC'S
2187 010606 022767 000000 170372 CMP #0,$ERPSW
2188 010614 001403 BEQ 8$ ;BRANCH IF YES
2189 010616 005067 170340 7$: CLR $TMP0 ;PUT EXPECTED PSW IN $TMP0
2190 010622 104062 ERROR 62 ;BAD CC'S
2191
2192 ;*****
2193 010624 012767 010632 170256 8$: MOV #64,$LPERR ;THE FOLLOWING SECTION TESTS STATE MUL.50 (-1 TIMES 3)
2194 010632 012700 177777 64$: MOV #177777,R0 ;SETUP ERROR LOOP
2195 ; ;PUT -1 (MULTIPLICAND) IN R0
2196 010636 000277 SCC ;R2 HAS MULTIPLIER (+3)
2197 010640 000250 CLN ;
2198 ; ;CC'S=0111
2199 010642 070002 MUL R2,R0 ;AND OSCILLOSCOPE SYNC POINT
2200 010644 013767 177776 170334 MOV @#PSW,$ERPSW ;EXECUTE INSTRUCTION UNDER TEST
2201 010652 010067 170276 MOV R0,$REG0 ;SAVE PSW
2202 010656 010167 170274 MOV R1,$REG1 ;SAVE R0 & R1
2203 010662 012767 177777 170272 MOV #-1,$TMP0 ;FOR TYPEOUT
2204 010670 012767 177775 170266 MOV #-3,$TMP1 ;SAVE EXPECTED
2205 010676 020027 177777 CMP R0,#177777 ;VALUES
2206 010702 001402 BEQ 9$ ;DID R0 LOAD CORRECTLY?
2207 010704 104063 ERROR 63 ;BRANCH IF YES
2208 010706 000420 BR 12$ ;R0 DID NOT LOAD CORRECTLY
2209 010710 020127 177775 9$: CMP R1,#177775 ;DID R1 LOAD PROPERLY?
2210 010714 001402 BEQ 10$ ;BRANCH IF YES
2211 010716 104064 ERROR 64 ;R1 DID NOT LOAD CORRECTLY.
2212 010720 000413 BR 12$
2213 010722 042767 177760 170256 10$: BIC #177760,$ERPSW ;MASK OFF THE CC'S
2214 010730 022767 000010 170250 CMP #10,$ERPSW ;DID THE CC'S LOAD PROPERLY?
2215 010736 001404 BEQ 12$ ;BRANCH IF YES
2216 010740 012767 000010 170214 11$: MOV #10,$TMP0 ;PUT EXPECTED PSW IN $TMP0
2217 010746 104065 ERROR 65 ;BAD CC'S DUE TO STATE MUL.50
2218 ;*****
2219 ;THE FOLLOWING VERIFIES THAT C SETS IF MULTIPLICATION OVERFLOWS OR UNDER FLOWS
2220 010750 012767 010756 170132 12$: MOV #63,$LPERR ;SETUP ERROR LOOP
2221 010756 012700 077777 63$: MOV #77777,R0 ;PUT LARGEST POSITIVE NO. IN R0
2222 ; ;R2 STILL CONTAINS +2
2223 010762 000240 NOP ;OSCILLOSCOPE SYNC POINT
2224 010764 070002 MUL R2,R0 ;EXECUTE INSTRUCTION
```

```
2225 010766 103401          BCS 13$          ;BRANCH IF C SET
2226 010770 104066          ERROR 66         ;C DID NOT SET ON OVERFLOW
2227 010772 012700 100000 13$: MOV #BIT15,R0    ;PUT SMALLEST NEGATIVE NO IN R0
2228                                     ;R2 STILL CONTAINS +2
2229 010776 070002          MUL R2,R0        ;EXECUTE INSTRUCTION
2230 011000 103401          BCS TST13       ;:BRANCH IF C SET
2231 011002 104067          ERROR 67         ;C DID NOT SET ON UNDERFLOW
2232                                     ;*****
2233 :*TEST 13          MUL*DM1
2234 :*
2235 :*          FORK A SHOULD NOT FAIL.
2236 :*
2237 :*          FORK B WILL FAIL TO RSD.00 IF THE R(MUL:ASHC+MFP) FIELD OF
2238 :*          THE INSTRUCTION DECODE ROM IS BAD.
2239 :*          IF STATE MUL.00 FAILS THE RESULT WILL BE BAD.
2240 :*
2241 :*          ROM FLOW-1,175,60,102,266/246,226/206,310
2242 :*          ;*****
2243 011004 000004          TST13: SCOPE
2244 011006 012767 011120 170160 MOV #TST14,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
2245 011014 012767 011120 170246 MOV #TST14,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
2246 011022 012706 001100 MOV #STACK,SP      ;INITIALIZE THE SP
2247 011026 012737 011110 000010 MOV #1$,@RESVEC    ;SETUP RESVEC
2248 011034 012701 000002 MOV #2,R1          ;PUT MULTIPLICAND IN R1
2249 011040 012702 001166 MOV #TMP2,R2       ;PUT ADDRESS OF MULTIPLIER IN R2
2250 011044 012712 000002 MOV #2,(R2)        ;PUT MULTIPLIER (+2) IN TMP2
2251 011050 005000 CLR R0             ;ENSURE R0 CLEAR
2252 011052 005067 170104 CLR $TMP0          ;STORE EXPECTED VALUE
2253 011056 012767 000004 170100 MOV #4,$TMP1      ;OF R0 & R1
2254 011064 000240 NOP                ;OSCILLOSCOPE SYNC POINT
2255 011066 070112 MUL (R2),R1        ;EXECUTE INSTRUCTION UNDER TEST
2256 011070 020127 000004 CMP R1,#4           ;DID MULTIPLY WORK?
2257 011074 001411 BEQ TST14           ;:BRANCH IF YES
2258 011076 010067 170052 MOV R0,$REG0       ;SAVE R0 & R1
2259 011102 010167 170050 MOV R1,$REG1       ;FOR TYPEOUT
2260 011106 104070 ERROR 70                ;STATE MUL.00 FAILED
2261 011110 012737 000012 000010 1$: MOV #12,@RESVEC    ;RESTORE RESVEC
2262 011116 104071 ERROR 71                ;FORK B FAILED
2263                                     ;*****
2264 :*TEST 14          DIV*DM0
2265 :*
2266 :*          FORK A SHOULD NOT FAIL.
2267 :*
2268 :*          SECTION 1
2269 :*          THE FIRST SECTION HAS A ZERO DIVISOR. IF STATE DIV.00
2270 :*          FAILS OR IRCF Z2(1) DOES NOT GET TO RACK E49 OR RACK
2271 :*          E49(B1) IS STUCK LOW EXECUTION WILL GO FROM DIV.10 TO
2272 :*          DIV.20. THIS WILL CAUSE THE DIVIDE TO ABORT (GO TO DVE.20)
2273 :*          AFTER STATE DIV.60 AND THE C BIT WILL BE CLEAR.
2274 :*
2275 :*          SECTION 2
2276 :*          THE NEXT SECTION DIVIDES 4 BY 2. IF RACK E49(B1) IS STUCK
2277 :*          HIGH THE ALGORITHM WILL ABORT THINKING THAT THE DIVISOR
2278 :*          IS ZERO. IF BEN04 FAILS THE DIVIDE WILL ABORT THINKING
2279 :*          THAT THE DIVIDEND IS THE MOST NEGATIVE NUMBER.
2280 :*          IF BEN16 FAILS THE DIVIDE WILL COMPLETE BUT R1 WILL CONTAIN
```



```

2281 : * 155776. IF BEN05 FAILS THE ALGORITHM WILL ABORT THRU STATE
2282 : * DVE.20. IF BEN04 (AFTER DIV.70) FAILS RO WILL END UP WITH 177777
2283 : * AND R1 WILL HAVE 177774. IF BEN03 FAILS RO WILL END UP WITH
2284 : * 20 AND R1 WILL HAVE 177774. IF BEN16 FAILS AFTER DVC.00 RO
2285 : * WILL HAVE 2 BUT R1 WILL HAVE 177776.
2286 : *
2287 : * SECTION 3
2288 : * THE NEXT SECTION DIVIDES 6 BY 2 TO TEST BEN16*DR0(1).
2289 : * A FAILURE WILL LEAVE THE REMAINDER (R1)=2 INSTEAD OF ZERO.
2290 : *
2291 : * SECTION 4
2292 : * THE NEXT SECTION DIVIDES 4 BY -2 TO TEST BEN15*SR15(1).
2293 : * IF THIS FAILS RO WILL CONTAIN 27777 AND R1 WILL CONTAIN 4.
2294 : *
2295 : * SECTION 5
2296 : * THE NEXT SECTION DIVIDES 1177777 BY 1 TO TEST BEN05*DIV QUIT.
2297 : * IF THIS FAILS RO WILL CONTAIN 177777.
2298 : *
2299 : * SECTION 6
2300 : * THE NEXT SECTION DIVIDES 1000000 B2 -2 TO TEST BEN05*DIV QUIT.
2301 : * THIS SECTION WILL ONLY FAIL IF GRAJ E5 (Z2(0)*LEFT SAVE (1)) IS BAD.
2302 : *
2303 : * SECTION 7
2304 : * THE NEXT SECTION DIVIDES 100000000000 BY 2 TO TEST
2305 : * BEN04*NEGATIVE DIVIDEND.
2306 : *
2307 : * SECTION 8
2308 : * THE NEXT SECTION DIVIDED 177776 177777 BY -1 TO TEST BEN05*DIV QUIT.
2309 : * THIS TEST WILL ONLY FAIL IF GRAJ E5(N(1)*SR15(1)) IS BAD.
2310 : *
2311 : * SECTION 9
2312 : * THE NEXT SECTION DIVIDES -5 BY 2 TO ENSURE THAT THE REMAINDER IS
2313 : * STORED AS A NEGATIVE NUMBER.
2314 : *
2315 : * SECTION 10
2316 : * THE NEXT SECTION DIVIDES -5 BY -2 TO TEST STATES DVC.20,DVC.40, & DVC.60
2317 : *
2318 : * SECTION 11
2319 : * THE NEXT SECTION DIVIDES -2**16 BY 2**14 TO TEST STATE DVN.20
2320 : *
2321 : * SECTION 12
2322 : * THE NEXT SECTION DIVIDES 100 000200 BY -177 TO TEST STATES
2323 : * DVD.00 AND DVD.10.
2324 : * *****
2325 011120 000004 TST14: SCOPE
2326 011122 012767 012600 170140 MOV #TST15,NEXTTST ;SAVE ADDRESS OF NEXT TEST
2327 :DIVISOR=0 SECTION (ALSO TESTS CONDITION CODE ROM FOR DIV)
2328 011130 005000 CLR R0 ;PUT DIVIDEND IN
2329 011132 012701 000004 MOV #4,R1 ;RO AND R1
2330 011136 005002 CLR R2 ;MAKE DIVISOR=0
2331 011140 000270 SEN ;ENSURE N SET
2332 :AND OSCILLOSCOPE SYNC POINT
2333 011142 071002 DIV R2,R0 ;EXECUTE INSTRUCTION UNDER TEST
2334 011144 013767 177776 170034 MOV @#PSW,$ERPSW ;SAVE PSW
2335 011152 042767 177760 170026 BIC #177760,$ERPSW ;MASK OFF THE CC'S
2336 011160 022767 000007 170020 CMP #7,$ERPSW ;DID THE CC'S COME OUT OK?

```

```
2337 011166 001412          BEQ      2$          ;BRANCH IF YES
2338 011170 022767 000002 170010  CMP      #2,$ERPSW  ;DID INSTR GO THROUGH DVE.20?
2339 011176 001002          BNE      3$          ;BRANCH IF NO
2340 011200 104075          ERROR   75          ;BEN02 FAILED
2341 011202 000404          BR       2$
2342 011204 012767 000007 167750 3$:  MOV      #7,$TMP0   ;SAVE EXPECTED VALUE
2343 011212 104076          ERROR   76          ;CC'S BAD IN STATE DVE.00
2344                                     ;*****
2345                                     ;SECTION TWO (ALSO CHECKS CC LOAD OF STATE DVC.70)
2346 011214 012767 011222 167666 2$:  MOV      #64,$LPERR ;SETUP ERROR LOOP
2347 011222 012702 000002 64$:  MOV      #2,R2      ;PUT DIVISOR IN R2 (R0 & R1 HOLD DIVIDEND)
2348 011226 000277          SCC
2349                                     ;AND OSCILLOSCOPE SYNC POINT
2350 011230 071002          DIV     R2,R0      ;EXECUTE INSTRUCTION UNDER TEST
2351 011232 013767 177776 167746  MOV      @#PSW,$ERPSW ;SAVE PSW
2352 011240 020027 000002          CMP     R0,#2      ;DID QUOTIENT COME OUT CORRECT?
2353 011244 001042          BNE     4$          ;BRANCH IF NO
2354 011246 005701          TST    R1          ;DID REMAINDER COME OUT CORRECT?
2355 011250 001013          BNE     5$          ;BRANCH IF NO
2356 011252 042767 177760 167726  BIC     #177760,$ERPSW ;MASK OFF CC'S
2357 011260 022767 000000 167720  CMP     #0,$ERPSW  ;ARE CC'S CORRECT?
2358 011266 001513          BEQ     7$          ;BRANCH IF YES
2359 011270 005067 167666          CLR     $TMP0      ;SAVE EXPECTED VALUE
2360 011274 104077          ERROR   77          ;BAD CC'S DUE TO STATE DVC.70
2361 011276 000507          BR      7$
2362 011300 012767 000002 167654 5$:  MOV      #2,$TMP0   ;SAVE EXPECTED
2363 011306 005067 167652          CLR     $TMP1      ;VALUES
2364 011312 020127 177776          CMP     R1,#177776 ;DID BEN16*DR0(0) FAIL?
2365 011316 001002          BNE     8$          ;BRANCH IF NO
2366 011320 104100          ERROR   100        ;RACK E50(B0) STUCK HIGH
2367 011322 000475          BR      7$
2368 011324 020127 000004          8$:  CMP     R1,#4      ;DID BEN04 FAIL?
2369 011330 001002          BNE     20$         ;BRANCH IF NO
2370 011332 104101          ERROR   101        ;BEN04 FAILED
2371 011334 000470          BR      7$
2372 011336 010067 167612          20$: MOV      R0,$REG0   ;SAVE R0 &
2373 011342 010167 167610          MOV     R1,$REG1   ;R1 FOR TYPEOUT
2374 011346 104102          ERROR   102        ;QUOTIENT OK BUT REMAINDER BAD
2375 011350 000462          BR      7$
2376 011352 012767 000002 167602 4$:  MOV      #2,$TMP0   ;SAVE EXPECTED
2377 011360 005067 167600          CLR     $TMP1      ;VALUES
2378 011364 022700 000020          CMP     #20,R0     ;DID BEN03 FAIL?
2379 011370 001002          BNE     9$          ;BRANCH IF NO
2380 011372 104103          ERROR   103        ;RACK E49(A1) STUCK LOW
2381 011374 000450          BR      7$
2382 011376 022700 077777          9$:  CMP     #77777,R0  ;DID BEN04 (-DIV SUB) FAIL?
2383 011402 001002          BNE     10$         ;BRANCH IF NO
2384 011404 104104          ERROR   104        ;BEN04 STUCK TO DIV SUB
2385 011406 000443          BR      7$
2386 011410 022700 177777          10$: CMP     #-1,R0     ;DID R0 GET A -1?
2387 011414 001002          BNE     13$         ;BRANCH IF NO
2388 011416 104110          ERROR   110        ;BEN04 (-N) FAILED
2389 011420 000436          BR      7$
2390 011422 020027 000000          13$: CMP     R0,#0     ;DID R0 STAY 0?
2391 011426 001406          BEQ     11$         ;BRANCH IF YES
2392 011430 010067 167520          MOV     R0,$REG0   ;SAVE R0 & R1
```

```
2393 011434 010167 167516      MOV    R1,$REG1      ;FOR TYPEOUT
2394 011440 104105      ERROR  105          ;INSTRUCTION FAILED
2395 011442 000425      BR     7$
2396 011444 020127 155776      11$:  CMP    R1,#155776  ;DID BEN16*SR15(0) FAIL?
2397 011450 001002      BNE   12$          ;BRANCH IF NO
2398 011452 104106      ERROR  106          ;BEN16 FAILED TO DIV.50
2399 011454 000420      BR     7$
2400 011456 020127 000004      12$:  CMP    R1,#4       ;DID R1 CHANGE?
2401 011462 001406      BEQ   14$          ;BRANCH IF NO
2402 011464 010067 167464      MOV    R0,$REG0     ;SAVE R0 & R1
2403 011470 010167 167462      MOV    R1,$REG1     ;FOR TYPEOUT
2404 011474 104107      ERROR  107          ;CAN'T DETERMINE FAILURE
2405 011476 000407      BR     7$
2406 011500 032767 000001 167500 14$:  BIT    #BIT0,$ERPSW ;DID BEN02 FAIL?
2407 011506 001402      BEQ   15$          ;BRANCH IF NO
2408 011510 104111      ERROR  111          ;BEN02 FAILED
2409 011512 000401      BR     7$
2410 011514 104112      15$:  ERROR  112          ;BEN05 FAILED
2411
2412      ;*****
2413      ;SECTION THREE
2413 011516 012767 011524 167364 7$:  MOV    #63$,$LPERR  ;SETUP ERROR LOOP
2414 011524 005000      63$:  CLR    R0          ;PUT DIVIDEND IN R0
2415 011526 012701 000006      MOV    #6,R1        ;PUT DIVIDEND IN R1 (DIVISOR-2 IN R2)
2416 011532 000240      NOP
2417 011534 071002      DIV   R2,R0        ;OSCILLOSCOPE SYNC POINT
2418 011536 020127 000002      CMP    R1,#2        ;EXECUTE INSTRUCTION UNDER TEST
2419 011542 001001      BNE   16$          ;DID BEN16*DRO(1) FAIL?
2420 011544 104113      ERROR  113          ;BRANCH IF NO
2421      ;*****
2422      ;SECTION FOUR (ALSO CHECK CC LOAD OF STATE DVC.90)
2423 011546 012767 011554 167334 16$:  MOV    #62$,$LPERR  ;SETUP ERROR LOOP
2424 011554 005000      62$:  CLR    R0
2425 011556 012701 000004      MOV    #4,R1        ;PUT DIVIDEND IN R0 & R1
2426 011562 012702 177776      MOV    #-2,R2       ;PUT DIVISOR IN R2
2427 011566 012767 177776 167366      MOV    #-2,$TMP0    ;SAVE EXPECTED
2428 011574 005067 167364      CLR    $TMP1        ;VALUES
2429 011600 000240      NOP
2430 011602 071002      DIV   R2,R0        ;OSCILLOSCOPE SYNC POINT
2431 011604 013767 177776 167374      MOV    @#PSW,$ERPSW ;EXECUTE INSTRUCTION UNDER TEST
2432 011612 020027 177776      MOV    @#PSW,$ERPSW ;SAVE CC'S
2433 011616 001413      CMP    R0,#177776  ;DID DIVIDE WORK?
2434 011620 020027 027777      BEQ   17$          ;BRANCH IF YES
2435 011624 001002      CMP    R0,#27777  ;DID BEN16*SR15(1) FAIL?
2436 011626 104114      BNE   18$          ;BRANCH IF NO
2437 011630 000432      ERROR  114          ;RACK E64(B0) STUCK LOW
2438 011632 010067 167316      18$:  BR     21$
2439 011636 010167 167314      MOV    R0,$REG0     ;SAVE R0 & R1 FOR
2440 011642 104115      MOV    R1,$REG1     ;TYPEOUT
2441      ERROR  115          ;EITHER STATE DVC.20 OR DVC.40 OR DVC.80
2442      ;OR DVC.90 FAILED
2443 011644 000424      BR     21$
2444 011646 020127 000000      17$:  CMP    R1,#0        ;DID REMAINDER COME OUT CORRECT?
2445 011652 001406      BEQ   19$          ;BRANCH IF YES
2446 011654 010067 167274      MOV    R0,$REG0     ;SAVE R0 & R1
2447 011660 010167 167272      MOV    R1,$REG1     ;FOR TYPEOUT
2448 011664 104116      ERROR  116          ;QUOTIENT OK, REMAINDER BAD
2448 011666 000413      BR     21$
```

```
2449 011670 042767 177760 167310 19$: BIC #177760,$ERPSW ;MASK OF CC'S
2450 011676 022767 000010 167302 CMP #10,$ERPSW ;DID CC'S COME OUT CORRECT?
2451 011704 001404 BEQ 21$ ;CONTINUE IF YES
2452 011706 012767 000010 167246 MOV #10,$TMP0 ;SAVE EXPECTED VALUE
2453 011714 104117 ERROR 117 ;BAD CC'S DUE TO STATE DVC.90
2454
2455 ;*****
2456 011716 012767 011724 167164 ;SECTION FIVE (DIV QUIT CAUSED BY N(0)*SR15(0))
2457 011724 012700 000002 21$: MOV #61$,$LPERR ;SETUP ERROR LOOP
2458 011730 005001 61$: MOV #2,$R0 ;SETUP R0 AND R1
2459 011732 012702 000001 CLR R1 ;TO CAUSE DIV QUIT ABORT
2460 011736 000277 MOV #1,$R2 ;PUT DIVISOR IN R2
2461 011740 000242 SCC ;SETUP CC'S TO COMPLIMENT
2462 CLV ;OF EXPECTED VALUE
2463 011742 071002 DIV R2,$R0 ;AND OSCILLOSCOPE SYNC POINT
2464 011744 013767 177776 167234 MOV @#PSW,$ERPSW ;EXECUTE INSTRUCTION UNDER TEST
2465 011752 020027 000002 CMP R0,#2 ;SAVE CC'S
2466 011756 001402 BEQ 22$ ;DID DIVIDE ABORT?
2467 011760 104120 ERROR 120 ;BRANCH IF YES
2468 011762 000413 BR 23$ ;DIV QUIT DID NOT GO LOW
2469 011764 042767 177760 167214 22$: BIC #177760,$ERPSW ;MASK OFF CC'S
2470 011772 022767 000002 167206 CMP #2,$ERPSW ;DID CC'S COME OUT OK?
2471 012000 001404 BEQ 23$ ;BRANCH IF YES
2472 012002 012767 000002 167152 MOV #2,$TMP0 ;STORE EXPECTED CC'S FOR TYPEOUT
2473 012010 104121 ERROR 121 ;CC'S BAD DUE TO EITHER DIV.30 OR DVE.20
2474
2475 ;*****
2476 012012 012767 012020 167070 ;SECTION SIX (DIV QUIT CAUSED BY SHFTR=0)
2477 012020 012700 000002 23$: MOV #60$,$LPERR ;SETUP ERROR LOOP
2478 012024 005001 60$: MOV #2,$R0 ;SETUP R0 & R1 TO
2479 012026 012702 177776 CLR R1 ;CAUSE DIV QUIT TO ABORT
2480 012032 000240 MOV #-2,$R2 ;SETUP DIVISOR (-2)
2481 012034 071002 NOP ;OSCILLOSCOPE SYNC POINT
2482 012036 022700 000002 DIV R2,$R0 ;EXECUTE INSTRUCTION UNDER TEST
2483 012042 001401 CMP #2,$R0 ;DID DIVIDE ABORT?
2484 012044 104122 BEQ 24$ ;BRANCH IF YES
2485 ERROR 122 ;GRAJ E5 IS BAD (Z2(0)*LEFT SAVE (1))
2486 ;*****
2487 012046 012767 012054 167034 ;SECTION SEVEN
2488 012054 012700 100000 24$: MOV #57$,$LPERR ;SETUP ERROR LOOP
2489 012060 005001 57$: MOV #BIT15,$R0 ;MAKE DIVIDEND
2490 012062 012702 000001 CLR R1 ;MOST NEGATIVE NUMBER
2491 012066 000257 MOV #1,$R2 ;SETUP DIVISOR
2492 012070 000264 CCC ;SET CC'S TO COMPLIMENT OF EXPECTED
2493 012072 071002 SEZ ;OSCILLOSCOPE SYNC POINT
2494 012074 013767 177776 167104 DIV R2,$R0 ;EXECUTE INSTRUCTION UNDER TEST
2495 012102 020027 100000 MOV @#PSW,$ERPSW ;SAVE PSW
2496 012106 001402 CMP R0,#BIT15 ;DID DIVIDE ABORT?
2497 012110 104123 BEQ 25$ ;BRANCH IF YES
ERROR 123 ;BEN04*N FAILED
```

```
2498 012112 000413  
2499 012114 042767 177760 167064 25$: BR 26$  
2500 012122 022767 000016 167056 BIC #177760,$ERPSW ;MASK OFF CC'S  
2501 012130 001404 CMP #16,$ERPSW ;DID CC'S LOAD PROPERLY?  
2502 012132 012767 000016 167022 BEQ 26$ ;BRANCH IF YES  
2503 012140 104124 MOV #16,$STMP0 ;STORE EXPECTED VALUE  
2504 ERROR 124 ;CC'S DID NOT LOAD PROPERLY  
2505 :*****  
2506 012142 012767 012150 166740 :SECTION EIGHT (DIV QUIT CAUSED BY N(1)*SR15(1))  
2507 012150 012700 177776 26$: MOV #56$,$LPERR ;SETUP ERROR LOOP  
2508 012154 012701 177777 56$: MOV #177776,R0 ;SETUP RO & R1 TO  
2509 012160 012702 177777 MOV #177777,R1 ;CAUSE DIV QUIT TO ABOR*  
2510 012164 000240 MOV #177777,R2 ;SETUP DIVISOR (-1)  
2511 012166 071002 NOP ;OSCILLOSCOPE SYNC POINT  
2512 012170 020027 000001 DIV R2,R0 ;EXECUTE INSTRUCTION UNDER TEST  
2513 012174 001401 CMP R0,#1 ;DID DIVIDE ABORT LEAVE RO 1  
2514 012176 000403 BEQ 27$ ;BRANCH IF YES  
2515 012200 020127 000001 27$: BR 35$  
2516 012204 001413 CMP R1,#1 ;DID DIVIDE ABORT?  
2517 012206 012767 000001 166746 35$: BEQ 28$ ;BRANCH IF YES  
2518 012214 010067 166734 MOV #1,$STMP0 ;STORE EXPECTED VALUE  
2519 012220 012767 000001 166736 MOV R0,$REG0 ;SAVE RO  
2520 012226 010167 166724 MOV #1,$STMP1  
2521 012232 104125 MOV R1,$REG1 ;SAVE R1  
2522 ERROR 125 ;EITHER GRAJ E5 BAD OR MICROSTATE BAD  
2523 :*****  
2524 012234 012767 012242 166646 :SECTION NINE  
2525 012242 012700 177777 28$: MOV #55$,$LPERR ;SETUP ERROR LOOP  
2526 012246 012701 177773 55$: MOV #-1,R0 ;PUT DIVIDEND IN  
2527 012252 012702 000002 MOV #-5,R1 ;RO & R1  
2528 012256 012767 177776 166676 MOV #2,R2 ;PUT DIVISOR IN R2  
2529 012264 012767 177777 166672 MOV #-2,$STMP0 ;SAVE EXPECTED VALUE  
2530 012272 000240 MOV #-1,$STMP1 ;SAVE EXPECTED VALUE  
2531 012274 071002 NOP ;OSCILLOSCOPE SYNC POINT  
2532 012276 010067 166652 DIV R2,R0 ;EXECUTE INSTRUCTION UNDER TEST  
2533 012302 010167 166650 MOV R0,$REG0 ;SAVE RO  
2534 012306 020127 177777 MOV R1,$REG1 ;SAVE R1  
2535 012312 001402 CMP R1,#-1 ;IS REMAINDER CORRECT?  
2536 012314 104126 BEQ 29$ ;BRANCH IF YES  
2537 012316 000404 ERROR 126 ;REMAINDER BAD  
2538 012320 022700 177776 29$: BR 30$  
2539 012324 001401 CMP #-2,R0 ;IS QUOTIENT CORRECT?  
2540 012326 104127 BEQ 30$ ;BRANCH IF YES  
2541 ERROR 127 ;QUOTIENT IS INCORRECT  
2542 :*****  
2543 012330 012767 012336 166552 :SECTION TEN  
2544 012336 012700 177777 30$: MOV #54$,$LPERR ;SETUP ERROR LOOP  
2545 012342 012701 177773 54$: MOV #-1,R0 ;SETUP  
2546 012346 012702 177776 MOV #-5,R1 ;DIVIDEND  
2547 012352 012767 000002 166602 MOV #-2,R2 ;AND DIVISOR  
2548 012360 012767 177777 166576 MOV #2,$STMP0 ;SAVE EXPECTED VALUES  
2549 012366 000240 MOV #-1,$STMP1 ;OF RO & R1  
2550 012370 071002 NOP ;OSCILLOSCOPE SYNC POINT  
2551 012372 010067 166556 DIV R2,R0 ;EXECUTE INSTRUCTION UNDER TEST  
2552 012376 010167 166554 MOV R0,$REG0 ;SAVE RO  
2553 012402 020027 000002 MOV R1,$REG1 ;SAVE R1  
CMP R0,#2 ;IS QUOTIENT CORRECT?
```

```
2554 012406 001402          BEQ      31$          ;BRANCH IF YES
2555 012410 104130          ERROR    130          ;QUOTIENT BAD
2556 012412 000404          BR       32$
2557 012414 020127 177777 3'S:    CMP      R1,#-1      ;IS REMAINDER CORRECT
2558 012420 001401          BEQ      32$          ;BRANCH IF YES
2559 012422 104131          ERROR    131          ;REMAINDER INCORRECT (QUOT. OK)
2560
2561          ;*****
2562 012424 012767 012432 166456 32$:    MOV      #53$,$LPERR ;SETUP ERROR LOOP
2563 012432 012700 177776 53$:    MOV      #177776,R0   ;SETUP DIVIDEND
2564 012436 005001          CLR      R1           ;AND DIVISOR TO GET A
2565 012440 012702 040000          MOV      #40000,R2   ;QUOT OF -10 & REMAINDER 0
2566 012444 012767 177770 166510          MOV      #-10,$TMP0  ;SAVE EXPECTED VALUE
2567 012452 005067 166506          CLR      $TMP1       ;SAVE EXPECTED VALUE
2568 012456 000240          NOP
2569 012460 071002          DIV      R2,R0       ;EXECUTE INSTRUCTION UNDER TEST
2570 012462 010067 166466          MOV      R0,$REG0    ;SAVE R0
2571 012466 010167 166464          MOV      R1,$REG1    ;SAVE R1
2572 012472 020027 177770          CMP      R0,#-10     ;IS QUOTIENT CORRECT?
2573 012476 001402          BEQ      33$          ;BRANCH IF YES
2574 012500 104132          ERROR    132          ;QUOTIENT IS BAD
2575 012502 000404          BR       34$
2576 012504 020127 000000 33$:    CMP      R1,#0       ;IS REMAINDER CORRECT?
2577 012510 001401          BEQ      34$          ;BRANCH IF YES
2578 012512 104133          ERROR    133          ;REMAINDER BAD (QUOT. OK)
2579
2580          ;*****
2581 012514 012767 012522 166366 34$:    MOV      #52$,$LPERR ;SETUP ERROR LOOP
2582 012522 012700 000100 52$:    MOV      #100,R0     ;SETUP DIVIDEND
2583 012526 012701 000200          MOV      #200,R1     ;AND DIVISOR TO GENERATE
2584 012532 012702 177601          MOV      #-177,R2    ;DIVISION OVERFLOW
2585 012536 000277          SCC
2586 012540 000242          CLV
2587 012542 071002          DIV      R2,R0       ;AND OSCILLOSCOPE SYNC POINT
2588 012544 013767 177776 166434          MOV      @#PSW,$ERPSW ;EXECUTE INSTRUCTION UNDER TEST
2589 012552 042767 177760 166426          BIC      #177760,$ERPSW ;SAVE PSW
2590 012560 022767 000002 166420          CMP      #2,$ERPSW   ;MASK OFF CC'S
2591 012566 001404          BEQ      TST15       ;ARE CC'S CORRECT?
2592 012570 012767 000002 166364          MOV      #2,$TMP0   ;TEST OK, GO TO NEXT TEST
2593 012576 104134          ERROR    134          ;SAVE EXPECTED
2594          ;*****
2595          ;TEST 15      MTP*DMO
2596          ;
2597          ; IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
2598          ;
2599          ; THE ONLY OTHER POSSIBLE FAILURES WOULD BE IN ROM STATES
2600          ; MTP.00 OR MTP.10.
2601          ;
2602          ; NOTE: THIS TEST ONLY TESTS THE CPU FUNCTIONS OF THIS INSTRUCTION.
2603          ; THE MEMORY MANAGEMENT TEST VERIFIES THE INTERMODE TRANSFER.
2604          ; AS FAR AS THE CPU IS CONCERNED THERE IS NO DIFFERENCE
2605          ; BETWEEN MTP1 AND MTPD.
2606          ;
2607          ; ROM FLOW-45,151,146,205
2608          ;*****
2609 012600 000004          TST15:  SCOPE
```

2610	012602	012767	012746	166364	MOV	#TST16,\$ESCAPE	:SAVE START ADDRESS OF NEXT TEST
2611	012610	012767	012746	166452	MOV	#TST16,NEXTTST	:SAVE START ADDRESS OF NEXT TEST
2612	012616	012737	012736	000010	MOV	#15,@RESVEC	:SETUP RESVEC
2613	012624	005037	177776		CLR	@PSW	:ENSURE PREVIOUS MODE KERNAL
2614	012630	012706	001074		MOV	#1074,SP	:SETUP THE SP
2615	012634	012716	100000		MOV	#BIT15,(SP)	:SET SIGN BIT ON STACK
2616	012640	005000			CLR	R0	:ENSURE R0 CLEAR
2617	012642	000277			SCC		:SET CC'S TO COMPLIMENT
2618	012644	000250			CLN		:OF EXPECTED VALUE (EXCEPT FOR C)
2619							:AND OSCILLOSCOPE SYNC POINT
2620	012646	006600			MTP1	R0	:EXECUTE INSTRUCTION UNDER TEST
2621	012650	013767	177776	166330	MOV	@PSW,\$ERPSW	:SAVE PSW
2622	012656	010067	166272		MOV	R0,\$REG0	:SAVE R0 & SEC CC'S
2623	012662	012767	100000	166272	MOV	#BIT15,\$TMP0	:SAVE EXPECTED VALUE
2624	012670	100403			BMI	2\$:BRANCH IF R0 LOADED
2625	012672	012700	100000		MOV	#BIT15,R0	:SAVE EXPECTED VALUE
2626	012676	104135			ERROR	135	:R0 DID NOT LOAD
2627	012700	022706	001076		2\$: CMP	#1076,SP	:DID SP INCREMENT?
2628	012704	001401			BEQ	3\$:BRANCH IF YES
2629	012706	104136			ERROR	136	:SP DID NOT INCREMENT
2630	012710	042767	177760	166270	3\$: BIC	#177760,\$ERPSW	:MASK OFF CC'S
2631	012716	022767	000011	166262	CMP	#11,\$ERPSW	:DID CC'S COME OUT CORRECT?
2632	012724	001410			BEQ	TST16	:BRANCH IF YES
2633	012726	012767	000011	166226	MOV	#11,\$TMP0	:SAVE EXPECTED VALUE
2634	012734	104137			ERROR	137	:CC'S BAD (CC ROM)
2635	012736	012737	000012	000010	1\$: MOV	#12,@RESVEC	:RESTORE RESVEC
2636	012744	104140			ERROR	140	:FORK A FAILED

:TEST 16 MTP*DM1

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

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

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

2649	012746	000004			TST16: SCOPE		
2650	012750	012767	013120	166312	MOV	#TST17,NEXTTST	:SAVE ADDRESS OF NEXT TEST
2651	012756	012737	013110	000010	MOV	#15,@RESVEC	:SETUP RESVEC
2652	012764	012706	001072		MOV	#1072,SP	:SETUP THE SP
2653	012770	012716	100000		MOV	#BIT15,(SP)	:SET THE SIGN BIT ON THE STACK
2654	012774	005066	000002		CLR	2(SP)	:ENSURE 1074 IS CLEAR
2655	013000	000240			NOP		:OSCILLOSCOPE SYNC POINT
2656	013002	006616			MTP1	(SP)	:EXECUTE INSTRUCTION UNDER TEST
2657	013004	005737	001074		TST	@1074	:DID 1074 GET SIGN BIT SET?
2658	013010	100413			BMI	2\$:BRANCH IF YES
2659	013012	022706	001074		CMP	#1074,SP	:DID MTP.10 FAIL TO RELOAD THE DR?
2660	013016	001002			BNE	3\$:BRANCH IF NO
2661	013020	104141			ERROR	141	:STATE MTP.10 FAILED
2662	013022	000406			BR	2\$	
2663	013024	010667	166124		3\$: MOV	SP,\$REG0	:SAVE SP
2664	013030	012767	001074	166120	MOV	#1074,\$REG1	:SAVE EXPECTED VALUE
2665	013036	104142			ERROR	142	:DON'T KNOW WHAT HAPPENED

```
2666 013040 012767 013046 166042 2$: MOV #64$, $LPERR ; SETUP ERROR LOOP
2667 013046 012706 001074 64$: MOV #1074, SP ; SETUP THE SP
2668 013052 000240 NOP ; OSCILLOSCOPE SYNC POINT
2669 013054 006617 MTP1 (PC) ; EXECUTE INSTRUCTION UNDER TEST
2670 013056 000000 4$: .WORD 0
2671 013060 005767 177772 TST 4$ ; DID 4$ GET BIT15 SET?
2672 013064 100403 BMI 5$ ; BRANCH IF YES
2673 013066 005067 177764 CLR 4$ ; RESTORE 4$
2674 013072 104143 ERROR 143 ; STATE MTP.10 DID NOT PUT PCB IN DR
2675 013074 005067 177756 5$: CLR 4$ ; RESTORE 4$
2676 013100 012737 000012 000010 MOV #12, @#RESVEC ; RESTORE RESVEC
2677 013106 000404 BR TST17 ; GO TO NEXT TEST
2678 013110 012737 000012 000010 1$: MOV #12, @#RESVEC ; RESTORE RESVEC
2679 013116 104144 ERROR 144 ; FORK A FAILED
2680 *****
2681 ; *TEST 17 MFP*DMO
2682 ;
2683 ; IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
2684 ; IF ANYTHING ELSE FAILS, THEN A ROM STATE IS BAD.
2685 ;
2686 ; ROM FLOW-46,304,250,222,300
2687 ; *****
2688 013120 000004 TST17: SCOPE
2689 013122 012767 013306 166044 MOV #TST20, $ESCAPE ; SAVE START ADDRESS OF NEXT TEST
2690 013130 012767 013306 166132 MOV #TST20, NEXTTST ; SAVE START ADDRESS OF NEXT TEST
2691 013136 012767 013172 165742 MOV #4$, $LPADR ; SET UP LOOP
2692 013144 012767 013172 165736 MOV #4$, $LPERR ; SETUP ERROR LOOP
2693 013152 013767 177776 166010 MOV @#PSW, $TMP3 ; SAVE PSW
2694 013160 012746 000340 MOV #PR7, -(SP) ; PUT NEW PSW ON STACK
2695 013164 012746 013172 MOV #4$, -(SP) ; PUT RETURN ADDR ON STACK
2696 013170 000006 RTT ; TURN T BIT OFF
2697 013172 012737 013276 000010 4$: MOV #1$, @#RESVEC ; SETUP RESVEC
2698 013200 012706 001076 MOV #1076, SP ; SETUP THE SP
2699 013204 005016 CLR (SP) ; ENSURE 1076 CLEAR
2700 013206 005066 177776 CLR -2(SP) ; ENSURE 1074 CLEAR
2701 013212 012700 100000 MOV #BIT15, R0 ; SET THE SIGN BIT IN R0
2702 013216 000277 SCC ; SETUP CC'S TO COMPLIMENT
2703 013220 000250 CLN ; OF EXPECTED VALUE (EXCEPT FOR C)
2704 ; AND OSCILLOSCOPE SYNC POINT
2705 013222 006500 MFPI R0 ; EXECUTE INSTRUCTION UNDER TEST
2706 013224 013767 177776 165754 MOV @#PSW, $ERPSW ; SAVE THE PSW
2707 013232 022706 001074 CMP #1074, SP ; DID THE SP DECREMENT?
2708 013236 001401 BEQ 2$ ; BRANCH IF YES
2709 013240 104145 ERROR 145 ; STATE MFP.10 DID NOT DEC. THE SP
2710 013242 005716 2$: TST (SP) ; DID R0 GET PUT ON THE STACK?
2711 013244 100401 BMI 3$ ; BRANCH IF YES
2712 013246 104146 ERROR 146 ; R0 DID NOT GET PUT ON THE STACK
2713 013250 042767 177760 165730 3$: BIC #177760, $ERPSW ; MASK OFF THE CC'S
2714 013256 022767 000011 165722 CMP #11, $ERPSW ; DID THE CC'S SET CORRECTLY?
2715 013264 001410 BEQ TST20 ; BRANCH IF YES
2716 013266 012767 000011 165666 MOV #11, $TMP0 ; SAVE EXPECTED VALUE
2717 013274 104147 ERROR 147 ; INCORRECT CC'S
2718 013276 012737 000012 000010 1$: MOV #12, @#RESVEC ; RESTORE RESVEC
2719 013304 104150 ERROR 150 ; FORK A FAILED
2720 *****
2721 ; *TEST 20 MFP*DM2
```



```
2722
2723
2724
2725
2726
2727
2728
2729
2730 013306 000004
2731 013310 012767 013446 165656
2732 013316 012767 013446 165744
2733 013324 012737 013426 000010
2734 013332 012706 001076
2735 013336 005016
2736 013340 005066 177776
2737 013344 012700 001164
2738 013350 012710 100000
2739 013354 000277
2740 013356 000250
2741
2742 013360 006520
2743 013362 013767 177776 165616
2744 013370 022706 001074
2745 013374 001401
2746 013376 104151
2747 013400 042767 177760 165600 2$:
2748 013406 022767 000011 165572
2749 013414 001414
2750 013416 012767 000011 165536
2751 013424 104152
2752 013426 012737 000012 000010 1$:
2753 013434 022700 001166
2754 013440 001401
2755 013442 104153
2756 013444 104154 3$:
2757
2758
2759
2760
2761
2762
2763
2764
2765
2766
2767
2768 013446 000004
2769 013450 012767 013562 165516
2770 013456 012767 013562 165604
2771 013464 012737 013522 000014 4$:
2772 013472 012706 001100
2773 013476 012737 013542 000004
2774 013504 012737 013544 000024
2775 013512 012703 000024
2776 013516 000237
2777 013520 000003
```

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

TST20: SCOPE
MOV #TST21,\$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
MOV #TST21,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
MOV #1\$,@#RESVEC ;SETUP RESVEC
MOV #1076,SP ;SETUP THE SP
CLR (SP) ;ENSURE WORDS ON
CLR -2(SP) ;STACK CLEAR
MOV #STMP1,R0 ;PUT ADDRESS OF STMP1 IN R0
MOV #BIT15,(R0) ;SET THE SIGN BIT IN STMP1
SCC ;SETUP CC'S TO COMPLIMENT OF
CLN ;EXPECTED VALUE (EXCEPT FOR C)
;AND OSCILLOSCOPE SYNC POINT
MFP (R0)+ ;EXECUTE INSTRUCTION UNDER TEST
MOV @#PSW,\$ERPSW ;SAVE PSW
CMP #1074,SP ;DID THE SP DECREMENT?
BEQ 2\$;BRANCH IF YES
ERROR 151 ;STATE MFP.00 IS BAD
BIC #177760,\$ERPSW ;MASK OFF THE CC'S
CMP #11,\$ERPSW ;DID THE CC'S SET CORRECTLY?
BEQ TST21 ;BRANCH IF YES
MOV #11,\$TMP0 ;SAVE EXPECTED VALUE
ERROR 152 ;INCORRECT CC'S DUE TO STATE MFP.00
MOV #12,@#RESVEC ;RESTORE RESVEC
CMP #STMP2,R0 ;DID R0 INCREMENT?
BEQ 3\$;BRANCH IF YES
ERROR 153 ;FORK A FAILED
ERROR 154 ;FORK B FAILED

*TEST 21 BPT

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

TST21: SCOPE
MOV #TST22,\$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
MOV #TST22,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
MOV #1\$,@#BPTVEC ;SETUP BPT VECTOR
MOV #STACK,SP ;SETUP THE SP
MOV #2\$,@#ERRVEC ;SETUP LOCATION 4
MOV #3\$,@#24 ;SETUP LOCATION 24
MOV #24,R3 ;SETUP R3
SPL 7 ;SETUP PSW
BPT ;EXECUTE INSTRUCTION UNDER TEST

```
2778 013522 042737 177437 001076 1$: BIC #177437,@#1076 ;MASK OFF PRIORITIES OF OLD PS
2779 013530 022737 000340 001076 CMP #340,@#1076 ;DID OLD PS GET STACKED?
2780 013536 001403 BEQ 5$ ;BRANCH IF YES
2781 013540 104160 ERROR 160 ;STATE TRP.00 FAILED
2782 013542 104161 2$: ERROR 161 ;TRAP VECTOR CAME UP BAD
2783 013544 104162 3$: ERROR 162 ;STATE TRP.00 FAILED
2784 013546 012737 032654 000004 5$: MOV #CPUSPUR,@#ERRVEC ;RESTORE ERR VEC
2785 013554 012737 032640 000014 MOV #SRTN,@#TBITVEC ;RESTORE T BIT VECTOR
2786
2787 ;*****
2788 ;*ALL THE LOGIC FOR (JMP+JSR)*DMO HAS BEEN TESTED.
2789 ;*****
2790
2791 ;*****
2792 ;*TEST 22 BIT TEST OF PIRQ REGISTER
2793 ;*
2794 ;* IF ONE OF THE BLOCK LEVELS IS STUCK HIGH OR TMCB
2795 ;* PS07(0)'S STUCK HIGH OR PDRD PRIORITY=0 IS STUCK HIGH,
2796 ;* A PIRQ TRAP LOOP WILL OCCUR WHEN THAT PIR LEVEL IS ENABLED.
2797 ;*
2798 ;* A COUNT PATTERN IS THEN RUN THRU THE REGISTER TO ENSURE THAT
2799 ;* THE ENCODER FUNCTIONS PROPERLY.
2800 ;*****
2801 013562 000004 TST22: SCOPE
2802 013564 012767 013766 165402 MOV #TST23,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
2803 013572 012767 013766 165470 MOV #TST23,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
2804 013600 012706 001100 MOV #STACK,SP ;INITIALIZE THE SP
2805 013604 012767 013642 165274 MOV #4,$, $LPADR ;SETUP LOOP ADR
2806 013612 012767 013642 165270 MOV #4,$, $LPERR ;SETUP ERROR LOOP
2807 013620 032767 000020 165342 BIT #BIT4,$TMP3 ;WAS T BIT ON?
2808 013626 001405 BEQ 4$ ;BRANCH IF NO
2809 013630 012746 000360 MOV #360,-(SP) ;PUT NEW PSW ON STACK
2810 013634 012746 013642 MOV #4,-(SP) ;PUT RETURN ADDR ON STACK
2811 013640 000006 RTT ;TURN T BIT ON
2812 013642 000237 4$: SPL 7 ;SET THE CPU PRIORITY AT 7.
2813 013644 005067 165312 CLR $TMP0 ;SETUP COMPARISON LOCATION
2814 013650 005037 177772 CLR @#PIRQ ;CLEAR PIRQ REGISTER
2815 013654 026737 165302 177772 CMP $TMP0,@#PIRQ ;DID PIRQ CLEAR?
2816 013662 001035 BNE 1$ ;BRANCH IF NO
2817 013664 012700 000177 MOV #177,R0 ;SETUP ITERATION COUNT
2818 013670 012767 001042 165264 MOV #1042,$TMP0 ;SETUP COMPARISON LOCATION
2819 013676 012701 000002 MOV #2,R1 ;SETUP R1
2820 013702 062737 001000 177772 2$: ADD #1000,@#PIRQ ;START COUNT PATTERN
2821 013710 026737 165246 177772 CMP $TMP0,@#PIRQ ;DID REGISTER SET CORRECT?
2822 013716 001017 BNE 1$ ;BRANCH IF NO
2823 013720 120137 177773 CMPB R1,@#PIRQ+1 ;IS PIRQ READY TO GO TO NEXT LEVEL?
2824 013724 001005 BNE 3$ ;BRANCH IF NO
2825 013726 062767 000042 165226 ADD #42,$TMP0 ;INCREMENT ENCODED VALUES IN TEST LOC.
2826 013734 005201 INC R1 ;SETUP R1 FOR ROTATE
2827 013736 006101 ROL R1 ;SET R1 TO NEXT CHECK LEVEL
2828 013740 062767 001000 165214 3$: ADD #1000,$TMP0 ;INC. PIRQ LEVEL IN TEST LOCATION
2829 013746 077023 SOB R0,2$ ;CONTINUE COUNT
2830 013750 005037 177772 CLR @#PIRQ ;ENSURE PIRQ CLEAR
2831 013754 000404 BR TST23 ;GO TO NEXT TEST
2832 013756 013767 177772 165232 1$: MOV @#PIRQ,$EPIRQ ;SAVE PIRQ FOR TYPEOUT
2833 013764 104163 ERROR 163 ;PIRQ REG. FAILED
```

```
2834 .....  
2835 *TEST 23 PIR LEVEL 1 INTERRUPT  
2836 *  
2837 * IF BEN13 FAILS EXECUTION WOULD GO TO ONE OF THE FOLLOWING:  
2838 * PUP.00, BRK.20, OR SER.00.  
2839 * PUP.00 WOULD START THE POWER UP ROUTINE.  
2840 * BRK.20 WOULD CAUSE A TRAP TO ZERO.  
2841 * SER.00 WOULD PUT 4 IN THE SP AND PERFORM A RED ZONE TRAP.  
2842 * IF TMCB PIRQ DOES NOT GET TO DAPE OR IF DAPE TV05*07 DOES NOT  
2843 * GO HIGH OR DOES NOT GET THRU TO THE ALU A TRAP TO 4 WILL OCCUR.  
2844 * IF THE INTERRUPT DOESN'T OCCUR AN ATTEMPT IS MADE TO ISOLATE THE  
2845 * FAILURE.  
2846 .....  
2847 TST23: SCOPE  
2848 013766 000004 MOV #TST24,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST  
2849 013770 012767 014336 165176 MOV #TST24,NEXTTST ;SAVE START ADDRESS OF NEXT TEST  
2850 013776 012767 014336 165264 MOV #10$,@#PWVEC ;SETUP  
2851 014004 012737 014274 000024 MOV #12$,@#0 ;SETUP LOCATION 0  
2852 014012 012737 014326 000000 MOV #PR7,@#2 ;PUT PRIORITY 7 IN 2  
2853 014020 012737 000340 000002 MOV #11$,@#4 ;FAILURE TRAP VECTORS  
2854 014026 012737 014302 000004 MOV #1$,@#PIRQVEC ;SETUP PIRQ VECTOR  
2855 014034 012737 014252 000240 BIT #BIT4,@#PSW ;IS T BIT ON?  
2856 014042 032737 000020 177776 BEQ 13$ ;BRANCH IF NO  
2857 014050 001404 MOV #360,@#PIRQVEC+2 ;SET T BIT IN PIRQ VECTOR  
2858 014052 012737 000360 000242 BR 14$ ;  
2859 014060 000403 13$: MOV #PR7,@#PIRQVEC+2 ;SETUP PIRQ VECTOR PSW  
2860 014062 012737 000340 000242 14$: MOV #STACK,SP ;SETUP THE SP  
2861 014070 012706 001100 SPL 0 ;SET PRIORITY LEVEL AT ZERO  
2862 014074 000230 BIS #BIT9,@#PIRQ ;SET LEVEL ONE  
2863 014076 052737 001000 177772 CLR @#PIRQ ;CLEAR LEVEL ONE  
2864 014104 005037 177772 ;TRY TO GET INTERRUPT AT FET.01 INSTEAD OF TST.10.  
2865 014110 012737 014132 000240 MOV #2$,@#PIRQVEC ;SETUP PIRQ VEC  
2866 014116 012737 001000 177772 MOV #BIT9,@#PIRQ ;SET LEVEL ONE  
2867 014124 005037 177772 CLR @#PIRQ ;  
2868 014130 000403 BR 3$ ;BRANCH IF NO INTERRUPT  
2869 014132 005037 177772 2$: CLR @#PIRQ ;CLEAR LEVEL 1  
2870 014136 104164 ERROR 164 ;STATE TST.10 HAS BAD BEN FIELD  
2871 ;TRY PIR LEVEL 4  
2872 014140 012737 014162 000240 3$: MOV #4$,@#PIRQVEC ;SETUP PIRQ VECTOR  
2873 014146 052737 010000 177772 BIS #BIT12,@#PIRQ ;SET LEVEL 4  
2874 014154 005037 177772 CLR @#PIRQ ;  
2875 014160 000403 BR 5$ ;BRANCH IF NO INTERRUPT  
2876 014162 005037 177772 4$: CLR @#PIRQ ;CLEAR LEVEL 4  
2877 014166 104165 ERROR 165 ;EITHER TMCB E62(1) IS BAD OR SOMETHING  
2878 ;IN THE HONOR PIR 1 LOGIC IS BAD.  
2879 ;TRY PIR 6  
2880 014170 012737 014212 000240 5$: MOV #6$,@#PIRQVEC ;SETUP PIRQ VECTOR  
2881 014176 052737 040000 177772 BIS #BIT14,@#PIRQ ;SET LEVEL 6  
2882 014204 005037 177772 CLR @#PIRQ ;  
2883 014210 000403 BR 7$ ;BRANCH IF NO INTERRUPT  
2884 014212 005037 177772 6$: CLR @#PIRQ ;  
2885 014216 104166 ERROR 166 ;EITHER TMCB E51(9) OR E55(10-11) OR  
2886 ;E62 IS BAD OR TMCA INH BELOW BR6 IS  
2887 ;STUCK LOW  
2888 ;TRY PIR 7  
2889 014220 012737 014242 000240 7$: MOV #8$,@#PIRQVEC ;SETUP PIRQ VECTOR
```

```
2890 014226 052737 100000 177772      BIS      #BIT15,@PIRQ      ;SET LEVEL 7
2891 014234 005037 177772      CLR      @PIRQ          ;CLEAR LEVEL 7
2892 014240 000403                BR       9$            ;BRANCH IF NO INTERRUPT
2893 014242 005037 177772      8$:     CLR      @PIRQ          ;
2894 014246 104167                ERROR   167           ;TMCA ABOVE BR7 MIGHT BE STUCK LOW
2895 014250 104170                9$:     ERROR   170           ;TMCE BRQ CLOCK MIGHT BE STUCK LOW
2896 014252 005037 177772      1$:     CLR      @PIRQ          ;CLEAR LEVEL 1
2897 014256 012737 032654 000004    MOV      #CPUSPUR,@ERRVEC ;RESTORE ERR VEC
2898 014264 012737 036266 000024    MOV      #SPWRDN,@PWRVEC  ;RESTORE THE POWER VECTOR
2899 014272 000421                BR       TST24         ;;GO TO NEXT TEST
2900 014274 005037 177772      10$:    CLR      @PIRQ          ;CLEAR LEVEL 1
2901 014300 104171                ERROR   171           ;BEN 13 FAILED
2902 014302 005037 177772      11$:    CLR      @PIRQ          ;
2903 014306 012737 032654 000004    MOV      #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
2904 014314 012706 001100                MOV      #STACK,SP     ;RESTORE THE SP
2905 014320 005037 177766                CLR      @PCPUERR      ;CLEAR ERROR REG
2906 014324 104172                ERROR   172           ;BEN 13 FAILED OR TRAP VECTOR FAILED
2907 014326 005037 177772      12$:    CLR      @PIRQ          ;CLEAR LEVEL 1
2908 014332 104377                ERROR   377           ;EITHER TMCB E53 IS NOT GOING HIGH
2909 014334 000454                454                 ;OR TMCB PIRQ DOES NOT GO LOW. (BEN 13 FAILURE)
```

```
*****
*TEST 24      PIR LEVEL 2 INTERRUPT
*
*      IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
*      THIS WILL ONLY HAPPEN IF TMCB E63(2) IS BAD.
*
*      IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(2)
*      IS BAD OR TMCB HONOR PIR2 IS BEING HELD HIGH.
*****
```

```
2910
2911
2912
2913
2914
2915
2916
2917
2918
2919 014336 000004                TST24:  SCOPE
2920 014340 012767 014474 164722    MOV      #TST25,NEXTTST ;SAVE ADDRESS OF NEXT TEST
2921 014346 012706 001100                MOV      #STACK,SP     ;SETUP THE SP
2922 014352 012737 014406 000000    MOV      #1$,@#0       ;SETUP LOC. 0 TO CATCH BEN 13 FAILURE
2923 014360 012737 014422 000240    MOV      #2$,@PIRQVEC  ;SETUP PIRQ VECTOR
2924 014366 000231                SPL      1              ;SET CPU PRIORITY TO 1
2925 014370 052737 002000 177772    BIS      #BIT10,@PIRQ   ;SET LEVEL 2
2926 014376 005037 177772                CLR      @PIRQ          ;CLEAR LEVEL 2
2927 014402 104174                ERROR   174           ;PIR 2 DID NOT INTERRUPT
2928 014404 000406                BR       2$            ;
2929 014406 005037 177772      1$:     CLR      @PIRQ          ;CLEAR LEVEL 2
2930 014412 012737 032654 000004    MOV      #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
2931 014420 104175                ERROR   175           ;BEN 13 FAILED
2932                ;ENSURE PIR09 & 10 NOT SHORTED
2933 014422 012767 014430 164460    2$:     MOV      #64$, $LPERR ;SETUP ERROR LOOP
2934 014430 005037 177772      64$:    CLR      @PIRQ          ;CLEAR LEVEL 2
2935 014434 012737 014460 000240    MOV      #3$,@PIRQVEC  ;SETUP VECTOR
2936 014442 000231                SPL      1              ;SET LEVEL 1
2937 014444 052737 001000 177772    BIS      #BIT9,@PIRQ    ;SET PIR 1
2938 014452 005037 177772                CLR      @PIRQ          ;CLEAR PIR 1
2939 014456 000406                BR       TST25         ;;GO TO NEXT TEST
2940 014460 013767 177772 164530    3$:     MOV      @PIRQ,$EPIRQ   ;SAVE PIRQ FOR TYPEOUT
2941 014466 005037 177772                CLR      @PIRQ          ;CLEAR LEVEL 1
2942 014472 104176                ERROR   176           ;PIR09 & 10 SHORTED
```

```
*****
*TEST 25      PIR LEVEL 3 INTERRUPT
*
```

2946
2947
2948
2949
2950
2951
2952 014474 000004
2953 014476 012767 014632 164564
2954 014504 012706 001076
2955 014510 012737 014560 000240
2956 014516 012737 014544 000000
2957 014524 000232
2958 014526 052737 004000 177772
2959 014534 005037 177772
2960 014540 104177
2961 014542 000406
2962 014544 005037 177772
2963 014550 012737 032654 000004
2964 014556 104200
2965 014560 012767 014566 164322
2966 014566 005037 177772
2967 014572 012737 014616 000240
2968 014600 000232
2969 014602 052737 002000 177772
2970 014610 005037 177772
2971 014614 000406
2972 014616 013767 177772 164372
2973 014624 005037 177772
2974 014630 104201

```

: * 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.
: *****
TST25: SCOPE
MOV #TST26,NEXTTST ;SAVE ADDRESS OF NEXT TEST
MOV #1076,SP ;INITIALIZE THE SP
MOV #1$,@PIRQVEC ;SETUP PIRQ VECTOR
MOV #2$,@#0 ;SETUP LOCATION 0
SPL 2 ;SET CPU AT LEVEL 2
BIS #BIT11,@PIRQ ;SET LEVEL 3 PIR
CLR @PIRQ ;CLEAR LEVEL 3
ERROR 177 ;INTERRUPT FAILED
BR 1$
2$: CLR @PIRQ ;CLEAR LEVEL 3
MOV #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
ERROR 200 ;BEN 13 FAILED
1$: MOV #64$, $LPERR ;SETUP ERROR LOOP
64$: CLR @PIRQ ;CLEAR LEVEL 3
MOV #3$,@PIRQVEC ;SETUP PIRQ VECTOR
SPL 2 ;SET CPU PRIORITY AT 2
BIS #BIT10,@PIRQ ;ENABLE PIR2
CLR @PIRQ ;CLEAR LEVEL 2
BR TST26 ;GO TO NEXT TEST
3$: MOV @PIRQ,$EPIRQ ;SAVE PIRQ
CLR @PIRQ ;CLEAR IT
ERROR 201 ;LEVEL 2 INTERRUPT WHEN CPU LEVEL
;2 ENABLED.
: *****
: *TEST 26 PIR LEVEL 4 INTERRUPT
: *
: * IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
: * THIS WILL ONLY HAPPEN IF TMCB E63(5) IS BAD.
: *
: * IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(5)
: * IS BAD OR TMCA HONOR PIR 4 IS BEING HELD HIGH.
: *****
TST26: SCOPE
MOV #TST27,NEXTTST ;SAVE ADDRESS OF NEXT TEST
MOV #STACK,SP ;INITIALIZE THE SP
MOV #1$,@PIRQVEC ;SETUP PIRQ VEC
MOV #2$,@#0 ;SETUP LOCATION 0
SPL 3 ;SET CPU AT 3
BIS #BIT12,@PIRQ ;ENABLE PIR 4
MOV #CPUSPUR,@ERRVEC
ERROR 202 ;INTERRUPT DID NOT OCCUR
BR 1$
2$: CLR @PIRQ ;CLEAR LEVEL 4
MOV #CPUSPUR,@ERRVEC
ERROR 203 ;BEN 13 FAILED
1$: MOV #64$, $LPERR ;SETUP ERROR LOOP
64$: CLR @PIRQ ;CLEAR LEVEL 4
MOV #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
MOV #3$,@PIRQVEC ;SETUP PIRQ VECTOR

```

2975
2976
2977
2978
2979
2980
2981
2982
2983
2984
2985 014632 000004
2986 014634 012767 015002 164426
2987 014642 012706 001100
2988 014646 012737 014720 000240
2989 014654 012737 014704 000000
2990 014662 000233
2991 014664 052737 010000 177772
2992 014672 012737 032654 000004
2993 014700 104202
2994 014702 000406
2995 014704 005037 177772
2996 014710 012737 032654 000004
2997 014716 104203
2998 014720 012767 014726 164162
2999 014726 005037 177772
3000 014732 012737 032654 000004
3001 014740 012737 014766 000240

```
3002 014746 000233          SPL      3          ;SET CPU AT LEVEL 3
3003 014750 052737 004000 177772  BIS      #BIT11,@PIRQ ;SET LEVEL 3
3004 014756 005037 177772          CLR      @PIRQ      ;CLEAR LEVEL 3
3005 014762 000237          SPL      7
3006 014764 000406          BR       TST27      ;;GO TO NEXT TEST
3007 014766 013767 177772 164222 3$:  MOV      @PIRQ,$EPIRQ ;SAVE PIRQ
3008 014774 005037 177772          CLR      @PIRQ
3009 015000 104204          ERROR    204       ;LEVEL 3 INTERRUPT WHEN CPU LEVEL 3 ENABLED
3010
3011          ;*****
3012          ;*TEST 27      PIR LEVEL 5 INTERRUPT
3013          ;*
3014          ;*      IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
3015          ;*      THIS WILL ONLY HAPPEN IF TMCB E63(11) IS BAD.
3016          ;*
3017          ;*      IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(11)
3018          ;*      IS BAD OR TMCA HONOR PIR 5 IS BEING HELD HIGH.
3019          ;*****
3019 015002 000004          TST27: SCOPE
3020 015004 012767 015156 164256  MOV      #TST30,NEXTTST ;SAVE ADDRESS OF NEXT TEST
3021 015012 012706 001100          MOV      #STACK,SP    ;INITIALIZE THE SP
3022 015016 012737 015074 000240  MOV      #1$,@PIRQVEC ;SETUP THE PIRQ VECTOR
3023 015024 012737 015060 000000  MOV      #2$,@0       ;SETUP LOCATION 0
3024 015032 000234          SPL      4          ;SET CPU ST LEVEL 4
3025 015034 052737 020000 177772  BIS      #BIT13,@PIRQ ;SET LEVEL 5
3026 015042 005037 177772          CLR      @PIRQ      ;CLEAR LEVEL 5
3027 015046 012737 032654 000004  MOV      #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
3028 015054 104205          ERROR    205       ;INTERRUPT DID NOT OCCUR
3029 015056 000406          BR       1$
3030 015060 005037 177772          CLR      @PIRQ      ;CLEAR LEVEL 5
3031 015064 012737 032654 000004  MOV      #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
3032 015072 104206          ERROR    206       ;BEN 13 FAILED
3033 015074 012767 015102 164006 1$:  MOV      #64$, $LPERR ;SETUP ERROR LOOP
3034 015102 012737 032654 000004 64$: MOV      #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
3035 015110 005037 177772          CLR      @PIRQ      ;CLEAR LEVEL 5
3036 015114 012737 015142 000240  MOV      #3$,@PIRQVEC ;SETUP PIRQ VECTOR
3037 015122 000234          SPL      4          ;SET CPU AT LEVEL 4
3038 015124 012737 010000 177772  MOV      #BIT12,@PIRQ ;SET LEVEL 4
3039 015132 005037 177772          CLR      @PIRQ      ;CLEAR LEVEL 4
3040 015136 000237          SPL      7
3041 015140 000406          BR       TST30      ;;GO TO NEXT TEST
3042 015142 013767 177772 164046 3$:  MOV      @PIRQ,$EPIRQ ;SAVE PIRQ
3043 015150 005037 177772          CLR      @PIRQ
3044 015154 104207          ERROR    207       ;LEVEL 4 INT. WHEN CPU LEVEL 4 ENABLED
3045
3046          ;*****
3047          ;*TEST 30      PIR LEVEL 6 INTERRUPT
3048          ;*
3049          ;*      IF BEN13 FAILS EXECUTION WILL GO TO BRK.20.
3050          ;*      THIS WILL ONLY HAPPEN IF EITHER TMCB E63(12)
3051          ;*      IS BAD, OR E61(1) IS BAD.
3052          ;*
3053          ;*      IF THE INTERRUPT DOES NOT OCCUR LEVEL 7 IS TRYED, TO TRY
3054          ;*      AND ISOLATE THE FAILURE BEFORE TMCB E55(9-8).
3055          ;*****
3056 015156 000004          TST30: SCOPE
3057 015160 012767 015356 164102  MOV      #TST31,NEXTTST ;SAVE ADDRESS OF NEXT TEST
```

```
3058 015166 012706 001100      MOV      #STACK,SP      ;INITIALIZE THE SP
3059 015172 012737 015274 000240  MOV      #1$,@PIRQVEC    ;SETUP THE PIRQ VECTOR
3060 015200 012737 015260 000000  MOV      #2$,@#0        ;SETUP LOCATION 0
3061 015206 000235          SPL      5              ;SET THE CPU AT LEVEL 5
3062 015210 052737 040000 177772  BIS      #BIT14,@PIRQ    ;SET LEVEL 6
3063 015216 012737 032654 000004  MOV      #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
3064 015224 012737 015250 000240  MOV      #3$,@PIRQVEC    ;SETUP THE PIRQ VECTOR
3065 015232 012737 100000 177772  MOV      #BIT15,@PIRQ    ;SET LEVEL 7
3066 015240 005037 177772          CLR      @PIRQ          ;CLEAR LEVEL 7
3067 015244 104210          ERROR   210           ;FAILURE IS AFTER TMCB E70
3068 015246 000412          BR      1$             ;
3069 015250 005037 177772      3$: CLR      @PIRQ          ;CLEAR LEVEL 7
3070 015254 104211          ERROR   211           ;FAILURE IS IN TMCB E70 OR BEFORE
3071 015256 000406          BR      1$             ;
3072 015260 012737 032654 000004  2$: MOV      #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
3073 015266 005037 177772          CLR      @PIRQ          ;CLEAR LEVEL 6
3074 015272 104212          ERROR   212           ;BEN 13 FAILED
3075 015274 012767 015302 163606  1$: MOV      #64$,SLPERR   ;SETUP ERROR LOOP
3076 015302 005037 177772      64$: CLR      @PIRQ          ;CLEAR LEVEL 6
3077 015306 012737 032654 000004  MOV      #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
3078 015314 012737 015342 000240  MOV      #4$,@PIRQVEC    ;SETUP PIRQ VECTOR
3079 015322 000235          SPL      5              ;SET PRIORITY AT 5
3080 015324 012737 020000 177772  MOV      #BIT13,@PIRQ    ;SET LEVEL 5
3081 015332 005037 177772          CLR      @PIRQ          ;CLEAR LEVEL 5
3082 015336 000237          SPL      7              ;
3083 015340 000406          BR      TST31          ;;GO TO NEXT TEST
3084 015342 013767 177772 163646  4$: MOV      @PIRQ,$EPIRQ   ;SAVE PIRQ
3085 015350 005037 177772          CLR      @PIRQ          ;CLEAR LEVEL 5
3086 015354 104213          ERROR   213           ;LEVEL 5 INTERRUPT WHEN CPU 5 ENABLED
3087          ;*****
3088          ;*TEST 31      PIR LEVEL 7 INTERRUPT
3089          ;*
3090          ;*      IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
3091          ;*      THIS WILL ONLY HAPPEN IF TMCB E63(6) IS BAD.
3092          ;*
3093          ;*      IF THE INTERRUPT DOES NOT OCCUR THEN EITHER TMCB E70(6)
3094          ;*      IS BAD OR TMCA HONOR PIR7 IS BEING HELD HIGH.
3095          ;*****
3096          ;TST31: SCOPE
3097 015356 000064          MOV      #TST32,NEXTTST ;SAVE ADDRESS OF NEXT TEST
3098 015360 012767 015530 163702  MOV      #STACK,SP      ;INITIALIZE THE SP
3099 015366 012706 001100      MOV      #1$,@PIRQVEC    ;SETUP THE PIRQ VECTOR
3100 015400 012737 015434 000000  MOV      #2$,@#0        ;SETUP LOCATION 0
3101 015406 000236          SPL      6              ;SET PRIORITY AT LEVEL 6
3102 015410 052737 100000 177772  BIS      #BIT15,@PIRQ    ;SET LEVEL 7
3103 015416 005037 177772          CLR      @PIRQ          ;CLEAR LEVEL 7
3104 015422 012737 032654 000004  MOV      #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
3105 015430 104214          ERROR   214           ;LEVEL 7 DID NOT INTERRUPT
3106 015432 000406          BR      1$             ;
3107 015434 005037 177772      2$: CLR      @PIRQ          ;CLEAR LEVEL 7
3108 015440 012737 032654 000004  MOV      #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
3109 015446 104215          ERROR   215           ;BEN 13 FAILED
3110 015450 012767 015456 163432  1$: MOV      #64$,SLPERR   ;SETUP ERROR LOOP
3111 015456 012737 032654 000004  64$: MOV      #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
3112 015464 012737 015514 000240  MOV      #3$,@PIRQVEC    ;SETUP PIRQ VECTOR
3113 015472 005037 177772          CLR      @PIRQ          ;CLEAR LEVEL 7
```

```
3114 015476 000236          SPL      6          ;SET LEVEL 6 IN CPU
3115 015500 012737 040000 177772  MOV      #BIT14,@PIRQ ;SET PIR 6
3116 015506 005037 177772          CLR      @PIRQ      ;CLEAR PIR 6
3117 015512 000406          BR       TST32      ;;GO TO NEXT TEST
3118 015514 013767 177772 163474 3$:  MOV      @PIRQ,$EPIRQ
3119 015522 005037 177772          CLR      @PIRQ      ;CLEAR LEVEL 6
3120 015526 04216          ERROR    216        ;LEVEL 6 INT. WHEN CPU AT 6
3121
3122
3123
3124
3125
3126
3127
3128
3129
3130
3131
3132
3133
3134 015530 000004          TST32: SCOPE
3135 015532 012767 016650 163530  MOV      #TST33,NEXTTST ;SAVE ADDRESS OF NEXT TEST
3136 015540 012737 015632 000004  MOV      #2$,@ERRVEC    ;SETUP TIMEOUT VECTOR
3137 015546 032737 000020 177776  BIT      #BIT4,@PSW     ;IS T BIT ON?
3138 015554 001404          BEQ     19$          ;BRANCH IF NO
3139 015556 012737 000360 000006  MOV      #360,@ERRVEC+2 ;SETUP ERROR VEC FOR T BIT
3140 015564 000403          BR     17$
3141 015566 012737 000340 000006 19$:  MOV      #PR7,@ERRVEC+2 ;SETUP ERROR VEC WITHOUT T BIT
3142 015574 012767 015602 163306 17$:  MOV      #12$, $LPERR   ;SETUP ERROR LOOP
3143 015602 012706 001100 12$:  MOV      #STACK,SP     ;INITIALIZE THE SP
3144
3145
3146 015606 013700 160000          ;EXECUTE A TIMEOUT ON A DATI
3147
3148 015612 032737 000020 177766  MOV      @#160000,R0    ;EXECUTE TIMEOUT ON DATI
3149 015620 001002          ;FAILURE-DID NOT TIMEOUT OR AERF DID NOT GO LOW
3150 015622 104217          BIT      #BIT4,@CPUERR ;DID TIMEOUT FLAG SET?
3151 015624 000423          BNE     1$          ;BRANCH IF YES
3152 015626 104220          ERROR    217        ;DID NOT TIMEOUT
3153 015630 000421          BR     7$
3154 015632 022716 160000 1$:  ERROR    220        ;BEN 13 FAILED
3155 015636 001003          BR     7$
3156 015640 104377          ;DID 160000 GET STACKED?
3157
3158 015642 000455          ;BRANCH IF NO
3159 015644 000413          ;EITHER PS RESTORE STUCK HIGH
3160 015646 012737 015674 000004 16$:  MOV      #7$,@ERRVEC    ;SETUP TIMEOUT VECTOR
3161 015654 012767 015662 163226  MOV      #13$, $LPERR   ;SETUP ERROR LOOP
3162 015662 012706 001100 13$:  MOV      #STACK,SP     ;INITIALIZE THE SP
3163
3164
3165 015666 010037 160000          ;EXECUTE A TIMEOUT ON A DATO
3166 015672 104221          MOV      R0,@#160000   ;EXECUTE TIMEOUT ON DATO
3167
3168          ERROR    221        ;DID NOT TIMEOUT
3169 015674 000237          ;PRIORITY CLEAR ON ABORT
          ; ENSURES PIR VECTOR DOES NOT COME IN ON ABORT
          7$:  SPL      7          ;ENSURE CPU AT LEVEL 7
```



```
3170 015676 012767 015746 163204      MOV      #15$, $LPERR      ;SETUP THE ERROR LOOP
3171 015704 012737 015766 000004      MOV      #11$, @WERRVEC   ;SETUP ERRVEC
3172 015712 012737 015774 000240      MOV      #9$, @WPIRQVEC   ;SETUP PIRQ VECTOR
3173 015720 032737 000020 177776      BIT      #BIT4, @WPSW     ;IS T BIT ON?
3174 015726 001404          BEQ      18$              ;BRANCH IF NO
3175 015730 012737 000360 000242      MOV      #360, @WPIRQVEC+2 ;SETUP PIRQ PSW
3176 015736 000403          BR       15$              ;
3177 015740 012737 000340 000006 18$:  MOV      #PR7, @WERRVEC+2 ;SETUP NEW PSW
3178 015746 012706 001100 15$:  MOV      #STACK, SP       ;INITIALIZE THE SP
3179 015752 052737 100000 177772      BIS      #BIT15, @WPIRQ   ;SET PIR LEVEL 7
3180 015760 000236          SPL      6              ;
3181 015762 005237 160000          INC      @W160000        ;EXECUTE REFERENCE TO BAD ADDRESS
3182 015766 005037 177772 11$:  CLR      @WPIRQ           ;CLEAR PIRQ REGISTER
3183 015772 000404          BR       10$              ;SKIP OVER ERROR CALL
3184 015774 005037 177772 9$:   CLR      @WPIRQ           ;CLEAR PIRQ REG
3185 016000 104377          ERROR   377             ;TMCC PRIORITY CLEAR DID NOT GO LOW
3186 016002 000435          435
3187 016004 012737 032654 000004 10$:  MOV      #CPUSPUR, @WERRVEC ;RESTORE ERRVEC
3188          ;CPU ERROR REGISTER BIT 4 TEST
3189 016012 022737 000020 177766      CMP      #BIT4, @WCPUERR  ;DID CPU ERROR REG TIMEOUT BIT SET?
3190 016020 001407          BEQ      5$              ;BRANCH IF YES
3191 016022 013767 177766 163124      MOV      @WCPUERR, $REGO  ;SAVE REG FOR TYPEOUT
3192 016030 012767 000020 163124      MOV      #BIT4, $TMP0     ;SAVE EXPECTED VALUE
3193 016036 104255          ERROR   255             ;CPU ERROR REG FAILED TO SET
3194 016040 005037 177766 5$:   CLR      @WCPUERR         ;CLEAR OUT BIT 4
3195 016044 022737 000000 177766      CMP      #0, @WCPUERR    ;DID REGISTER CLEAR?
3196 016052 001401          BEQ      PDINTE          ;BRANCH IF YES
3197 016054 104256          ERROR   256             ;CPU ERROR REG DOES NOT CLEAR
3198          ;*****
3199          ;SBTTL PERIPHERAL DETERMINATOR & INTERRUPT ENABLE ROUTINES
3200          ;* THIS SECTION OF CODE TRYS TO FIND A DEVICE ON BR5 AND BR6.
3201          ;* WHEN IT FINDS A DEVICE IT PUTS THE ADDRESS OF THAT DEVICES
3202          ;* SUBROUTINE IN A LOCATION.
3203          ;*
3204          ;* THE CODE TO INITIATE AN INTERRUPT SEQUENCE ON CERTAIN
3205          ;* DEVICES IS ALSO HERE. WHEN A TEST REQUIRES AN INTERRUPT ON
3206          ;* A CERTAIN LEVEL IT LOOKS AT INTERX TO DETERMINE IF A
3207          ;* DEVICE IS AVAILABLE AND IF SO DOES A JSR TO THAT DEVICES
3208          ;* INTERRUPT ENABLE ROUTINE.
3209          ;*****
3210 016056          PDINTE:
3211 016056 012767 016650 163110      MOV      #TST33, $ESCAPE  ;SAVE START ADDRESS OF NEXT TEST
3212 016064 012767 016650 163176      MOV      #TST33, NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3213 016072 005767 163002          TST      $PASS           ;IS THIS FIRST PASS?
3214 016076 001004          BNE     1$              ;BRANCH IF NO
3215 016100 032737 040000 177570      BIT      #SW14, @W$WR     ;IS SWITCH 14 ON?
3216 016106 001402          BEQ     2$              ;BRANCH IF NO
3217 016110 000 77 163154 1$:   JMP      @NEXTTST         ;GO TO NEXT TEST
3218 016114 012 06 001100 2$:   MOV      #STACK, SP       ;INITIALIZE THE SP
3219 016120 012737 016134 000004      MOV      #3$, @WERRVEC   ;SETUP ERROR VECTOR
3220 016126 005777 163106          TST      @RSCS1         ;IS RS AVAILABLE?
3221 016132 000430          BR       6$              ;YES
3222 016134 012737 016150 000004 3$:  MOV      #4$, @WERRVEC   ;SETUP ERROR VECTOR
3223 016142 005777 163076          TST      @RPCS1         ;IS RP AVAILABLE?
3224 016146 000431          BR       7$              ;YES
3225 016150 012737 016164 000004 4$:  MOV      #5$, @WERRVEC   ;SETUP ERROR VECTOR
```

```

3226 016156 005777 163072          TST    @TMCS1          ;IS TM AVAILABLE?
3227 016162 000432                   BR     8$              ;YES
3228 016164 012737 016274 000004 5$:   MOV    #10$,@#ERRVEC  ;SETUP ERROR VECTOR
3229 016172 005777 163052          TST    @RKCS1         ;IS RK AVAILABLE?
3230 016176 016767 163046 163024   MOV    RKCS1,INT5ST   ;YES,SAVE RK STATUS
3231 016204 016767 163042 163014   MOV    RKVEC,INT5VEC  ;SAVE RK VECTOR
3232 016212 000424                   BR     9$              ;EXIT
3233 016214 016767 163020 163006 6$:   MOV    RSCS1,INT5ST   ;SAVE RS STATUS
3234 016222 016767 163014 162776   MOV    RSVEC,INT5VEC  ;SAVE RS VECTOR
3235 016230 000415                   BR     9$              ;EXIT
3236 016232 016767 163006 162770 7$:   MOV    RPCS1,INT5ST   ;SAVE RP STATUS
3237 016240 016767 163002 162760   MOV    RPVEC,INT5VEC  ;SAVE RP VECTOR
3238 016246 000406                   BR     9$              ;EXIT
3239 016250 016767 163000 162752 8$:   MOV    TMCS1,INT5ST   ;SAVE TM STATUS
3240 016256 016767 162774 162742   MOV    TMVEC,INT5VEC  ;SAVE TM VECTOR
3241 016264 012767 016464 162732 9$:   MOV    #INT5SU,INTERS ;SAVE ADDRESS OF INTER 5 SUBROUTINE
3242 016272 000406                   BR     LEVE6           ;GO CHECK LEVEL 6
3243 016274 032737 000440 177570 10$:  BIT    #440,@#SWR     ;SWITCH 8 OR 5 ON?
3244 016302 001002                   BNE   LEVE6           ;BRANCH IF YES
3245 016304 104400 072420          TYPE    ,EM710
3246
3247          ;FIND OUT WHAT IS AVAILABLE ON LEVEL 6
3248 016310 012737 016360 000004  LEVE6: MOV    #1$,@#ERRVEC ;SETUP LOCATION 4
3249 016316 005777 162736          TST    @LKSTAT        ;IS LINE CLOCK AVAILABLE?
3250 016322 012767 016530 162702   MOV    #SKW11L,INTER6 ;PUT ADDRESS OF KW11-L SUBROUTINE IN STORAGE
3251 016330 016767 162726 162676   MOV    LKVEC,INT6VEC  ;SAVE BR 6 INTERR VEC
3252 016336 016767 162716 162672   MOV    LKSTAT,INT6ST  ;SAVE ADDRESS OF BR 6 STATUS
3253 016344 012737 032654 000 4    MOV    #CPUSPUR,@#ERRVEC
3254 016352 005037 177766          CLR    @#CPUERR       ;ENSURE TIMEOUT BIT CLEAR
3255 016356 000534                   BR     TST33           ;:PERIPHERAL DETERMINATOR FINISHED
3256 016360 104400 073107          1$:   TYPE    ,EM725
3257
3258          ;LINE CLOCK NOT AVAILABLE, SEE IF PROGRAMMABLE CLOCK AVAILABLE
3259 016364 012737 016434 000004   MOV    #2$,@#ERRVEC  ;SETUP LOCATION 4
3260 016372 005777 162666          TST    @PLKSTAT       ;IS PROGRAMMABLE AVAILABLE?
3261 016376 012767 016574 162626   MOV    #SKW11P,INTER6 ;YES, PUT ADDRESS OF KW11-P SUBROUTINE IN STORAGE
3262 016404 016767 162656 162622   MOV    PLKVEC,INT6VEC ;SAVE BR 6 INTERR VEC
3263 016412 016767 162646 162616   MOV    PLKSTAT,INT6ST ;SAVE ADDRESS OF BR 6 STATUS
3264 016420 005037 177766          CLR    @#CPUERR       ;ENSURE TIMEOUT BIT CLEAR
3265 016424 012737 032654 000004   MOV    #CPUSPUR,@#ERRVEC ;RESTORE ERROR VECTOR
3266 016432 000506                   BR     TST33           ;:PERIPHERAL DETERMINATOR FINISHED
3267 016434 012737 032654 000004 2$:   MOV    #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
3268 016442 005037 177766          CLR    @#CPUERR       ;ENSURE TIMEOUT BIT CLEAR
3269 016446 032737 000500 177570   BIT    #500,@#SWR     ;SWITCH 6 OR 8 ON?
3270 016454 001002                   BNE   3$              ;BRANCH IF YES
3271 016456 104400 072440          TYPE    ,EM711       ;TYPE MESSAGE(NO DEVICE AVAILABLE)
3272 016462
3273 016462 000472                   BR     TST33           ;:PERIPHERAL DETERMINATOR FINISHED
3274
3275          ;THIS CODE SETS UP THE DEVICE ON LEVEL 5 TO INTERRUPT
3276          ;AND IS CALLED BY A JSR PC,@INTERS5
3276 016464 011600          INT5SU: MOV    (SP),R0    ;SAVE RETURN PC
3277 016466 012777 016520 162532   MOV    #ENDBR5,@INTSVEC ;SETUP LEVEL 5 VECTOR
3278 016474 005001                   CLR    R1              ;SETUP WAIT COUNTER
3279 016476 012777 000311 162524   MOV    #311,@INT5ST   ;SET LEVEL 5 INTERRUPT
3280 016504 005201          2$:   INC    R1              ;WAIT FOR
3281 016506 001376                   BNE   2$              ;INTERRUPT

```

```
3282 016510 005077 162514          CLR    @INT5ST      ;CLEAR INTERRUPT FLAG
3283 016514 062700 000002          ADD    #2,R0        ;ADJUST RETURN PC
3284 016520 005077 162504          ENDBR5: CLR @INT5ST ;CLEAR LEVEL 5 STATUS
3285 016524 010046                    MOV    R0,-(SP)     ;PUT RETURN PC ON STACK
3286 016526 000207                    RTS     PC           ;RETURN
3287
3288          ;THIS CODE SETS UP THE KW11-L TO INTERRUPT AND IS CALLED BY A JSR PC,@INTER6
3289 016530 011600          $KW11L: MOV (SP),R0 ;SAVE THE RETURN PC
3290 016532 012777 016566 162474      MOV    #1$,@INT6VEC ;SETUP INTERRUPT VECTOR
3291 016540 005001          CLR    R1           ;SETUP COUNTER
3292 016542 012777 000100 162466      MOV    #BIT6,@INT6ST ;SET INTERRUPT ENABLE BIT & CLR MONITOR BIT
3293 016550 005201          2$: INC  R1         ;WAIT FOR
3294 016552 001376          BNE   2$           ;INTERRUPT
3295 016554 005077 162456          CLR    @INT6ST     ;CLEAR INTERRUPT BIT
3296 016560 062700 000002          ADD    #2,R0        ;ADJUST RETURN PC
3297 016564 000427          BR    $ENDBR6      ;RETURN
3298 016566 005077 162444          1$: CLR  @INT6ST   ;CLEAR INTERRUPT FLAG
3299 016572 000424          BR    $ENDBR6      ;RETURN
3300
3301          ;THIS CODE SETS UP THE KW11-P TO INTERRUPT AND IS CALLED BY A JSR PC,@INTER6
3302 016574 011600          $KW11P: MOV (SP),R0 ;SAVE THE RETURN PC
3303 016576 012777 016640 162430      MOV    #1$,@INT6VEC ;SETUP THE INTERRUPT VECTOR
3304 016604 012737 000001 172544      MOV    #1,@PLKC     ;SET THE COUNTER FOR ONE COUNT
3305 016612 005001          CLR    R1           ;SETUP THE WAIT COUNTER
3306 016614 012777 000105 162414      MOV    #105,@INT6ST ;START COUNTER
3307 016622 005201          2$: INC  R1         ;WAIT FOR
3308 016624 001376          BNE   2$           ;INTERRUPT
3309 016626 005077 162404          CLR    @INT6ST     ;CLEAR INTERRUPT BIT
3310 016632 062700 000002          ADD    #2,R0        ;ADJUST R0 FOR RETURN
3311 016636 000402          BR    $ENDBR6      ;RETURN
3312 016640 005077 162372          1$: CLR  @INT6ST   ;CLEAR INTERRUPT FLAG
3313 016644 010046          $ENDBR6:MOV R0,-(SP) ;PUT RETURN PC ON STACK.
3314 016646 000207          RTS     PC           ;RETURN
3315
3316          ;*****
3317          ;*TEST 33          BR LEVEL 4 INTERRUPT
3318          ;*
3319          ;*      BEN 13 SHOULD NOT FAIL.
3320          ;*      IF THE INTERRUPT DOESN'T OCCUR AN ATTEMPT IS MADE TO
3321          ;*      ISOLATE THE FAILURE.
3322          ;*****
3323 016650 000004          TST33: SCOPE
3324 016652 012767 017144 162410      MOV    #TST34,NEXTTST ;SAVE ADDRESS OF NEXT TEST
3325 016660 012767 003706 162216      MOV    #D1990,$ICNT  ;ADJUST ITERATION COUNT
3326 016666 012767 016674 162212      MOV    #14$, $LPADR  ;SETUP LOOP ADR
3327 016674 032737 000400 177570      14$: BIT  #SW8,@SWR   ;SWITCH 8 ON?
3328 016702 001012          BNE   8$           ;BRANCH IF YES
3329 016704 032737 000020 177570      BIT  #SW4,@SWR     ;IS SWITCH 4 ON?
3330 016712 001406          BEQ  8$           ;BRANCH IF NO
3331 016714 005767 162160          TST  $PASS        ;IS THIS FIRST PASS?
3332 016720 001002          BNE  9$           ;BRANCH IF NO
3333 016722 104400 072460          TYPE  ,EM712      ;TYPE MESSAGE (SKIPPING TEST)
3334 016726          9$:
3335 016726 000506          BR    TST34       ;GO TO NEXT TEST
3336 016730 012706 001100          8$: MOV  #STACK,SP   ;INITIALIZE THE SP
3337 016734 012737 017010 000064      MOV  #1$,@INTPVEC  ;SETUP THE TERMINAL INTERRUPT VECTOR
```

```
3338 016742 032737 000020 177776 BIT #BIT4,@PSW ;IS T BIT ON?
3339 016750 001404 BEQ 10$ ;BRANCH IF NO
3340 016752 012737 000360 000066 MOV #360,@TPVEC+2 ;SETUP TPVEC PSW
3341 016760 000403 BR 11$
3342 016762 012737 000340 000066 10$: MOV #PR7,@TPVEC+2 ;PUT PRIORITY 7 IN NEW PSW
3343 016770 000237 11$: SPL 7 ;SET CPU AT LEVEL 7
3344 016772 005000 CLR R0 ;SETUP R0
3345 016774 052777 000100 162140 BIS #100,@STPS ;GET INTERRUPT ON BR 4
3346 017002 005200 2$: INC R0 ;WAIT AND SEE
3347 017004 001376 BNE 2$ ;IF INTERRUPT OCCURS
3348 017006 000403 BR 3$ ;NO INTERRUPT
3349 017010 005077 162126 1$: CLR @STPS ;CLEAR INTERRUPT ENABLE
3350 017014 104222 ERROR 222 ;EITHER TMCB PS07(0) IS NOT GETTING TO
3351 ;TMCB E77 OR L77 IS BAD
3352 017016 3$:
3353 017016 012767 017144 162150 MOV #TST34,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
3354 017024 012767 017144 162236 MOV #TST34,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3355 017032 012737 017140 000064 MOV #4,$@TPVEC ;SETUP THE INTERRUPT VECTOR
3356 017040 005000 CLR R0 ;SETUP R0
3357 017042 000233 SPL 3 ;SET CPU AT LEVEL 3
3358 017044 052777 000100 162070 BIS #100,@STPS ;GET A BR 4
3359 017052 005200 5$: INC R0 ;WAIT FOR
3360 017054 001376 BNE 5$ ;INTERRUPT
3361 017056 005077 162060 CLR @STPS ;CLEAR TP INTERRUPT BIT
3362 ;BR 4 INTERRUPT FAILED. IS THERE A BR 6 DEVICE AVAILABLE?
3363 017062 005767 162144 TST INTER6 ;TFST LEVEL 6
3364 017066 001422 BEQ 7$
3365 017070 016700 162140 MOV INT6VEC,R0 ;GET ADDR OF INTER 6 VECTOR
3366 017074 062700 000002 ADD #2,R0 ;ADJUST TO PSW VEC
3367 017100 032737 000020 177776 BIT #BIT4,@PSW ;IS T BIT ON?
3368 017106 001403 BEQ 12$ ;BRANCH IF NO
3369 017110 012710 000360 MOV #360,(R0) ;SETUP PSW
3370 017114 000402 BR 13$
3371 017116 012710 000340 12$: MOV #PR7,(R0) ;SETUP PSW NO T BIT
3372 017122 000235 13$: SPL 5 ;SET CPU AT 5
3373 017124 004777 162102 JSR PC,@INTER6 ;EXECUTE INTERRUPT
3374 017130 000402 BR 6$ ;RETURN HERE IF 6 INTERRUPTS
3375 017132 104223 ERROR 223 ;BOTH BR 4 AND BR 6 FAILED
3376 017134 104224 7$: ERROR 224 ;BR 4 FAILED
3377 017136 104225 6$: ERROR 225 ;BR 4 FAILED BUT BR 6 OK
3378 017140 005077 161776 4$: CLR @STPS ;CLEAR PRINTER INTERRUPT FLAG
```

```
3379
3380 *****
3381 *TEST 34 BR LEVEL 5 INTERRUPT
3382 *
3383 * THE ONLY POSSIBLE FAILURE IS THAT TMCA HONOR BR 5
3384 * DOES NOT GO LOW OR TMCB E62(6) IS BAD.
3385 *****
3386 TST34: SCOPE
3387 017144 000004 MOV #TST35,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
3388 017146 012767 017274 162020 MOV #TST35,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3389 017154 012767 017274 162106 BIT #SW8,@SWR ;SWITCH 8 ON?
3390 017162 032737 000400 177570 BNE 1$ ;BRANCH IF YES
3391 017170 001012 BIT #SW5,@SWR ;IS SWITCH 5 UP?
3392 017172 032737 000040 177570 BEQ 1$ ;BRANCH IF NO
3393 017200 001406 TST $PASS ;IS THIS FIRST PASS?
```

```
3394 017206 001002 BNE 3$ ;BRANCH IF NO
3395 017210 104400 072505 TYPE ,EM713 ;TYPE MESSAGE (TEST BEING SKIPPED)
3396 017214 3$:
3397 017214 000427 BR TST35 ;GO TO NEXT TEST
3398 017216 005767 162002 1$: TST INTER5 ;IS THERE A DEVICE AVAILABLE?
3399 017222 001424 BEQ TST35 ;BRANCH IF NO
3400 017224 012706 001100 2$: MOV #STACK,SP ;INITIALIZE THE SP
3401 017230 016700 161772 MOV INT5VEC,R0 ;GET ADDR OF BR 5 VECTOR
3402 017234 062700 000002 ADD #2,R0 ;ADJUST TO PSW ADDR
3403 017240 032737 000020 177776 BIT #BIT4,@#PSW ;IS T BIT ON?
3404 017246 001403 BEQ 5$ ;BRANCH IF NO
3405 017250 012710 000360 MOV #360,(R0) ;PUT NEW PSW IN VECTOR
3406 017254 000402 BR 6$ ;
3407 017256 012710 000340 5$: MOV #PR7,(R0) ;PUT NEW PSW IN VEC NO T BIT
3408 017262 000234 6$: SPL 4 ;SET CPU AT LEVEL 4
3409 017264 004777 161734 JSR PC,@INTER5 ;EXECUTE LEVEL 5 INTERRUPT
3410 017270 000401 BR TST35 ;GO TO NEXT TEST
3411 017272 104226 ERROR 226 ;BR 5 DID NOT INTERRUPT
```

3412
3413
3414
3415
3416
3417
3418

```
*****
*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.
*****
```

```
3419 017274 000004 TST35: SCOPE
3420 017276 012767 017424 161670 MOV #TST36,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
3421 017304 012767 017424 161756 MOV #TST36,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3422 017312 032737 000400 177570 BIT #SW8,@#SWR ;SWITCH 8 ON?
3423 017320 001012 BNE 1$ ;BRANCH IF YES
3424 017322 032737 000100 177570 BIT #SW6,@#SWR ;IS SWITCH 6 UP?
3425 017330 001406 BEQ 1$ ;BRANCH IF NO
3426 017332 005767 161542 TST $PASS ;IS THIS FIRST PASS?
3427 017336 001002 BNE 3$ ;BRANCH IF NO
3428 017340 104400 072532 TYPE ,EM715 ;TYPE MESSAGE (TEST BEING SKIPPED)
3429 017344 3$:
3430 017344 000427 BR TST36 ;GO TO NEXT TEST
3431 017346 005767 161660 1$: TST INTER6 ;IS THERE A DEVICE AVAILABLE?
3432 017352 001424 BEQ TST36 ;BRANCH IF NO
3433 017354 012706 001100 2$: MOV #STACK,SP ;INITIALIZE THE SP
3434 017360 016700 161650 MOV INT6VEC,R0 ;GET ADR OF BR 6 VECTOR
3435 017364 062700 000002 ADD #2,R0 ;ADJUST TO PSW ADDR
3436 017370 032737 000020 177776 BIT #BIT4,@#PSW ;IS T BIT ON?
3437 017376 001403 BEQ 5$ ;BRANCH IF NO
3438 017400 012710 000360 MOV #360,(R0) ;SETUP NEW PSW
3439 017404 000402 BR 6$ ;
3440 017406 012710 000340 5$: MOV #PR7,(R0) ;SETUP NEW PSW
3441 017412 000235 6$: SPL 5 ;SET CPU AT LEVEL 5
3442 017414 004777 161612 JSR PC,@INTER6 ;EXECUTE LEVEL 6 INTERRUPT
3443 017420 000401 BR TST36 ;GO TO NEXT TEST
3444 017422 104227 ERROR 227 ;BR 6 DID NOT INTERRUPT
```

3445
3446
3447
3448
3449

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

```

3450      : *      IF BEN 13 FAILS THE TRAP WILL NOT OCCUR.
3451      : *
3452      : *      IF THE PROCESSOR FAILS TO TRAP EITHER TMCD SL YEL IS
3453      : *      NOT GOING HIGH OR TMCA HONOR SLY IS NOT GOING LOW
3454      : *      OR TMCB E70(3) IS BAD OR BEN13 FAILED.
3455      : *
3456      : *      IF TMCC PRIORITY CLEAR DOES NOT GO LOW THE PROCESSOR WILL HANG
3457      : *      UP IN A RED ZONE TRAP LOOP.
3458      : *
3459      : *      A JSR WITH A BAD SP IS THEN EXECUTED TO ENSURE TMCC
3460      : *      KERNAL R6 GOES HIGH WHEN ENABLED BY 'STACK REFERENCE * KERNAL MODE'.
3461      : *
3462      : *      IF THE TRAP WORKS TESTS WILL BE PERFORMED TO ENSURE ALL THE
3463      : *      APPROPRIATE CONDITIONS DISABLE THE TRAP EXCEPT
3464      : *      THE PRIORITY ARBITRATOR.
3465      : *

```

```

*****
TST36:  SCOPE
3466 017424 000004      MOV      #11$, $LPADR      ; SETUP LP ADR
3467 017426 012767 017472 161452      MOV      #11$, $LPERR     ; SETUP ERROR LOOP
3468 017434 012767 017472 161446      MOV      @#PSW, $TMP3     ; SAVE PSW
3469 017442 013767 177776 161520      BIT      #BIT4, @#PSW    ; IS T BIT ON?
3470 017450 032737 000020 177776      BEQ      11$             ; BRANCH IF NO
3471 017456 001405      MOV      #PR7, -(SP)     ; PUT NEW PSW ON STACK
3472 017460 012746 000340      MOV      #11$, -(SP)    ; PUT RETURN ADR ON STACK
3473 017464 012746 017472      RTT                       ; TURN T BIT OFF
3474 017470 000006      MOV      #PR7, @#ERRVEC+2 ; RESTORE ERR VEC PSW
3475 017472 012737 000340 000006 1$:  MOV      #1$, @#ERRVEC   ; SETUP ERRVEC
3476 017500 012737 017536 000004      MOV      #2$, $LPERR     ; SETUP ERROR LOOP
3477 017506 012767 017514 161374      MOV      #376, SP       ; SETUP THE SP
3478 017514 012706 000376 2$:  MOV      #-1, R0         ; SETUP R0
3479 017520 012700 177777      MOV      R0, (SP)       ; EXECUTE INSTRUCTION UNDER TEST
3480 017524 010016      MOV      #STACK, SP     ; REINITIALIZE THE SP
3481 017526 012706 001100      ERROR   230            ; YELLOW ZONE DID NOT OCCUR
3482 017532 104230      BR      8$
3483 017534 000407      CMP      #-1, @#376     ; DID RED ZONE OCCUR?
3484 017536 022737 177777 000376 1$:  BEQ      8$             ; BRANCH IF NO
3485 017544 001403      MOV      #STACK, SP     ; RESTORE SP
3486 017546 012706 001100      ERROR   254            ; RED ZONE IN YELLOW REGION
3487 017552 104254
3488
3489      : JSR WITH A BAD SP
3490 017554 012737 017614 000004 8$:  MOV      #6$, @#ERRVEC   ; SETUP ERRVEC
3491 017562 012767 017570 161320      MOV      #63$, $LPERR   ; SETUP THE ERROR LOOP
3492 017570 012706 000376 63$:  MOV      #376, SP       ; SETUP THE SP
3493 017574 004767 000000      JSR      PC, 7$        ; EXECUTE INSTRUCTION UNDER TEST
3494 017600 012706 001100 7$:  MOV      #STACK, SP     ; RESET THE SP
3495 017604 012737 032654 000004      MOV      #CPUSPUR, @#ERRVEC ; RESTORE ERRVEC
3496 017612 104232      ERROR   232            ; TMCC KERNAL R6 DID NOT GO HIGH ON JSR
3497
3498      : DISABLE WITH STACK LIMIT REGISTER
3499 017614 012737 017642 000004 6$:  MOV      #3$, @#ERRVEC   ; SETUP ERRVEC
3500 017622 012767 017630 161260      MOV      #62$, $LPERR   ; SETUP ERROR LOOP
3501 017630 012706 000740 62$:  MOV      #740, SP       ; SETUP THE SP
3502 017634 013716 000742      MOV      @#742, (SP)    ; EXECUTE INSTRUCTION UNDER TEST
3503 017640 000406      BR      4$             ; GO TO NEXT SECTION
3504 017642 012737 032654 000004 3$:  MOV      #CPUSPUR, @#ERRVEC ; RESTORE ERRVEC
3505 017650 012706 001100      MOV      #STACK, SP     ; RESTORE THE SP

```

```

3506 017654 104233          ERROR 233          ;PDRC STACK LIMIT DID NOT DISABLE TRAP
3507
3508          ;DISABLE WITH TMCC KERNAL R6
3509          ;DISABLE KERNAL R6 WITH DATI
3510 017656 012737 017702 000004 4$:  MOV  #5$,@#ERRVEC  ;SETUP ERRVEC
3511 017664 012767 017672 161216  MOV  #61$, $LPERR  ;SETUP THE ERROR LOOP
3512 017672 012706 000376 61$:  MOV  #376,SP      ;SETUP THE SP
3513 017676 011606          MOV  (SP),SP      ;EXECUTE INSTRUCTION UNDER TEST
3514 017700 000415          BR   9$          ;GO TO NEXT SECTION
3515 017702 012706 001100 5$:  MOV  #STACK,SP    ;RESTORE THE SP
3516 017706 012737 032654 000004  MOV  #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
3517 017714 104234          ERROR 234        ;TMCC KERNAL R6 DID NOT DISABLE ON DATI
3518 017716 000406          BR   9$
3519
3520          ;USED FOR ERROR LOOP IF BIT 3 IN ERROR REG
3521          ;DOES NOT SET
3522 017720 012706 000376 60$:  MOV  #376,SP      ;SETUP THE SP
3523 017724 012737 017734 000004  MOV  #9$,@#ERRVEC  ;SETUP ERROR VECTOR
3524 017732 005016          CLR  (SP)        ;CAUSE TRAP
3525 017734 012767 017720 161146 9$:  MOV  #60$, $LPERR  ;SETUP ERROR LOOP
3526 017742 012706 001100  MOV  #STACK,SP    ;RESTORE THE SP
3527 017746 022737 000010 177766  CMP  #BIT3,@#CPUERR ;DID YEL ZONE BIT SET?
3528 017754 001407          BEQ  10$        ;BRANCH IF YES
3529 017756 013767 177766 161170  MOV  @#CPUERR,$REGO ;SAVE FOR TYPEOUT
3530 017764 012767 000010 161170  MOV  #BIT3,$TMP0  ;SAVE EXPECTED VALUE

```

```
3531 017772 104257          ERROR 257          ;YEL ZONE BIT DID NOT SET IN CPUERR
3532 017774 012767 020002 161106 10$: MOV #57$, $LPERR ;SETUP ERROR LOOP
3533 020002 005037 177766          57$: CLR @#CPUERR ;CLEAR YEL ZONE BIT
3534 020006 005737 177766          TST @#CPUERR ;DID IT CLEAR?
3535 020012 001401          BEQ TST37 ;:BRANCH IF YES
3536 020014 104260          ERROR 260          ;BIT3 DID NOV CLEAR IN CPUERR
3537
3538 :*****
3539 :*TEST 37 ROM FIELD CHECK OF PC MANIPULATOR STATES
3540 :*
3541 :* THIS TEST EXECUTES THE MACHINE STATES THAT MANIPULATE
3542 :* THE PC TO ENSURE THAT THE PCB ROM FIELD OR DRX ROM FIELD
3543 :* OR SRX ROM FIELD OF THESE STATES IS FUNCTIONAL.
3544 :* THESE STATES ARE S13.00, S45.00, MTP.10, D45.80, D45.90,
3545 :* D45.00, AND D45.01.
3546 :*****
3546 020016 000004          TST37: SCOPE
3547 020020 012767 020050 161060          MOV #18$, $LPADR ;SETUP LOOP ADR
3548 020026 032767 000020 161134          BIT #BIT4, $TMP3 ;WAS T BIT ON?
3549 020034 001405          BEQ 18$ ;BRANCH IF NO
3550 020036 012746 000360          MOV #360, -(SP) ;PUT NEW PSW ON STACK
3551 020042 012746 020050          MOV #18$, -(SP) ;PUT RETURN ADDR ON STACK
3552 020046 000006          RTT ;TURN T BIT ON
3553
3554 020050 012767 020056 161032          ;BIN*SM1*SF7*DM0(S13.00)
3555 020056 011700          18$: MOV #64$, $LPERR ;SETUP ERROR LOOP
3556 020060 020027 020027          64$: MOV (PC), R0 ;EXECUTE INSTRUCTION UNDER TEST
3557 020064 001402          CMP R0, #20027 ;DID TEST WORK
3558 020066 104377          BEQ 1$ ;BRANCH IF YES
3559 020070 000443          ERROR 377 ;STATE S13.00 FAILED
3560 443
3561 020072 012767 020100 161010          ;BIN*SM4*SF7*DM2*DF7 (S45.00)
3562 020100 012767 024727 000000          1$: MOV #63$, $LPERR ;SETUP ERROR LOOP
3563 020106 024727          63$: MOV #24727, 2$ ;PUT INSTRUCTION IN 2$
3564 020110 000240          2$: CMP -(PC), (PC)+ ;EXECUTE INSTRUCTION UNDER TEST
3565 020112 001402          NOP ;USED TO BE SAFE
3566 020114 104377          BEQ 3$ ;BRANCH IF TEST OK
3567 020116 000444          ERROR 377 ;STATE S45.00 FAILED
3568 444
3569 020120 012767 020126 160762          ;MTP*DM2*DF7 (MTP.10)
3570 020126 012706 001100          3$: MOV #62$, $LPERR ;SETUP ERROR LOOP
3571 020132 012746 177777          62$: MOV #STACK, SP ;INITIALIZE THE SP
3572 020136 005067 000002          MOV #-1, -(SP) ;SET 1076 TO ALL ONES
3573 020142 006627          CLR 4$ ;ENSURE 4$ CLEAR
3574 020144 000000          MTP1 (PC)+ ;EXECUTE INSTRUCTION UNDER TEST
3575 020146 022767 177777 177770          4$: .WORD 0
3576 020154 001402          CMP #-1, 4$ ;DID INSTRUCTION WORK?
3577 020156 104377          BEQ 7$ ;BRANCH IF YES
3578 020160 000445          ERROR 377 ;STATE MTP.10 FAILED
3579 445
3580 020162 012767 020170 160720          ;BIN*SM2*SF7*DM4*DF7 (D45.80)
3581 020170 012747          7$: MOV #61$, $LPERR ;SETUP ERROR LOOP
3582 020172 000402          61$: MOV (PC)+, -(PC) ;EXECUTE INSTRUCTION UNDER TEST
3583 020174 104377          BR 10$ ;WILL EXECUTE THIS IF INSTR. WORKS
3584 020176 000447          ERROR 377 ;STATE D45.80 FAILED
3585 447
3586 020200 012767 020206 160702          ;BIN*SM1*SF0*DM4*DF7 (D45.90)
3586 020200 012767 020206 160702          10$: MOV #60$, $LPERR ;SETUP ERROR LOOP
```



```

3587 020206 012767 141047 000024 60$: MOV #141047,11$ ;SETUP INSTRUCTION TO EXECUTE
3588 020214 012700 001162          MOV #STMP0,R0 ;PUT ADDRESS OF STMP0 IN R0
3589 020220 005200          INC R0
3590 020222 112710 000002          MOV#B #BIT1,(R0) ;SET BIT1 IN STMP1 HIGH BYTE
3591 020226 012705 001166          MOV #STMP2,R5 ;SETUP R5
3592 020232 012767 001000 160724 11$: MOV #BIT9,STMP1 ;SETUP STMP1
3593 020240 141047          BICB (R0),-(PC) ;EXECUTE INSTRUCTION UNDER TEST
3594 020242 005767 160716          TST STMP1 ;DID STMP1 GET CLEARED?
3595 020246 001402          BEQ 12$ ;BRANCH IF YES
3596 020250 104377          ERROR 377 ;STATE D45.00 FAILED
3597 020252 000450          450
3598          ;DAC*DM4*DF7 (D45.00)
3599 020254 012767 020262 160626 12$: MOV #57$,SLPERR ;SETUP ERROR LOOP
3600 020262 012767 140047 000016 57$: MOV #140047,13$ ;SETUP INSTRUCTION TO EXECUTE
3601 020270 012700 000002          MOV #BIT1,R0 ;SETUP R0 TO CHANGE DR FROM 7 TO 5
3602 020274 012705 001165          MOV #STMP1+1,R5 ;PUT ADDRESS OF STMP1 HIGH BYTE IN R5
3603 020300 012767 000002 160656 13$: MOV #BIT1,STMP1 ;SET UP STMP1 SO INSTRUCTION CLEARS IT
3604 020306 140047          BICB R0,-(PC) ;EXECUTE INSTRUCTION UNDER TEST
3605 020310 005767 160650          TST STMP1 ;DID STMP1 CLEAR?
3606 020314 001402          BEQ 14$ ;BRANCH IF YES
3607 020316 104377          ERROR 377 ;STATE D45.90 FAILED
3608 020320 000451          451
3609          ;DAC*DM5*DF7 (D45.01)
3610 020322 012767 020330 160560 14$: MOV #56$,SLPERR ;SETUP ERROR LOOP
3611 020330 012767 130057 000052 56$: MOV #130057,15$ ;SETUP INSTRUCTION TO EXECUTE
3612 020336 032737 000020 177776          BIT #BIT4,@#PSW ;IS T BIT ON?
3613 020344 001404          BEQ 20$ ;BRANCH IF NO
3614 020346 012737 000360 000066          MOV #360,@#TPVEC+2 ;SETUP TP VEC PSW
3615 020354 000403          BR 17$
3616 020356 012737 000340 000066 20$: MOV #PR7,@#TPVEC+2 ;SETUP BR4 INTERRUPT VECTOR
3617 020364 012737 020424 000064 17$: MOV #16$,@#TPVEC ;SET PROCESSOR PRIORITY BELOW BR4
3618 020372 000233          SPL 3 ;SET INTERRUPT BIT
3619 020374 152777 000100 160540          BISB #BIT6,@$TPS ;SEND CHARACTER TO PRINTER
3620 020402 012777 000015 160534          MOV #15,@$TPB ;EXECUTE INSTRUCTION UNDER TEST
3621 020410 130057          BITB R0,@-(PC) ;WILL EXECUTE IF STATE FAILS
3622 020412 142777 000100 160522          BICB #BIT6,@$TPS ;STATE D45.01 FAILED
3623 020420 104377          ERROR 377
3624 020422 000452          452
3625 020424 142777 000100 160510 16$: BICB #BIT6,@$TPS ;TEST OK, CLEAR INTERRUPT BIT
3626          ;CONTINUE

```

```

3627
3628          ;*****
3629          ;*TEST 40          RED ZONE TRAP
3630          ;*
3631          ;*          A RED ZONE TRAP IS FIRST ATTEMPTED WITH THE SP AT 336.
3632          ;*          IF BEN13 FAILS EXECUTION WILL GO TO EITHER BRK.30 OR BRK.20
3633          ;*          OR PUP.00.
3634          ;*          BRK.80 WILL CAUSE THE OLD PSW AND PC TO BE STACKED ON THE
3635          ;*          OLD STACK INSTEAD OF LOCATIONS 2 AND 0.
3636          ;*          BRK.20 WILL MAKE IT LOOK LIKE THE RED ZONE FAILED.
3637          ;*          PUP.00 WILL CAUSE A TRAP TO LOCATION 24.
3638          ;*
3639          ;*          IF THE PROCESSOR FAILS TO TRAP EITHER TMCD SL RED IS
3640          ;*          NOT GOING LOW OR TMCC ABORT IS NOT GOING LOW.
3641          ;*
3642          ;*          IF UBCB ABORT RESTART FAILS TO GO LOW OR E10(13)

```

```
3643      ;*      IS BAD THE PROCESSOR WILL HANG IN THE PAUSE STATE.
3644      ;*****
3645      TST40:  SCOPE
3646      MOV      #TST41,NEXTTST ;SAVE ADDRESS OF NEXT TEST
3647      MOV      #14$, $LADR    ;SETUP LOOP ADR
3648      MOV      @#PSW,$TMP3    ;SAVE PSW
3649      BIT      #BIT4,@#PSW    ;IS T BIT ON?
3650      BEQ      14$           ;BRANCH IF NO
3651      MOV      #PR7,-(SP)     ;PUT NEW PSW ON STACK
3652      MOV      #14$,-(SP)    ;PUT RETURN ADR ON STACK
3653      RTT
3654      MOV      #PR7,@#ERRVEC+2 ;RESTORE ERRVEC PSW
3655      MOV      #64$, $LPERR    ;SETUP ERROR LOOP
3656      MOV      #3$, @#PWRVEC   ;SETUP LOCATION 24
3657      MOV      #1$, @#ERRVEC   ;SETUP ERRVEC
3658      MOV      #336, SP       ;SET THE SP TO RED ZONE
3659      MOV      #-1, R0        ;SETUP R0
3660      MOV      R0, (SP)       ;EXECUTE THE TRAP INSTRUCTION
3661      MOV      #STACK, SP     ;RESET THE SP
3662      MOV      #CPUSPUR, @#EPRVEC ;RESTORE ERRVEC
3663      ERROR    235           ;RED ZONE REFERENCE FAILED TO TRAP
3664      BR      8$
3665      MOV      #STACK, SP     ;RESET THE SP
3666      MOV      #CPUSPUR, @#ERRVEC ;RESTORE LOCATION 4
3667      ERROR    237           ;BEN 13 FAILED TO PUP.00
3668      BR      8$
3669      CMP      @#0, #6$       ;DID BEN 13 FAIL?
3670      BEQ      7$           ;BRANCH IF NO
3671      MOV      #STACK, SP     ;RESET THE SP
3672      MOV      #CPUSPUR, @#ERRVEC ;RESTORE ERRVEC
3673      ERROR    240           ;BEN 13 FAILED TO BRK.80
3674      BR      8$
3675      CMP      #-1, @#336     ;; DID YEL ZONE OCCUR?
3676      BNE      8$           ;BRANCH IF NO
3677      CLR      @#336         ;SETUP FOR LOOPING
3678      ERROR    251           ;YEL ZONE IN RED REGION
3679
3680      ;TEST TO ENSURE PSW REFERENCE VIA THE SP CAUSES A RED ZONE TRAP
3681      MOV      #63$, $LPERR    ;SETUP ERROR LOOP
3682      MOV      #4$, @#ERRVEC   ;SETUP ERRVEC
3683      MOV      #PSW, SP       ;PUT ADDRESS OF PSW IN SP
3684      CLR      (SP)          ;EXECUTE THE TRAP CAUSING INSTRUCTION
3685      MOV      #1076, SP      ;RESET THE SP
3686      MOV      #CPUSPUR, @#ERRVEC ;RESTORE ERRVEC
3687      ERROR    241           ;NO RED ZONE ON STACK OVERFLOW
3688
3689      ;TEST TO ENSURE SL REG GREATER THAN BADRR CAUSES A RED ZONE
3690      MOV      #62$, $LPERR    ;SETUP ERROR LOOP
3691      MOV      #5$, @#ERRVEC   ;SETUP RESVEC
3692      MOV      #400, @#STKLMT  ;SET STACK LIMIT REGISTER TO 400
3693      MOV      #336, SP       ;SET THE SP
3694      CLR      (SP)          ;EXECUTE THE TRAP CAUSING INSTRUCTION
3695      MOV      #STACK, SP     ;RESET THE SP
3696      CLR      @#STKLMT      ;ENSURE SL REG. CLEAR
3697      MOV      #CPUSPUR, @#ERRVEC ;RESTORE ERRVEC
3698      ERROR    242           ;NO RED ZONE WHEN SL REG>BUS ADDR
```

```
3699  
3700 : TEST OF TMCD YEL ZONE (E15)  
3701 020750 012767 020762 160132 5$: MOV #61$, $LPERR ; SETUP ERROR LOOP  
3702 020756 005037 177774 CLR @#STKLMT ; ENSURE SL CLEAR  
3703 020762 012737 021002 000004 61$: MOV #9$, @#ERRVEC ; SETUP ERRVEC  
3704 020770 012706 000240 MOV #240, SP ; SETUP THE SP  
3705 020774 012700 177777 MOV #-1, R0 ; SETUP R0  
3706 021000 010016 MOV R0, (SP) ; EXECUTE THE TRAP CAUSING INSTRUCTION  
3707 021002 022737 177777 000240 9$: CMP #-1, @#240 ; DID YEL ZONE OCCUR?  
3708 021010 001010 BNE 10$ ; BRANCH IF NO  
3709 021012 012706 001100 MOV #STACK, SP ; RESTORE THE SP  
3710 021016 012737 032654 000004 MOV #CPUSPUR, @#ERRVEC ; RESTORE ERRVEC  
3711 021024 005037 000240 CLR @#240 ; FOR LOOPING  
3712 021030 104252 ERROR 252 ; TMCD YEL ZONE DID NOT GO LOW  
3713 021032 012767 021040 160050 10$: MOV #60$, $LPERR ; SETUP ERROR LOOP  
3714 021040 012737 021054 000004 60$: MOV #11$, @#ERRVEC ; SETUP ERRVEC  
3715 021046 012706 000140 MOV #140, SP ; SETUP THE SP  
3716 021052 010016 MOV R0, (SP) ; EXECUTE THE TRAP CAUSING INSTR.  
3717 021054 022737 177777 000140 11$: CMP #-1, @#140 ; DID YEL ZONE OCCUR?  
3718 021062 001010 BNE 12$ ; BRANCH IF NO  
3719 021064 012706 001100 MOV #STACK, SP ; RESTORE SP  
3720 021070 012737 032654 000004 MOV #CPUSPUR, @#ERRVEC ; RESTORE ERRVEC  
3721 021076 005037 000140 CLR @#140 ; FOR LOOPING  
3722 021102 104253 ERROR 253 ; TMCD YEL ZONE DID NOT GO LOW  
3723 021104 012706 001100 12$: MOV #STACK, SP ; RESTORE SP  
3724 021110 012737 032654 000004 MOV #CPUSPUR, @#ERRVEC ; RESTORE ERRVEC  
3725 021116 022737 000004 177766 CMP #BIT2, @#CPUERR ; DID RED ZONE BIT IN CPU ERROR SET?  
3726 021124 001407 BEQ 13$ ; BRANCH IF YES  
3727 021126 013767 177766 160020 MOV @#CPUERR, $REGO ; SAVE FOR TYPEOUT  
3728 021134 012767 000004 160020 MOV #BIT2, $TMP0 ; SAVE EXPECTED VALUE  
3729 021142 104261 ERROR 261 ; RED ZONE BIT IN CPU ERROR DID NOT SET  
3730 021144 012767 021152 157736 13$: MOV #57$, $LPERR ; SETUP ERROR LOOP  
3731 021152 005037 177766 57$: CLR @#CPUERR ; CLEAR RED ZONE BIT  
3732 021156 005737 177766 TST @#CPUERR ; DID REG CLEAR?  
3733 021162 001401 BEQ TST41 ; BRANCH IF YES  
3734 021164 104262 ERROR 262 ; RED ZONE BIT DID NOT CLEAR  
3735  
3736  
3737 : *****  
3738 : *TEST 41 BIT TEST OF STACK LIMIT REGISTER  
3739 : *  
3740 : * FIRST A 125252 AND 52525 PATTERN IS PUT IN THE REGISTER TO ENSURE  
3741 : * THAT THE REGISTER DOESN'T HAVE ANY STUCK BITS AND THAT  
3742 : * THE DMUX SELECT AND INPUT LINES WORK.  
3743 : *  
3744 : * IF SCCE SL ADRS DOES NOT GET TO TMCD OR IF TMCD E28 OR E14  
3745 : * IS BAD THE BR WILL BE SELECTED. THE PB REGISTER IS LOADED  
3746 : * WITH 200 SO IF TMCD LO BYTE EN DOES NOT GO LOW AN  
3747 : * ERROR WILL BE DETECTED.  
3748 : *****  
3748 021166 000004 TST41: SCOPE  
3749 021170 012767 021432 157776 MOV #TST42, $ESCAPE ; SAVE START ADDRESS OF NEXT TEST  
3750 021176 012767 021432 160064 MOV #TST42, NEXTTST ; SAVE START ADDRESS OF NEXT TEST  
3751 021204 012737 036266 000024 MOV #SPWRDN, @#PWVEC ; RESTORE POWER VECTOR  
3752 021212 005005 CLR R5 ; CLEAR R5  
3753 021214 012737 125252 177774 MOV #125252, @#STKLMT ; PUT PATTERN IN STACK LIMIT REG  
3754 021222 012737 000200 177770 MOV #200, @#177770 ; PUT 200 IN PB REGISTER
```

```

3755 021230 012700 125000      MOV      #125000,R0      ;SETUP R0 TO LOOK LIKE STK LIMIT
3756 021234 000237              SPL      7              ;PUT PRIORITY BITS IN KNOWN CONFIGURATION
3757 021236 020037 177774      CMP      R0,@#STKLMT    ;EXECUTE TEST ON STKLMT REG.
3758 021242 001404              BEQ      1$             ;BRANCH IF TEST OK
3759 021244 005205              INC      R5             ;INCREMENT TEST FAIL INDICATOR
3760 021246 013767 177774 157744  MOV      @#STKLMT,E1STKLM ;SAVE ERROR VALUE
3761 021254 012737 052400 177774 1$:  MOV      #52400,@#STKLMT ;PUT COMPLEMENT PATTERN REG.
3762 021262 012700 052400      MOV      #52400,R0      ;PUT IN R0
3763 021266 020037 177774      CMP      R0,@#STKLMT    ;EXECUTE TEST ON REG.
3764 021272 001415              BEQ      2$             ;BRANCH IF TEST OK
3765 021274 005705              TST      R5             ;DID FIRST TEST FAIL?
3766 021276 001023              BNE      3$             ;BRANCH IF YES
3767 021300 013767 177774 157714  MOV      @#STKLMT,E2STKLM ;SAVE ERROR VALUE
3768 021306 012767 052400 157646  MOV      #52400,$TMP0    ;SAVE EXPECTED VALUE
3769 021314 005037 177774      CLR      @#STKLMT       ;CLEAR THE REG
3770 021320 012706 001100      MOV      #STACK,SP      ;RESTORE THE SP
3771 021324 104243              ERROR   243             ;52400 PATTERN FAILED
3772 021326 005705              TST      R5             ;DID FIRST TEST FAIL?
3773 021330 001436              BEQ      4$             ;BRANCH IF NO
3774 021332 012767 125000 157622  MOV      #125000,$TMP0    ;SAVE EXPECTED VALUE
3775 021340 005037 177774      CLR      @#STKLMT       ;CLEAR REG.
3776 021344 104244              ERROR   244             ;125252 PATTERN FAILED
3777 021346 005037 177774 3$:  CLR      @#STKLMT       ;CLEAR STACK LIMIT REG
3778 021352 026727 157642 013767  CMP      E1STKLM,#13767  ;DID BR GET SELECTED ON STACK LIMIT REF.?
3779 021360 001001              BNE      5$             ;BRANCH IF NO
3780 021362 104245              ERROR   245             ;BR SELECTED BY DMUX
3781 021364 026727 157630 125200 5$:  CMP      E1STKLM,#125200 ;DID PB GET GATED ALSO?
3782 021372 001001              BNE      6$             ;BRANCH IF NO
3783 021374 104246              ERROR   246             ;TMCD LO BYTE EN DOES NOT GO LOW
3784 021376 022767 000340 157614 6$:  CMP      #340,E1STKLM    ;DID PS GET SELECTED?
3785 021404 001001              BNE      7$             ;BRANCH IF NO
3786 021406 104247              ERROR   247             ;D MUX SELECTED PSW
3787 021410 012767 125000 157544 7$:  MOV      #125000,$TMP0    ;
3788 021416 012767 052400 157540  MOV      #52400,$TMP1    ;
3789 021424 104250              ERROR   250             ;BOTH PATTERNS FAILED BUT DON'T KNOW WHY
3790 021426 005037 177774 4$:  CLR      @#STKLMT       ;CLEAR THE STACK LIMIT REG.

```

3791
3792 *****

3793 *TEST 42 SL REGISTER COMPARATOR TEST 1

3794
3795 * THIS TEST RUNS A HIGH BYTE COUNT PATTERN THRU THE BUS
3796 * ADDRESS MUX FOR EACH PATTERN OF THE STACK LIMIT REGISTER.
3797 * FOR EACH PATTERN OF ADDRESSES THERE WILL BE ONE YEL TRAP
3798 * AT THE ADDRESS CORRESPONDING TO THE SL REG+340 AND A RED
3799 * TRAP AT EVERY ADDRESS BELOW THIS.

3800 * THIS TEST ONLY TESTS ADDRESSES UP TO THE I/O PAGE.
3801 * THE I/O PAGE ADDRESSES WILL BE TESTED SEPARATELY WITH
3802 * MEMORY MANAGEMENT ENABLED AND THE I/O PAGE MAPPED INTO RESIDENT
3803 * MEMORY

3804 * 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

3805
3806
3807
3808
3809
3810

```

3811          10          NO TRAP ON RED ZONE ADDRESS
3812          12          NO TRAP ON YELLOW ZONE ADDRESS
3813
3814          *          THE LOW BYTE ADDRESS IN THE STACK POINTER IS ALWAYS
3815          *          340 AND WILL NOT BE TYPED ON AN ERROR.
3816          *          *****
3817 021432 000004 TST42: SCOPE
3818          *          *****
3819          *NOTE: IF THE LOOP ON ERROR SWITCH IS UP (SWITCH 9) THE TEST WILL
3820          *          LOOP ON THE FIRST ERROR WITH NO ERROR TYPEOUT. OTHERWISE ALL
3821          *          ERRORS WILL BE RECORDED IN A TABLE AND TYPED OUT AT THE
3822          *          END OF THE TEST.
3823          *          IF SWITCH 3 (DISABLE MEMORY MANAGEMENT TESTS) IS NOT ON,
3824          *          THIS TEST IS SKIPPED AND TEST 70 WILL EXECUTE.
3825          *          *****
3826 021434 012767 022042 157532      MOV      #TST43,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
3827 021442 012767 022042 157620      MOV      #TST43,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3828 021450 012767 003706 157426      MOV      #^D1990,$ICNT ;SETUP ITERATION COUNT
3829 021456 012767 021464 157422      MCV     #13$,SLPADR ;SETUP LOOP ADDRESS
3830 021464 032737 000010 177570 13$: BIT     #BIT3,@#SWR ;IS SWITCH 3 ON?
3831 021472 001002          BNE     11$ ;BRANCH IF YES
3832 021474 000177 157474          JMP     @ESCAPE ;GO TO NEXT TEST
3833 021500 012737 021610 000004 11$: MOV     #1$,@#ERRVEC ;SETUP ERRVEC
3834 021506 005067 157454          CLR     $TMP2 ;INITIALIZE ERROR OVERFLOW FLAG
3835 021512 012703 000340          MOV     #340,R3 ;SET SOB COUNT FOR SL REGISTER
3836 021516 012700 120000          MOV     #120000,R0 ;INITIALIZE ERROR DATA POINTER
3837 021522 005060 000002          CLR     2(R0) ;INITIALIZE ERROR DATA BUFFER
3838 021526 012701 000340          MOV     #340,R1 ;INITIALIZE YELLOW ZONE ADDRESS(TRAP CASE)
3839 021532 012704 000344          MOV     #344,R4 ;SETUP YELLOW ZONE ADDR(NO TRAP)
3840 021536 012702 000340 2$: MOV     #340,R2 ;SET SOB COUNT FOR SP
3841 021542 012705 000344          MOV     #344,R5 ;INITIALIZE SECONDARY STORAGE FOR SP
3842 021546 016567 177776 157406 3$: MOV     -2(R5),$TMP0 ;SAVE WORDS AT
3843 021554 016567 177774 157402          MOV     -4(R5),$TMP1 ;STACK UNDER TEST
3844 021562 010506 4$: MCV     R5,R6 ;SET THE SP
3845 021564 011616          MOV     (R6),(R6) ;EXECUTE TEST INSTRUCTION
3846          ;NO TRAP. DETERMINE IF THIS IS CORRECT.
3847 021566 020604          CMP     R6,R4 ;IS ADDRESS > YELL ZONE BOUNDRY?
3848 021570 10*066          BHI     5$ ;BRANCH IF YES
3849 021572 001403          BEQ     6$ ;BRANCH IF ADDRESS = YELL ZONE BOUNDRY
3850          ;NO TRAP ADDRESS IS LESS THAN YELLOW ZONE BOUNDRY
3851 021574 052710 000010          BIS     #BIT3,(R0) ;SET ERROR TYPE IN DATA BUFFER
3852 021600 000442          BR     10$ ;GO RECORD DATA
3853          ;NO TRAP EQUALS YELLOW ZONE BOUNDRY
3854 021602 052710 000012 6$: BIS     #12,(R0) ;SET ERROR TYPE IN DATA BUFFER
3855 021606 000437          BR     10$ ;GO RECORD DATA
3856
3857          ;GOT A TRAP. RESTORE AND DETERMINE IF IT IS CORRECT.
3858 021610 016765 157346 177776 1$: MOV     $TMP0,-2(R5) ;RESTORE WORDS AT
3859 021616 016765 157342 177774          MOV     $TMP1,-4(R5) ;OLD STACK UNDER TEST
3860 021624 032737 000004 177766          BIT     #BIT2,@#CPUERR ;WAS IT A RED ZONE?
3861 021632 001406          BEQ     8$ ;BRANCH IF NO
3862 021634 020106          CMP     R1,R6 ;IS ADDRESS < YELL ZONE BOUNDRY?
3863 021636 101043          BHI     5$ ;BRANCH IF YES
3864 021640 001422          BEQ     10$ ;BRANCH IF ADDRESS = YELL ZONE BOUNDRY
3865          ;RED ZONE TRAP ON LEGAL ADDRESS
3866 021642 052710 000002          BIS     #BIT1,(R0) ;SET ERROR TYPE IN DATA BUFFER

```

```

3867 021646 000417          BR      10$          ;GO RECORD DATA
3868          ;NOT A RED ZONE. IS IT A YELLOW ZONE?
3869 021650 032737 000010 177766 8$: BIT      #BIT3,@#CPUERR ;IS THIS A YELLOW ZONE TRAP?
3870 021656 001002          BNE      12$          ;BRANCH IF NO
3871 021660 000167 010770          JMP      CPUSPUR      ;GO TO SPURIOUS ROUTINE
3872 021664 020106          12$: CMP      R1,R6      ;IS ADDRESS = YELL ZONE BOUNDRY?
3873 021666 001427          BEQ      5$           ;BRANCH IF YES
3874 021670 101003          BHI      9$           ;BRANCH IF ADDRESS IS < YELL ZONE BOUNDRY
3875          ;YELLOW ZONE TRAP ON LEGAL ADDRESS
3876 021672 052710 000006          BIS      #6,(R0)      ;SET ERROR TYPE IN DATA BUFFER
3877 021676 000403          BR      10$          ;GO RECORD DATA
3878          ;YELLOW ZONE TRAP ON RED ZONE ADDRESS
3879 021700 052710 000004          9$: BIS      #BIT2,(R0) ;SET ERROR TYPE IN DATA BUFFER
3880 021704 000400          BR      10$          ;GO RECORD DATA
3881          ;
3882          ;RECORD ERROR DATA
3883 021706 032737 001000 177570 10$: BIT      #BIT9,@#SWR    ;IS LOOP ON ERROR ENABLED?
3884 021714 001322          BNE      4$           ;BRANCH IF YES
3885 021716 005767 157244          TST      $TMP2        ;HAS ERROR BUFFER OVERFLOWED?
3886 021722 001011          BNE      5$           ;BRANCH IF YES
3887 021724 005200          INC      R0           ;SET POINTER TO HIGH BYTE
3888 021726 113720 177775          MOVB    @#STKLM+1,(R0)+ ;SAVE ERROR STACK LIMIT
3889 021732 010620          MOV      R6,(R0)+    ;SAVE ERROR SP
3890 021734 020027 157774          CMP      R0,#157774  ;IS BUFFER AT PAGE 7?
3891 021740 001002          BNE      5$           ;BRANCH IF NO
3892 021742 005267 157220          INC      $TMP2        ;SET BUFFER OVERFLOW FLAG
3893          ;
3894 021746 062705 000400          5$: ADD      #400,R5    ;GO TO NEXT STACK ADDRESS
3895 021752 005037 177766          CLR      @#CPUERR    ;CLEAR ERROR REG
3896 021756 000240          NOP
3897 021760 005302          DEC      R2           ;REPLACES A
3898 021762 001271          BNE      3$           ;SOB INSTRUCTION
3899 021764 062737 000400 177774          ADD      #400,@#STKLMT ;GO TO NEXT SL ADDRESS
3900 021772 062701 000400          ADD      #400,R1     ;SET NEXT YELLOW ZONE ADDRESS
3901 021776 000240          NOP
3902 022000 062704 000400          ADD      #400,R4     ;SET NEXT YELLOW ZONE ADDR(NO TRAP)
3903 022004 005303          DEC      R3           ;THIS REPLACES
3904 022006 001253          BNE      2$           ;A SOB
3905          ;
3906          ;DONE WITH TEST. WAS THERE AN ERROR?
3907 022010 005037 177774          CLR      @#STKLMT    ;RESET THE SL REG
3908 022014 012706 001100          MOV      #STACK,SP   ;AND SP
3909 022020 012737 032654 000004          MOV      #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
3910 022026 005737 120002          TST      @#120002    ;WAS THERE AN ERROR?
3911 022032 001403          BEQ      TST43        ;BRANCH IF NO
3912 022034 010067 157114          MOV      R0,$REGO    ;SAVE ERROR DATA POINTER
3913 022040 104263          ERROR  263          ;STACK LIMIT COMPARATORS FAILED
3914          ;*****
3915          ;*TEST 43 ODD ADDRESS ERROR
3916          ;*
3917          ;*
3918          ;* BEN 13 SHOULD NOT FAIL.
3919          ;* IF THE PROCESSOR FAILS TO TRAP IN ALL SECTIONS EITHER TMCC ODD
3920          ;* ADRS ERR IS NOT GOING LOW OR TMCC BUS ERROR IS NOT GOING LOW.
3921          ;*
3922          ;* EACH TYPE OF ODD ADDRESS ERROR IS TESTED INDIVIDUALLY TO
          ;* ALLOW MAXIMUM ISOLATION.

```

```
3923 :* NOTE:AN ODD ADDRESS ON 'KERNEL DATI' CANNOT BE TESTED.  
3924 :* THIS SIGNAL COMES UP WHEN A TRAP VECTOR IS READ IN FROM THE BUS.  
3925 :*****  
3926 022042 000004 TST43: SCOPE  
3927 022044 012767 022476 157216 MOV #TST44,NEXTTST ;SAVE ADDRESS OF NEXT TEST  
3928 022052 005005 CLR R5 ;INITIALIZE ERROR COUNT  
3929 ;NON BYTE REFERENCE ON DATIP  
3930 022054 012706 001100 MOV #STACK,SP ;INITIALIZE THE SP  
3931 022060 012737 022140 000004 MOV #1$,@ERRVEC ;SETUP ERRVEC  
3932 022066 012700 001163 MOV #STMP0+1,R0 ;PUT ODD ADDRESS IN R0  
3933 022072 012767 177777 157062 MOV #-1,$TMP0 ;PUT -1 IN $TMP0 TO SEE IF TEST CHANGES IT  
3934 022100 005210 INC (R0) ;EXECUTE ODD ADDRESS INSTRUCTION  
3935 ;TRAP DID NOT OCCUR. TRY A DATI.  
3936 022102 012737 022126 000004 MOV #2$,@ERRVEC ;SETUP ERRVEC  
3937 022110 110030 MOVB R0,@(R0)+ ;EXECUTE DATI TO CAUSE ODD ADDR  
3938 022112 012737 032654 000004 MOV #CPUSPUR,@ERRVEC ;RESTORE ERRVEC  
3939 022120 005205 INC R5 ;SET ERROR COUNT  
3940 022122 104265 ERROR 265 ;NEITHER -BYIN OR DATI CAUSE ODD ADDR  
3941 022124 000423 BR 3$  
3942 022126 012737 032654 000004 2$: MOV #CPUSPUR,@ERRVEC ;RESTORE ERRVEC  
3943 022134 104266 ERROR 266 ;DATI TRAPPED BUT -BYIN DIDN'T  
3944 022136 000416 BR 3$  
3945 ;  
3946 ;NON BYTE REFERENCE WORKED. NOW TRY DATI.  
3947 022140 005037 177766 1$: CLR @CPUERR ;CLEAR ERROR REG  
3948 022144 012706 001100 MOV #STACK,SP ;INITIALIZE THE SP  
3949 022150 012767 022140 156732 MOV #1$,LPERR ;CHANGE LPERR ADDRESS TO THIS SECTION  
3950 022156 012737 022174 000004 MOV #3$,@ERRVEC ;SETUP ERRVEC  
3951 022164 012700 001163 MOV #STMP0+1,R0 ;PUT ODD ADDR IN R0  
3952 022170 110030 MOVB R0,@(R0)+ ;EXECUTE ODD ADDRESS INSTRUCTION  
3953 022172 000404 BR 13$  
3954 022174 032737 000100 177766 3$: BIT #BIT6,@CPUERR ;DID ODD ADDR BIT SET?  
3955 022202 001037 BNE 12$ ;BRANCH IF YES  
3956 ;TRAP FAILED OR ERROR REG FAILED. TRY A DATO. (REVERSES INPUTS TO TMCC E5(12,13))  
3957 022204 005037 177766 13$: CLR @CPUERR ;CLEAR ODD ADDR BIT  
3958 022210 012706 001077 MOV #STACK-1,SP ;MAKE SP AN ODD ADDRESS  
3959 022214 012737 022250 000004 MOV #4$,@ERRVEC ;SETUP ERRVEC  
3960 022222 012737 022232 000030 MOV #5$,@EMTVEC ;SETUP EMT VECTOR  
3961 022230 104000 EMT 0 ;EXECUTE DATO TO SP  
3962 022232 012706 001100 5$: MOV #STACK,SP ;RESTORE SP  
3963 022236 012737 032654 000004 MOV #CPUSPUR,@ERRVEC ;RESTORE ERRVEC  
3964 022244 104267 ERROR 267 ;BOTH DATI AND DATO FAILED  
3965 022246 000446 BR 6$  
3966 022250 012706 001100 4$: MOV #STACK,SP ;RESTORE THE SP  
3967 022254 012737 032654 000004 MOV #CPUSPUR,@ERRVEC ;RESTORE ERRVEC  
3968 022262 032737 000100 ;177766 BIT #BIT6,@CPUERR ;DID ODD ADDR BIT SET?  
3969 022270 001401 BEQ 14$ ;BRANCH IF NO  
3970 022272 104270 ERROR 270 ;DATO WORKS BUT DATI FAILED  
3971 022274 104377 14$: ERROR 377  
3972 022276 000456 456  
3973 022300 000431 BR 6$  
3974 ;  
3975 ;EITHER DATI WORKED OR -BYIN AND DATI FAILED. NOW TRY DATO.  
3976 022302 012767 022174 156600 12$: MOV #3$,LPERR ;CHANGE LPERR ADDRESS TO THIS SECTION  
3977 022310 012706 001077 MOV #STACK-1,SP ;MAKE SP AN ODD ADDRESS  
3978 022314 012737 022364 000004 MOV #6$,@ERRVEC ;SETUP ERRVEC
```



```
3979 022322 012737 022332 000030      MOV      #7$, @EMTVEC      ;SETUP EMTVEC TO CATCH A FAILURE
3980 022330 104000                    EMT      0                  ;EXECUTE DATO TO ODD ADDRESS
3981                    ;TRAP FAILED
3982 022332 012706 001076      7$:  MOV      #1076, SP        ;RESET SP
3983 022336 012737 032654 000004      MOV      #CPUSPUR, @ERRVEC ;RESTORE ERRVEC
3984 022344 012737 033536 000030      MOV      #ERROR, @EMTVEC  ;RESTORE EMTVEC
3985 022352 005705                    TST      R5
3986 022354 001002                    BNE     11$
3987 022356 104271                    ERROR   271                ;NO TRAP ON DATO BUT DATI & -BYIN OK
3988 022360 000401                    BR      6$
3989 022362 104274      11$:  ERROR   274                ;NO TRAPS
3990
3991                    ;EITHER DATO WORKED OR -BYIN AND DATI AND DATO FAILED OR DATO
3992                    ;FAILED BUT -BYIN AND DATI OK. TRY SM357 AND SRC1 DATI.
3993 022364 012737 033536 000030      6$:  MOV      #ERROR, @EMTVEC ;RESTORE EMT VEC
3994 022372 012767 022400 156510      MOV      #8$, $LPERR      ;CHANGE LOOP ERROR ADDR TO THIS SECT.
3995 022400 012706 001076      8$:  MOV      #1076, SP        ;RESET SP
3996 022404 012737 022452 000004      MOV      #9$, @ERRVEC     ;SETUP ERRVEC
3997 022412 012700 001163      MOV      #STMP0+1, R0      ;PUT ODD ADDRESS IN R0
3998 022416 012767 001164 156536      MOV      #STMP1, STMP0    ;PUT EVEN ADDRESS IN STMP0
3999 022424 113001                    MOVB    @(R0)+, R1        ;EXECUTE INSTRUCTION TO CAUSE TRAP
4000 022426 012737 032654 000004      MOV      #CPUSPUR, @ERRVEC ;RESTORE ERRVEC
4001 022434 012767 022476 156532      MOV      #TST44, $ESCAPE  ;SAVE START ADDRESS OF NEXT TEST
4002 022442 012767 022476 156620      MOV      #TST44, NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
4003 022450 104272                    ERROR   272                ;SM357*SRC1 DATI FAILED TO TRAP
4004
4005                    ;SM357*SRC1 DATI WORKS CHECK CPU ERROR REG.
4006 022452 012737 032654 000004      9$:  MOV      #CPUSPUR, @ERRVEC ;RESTORE ERRVEC
4007 022460 032737 000100 177766      BIT      #BIT6, @CPUERR   ;IS ODD ADDRESS BIT SET?
4008 022466 001001                    BNE     10$                ;BRANCH IF YES
4009 022470 104273                    ERROR   273                ;CPU ERROR REG BIT DOES NOT SET
4010 022472 005037 177766      10$:  CLR      @CPUERR        ;CLEAR ERROR REG.
4011                    ;CONTINUE
4012
4013                    ;*****
4014                    ;*TEST 44      ILLEGAL INSTRUCTIONS
4015                    ;*      THIS TEST ENSURES THAT ILLEGAL OP CODES TRAP TO LOCATION 10.
4016                    ;*      ONLY THOSE OP CODES THAT HAVE A SINGLE BIT THAT
4017                    ;*      DISTINGUISHES THEM FROM A LEGAL INSTRUCTION WILL BE TESTED.
4018                    ;*****
4018 022476 000004      TST44: SCOPE
4019 022500 012767 023142 156562      MOV      #KBTST, NEXTTST  ;SAVE ADDRESS OF NEXT TEST
4020 022506 012767 022542 156372      MOV      #31$, $LPADR     ;SETUP LOOP ADR
4021 022514 032737 000020 177776      BIT      #BIT4, @PSW      ;IS T BIT ON?
4022 022522 001404                    BEQ     30$                ;BRANCH IF NO
4023 022524 012737 000360 000012      MOV      #360, @RESVEC+2 ;SETUP RESVEC PSW
4024 022532 000403                    BR      31$                ;GO DO TEST
4025                    ;SECTION 1-10 THRU 77
4026 022534 012737 000340 000012      30$:  MOV      #PR7, @RESVEC+2 ;RESTORE RESVEC PSW
4027 022542 012767 022556 156340      31$:  MOV      #1$, $LPERR     ;SETUP ERROR LOOP
4028 022550 012737 022574 000010      MOV      #2$, @RESVEC     ;SETUP RESVEC
4029 022556 012706 001100      1$:  MOV      #STACK, SP      ;INITIALIZE THE SP
4030 022562 012746 022570      MOV      #3$, -(SP)       ;PUT ADDRESS OF 3$ ON STACK
4031 022566 000010                    ;EXECUTE OP CODE 10
4032                    ;FAILURE-RTT OCCURED
4033 022570 104377      3$:  ERROR   377                ;OP CODE 10 FAILED TO TRAP
4034 022572 000453                    ;453
```



```
4035 022574 012767 022610 156306 2$: MOV #4$, $LPERR ;SETUP ERROR LOOP
4036 022602 012737 022616 000010 MOV #5$, @#RESVEC ;SETUP RESVEC
4037 022610 000015 4$: 15 ;EXECUTE OP CODE 15
4038 ;FAILURE
4039 022612 104377 ERROR 377 ;OP CODE 15 FAILED TO TRAP
4040 022614 000453 453
4041 022616 012767 022632 156264 5$: MOV #6$, $LPERR ;SETUP ERROR LOOP
4042 022624 012737 022640 000010 MOV #7$, @#RESVEC ;SETUP RESVEC
4043 022632 000025 6$: 25 ;EXECUTE OP CODE 25
4044 ;FAILURE
4045 022634 104377 ERROR 377 ;OP CODE 25 FAILED TO TRAP
4046 022636 000453 453
4047 022640 012767 022654 156242 7$: MOV #8$, $LPERR ;SETUP ERROR LOOP
4048 022646 012737 022662 000010 MOV #9$, @#RESVEC ;SETUP RESVEC
4049 022654 000045 8$: 45 ;EXECUTE OP CODE 45
4050 ;FAILURE
4051 022656 104377 ERROR 377
4052 022660 000453 453 ;OP CODE 45 FAILED TO TRAP
4053
4054
4055 ;*****
4056 022662 012767 022676 156220 9$: MOV #10$, $LPERR ;SETUP ERROR LOOP
4057 022670 012737 022714 000010 MOV #11$, @#RESVEC ;SETUP RESVEC
4058 022676 012706 001100 10$: MOV #STACK, SP ;INITIALIZE THE SP
4059 022702 012700 022710 MOV #12$, R0 ;SETUP R0 INCASE RTS
4060 022706 000210 210 ;EXECUTE OP CODE 210
4061 ;FAILURE-RTS EXECUTED
4062 022710 104377 12$: ERROR 377 ;OP CODE 210 FAILED TO TRAP
4063 022712 000453 453
4064 022714 012767 022730 156166 11$: MOV #13$, $LPERR ;SETUP ERROR LOOP
4065 022722 012737 022746 000010 MOV #14$, @#RESVEC ;SETUP RESVEC
4066 022730 012706 001100 13$: MOV #STACK, SP ;INITIALIZE THE SP
4067 022734 012700 022742 MOV #15$, R0 ;SETUP R0 TO CATCH RTS
4068 022740 000220 220 ;EXECUTE OP CODE 220
4069 ;FAILURE-RTS EXECUTED
4070 022742 104377 15$: ERROR 377 ;OP CODE 220 FAILED TO TRAP
4071 022744 000453 453
4072
4073 ;*****
4074 ;SECTION 3-7000 THRU 7777
4075 022746 012767 022762 156134 14$: MOV #27$, $LPERR ;SETUP ERROR LOOP
4076 022754 012737 022770 000010 MOV #16$, @#RESVEC ;SETUP RESVEC
4077 022762 007000 27$: 7000 ;EXECUTE OP CODE 7000
4078 ;FAILURE
4079 022764 104377 ERROR 377 ;OP CODE 7000 FAILED TO TRAP
4080 022766 000453 453
4081
4082 ;*****
4083 ;SECTION 4-75000 THRU 76000
4084 022770 012767 023004 156112 16$: MOV #17$, $LPERR ;SETUP ERROR LOOP
4085 022776 012737 023012 000010 MOV #18$, @#RESVEC ;SETUP RESVEC
4086 023004 075000 17$: 75000 ;EXECUTE OP CODE 75000
4087 ;FAILURE
4088 023006 104377 ERROR 377 ;OP CODE 75000 FAILED TO TRAP
4089 023010 000453 453
4090 023012 012767 023026 156070 18$: MOV #19$, $LPERR ;SETUP ERROR LOOP
```

4091 023020 012737 023034 000010
4092 023026 076000
4093
4094 023030 104377
4095 023032 000453
4096
4097
4098
4099 023034 012767 023050 156046
4100 023042 012737 023056 000010
4101 023050 106400
4102
4103 023052 104377
4104 023054 000453
4105
4106
4107
4108 023056 012767 023072 156024
4109 023064 012737 023104 000010
4110 023072 012706 001100
4111 023076 106700
4112
4113 023100 104377
4114 023102 000453
4115 023104 012767 023120 155776
4116 023112 012737 023126 000010
4117 023120 107000
4118
4119 023122 104377
4120 023124 000453
4121 023126 012737 000012 000010
4122 023134 012737 000340 000012
4123
4124
4125
4126
4127
4128
4129
4130
4131
4132
4133 023142 005227 177777
4134 023146 001032
4135 023150 005037 001272
4136 023154 012737 023174 000010
4137 023162 000007
4138
4139 023164 012737 000001 001272
4140 023172 000403
4141
4142 023174
4143 023174 012716 023202
4144 023200 000002
4145 023202 012737 000012 000010
4146

```
MOV #20$,@#RESVEC ;SETUP RESVEC
19$: MOV 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$: MOV 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$: MOV #STACK,SP ;INITIALIZE THE SP
MOV 106700 ;EXECUTE OP CODE 106700
;FAILURE
ERROR 377 ;OP CODE 106700 FAILED TO TRAP
453
24$: MOV #25$, $LPERR ;SETUP ERROR LOOP
MOV #26$,@#RESVEC ;SETUP RESVEC
25$: MOV 107000 ;EXECUTE OP CODE 107000
;FAILURE
ERROR 377 ;OP CODE 107000 FAILED TO TRAP
453
26$: MOV #12,@#RESVEC ;RESTORE RESVEC
MOV #PR7,@#RESVEC+2 ;RESTORE RESVEC PSW
;CONTINUE

:*****
: THIS ROUTINE IS USED TO DETERMINE WHICH CPU WE ARE RUNNING ON FOR THE NEXT
: TEST. THE ROUTINE EXECUTES AN MFPT INSTRUCTION WHICH ONLY EXISTS ON A KB11-E
: OR KB11-EM. ON A KB11-B/C OR KB11-CM THE MFPT WILL TRAP. AFTER DETERMINING
: WHICH CPU IS BEING RUN ON A MESSAGE WILL BE PRINTED. THIS TEST COULD NOT
: BE RUN BEFORE THE PREVIOUS RESERVED INSTRUCTION TEST.

KBTST: INC #-1 ;FIRST TIME?
BNE KBDONE ;BRANCH IF NOT
CLR @#KB11E ;CLEAR KB11E AND KB11EM FLAGS
MOV #MFPTTR,@#RESVEC ;SET UP TRAP ADDRESS FOR MFPT AT RESERV VECTOR
MFPT ;EXECUTE MFPT. WILL TRAP ON 1170 (KB11B/C) OR
;KB11-CM
MOV #1,@#KB11E ;HERE IF KB11E OR KB11EM. SET FLAG
BR ENDKB ;DONE DETERMINING WHICH CPU

MFPTTR: ;HERE IF MFPT TRAPPED.
MOV #ENDKB,(SP) ;SET UP RETURN ADDRESS FOR RTI
RTI ;RETURN
ENDKB: MOV #12,@#RESVEC ;RESTORE RESERVE VECTOR
```

4147 023210 104400 036426
4148 023214 005737 001272
4149 023220 001003
4150 023222 104400 036465
4151 023226 000402
4152 023230 104400 036532
4153 023234
4154
4155
4156
4157
4158
4159
4160
4161
4162
4163
4164
4165
4166
4167
4168 023234 000004
4169 023236 012767 023504 155730
4170 023244 012767 023504 156016
4171
4172
4173 023252 012746 000340
4174 023256 012746 023272
4175 023262 013767 177776 155700
4176 023270 000006
4177
4178 023272 012767 023300 155610
4179 023300 012706 001100
4180 023304 012737 023370 000014
4181 023312 012737 023366 000010
4182 023320 012746 000360
4183 023324 012746 023332
4184 023330 000002
4185 023332 012746 000340
4186 023336 012746 023352
4187 023342 013767 177776 155636
4188 023350 000006
4189 023352 032767 000020 155626
4190 023360 001401
4191 023362 104275
4192 023364 104276
4193 023366 104300
4194 023370 005767 155676
4195 023374 001443
4196 023376 012746 000340
4197 023402 012746 023416
4198 023406 013767 177776 155554
4199 023414 000006
4200 023416 012767 023424 155464
4201 023424 012706 001100
4202 023430 012737 023450 000014

TYPE ,MSG1 ;<15><12>CPU UNDER TEST FOUND TO BE A
TST @KB11E ;IS THIS A KB11-E OR KB11-EM?
BNE 101\$;BR IF EITHER ONE
TYPE ,MSG3 ;KB11-B/C OR KB11-CM<15><12>
BR KBDONE ;SKIP OTHER MESSAGE
101\$: TYPE ,MSG5 ;KB11-E OR KB11EM<15><12>
KBDONE:
:*****
:*****
:TEST 45 T BIT TRAP
:
: IF BEN 13 FAILS EXECUTION WOULD GO TO BRK.20.
: THIS WOULD LOOK LIKE THE TRAP DIDN'T OCCUR.
:
: IF THE TRAP DOESN'T OCCUR THEN EITHER PDRD PS04(1) DOES NOT GET
: TO TMCB AS A HIGH OR IT DOES NOT GET TO TMCB E51(10)
: AS A LOW OR E51 IS BAD OR IRCD RTT DOES NOT GET TO TMCB AS A HIGH.
:
: THIS TEST ALSO CHECKS THAT PS<08> SET WILL INHIBIT A T BIT TRAP IF
: THIS IS A KB11-E OR KB11-EM.
:*****
TST45: SCOPE
MOV #TST46,\$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
MOV #TST46,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
:*****
:THIS CODE TURNS THE T BIT OFF IF IT IS ON.
MOV #PR7,-(SP) ;PUT PRIORITY LEVEL 7 ON SP
MOV #1\$,-(SP) ;PUT RETURN ADDRESS ON SP
MOV @PSW,\$TMP3 ;SAVE PSW
RTT ;TURN OFF T BIT
:*****
1\$: MOV #2\$, \$LPERR ;SETUP LOOP ADDRESS
2\$: MOV #1100,SP ;INITIALIZE THE SP
MOV #3\$, @TBITVEC ;SET UP T BIT VECTOR
MOV #8\$, @RESVEC ;SETUP RESVEC
MOV #360,-(SP) ;PUT NEW PSW ON STACK (ENABLE T BIT)
MOV #5\$, -(SP) ;PUT RETURN LOCATION ON STACK
RTI ;TURN ON T BIT
5\$: MOV #PR7,-(SP) ;SETUP STACK TO
MOV #6\$, -(SP) ;TURN OFF T BIT
MOV @PSW,\$ERPSW ;SAVE PSW
RTT ;TURN T BIT OFF
6\$: BIT #BIT4,\$ERPSW ;DID T BIT SET?
BEQ 7\$;BRANCH IF NO
ERROR 275 ;T BIT TRAP FAILED
7\$: ERROR 276 ;T BIT NEVER SET
8\$: ERROR 300 ;TRAP VECTOR DECODE FAILED
3\$: TST KB11E ;KB11-E OR KB11-EM?
BEQ 11\$;DONE IF NOT
MOV #PR7,-(SP) ;PUT PRIORITY LEVEL 7 ON SP
MOV #15\$, -(SP) ;PUT RETURN ADDRESS ON SP
MOV @PSW,\$TMP3 ;SAVE PSW
RTT ;TURN OFF T BIT
15\$: MOV #12\$, \$LPERR ;SET UP LOOP ADDRESS
12\$: MOV #STACK,SP ;INIT STACK POINTER
MOV #10\$, @TBITVEC ;SET UP T BIT VECTOR

```

4203 023436 012746 000760      MOV      #760,-(SP)      ;NEW PSW ON STACK, ENABLE T BIT AND
4204                                ;SET PS<08> (SUSPEND BIT)
4205 023442 012746 023504      MOV      #11$,-(SP)     ;RETURN LOCATION ON STACK. SHOULD GO TO 11$
4206                                ;INSTEAD OF 10$ ON RTI
4207 023446 000002                                ;TURN ON T BIT AND PS<08>
4208 023450 012746 000340      10$:    MOV      #PR7,-(SP) ;SETUP STACK TO
4209 023454 012746 023470      MOV      #13$,-(SP)     ;TURN OFF T BIT
4210 023460 013767 177776 155520  MOV      @WPSW,$ERPSW   ;SAVE PSW
4211 023466 000006                                ;TURN T BIT OFF
4212 023470 032767 000400 155510 13$:    BIT      #BIT8,$ERPSW  ;DID PS<08> SET?
4213 023476 001401                                ;BR IF NOT
4214 023500 104277                                ;SETTING PS<08> FAILED TO DISABLE T-TRAP
4215 023502 104376      14$:    ERROR   277
4216 023504      11$:    ERROR   376      ;PS<08> FAILED TO SET
4217                                ;CONTINUE

```

*TEST 46 T BIT TRAP AND RTT

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

TST46: SCOPE

```

4224 023504 000004      MOV      #TST47,NEXTTST ;SAVE ADDRESS OF NEXT TEST
4225 023506 012767 023556 155554  CLR      R0             ;ENSURE R0 CLEAR
4226 023514 005000      MOV      #STACK,SP     ;INITIALIZE THE SP
4227 023516 012706 001100      MOV      #360,-(SP)    ;PUT NEW PSW ON STACK (ENABLE T BIT)
4228 023522 012746 000360      MOV      #1$,-(SP)     ;PUT PC ON STACK
4229 023526 012746 023542      MOV      #2$,@TBITVEC  ;SETUP T BIT VECTOR
4230 023532 012737 023546 000014  RTT                                ;TURN T BIT ON WITH RTT
4231 023540 000006                                ;THIS SHOULD EXECUTE
4232 023542 012700 000001      1$:    MOV      #1,R0
4233 023546 022700 000001      2$:    CMP      #1,R0
4234 023552 001401                                ;DID MOV INSTRUCTION EXECUTE?
4235 023554 104301      BEQ     TST47          ;BRANCH IF YES
4236                                ;RTT DID NOT DISABLE T BIT
4237                                ;ERROR 301

```

*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

4259	:	*	13	BR 5	PIR 5
4260	:	*	14	BR 5	PIR 6
4261	:	*	15	BR 5	PIR 7
4262	:	*	16	PIR 5	BR 6
4263	:	*	17	PIR 5	SL YELLOW
4264	:	*	18	BR 6	PIR 6
4265	:	*	19	BR 6	PIR 7
4266	:	*	20	PIR 6	SL YELLOW
4267	:	*	21	PIR 7	SL YELLOW

```
*****
TST47: SCOPE
4269 023556 000004
4270 023560 012767 026366 155502      MOV    #TST50,NEXTTST ;SAVE ADDRESS OF NEXT TEST
4271 023566 012767 003706 155310      MOV    #D1990,$ICNT  ;SETUP ITERATION COUNT
4272 023574 012767 023610 155304      MOV    #103$,$LPADR  ;SETUP LOOP ADDRESS
4273 023602 012737 032640 000014      MOV    #SRTN,@#TBITVEC ;RESTORE T BIT VEC
4274 023610 005005      103$: CLR    R5            ;CLEAR BR5 DISABLE FLAG
4275 023612 005004      CLR    R4            ;CLEAR BR6 DISABLE FLAG
4276 023614 005003      CLR    R3            ;CLEAR BR4 DISABLE FLAG
4277 023616 032737 000020 001170      BIT    #BIT4,@#STMP3 ;WAS T BIT ON?
4278 023624 001417      BEQ    71$           ;BRANCH IF NO
4279 023626 012737 000360 000066      MOV    #360,@#TPVEC+2
4280 023634 012737 000360 000242      MOV    #360,@#PIRQVEC+2
4281 023642 012737 000360 000006      MOV    #360,@#ERRVEC+2
4282 023650 012746 000360      MOV    #360,-(SP)    ;SET UP STACK PSW
4283 023654 012746 023706      MOV    #72$,-(SP)    ;PUT RETURN ADDRESS ON STACK
4284 023660 000006      RTT                    ;TURN T BIT ON
4285 023662 000411      BR    72$
4286 023664 012737 000340 000066      71$: MOV    #PR7,@#TPVEC+2 ;PUT PRIORITY 7 IN PRINTER VECTOR
4287 023672 012737 000340 000006      MOV    #PR7,@#ERRVEC+2 ;RESTORE ERRVEC PSW
4288 023700 012737 000340 000242      MOV    #PR7,@#PIRQVEC+2 ;PUT PRIORITY 7 IN PIRQ VECTOR
4289 023706 032737 000400 177570      72$: BIT    #SW8,@#SWR    ;SWITCH 8 ON?
4290 023714 001006      BNE    100$          ;BRANCH IF YES
4291 023716 032737 000040 177570      BIT    #SW5,@#SWR    ;IS BR 5 TESTING DISABLED?
4292 023724 001402      BEQ    100$          ;BRANCH IF NO
4293 023726 005205      102$: INC    R5            ;SET BR 5 DISABLE FLAG
4294 023730 000403      BR    101$          ;CONTINUE
4295 023732 005767 155266      100$: TST    INTER5      ;IS THERE A BR 5 DEVICE?
4296 023736 001773      BEQ    102$          ;BRANCH IF NO
*****
SECTION 1 - BR4 AND PIR1
4299 023740 032737 000400 177570      101$: BIT    #SW8,@#SWR    ;SWITCH 8 ON?
4300 023746 001006      BNE    68$           ;BRANCH IF YES
4301 023750 032737 000020 177570      BIT    #SW4,@#SWR    ;IS BR4 TESTING DISABLED?
4302 023756 001402      BFO    68$           ;BRANCH IF NO
4303 023760 005203      INC    R3            ;SET BR4 FLAG
4304 023762 000430      BR    2$            ;GO TO NEXT SECTION
4305 023764 012767 024006 155116      68$: MOV    #1$,$LPERR    ;SETUP ERROR LOOP
4306 023772 012737 024044 000064      MOV    #2$,@#TPVEC    ;SETUP PRINTER VECTOR
4307 024000 012737 024032 000240      MOV    #3$,@#PIRQVEC  ;SETUP PIRQ VECTOR
4308 024006 012706 001076      1$: MOV    #1076,SP     ;INITIALIZE THE SP
4309 024012 000237      SPL    7            ;ENSURE CPU AT LEVEL 7
4310 024014 052737 001000 177772      BIS    #BIT9,@#PIRQ    ;SET PIR LEVEL 1
4311 024022 004767 002204      JSR    PC,LEVEL4     ;GO GET BR4
4312 024026 000230      SPL    0            ;SET CPU AT ZERO
4313 024030 000240      NOP                    ;USED WITH SPL
4314      ;FAILURE PIR CAME THRU
```

```
4315 024032 005037 177772 3$: CLR @PIRQ ;CLEAR THE PIRQ
4316 024036 005077 155100 CLR @STPS ;CLEAR THE PRINTER FLAG
4317 024042 104302 ERROR 302 ;PIR 1 DID NOT DISABLE ON BR
4318 *****
4319 :SECTION 2 - PIR 1 AND SL YELLOW
4320 024044 005077 155072 2$: CLR @STPS ;CLEAR PRINTER INT FLAG
4321 024050 012767 024076 155032 MOV #4$, $LPERR ;SETUP ERROR LOOP
4322 024056 012737 024134 000004 MOV #5$, @ERRVEC ;SETUP LOCATION 4
4323 024064 012737 024114 000240 MOV #6$, @PIRQVEC ;SETUP PIRQ VECTOR
4324 024072 005037 177772 CLR @PIRQ ;CLEAR LEVEL 1
4325 024076 012706 000376 4$: MOV #376, SP ;SETUP THE SP
4326 024102 052737 001000 177772 BIS #BIT9, @PIRQ ;SET LEVEL 1
4327 024110 000230 SPL 0 ;SET CPU AT ZERO
4328 024112 011616 MOV (SP), (SP) ;EXECUTE YELL ZONE INSTR
4329 :FAILURE, PIR CAME THRU
4330 024114 012706 001100 6$: MOV #STACK, SP ;RESET SP
4331 024120 012737 032654 000004 MOV #CPUSPUR, @ERRVEC ;RESTORE LOCATION 4
4332 024126 005037 177772 CLR @PIRQ ;CLEAR LEVEL 1
4333 024132 104303 ERROR 303 ;PIR CAME THRU ON YELLOW ZONE
4334 :
4335 *****
4336 :SECTION 3 - PIR 2 AND SL YELLOW
4337 024134 005037 177772 5$: CLR @PIRQ ;CLEAR LEVEL 1
4338 024140 012767 024162 154742 MOV #7$, $LPERR ;SETUP ERROR LOOP
4339 024146 012737 024220 000004 MOV #8$, @ERRVEC ;SETUP LOCATION 4
4340 024154 012737 024200 000240 MOV #9$, @PIRQVEC ;SETUP PIRQ VECTOR
4341 024162 012706 000376 7$: MOV #376, SP ;SETUP THE SP
4342 024166 052737 002000 177772 BIS #BIT10, @PIRQ ;SET LEVEL 2
4343 024174 000231 SPL 1 ;SET CPU AT LEVEL 1
4344 024176 011616 MOV (SP), (SP) ;EXECUTE TRAP CAUSING INSTR
4345 :FAILURE, PIR CAME THRU
4346 024200 012706 001100 9$: MOV #STACK, SP ;RESET SP
4347 024204 012737 032654 000004 MOV #CPUSPUR, @ERRVEC ;RESET ERRVEC
4348 024212 005037 177772 CLR @PIRQ ;CLEAR LEVEL 2
4349 024216 104304 ERROR 304 ;PIR 2 CAME THRU ON YELLOW ZONE
4350 :
4351 *****
4352 :SECTION 4 - PIR 3 AND YEL ZONE
4353 024220 005037 177772 8$: CLR @PIRQ ;CLEAR LEVEL 2
4354 024224 012767 024246 154656 MOV #10$, $LPERR ;SETUP ERROR LOOP
4355 024232 012737 024304 000004 MOV #11$, @ERRVEC ;SETUP ERR VECTOR
4356 024240 012737 024264 000240 MOV #12$, @PIRQVEC ;SETUP PIRQ VECTOR
4357 024246 012706 000376 10$: MOV #376, SP ;SETUP THE SP
4358 024252 052737 004000 177772 BIS #BIT11, @PIRQ ;SET LEVEL 3
4359 024260 000232 SPL 2 ;SET CPU AT LEVEL 2
4360 024262 011616 MOV (SP), (SP) ;EXECUTE TRAP CAUSING INSTR
4361 :FAILURE, PIR CAME THRU
4362 024264 012706 001100 12$: MOV #STACK, SP ;RESET SP
4363 024270 012737 032654 000004 MOV #CPUSPUR, @ERRVEC ;RESET LOCATION 4
4364 024276 005037 177772 CLR @PIRQ ;CLEAR LEVEL 3
4365 024302 104305 ERROR 305 ;PIR 3 CAME THRU ON YELLOW ZONE
4366 :
4367 *****
4368 :SECTION 5- PIR4 AND BR4
4369 024304 012706 001100 11$: MOV #STACK, SP ;RESTORE THE SP
4370 024310 005703 TST R3 ;IS BR4 TESTING DISABLED?
4370 024312 001402 BEQ 70$ ;BRANCH IF NO
```

```
4371 024314 000167 000410          JMP      29$          ;SKIP BR4 SECTIONS
4372 024320 012767 024342 154562 70$:  MOV     #13$, $LPERR ;SETUP ERROR LOOP
4373 024326 012737 024376 000240    MOV     #15$, @#PIRQVEC ;SETUP PIRQ VEC
4374 024334 012737 024364 000064    MOV     #14$, @#TPVEC  ;SETUP TPVEC
4375 024342 012706 001100          MOV     #STACK, SP   ;INITIALIZE THE SP
4376 024346 012737 010000 177772    MOV     #BIT12, @#PIRQ ;SET PIR LEVEL 4
4377 024354 004767 001652          JSR     PC, LEVEL4   ;GO GET BR 4
4378 024360 000233          SPL     3            ;LOWER CPU
4379 024362 000240          NOP                ;REQUIRED BECAUSE OF SPL
4380          ;FAILURE, BR4 CAME THRL
4381 024364 005077 154552 14$:  CLR     @STPS        ;CLEAR PRINTER INT FLAG
4382 024370 005037 177772          CLR     @#PIRQ      ;CLEAR LEVEL 4
4383 024374 104306          ERROR   306         ;BR4 CAME THRU ON PIR4
4384          ;
4385          ;*****
4386          ;SECTION 6- PIR 5 AND BR4
4387 024376 012767 024420 154504 15$:  MOV     #16$, $LPERR ;SETUP ERROR LOOP
4388 024404 012737 024442 000064    MOV     #17$, @#TPVEC ;SETUP BR4 VEC
4389 024412 012737 024454 000240    MOV     #18$, @#PIRQVEC ;SETUP PIRQ VEC
4390 024420 012706 001100          MOV     #STACK, SP   ;INITIALIZE THE SP
4391 024424 012737 020000 177772    MOV     #BIT13, @#PIRQ ;SET PIR LEVEL 5
4392 024432 004767 001574          JSR     PC, LEVEL4   ;GO GET BR4
4393 024436 000233          SPL     3            ;SET CPU AT 3
4394 024440 000240          NOP                ;REQUIRED BECAUSE OF SPL
4395          ;FAILURE, BR4 CAME THRU
4396 024442 005077 154474 17$:  CLR     @STPS        ;CLEAR BR4 INT FLAG
4397 024446 005037 177772          CLR     @#PIRQ      ;CLEAR PIR5
4398 024452 104307          ERROR   307         ;BR4 CAME THRU ON PIR5
4399          ;
4400          ;*****
4401          ;SECTION 7- BR5 AND BR4
4402          ;
4403 024454 005037 177772 18$:  CLR     @#PIRQ      ;CLEAR PIR LEVEL 5
4404 024460 005077 154456          CLR     @STPS        ;CLEAR PRINTER INT FLAG
4405 024464 005705          TST     R5           ;IS BR 5 TESTING DISABLED?
4406 024466 001040          BNE     23$         ;BRANCH IF YES
4407 024470 012767 024536 154412 20$:  MOV     #21$, $LPERR ;SETUP ERROR LOOP
4408 024476 012737 024556 000064    MOV     #22$, @#TPVEC ;SETUP BR4 VECTOR
4409 024504 016700 154516          MOV     INT5VEC, R0  ;GET BR5 VECTOR
4410 024510 012720 024570          MOV     #23$, (R0)+ ;PUT ADDRESS OF 23$ IN VECTOR
4411 024514 032737 000020 177776    BIT     #BIT4, @#PSW ;IS T BIT ON?
4412 024522 001403          BEQ     73$         ;BRANCH IF NO
4413 024524 012710 000360          MOV     #360, (R0)  ;SETUP VECTOR PSW
4414 024530 000402          BR      21$
4415 024532 012710 000340 73$:  MOV     #PR7, (R0)  ;PUT PR7 IN VECTOR+2
4416 024536 012706 001100 21$:  MOV     #STACK, SP  ;INITIALIZE THE SP
4417 024542 004767 001474          JSR     PC, LEVEL5   ;GO GET BR5
4418 024546 004767 001460          JSR     PC, LEVEL4   ;GO GET BR4
4419 024552 000233          SPL     3            ;SET CPU AT LEVEL 3
4420 024554 000240          NOP                ;REQUIRED BECAUSE OF SPL
4421          ;FAILURE, BR4 CAME IN
4422 024556 004767 001526 22$:  JSR     PC, KILBR5   ;GET RID OF BR 5
4423 024562 005077 154354          CLR     @STPS        ;CLEAR BR4 INT ENABLE
4424 024566 104310          ERROR   310         ;BR4 CAME THRU ON BR5
4425          ;
4426          ;*****
```

```
4427  
4428 024570 004767 001514  
4429 024574 012767 024616 154306  
4430 024602 012737 024640 000064  
4431 024610 012737 024652 000240  
4432 024616 012706 001076  
4433 024622 012737 040000 177772  
4434 024630 004767 001376  
4435 024634 000233  
4436 024636 000240  
4437  
4438 024640 005037 177772  
4439 024644 005077 154272  
4440 024650 104311  
4441  
4442  
4443  
4444 024652 012767 024674 154230  
4445 024660 012737 024716 000064  
4446 024666 012737 024730 000240  
4447 024674 012706 001076  
4448 024700 012737 100000 177772  
4449 024706 004767 001320  
4450 024712 000233  
4451 024714 000240  
4452  
4453 024716 005077 154220  
4454 024722 005037 177772  
4455 024726 104312  
4456  
4457  
4458  
4459 024730 005077 154206  
4460 024734 005037 177772  
4461 024740 005705  
4462 024742 001041  
4463 024744 012767 025012 154136  
4464 024752 012737 025034 000240  
4465 024760 016700 154242  
4466 024764 012720 025046  
4467 024770 032737 000020 177776  
4468 024776 001403  
4469 025000 012710 000360  
4470 025004 000462  
4471 025006 012710 000340  
4472 025012 012706 001100  
4473 025016 012737 010000 177772  
4474 025024 004767 001212  
4475 025030 000233  
4476 025032 000240  
4477  
4478 025034 004767 001250  
4479 025040 005037 177772  
4480 025044 104313  
4481  
4482
```

```
:SECTION 8- PIR6 AND BR4  
23$: JSR PC,KILBR5 ;GET RID OF BR 5  
MOV #24$,SLPERR ;SETUP ERROR LOOP  
MOV #25$,@WTPVEC ;SETUP BR4 INTERRUPT VECTOR.  
MOV #26$,@WPIRQVEC ;SETUP PIRQ VECTOR  
24$: MOV #1076,SP ;INITIALIZE THE SP  
MOV #BIT14,@WPIRQ ;SET PIR LEVEL 6  
JSR PC,LEVEL4 ;GO GET BR4  
SPL 3 ;SET CPU AT LEVEL 3  
NOP ;REQUIRED BECAUSE OF SPL  
:FAILURE, BR4 CAME IN  
25$: CLR @WPIRQ ;CLEAR PIR LEVEL 6  
CLR @STPS ;CLEAR PRINTER INTER FLAG  
ERROR 311 ;BR4 CAME IN ON PIR6  
:  
:*****  
:SECTION 9- PIR 7 AND BR4  
26$: MOV #27$,SLPERR ;SETUP ERROR LOOP  
MOV #28$,@WTPVEC ;SETUP BR4 VECTOR  
MOV #29$,@WPIRQVEC ;SETUP PIRQ VECTOR  
27$: MOV #1076,SP ;SETUP THE SP  
MOV #BIT15,@WPIRQ ;SET PIR LEVEL 7  
JSR PC,LEVEL4 ;GO GET BR4  
SPL 3 ;LOWER CPU TO 3  
NOP ;REQUIRED BECAUSE OF SPL  
:FAILURE, BR4 CAME IN  
28$: CLR @STPS ;CLEAR PRINTER INTERR FLAG  
CLR @WPIRQ ;CLEAR PIR LEVEL 7  
ERROR 312 ;BR4 CAME IN ON PIR7  
:  
:*****  
:SECTION 10- BR5 AND PIR4  
29$: CLR @STPS ;CLEAR BR4 INTERR FLAG  
CLR @WPIRQ ;CLEAR PIRQ LEVEL 7  
TST R5 ;IS THERE A BR5 DEVICE?  
BNE 32$ ;BRANCH TO SECTION 11 IF NO  
MOV #30$,SLPERR ;SETUP ERROR LOOP  
MOV #31$,@WPIRQVEC ;SETUP PIRQ VECTOR  
MOV INT5VEC,R0 ;GET ADDR OF BR 5 VECTOR  
MOV #32$, (R0)+ ;SETUP BR5 VECTOR  
4467: BIT #BIT4,@WPSW ;IS T BIT ON?  
BEQ 74$ ;BRANCH IF NO  
MOV #360, (R0)  
BR 75$  
74$: MOV #PR7, (R0)  
30$: MOV #STACK,SP ;SETUP THE SP  
MOV #BIT12,@WPIRQ ;SET PIR4  
JSR PC,LEVEL5 ;GO GET BR5  
SPL 3 ;LOWER CPU TO LEVEL 3  
NOP ;REQUIRED BECAUSE OF SPL  
:FAILURE, PIR4 CAME IN  
31$: JSR PC,KILBR5 ;GET RID OF BR 5  
CLR @WPIRQ ;CLEAR PIR LEVEL 4  
ERROR 313 ;PIR4 CAME IN ON BR5  
:  
:*****
```



```
4483 :SECTION 11- BR6 AND PIR4
4484 025046 005037 177772 32$: CLR @PIRQ ;CLEAR PIR LEVEL 4
4485 025052 004767 001232 JSR PC,KILBR5 ;GET RID OF BR 5
4486 025056 032737 000400 177570 BIT #SW8,@SWR ;SWITCH 8 ON?
4487 025064 001004 BNE 80$ ;BRANCH IF YES
4488 025066 032737 000100 177570 BIT #SW6,@SWR ;IS BR6 TESTING DISABLED?
4489 025074 001003 BNE 33$ ;BRANCH IF YES
4490 025076 005767 154130 80$: TST INTER6 ;IS THERE A BR6 DEVICE?
4491 025102 001002 BNE 34$ ;BRANCH IF YES
4492 025104 005204 33$: INC R4 ;SET BR 6 DISABLE FLAG
4493 025106 000441 BR 37$ ;GO TO SECTION 12
4494 025110 012767 025156 153772 34$: MOV #35$,SLPERR ;SETUP ERROR LOOP
4495 025116 012737 025200 000240 MOV #36$,@PIRQVEC ;SETUP PIR VECTOR
4496 025124 016701 154104 MOV INT6VEC,R1 ;GET BR 6 VECTOR
4497 025130 012721 025212 MOV #37$, (R1)+ ;PU ADDRESS OF 37$ IN VECTOR
4498 025134 032737 000020 177776 BIT #BIT4,@PSW ;IS T BIT ON?
4499 025142 001403 BEQ 75$ ;BRANCH IF NO
4500 025144 012711 000360 MOV #360, (R1)
4501 025150 000402 BR 35$
4502 025152 012711 000340 75$: MOV #PR7, (R1) ;PUT PRIORITY OF 7 IN VECTOR+2
4503 025156 012706 001100 35$: MOV #STACK,SP ;INITIALIZE THE SP
4504 025162 012737 010000 177772 MOV #BIT12,@PIRQ ;SET PIR LEVEL 4
4505 025170 004767 001056 JSR PC,LEVEL6 ;GO GET BR6
4506 025174 000233 SPL 3 ;LOWER CPU TO LEVEL 3
4507 025176 000240 NOP ;REQUIRED BECAUSE OF SPL
4508 :FAILURE, PIR4 CAME IN
4509 025200 005077 154032 36$: CLR @INT6ST ;CLEAR BR6 INTERR FLAG
4510 025204 005037 177772 CLR @PIRQ ;CLEAR PIR LEVEL 4
4511 025210 104314 ERROR 314 ;PIR4 CAME IN ON BR6
4512 :
4513 :*****
4514 :SECTION 12- PIR4 AND STACK LIMIT YELLOW
4515 025212 005077 154020 37$: CLR @INT6ST ;CLEAR BR6 INTERR. FLAG
4516 025216 005037 177772 CLR @PIRQ ;CLEAR LEVEL 4
4517 025222 012767 025244 153660 MOV #38$,SLPERR ;SETUP ERROR LOOP
4518 025230 012737 025262 000240 MOV #39$,@PIRQVEC ;SETUP PIRQ VECTOR
4519 025236 012737 025302 000004 MOV #40$,@ERRVEC ;SETUP YELL ZONE VECTOR
4520 025244 012706 000376 38$: MOV #376,SP ;SETUP THE SP
4521 025250 052737 010000 177772 BIS #BIT12,@PIRQ ;SET LEVEL 4
4522 025256 000233 SPL 3 ;SET CPU AT LEVEL 3
4523 025260 011616 MOV (SP), (SP) ;EXECUTE TRAP CAUSING INSTR
4524 :FAILURE, PIR 4 CAME THRU
4525 025262 012706 001100 39$: MOV #STACK,SP ;RESET THE SP
4526 025266 012737 032654 000004 MOV #CPUSPUR,@ERRVEC ;RESTORE ERR VEC
4527 025274 005037 177772 CLR @PIRQ ;CLEAR PIR LEVEL 4
4528 025300 104315 ERROR 315 ;PIR4 CAME IN ON SL YELLOW
4529 :
4530 :*****
4531 :SECTION 13- BR5 AND PIR5
4532 025302 012706 001100 40$: MOV #STACK,SP ;RESTORE THE SP
4533 025306 012737 032654 000004 MOV #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
4534 025314 005037 177772 CLR @PIRQ ;CLEAR PIR 4
4535 025320 005705 TST R5 ;IS BR5 DISABLED?
4536 025322 001105 BNE 49$ ;BRANCH IF YES TO SECTION 16
4537 025324 012767 025346 153556 MOV #41$,SLPERR ;SETUP ERROR LOOP
4538 025332 012777 025370 153666 MOV #42$,@INT5VEC ;SETUP BR5 VECTOR
```

4539	025340	012737	025402	000240		MOV	#438,@PIRQVEC	:SETUP PIRQ VECTOR
4540	025346	012706	001076		4'S:	MOV	#1076,SP	:INITIALIZE THE SP
4541	025352	012737	020000	177772		MOV	#BIT15,@PIRQ	:SET PIR LEVEL 5
4542	025360	004767	000656			JSR	PC,LEVEL5	:GO GET BR5

```
4543 025364 000234          SPL      4          ;LOWER CPU TO LEVEL 4
4544 025366 000240          NOP          ;REQUIRED BECAUSE OF SPL
4545          ;FAILURE, BR5 CAME IN
4546 025370 004767 000714 42$: JSR      PC,KILBR5 ;GET RID OF BR 5
4547 025374 005037 177772   CLR      @#PIRQ    ;CLEAR PIR5
4548 025400 104316          ERROR    316      ;BR5 CAME IN ON PIR5
4549          ;
4550          ;*****
4551          ;SECTION 14- BR5 AND PIR6
4552 025402 012767 025424 153500 43$: MOV      #44$,SLPERR ;SETUP ERROR LOOP
4553 025410 012777 025446 153610   MOV      #45$,@INT5VEC ;SETUP BR5 VEC
4554 025416 012737 025460 000240   MOV      #46$,@#PIRQVEC ;SETUP PIRQ VECTOR
4555 025424 012706 001076 44$: MOV      #1076,SP    ;INITIALIZE THE SP
4556 025430 012737 040000 177772   MOV      #BIT14,@#PIRQ  ;SET PIR LEVEL 6
4557 025436 004767 000600   JSR      PC,LEVEL5    ;GO GET BR5
4558 025442 000234          SPL      4          ;SET CPU AT LEVEL 4
4559 025444 000240          NOP          ;REQUIRED BECAUSE OF SPL
4560          ;FAILURE, BR5 CAME IN
4561 025446 004767 000636 45$: JSR      PC,KILBR5 ;GET RID OF BR5
4562 025452 005037 177772   CLR      @#PIRQ    ;CLEAR PIR LEVEL 6
4563 025456 104317          ERROR    317      ;BR5 CAME IN ON PIR6
4564          ;
4565          ;*****
4566          ;SECTION 15- BR5 AND PIR7
4567 025460 012767 025502 153422 46$: MOV      #47$,SLPERR ;SETUP ERROR LOOP
4568 025466 012777 025524 153532   MOV      #48$,@INT5VEC ;SETUP BR5 VECTOR
4569 025474 012737 025536 000240   MOV      #49$,@#PIRQVEC ;SETUP PIRQ VECTOR
4570 025502 012706 001076 47$: MOV      #1076,SP    ;INITIALIZE THE SP
4571 025506 012737 100000 177772   MOV      #BIT15,@#PIRQ  ;SET PIR LEVEL 7
4572 025514 004767 000522   JSR      PC,LEVEL5    ;GO GET BR5
4573 025520 000234          SPL      4          ;ALLOW BRQ
4574 025522 000240          NOP          ;REQUIRED BECAUSE OF SPL
4575          ;FAILURE, BR5 CAME IN
4576 025524 004767 000560 48$: JSR      PC,KILBR5 ;GET RID OF BR5
4577 025530 005037 177772   CLR      @#PIRQ    ;CLEAR PIR LEVEL 7
4578 025534 104320          ERROR    320      ;BR5 CAME IN ON PIR7
4579          ;
4580          ;*****
4581          ;SECTION 16- PIR5 AND BR6
4582 025536 004767 000546 49$: JSR      PC,KILBR5 ;GET RID OF BR 5
4583 025542 005704          TST      R4          ;IS BR6 TESTING DISABLED?
4584 025544 001030          BNE      52$        ;BRANCH IF YES TO SECTION 17
4585 025546 012767 025570 153334   MOV      #50$,SLPERR  ;SETUP ERROR LOOP
4586 025554 012737 025614 000240   MOV      #51$,@#PIRQVEC ;SETUP PIR VECTOR
4587 025562 012777 025626 153444   MOV      #52$,@INT6VEC ;SETUP BR6 VECTOR
4588 025570 012706 001076 50$: MOV      #1076,SP    ;INITIALIZE THE SP
4589 025574 012737 020000 177772   MOV      #BIT13,@#PIRQ  ;SET IR LEVEL 5
4590 025602 004767 000444   JSR      PC,LEVEL6    ;GO GET BR6
4591 025606 000234          SPL      4          ;ALLOW BRQ
4592 025610 000240          NOP          ;REQUIRED BECAUSE OF SPL
4593          ;FAILURE, PIR 5 CAME IN
4594 025612 000000          HALT          ;DEBUG ONLY
4595 025614 005077 153416 51$: CLR      @INT6ST    ;CLEAR BR6 INTERR FLAG
4596 025620 005037 177772   CLR      @#PIRQ    ;CLEAR PIR LEVEL 5
4597 025624 104321          ERROR    321      ;PIR 5 CAME IN ON BR6
4598          ;
```

4599
4600
4601 025626 005077 153404
4602 025632 012767 025654 153250
4603 025640 012737 025672 000240
4604 025646 012737 025712 000004
4605 025654 012706 000376
4606 025660 052737 020000 177772
4607 025666 000234
4608 025670 011616
4609
4610 025672 005037 177772
4611 025676 012737 032654 000004
4612 025704 012706 001100
4613 025710 104322
4614
4615
4616
4617 025712 012706 001100
4618 025716 005704
4619 025720 001056
4620 025722 012767 025744 153160
4621 025730 012777 025766 153276
4622 025736 012737 026000 000240
4623 025744 012706 001076
4624 025750 012737 040000 177772
4625 025756 004767 000270
4626 025762 000235
4627 025764 000240
4628
4629 025766 005077 153244
4630 025772 005037 177772
4631 025776 104323
4632
4633
4634
4635 026000 012767 026022 153102
4636 026006 012777 026044 153220
4637 026014 012737 026056 000240
4638 026022 012706 001076
4639 026026 012737 100000 177772
4640 026034 004767 000212
4641 026040 000235
4642 026042 000240
4643
4644 026044 005077 153166
4645 026050 005037 177772
4646 026054 104324
4647
4648
4649
4650 026056 012767 026100 153024
4651 026064 012737 026116 000240
4652 026072 012737 026130 000004
4653 026100 012706 000376
4654 026104 052737 040000 177772

```
*****  
:SECTION 17- PIR 5 AND STACK LIMIT YELLOW  
52$: CLR @INT6ST ;CLEAR BR6 INTERR FLAG  
MOV #53$,SLPERR ;SETUP ERROR LOOP  
MOV #54$,@PIRQVEC ;SETUP PIRQVEC  
MOV #55$,@ERRVEC ;SETUP LOCATION 4  
53$: MOV #376,SP ;SETUP THE SP  
BIS #BIT13,@PIRQ ;SET LEVEL 5  
SPL 4 ;SET CPU AT 4  
MOV (SP),(SP) ;EXECUTE YEL ZONE INSTR  
:FAILURE, PIR5 CAME IN ON SL YELLOW  
54$: CLR @PIRQ ;CLEAR LEVEL 5  
MOV #CPUSPUR,@ERRVEC ;RESTORE ERRVEC  
MOV #STACK,SP ;RESET THE SP  
ERROR 322 ;PIR5 CAME IN ON SL YELLOW  
*****  
:SECTION 18- BR6 AND PIR6  
55$: MOV #STACK,SP ;RESTORE THE SP  
TST R4 ;IS BR6 TESTING DISABLED?  
BNE 61$ ;BRANCH IF YES TO SECTION 20  
MOV #56$,SLPERR ;SETUP ERROR LOOP  
MOV #57$,@INT6VEC ;SETUP BR6 INTERR VECTOR  
MOV #58$,@PIRQVEC ;SETUP PIR VECTOR  
56$: MOV #1076,SP ;SETUP THE SP  
MOV #BIT14,@PIRQ ;SET PIR LEVEL 6  
JSR PC,LEVEL6 ;GO GET BR6  
SPL 5 ;LOWER CPU TO 5  
NOP ;REQUIRED BECAUSE OF SPL  
:FAILURE, BR6 CAME IN ON PIR6  
57$: CLR @INT6ST ;CLEAR BR6 INTERR FLAG  
CLR @PIRQ ;CLEAR LEVEL 6  
ERROR 323 ;BR6 CAME IN ON PIR6  
*****  
:SECTION 19- BR6 AND PIR7  
58$: MOV #59$,SLPERR ;SETUP ERROR LOOP  
MOV #60$,@INT6VEC ;SETUP BR6 VECTOR  
MOV #61$,@PIRQVEC ;SETUP PIRQ VECTOR  
59$: MOV #1076,SP ;SETUP THE SP  
MOV #BIT15,@PIRQ ;SET PIR 7  
JSR PC,LEVEL6 ;GO GET BR6  
SPL 5 ;LOWER CPU TO 5  
NOP ;REQUIRED BECAUSE OF SPL  
:FAILURE, BR6 CAME IN ON PIR7  
60$: CLR @INT6ST ;CLEAR BR6 INTERR FLAG  
CLR @PIRQ ;CLEAR LEVEL 7  
ERROR 324 ;BR6 CAME IN ON PIR7  
*****  
:SECTION 20- PIR6 AND SL YELLOW  
61$: MOV #62$,SLPERR ;SETUP LOOP ADDRESS  
MOV #63$,@PIRQVEC ;SETUP PIRQ VECTOR  
MOV #64$,@ERRVEC ;SETUP LOCATION 4  
62$: MOV #376,SP ;SETUP THE SP  
BIS #BIT14,@PIRQ ;SET LEVEL 6
```

```
4655 026112 000235          SPL      5          ;SET CPU AT LEVEL 5
4656 026114 011616          MOV      (SP),(SP) ;EXECUTE YELL ZONE INSTR
4657          ;FAILURE, PIR 6 CAME IN ON SL YELLOW
4658 026116 005037 177772    63$: CLR      @PIRQ    ;CLEAR PIR 6
4659 026122 012706 001100    MOV      #STACK,SP ;RESTORE THE SP
4660 026126 104325          ERROR    325       ;PIR 6 CAME IN ON SL YELLOW
4661
4662          ;*****
4663          ;SECTION 21-PIR7 AND STACK LIMIT YELLOW
4664 026130 005077 153102    64$: CLR      @INT6ST ;ENSURE BR6 FLAG CLEAR
4665 026134 012767 026156 152746 MOV      #65$,@SLPERR ;SETUP ERROR LOOP
4666 026142 012737 026174 000240 MOV      #66$,@PIRQVEC ;SETUP PIRQ VECTOR
4667 026150 012737 026206 000004 MOV      #67$,@ERRVEC ;SETUP THE ERROR VECTOR
4668 026156 012706 000376    65$: MOV      #376,SP ;SETUP THE SP
4669 026162 052737 100000 177772 BIS      #BIT15,@PIRQ ;SET LEVEL 7
4670 026170 000236          SPL      6          ;SET UP AT 6
4671 026172 011616          MOV      (SP),(SP) ;CAUSE YELL ZONE
4672          ;FAILURE, PIR 7 CAME IN ON SL YELLOW
4673 026174 005037 177772    66$: CLR      @PIRQ    ;CLEAR LEVEL 7
4674 026200 012706 001100    MOV      #STACK,SP ;RESTORE THE SP
4675 026204 104326          ERROR    326       ;PIR7 CAME IN ON SL YELLOW
4676
4677          ;END OF TEST
4678 026206 012706 001100    67$: MOV      #STACK,SP ;RESET THE SP
4679 026212 005037 177772    CLR      @PIRQ    ;CLEAR LEVEL 7
4680 026216 012737 032654 000004 MOV      #CPUSPUR,@ERRVEC ;RESTORE LOCATION 4
4681 026224 005037 177766    CLR      @CPUERR  ;CLEAR OUT ERROR REG
4682 026230 000456          BR       TST50     ;CONTINUE
4683
4684          ;ROUTINES TO GET BUS REQUESTS
4685 026232 052777 000100 152702 LEVEL4: BIS      #100,@STPS ;GET A BR 4
4686 026240 000445          BR       WAIT     ;GO WAIT
4687 026242 012777 000311 152760 LEVEL5: MOV      #311,@INT5ST ;START BR 5 INTERRUPT
4688 026250 000441          BR       WAIT     ;GO WAIT
4689 026252 026727 152756 000100 LEVEL6: CMP      INT6VEC,#100 ;IS BR6 DEVICE KW11-L?
4690 026260 001004          BNE      1$       ;BRANCH IF NO
4691 026262 012777 000100 152746 MOV      #BIT6,@INT6ST ;SET INTERR BIT
4692 026270 000431          BR       WAIT     ;GO WAIT
4693 026272 012737 000001 172544 1$: MOV      #1,@PLKC ;SET KW11-P COUNTER
4694 026300 012777 000105 152730 MOV      #105,@INT6ST ;SET INTERR BIT
4695 026306 000422          BR       WAIT     ;GO WAIT
4696 026310 012777 026346 152710 KILBR5: MOV      #2$,@INT5VEC ;SET UP INTER 5 VEC
4697 026316 042777 000100 152704 BIC      #BIT6,@INT5ST ;ENSURE INTERRUPT FLAG CLEAR
4698 026324 005037 177772    CLR      @PIRQ    ;ENSURE PIRQ REG CLEAR
4699 026330 005077 152606    CLR      @STPS    ;LET BR 5 COME IN
4700 026334 000230          SPL      0
4701 026336 005000          CLR      R0
4702 026340 005200    3$: INC      R0
4703 026342 001376          BNE      3$
4704 026344 000406          BR       ENDGET   ;DON'T CHANGE STACK
4705 026346 062706 000004    2$: ADD      #4,SP ;RESTORE THE SP
4706 026352 000403          BR       ENDGET
4707 026354 005000    WAIT: CLR      R0 ;WAIT FOR
4708 026356 005200    2$: INC      R0 ;THE INTERRUPT
4709 026360 001376          BNE      2$ ;TO COME IN
4710 026362 000237    ENDGET: SPL     7
```

```
4711 026364 000207          RTS      PC          ;RETURN
4712          ;*****
4713          :TEST 50      GPR SET 1 SELECT TEST
4714          ;
4715          :      THIS TEST FIRST ENSURES THAT PSW BIT 11 SETS AND CLEARS.
4716          :      IT THEN ENSURES THAT GRAC GDREG SET 1 AND GSREG SET 1 GOES
4717          :      HIGH FOR THE MUX SELECTS LL, LH, AND HL.
4718          :      MUX SELECT HW WILL BE TESTED IN SUPERVISOR MODE.
4719          ;*****
4720 026366 000004          TST50:  SCOPE
4721 026370 012767 027016 152576      MOV      #TST51,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
4722 026376 012767 027016 152664      MOV      #TST51,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
4723 026404 052737 004000 177776      BIS      #BIT11,@PSW    ;SET REG SET SELECT BIT
4724 026412 032737 004000 177776      BIT      #BIT11,@PSW    ;DID BIT 11 SET?
4725 026420 001004          BNE      20$           ;BRANCH IF YES
4726 026422 042737 004000 177776      BIC      #BIT11,@PSW
4727 026430 104327          ERROR   327          ;EITHER PSW BIT 11 DOES NOT SET OR TMCF
4728          ;CLK HI PS DOES NOT GO LOW OR PSW BIT
4729          ;11 DOES NOT GET TO OR THRU THE DMUX.
4730 026432 105037 177777 20$:      CLR      @PSW+1        ;CLEAR BIT 11
4731 026436 032737 004000 177776      BIT      #BIT11,@PSW    ;DID BIT 11 CLEAR?
4732 026444 001402          BEQ      1$           ;BRANCH IF YES
4733 026446 104377          ERROR   377
4734 026450 000457          ;PSW BIT 11 DID NOT CLEAR
4735          ;*****
4736          :UPAD 5 TEST
4737 026452 005067 152516 1$:      CLR      $ESCAPE
4738 026456 012767 026464 152424      MOV      #10$, $LPERR
4739 026464 012702 177777 10$:      MOV      #-1,R2        ;SETUP R2
4740 026470 052737 004000 177776      BIS      #BIT11,@PSW    ;SELECT REG SET 1
4741 026476 005002          CLR      R12         ;CLEAR R12
4742 026500 042737 004000 177776      BIC      #BIT11,@PSW    ;GO BACK TO REG SET 0
4743 026506 005702          TST      R2          ;DID R2 CLEAR?
4744 026510 001002          BNE      2$           ;BRANCH IF NO
4745 026512 104330          ERROR   330         ;CLEAR R10 ACTUALLY CLEARED R2
4746 026514 000403          BR      3$
4747 026516 020202 2$:      CMP      R2,R2        ;WAS R2 SOURCE AFFECTED BY R12?
4748 026520 001401          BEQ      3$           ;BRANCH IF NO
4749 026522 104331          ERROR   331         ;R2 SRC WAS AFFECTED BY CLR R12
4750          ;*****
4751          :UPAD 0 TEST
4752 026524 012767 026532 152356 3$:      MOV      #11$, $LPERR
4753 026532 005002 11$:      CLR      R2          ;SETUP R2
4754 026534 052737 004000 177776      BIS      #BIT11,@PSW    ;GO TO REG SET 1
4755 026542 012202          MOV      (R12)+,R12    ;EXECUTE A UPAD 0 INSTRUCTION
4756 026544 042737 004000 177776      BIC      #BIT11,@PSW    ;COME BACK TO SET 0.
4757 026552 005702          TST      R2          ;DID DESTINATION CHANGE
4758 026554 001402          BEQ      4$           ;BRANCH IF NO
4759 026556 104332          ERROR   332         ;R2 DST GOT CHANGED ON (R12)+
4760 026560 000403          BR      5$
4761 026562 020202 4$:      CMP      R2,R2        ;DID SRC R2 GET CHANGED?
4762 026564 001401          BEQ      5$           ;BRANCH IF NO
4763 026566 104333          ERROR   333         ;R2 SRC GOT CHANGED ON (R12)+
4764          ;*****
4765          :UPAD 2 TEST
4766 026570 012767 026576 152312 5$:      MOV      #12$, $LPERR
```

```
4767 026576 012705 026660      12$:  MOV    #9$,R5          ;SETUP R5
4768 026602 052737 004000 177776  BIS    #BIT11,@#PSW      ;GO TO SET 1
4769 026610 012705 026634      MOV    #6$,R15          ;SETUP R15
4770 026614 020527 026634      CMP    R15,#6$          ;IS THERE STUCK BITS IN R15 SRC?
4771 026620 001401      BEQ    7$                ;BRANCH IF NO
4772 026622 000475      BR     TST51             ;:CAN'T DO THIS TEST, GO TO STUCK BIT TEST
4773 026624 020505      7$:  CMP    R15,R15        ;IS THERE STUCK BITS IN R15 DST?
4774 026626 001401      BEQ    8$                ;BRANCH IF NO
4775 026630 000472      BR     TST51             ;:CAN'T DO THIS TEST, GO TO STUCK BIT TEST
4776 026632 006400      8$:  MARK    0              ;EXECUTE A UPAD 2 INSTRUCTION
4777 026634 026705 177774      6$:  CMP    6$,R15        ;DID R15 DST GET LOADED?
4778 026640 001425      BEQ    16$              ;BRANCH IF YES
4779 026642 042737 004000 177776  BIC    #BIT11,@#PSW      ;GO BACK TO SET 0
4780 026650 012706 001100      MOV    #STACK,SP        ;RESTORE THE SP
4781 026654 104334      ERKOR  334              ;R15 DST BAD AFTER UPAD2
4782 026656 000416      BR     16$
4783 026660 022705 026660      9$:  CMP    #9$,R15        ;IS DST ALSO BAD?
4784 026664 001407      BEQ    15$              ;BRANCH IF YES
4785 026666 042737 004000 177776  BIC    #BIT11,@#PSW      ;RESTORE THE SP
4786 026674 012706 001100      MOV    #STACK,SP        ;R15 SRC DOES NOT SELECT ON UPAD2
4787 026700 104335      ERROR  335
4788 026702 000404      BR     16$
4789 026704 042737 004000 177776  15$: BIC    #BIT11,@#PSW      ;PDRD PS11 DOES NOT GET TO GRAC
4790 026712 104336      ERROR  336
4791      ;:*****
4792      ;:TEST GRAB E19(2 & 3)
4793 026714 012706 001100      16$: MOV    #STACK,SP        ;RESET THE SP
4794 026720 012767 026726 152152  MOV    #14$,SLPERR
4795 026726 042737 004000 177776  14$: BIC    #BIT11,@#PSW      ;GO BACK TO SET 0
4796 026734 005004      CLR    R4                ;ENSURE R4 CLEAR
4797 026736 052737 004000 177776  BIS    #BIT11,@#PSW      ;GO TO SET 1
4798 026744 005204      INC    R14               ;INCREMENT R14
4799 026746 042737 004000 177776  BIC    #BIT11,@#PSW      ;GO BACK TO SET 0
4800 026754 005704      TST    R4                ;WAS R4 AFFECTED?
4801 026756 001401      BEQ    17$              ;BRANCH IF NO
4802 026760 104341      ERKOR  341              ;GRAB DST SET 1 DID NOT GO LOW ON R14
4803      ;:*****
4804      ;:TEST GRAB E18(2 & 3)
4805 026762 012767 026770 152120  17$: MOV    #13$,SLPERR
4806 026770 052737 004000 177776  13$: BIS    #BIT11,@#PSW      ;GO TO SET 1
4807 026776 005004      CLR    R14               ;ENSURE EVEN ADDRESS IN R14
4808 027000 012404      MOV    (R14)+,R14        ;EXECUTE A UPAD 0
4809 027002 042737 004000 177776  BIC    #BIT11,@#PSW      ;GO BACK TO SET 0
4810 027010 005704      TST    R4                ;WAS R4 AFFECTED?
4811 027012 001401      BEQ    TST51             ;:BRANCH IF NO
4812 027014 104342      ERROR  342              ;GRAB SRC SET 1 DID NOT GO LOW ON R14
4813      ;:*****
4814      ;:*TEST 51 REGISTER SET 1 STUCK BIT TEST
4815      ;:*
4816      ;:* THIS TEST ENSURES THAT ALL BITS IN GPR'S R10 THRU R15 WORK
4817      ;:*****
4818      ;:TST51: SCOPE
4819 027016 000004      MOV    #TST52,$ESCAPE   ;SAVE START ADDRESS OF NEXT TEST
4820 027020 012767 027232 152146  MOV    #TST52,NEXTTST   ;SAVE START ADDRESS OF NEXT TEST
4821 027026 012767 027232 152234  CLR    $TMP0             ;INITIALIZE PASS COUNT
4822 027034 005067 152122
```

4823	027040	052737	004000	177776		BIS	#BIT11,@PSW	:GO TO REG SET 1
4824	027046	012737			4\$:	MOV	(PC)+,@(PC)+	:SAVE EXPECTED VALUE
4825	027050	125252			5\$:	.WORD	125252	
4826	027052	001164				.WORD	\$TMP1	:ADDRESS OF \$TMP1
4827	027054	012700				MOV	(PC)+,R10	:PUT PATTERN IN R10
4828	027056	125252			6\$:	.WORD	125252	
4829	027060	022700				CMP	(PC)+,R10	:IS R10 DST OK?
4830	027062	125252			7\$:	.WORD	125252	
4831	027064	001040				BNE	1\$:BRANCH IF NO
4832	027066	020027				CMP	R10,(PC)+	:IS R10 SRC OK?
4833	027070	125252			8\$:	.WORD	125252	
4834	027072	001034				BNE	2\$:BRANCH IF NO
4835	027074	010001				MOV	R10,R11	
4836	027076	020001				CMP	R10,R11	:IS R11 DST OK?
4837	027100	001032				BNE	1\$:BRANCH IF NO
4838	027102	020100				CMP	R11,R10	:IS R11 SRC OK?
4839	027104	001027				BNE	2\$:BRANCH IF NO
4840	027106	010102				MOV	R11,R12	
4841	027110	020002				CMP	R10,R12	:IS R12 DST OK?
4842	027112	001025				BNE	1\$:BRANCH IF NO
4843	027114	020200				CMP	R12,R10	:IS R12 SRC OK?
4844	027116	001022				BNE	2\$:BRANCH IF NO
4845	027120	010003				MOV	R10,R13	
4846	027122	020003				CMP	R10,R13	:IS R13 DST OK?
4847	027124	001020				BNE	1\$:BRANCH IF NO
4848	027126	020300				CMP	R13,R10	:IS R13 SRC OK?
4849	027130	001015				BNE	2\$:BRANCH IF NO
4850	027132	010004				MOV	R10,R14	
4851	027134	020004				CMP	R10,R14	:IS R14 DST OK?
4852	027136	001013				BNE	1\$:BRANCH IF NO
4853	027140	020400				CMP	R14,R10	:IS R14 SRC OK?
4854	027142	001010				BNE	2\$:BRANCH IF NO
4855	027144	010005				MOV	R10,R15	
4856	027146	020005				CMP	R10,R15	:IS R15 DST OK
4857	027150	001006				BNE	1\$:BRANCH IF NO
4858	027152	020500				CMP	R15,R10	:IS R15 SRC OK
4859	027154	001410				BEQ	3\$:BRANCH IF YES
4860	027156	042737	004000	177776		BIC	#BIT11,@PSW	
4861	027164	104337			2\$:	ERROR	337	:BAD BITS IN GPR SET 1 SRC
4862	027166	042737	004000	177776	1\$:	BIC	#BIT11,@PSW	
4863	027174	104340				ERROR	340	:BAD BITS IN GPR SET 1 DST
4864	027176	005767	151760		3\$:	TST	\$TMP0	:IS THIS FIRST PASS?
4865	027202	001013				BNE	TST52	:BRANCH IF NO
4866	027204	005267	151752			INC	\$TMP0	:SET PASS COUNT
4867	027210	005167	177634			COM	5\$:GO TO
4868	027214	005167	177636			COM	6\$:COMPLIMENT
4869	027220	005167	177636			COM	7\$:PATTERN
4870	027224	005167	177640			COM	8\$	
4871	027230	000706				BR	4\$:GO TEST COMPLIMENT PATTERN

4872
4873
4874
4875
4876
4877
4878

```
*****  
: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.  
:*****
```


4879 027232 000004
4880 027234 012767 027402 152026
4881 027242 112737 000120 177777
4882 027250 122737 000120 177777
4883 027256 001412
4884 027260 012767 050000 151674
4885 027266 013767 177776 151712
4886 027274 042767 000377 151704
4887 027302 104343
4888 027304 112737 000350 177777
4889 027312 122737 000350 177777
4890 027320 001412
4891 027322 012767 164000 151632
4892 027330 013767 177776 151650
4893 027336 042767 000377 151642
4894 027344 104344
4895 027346 105037 177777
4896 027352 105737 177777
4897 027356 001411
4898 027360 005067 151576
4899 027364 013767 177776 151614
4900 027372 042767 000377 151606
4901 027400 104345

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

4908
4909 027402 000004
4910 027404 012767 027652 151656
4911
4912 027412 012767 027420 151470
4913 027420 012706 001100
4914 027424 052737 040000 177776
4915 027432 005006
4916 027434 042737 040000 177776
4917 027442 005706
4918 027444 001003
4919 027446 012706 001100
4920 027452 104346
4921
4922
4923 027454 012767 027462 151426
4924 027462 012706 001100
4925 027466 052737 040000 177776
4926 027474 012706 001776
4927 027500 012606
4928 027502 042737 040000 177776
4929 027510 020627 001100
4930 027514 001403
4931 027516 012706 001100
4932 027522 104347
4933
4934

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
6\$: MOV #STACK,KSP ;INITIALIZE THE SP
BIS #BIT14,@PSW ;GO TO SUPER MODE
MOV #1776,SSP ;SETUP SUPER SP
MOV (SSP)+,SSP ;EXECUTE A UPAD 0 TYPE OPERATION
BIC #BIT14,@PSW ;GO BACK TO KERNEL
CMP KSP,#STACK ;DID KSP CHANGE?
BEQ 2\$;BRANCH IF NO
MOV #STACK,KSP ;RESTORE THE KSP
ERROR 347 ;SUPER SP DOES NOT SELECTON UPAD 0
;UPAD 5-USP

```

4935 027524 012767 027532 151356 2$: MOV #7$, $LPERR ;SETUP ERROR LOOP
4936 027532 052737 040000 177776 7$: BIS #BIT14, @#PSW ;GO TO SUPER MODE
4937 027540 012706 177777 MOV #-1, SSP ;SET THE SSP TO A NON-ZERO NUMBER
4938 027544 052737 100000 177776 BIS #BIT15, @#PSW ;GO TO USER MODE
4939 027552 005006 CLR USP ;CLEAR R17
4940 027554 042737 100000 177776 BIC #BIT15, @#PSW ;GO BACK TO SUPER MODE
4941 027562 005706 TST SSP ;DID SSP CLEAR?
4942 027564 001003 BNE 3$ ;BRANCH IF NO
4943 027566 105037 177777 CLR# @#PSW+1 ;GO BACK TO KERNEL
4944 027572 104350 ERROR 350 ;USER SP DOES NOT SELECT ON UPAD 5
4945
4946 ;UPAD 0-USP
4947 027574 012767 027602 151306 3$: MOV #8$, $LPERR ;SETUP ERROR LOOP
4948 027602 052737 040000 177776 8$: BIS #BIT14, @#PSW ;GO TO SUPER MODE
4949 027610 005006 CLR SSP ;ENSURE SSP CLEAR
4950 027612 052737 100000 177776 BIS #BIT15, @#PSW ;GO TO USER MODE
4951 027620 012706 001776 MOV #1776, USP ;SET USP TO EVEN NUMBER
4952 027624 012606 MOV (USP)+, USP ;EXECUTE A UPAD 0 TYPE INSTRUCTION
4953 027626 042737 100000 177776 BIC #BIT15, @#PSW ;GO BACK TO SUPER
4954 027634 005706 TST SSP ;DID SSP CHANGE?
4955 027636 001403 BEQ 4$ ;BRANCH IF NO
4956 027640 105037 177777 CLR# @#PSW+1 ;GO BACK TO KERNEL
4957 027644 104351 ERROR 351 ;USER SP DOES NOT SELECT ON UPAD 0
4958 027646 105037 177777 4$: CLR# @#PSW+1 ;GO BACK TO KERNEL
4959 ;CONTINUE

```

```

*****
*TEST 54 SUPER AND USER SP BIT TEST
*
* THIS TEST ENSURES THAT THE SUPERVISOR AND USER STACK POINTERS
* DON'T HAVE ANY STUCK BITS.
*****

```

```

4967 027652 000004 TST54: SCOPE
4968 027654 012767 030016 151312 MOV #TST55, $ESCAPE ;SAVE START ADDRESS OF NEXT TEST
4969 027662 012767 030016 151400 MOV #TST55, NEXTTST ;SAVE START ADDRESS OF NEXT TEST
4970 027670 005067 151270 CLR $TMP1 ;CLEAR LOOP COUNT
4971 027674 052737 040000 177776 BIS #BIT14, @#PSW ;GO TO SUPER MODE
4972 027702 012706 4$: MOV (PC)+, SP ;PUT PATTERN IN SP
4973 027704 052525 6$: .WORD 52525 ;DATA PATTERN
4974 027706 022706 CMP (PC)+, SP ;IS DST OK?
4975 027710 052525 7$: .WORD 52525 ;DATA PATTERN
4976 027712 001403 BEQ 1$ ;BRANCH IF YES
4977 027714 105037 177777 CLR# @#PSW+1 ;GO BACK TO KERNEL
4978 027720 104352 ERROR 352 ;DST FAILED
4979 027722 020627 1$: CMP SP, (PC)+ ;IS SRC OK?
4980 027724 052525 8$: .WORD 52525 ;DATA PATTERN
4981 027726 001403 BEQ 2$ ;BRANCH IF YES
4982 027730 105037 177777 CLR# @#PSW+1 ;GO BACK TO KERNEL
4983 027734 104353 ERROR 353 ;SRC FAILED
4984 027736 005737 177776 2$: TST @#PSW ;IS BIT 15 SET?
4985 027742 100404 BMI 3$ ;BRANCH IF YES
4986 027744 052737 100000 177776 BIS #BIT15, @#PSW ;GO TO USER MODE
4987 027752 000753 BR 4$ ;GO CHECK USER STACK POINTER
4988 027754 005767 151204 3$: TST $TMP1 ;IS THIS SECOND PASS?
4989 027760 001014 BNE 5$ ;BRANCH IF YES
4990 027762 005167 177716 COM 6$ ;CHANGE TEST

```

```
4991 027766 005167 177716 COM 7$ ;TO COMPLIMENT
4992 027772 005167 177726 COM 8$ ;PATTERN
4993 027776 042737 100000 177776 BIC #BIT15,@MPSW ;GO BACK TO SUPER MODE
4994 030004 005267 151154 INC $TMP1 ;SET PASS COUNT
4995 030010 000734 BR 4$ ;GO TEST COMPLIMENT PATTERN
4996 030012 105037 177777 5$: CLRB @MPSW+1 ;GO BACK TO KERNAL
4997 ;CONTINUE
4998
4999
5000 ;*****
5001 ;*TEST 55 MTP*DMO*DF6*PREVIOUS MODE(SUPER*USER)
5002 ;*
5003 ;* THIS TEST ENSURES THAT THE CORRECT SP'S ARE SELECTED WHEN EXECUTING
5004 ;* A MTP WITH DIFFERENT PREVIOUS MODE BITS SELECTED.
5005 ;*****
5005 030016 000004 TST55: SCOPE
5006 030020 012767 030252 151242 MOV #TST56,NEXTTST ;SAVE ADDRESS OF NEXT TEST
5007
5008 ;SECTION 1-POP THE KSP AND PUT IT IN THE SSP
5009 030026 012706 001100 MOV #STACK,KSP ;SETUP THE KERNEL SP
5010 030032 012746 177777 MOV #-1,-(KSP) ;PUT -1 ON THE STACK
5011 030036 005000 CLR R0 ;CLEAR ERROR COUNT
5012 030040 052737 010000 177776 BIS #BIT12,@MPSW ;SET PREVIOUS MODE SUPER
5013 030046 106606 MTPD SP ;EXECUTE INSTRUCTION UNDER TEST
5014 030050 022706 001100 CMP #STACK,KSP ;DID THE KSP DST COME OUT OK?
5015 030054 001401 BEQ 2$ ;BRANCH IF YES
5016 030056 005200 INC R0 ;SET THE ERROR COUNT
5017 030060 020627 001100 2$: CMP KSP,#STACK ;DID THE KSP SRC COME OUT OK?
5018 030064 001410 BEQ 3$ ;BRANCH IF YES
5019 030066 012706 001100 MOV #STACK,KSP ;RESTORE THE SP
5020 030072 005700 TST R0 ;DID DST ALSO FAIL?
5021 030074 001002 BNE 4$ ;BRANCH IF YES
5022 030076 104354 EPROR 354 ;KSP SRC WAS CHANGED
5023 030100 000431 BR 9$
5024 030102 104355 4$: ERROR 355 ;BOTH KSP SRC AND DST CHANGED
5025 030104 000427 BR 9$
5026 030106 005700 3$: TST R0 ;DID KSP DST CHANGE
5027 030110 001404 BEQ 5$ ;BRANCH IF NO
5028 030112 012706 001100 MOV #STACK,KSP ;RESTORE THE SP
5029 030116 104356 ERROR 356 ;KSP DST WAS CHANGED
5030 030120 000421 BR 9$
5031 030122 052737 040000 177776 5$: BIS #BIT14,@MPSW ;GO TO SUPER MODE
5032 030130 022706 177777 CMP #-1,SSP ;DID SSP LOAD OK?
5033 030134 001412 BEQ 6$ ;BRANCH IF YES
5034 030136 010667 151012 MOV SSP,$REG0 ;SAVE THE SSP
5035 030142 005006 CLR SSP ;FOR LOOPING
5036 030144 042737 040000 177776 BIC #BIT14,@MPSW ;GO BACK TO KERNEL
5037 030152 012767 177777 150776 MOV #-1,$REG1 ;SAVE EXPECTED VALUE
5038 030160 104357 ERROR 357 ;SSP DID NOT LOAD PROPERLY
5039
5040 ;SECTION 2-POP THE KSP AND PUT IT IN THE USP
5041 030162 005006 6$: CLR SSP ;ENSURE SSP CLEAR FOR ITERATIONS
5042 030164 042737 040000 177776 9$: BIC #BIT14,@MPSW ;GO BACK TO KERNEL
5043 030172 012767 030200 150710 MOV #7,$LPERR ;SET UP ERROR LOOP
5044 030200 012706 001100 7$: MOV #STACK,KSP ;SETUP THE SP
5045 030204 012746 177777 MOV #-1,-(KSP) ;PUT -1 ON THE KERNEL STACK
5046 030210 052737 030000 177776 BIS #30000,@MPSW ;SET PREVIOUS MODE USER
```

5047	030216	106606			MTPD	SP	:EXECUTE INSTRUCTION UNDER TEST
5048	030220	052737	140000	177776	BIS	#140000,@#PSW	:GO TO USER MODE
5049	030226	020627	177777		CMP	USP,#-1	:DID USER SP GET LOADED?
5050	030232	001404			BEQ	8\$:BRANCH IF YES
5051	030234	005006			CLR	JSP	:CLEAR USER SP
5052	030236	105037	177777		CLRB	@#PSW+1	:GO BACK TO KERNEL
5053	030242	104360			ERROR	360	:USER SP DID NOT LOAD ON MTP
5054	030244	005006			8\$: CLR	USP	:ENSURE USP CLEAR, IF ITERATING
5055	030246	105037	177777		CLRB	@#PSW+1	:GO BACK TO KERNEL
5056							:CONTINUE

5057
5058 :*****
5059 :*TEST 56 MFP*DMO*DF6*PREVIOUS MODE SUPER

5060 :
5061 : THE ONLY POSSIBLE WAY THIS TEST CAN FAIL IS THAT
5062 : IRCC DMO (MFP+MTP) DOES NOT GO HIGH ON MFP.
5063 : THIS WILL ONLY HAPPEN IF THE IR DECODE ROM HAS
5064 : A BAD FIELD (R(MFP+MTP)).

5065 :*****
5066 TST56: SCOPE
5067 MOV #TST57,\$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
5068 MOV #TST57,NEXTTST ;SAVE START ADDRESS OF NEXT TEST

5069 :PUSH THE SSP ONTO THE KERNEL STACK

5070	030270	012767	030334	150610	MOV	#1\$, \$LPADR	:SETUP LOOP ADR
5071	030276	012767	030334	150604	MOV	#1\$, \$LPERR	:SETUP ERROR LOOP
5072	030304	013767	177776	150656	MOV	@#PSW,\$TMP3	:SAVE PSW
5073	030312	032737	000020	177776	BIT	#BIT4,@#PSW	:IS T BIT ON?
5074	030320	001405			BEQ	1\$:BRANCH IF NO
5075	030322	012746	000340		MOV	#PR7,-(SP)	:PUT NEW PSW ON STACK
5076	030326	012746	030334		MOV	#1\$,-(SP)	:PUT RETURN ADDR ON STACK
5077	030332	000006			RTT		:TURN T BIT OFF
5078	030334	052737	040000	177776	1\$: BIS	#BIT14,@#PSW	:GO TO SUPER MODE
5079	030342	005006			CLR	SSP	:CLEAR THE SUPER SP
5080	030344	105037	177777		CLRB	@#PSW+1	:GO BACK TO KERNEL
5081	030350	012706	001100		MOV	#STACK,KSP	:SETUP THE KSP
5082	030354	012766	177777	177776	MOV	#-1,-2(KSP)	:PUT KNOWN VALUE ON STACK
5083	030362	052737	010000	177776	BIS	#BIT12,@#PSW	:SET PREVIOUS MODE SUPER
5084	030370	106506			MFPD	SP	:EXECUTE INSTRUCTION UNDER TEST
5085	030372	005716			TST	(KSP)	:DID STACK GET SUPER SP?
5086	030374	001401			BEQ	TST57	:BRANCH IF YES
5087	030376	104361			ERROR	361	:IR DECODE ROM BAD

5088 :*****
5089 :*TEST 57 UPAD 7 IN USFR MODE

5090 :
5091 : THIS TEST ENSURES THAT A UPAD 7(OCCURS IN RTI) CAUSES THE USER
5092 : STACK POINTER TO BE USED TO FETCH THE NEW PS AND PC.
5093 :
5094 : IF PDRD PS14(1) DOES NOT GET TO THE GSAM(ON GRAC)
5095 : THE TEST WILL BLOW UP.

5096 :*****
5097 TST57: SCOPE
5098 MOV #TST60,NEXTTST ;SAVE ADDRESS OF NEXT TEST
5099 MOV #STACK,KSP ;SETUP THE KERNAL SP
5100 BIS #BIT14,@#PSW ;GO TO SUPER MODE
5101 MOV #1064,SSP ;SETUP THE SSP
5102

```
5103 030426 012716 030466      MOV      #1$, (SSP)      ;PUT ADDRESS OF 1$ ON SUPER STACK
5104 030432 012766 140340 000002  MOV      #140340,2(SSP) ;PUT NEW PSW ON SUPER STACK
5105 030440 052737 100000 177776  BIS      #BIT15,@#PSW   ;GO TO USER MODE
5106 030446 012706 001060      MOV      #1060,USP     ;SETUP THE USP
5107 030452 012716 030536      MOV      #2$, (USP)    ;PUT ADDRESS OF 2$ ON USER STACK
5108 030456 012766 140340 000002  MOV      #140340,2(USP) ;PUT NEW PSW ON USER STACK
5109 030464 000002      RTI                    ;EXECUTE INSTRUCTION THAT USES UPAD 7
5110      ;FAILURE, SUPER STACK POINTER WAS READ
5111 030466 022706 001070 1$:      CMP      #1070,USP     ;DID USP DST GET WRITTEN?
5112 030472 001004      BNE      3$           ;BRANCH IF NO
5113 030474 105037 177777      CLRB    @#PSW+1       ;GO BACK TO KERNEL
5114 030500 104362      ERROR   362          ;SSP WAS READ AND USP WAS WRITTEN
5115 030502 000415      BR      2$           ;
5116 030504 042737 100000 177776 3$:      BIC      #BIT15,@#PSW ;GO TO SUPER MODE
5117 030512 022706 001070      CMP      #1070,SSP    ;WAS THE SSP WRITTEN?
5118 030516 001004      BNE      4$           ;BRANCH IF NO
5119 030520 105037 177777      CLRB    @#PSW+1       ;GO BACK TO KERNEL
5120 030524 104363      ERROR   363          ;SSP WAS READ AND WRITTEN
5121 030526 000403      BR      2$           ;
5122 030530 105037 177777 4$:      CLRB    @#PSW+1       ;GO BACK TO KERNEL
5123 030534 104364      ERROR   364          ;DON'T KNOW WHAT HAPPENED
5124      ;READ OK,NOW CHECK WRITE
5125 030536 052737 140000 177776 2$:      BIS      #140000,@#PSW ;SETUP PSW
5126 030544 022706 001064      CMP      #1064,USP    ;DID THE USP GET WRITTEN?
5127 030550 001415      BEQ     5$           ;BRANCH IF YES
5128 030552 042737 100000 177776  BIC      #BIT15,@#PSW ;GO TO SUPER MODE
5129 030560 022706 001064      CMP      #1064,SSP    ;DID THE SSP GET WRITTEN?
5130 030564 001004      BNE     6$           ;BRANCH IF NO
5131 030566 105037 177777      CLRB    @#PSW+1       ;GO TO KERNEL MODE
5132 030572 104365      ERROR   365          ;USP WAS READ BUT SSP WAS WRITTEN
5133 030574 000403      BR      5$           ;
5134 030576 105037 177777 6$:      CLRB    @#PSW+1       ;GO TO KERNEL MODE
5135 030602 104366      ERROR   366          ;USP WAS READ BUT REG 7 WAS WRITTEN
5136 030604 105037 177777 5$:      CLRB    @#PSW+1       ;GO TO KERNEL
5137      ;CONTINUE
```

```
5138
5139
5140      ;*****
5141      ;*TEST 60      SPL*SUPERVISOR MODE
5142      ;*
5143      ;*      THIS TEST ENSURES THAT SPL DOES NOT LOAD THE PSW IN SUPER+USER MODE.
5144      ;*
5145      ;*****
5145 030610 000004  TST60:  SCOPE
5146 030612 000237      SPL      7           ;SET CPU AT LEVEL 7
5147 030614 052737 040000 177776  BIS      #BIT14,@#PSW ;GO TO SUPER MODE
5148 030622 000230      SPL      0           ;TRY AND CLEAR PRIORITIES
5149 030624 042737 040000 177776  BIC      #BIT14,@#PSW ;GO BACK TO KERNEL
5150 030632 013767 177776 150346  MOV      @#PSW,$ERPSW ;SAVE PSW
5151 030640 042767 177437 150340  BIC      #^C<PR7>,$ERPSW ;MASK OFF PRIORITIES
5152 030646 022767 000340 150332  CMP      #PR7,$ERPSW  ;DID SPL CLR PRIORITIES
5153 030654 001404      BEQ     TST61        ;BRANCH IF NO
5154 030656 012767 000340 150276  MOV      #PR7,$TMP0   ;SAVE EXPECTED VALUE
5155 030664 104367      ERROR   367          ;SPL WORKED IN SUPER MODE
5156      ;*****
5157      ;*TEST 61      PSW CLOCKING TEST
5158      ;*
```

5159
5160
5161
5162
5163
5164
5165
5166
5167
5168
5169
5170
5171
5172
5173
5174
5175 030666 000004
5176 030670 013767 177776 150266
5177 030676 012767 174000 150256
5178 030704 005767 150362
5179 030710 001403
5180 030712 052767 000400 150242
5181 030720
5182 030720 012767 031432 150342
5183 030726 005067 150254
5184
5185
5186 030732 012767 030740 150150
5187 030740 012706 001072
5188 030744 012716 030760
5189 030750 012766 177757 000002
5190 030756 000002
5191 030760 005767 150306
5192 030764 001405
5193 030766 122737 000371 177777
5194 030774 001413
5195 030776 000404
5196 031000 122737 000370 177777
5197 031006 001406
5198 031010 113767 177777 150171
5199 031016 105037 177777
5200 031022 104370
5201
5202
5203 031024 012767 174000 150130
5204 031032 012767 031040 150050
5205 031040 152737 000370 177777
5206 031046 012706 001072
5207 031052 012716 031064
5208 031056 005066 000002
5209 031062 000002
5210 031064 122737 000370 177777
5211 031072 001406
5212 031074 113767 177777 150105
5213 031102 105037 177777
5214 031106 104371

..* THIS TEST ENSURES THAT ALL THE BITS IN THE PSW GET LOADED WHEN THE
..* FOLLOWING SIGNALS ARE TRUE:
..* 1) LOAD PS*KERNEL MODE, AND 2) LOAD PS*KERNEL DATI.
..*
..* IT ALSO ENSURES THAT THE PRESET LOGIC ON BITS 11, 12, 13,
..* 14, AND 15 FUNCTIONS PROPERLY.
..*
..* FOLLOWING IS A TABLE TO DESCRIBE THE PSW FOR EACH SECTION
..*
..* SECTION PSW AT START PSW ON STK(OR VECTOR) EXPEC PSW
..* 1 000XXX 174XXX 174XXX (175XXX IF KB11-E/EM)
..* 2 174XXX 000XXX 174XXX
..* 3 040XXX 134XXX 174XXX
..* 4 144XXX 000XXX 030XXX
..* 5 030XXX 000XXX 000XXX
..*
..*
TST61: SCOPE
MOV @PSW,\$TMP1 ;SAVE PSW
MOV #174000,\$TMP0 ;SAVE EXPECTED VALUE OF PSW
TST KB11E ;KB11-E OR KB11-EM PROCESSOR?
BEQ 22\$;BR IF NOT
BIS #400,\$TMP0 ;KB11-E HAS PS08 WHICH SHOULD SET ON NEXT TEST
22\$:
MOV #TST62,NEXTTST ;SAVE ADDRESS OF NEXT TEST
CLR \$ERPSW ;ENSURE \$ERPSW CLEAR
..*
..* RTI IN KERNEL MODE WITH NEW PSW<15:11>174 (175 IN KB11-E) AND OLD PSW<15:11>000
MOV #11,\$SLPERR ;SETUP ERROR LOOP
11\$: 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\$: TST KB11E ;IS THIS A KB11-E OR KB11-EM PROCESSOR?
BEQ 20\$;BR IF NOT
CMPB #371,@PSW+1 ;NEW PSW LOAD OK?
BEQ 2\$;BR IF YES
BR 21\$;OTHERWISE REPORT ERROR
20\$: CMPB #370,@PSW+1 ;DID NEW PSW LOAD OK?
BEQ 2\$;BRANCH IF YES
21\$: MOVB @PSW+1,\$ERPSW+1 ;SAVE ERROR PSW
CLRB @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 #174000,\$TMP0 ;SAVE EXPECTED VALUE OF PSW
MOV #12,\$SLPERR ;SETUP ERROR LOOP
12\$: BISB #370,@PSW+1 ;SETUP THE PSW
MOV #1072,\$SP ;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,@PSW+1 ;DID PSW STAY THE SAME
BEQ 4\$;BRANCH IF YES
MOVB @PSW+1,\$ERPSW+1 ;SAVE PSW
CLRB @PSW+1 ;GO BACK TO KERNEL
ERROR 371 ;A HIGH BYTE BIT CLEARED IN PSW

```
5215
5216
5217
5218 031110 012767 031116 147772
5219 031116 012737 040340 177776
5220 031124 012706 001072
5221 031130 012716 031144
5222 031134 012766 134000 000002
5223 031142 000002
5224 031144 122737 000370 177777
5225 031152 001406
5226 031154 113767 177777 150025
5227 031162 105037 177777
5228 031166 104372
5229
5230
5231 031170 012767 031176 147712
5232 031176 112737 000310 177777
5233 031204 012737 031222 000020
5234 031212 012737 000340 000022
5235 031220 000004
5236 031222 122737 000060 177777
5237 031230 001411
5238 031232 113767 177777 147747
5239 031240 012767 030000 147714
5240 031246 105037 177777
5241 031252 104373
5242
5243
5244 031254 012767 031262 147626
5245 031262 112737 000060 177777
5246 031270 012737 031306 000020
5247 031276 012737 030340 000022
5248 031304 000004
5249 031306 105737 177777
5250 031312 001430
5251 031314 113767 177777 147665
5252 031322 005067 147634
5253 031326 104374
5254
5255
5256 031330 012737 074357 177776
5257 031336 000005
5258 031340 013767 177776 147640
5259 031346 026727 147634 074340
5260 031354 001412
5261 031356 012767 074340 147576
5262 031364 005037 177776
5263 031370 104377
5264 031372 000461
5265 031374 012737 033270 000020
5266 031402 032767 000020 147554
5267 031410 001410
5268 031412 012706 001074
5269 031416 016716 147646
5270 031422 012766 000360 000002

:*****
:RTI IN SUPERVISOR MODE WITH NEW PSW<15:11>134 AND OLD PSW<15:11>040
:CHECKS PRESETS ON BITS 11, 12, 13, AND 15
4$: MOV #13$, $LPERR ;SETUP ERROR LOOP
13$: MOV #40340, @PSW ;GO TO SUPER AND CLEAR PREVIOUS MODE AND REG BITS
MOV #1072, SSP ;SET THE SSP
MOV #5$, (SSP) ;PUT ADDRESS OF 5$ ON STACK
MOV #134000, 2(SSP) ;PUT NEW PSW ON STACK
RTI ;EXECUTE INSTRUCTION
5$: CMPB #370, @PSW+1 ;DID ALL BITS SET?
BEQ 10$ ;BRANCH IF YES
MOVB @PSW+1, $ERPSW+1 ;SAVE PSW HIGH BYTE
CLRB @PSW+1 ;GO BACK TO KERNEL
ERROR 372 ;BITS <15,13:11> DID NOT PRESET
:*****
:IOT IN USER MODE WITH NEW PSW<15:11>000 AND OLD PSW<15:11>144
10$: MOV #14$, $LPERR ;SETUP ERROR LOOP
14$: MOVB #310, @PSW+1 ;SETUP PSW
MOV #6$, @IOTVEC ;PUT ADDRESS OF 6$ IN IOT VECTOR
MOV #PR7, @IOTVEC+2 ;PUT NEW PSW IN IOT VECTOR+2
IOT ;EXECUTE INSTRUCTION
6$: CMPB #60, @PSW+1 ;DID NEW PSW COME UP OK?
BEQ 7$ ;BRANCH IF YES
MOVB @PSW+1, $ERPSW+1 ;SAVE PSW
MOV #30000, $TMP0 ;SAVE EXPECTED VALUE
CLRB @PSW+1 ;MAKE SURE KERNEL MODE
ERROR 373 ;PSW HIGH BYTE WRONG
:*****
:IOT IN KERNEL MODE WITH NEW PSW<15:11>000 AND OLD PSW<15:11>030
7$: MOV #15$, $LPERR ;SETUP ERROR LOOP
15$: MOVB #60, @PSW+1 ;SETUP PSW
MOV #8$, @IOTVEC ;SETUP IOT VECTOR
MOV #30340, @IOTVEC+2 ;SETUP NEW PSW
IOT ;EXECUTE INSTRUCTION
8$: TSTB @PSW+1 ;IS PSW HIGH BYTE CLEAR?
BEQ 9$ ;BRANCH IF YES
MOVB @PSW+1, $ERPSW+1 ;SAVE PSW
CLR $TMP0 ;SAVE EXPECTED VALUE
ERROR 374 ;PREVIOUS MODE BITS DID NOT CLEAR
:*****
:RESET IN SUPER MODE
MOV #74357, @PSW ;SETUP PSW
RESET ;EXECUTE INSTRUCTION
MOV @PSW, $ERPSW ;SAVE PSW
CMP $ERPSW, #74340 ;WAS PSW OK?
BEQ 16$ ;BRANCH IF YES
MOV #74340, $TMP0 ;SAVE EXPECTED VALUE
CLR @PSW
ERROR 377 ;RESET AFFECTED BITS IN PSW
9$: MOV #$$SCOPE, @IOTVEC ;RESTORE IOTVE
16$: BIT #BIT4, $TMP1 ;WAS T BIT ON?
BEQ TST62 ;BRANCH IF NO
MOV #1074, SP ;SETUP THE STACK
MOV NEXTTST, (SP) ;PUT ADDRESS OF NEXT TEST ON STACK
MOV #360, 2(SP) ;SET T BIT ON STACK
```



```

5271 031430 000002
5272
5273
5274
5275
5276
5277
5278
5279
5280
5281
5282
5283 031432 000004
5284 031434 012767 031546 147626
5285 031442 012706 001100
5286 031446 012737 031474 000004
5287 031454 052737 040000 177776
5288 031462 000000
5289
5290 031464 105037 177777
5291 031470 104377
5292 031472 000417
5293 031474 032737 000200 177766
5294 031502 001010
5295 031504 013767 177766 147442
5296 031512 012767 000200 147442
5297 031520 104377
5298 031522 000420
5299 031524 005037 177766
5300 031530 005737 177766
5301 031534 001401
5302 031536 104256
5303 031540 012737 032654 000004
5304
5305
5306
5307
5308
5309
5310
5311
5312
5313 031546 000004
5314 031550 012767 032042 147512
5315 031556 012767 003706 147320
5316 031564 012767 031572 147314
5317 031572 012737 031642 000064
5318 031600 012767 031606 147302
5319 031606 012706 001100
5320 031612 000233
5321 031614 012777 000100 147320
5322 031622 012777 000015 147314
5323 031630 000001
5324
5325 031632 005077 147304
5326 031636 104377

```

```

RTI ;TURN T BIT ON
*****
*TEST 62 ILLEGAL HALT
* THIS TEST ENSURES THAT A HALT IN SUPER OR USER MODE WILL TRAP TO
* LOCATION 4.
*
* IF BEN6 FAILS EXECUTION WOULD GO TO FET.04 WHICH WOULD
* CAUSE THE HALT TO LOOK LIKE A NOP.
* IF TMCE SET CONF GOES LOW THE PROCESSOR WILL HALT AT LOCALLOCATION 1$.
*
* THE CPU ERROR REGISTER BIT 7 IS ALSO TESTED HERE.
*****
TST62: SCOPE
MOV #TST63,NEXTTST ;SAVE ADDRESS OF NEXT TEST
MOV #STACK,SP ;SETUP THE SP
MOV #1$,@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
1$: BIT #BIT7,@CPUERR ;IS ERROR REG OK?
BNE 2$ ;BRANCH IF YES
MOV @CPUERR,$REGO ;SAVE FOR TYP0UT
MOV #200,$TMPO ;SAVE EXPECTED VALUE
ERROR 377 ;BIT 7 DID NOT SET
420
2$: CLR @CPUERR ;CLEAR BIT 7
TST @CPUERR ;DID BIT 7 CLEAR?
BEQ 3$ ;BRANCH IF YES
ERROR 256 ;BIT 7 DID NOT CLEAR
3$: MOV #CPUSPUR,@ERRVEC ;RESTORE ERRVEC
;CONTINUE
*****
*TEST 63 WAIT
*
* THIS TEST ENSURES THAT THE WAIT INSTRUCTION WORKS PROPERLY.
* IT FIRST EXECUTES WITH A LEVEL 4 INTERRUPT.
* THEN THE T BIT IS ENABLED TO ENSURE THAT THE INTERRUPT
* OCCURS AND NOT THE T BIT TRAP.
*****
TST63: SCOPE
MOV #TST64,NEXTTST ;SAVE ADDRESS OF NEXT TEST
MOV #^D1990,$ICNT ;ADJUST ITERATION COUNT
MOV #10$, $LPADR ;SETUP LOOP ADR
10$: MOV #2$,@TPVEC ;SETUP TELEPRINTER VECTOR
MOV #1$, $LPERR ;SETUP ERROR LOOP
1$: 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
ERROR 377 ;WAIT INSTRUCTION DID NOT

```



```
5327 031640 000440          440          ;WAIT FOR INTERRUPT
5328
5329 031642 012767 031664 147240 ;TEST OK, NO TRY WITH T BIT
5330 031650 012737 031720 000014 2$: MOV #3$, $LPERR ;SETUP ERROR LOOP
5331 031656 012737 031742 000064   MOV #4$, @TBITVEC ;SETUP T BIT VECTOR
5332 031664 012706 001100   MOV #5$, @TPVEC ;SETUP TELEPRINTER VECTOR
5333 031670 012746 000020 3$: MOV #STACK, SP ;INITIALIZE THE SP
5334 031674 012746 031716   MOV #20, -(SP) ;PUT SP ON STACK TO TURN ON T BIT
5335 031700 052777 000100 147234   MOV #6$, -(SP) ;PUT RETURN ADDRESS ON STACK
5336 031706 012777 000015 147230   BIS #BIT6, @STPS ;SET PRINTER INTERRUPT BIT
5337 031714 000006   MOV #15, @STPB ;SEND CHARACTER
5338 031716 000001   RTT ;TURN T BIT ON
5339   WAIT ;EXECUTE INSTRUCTION UNDER TEST
5340 031720 005077 147216 4$: CLR @STPS ;CLEAR PRINTER STATUS
5341 031724 012706 001100   MOV #STACK, SP ;RESTORE THE SP
5342 031730 012737 032640 000014   MOV #SRTRN, @TBITVEC ;RESTORE T BIT VECTOR
5343 031736 104377   ERROR 377 ;T BIT CAUSED WAIT TO
5344 031740 000441   441 ;QUIT
5345
5346 031742 012737 032640 000014 ;TEST OK, NOW TRY PIRQ
5347 031750 012767 031772 147132 5$: MOV #SRTRN, @TBITVEC ;RESTORE T BIT VECTOR
5348 031756 012737 032016 000064   MOV #7$, $LPERR ;SETUP ERROR LOOP
5349 031764 012737 032032 000240   MOV #8$, @TPVEC ;SETUP TP VECTOR
5350 031772 012706 001100   MOV #9$, @PIRQVEC ;SETUP PIRQ VECTOR
5351 031776 012737 100000 177772 7$: MOV #STACK, SP ;SETUP THE SP
5352 032004 012777 000015 147132   MOV #BIT15, @PIRQ ;SET PIR LEVEL 7
5353 032012 000233   MOV #15, @STPB ;SEND CHARACTER
5354 032014 000001   SPL 3 ;LOWER CPU
5355   WAIT ;EXECUTE INSTRUCTION UNDER TEST
5356 032016 005077 147120 8$: CLR @STPS ;CLEAR TP STATUS
5357 032022 005037 177772   CLR @PIRQ ;CLEAR PIR LEVEL 7
5358 032026 104377   ERROR 377 ;PIRQ DID NOT CAUSE
5359 032030 000442   442 ;INTERRUPT
5360
5361 032032 005077 147104 9$: CLR @STPS ;CLEAR INTERRUPT FLAG
5362 032036 005037 177772   CLR @PIRQ ;CLEAR PIR LEVEL 7
5363   ;CONTINUE
5364
5365
5366
5367
5368
5369
5370
5371
5372
5373
5374
5375
5376
5377
5378 032042 000004
5379 032044 012767 032114 147216
5380 032052 005737 001272
5381 032056 001002
5382 032060 000167 000030

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

5383	032064			
5384	032064	012703	000002	
5385	032070	000007		
5386	032072	042700	001400	
5387	032076	020003		
5388	032100	001405		
5389	032102	010337	001164	
5390	032106	010037	001162	
5391	032112	104173		
5392	032114			
5393	032114			
5394				
5395				
5396				
5397				
5398				
5399				
5400				
5401				
5402				
5403				
5404				
5405				
5406				
5407	032114	000004		
5408	032116	012767	032354	147144
5409	032124	005767	146750	
5410	032130	001006		
5411	032132	032737	000001	177570
5412	032140	001403		
5413	032142	104400	072557	
5414	032146	000502		
5415	032150	005737	000042	
5416	032154	001077		
5417	032156	012700	166667	
5418	032162	104400	072602	
5419	032166	032737	000200	177570
5420	032174	001416		
5421	032176	012767	032220	146756
5422	032204	016746	146752	
5423				
5424	032210	104402		
5425	032212	104400	072741	
5426	032216	000005		
5427	032220	032737	000200	177570
5428	032226	001416		
5429	032230	000772		
5430	032232	012767	032254	146722
5431	032240	016746	146716	
5432				
5433	032244	104402		
5434	032246	104400	072741	
5435	032252	000005		
5436	032254	032737	000200	177570
5437	032262	001773		
5438				

```

DOMFPT:
MOV      #2,R3          ;R3 IS DATA PATTERN. BIT 1 WILL ALWAYS BE SET
MFPT     ;EXECUTE INSTRUCTION
BIC      #1400,R0       ;MASK OFF CISP AND FP BITS
CMP      R0,R3          ;MATCH?
BEQ      DONE7          ;DONE IF SO
MOV      R3,@$STMP1     ;SET UP EXPECTED (GOOD) DATA
MOV      R0,@$STMP0     ;SET UP RECIEVED (BAD) DATA
ERROR    173           ;ERROR PRINTOUT

```

```

DONE7:
MMSET:
:*****
:TEST 65      OPERATOR INTERVENTION TEST
:
:   THIS TEST ENSURES THAT THE RESET AND WAIT FLOWS PUT R0
:   AND THE PC IN THE LIGHTS. THE TEST IS ONLY EXECUTED ON
:   PASS 1 AND CAN BE DISABLED ALTOGETHER WITH SWITCH 0.
:
:   THE RESET LOOP IS STOPED BY CHANGING THE POSITION OF
:   SWITCH 7.
:
:   THE WAIT IS EXITED BY TYPING A CHARACTER
:   ON THE TERMINAL.
:*****

```

```

TST65: SCOPE
MOV      #$EOP,NEXTTST ;SAVE ADDRESS OF $EOP
TST      $PASS          ;IS THIS FIRST PASS?
BNE      1$             ;BRANCH IF NO
BIT      #SW0,@$SWR     ;IS SWITCH 0 ON?
BEQ      2$             ;BRANCH IF NO
TYPE     ,EM717         ;TYPE MESSAGE(SKIPPING TEST)
1$: BR      $EOP         ;EXIT
2$: TST      @#42        ;IS MONITOR ACT11?
BNE      $EOP           ;BRANCH IF YES
MOV      #166667,P0     ;SETUP R0
TYPE     ,EM720         ;TYPE MESSAGE
BIT      #SW7,@$SWR     ;IS SWITCH 7 ON?
BEQ      3$             ;BRANCH IF NO
MOV      #4$+2,$TMP0    ;SAVE PC
MOV      $TMP0,-(SP)    ;SAVE $TMP0 FOR TYPEOUT
;TYPE ADDRESS
;GO TYPE--OCTAL ASCII(ALL DIGITS)
TYPE     ,EM721         ;TYPE MESSAGE
4$: RESET ;EXECUTE INSTRUCTION
BIT      #SW7,@$SWR     ;HAS SWITCH 7 CHANGED?
BEQ      5$             ;BRANCH IF YES
BR       4$             ;LOOP
3$: MOV      #6$+2,$TMP0 ;SAVE PC
MOV      $TMP0,-(SP)    ;SAVE $TMP0 FOR TYPEOUT
;TYPE ADDRESS
;GO TYPE--OCTAL ASCII(ALL DIGITS)
TYPE     ,EM721         ;TYPE MESSAGE
6$: RESET ;EXECUTE INSTRUCTION
BIT      #SW7,@$SWR     ;HAS SWITCH 7 CHANGED?
BEQ      6$             ;BRANCH IF NO

```

```
5439          :WAIT TEST
5440 032264 104400 072602 5$: TYPE ,EM720          :TYPE MESSAGE
5441 032270 012767 032334 146664 MOV #7$, $TMP0      :SAVE PC
5442 032276 016746 146660 MOV $TMP0, -(SP)    :SAVE $TMP0 FOR TYPEOUT
5443          :TYPE ADDRESS
5444 032302 104402          TYPDC          :GO TYPE--OCTAL ASCII(ALL DIGITS)
5445 032304 104400 072777 TYPE ,EM722          :TYPE MESSAGE
5446 032310 005077 146624 CLR @ $TKB          :CLEAR KEYBOARD BUFFER
5447 032314 012737 032334 000060 MOV #7$, @ $TKVEC   :SETUP KEYBOARD VECTOR
5448 032322 052777 000100 146606 BIS #BIT6, @ $TKS   :SET INTERRUPT ENABLE BIT
5449 032330 000233          SPL 3          :LOWER PRIORITY FOR INTERRUPT
5450 032332 000001          WAIT          :EXECUTE INSTRUCTION
5451 032334 012737 034644 000060 7$: MOV #QUIT, @ $TKVEC  :RESTORE TKVECTOR
5452 032342 005077 146572 CLR @ $TKB          :ENSURE BUFFER CLEAR
5453 032346 152777 000100 146562 BISB #BIT6, @ $TKS  :SET INTERRUPT ENABLE FLAG
5454          :CONTINUE
5455          ;:*****
5456          .SBTTL END OF PASS ROUTINE
5457
5458          ;*INCREMENT THE PASS NUMBER ($PASS)
5459          ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
5460          ;*TYPE 'END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYYY'
5461          ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
5462          ;*IF SW12=1 INHIBIT TRACE TRAP
5463          ;*IF THERES A MONITOR GO TO IT
5464          ;*IF THERE ISN'T JUMP TO LOOP
5465
5466          $EOP:
5467 032354          SCOPE
5468 032354 000004          CLR $TSTNM          ;;ZERO THE TEST NUMBER
5469 032356 005067 146520          CLR $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
5470 032362 005067 146604          INC $PASS          ;;INCREMENT THE PASS NUMBER
5471 032366 005267 146506          BIC #100000, $PASS      ;;DON'T ALLOW A NEG. NUMBER
5472 032372 042767 100000 146500 DEC (PC)+          ;;LOOP?
5473 032400 005327          $EOPCT: .WORD 1
5474 032402 000001          BGT $DOAGN          ;;YES
5475 032404 003076          MOV (PC)+, @ (PC)+      ;;RESTORE COUNTER
5476 032406 012737          $ENDCT: .WORD 1
5477 032410 000001          TYPE ,65$          ;;TYPE ASCIZ STRING
5478 032412 032402          BR 64$          ;;GET OVER THE ASCIZ
5479 032414 104400 032422          ;;65$: .ASCIZ <12><15>/END PASS #/
5480 032420 000407          64$:
5481          MOV $PASS, -(SP)          ;;SAVE $PASS FOR TYPEOUT
5482 032440          ;;TYPE PASS NUMBER
5483 032440 016746 146434          TYPDS          ;;GO TYPE--DECIMAL ASCII WITH SIGN
5484          TYPE ,67$          ;;TYPE ASCIZ STRING
5485 032444 104410          BR 66$          ;;GET OVER THE ASCIZ
5486 032446 104400 032454          ;;67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
5487 032452 000421          66$:
5488          MOV $ERTTL, -(SP)          ;;SAVE $ERTTL FOR TYPEOUT
5489 032516          ;;TOTAL NUMBER OF ERRORS
5490 032516 016746 146370          TYPDS          ;;GO TYPE--DECIMAL ASCII WITH SIGN
5491          TYPE , $CR LF          ;;TYPE CARRIAGE RETURN, LINE FEED
5492 032522 104410          CLR $ERTTL          ;;CLEAR ERROR TOTAL
5493 032524 104400 001203
5494 032530 005067 146356
```

```
5495 032534 013700 000042 $GET42: MOV @#42,R0 ;;GET MONITOR ADDRESS
5496 032540 001420 BEQ $DOAGN ;;BRANCH IF NO MONITOR
5497 032542 005046 CLR -(SP) ;;INSURE THE 'T' BIT IS CLEAR
5498 032544 012746 032552 MOV #SCLR.T,-(SP) ;;SETUP FOR AN RTI OR RTT
5499 032550 000433 BR $RTRN ;;GO DO AN RTI OR RTT TO LOAD THE PSW
5500 ;;WITH A CLEARED 'T' BIT
5501 032552 $CLR.T: MOV #1,R3 ;;MAKE R3=1
5502 032552 012703 000001 JSR PC,CHAINQ ;;GO RESTORE MONITOR
5503 032556 004767 002120 MOV @#42,R0 ;;INSURE R0 CONTAINS THE MONITORS
5504 032562 013700 000042 BEQ $DOAGN ;;RETURN ADDRESS
5505 032566 001405 RESET ;;CLEAR THE WORLD
5506 032570 000005 $ENDAD: JSR PC,(R0) ;;GO TO MONITOR
5507 032572 004710 NOP ;;SAVE ROOM
5508 032574 000240 NOP ;;FOR
5509 032576 000240 NOP ;;ACT11
5510 032600 000240 $DOAGN: MOV @#PS,-(SP) ;;PUT THE PS ON THE STACK AND
5511 032602 BIC #20,(SP) ;;CLEAR THE 'T' BIT
5512 032602 013746 177776 BIT #BIT12,@#SWR ;;RUN WITH TRACE TRAP?
5513 032606 042716 000020 BNE 1$ ;;BR IF NO
5514 032612 032737 010000 177570 COM $TBIT ;;IS IT TIME FOR TRACE TRAP
5515 032620 001005 BMI 1$ ;;BR IF NO
5516 032622 005167 000020 BIS #20,(SP) ;;SET TRACE TRAP
5517 032626 100402 1$: MOV #SLOOP,-(SP) ;;JUMP TO START OF TEST
5518 032630 052716 000020 $RTRN: RTI ;;RETURN--THIS IS CHANGED TO
5519 032634 012746 032642 ;;AN 'RTT' IF 'RTT' IS A LEGAL
5520 032640 000002 ;;INSTRUCTION
5521
5522
5523 032642 $LOOP: JMP @#LOOP ;;RETURN
5524 032642 000137 005340 $TBIT: .WORD 0 ;;'T' BIT STATE INDICATOR
5525 032646 000000 $ENULL: .BYTE -1,-1,0 ;;NULL CHARACTER STRING
5526 032650 377 377 000 .EVEN
5527 032654
5528
5529
5530
5531
5532
5533
5534
5535 032654 011667 146236 (PUSP)R:MOV (SP),$ERRPC ;;SAVE THE ERROR PC
5536 032660 012706 001100 MOV #STACK,SP ;;RESTORE THE SP
5537 032664 104400 032672 TYPE ,65$ ;;TYPE ASCIZ STRING
5538 032670 000414 BR 64$ ;;GET OVER THE ASCIZ
5539
5540 032722 ;;65$: .ASCIZ /UNEXPECTED TRAP TO 4/<15><12>
5541 032722 032737 020000 177570 64$: BIT #BIT13,@#SWR ;;IS INHIBIT ERROR TYPEOUT ON?
5542 032730 001045 BNE 1$ ;;BRANCH IF YES
5543 032732 104400 032740 TYPE ,67$ ;;TYPE ASCIZ STRING
5544 032736 000417 BR 66$ ;;GET OVER THE ASCIZ
5545
5546 032776 ;;67$: .ASCIZ /ERRORPC TSTNUM CPUERR REG/<15><12>
5547 032776 013767 177766 146156 66$: MOV @#CPUERR,$TMP0 ;;GET CPU ERROR REG
5548 033004 116767 146072 146176 MOVB $TSTNM,$$TSTNM ;;GET TEST NUMBER
5549 033012 016746 146100 MOV $ERRPC,-(SP) ;;SAVE $ERRPC FOR TYPEOUT
5550 ;;TYPE ERROR PC
```

```

5551 033016 104402          TYP0C          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
5552 033020 104400 034640  TYPE          ,TWOSP
5553 033024 016746 146160  MOV          $$TSTNM,-(SP) ;;SAVE $$TSTNM FOR TYPEOUT
5554                                     ;;TYPE TEST NUMBER
5555 033030 104402          TYP0C          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
5556 033032 104400 034640  TYPE          ,TWOSP
5557 033036 016746 146120  MOV          $TMP0,-(SP)  ;;SAVE $TMP0 FOR TYPEOUT
5558                                     ;;TYPE ERROR REGISTER
5559 033042 104402          TYP0C          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
5560 033044 005037 177766  1$: CLR          @CPUERR    ;CLEAR CPU ERROR REG
5561 033050 016767 146214 146116 MOV          NEXTTST,$ESCAPE ;SETUP ESCAPE ADDRESS
5562 033056 104264          ERROR          264      ;MAKE THE ERROR CALL TO SYSMAC
5563 033060 011667 146032  CACHSPU:MOV      (SP),$ERRPC ;SAVE ERROR PC
5564 033064 012706 001100  MOV          #STACK,SP   ;RESTORE STACK
5565 033070 104400 033076  TYPE          ,65$      ;;TYPE ASCIIZ STRING
5566 033074 000415          BR           64$      ;;GET OVER THE ASCIIZ
5567                                     ;;65$: .ASCIIZ /UNEXPECTED TRAP TO 114/<15><12>
5568 033130          64$:
5569 033130 032737 020000 177570 BIT          #BIT13,@SWR   ;IS SWITCH 13 ON?
5570 033136 001045          BNE          1$      ;BRANCH IF YES
5571 033140 104400 033146  TYPE          ,67$      ;;TYPE ASCIIZ STRING
5572 033144 000417          BR           66$      ;;GET OVER THE ASCIIZ
5573                                     ;;67$: .ASCIIZ /ERRORPC TSTNUM MEMERR REG/<15><12>
5574 033204          66$:
5575 033204 013767 177744 145750 MOV          @MEMERR,$TMP0 ;SAVE MEMORY ERROR REG
5576 033212 116767 145664 145770 MOVB         $TSTNM,$$TSTNM ;SAVE TEST NUMBER
5577 033220 016746 145672  MOV          $ERRPC,-(SP) ;;SAVE $ERRPC FOR TYPEOUT
5578                                     ;;TYPE ERROR PC
5579 033224 104402          TYP0C          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
5580 033226 104400 034640  TYPE          ,TWOSP
5581 033232 016746 145752  MOV          $$TSTNM,-(SP) ;;SAVE $$TSTNM FOR TYPEOUT
5582                                     ;;TYPE TEST NUMBER
5583 033236 104402          TYP0C          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
5584 033240 104400 034640  TYPE          ,TWOSP
5585 033244 016746 145712  MOV          $TMP0,-(SP)  ;;SAVE $TMP0 FOR TYPEOUT
5586                                     ;;TYPE MEM ERROR REG
5587 033250 104402          TYP0C          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
5588 033252 013737 177744 177744 1$: MOV          @MEMERR,@MEMERR ;CLEAR MEMERR REG.
5589 033260 016767 146004 145706 MOV          NEXTTST,$ESCAPE ;SETUP ESCAPE ADDRESS
5590 033266 104264          ERROR          264
5591                                     ;;*****
5592
5593 .SBTTL SCOPE HANDLER ROUTINE
5594
5595 ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
5596 ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
5597 ;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
5598 ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
5599 ;*SW14=1 LOOP ON TEST
5600 ;*SW11=1 INHIBIT ITERATIONS
5601 ;*SW09=1 LOOP ON ERROR
5602 ;*SW08=1 LOOP ON TEST IN SWR<7:0>
5603 ;*CALL
5604 ;* SCOPE          ;;SCOPE=IOT
5605
5606 033270 $SCOPE:

```

```

5607 033270 006137 177570          ROL    @#SWR          ;;LOOP ON PRESENT TEST?
5608 033274 100511          BMI    $OVER         ;;YES IF SW14=1
5609          ;#####START OF CODE FOR THE XOR TESTER#####
5610 033276 000416          $XTSTR: BR    6$     ;;IF RUNNING ON THE 'XOR' TESTER CHANGE
5611          ;;THIS INSTRUCTION TO A 'NOP' (NOP=240)
5612 033300 013746 000004          MOV    @#ERRVEC,-(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
5613 033304 012737 033324 000004          MOV    #5$,@#ERRVEC  ;;SET FOR TIMEOUT
5614 033312 005737 177060          TST   @#177060      ;;TIME OUT ON XOR?
5615 033316 012637 000004          MOV    (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
5616 033322 000463          BR     $$VLAD        ;;GO TO THE NEXT TEST
5617 033324 022626          5$:    CMP   (SP)+,(SP)+ ;;CLEAR THE STACK AFTER A TIME OUT
5618 033326 012637 000004          MOV    (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
5619 033332 000423          BR     7$           ;;LOOP ON THE PRESENT TEST
5620 033334          6$:;#####END OF CODE FOR THE XOR TESTER#####
5621 033334 032737 000400 177570          BIT    #BIT08,@#SWR  ;;LOOP ON SPEC. TEST?
5622 033342 001404          BEQ   2$           ;;BR IF NO
5623 033344 123767 177570 145530          CMPB  @#SWR,$STNM   ;;ON THE RIGHT TEST?   SWR<7:0>
5624 033352 001462          BEQ   $OVER        ;;BR IF YES
5625 033354 105767 145523          2$:    TSTB  $ERFLG   ;;HAS AN ERROR OCCURRED?
5626 033360 001421          BEQ   3$           ;;BR IF NO
5627 033362 126767 145527 145513          CMPB  $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
5628 033370 101015          BHI   3$           ;;BR IF NO
5629 033372 032737 001000 177570          BIT    #BIT09,@#SWR ;;LOOP ON ERROR?
5630 033400 001404          BEQ   4$           ;;BR IF NO
5631 033402 016767 145502 145476          7$:    MOV    $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
5632 033410 000443          BR     $OVER
5633 033412 105067 145465          4$:    CLRB  $ERFLG   ;;ZERO THE ERROR FLAG
5634 033416 005067 145550          CLR   $TIMES       ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
5635 033422 000415          BR     1$         ;;ESCAPE TO THE NEXT TEST
5636 033424 032737 004000 177570          3$:    BIT    #BIT11,@#SWR ;;INHIBIT ITERATIONS?
5637 033432 001011          BNE   1$           ;;BR IF YES
5638 033434 005767 145440          TST   $PASS        ;;IF FIRST PASS OF PROGRAM
5639 033440 001406          BEQ   1$           ;;INHIBIT ITERATIONS
5640 033442 005267 145436          INC   $ICNT        ;;INCREMENT ITERATION COUNT
5641 033446 026767 145520 145430          CMP   $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
5642 033454 002021          BGE   $OVER        ;;BR IF MORE ITERATION REQUIRED
5643 033456 012767 000001 145420          1$:    MOV    #1,$ICNT  ;;REINITIALIZE THE ITERATION COUNTER
5644 033464 016767 000044 145500          MOV    $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
5645 033472 105267 145404          $$VLAD: INCB  $STNM   ;;COUNT TEST NUMBERS
5646 033476 011667 145404          MOV    (SP),$LPADR  ;;SAVE SCOPE LOOP ADDRESS
5647 033502 011667 145402          MOV    (SP),$LPERR  ;;SAVE ERROR LOOP ADDRESS
5648 033506 005067 145462          CLR   $ESCAPE      ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
5649 033512 112767 000001 145375          MOVB  #1,$ERMAX    ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
5650 033520 016737 145356 177570          $OVER: MOV    $STNM,@#DISPLAY ;;DISPLAY TEST NUMBER
5651 033526 016716 145354          MOV    $LPADR,(SP)  ;;FUDGE RETURN ADDRESS
5652 033532 000002          RTI                ;;FIXES PS
5653 033534 003720          $MXCNT: 2000.      ;;MAX. NUMBER OF ITERATIONS
5654          ;:*****
5655
5656          .SBTTL  ERROR HANDLER ROUTINE
5657
5658          ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
5659          ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
5660          ;*AND GO TO ETYPDM ON ERROR
5661          ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
5662          ;*SW15=1      HALT ON ERROR

```

```
5663          ;*          HALT CAN OCCUR BEFORE AND AFTER THE ERROR TYPEOUT
5664          ;*SW13=1    INHIBIT ERROR TYPEOUTS
5665          ;*SW10=1    BELL ON ERROR
5666          ;*SW09=1    LOOP ON ERROR
5667          ;*CALL
5668          ;*          ERROR      N          ;;ERROR EMT AND N=ERROR ITEM NUMBER
5669
5670          033536          $ERROR:
5671          033536 116767 145340 145444          MOVB  $STNM,$STNM          ;;GET TEST NUMBER FOR TYPE OUT
5672          033544 105267 145333          7$:  INCB  $ERFLG          ;;SET THE ERROR FLAG
5673          033550 001775          BEQ   7$          ;;DON'T LET THE FLAG GO TO ZERO
5674          033552 016737 145324 177570          MOV   $STNM,@DISPLAY          ;;DISPLAY TEST NUMBER AND ERROR FLAG
5675          033560 005737 177570          TST  @SWR          ;;HALT ON ERROR = 1?
5676          033564 100001          BPL  8$          ;;BRANCH IF NO
5677          033566 000000          HALT          ;;YES--HALT
5678          033570 032737 002000 177570          8$:  BIT  #BIT10,@SWR          ;;BELL ON ERROR?
5679          033576 001402          BEQ  1$          ;;NO - SKIP
5680          033600 104400 001176          TYPE ,SBELL          ;;RING BELL
5681          033604 005267 145302          1$:  INC  $ERTTL          ;;COUNT THE NUMBER OF ERRORS
5682          033610 011667 145302          MOV  (SP),$ERRPC          ;;GET ADDRESS OF ERROR INSTRUCTION
5683          033614 162767 000002 145274          SUB  #2,$ERRPC
5684          033622 117767 145270 145264          MOVB @ERRPC,$ITEMB          ;;STRIP AND SAVE THE ERROR ITEM CODE
5685          033630 032737 020000 177570          BIT  #BIT13,@SWR          ;;SKIP TYPEOUT IF SET
5686          033636 001004          BNE  2$          ;;SKIP TYPEOUTS
5687          033640 004767 000056          JSR  PC,ETYPDM          ;;GO TO USER ERROR ROUTINE
5688          033644 104400 001203          TYPE ,$CRLF
5689          033650 005737 177570          2$:  TST  @SWR          ;;HALT ON ERROR
5690          033654 100001          BPL  9$          ;;SKIP IF CONTINUE
5691          033656 000000          HALT          ;;HALT ON ERROR.
5692          033660 022767 032572 144154          9$:  CMP  #SENDAD,42          ;;ACT-11?
5693          033666 001001          BNE  3$          ;;BRANCH IF NO
5694          033670 000000          HALT          ;;YES
5695          033672 032737 001000 177570          3$:  BIT  #BIT09,@SWR          ;;LOOP ON ERROR SWITCH SET?
5696          033700 001402          BEQ  4$          ;;BR IF NO
5697          033702 016716 145202          MOV  $LPERR,(SP)          ;;FUDGE RETURN FOR LOOPING
5698          033706 005767 145262          4$:  TST  $ESCAPE          ;;CHECK FOR AN ESCAPE ADDRESS
5699          033712 001402          BEQ  5$          ;;BR IF NONE
5700          033714 016716 145254          MOV  $ESCAPE,(SP)          ;;FUDGE RETURN ADDRESS FOR ESCAPE
5701          033720          5$:
5702          033720 000002          RTI          ;;RETURN
5703          ;*****
5704          ;SBTTL ERROR MESSAGE TYPEOUT ROUTINE
5705          ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
5706          ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
5707          ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
5708          ;*****
5709          033722 026727 145262 000042          ETYPDM: CMP  $STNM,#42          ;;IS THIS TEST 42?
5710          033730 001464          BEQ  TYP5LE          ;;BRANCH IF YES
5711          033732 026727 145252 000070          CMP  $STNM,#70          ;;IS THIS TEST 70?
5712          033740 001460          BEQ  TYP5LE          ;;BRANCH IF YES
5713          033742 104400 001203          TYPE , $CRLF          ;;"CARRIAGE RETURN" & "LINE FEED"
5714          033746 005000          CLR  R0          ;;PICKUP THE ITEM INDEX
5715          033750 153700 001114          BISB @ITEMB,R0
5716          033754 126727 145134 000377          CMPB $ITEMB,#377          ;;IS MESSAGE NUMBER OVER 377
5717          033762 001006          BNE  1$          ;;BRANCH IF NO
5718          033764 016600 000002          MOV  2(SP),R0          ;;GET RETURN PC FROM STACK
```

```

5719 033770 011000      MOV      (R0),R0      ;GET MESSAGE NUMBER
5720 033772 062766 000002 000002  ADD      #2,2(SP)    ;CORRECT PC
5721 034000 005300      DEC      R0          ;ADJUST THE INDEX SO THAT IT WILL
5722 034002 010001      MOV      R0,R1
5723 034004 070127 000006  MUL      #6,R1      ;:WORK FOR THE ERROR TABLE
5724 034010 010100      MOV      R1,R0
5725 034012 062700 001274  ADD      #SERRTB,R0 ;:FORM TABLE POINTER
5726 034016 012067 000004  MOV      (R0)+,2$   ;:PICKUP 'ERROR MESSAGE' POINTER
5727 034022 001404      BEQ     3$          ;:SKIP TIMEOUT IF NO POINTER
5728 034024 104400      TYPE   3$          ;:TYPE THE 'ERROR MESSAGE'
5729 034026 000000      .WORD  0          ;:'ERROR MESSAGE' POINTER GOES HERE
5730 034030 104400 001203  TYPE   ,SCLRF      ;:'CARRIAGE RETURN' & 'LINE FEED'
5731 034034 012067 000004  MOV      (R0)+,4$   ;:PICKUP 'DATA HEADER' POINTER
5732 034040 001404      BEQ     5$          ;:SKIP TIMEOUT IF 0
5733 034042 104400      TYPE   5$          ;:TYPE THE 'DATA HEADER'
5734 034044 000000      .WORD  0          ;:'DATA HEADER' POINTER GOES HERE
5735 034046 104400 001203  TYPE   ,SCLRF      ;:'CARRIAGE RETURN' & 'LINE FEED'
5736 034052 011000      MOV      (R0),R0   ;:PICKUP 'DATA TABLE' POINTER
5737 034054 001003      BNE     7$          ;:GO TYPE THE DATA
5738 034056 104400 001203  TYPE   ,SCLRF      ;:'CARRIAGE RETURN' & 'LINE FEED'
5739 034062 000207      RTS     PC         ;:RETURN
5740 034064      7$:
5741 034064 013046      MOV      @ (R0)+,-(SP) ;:SAVE @ (R0)+ FOR TIMEOUT
5742 034066 104402      TYPOC      ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
5743 034070 005710      TST      (R0)      ;:IS THERE ANOTHER NUMBER?
5744 034072 001771      BEQ     6$          ;:BR IF NO
5745 034074 104400 034640  TYPE   ,TWOSP      ;:TYPE TWO(2) SPACES
5746 034100 000771      BR      7$          ;:LOOP
5747
5748 ;:*****
5749 ;SBTTL STACK LIMIT TEST TYPE OUT ROUTINE
5750 ;* THIS ROUTINE TYPES THE ADDRESS AND STACK LIMIT REGISTER
5751 ;*VALUES THAT FAILED IN TEST 153 OR 201 IN OCTAL AND BINARY.
5752 ;:*****
5752 034102 104400      TYPSE: TYPE      ;:TYPE
5753 034104 055262      EM263      ;:ERROR MESSAGE
5754 034106 104400      TYPE      ;:TYPE
5755 034110 055451      DH263      ;:DATA HEADER
5756 034112 016727 145000  MOV      $ERRPC,(PC)+ ;:GET ERROR PC
5757 034116 000000      .WORD  0          ;:ERROR PC
5758 034120 016746 177772  MOV      1$,-(SP)   ;:SAVE 1$ FOR TIMEOUT
5759      ;:TYPE IT
5760 034124 104402      TYPOC      ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
5761 034126 104400      TYPE      ;:TYPE TWO SPACES
5762 034130 034640      TWOSP
5763 034132 016727 145052  MOV      $STSTNM,(PC)+ ;:GET TEST NUMBER
5764 034136 000000      .WORD  0          ;:TEST NUMBER
5765 034140 016746 177772  MOV      2$,-(SP)   ;:SAVE 2$ FOR TIMEOUT
5766      ;:TYPE IT
5767 034144 104402      TYPOC      ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
5768 034146 104400      TYPE      ;:TYPE CR LF
5769 034150 001203      $CLRF
5770 034152 104400      TYPE      ;:TYPE DATA HEADER
5771 034154 055476      DH263A
5772 034156 005067 000020  CLR      $TYPEE    ;:INITIALIZE $TYPEE

```


5773	034162	005001			STYPE:	CLR	R1	:ENSURE R1 CLEAR
5774	034164	005067	000134			CLR	SLDATA	:
5775	034170	005067	000174			CLR	SPDATA	:
5776	034174	012700	120000			MOV	#120000,R0	:GET ERROR DATA BUFFER POINTER
5777	034200	122710			BUFFDP:	CMPB	(PC)+,(R0)	:IS THIS CORRECT TYPE?
5778	034202	000001			\$TYPEE:	.WORD	1	
5779	034204	001417				BEQ	TYPEHE	:BRANCH IF YES, GO TYPE IT
5780	034206	062700	000004			ADD	#4,R0	:STEP R0 TO NEXT ENTRY
5781	034212	020067	144736		NEXTEN:	CMP	R0,\$REGO	:LAST ENTRY?
5782	034216	001401				BEQ	2\$:BRANCH IF YES
5783	034220	000767				BR	BUFFDP	:GO CHECK NEXT ENTRY
5784	034222	000241			2\$:	CLC		:ENSURE C CLEAR
5785	034224	062767	000002	177750		ADD	#2,\$TYPEE	:SELECT NEXT ERROR TYPE
5786	034232	026727	177744	000014		CMP	\$TYPEE,#14	:IS TYPE ROUTINE DONE?
5787	034240	001350				BNE	STYPE	:BRANCH IF NO
5788	034242	000467				BR	\$\$DONE	:RETURN
5789					:OUTPUT	THE	TYPE	HEADER IF FIRST
5790	034244	005701			TYPEHE:	TST	R1	:ERROR OF THIS TYPE
5791	034246	001014				BNE	4\$:FIRST ERROR OF THIS TYPE?
5792	034250	016701	177726			MOV	\$TYPEE,R1	:BRANCH IF NO
5793	034254	062701	056060			ADD	#INDEX,R1	:GET TYPE NUMBER
5794	034260	011167	000006			MOV	(R1),3\$:GET ADDRESS OF TYPE HEADER ADDRESS
5795	034264	104400	034622			TYPE	,SP16	:GET ADDRESS OF HEADER
5796	034270	104400				TYPE		:TYPE SPACES
5797	034272	000000			3\$:	.WORD		:GO TYPE THE HEADER
5798	034274	104400				TYPE		:ADDRESS OF HEADER
5799	034276	001203				\$CRLF		:TYPE A CRLF
5800					:OUTPUT	THE	DATA	
5801	034300	126067	000001	000017	4\$:	CMPB	1(R0),SLDATA+1	:IS SL DATA SAME AS LAST SL DATA?
5802	034306	001005				BNE	8\$:BRANCH IF NO
5803	034310	104400				TYPE		:TYPE 38 SPACES
5804	034312	034600				SP38		
5805	034314	062700	000002			ADD	#2,R0	:STEP R0 TO SP DATA
5806	034320	000422				BR	SPDATA-2	:GO TO SP DATA OUTPUT
5807	034322	012027			8\$:	MOV	(R0)+,(PC)+	:GET SL DATA
5808	034324	000000			SI DATA:	.WORD		
5809	034326	042767	000377	177770		BIC	#377,SLDATA	:MASK HIGH BYTE
5810	034334	016746	177764			MOV	SLDATA,-(SP)	:SAVE SLDATA FOR TYPEOUT
5811								:TYPE IT
5812	034340	104402				TYPOC		:GO TYPE--OCTAL ASCII(ALL DIGITS)
5813	034342	016746	177756			MOV	SLDATA,-(SP)	:PUT DATA ON STACK
5814	034346	004767	000150			JSR	PC,TYPEBN	:GO TYPE BINARY
5815	034352	021067	000012			CMP	(R0),SPDATA	:IS SP DATA SAME AS LAST SP DATA?
5816	034356	001003				BNE	9\$:BRANCH IF NO
5817	034360	062700	000002			ADD	#2,R0	:STEP R0 TO NEXT SL DATA
5818	034364	000413				BR	TERM	:GO TERMINATE LINE
5819	034366	012027			9\$:	MOV	(R0)+,(PC)+	:GET SP ERROR DATA
5820	034370	000000			SPDATA:	.WORD		
5821	034372	104400	034636			TYPE	.FOURSP	:TYPE FOUR SPACES
5822	034376	016746	177766			MOV	SPDATA,-(SP)	:SAVE SPDATA FOR TYPEOUT
5823								:GO TYPE IT IN OCTAL
5824	034402	104402				TYPOC		:GO TYPE--OCTAL ASCII(ALL DIGITS)
5825	034404	016746	177760			MOV	SPDATA,-(SP)	:PUT DATA ON STACK
5826	034410	004767	000106			JSR	PC,TYPEBN	:GO TYPE BINARY
5827	034414	104400			TERM:	TYPE		:TYPE A CR LF
5828	034416	001203				\$CRLF		

```

5829 034420 000674          BR      NEXTEN      ;GO CHECK NEXT ENTRY
5830 034422 026727 144526 157774 $$$DONE: CMP      $REGO,#157774 ;DID BUFFER OVERFLOW?
5831 034430 001033          BNE      1$          ;BRANCH IF NO
5832 034432 104400 034440          TYPE     ,65$       ;:TYPE ASCIZ STRING
5833 034436 000430          BR      64$         ;:GET OVER THE ASCIZ
5834          ;:65$: .ASCIZ <15><12>/PROBABLY MORE ERRORS BUT BUFFER OVERFLOWED/<15><12>
5835 034520 64$:
5836 034520 000207          1$:      RTS      PC          ;RETURN TO $ERROR
5837
5838 034522 016602 000002          TYPFBN: MOV      2(SP),R2      ;GET NUMBER TO BE TYPED
5839 034526 104400          TYPE     ;TYPE FOUR SPACES
5840 034530 034636          FOURSP
5841 034532 012703 000010          MOV      #10,R3         ;SETUP SOB COUNT
5842 034536 112767 000060 000030 3$:      MOVB     #60,BINARY      ;PUT ASCII 0 IN LOCATION BINARY
5843 034544 006102          ROL      R2            ;GET BIT TO BE TYPED
5844 034546 103005          BCC      1$          ;BRANCH IF IT IS 0
5845 034550 005567 000020          ADC      BINARY       ;MAKE IT ASCII
5846
5847 034554 104400 034574          TYPE     ,BINARY      ;GO TYPE IT
5848 034560 000402          BR      2$          ;GO TO END OF LOOP
5849 034562 104400 034637          1$:      TYPE     ,THRESP    ;BIT WAS 0 TYPE 3 SPACES
5850 034566 077315          2$:      SOB      R3,3$
5851 034570 012616          MOV      (SP)+,(SP)    ;READ JUST STACK
5852 034572 000207          RTS      PC          ;RETURN
5853 034574 000 040 040 BINARY: .BYTE 0,40,40,0 ;STORAGE FOR ASCII CHARACTERS & TERMINATOR
5854 034577 000
5855 034600 020040 020040 020040 SP38: .ASCII / /
5856 034606 020040 020040 020040
5857 034614 020040 020040 020040
5858 034622 020040 020040 020040 SP16: .ASCII / /
5859 034630 020040 020040 040
5860 034635 040 FIVESP: .ASCII / /
5861 034636 040 FOURSP: .ASCII / /
5862 034637 040 THRESP: .ASCII / /
5863 034640 020040 000 TWOSP: .ASCIZ / /
5864 034644 .EVEN
5865
5866          ;:*****
5867          .SBTTL MONITOR RESTORE ROUTINE
5868          ;*THIS ROUTINE IS ENTERED BY TYPING A CHARACTER ON THE KEYBOARD
5869          ;*IF THE CHARACTER IS NOT A CTRL C EXECUTION IS RETURNED TO THE
5870          ;*TEST FOLLOWING THE ONE THAT WAS INTERRUPTED.
5871          ;*IF IT IS A CTRL C THE MONITOR IS RESTORED AND THE
5872          ;*PROCESSOR HALTS.
5873          QUIT: CLR      R3          ;NOT ENTERED THROUGH XXDP CHAIN
5874          MOV      @STKB,R0        ;GET CHARACTER
5875          BIC      #BIT7,R0        ;GET RID OF PARITY BIT
5876          CMP      #3,R0          ;WAS IT A CONTROL C?
5877          BNE      DUMMY         ;BRANCH IF NO
5878          RESET     ;CLEAR THE WORLD
5879          TYPE     ,65$       ;:TYPE ASCIZ STRING
5880          BR      64$         ;:GET OVER THE ASCIZ
5881          ;:65$: .ASCIZ /^C/<15><12>
5882          64$:
5883          CHAINQ: MOV     #^D1500,R0 ;SETUP SOB COUNT
5884          MOV     #SUBTAB+2,R1 ;GET END ADDRESS OF PROGRAM
5885          MOV     #160000,R2 ;GET TOP ADDRESS OF MONITOR

```

5885 034716 012142
5886 034720 077002
5887 034722 005303
5888 034724 001001
5889 034726 000207
5890 034730
5891 034730 104400 034736
5892 034734 000413
5893
5894 034764
5895 034764 000000
5896 034766 005077 144146
5897 034772 152777 000100 144136
5898 035000 104400 035006
5899 035004 000417
5900
5901 035044
5902 035044 000177 144220
5903
5904
5905
5906
5907 035050 016605 000002
5908 035054 162705 000002
5909 035060 005004
5910 035062 116704 144014
5911 035066 006104
5912 035070 026405 073124
5913 035074 001523
5914 035076 006004
5915 035100 005304
5916 035102 010467 144046
5917
5918
5919 035106 012767 000002 144046
5920 035114 012767 000176 144042
5921 035122 012703 000100
5922 035126 026305 073124
5923 035132 001426
5924 035134 101014
5925
5926 035136 010367 144020
5927 035142 016702 144016
5928 035146 166702 144010
5929 035152 000241
5930 035154 006002
5931 035156 060203
5932 035160 042703 000001
5933 035164 000760
5934
5935
5936 035166 010367 143772
5937 035172 010302
5938 035174 166702 143762
5939 035200 000241
5940 035202 006002

```

1$:  MOV    (R1)+,-(R2)    ;RESTORE THE MONITOR
     SOB    RC,1$
     DEC    R3
     BNE    2$
     RTS    PC

2$:  TYPE    ,65$           ;;TYPE ASCIZ STRING
     BR     64$           ;;GET OVER THE ASCIZ
;;65$: .ASCIZ /MONITOR RESTORED/<15><12><15><12>
64$:

DUMMY: HALT
      CLR    @STKB        ;CLEAR KEYBOARD BUFFER
      BISB  #BIT6,@STKS  ;RESET INTERRUPT ENABLE BIT
      TYPE  ,65$         ;;TYPE ASCIZ STRING
      BR     64$         ;;GET OVER THE ASCIZ
;;65$: .ASCIZ <15><12>/TYPE A CTRL C TO QUIT!!!!/<15><12>
64$:

      JMP    @NEXTTST    ;RETURN
;*****
;SBTTL CHECK TEST SEQUENCE ROUTINE
;*THIS ROUTINE IS CALLED IN THE SCOPE ROUTINE. IT VERIFYS
;*THAT A TEST HAS NOT BEEN SKIPPED.
SEGENC: MOV    2(SP),R5    ;GET ADDRESS OF SCOPE+2
        SUB    #2,R5      ;GET ADDRESS OF SCOPE CALL
        CLR    R4         ;ENSURE R4 CLEAR
        MOVB  $TSTNM,R4   ;GET TEST NUMBER
        ROL   R4         ;ADJUST
        CMP   ADRTAB(R4),R5 ;HAS TEST BEEN SKIPPED?
        BEQ   1$         ;BRANCH IF NO
        ROR   R4
        DEC   R4
        MOV   R4,$REG0    ;SAVE PREVIOUS TST NUM FOR TYP0UT

;FIND OUT WHICH TEST THIS IS.
        MOV   #2,$TMP0    ;SET BUFFER HEADER POINTER
        MOV   #176,$TMP1  ;SET BUFFER END POINTER
        MOV   #100,R3     ;SET R3 AT MIDDLE OF BUFFER
4$:    CMP   ADRTAB(R3),R5 ;IS IT THIS TEST?
        BEQ   2$         ;BRANCH IF YES
        BHI   3$         ;MOVE UP TABLE
;CORRECT TEST IS FURTHER DOWN TABLE
        MOV   R3,$TMP0    ;UPDATE HEADER POINTER
        MOV   $TMP1,R2    ;GET END POINTER
        SUB   $TMP0,R2    ;FIND RANGE OF REMAINING TABLE
        CLC
        ROR   R2         ;FIND MIDPOINT OF RANGE
        ADD   R2,R3       ;MAKE R3 MID POINT OF REMAINING TABLE
5$:    BIC   #BIT0,R3     ;ENSURE EVEN ADDRESS
        BR    4$         ;GO CHECK AGAIN

;CORRECT TEST IS FURTHER UP THE TABLE
3$:    MOV   R3,$TMP1    ;UPDATE END POINTER
        MOV   R3,R2      ;GET END POINTER
        SUB   $TMP0,R2    ;FIND RANGE OF REMAINING TABLE
        CLC
        ROR   R2         ;FIND MID POINT OF RANGE

```

```
5941 035204 160203          SUB      R2,R3          ;MAKE R3 MIDPOINT OF REMAINING TABLE
5942 035206 000764          BR       5$            ;MAKE EVEN ADDRESS AND CHECK AGAIN
5943
5944          ;FOUND THE CORRECT TEST
5945 035210 000241          2$: CLC
5946 035212 006003          ROR      R3            ;GET TEST NUMBER
5947 035214 110367 143662    MOVB     R3,$STSTM     ;UPDATE TEST NUMBER
5948 035220 010367 143732    MOV      R3,$REG1     ;SAVE FOR TYPEOUT
5949 035224 104400 035232    TYPE    ,65$         ;:TYPE ASCIZ STRING
5950 035230 000404          BR       64$         ;:GET OVER THE ASCIZ
5951          ;;65$: .ASCIZ /TEST /
5952          64$:
5953 035242          MOV      $REG0,-(SP)   ;:SAVE $REG0 FOR TYPEOUT
5954 035246 016746 143706    TYPOC   ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
5955 035250 104400 035256    TYPE    ,67$         ;:TYPE ASCIZ STRING
5956 035254 000426          BR       66$         ;:GET OVER THE ASCIZ
5957          ;;67$: .ASCIZ / FAILED, CAUSING ECECUTION TO GO TO TEST /
5958          66$:
5959 035332          MOV      $REG1,-(SP)   ;:SAVE $REG1 FOR TYPEOUT
5960 035336 016746 143620    TYPOC   ;:GO TYPE--OCTAL ASCII(ALL DIGITS)
5961 035340 104400 001203    TYPE    ,S$RLF
5962 035344 000207          1$: RTS      PC          ;RETURN
5963          ;:*****
5964          .SBTTL TYPE ROUTINE
5965
5966          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
5967          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
5968          ;*NOTE1:          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
5969          ;*NOTE2:          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
5970          ;*NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
5971          ;*
5972          ;*CALL:
5973          ;*1) USING A TRAP INSTRUCTION
5974          ;*      TYPE    ,MESADR          ;:MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
5975          ;*
5976          ;*OR
5977          ;*      TYPE
5978          ;*      MESADR
5979          ;*
5980          ;*2) USING A JSR INSTRUCTION
5981          ;*      MOV      PS,-(SP)        ;:PUSH PROCESSOR STATUS WORD ON THE STACK
5982          ;*      JSR      PC,$TYPE        ;:CALL TYPE ROUTINE
5983          ;*      MESADDR          ;:FIRST ADRESS OF MESSAGE
5984          ;*
5985 035346 105767 143577    $TYPE: TSTB     $TFPLG          ;:IS THERE A TERMINAL?
5986 035352 100002          BPL     1$            ;:BR IF YES
5987 035354 000000          HALT
5988 035356 000407          BR     3$            ;:HALT HERE IF NO TERMINAL
5989 035360 010046          1$: MOV      R0,-(SP)        ;:LEAVE
5990 035362 017600 000002    MOV      @2(SP),R0     ;:SAVE R0
5991 035366 112046          2$: MOVB     (R0)+,-(SP)   ;:GET ADDRESS OF ASCIZ STRING
5992 035370 001005          BNE     4$            ;:PUSH CHARACTER TO BE TYPED ONTO STACK
5993 035372 005726          TST     (SP)+         ;:BR IF IT ISN'T THE TERMINATOR
5994 035374 012600          MOV     (SP)+,R0      ;:IF TERMINATOR POP IT OFF THE STACK
5995 035376 062716 000002    3$: ADD     #2,(SP)      ;:RESTORE R0
5996 035402 000002          RTI          ;:ADJUST RETURN PC
                    ;:RETURN
```

5997 035404 122716 000011
5998 035410 001426
5999 035412 122716 000200
6000 035416 001004
6001 035420 005726
6002 035422 104400 001203
6003 035426 000757
6004 035430 004767 000056
6005 035434 126726 143510
6006 035440 001352
6007 035442 016746 143500
6008
6009 035446 105366 000001
6010 035452 002770
6011 035454 004767 000032
6012 035460 105367 000072
6013 035464 000770
6014
6015
6016
6017 035466 112716 000040
6018 035472 004767 000014
6019 035476 132767 000007 000052
6020 035504 001372
6021 035506 005726
6022 035510 000726
6023 035512 105777 143424
6024 035516 100375
6025 035520 116677 000002 143416
6026 035526 122766 000015 000002
6027 035534 001003
6028 035536 105067 000014
6029 035542 000406
6030 035544 122766 000012 000002
6031 035552 001402
6032 035554 105227
6033 035556 000000
6034 035560 000207
6035
6036
6037
6038
6039
6040
6041
6042
6043
6044
6045
6046
6047
6048
6049
6050
6051
6052

```
4$: CMPB #HT,(SP) ;;BRANCH IF <HT>
    BEQ 8$
    CMPB #CRLF,(SP) ;;BRANCH IF NOT
    BNE 5$
    TST (SP)+ ;;POP <CR><LF> EQUIV
    TYPE $CRLF
    BR 2$ ;;GET NEXT CHARACTER
5$: JSR PC,$TYPEC ;;GO TYPE THIS CHARACTER
6$: CMPB $FILLC,(SP)+ ;;IS IT TIME FOR FILLER CHARS.?
    BNE 2$ ;;IF NO GO GET NEXT CHAR.
    MOV $NULL,-(SP) ;;GET # OF FILLER CHARS. NEEDED
    ;;AND THE NULL CHAR.
7$: DECB 1(SP) ;;DOES A NULL NEED TO BE TYPED?
    BLT 6$ ;;BR IF NO--GO POP THE NULL OFF OF STACK
    JSR PC,$TYPEC ;;GO TYPE A NULL
    DECB $CHARCNT ;;DON'T COUNT THE NULL AS A CHARACTER
    BR 7$ ;;LOOP

;;HORIZONTAL TAB PROCESSOR
8$: MOVB #' ,(SP) ;;REPLACE TAB WITH SPACE
9$: JSR PC,$TYPEC ;;TYPE A SPACE
    BITB #7,$CHARCNT ;;BRANCH IF NOT AT
    BNE 9$ ;;TAB STOP
    TST (SP)+ ;;POP SPACE OFF STACK
    BR 2$ ;;GET NEXT CHARACTER
$TYPEC: TSTB @ $TPS ;;WAIT UNTIL PRINTER IS READY
        BPL $TYPEC
        MOVB 2(SP),@ $TPB ;;LOAD CHAR TO BE TYPED INTO DATA REG.
        CMPB #CR,2(SP) ;;BRANCH IF
        BNE 1$ ;;NOT <CR>
        CLRB $CHARCNT
        BR $TYPEX ;;EXIT
1$: CMPB #LF,2(SP) ;;BRANCH IF
    BEQ $TYPEX ;;<LF>
    INCB (PC)+ ;;INC SPACE
    $CHARCNT: .WORD 0 ;;COUNT
    $TYPEX: RTS PC

;;*****
.SBTTL BINARY TO OCTAL (ASCII) AND TYPE
;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
;*OCTAL (ASCII) NUMBER AND TYPE IT.
;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
;*CALL:
;* MOV NUM,-(SP) ;;NUMBER TO BE TYPED
;* TYPOS ;;CALL FOR TYPEOUT
;* .BYTE N ;;N-1 TO 6 FOR NUMBER OF DIGITS TO TYPE
;* .BYTE M ;;M=1 OR 0
;* ;;1=TYPE LEADING ZEROS
;* ;;0=SUPPRESS LEADING ZEROS
;*$TYPON---ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
;*$TYPOS OR $TYPOC
```

```

6053      ;*CALL:
6054      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
6055      ;*      TYPON      ;;CALL FOR TYPEOUT
6056      ;*
6057      ;*$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER
6058      ;*CALL:
6059      ;*      MOV      NUM,-(SP)      ;;NUMBER TO BE TYPED
6060      ;*      TYPOC      ;;CALL FOR TYPEOUT
6061
6062 035562 017646 000000      $TYPOS: MOV      @ (SP),-(SP)      ;;PICKUP THE MODE
6063 035566 116667 000001 000211  MOVB     1(SP), $OFILL      ;;LOAD ZERO FILL SWITCH
6064 035574 112667 000207      MOVB     (SP)+, $SOMODE+1    ;;NUMBER OF DIGITS TO TYPE
6065 035600 062716 000002      ADD      #2,(SP)           ;;ADJUST RETURN ADDRESS
6066 035604 000406      BR      $TYPON
6067 035606 112767 000001 000171  $TYPOC: MOVB     #1, $OFILL      ;;SET THE ZERO FILL SWITCH
6068 035614 112767 000006 000165  MOVB     #6, $SOMODE+1      ;;SET FOR SIX(6) DIGITS
6069 035622 112767 000005 000154  $TYPON: MOVB     #5, $OCNT      ;;SET THE ITERATION COUNT
6070 035630 010346      MOV      R3,-(SP)          ;;SAVE R3
6071 035632 010446      MOV      R4,-(SP)          ;;SAVE R4
6072 035634 010546      MOV      R5,-(SP)          ;;SAVE R5
6073 035636 116704 000145      MOVB     $SOMODE+1,R4      ;;GET THE NUMBER OF DIGITS TO TYPE
6074 035642 005404      NEG      R4
6075 035644 062704 000006      ADD      #6,R4             ;;SUBTRACT IT FOR MAX. ALLOWED
6076 035650 110467 000132      MOVB     R4, $SOMODE      ;;SAVE IT FOR USE
6077 035654 116704 000125      MOVB     $OFILL,R4        ;;GET THE ZERO FILL SWITCH
6078 035660 016605 000012      MOV      12(SP),R5        ;;PICKUP THE INPUT NUMBER
6079 035664 005003      CLR      R3               ;;CLEAR THE OUTPUT WORD
6080 035666 006105      1$:    ROL      R5           ;;ROTATE MSB INTO 'C'
6081 035670 000404      BR      3$
6082 035672 006105      2$:    ROL      R5           ;;GO DO MSB
6083 035674 006105      ROL      R5               ;;FORM THIS DIGIT
6084 035676 006105      ROL      R5
6085 035700 010503      MOV      R5,R3
6086 035702 006103      3$:    ROL      R3           ;;GET LSB OF THIS DIGIT
6087 035704 105367 000076      DECB     $SOMODE          ;;TYPE THIS DIGIT?
6088 035710 100016      BPL      7$              ;;BR IF NO
6089 035712 042703 177770      BIC      #177770,R3       ;;GET RID OF JUNK
6090 035716 001002      BNE      4$              ;;TEST FOR 0
6091 035720 005704      TST      R4               ;;SUPPRESS THIS 0?
6092 035722 001403      BEQ      5$              ;;BR IF YES
6093 035724 005204      4$:    INC      R4           ;;DON'T SUPPRESS ANYMORE 0'S
6094 035726 052703 000060      BIS      #'0,R3          ;;MAKE THIS DIGIT ASCII
6095 035732 052703 000040      5$:    BIS      #' ,R3       ;;MAKE ASCII IF NOT ALREADY
6096 035736 110367 000040      MOVB     R3,8$           ;;SAVE FOR TYPING
6097 035742 104400 036002      TYPE     ,8$             ;;GO TYPE THIS DIGIT
6098 035746 105367 000032      7$:    DECB     $OCNT'      ;;COUNT BY 1
6099 035752 003347      BGT      2$              ;;BR IF MORE TO DO
6100 035754 002402      BLT      6$              ;;BR IF DONE
6101 035756 005204      INC      R4               ;;INSURE LAST DIGIT ISN'T A BLANK
6102 035760 000744      BR      2$              ;;GO DO THE LAST DIGIT
6103 035762 012605      6$:    MOV      (SP)+,R5     ;;RESTORE R5
6104 035764 012604      MOV      (SP)+,R4         ;;RESTORE R4
6105 035766 012603      MOV      (SP)+,R3         ;;RESTORE R3
6106 035770 016666 000002 000004  MOV      2(SP),4(SP)      ;;SET THE STACK FOR RETURNING
6107 035776 012616      MOV      (SP)+,(SP)
6108 036000 000002      RTI

```

```

6109 036002 000
6110 036003 000
6111 036004 000
6112 036005 000
6113 036006 000000
6114
6115
6116
6117
6118
6119
6120
6121
6122
6123
6124
6125
6126
6127 036010
6128 036010 010046
6129 036012 010146
6130 036014 010246
6131 036016 010346
6132 036020 010546
6133 036022 012746 020200
6134 036026 016605 000020
6135 036032 100004
6136 036034 005405
6137 036036 112766 000055 000001
6138 036044 005000
6139 036046 012703 036224
6140 036052 112723 000040
6141 036056 005002
6142 036060 016001 036214
6143 036064 160105
6144 036066 002402
6145 036070 005202
6146 036072 000774
6147 036074 060105
6148 036076 005702
6149 036100 001002
6150 036102 105716
6151 036104 100407
6152 036106 106316
6153 036110 103003
6154 036112 116663 000001 177777
6155 036120 052702 000060
6156 036124 052702 000040
6157 036130 110223
6158 036132 005720
6159 036134 020027 000010
6160 036140 002746
6161 036142 003002
6162 036144 010502
6163 036146 000764
6164 036150 105726
    
```

```

8$: .BYTE 0 ;; STORAGE FOR ASCII DIGIT
    .BYTE 0 ;; TERMINATOR FOR TYPE ROUTINE
SOCNT: .BYTE 0 ;; OCTAL DIGIT COUNTER
$OFILL: .BYTE 0 ;; ZERO FILL SWITCH
$OMODE: .WORD 0 ;; NUMBER OF DIGITS TO TYPE
;*****
.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
; *THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
; *SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
; *NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
; *BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
; *REPLACED WITH SPACES.
; *CALL:
; * MOV NUM,-(SP) ;; PUT THE BINARY NUMBER ON THE STACK
; * TYPDS ;; GO TO THE ROUTINE
$TYPDS:
MOV R0,-(SP) ;; PUSH R0 ON STACK
MOV R1,-(SP) ;; PUSH R1 ON STACK
MOV R2,-(SP) ;; PUSH R2 ON STACK
MOV R3,-(SP) ;; PUSH R3 ON STACK
MOV R5,-(SP) ;; PUSH R5 ON STACK
MOV #20200,-(SP) ;; SET BLANK SWITCH AND SIGN
MOV 20(SP),R5 ;; GET THE INPUT NUMBER
BPL 1$ ;; BR IF INPUT IS POS.
NEG R5 ;; MAKE THE BINARY NUMBER POS.
MOVB #'-,1(SP) ;; MAKE THE ASCII NUMBER NEG.
1$: CLR R0 ;; ZERO THE CONSTANTS INDEX
MOV # $DBLK,R3 ;; SETUP THE OUTPUT POINTER
MOVB #' ,(R3)+ ;; SET THE FIRST CHARACTER TO A BLANK
2$: CLR R2 ;; CLEAR THE BCD NUMBER
MOV $DTBL(R0),R1 ;; GET THE CONSTANT
3$: SUB R1,R5 ;; FORM THIS BCD DIGIT
BLT 4$ ;; BR IF DONE
INC R2 ;; INCREASE THE BCD DIGIT BY 1
BR 3$
4$: ADD R1,R5 ;; ADD BACK THE CONSTANT
TST R2 ;; CHECK IF BCD DIGIT=0
BNE 5$ ;; FALL THROUGH IF 0
TSTB (SP) ;; STILL DOING LEADING 0'S?
BMI 7$ ;; BR IF YES
5$: ASLB (SP) ;; MSD?
BCC 6$ ;; BR IF NO
MOVB 1(SP),-1(R3) ;; YES--SET THE SIGN
6$: BIS #'0,R2 ;; MAKE THE BCD DIGIT ASCII
7$: BIS #' ,R2 ;; MAKE IT A SPACE IF NOT ALREADY A DIGIT
MOVB R2,(R3)+ ;; PUT THIS CHARACTER IN THE OUTPUT BUFFER
TST (R0)+ ;; JUST INCREMENTING
CMP R0,#10 ;; CHECK THE TABLE INDEX
BLT 2$ ;; GO DO THE NEXT DIGIT
BGT 8$ ;; GO TO EXIT
MOV R5,R2 ;; GET THE LSD
BR 6$ ;; GO CHANGE TO ASCII
8$: TSTB (SP)+ ;; WAS THE LSD THE FIRST NON-ZERO?
    
```

```

6165 036152 100003
6166 036154 116663 177777 177776
6167 036162 105013
6168 036164 012605
6169 036166 012603
6170 036170 012602
6171 036172 012601
6172 036174 012600
6173 036176 104400 036224
6174 036202 016666 000002 000004
6175 036210 012616
6176 036212 000002
6177 036214 023420
6178 036216 001750
6179 036220 000144
6180 036222 000012
6181 036224 000004
6182
6183
6184
6185
6186
6187
6188
6189
6190
6191 036234 010046
6192 036236 016600 000002
6193 036242 005740
6194 036244 111000
6195 036246 016000 036254
6196 036252 000200
6197
6198
6199
6200
6201
6202
6203
6204
6205
6206 036254
6207 036254 035346
6208 036256 035606
6209 036260 035562
6210 036262 035622
6211 036264 036010
6212
6213
6214
6215
6216
6217 036266 012737 036410 000024
6218 036274 012737 000340 000026
6219 036302 010046
6220 036304 010146
    
```

```

BPL 9$ ::BR IF NO
MOV -1(SP),-2(R3) ::YES--SET THE SIGN FOR TYPING
9$: CLRB (R3) ::SET THE TERMINATOR
MOV (SP)+,R5 ::POP STACK INTO R5
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
TYPE $DBLK ::NOW TYPE THE NUMBER
MOV 2(SP),4(SP) ::ADJUST THE STACK
MOV (SP)+,(SP)
RTI ::RETURN TO USER

$DTBL: 10000.
      1000.
      100.
      10.
$DBLK: .BLKW 4
::*****

.SBTTL TRAP DECODER

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

$TRAP: MOV R0,-(SP) ::SAVE R0
      MOV 2(SP),R0 ::GET TRAP ADDRESS
      TST -(R0) ::BACKUP BY 2
      MOV (R0),R0 ::GET RIGHT BYTE OF TRAP
      MOV $TRPAD(R0),R0 ::INDEX TO TABLE
      RTS R0 ::GO TO ROUTINE

.SBTTL TRAP TABLE

:*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,@PWRVEC ::SET FOR FAST UP
      MOV #340,@PWRVEC+2 ::PRIO:7
      MOV R0,-(SP) ::PUSH R0 ON STACK
      MOV R1,-(SP) ::PUSH R1 ON STACK
    
```



```
6221 036306 010246      MOV      R2,-(SP)      ;;PUSH R2 ON STACK
6222 036310 010346      MOV      R3,-(SP)      ;;PUSH R3 ON STACK
6223 036312 010446      MOV      R4,-(SP)      ;;PUSH R4 ON STACK
6224 036314 010546      MOV      R5,-(SP)      ;;PUSH R5 ON STACK
6225 036316 010667 000072  MOV      SP,$SAVR6      ;;SAVE SP
6226 036322 012737 036334 000024  MOV      #SPWRUP,@#PWRVEC ;;SET UP VECTOR
6227 036330 000000      HALT
6228 036332 000776      BR      .-2           ;;HANG UP
6229
6230
6231 036334 016706 000054      :POWER UP ROUTINE
6232 036340 005067 000050  $PWRUP: MOV      $SAVR6,SP      ;;GET SP
6233 036344 005267 000044  1$: CLR      $SAVR6           ;;WAIT LOOP FOR THE TTY
6234 036350 001375      INC      $SAVR6           ;;WAIT FOR THE INC
6235 036352 012605      BNE     1$              ;;OF WORD
6236 036354 012604      MOV      (SP)+,R5       ;;POP STACK INTO R5
6237 036356 012603      MOV      (SP)+,R4       ;;POP STACK INTO R4
6238 036360 012602      MOV      (SP)+,R3       ;;POP STACK INTO R3
6239 036362 012601      MOV      (SP)+,R2       ;;POP STACK INTO R2
6240 036364 012600      MOV      (SP)+,R1       ;;POP STACK INTO R1
6241 036366 012737 036266 000024  MOV      (SP)+,R0       ;;POP STACK INTO R0
6242 036374 012737 000340 000026  MOV      #SPWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
6243 036402 104400      MOV      #340,@#PWRVEC+2 ;;PRIO:7
6244 036404 036416      TYPE     $POWER          ;;REPORT THE POWER FAILURE
6245 036406 000002      $PWRMG: .WORD $POWER     ;;POWER FAIL MESSAGE POINTER
6246 036410 000000      RTI
6247 036412 000776      $ILLUP: HALT            ;;THE POWER UP SEQUENCE WAS STARTED
6248 036414 000000      BR      .-2           ;;BEFORE THE POWER DOWN WAS COMPLETE
6249 036416 005015 047520 042527  $SAVR6: 0                ;;PUT THE SP HERE
6250 036424 000122      $POWER: .ASCIZ <15><12>'POWER'
6251
6252 036426 041600 052520 052440  .EVEN
6253 036434 042116 051105 052040  MSG1:  .ASCIZ<CRLF>    'CPU UNDER TEST FOUND TO BE A '
6254 036442 051505 020124 047506
6255 036450 047125 020104 047524
6256 036456 041040 020105 020101
6257 036464 000
6258 036465 113 030502 026461  MSG3:  .ASCIZ 'KB11-B/C OR KB11-CM          '<CRLF>
6259 036472 027502 020103 051117
6260 036500 045440 030502 026461
6261 036506 046503 020040 020040
6262 036514 020040 020040 020040
6263 036522 020040 020040 020040
6264 036530 000200
6265 036532 020040 000040  MSG5:  .ASCIZ '' ''
6266 036536 047200 052117 035105  MSG6:  .ASCIZ <CRLF>'NOTE:''
6267 036544 000
6268 036546      .EVEN
6269
6270 036546 050123 020114 040506  EM1:  .ASCIZ /SPL FAILED OR PSW PRIORITY BITS STUCK/
6271 036554 046111 042105 047440
6272 036562 020122 051520 020127
6273 036570 051120 047511 044522
6274 036576 054524 041040 052111
6275 036604 020123 052123 041525
6276 036612 000113
```

6277	036614	051105	047522	020122	DH1:	.ASCII	/ERROR PC	SPL 5 PSW	SPL 2 PSW	TST NUM/<CRLF>
6278	036622	041520	020040	051440						
6279	036630	046120	032440	050040						
6280	036636	053523	020040	020040						
6281	036644	020040	051440	046120						
6282	036652	031040	050040	053523						
6283	036660	020040	052040	052123						
6284	036666	047040	046525	200						
6285	036673	011	042440	050130		.ASCIZ	/	EXPECT ACTUAL	EXPECT ACTUAL/	
6286	036700	041505	020124	041501						
6287	036706	052524	046101	020040						
6288	036714	042440	050130	041505						
6289	036722	020124	041501	052524						
6290	036730	046101	000							
6291		036734								
6292	036734	001116	001214	001206	DT1:	.EVEN				
6293	036742	001212	001162	001210		.WORD	\$ERRPC,\$PR5,\$ERPSW,\$PRC,\$TMP0,\$STSTNM,0			
6294	036750	000000								
6295	036752	051120	032440	047440	EM2:	.ASCIZ	/PR 5 OK PR 2 BAD/			
6296	036760	020113	051120	031040						
6297	036766	041040	042101	000						
6298	036773	120	020122	020062	EM3:	.ASCIZ	/PR 2 OK BUT PR 5 BAD/			
6299	037000	045517	041040	052125						
6300	037006	050040	020122	020065						
6301	037014	040502	000104							
6302	037020	040522	043103	042440	EM4:	.ASCIZ	/RACF E3 BAD(NOT GOING HIGH ON SPL)/			
6303	037026	020063	040502	024104						
6304	037034	047516	020124	047507						
6305	037042	047111	020107	044510						
6306	037050	044107	047440	020116						
6307	037056	050123	024514	000						
6308	037063	105	051122	051117	DH4:	.ASCIZ	/ERRORPC TEST NUMBER/			
6309	037070	041520	052040	051505						
6310	037076	020124	052516	041115						
6311	037104	051105	000							
6312		037110								
6313	037110	001116	001210	000000	DT4:	.EVEN				
6314	037116	040522	043103	042440	EM5:	.WORD	\$ERRPC,\$STSTNM,0			
6315	037124	020065	040502	024104		.ASCIZ	/RACF E5 BAD(AFIR04(1)*AFIR03(1)*AFIR05(1)*(RTS:CCOP))/			
6316	037132	043101	051111	032060						
6317	037140	030450	025051	043101						
6318	037146	051111	031460	030450						
6319	037154	025051	043101	051111						
6320	037162	032460	030450	025051						
6321	037170	051050	051524	041472						
6322	037176	047503	024520	000051						
6323	037204	042522	042523	020124	EM6:	.ASCIZ	/RASET DIDN'T SEND OUT INIT/			
6324	037212	044504	047104	052047						
6325	037220	051440	047105	020104						
6326	037226	052517	020124	047111						
6327	037234	052111	000							
6328	037237	122	041501	020106	EM7:	.ASCIZ	/RACF E5 BAD/			
6329	037244	032505	041040	042101						
6330	037252	000								
6331	037253	122	041501	020106	EM10:	.ASCIZ	/RACF E8 BAD/			
6332	037260	034105	041040	042101						

6333	037266	000							
6334	037267	122	041501	020106	EM11:	.ASCIZ	'RACF E17 BAD(AFI07(0)*X/CLASS''		
6335	037274	030505	020067	040502					
6336	037302	024104	043101	051111					
6337	037310	033460	030050	025051					
6338	037316	027530	046103	051501					
6339	037324	000123							
6340	037326	040522	043103	042440	EM12:	.ASCIZ	'RACF E5 BAD(AFI06(0)*X/CLASS)''		
6341	037334	020065	040502	024104					
6342	037342	043101	051111	033060					
6343	037350	030050	025051	027530					
6344	037356	046103	051501	024523					
6345	037364	000							
6346	037365	120	041103	042040	EM13:	.ASCIZ	/PCB DIDN'T LOAD FROM R5/		
6347	037372	042111	023516	020124					
6348	037400	047514	042101	043040					
6349	037406	047522	020115	032522					
6350	037414	000							
6351	037415	123	020120	044504	EM14:	.ASCIZ	/SP DIDN'T LOAD PROPERLY/		
6352	037422	047104	052047	046040					
6353	037430	040517	020104	051120					
6354	037436	050117	051105	054514					
6355	037444	000							
6356	037445	105	051122	051117	DH14:	.ASCII	/ERRORPC SP TST NUM/<CRLF>		
6357	037452	041520	020040	020040					
6358	037460	020040	051440	004520					
6359	037466	052040	052123	047040					
6360	037474	046525	200						
6361	037477	040	020040	020040		.ASCIZ	/ EXPECT ACTUAL/		
6362	037504	020040	042440	050130					
6363	037512	041505	020124	040440					
6364	037520	052103	040525	000114					
6365						.EVEN			
6366	037526	001116	001156	001154	DT14:	.WORD	\$ERRPC,\$REG1,\$RET0,\$\$TSTNM,0		
6367	037534	001210	000000						
6368	037540	032522	042040	042111	EM15:	.ASCIZ	/R5 DIDN'T LOAD PROPERLY/		
6369	037546	023516	020124	047514					
6370	037554	042101	050040	047522					
6371	037562	042520	046122	000131					
6372	037570	051105	047522	050122	DH15:	.ASCII	/ERRORPC R5 TST NUM/<CRLF>		
6373	037576	020103	020040	020040					
6374	037604	020040	032522	020040					
6375	037612	020040	020040	020040					
6376	037620	051524	020124	052516					
6377	037626	100115							
6378	037630	020011	054105	042520		.ASCIZ	/ EXPECT ACTUAL/		
6379	037636	052103	020040	041501					
6380	037644	052524	046101	000					
6381	037651	122	041501	020106	EM16:	.ASCIZ	'RACF X/CLASS DIDN'T GO HIGH''		
6382	037656	027530	046103	051501					
6383	037664	020123	044504	047104					
6384	037672	052047	043440	020117					
6385	037700	044510	044107	000					
6386	037705	122	041501	020106	EM17:	.ASCIZ	/RACF E34 BAD OR NOT GETTING THRU RACL RADR05/		
6387	037712	031505	020064	040502					
6388	037720	020104	051117	047040					

6389	037726	052117	043440	052105	
6390	037734	044524	043516	052040	
6391	037742	051110	020125	040522	
6392	037750	046103	051040	042101	
6393	037756	030122	000065		
6394	037762	051107	045101	051440	EM20: .ASCIZ /GRAJ SC05 L DOESN'T GET THRU TO RACK BRCAB04 L AS A HIGH/
6395	037770	030103	020065	020114	
6396	037776	047504	051505	023516	
6397	040004	020124	042507	020124	
6398	040012	044124	052522	052040	
6399	040020	020117	040522	045503	
6400	040026	041040	041522	041101	
6401	040034	032060	046040	040440	
6402	040042	020123	020101	044510	
6403	040050	044107	000		
6404	040053	122	020061	044504	EM21: .ASCIZ /R1 DIDN'T SHIFT/
6405	040060	047104	052047	051440	
6406	040066	044510	052106	000	
6407	040073	122	020061	044123	EM22: .ASCII /R1 SHIFTED BUT CARRY DIDN'T SET/<CRLF>
6408	040100	043111	042524	020104	
6409	040106	052502	020124	040503	
6410	040114	051122	020131	044504	
6411	040122	047104	052047	051440	
6412	040130	052105	200		
6413	040133	123	044510	052106	.ASCIZ /SHIFT COUNTER COULD BE STUCK/
6414	040140	041440	052517	052116	
6415	040146	051105	041440	052517	
6416	040154	042114	041040	020105	
6417	040162	052123	041525	000113	
6418	040170	051107	045101	051440	EM23: .ASCIZ /GRAJ SC05 L DOESN'T GET THRU TO RACK BRCAB04 L AS A LOW/
6419	040176	030103	020065	020114	
6420	040204	047504	051505	023516	
6421	040212	020124	042507	020124	
6422	040220	044124	052522	052040	
6423	040226	020117	040522	045503	
6424	040234	041040	041522	041101	
6425	040242	032060	046040	040440	
6426	040250	020123	020101	047514	
6427	040256	000127			
6428	040260	051501	020110	044522	EM24: .ASCIZ /ASH RIGHT DIDN'T SIGN FILL/
6429	040266	044107	020124	044504	
6430	040274	047104	052047	051440	
6431	040302	043511	020116	044506	
6432	040310	046114	000		
6433	040313	122	020061	044504	EM25: .ASCIZ /R1 DIDN'T SHIFT CORRECTLY/
6434	040320	047104	052047	051440	
6435	040326	044510	052106	04 440	
6436	040334	051117	042522	052103	
6437	040342	054514	000		
6438	040345	105	051122	051117	DH25: .ASCII /ERRORPC R1 TST NUM/<CRLF>
6439	040352	041520	020040	020040	
6440	040360	020040	051040	004461	
6441	040366	052040	052123	047040	
6442	040374	046525	200		
6443	040377	011	042440	050130	.ASCIZ / EXPECT ACTUAL/
6444	040404	041505	020124	040440	

6445	040412	052103	040525	000114		
6446						
6447	040420	001116	001164	001156	DT25:	.EVEN .WORD \$ERRPC,\$STMP1,\$REG1,\$STSTNM,0
6448	040426	001210	000000			
6449	040432	030522	051440	044510	EM26:	.ASCIZ /R1 SHIFTED BUT CARRY DIDN'T SET/
6450	040440	052106	042105	041040		
6451	040446	052125	041440	051101		
6452	040454	054522	042040	042111		
6453	040462	023516	020124	042523		
6454	040470	000124				
6455	040472	051501	027110	030062	EM27:	.ASCIZ /ASH.20 DIDN'T LOAD CC'S CORRECTLY/
6456	040500	042040	042111	023516		
6457	040506	020124	047514	042101		
6458	040514	041440	023503	020123		
6459	040522	047503	051122	041505		
6460	040530	046124	000131			
6461	040534	030522	051440	044510	EM30:	.ASCIZ /R1 SHIFTED WHEN SHIFT COUNT=0/
6462	040542	052106	042105	053440		
6463	040550	042510	020116	044123		
6464	040556	043111	020124	047503		
6465	040564	047125	036524	000060		
6466	040572	051501	027110	030064	EM31:	.ASCIZ /ASH.40 DIDN'T LOAD CC'S CORRECTLY/
6467	040600	042040	042111	023516		
6468	040606	020124	047514	042101		
6469	040614	041440	023503	020123		
6470	040622	047503	051122	041505		
6471	040630	046124	000131			
6472	040634	040522	043103	052440	EM32:	.ASCIZ 'RACF U/CLASS DOESN'T GO HIGH ON ASH'
6473	040642	041457	040514	051523		
6474	040650	042040	042517	047123		
6475	040656	052047	043440	020117		
6476	040664	044510	044107	047440		
6477	040672	020116	051501	000110		
6478	040700	051501	027110	030060	EM33:	.ASCIZ /ASH.00 FAILED/
6479	040706	043040	044501	042514		
6480	040714	000104				
6481	040716	051111	041103	042440	EM34:	.ASCIZ /IRCB E35(B2) NOT GOING LOW/
6482	040724	032463	041050	024462		
6483	040732	047040	052117	043440		
6484	040740	044517	043516	046040		
6485	040746	053517	000			
6486	040751	111	041522	020102	EM35:	.ASCII! /IRCB (MUL:ASHC)+MFP L DIDN'T GO LOW OR IRCB E37 BAD/<CRLF>
6487	040756	046450	046125	040472		
6488	040764	044123	024503	046453		
6489	040772	050106	046040	042040		
6490	041000	042111	023516	020124		
6491	041006	047507	046040	053517		
6492	041014	047440	020122	051111		
6493	041022	041103	042440	033463		
6494	041030	041040	042101	200		
6495	041035	117	020122	051111		.ASCIZ /OR IRCB E35(B1) STUCK LOW/
6496	041042	041103	042440	032463		
6497	041050	041050	024461	051440		
6498	041056	052524	045503	046040		
6499	041064	053517	000			
6500	041067	122	041501	020105	EM36:	.ASCIZ /RACE (MUL:ASHC+MFP) DIDN'T GO HIGH OR RACE E44 BAD/

```
6501 041074 046450 046125 040472
6502 041102 044123 025503 043115
6503 041110 024520 042040 042111
6504 041116 023516 020124 047507
6505 041124 044040 043511 020110
6506 041132 051117 051040 041501
6507 041140 020105 032105 020064
6508 041146 040502 000104
6509 041152 040522 042503 042440 EM37: .ASCIZ /RACE E45 BAD (AFIRO4(1)*(MUL:ASHC+MFP))/
6510 041160 032464 041040 042101
6511 041166 024040 043101 051111
6512 041174 032060 030450 025051
6513 041202 046450 046125 040472
6514 041210 044123 025503 043115
6515 041216 024520 000051
6516 041222 040522 042503 042440 EM40: .ASCIZ /RACE E33 BAD (AFIRO5(1)*(MUL:ASHC+MFP))/
6517 041230 031463 041040 042101
6518 041236 024040 043101 051111
6519 041244 032460 030450 025051
6520 041252 046450 046125 040472
6521 041260 044123 025503 043115
6522 041266 024520 000051
6523 041272 030122 042040 042111 EM41: .ASCIZ /RO DIDN'T SIGN FILL ON RIGHT SHIFT/
6524 041300 023516 020124 044523
6525 041306 047107 043040 046111
6526 041314 020114 047117 051040
6527 041322 043511 052110 051440
6528 041330 044510 052106 000
6529 041335 105 051122 051117 DH41: .ASCII /ERRORPC RO TST NUM/<CRLF>
6530 041342 041520 020040 020040
6531 041350 020040 020040 030122
6532 041356 020011 052040 052123
6533 041364 047040 046525 200
6534 041371 011 020040 054105 .ASCIZ / EXPECT ACTUAL/
6535 041376 042520 052103 020040
6536 041404 041501 052524 046101
6537 041412 000
6538 041414
6539 041414 001116 001162 001154 DT41: .EVEN
        .WORD $ERRPC,$TMP0,$REG0,$$TSTNM,0
6540 041422 001210 000000
6541 041426 040502 020104 041503 EM42: .ASCIZ /BAD CC'S ON RIGHT SHIFT/
6542 041434 051447 047440 020116
6543 041442 044522 044107 020124
6544 041450 044123 043111 000124
6545 041456 051105 047522 050122 DH42: .ASCII /ERRORPC PSW TST NUM/<CRLF>
6546 041464 020103 020040 020040
6547 041472 020040 051520 004527
6548 041500 020040 051524 020124
6549 041506 052516 100115
6550 041512 020011 054105 042520 .ASCIZ / EXPECT ACTUAL/
6551 041520 052103 020040 041501
6552 041526 052524 046101 000
6553 041534
6554 041534 001116 001162 001206 DT42: .EVEN
        .WORD $ERRPC,$TMP0,$ERPSW,$$TSTNM,0
6555 041542 001210 000000
6556 041546 030122 030074 020076 EM43: .ASCIZ /RO<0> DIDN'T GO TO R1<15>/
```

6557	041554	044504	047104	052047					
6558	041562	043440	020117	047524					
6559	041570	051040	036061	032461					
6560	041576	000076							
6561	041600	051105	047522	050122	DH43:	.ASCII	/ERRORPC	RO	R1 TST NUM/<CRLF>
6562	041606	020103	020040	020040					
6563	041614	020040	030122	020011					
6564	041622	020040	020040	020040					
6565	041630	051040	004461	020040					
6566	041636	051524	020124	052516					
6567	041644	100115							
6568	041646	020011	054105	042520		.ASCIZ	/	EXPECT ACTUAL	EXPECT ACTUAL/
6569	041654	052103	020040	041501					
6570	041662	052524	046101	020040					
6571	041670	042440	050130	041505					
6572	041676	020124	040440	052103					
6573	041704	040525	000114						
6574									
6575	041710	001116	001162	001154	DT43:	.EVEN			
6576	041716	001164	001156	001210		.WORD	\$ERRPC,\$TMP0,\$REG0,\$TMP1,\$REG1,\$\$TSTNM,0		
6577	041724	000000							
6578	041726	030122	042040	042111	EM44:	.ASCIZ	/RO DIDN'T GET SHIFTED LEFT PROPERLY/		
6579	041734	023516	020124	042507					
6580	041742	020124	044123	043111					
6581	041750	042524	020104	042514					
6582	041756	052106	050040	047522					
6583	041764	042520	046122	000131					
6584	041772	040502	020104	041503	EM45:	.ASCIZ	/BAD CC'S ON LEFT SHIFT/		
6585	042000	051447	047440	020116					
6586	042006	042514	052106	051440					
6587	042014	044510	052106	000					
6588	042021	115	050106	020124	EM46:	.ASCIZ	/MFPT LOADED RO INCORRECTLY/		
6589	042026	047514	042101	042105					
6590	042034	051040	020060	047111					
6591	042042	047503	051122	041505					
6592	042050	046124	000131						
6593	042054	051105	047522	050122	DH46:	.ASCII	/ERRORPC	RO	TST NUM/<CRLF>
6594	042062	020103	020040	020040					
6595	042070	020040	030122	020011					
6596	042076	051524	020124	052516					
6597	042104	100115							
6598	042106	020040	020040	020040		.ASCIZ	/	EXPECT ACTUAL/	
6599	042114	020040	054105	042520					
6600	042122	052103	020040	041501					
6601	042130	052524	046101	000					
6602		042136							
6603	042136	001116	001164	001162	DT46:	.EVEN			
6604	042144	001210	000000			.WORD	\$ERRPC,\$TMP1,\$TMP0,\$\$TSTNM,0		
6605	042150	040502	020104	041503	EM47:	.ASCIZ	/BAD CC'S ON NO SHIFT/		
6606	042156	051447	047440	020116					
6607	042164	047516	051440	044510					
6608	042172	052106	000						
6609	042175	122	020061	044504	EM50:	.ASCIZ	/R' DIDN'T ROTATE PROPERLY/		
6610	042202	047104	052047	051040					
6611	042210	052117	052101	020105					
6612	042216	051120	050117	051105					

6613	042224	054514	000		
6614	042227	102	052111	051440	EM51: .ASCIZ /BIT STUCK IN SC WITH 52 PATTERN/
6615	042234	052524	045503	044440	
6616	042242	020116	041523	053440	
6617	042250	052111	020110	031065	
6618	042256	050040	052101	042524	
6619	042264	047122	000		
6620	042267	105	051122	051117	DH51: .ASCII /ERRORPC R0 R1 C BIT TST NUM/<CRLF>
6621	042274	041520	020040	020040	
6622	042302	020040	051040	004460	
6623	042310	020040	020040	020040	
6624	042316	051040	004461	020040	
6625	042324	020040	020103	044502	
6626	042332	004524	052040	052123	
6627	042340	047040	046525	200	
6628	042345	011	042440	050130	.ASCIZ / EXPECT ACTUAL EXPECT ACTUAL EXPECT ACTUAL/
6629	042352	041505	020124	040440	
6630	042360	052103	040525	020114	
6631	042366	042440	050130	041505	
6632	042374	020124	040440	052103	
6633	042402	040525	020114	042440	
6634	042410	050130	041505	020124	
6635	042416	040440	052103	040525	
6636	042424	000114			
6637					
6638	042426	001116	001162	001154	DT51: .EVEN
6639	042434	001164	001156	001166	.WORD \$ERRPC,\$TMP0,\$REG0,\$TMP1,\$REG1,\$TMP2,\$ERPSW,\$\$TSTNM,0
6640	042442	001206	001210	000000	
6641	042450	044502	020124	052123	EM52: .ASCIZ /BIT STUCK IN SHIFT COUNTER WITH 25 PATTERN/
6642	042456	041525	020113	047111	
6643	042464	051440	044510	052106	
6644	042472	041440	052517	052116	
6645	042500	051105	053440	052111	
6646	042506	020110	032462	050040	
6647	042514	052101	042524	047122	
6648	042522	000			
6649	042523	111	041522	020102	EM53: .ASCIZ /IRCB E35(B3) NOT GOING LOW/
6650	042530	031505	024065	031502	
6651	042536	020051	047516	020124	
6652	042544	047507	047111	020107	
6653	042552	047514	000127		
6654	042556	051501	027103	030060	EM54: .ASCIZ /ASC.00 FAILED/
6655	042564	043040	044501	042514	
6656	042572	000104			
6657	042574	024122	052515	035114	EM55: .ASCIZ /R(MUL:ASHC+MFP) FIELD IN INSTR DECODE ROM BAD/
6658	042602	051501	041510	046453	
6659	042610	050106	020051	044506	
6660	042616	046105	020104	047111	
6661	042624	044440	051516	051124	
6662	042632	042040	041505	042117	
6663	042640	020105	047522	020115	
6664	042646	040502	000104		
6665	042652	051107	042101	042040	EM56: .ASCIZ /GRAD DROO STJCK HIGH OR RACK E64(C1) BAD/
6666	042660	030122	020060	052123	
6667	042666	041525	020113	044510	
6668	042674	044107	047440	020122	

6669	042702	040522	045503	042440	
6670	042710	032066	041450	024461	
6671	042716	041040	042101	000	
6672	042723	107	040522	020104	EM57: .ASCIZ /GRAD DROO STUCK LOW OR RACK E64(C1) BAD/
6673	042730	051104	030060	051440	
6674	042736	052524	045503	046040	
6675	042744	053517	047440	020122	
6676	042752	040522	045503	042440	
6677	042760	032066	041450	024461	
6678	042766	041040	042101	000	
6679	042773	107	040522	020112	EM60: .ASCII /GRAJ SC=0 L NOT GETTING TO RACK E50(C1) OR/<CRLF>
6680	043000	041523	030075	046040	
6681	043006	047040	052117	043440	
6682	043014	052105	044524	043516	
6683	043022	052040	020117	040522	
6684	043030	045503	042440	030065	
6685	043036	041450	024461	047440	
6686	043044	100122			
6687	043046	042440	030065	041040	.ASCIZ / E50 BAD (FAILED LOW)/
6688	043054	042101	024040	040506	
6689	043062	046111	042105	046040	
6690	043070	053517	000051		
6691	043074	030122	047440	020122	EM61: .ASCIZ /R0 OR R1 OR BOTH BAD ON POSITIVE MULTIPLICAND/
6692	043102	030522	047440	020122	
6693	043110	047502	044124	041040	
6694	043116	042101	047440	020116	
6695	043124	047520	044523	044524	
6696	043132	042526	046440	046125	
6697	043140	044524	046120	041511	
6698	043146	047101	000104		
6699	043152	040502	020104	041503	EM62: .ASCIZ /BAD CC'S ON POSITIVE MULTIPLICAND/
6700	043160	051447	047440	020116	
6701	043166	047520	044523	044524	
6702	043174	042526	046440	046125	
6703	043202	044524	046120	041511	
6704	043210	047101	000104		
6705	043214	030122	041040	042101	EM63: .ASCIZ /R0 BAD ON NEGATIVE MULTIPLICAND/
6706	043222	047440	020116	042516	
6707	043230	040507	044524	042526	
6708	043236	046440	046125	044524	
6709	043244	046120	041511	047101	
6710	043252	000104			
6711	043254	030522	041040	042101	EM64: .ASCIZ /R1 BAD ON NEGATIVE MULTIPLICAND/
6712	043262	047440	020116	042516	
6713	043270	040507	044524	042526	
6714	043276	046440	046125	044524	
6715	043304	046120	041511	047101	
6716	043312	000104			
6717	043314	040502	020104	041503	EM65: .ASCIZ /BAD CC'S DUE TO STATE MUL.50/
6718	043322	051447	042040	042525	
6719	043330	052040	020117	052123	
6720	043336	052101	020105	052515	
6721	043344	027114	030065	000	
6722	043351	103	042040	042111	EM66: .ASCIZ /C DIDN'T SET ON OVERFLOW/
6723	043356	023516	020124	042523	
6724	043364	020124	047117	047440	

6725	043372	042526	043122	047514	
6726	043400	000127			
6727	043402	020103	044504	047104	EM67: .ASCIZ /C DIDN'T SET ON UNDERFLOW/
6728	043410	052047	051440	052105	
6729	043416	047440	020116	047125	
6730	043424	042504	043122	047514	
6731	043432	000127			
6732	043434	052515	027114	030060	EM70: .ASCIZ /MUL.00 FAILED/
6733	043442	043040	044501	042514	
6734	043450	000104			
6735	043452	051501	027110	030063	EM72: .ASCIZ /ASH.30 DIDN'T LOAD CC'S CORRECTLY/
6736	043460	042040	042111	023516	
6737	043466	020124	047514	042101	
6738	043474	041440	023503	020123	
6739	043502	047503	051122	041505	
6740	043510	046124	000131		
6741	043514	051501	027110	030464	EM73: .ASCIZ /ASH.41 FAILED/
6742	043522	043040	044501	042514	
6743	043530	000104			
6744	043532	051501	027110	030464	EM74: .ASCIZ /ASH.41 DIDN'T LOAD CC'S CORRECTLY/
6745	043540	042040	042111	023516	
6746	043546	020124	047514	042101	
6747	043554	041440	023503	020123	
6748	043562	047503	051122	041505	
6749	043570	046124	000131		
6750	043574	051111	043103	055040	EM75: .ASCIZ /IRCF Z2(1) DOESN'T GO HIGH OR RACK E49(B1) STUCK LOW/
6751	043602	024062	024461	042040	
6752	043610	042517	047123	052047	
6753	043616	043440	020117	044510	
6754	043624	044107	047440	020122	
6755	043632	040522	045503	042440	
6756	043640	034464	041050	024461	
6757	043646	051440	052524	045503	
6758	043654	046040	053517	000	
6759	043661	103	023503	020123	EM76: .ASCIZ /CC'S BAD IN DVE.00/
6760	043666	040502	020104	047111	
6761	043674	042040	042526	030056	
6762	043702	000060			
6763	043704	041503	051447	041040	EM77: .ASCIZ /CC'S BAD IN DVC.70/
6764	043712	042101	044440	020116	
6765	043720	053104	027103	030067	
6766	043726	000			
6767	043727	122	041501	020113	EM100: .ASCIZ /RACK F50(B0) STUCK HIGH/
6768	043734	032505	024060	030102	
6769	043742	020051	052123	041525	
6770	043750	020113	044510	044107	
6771	043756	000			

6772	043757	107	040522	020112
6773	043764	044504	020126	052523
6774	043772	020102	020114	047516
6775	044000	020124	047507	047111
6776	044006	020107	047514	020127
6777	044014	051117	047040	052117
6778	044022	043440	052105	044524
6779	044030	043516	052040	051110
6780	044036	100125	000	
6781	044041	124	020117	040522
6782	044046	045503	042440	034464
6783	044054	047440	020122	032105
6784	044062	024071	030104	020051
6785	044070	052123	041525	020113
6786	044076	044510	044107	000
6787	044103	121	047525	020124
6788	044110	045517	051040	046505
6789	044116	044501	042116	051105
6790	044124	041040	042101	024040
6791	044132	047522	020115	052123
6792	044140	052101	020105	040506
6793	044146	046111	051125	024505
6794	044154	000		
6795	044155	122	041501	020113
6796	044162	032105	024071	030501
6797	044170	020051	052123	041525
6798	044176	020113	047514	000127
6799	044204	051107	045101	042040
6800	044212	053111	051440	041125
6801	044220	046040	047040	052117
6802	044226	043440	044517	043516
6803	044234	044040	043511	020110
6804	044242	051117	051040	041501
6805	044250	020113	032105	024071
6806	044256	030104	020051	052123
6807	044264	041525	020113	047514
6808	044272	000127		
6809	044274	052521	047524	042511
6810	044302	052116	023040	051040
6811	044310	046505	044501	042116
6812	044316	051105	041040	042101
6813	044324	000		
6814	044325	107	040522	020110
6815	044332	051123	032461	047040
6816	044340	052117	043440	052105
6817	044346	044524	043516	052040
6818	044354	020117	040522	045503
6819	044362	042440	032066	047440
6820	044370	020122	033105	024064
6821	044376	030102	020051	052123
6822	044404	041525	020113	044510
6823	044412	044107	000	
6824	044415	111	041522	020110
6825	044422	024116	024461	047040
6826	044430	052117	043440	052105
6827	044436	044524	043516	052040

EM101: .ASCIZ /GRAJ DIV SUB L NOT GOING LOW OR NOT GETTING THRU/<CR LF>

.ASCIZ /TO RACK E49 OR E49(D0) STUCK HIGH/

EM102: .ASCIZ /QUOT OK REMAINDER BAD (ROM STATE FAILURE)/

EM103: .ASCIZ /RACK E49(A1) STUCK LOW/

EM104: .ASCIZ /GRAJ DIV SUB L NOT GOING HIGH OR RACK E49(D0) STUCK LOW/

EM105: .ASCIZ /QUOTIENT & REMAINDER BAD/

EM106: .ASCIZ /GRAH SR15 NOT GETTING TO RACK E64 OR E64(B0) STUCK HIGH/

EM110: .ASCIZ /IRCH N(1) NOT GETTING TO RACK E63 OR E63(D0) STUCK HIGH/

6828	044444	020117	040522	045503	
6829	044452	042440	031466	047440	
6830	044460	020122	033105	024063	
6831	044466	030104	020051	052123	
6832	044474	041525	020113	044510	
6833	044502	044107	000		
6834	044505	122	041501	020113	EM111: .ASCIZ /RACK E49(B1) STUCK HIGH/
6835	044512	032105	024071	030502	
6836	044520	020051	052123	041525	
6837	044526	020113	044510	044107	
6838	044534	000			
6839	044535	107	040522	020112	EM112: .ASCIZ /GRAJ DIV QUIT L NOT GOING HIGH OR NOT GETTING/<CRLF>
6840	044542	044504	020126	052521	
6841	044550	052111	046040	047040	
6842	044556	052117	043440	044517	
6843	044564	043516	044040	043511	
6844	044572	020110	051117	047040	
6845	044600	052117	043440	052105	
6846	044606	044524	043516	000200	
6847	044614	047524	051040	041501	.ASCIZ /TO RACK E63 OR E63(C0) STUCK LOW/
6848	044622	020113	033105	020063	
6849	044630	051117	042440	031466	
6850	044636	041450	024460	051440	
6851	044644	052524	045503	046040	
6852	044652	053517	000		
6853	044655	122	041501	020113	EM113: .ASCIZ /RACK E50(B0) STUCK LOW/
6854	044662	032505	024060	030102	
6855	044670	020051	052123	041525	
6856	044676	020113	047514	000127	
6857	044704	040522	045503	042440	EM114: .ASCIZ /RACK E64(B0) STUCK LOW/
6858	044712	032066	041050	024460	
6859	044720	051440	052524	045503	
6860	044726	046040	053517	000	
6861	044733	104	041526	031056	EM115: .ASCIZ /DVC.20,DVC.40,DVC.80,OR DVC.90 FAILED/
6862	044740	026060	053104	027103	
6863	044746	030064	042054	041526	
6864	044754	034056	026060	051117	
6865	044762	042040	041526	034456	
6866	044770	020060	040506	046111	
6867	044776	042105	000		
6868	045001	102	042101	041440	EM117: .ASCIZ /BAD CC'S IN DVC.90 OR RACK E63(D0) STUCK LOW/
6869	045006	023503	020123	047111	
6870	045014	042040	041526	034456	
6871	045022	020060	051117	051040	
6872	045030	041501	020113	033105	
6873	045036	024063	030104	020051	
6874	045044	052123	041525	020113	
6875	045052	047514	000127		
6876	045056	051107	045101	042040	EM120: .ASCIZ /GRAJ DIV QUIT DIDN'T GO LOW OR RACK E63(C0) STUCK HIGH/
6877	045064	053111	050440	044525	
6878	045072	020124	044504	047104	
6879	045100	052047	043440	020117	
6880	045106	047514	020127	051117	
6881	045114	051040	041501	020113	
6882	045122	033105	024063	030103	
6883	045130	020051	052123	041525	

6884	045136	020113	044510	044107	
6885	045144	000			
6886	045145	103	023503	020123	FM121: .ASCIZ /CC'S BAD DUE TO EITHER DIV.30 OR DVE.20/
6887	045152	040502	020104	052504	
6888	045160	020105	047524	042440	
6889	045166	052111	042510	020122	
6890	045174	044504	027126	030063	
6891	045202	047440	020122	053104	
6892	045210	027105	030062	000	
6893	045215	107	040522	020112	FM122: .ASCIZ /GRAJ E5 BAD(72(0)*LEFT SAVE(1))/
6894	045222	032505	041040	042101	
6895	045230	055050	024062	024460	
6896	045236	046052	043105	020124	
6897	045244	040523	042526	030450	
6898	045252	024451	000		
6899	045255	122	041501	020113	EM123: .ASCIZ /RACK E63(D0) STUCK LOW/
6900	045262	033105	024063	030104	
6901	045270	020051	052123	041525	
6902	045276	020113	047514	000127	
6903	045304	041503	051447	042040	EM124: .ASCIZ /CC'S DIDN'T LOAD PROPERLY/
6904	045312	042111	023516	020124	
6905	045320	047514	042101	050040	
6906	045326	047522	042520	046122	
6907	045334	000131			
6908	045336	051107	045101	042440	EM125: .ASCIZ /GRAJ E5(N(1)*SR15(1)) BAD OR ROM STATE BAD/
6909	045344	024065	024116	024461	
6910	045352	051452	030522	024065	
6911	045360	024461	020051	040502	
6912	045366	020104	051117	051040	
6913	045374	046517	051440	040524	
6914	045402	042524	041040	042101	
6915	045410	000			
6916	045411	121	047525	020124	EM127: .ASCIZ /QUOT BAD REMAINDER OK (ROM STATE FAILURE)/
6917	045416	040502	020104	042522	
6918	045424	040515	047111	042504	
6919	045432	020122	045517	024040	
6920	045440	047522	020115	052123	
6921	045446	052101	020105	040506	
6922	045454	046111	051125	024505	
6923	045462	000			
6924	045463	121	047525	044524	EM130: .ASCIZ /QUOTIENT BAD (ROM STATE FAILURE)/
6925	045470	047105	020124	040502	
6926	045476	020104	051050	046517	
6927	045504	051440	040524	042524	
6928	045512	043040	044501	052514	
6929	045520	042522	000051		
6930	045524	040502	020104	041503	EM134: .ASCIZ /BAD CC'S ON DIV OVERFLOW, STATE DVD.10/
6931	045532	051447	047440	020116	
6932	045540	044504	020126	053117	
6933	045546	051105	046106	053517	
6934	045554	020054	052123	052101	
6935	045562	020105	053104	027104	
6936	045570	030061	000		
6937	045573	122	020060	040502	EM135: .ASCIZ /RO BAD/
6938	045600	000104			
6939	045602	052123	052101	020105	EM136: .ASCIZ /STATE MTP.00 DIDN'T INC SP/

6940	045610	052115	027120	030060	
6941	045616	042040	042111	023516	
6942	045624	020124	047111	020103	
6943	045632	050123	000		
6944	045635	122	041501	020106	EM140: .ASCIZ 'RACF X/CLASS DOESN'T GO HIGH'
6945	045642	027530	046103	051501	
6946	045650	020123	047504	051505	
6947	045656	023516	020124	047507	
6948	045664	044040	043511	000110	
6949	045672	052115	027120	030061	EM141: .ASCIZ /MTP.10 FAILED TO RELOAD THE DR/
6950	045700	043040	044501	042514	
6951	045706	020104	047524	051040	
6952	045714	046105	040517	020104	
6953	045722	044124	020105	051104	
6954	045730	000			
6955	045731	123	020120	047514	EM142: .ASCIZ /SP LOADED INCORRECTLY/
6956	045736	042101	042105	044440	
6957	045744	041516	051117	042522	
6958	045752	052103	054514	000	
6959	045757	115	050124	030456	EM143: .ASCIZ /MTP.10 DIDN'T PUT FCB IN DR/
6960	045764	020060	044504	047104	
6961	045772	052047	050040	052125	
6962	046000	050040	041103	044440	
6963	046006	020116	051104	000	
6964	046013	122	041501	020106	EM144: .ASCIZ /RACF E20(4) STUCK HIGH/
6965	046020	031105	024060	024464	
6966	046026	051440	052524	045503	
6967	046034	044040	043511	000110	
6968	046042	043115	027120	030061	EM145: .ASCIZ /MFP.10 DIDN'T DECREMENT THE SP/
6969	046050	042040	042111	023516	
6970	046056	020124	042504	051103	
6971	046064	046505	047105	020124	
6972	046072	044124	020105	050123	
6973	046100	000			
6974	046101	122	020060	044504	EM146: .ASCIZ /RO DIDN'T GET PUT ON THE STACK/
6975	046106	047104	052047	043440	
6976	046114	052105	050040	052125	
6977	046122	047440	020116	044124	
6978	046130	020105	052123	041501	
6979	046136	000113			
6980	046140	040502	020104	041503	EM147: .ASCIZ /BAD CC'S, CC CONTROL ROM/
6981	046146	051447	020054	041503	
6982	046154	041440	047117	051124	
6983	046162	046117	051040	046517	
6984	046170	000			
6985	046171	115	050106	030056	EM151: .ASCIZ /MFP.00 BAD/
6986	046176	020060	040502	000104	
6987	046204	040502	020104	041503	EM152: .ASCIZ /BAD CC'S DUE TO MFP.00/
6988	046212	051447	042040	042525	
6989	046220	052040	020117	043115	
6990	046226	027120	030060	000	
6991	046233	124	050122	030056	EM155: .ASCIZ /TRP.01 FAILED TO LOAD BR/
6992	046240	020061	040506	046111	
6993	046246	042105	052040	020117	
6994	046254	047514	042101	041040	
6995	046262	000122			

6996	046264	051111	042103	044440	EM156:	.ASCII	/IRCD IOT DOESN'T GO LOW OR DAPE TV04 DOES/<CRLF>
6997	046272	052117	042040	042517			
6998	046300	047123	052047	043440			
6999	046306	020117	047514	020127			
7000	046314	051117	042040	050101			
7001	046322	020105	053124	032060			
7002	046330	042040	042517	100123			
7003	046336	047516	020124	047507		.ASCIZ	/NOT GO HIGH OR DOESN'T GET TO THE ALU/
7004	046344	044040	043511	020110			
7005	046352	051117	042040	042517			
7006	046360	047123	052047	043440			
7007	046366	052105	052040	020117			
7008	046374	044124	020105	046101			
7009	046402	000125					
7010	046404	051124	027120	030460	EM157:	.ASCIZ	/TRP.01 FAILED TO LOAD DR/
7011	046412	043040	044501	042514			
7012	046420	020104	047524	046040			
7013	046426	040517	020104	051104			
7014	046434	000					
7015	046435	124	050122	030056	EM160:	.ASCIZ	/TRP.00 FAILED TO LOAD BR/
7016	046442	020060	040506	046111			
7017	046450	042105	052040	020117			
7018	046456	047514	042101	041040			
7019	046464	000122					
7020	046466	051111	042103	047440	EM161:	.ASCII	/IRCD OPCODE3 DOESN'T GO LOW OR DOESN'T/<CRLF>
7021	046474	041520	042117	031505			
7022	046502	042040	042517	047123			
7023	046510	052047	043440	020117			
7024	046516	047514	020127	051117			
7025	046524	042040	042517	047123			
7026	046532	052047	200				
7027	046535	107	052105	052040		.ASCIZ	/GET THRU TO DAPE TV03/
7028	046542	051110	020125	047524			
7029	046550	042040	050101	020105			
7030	046556	053124	031460	000			
7031	046563	124	050122	030056	EM162:	.ASCIZ	/TRP.00 FAILED TO LOAD DR/
7032	046570	020060	040506	046111			
7033	046576	042105	052040	020117			
7034	046604	047514	042101	042040			
7035	046612	000122					
7036	046614	044502	020124	040506	EM163:	.ASCIZ	/BIT FAILED IN PIRQ REG/
7037	046622	046111	042105	044440			
7038	046630	020116	044520	050522			
7039	046636	051040	043505	000			
7040	046643	105	051122	051117	DH163:	.ASCII	/ERRORPC PIRQ TST NUM/<CRLF>
7041	046650	041520	020040	020040			
7042	046656	020040	050040	051111			
7043	046664	004521	020040	051524			
7044	046672	020124	052516	100115			
7045	046700	020011	054105	042520		.ASCIZ	/ EXPECT ACTUAL/
7046	046706	052103	020040	041501			
7047	046714	052524	046101	000			
7048		046722				.EVEN	
7049	046722	001116	001162	001216	DT163:	.WORD	\$ERRPC,\$TMP0,\$EPIRQ,\$\$TSTNM,0
7050	046730	001210	000000				
7051	046734	042506	027124	030060	EM164:	.ASCIZ	/FET.00 HAD BAD BEN FIELD/

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

7108	047436	046040	053517	047440	
7109	047444	100122			
7110	047446	047516	020124	042507	.ASCII /NOT GETTING TO RACK E64 OR RACK E64(A1) BAD/<CRLF>
7111	047454	052124	047111	020107	
7112	047462	047524	051040	041501	
7113	047470	020113	033105	020064	
7114	047476	051117	051040	041501	
7115	047504	020113	033105	024064	
7116	047512	030501	020051	040502	
7117	047520	100104			
7118	047522	051117	052040	041515	.ASCII /OR TMCB PIRQ NOT GETTING TO DAPE OR DAPE TV05*07/<CRLF>
7119	047530	020102	044520	050522	
7120	047536	047040	052117	043440	
7121	047544	052105	044524	043516	
7122	047552	052040	020117	040504	
7123	047560	042520	047440	020122	
7124	047566	040504	042520	052040	
7125	047574	030126	025065	033460	
7126	047602	200			
7127	047603	116	052117	043440	.ASCIIZ /NOT GOING HIGH OR NOT GETTING TO THE ALU/
7128	047610	044517	043516	044040	
7129	047616	043511	020110	051117	
7130	047624	047040	052117	043440	
7131	047632	052105	044524	043516	
7132	047640	052040	020117	044124	
7133	047646	020105	046101	000125	
7134	047654	046524	041103	042440	EM174: .ASCIIZ /TMCB E62(2) BAD OR TMCB HONOR PIR 2 NOT GOING LOW/
7135	047662	031066	031050	020051	
7136	047670	040502	020104	051117	
7137	047676	052040	041515	020102	
7138	047704	047510	047516	020122	
7139	047712	044520	020122	020062	
7140	047720	047516	020124	047507	
7141	047726	047111	020107	047514	
7142	047734	000127			
7143	047736	046524	041103	042440	EM175: .ASCIIZ /TMCB E63 BAD/
7144	047744	031466	041040	042101	
7145	047752	000			
7146	047753	114	053105	046105	EM176: .ASCIIZ /LEVEL 2 INTERRUPT WHEN LEVEL 1 ON/
7147	047760	031040	044440	052116	
7148	047766	051105	052522	052120	
7149	047774	053440	042510	020116	
7150	050002	042514	042526	020114	
7151	050010	020061	047117	000	
7152	050015	124	041515	020102	EM177: .ASCIIZ /TMCB E62(3) BAD OR TMCB HONOR PIR 3 NOT GOING LOW/
7153	050022	033105	024062	024463	
7154	050030	041040	042101	047440	
7155	050036	020122	046524	041103	
7156	050044	044040	047117	051117	
7157	050052	050040	051111	031440	
7158	050060	047040	052117	043440	
7159	050066	044517	043516	046040	
7160	050074	053517	000		
7161	050077	114	053105	046105	EM201: .ASCIIZ /LEVEL 2 INTERRUPT WHEN CPU LEVEL 2 ON/
7162	050104	031040	044440	052116	
7163	050112	051105	052522	052120	

7164	050120	053440	042510	020116	
7165	050126	050103	020125	042514	
7166	050134	042526	020114	020062	
7167	050142	047117	000		
7168	050145	105	051122	051117	DH201: .ASCIZ /ERRORPC PIRQ TST NUM/
7169	050152	041520	020040	044520	
7170	050160	050522	020040	020040	
7171	050166	051524	020124	052516	
7172	050174	000115			
7173					
7174	050176	001116	001216	001210	DT201: .EVEN
7175	050204	000000			.WORD \$ERRPC,\$EPIRQ,\$\$TSTNM,0
7176	050206	046524	041103	042440	EM202: .ASCIZ /TMCB E62(5) BAD OR TMCA HONOR PIR 4 NOT GOING LOW/
7177	050214	031066	032450	020051	
7178	050222	040502	020104	051117	
7179	050230	052040	041515	020101	
7180	050236	047510	047516	020122	
7181	050244	044520	020122	020064	
7182	050252	047516	020124	047507	
7183	050260	047111	020107	047514	
7184	050266	000127			
7185	050270	042514	042526	020114	EM204: .ASCIZ /LEVEL 3 INTERRUPT WHEN CPU LEVEL 3 ON/
7186	050276	020063	047111	042524	
7187	050304	051122	050125	020124	
7188	050312	044127	047105	041440	
7189	050320	052520	046040	053105	
7190	050326	046105	031440	047440	
7191	050334	000116			
7192	050336	046524	041103	042440	EM205: .ASCIZ /TMCB E62(11) BAD OR TMCA HONOR PIR 5 NOT GOING LOW/
7193	050344	031066	030450	024461	
7194	050352	041040	042101	047440	
7195	050360	020122	046524	040503	
7196	050366	044040	047117	051117	
7197	050374	050040	051111	032440	
7198	050402	047040	052117	043440	
7199	050410	044517	043516	046040	
7200	050416	053517	000		
7201	050421	114	053105	046105	EM207: .ASCIZ /LEVEL 4 INTERRUPT WHEN CPU LEVEL 4 ON/
7202	050426	032040	044440	052116	
7203	050434	051105	052522	052120	
7204	050442	053440	042510	020116	
7205	050450	050103	020125	042514	
7206	050456	042526	020114	020064	
7207	050464	047117	000		
7208	050467	124	041515	020102	EM210: .ASCIZ /TMCB E51(11) BAD OR TMCB E55(8-9) BAD/
7209	050474	032505	024061	030461	
7210	050502	020051	040502	020104	
7211	050510	051117	052040	041515	
7212	050516	020102	032505	024065	
7213	050524	026470	024471	041040	
7214	050532	042101	000		
7215	050535	124	041515	020102	EM211: .ASCIZ /TMCB E70(1) BAD OR TMCA HONOR PIR6 NOT GOING LOW/
7216	050542	033505	024060	024461	
7217	050550	041040	042101	047440	
7218	050556	020122	046524	040503	
7219	050564	044040	047117	051117	

7220	050572	050040	051111	020066	
7221	050600	047516	020124	047507	
7222	050606	047111	020107	047514	
7223	050614	000127			
7224	050616	046524	041103	042440	EM212: .ASCIZ /TMCB E63(12) BAD OR TMCB E61(1) BAD/
7225	050624	031466	030450	024462	
7226	050632	041040	042101	047440	
7227	050640	020122	046524	041103	
7228	050646	042440	030466	030450	
7229	050654	020051	040502	000104	
7230	050662	042514	042526	020114	EM213: .ASCII7 /LEVEL 5 INTERRUPT WHEN CPU LEVEL 5 ON/
7231	050670	020065	047111	042524	
7232	050676	051122	050125	020124	
7233	050704	044127	047105	041440	
7234	050712	052520	046040	053105	
7235	050720	046105	032440	047440	
7236	050726	000116			
7237	050730	046524	041103	042440	EM214: .ASCIZ /TMCB E70(6) BAD OR TMCA HONOR PIR 7 NOT GOING LOW/
7238	050736	030067	033050	020051	
7239	050744	040502	020104	051117	
7240	050752	052040	041515	020101	
7241	050760	047510	047516	020122	
7242	050766	044520	020122	020067	
7243	050774	047516	020124	047507	
7244	051002	047111	020107	047514	
7245	051010	000127			
7246	051012	042514	042526	020114	EM216: .ASCIZ /LEVEL 6 INTERRUPT WHEN CPU LEVEL 6 ON/
7247	051020	020066	047111	042524	
7248	051026	051122	050125	020124	
7249	051034	044127	047105	041440	
7250	051042	052520	046040	053105	
7251	051050	046105	033040	047440	
7252	051056	000116			
7253	051060	044524	042515	052517	EM217: .ASCIZ /TIMEOUT ON DATI DIDN'T WORK/
7254	051066	020124	047117	042040	
7255	051074	052101	020111	044504	
7256	051102	047104	052047	053440	
7257	051110	051117	000113		
7258	051114	046524	041503	040440	EM220: .ASCII /TMCC AERF(1) L NOT GOING LOW/<CRLF>
7259	051122	051105	024106	024461	
7260	051130	046040	047040	052117	
7261	051136	043440	044517	043516	
7262	051144	046040	053517	200	
7263	051151	117	020122	046524	.ASCIZ /OR TMCB E53(11) BAD/
7264	051156	041103	042440	031465	
7265	051164	030450	024461	041040	
7266	051172	042101	000		
7267	051175	124	046511	047505	EM221: .ASCIZ /TIMEOUT ON DATO DIDN'T WORK/
7268	051202	052125	047440	020116	
7269	051210	040504	047524	042040	
7270	051216	042111	023516	020124	
7271	051224	047527	045522	000	
7272	051231	124	041515	020102	EM222: .ASCIZ /TMCB PS07(0) NOT GETTING TO TMCB E77 OR E77 BAD/
7273	051236	051520	033460	030050	
7274	051244	020051	047516	020124	
7275	051252	042507	052124	047111	

7276	051260	020107	047524	052040
7277	051266	041515	020102	033505
7278	051274	020067	051117	042440
7279	051302	033467	041040	042101
7280	051310	000		
7281	051311	102	032122	023040
7282	051316	041040	033122	043040
7283	051324	044501	042514	000104
7284	051332	051102	020064	040506
7285	051340	046111	042105	020056
7286	051346	044505	044124	051105
7287	051354	052040	041515	020102
7288	051362	047510	047516	020122
7289	051370	051102	020064	047516
7290	051376	020124	047507	047111
7291	051404	020107	047514	020127
7292	051412	051117	200	
7293	051415	124	041515	020102
7294	051422	033105	024062	024464
7295	051430	041040	042101	047440
7296	051436	020122	047111	042524
7297	051444	051122	050125	020124
7298	051452	051117	041040	020107
7299	051460	047514	044507	020103
7300	051466	047117	052440	041502
7301	051474	041040	042101	000
7302	051501	102	032122	043040
7303	051506	044501	042514	020104
7304	051514	051102	020066	045517
7305	051522	042440	052111	042510
7306	051530	020122	046524	041103
7307	051536	044040	047117	051117
7308	051544	041040	032122	047040
7309	051552	052117	043440	044517
7310	051560	043516	200	
7311	051563	114	053517	047440
7312	051570	020122	046524	041103
7313	051576	042440	031066	032050
7314	051604	020051	040502	000104
7315	051612	046524	040503	044040
7316	051620	047117	051117	041040
7317	051626	032522	047040	052117
7318	051634	043440	044517	043516
7319	051642	046040	053517	047440
7320	051650	020122	046524	041103
7321	051656	042440	031066	033050
7322	051664	020051	040502	000104
7323	051672	046524	040503	044040
7324	051700	047117	051117	041040
7325	051706	033122	047040	052117
7326	051714	043440	044517	043516
7327	051722	046040	053517	047440
7328	051730	020122	046524	041103
7329	051736	042440	031066	030450
7330	051744	024462	041040	042101
7331	051752	000		

EM223: .ASCIZ /BR4 & BR6 FAILED/

EM224: .ASCII /BR4 FAILED. EITHER TMCB HONOR BR4 NOT GOING LOW OR/<CRLF>

.ASCIZ /TMCB E62(4) BAD OR INTERRUPT OR BG LOGIC ON UBC BAD/

EM225: .ASCII /BR4 FAILED BR6 OK EITHER TMCB HONOR BR4 NOT GOING/<CRLF>

.ASCIZ /LOW OR TMCB E62(4) BAD/

EM226: .ASCIZ /TMCA HONOR BR5 NOT GOING LOW OR TMCB E62(6) BAD/

EM227: .ASCIZ /TMCA HONOR BR6 NOT GOING LOW OR TMCB E62(12) BAD/

7332	051753	131	046105	055040
7333	051760	047117	020105	040506
7334	051766	046111	042105	042440
7335	051774	052111	042510	020122
7336	052002	046524	042103	051440
7337	052010	020114	042531	020114
7338	052016	047516	020124	047507
7339	052024	047111	020107	044510
7340	052032	044107	047440	100122
7341	052040	051117	052040	041515
7342	052046	020101	047510	047516
7343	052054	020122	046123	020131
7344	052062	047516	020124	047507
7345	052070	047111	020107	047514
7346	052076	020127	051117	052040
7347	052104	041515	020102	033505
7348	052112	024060	024463	041040
7349	052120	042101	200	
7350	052123	117	020122	042502
7351	052130	030516	020063	040506
7352	052136	046111	042105	020055
7353	052144	044505	044124	051105
7354	052152	052040	041515	020101
7355	052160	047510	047516	020122
7356	052166	046123	020131	047516
7357	052174	020124	042507	052124
7358	052202	047111	100107	
7359	052206	047524	052040	041515
7360	052214	020102	032505	020063
7361	052222	051117	042440	031465
7362	052230	031450	020051	040502
7363	052236	000104		
7364	052240	046524	041503	042440
7365	052246	033061	034050	020051
7366	052254	047516	020124	047507
7367	052262	047111	020107	047514
7368	052270	020127	051117	052040
7369	052276	041515	020103	031505
7370	052304	020066	040502	000104
7371	052312	042120	041522	051440
7372	052320	040524	045503	046040
7373	052326	046511	052111	047040
7374	052334	052117	043440	052105
7375	052342	044524	043516	052040
7376	052350	020117	046524	042103
7377	052356	040440	020123	020101
7378	052364	047514	020127	051117
7379	052372	200		
7380	052373	124	041515	020104
7381	052400	034105	041040	042101
7382	052406	000		
7383	052407	125	041502	020103
7384	052414	040504	044524	047040
7385	052422	052117	043440	052105
7386	052430	044524	043516	052040
7387	052436	020117	046524	041503

EM230: .ASCII /YEL ZONE FAILED EITHER TMCD SL YEL NOT GOING HIGH OR/<CRLF>

.ASCII /OR TMCA HONOR SLY NOT GOING LOW OR TMCB E70(3) BAD/<CRLF>

.ASCII /OR BEN13 FAILED- EITHER TMCA HONOR SLY NOT GETTING/<CRLF>

.ASCIZ /TO TMCB E53 OR E53(3) BAD/

EM232: .ASCIZ /TMCC E16(8) NOT GOING LOW OR TMCC E36 BAD/

EM233: .ASCII /PDR(STACK LIMIT NOT GETTING TO TMCD AS A LOW OR/<CRLF>

.ASCIZ /TMCD E8 BAD/

EM234: .ASCII /UBCC DATI NOT GETTING TO TMCC AS A LOW OR EITHER/<CRLF>

7388	052444	040440	020123	020101	
7389	052452	047514	020127	051117	
7390	052460	042440	052111	042510	
7391	052466	100122			
7392	052470	046524	041503	042440	.ASCIZ /TMCC E30 OR E36 BAD/
7393	052476	030063	047440	020122	
7394	052504	031505	020066	040502	
7395	052512	000104			
7396	052514	046524	042103	051440	EM235: .ASCII /TMCD SL RED NOT GOING LOW OR TMCC ABORT/<CRLF>
7397	052522	020114	042522	020104	
7398	052530	047516	020124	047507	
7399	052536	047111	020107	047514	
7400	052544	020127	051117	052040	
7401	052552	041515	020103	041101	
7402	052560	051117	100124		
7403	052564	047516	020124	047507	.ASCII /NOT GOING LOW /
7404	052572	047111	020107	047514	
7405	052600	020127			
7406	052602	051117	052040	041515	.ASCIZ /OR TMCB E50(6) DIDN'T GO HIGH ON TMCC SERF(1)L/
7407	052610	020102	032505	024060	
7408	052616	024466	042040	042111	
7409	052624	023516	020124	047507	
7410	052632	044040	043511	020110	
7411	052640	047117	052040	041515	
7412	052646	020103	042523	043122	
7413	052654	030450	046051	000	
7414	052661	124	041515	020103	EM237: .ASCII /TMCC SERF(1) NOT GOING LOW OR/<CRLF>
7415	052666	042523	043122	030450	
7416	052674	020051	047516	020124	
7417	052702	047507	047111	020107	
7418	052710	047514	020127	051117	
7419	052716	200			
7420	052717	116	052117	043440	.ASCIZ /NOT GETTING TO TMCB E50(2&1)/
7421	052724	052105	044524	043516	
7422	052732	052040	020117	046524	
7423	052740	041103	042440	030065	
7424	052746	031050	030446	000051	
7425	052754	046524	041103	050040	EM240: .ASCII /TMCB PF(0)*(SF+-TF) NO' GOING HIGH OR/<CRLF>
7426	052762	024106	024460	024052	
7427	052770	043123	026453	043124	
7428	052776	020051	047516	020124	
7429	053004	047507	047111	020107	
7430	053012	044510	044107	047440	
7431	053020	100122			
7432	053022	047516	020124	042507	.ASCIZ /NOT GETTING TO RACK BRCAB04/
7433	053030	052124	047111	020107	
7434	053036	047524	051040	041501	
7435	053044	020113	051102	040503	
7436	053052	030102	000064		
7437	053056	041523	042503	051440	EM241: .ASCII /SCCE STACK OVERFLOW NOT GOING HIGH OR/<CRLF>
7438	053064	040524	045503	047440	
7439	053072	042526	043122	047514	
7440	053100	020127	047516	020124	
7441	053106	047507	047111	020107	
7442	053114	044510	044107	047440	
7443	053122	100122			

7444	053124	047516	020124	042507		.ASCIZ	/NOT GETTING TO TMCD E31 OR E31 BAD/
7445	053132	052124	047111	020107			
7446	053140	047524	052040	041515			
7447	053146	020104	031505	020061			
7448	053154	051117	042440	030463			
7449	53162	041040	042101	000			
7450	053167	120	051104	020103	EM242:	.ASCII	/PDRC RED ZONE NOT GOING HIGH OR/<CRLF>
7451	053174	042522	020104	047532			
7452	053202	042516	047040	052117			
7453	053210	043440	044517	043516			
7454	053216	044040	043511	020110			
7455	053224	051117	200				
7456	053227	116	052117	043440		.ASCIZ	/NOT GETTING TO TMCD E31 OR TMCD E31 BAD OR SL REG BIT 0 BAD/
7457	053234	052105	044524	043516			
7458	053242	052040	020117	046524			
7459	053250	042103	042440	030463			
7460	053256	047440	020122	046524			
7461	053264	042103	042440	030463			
7462	053272	041040	042101	047440			
7463	053300	020122	046123	051040			
7464	053306	043505	041040	052111			
7465	053314	030040	041040	042101			
7466	053322	000					
7467	053323	065	032062	030060	EM243:	.ASCIZ	/52400 PATTERN FAILED, 125000 PATTERN OK/
7468	053330	050040	052101	042524			
7469	053336	047122	043040	044501			
7470	053344	042514	026104	030440			
7471	053352	032462	030060	020060			
7472	053360	040520	052124	051105			
7473	053366	020116	045517	000			
7474	053373	105	051122	051117	DH243:	.ASCII	/ERRORPC SL REG TST NUM/<CRLF>
7475	053400	041520	020040	020040			
7476	053406	020040	046123	051040			
7477	053414	043505	020040	020040			
7478	053422	020040	051524	020124			
7479	053430	052516	100115				
7480	053434	020011	054105	042520		.ASCIZ	/ EXPECT ACTUAL/
7481	053442	052103	020040	041501			
7482	053450	052524	046101	000			
7483		05346					
7484	053456	001116	001162	001222	DT243:	.EVEN	\$ERRPC,\$TMPO,E2STKLM,\$\$TSTNM,0
7485	053464	001210	000000				
7486	053470	031061	030065	030060	EM244:	.ASCIZ	/125000 PATTERN FAILED 52400 PATTERN OK/
7487	053476	050040	052101	042524			
7488	053504	047122	043040	044501			
7489	053512	042514	020104	031065			
7490	053520	030064	020060	040520			
7491	053526	052124	051105	020116			
7492	053534	045517	000				
7493		053540					
7494	053540	001116	001162	001220	DT244:	.EVEN	\$ERRPC,\$TMPO,E1STKLM,\$\$TSTNM,0
7495	053546	001210	000000				
7496	053552	041523	042503	051440	EM245:	.ASCII	/SCCE SL ADDRESS NOT GETTING TO TMCD OR/<CRLF>
7497	053560	020114	042101	051104			
7498	053566	051505	020123	047516			
7499	053574	020124	042507	052124			

7500	053602	047111	020107	047524	
7501	053610	052040	041515	020104	
7502	053616	051117	200		
7503	053621	124	041515	020104	.ASCIZ /TMCD E28 OR E14 BAD/
7504	053626	031105	020070	051117	
7505	053634	042440	032061	041040	
7506	053642	042101	000		
7507	053645	124	041515	020104	EM246: .ASCII /TMCD LOW BYTE EN DOESN'T GO LOW OR/<CRLF>
7508	053652	047514	020127	054502	
7509	053660	042524	042440	020116	
7510	053666	047504	051505	023516	
7511	053674	020124	047507	046040	
7512	053702	053517	047440	100122	
7513	053710	047516	020124	042507	.ASCIZ /NOT GETTING THRU TO THE DMUX (PDRE) AS A LOW /
7514	053716	052124	047111	020107	
7515	053724	044124	052522	052040	
7516	053732	020117	044124	020105	
7517	053740	046504	054125	024040	
7518	053746	042120	042522	020051	
7519	053754	051501	040440	046040	
7520	053762	053517	000040		
7521	053766	046524	042103	042040	EM247: .ASCII /TMCD DMX S1 STUCK HIGH OR IT DOESN'T/<CRLF>
7522	053774	054115	051440	020061	
7523	054002	052123	041525	020113	
7524	054010	044510	044107	047440	
7525	054016	020122	052111	042040	
7526	054024	042517	047123	052047	
7527	054032	200			
7528	054033	107	052105	052040	.ASCIZ /GET THRU TO THE DMUX(PDRE) AS A LOW/
7529	054040	051110	020125	047524	
7530	054046	052040	042510	042040	
7531	054054	052515	024130	042120	
7532	054062	042522	020051	051501	
7533	054070	040440	046040	053517	
7534	054076	000			
7535	054077	102	052117	020110	EM250: .ASCIZ /BOTH PATTERNS FAILED/
7536	054104	040520	052124	051105	
7537	054112	051516	043040	044501	
7538	054120	042514	000104		
7539	054124	051105	047522	050122	DH250: .ASCII /ERRORPC SL REG SL REG TST NUM/<CRLF>
7540	054132	020103	020040	020040	
7541	054140	051440	020114	042522	
7542	054146	004507	020040	020040	
7543	054154	020040	046123	051040	
7544	054162	043505	020040	020040	
7545	054170	052040	052123	047040	
7546	054176	046525	200		
7547	054201	011	042440	050130	.ASCIZ / EXPECT ACTUAL EXPECT ACTUAL/
7548	054206	041505	020124	040440	
7549	054214	052103	040525	020114	
7550	054222	042440	050130	041505	
7551	054230	020124	040440	052103	
7552	054236	040525	000114		
7553					
7554	054242	001116	001162	001220	DT250: .EVEN
7555	054250	001164	001222	001210	.WORD \$ERRPT,\$TMP0,E1STKLM,\$TMP1,E2STKLM,\$STSTNM,C

7556	054256	000000			
7557	054260	046524	042103	054440	EM251: .ASCII /TMCD YEL ZONE DIDN'T GO LOW OR IT DIDN'T GET THRU TO E31/<CRLF>
7558	054266	046105	055040	047117	
7559	054274	020105	044504	047104	
7560	054302	052047	043440	020117	
7561	054310	047514	020127	051117	
7562	054316	044440	020124	044504	
7563	054324	047104	052047	043440	
7564	054332	052105	052040	051110	
7565	054340	020125	047524	042440	
7566	054346	030463	200		
7567	054351	117	020122	046524	.ASCIIZ /OR TMCE CACHE BEND DIDN'T GO HIGH ON SL RED/
7568	054356	042503	041440	041501	
7569	054364	042510	041040	047105	
7570	054372	020104	044504	047104	
7571	054400	052047	043440	020117	
7572	054406	044510	044107	047440	
7573	054414	020116	046123	051040	
7574	054422	042105	000		
7575	054425	124	041515	020104	EM252: .ASCIIZ /TMCD YEL ZONE DOESN'T GO LOW ON ADR 240/
7576	054432	042531	020114	047532	
7577	054440	042516	042040	042517	
7578	054446	047123	052047	043440	
7579	054454	020117	047514	020127	
7580	054462	047117	040440	051104	
7581	054470	031040	030064	000	
7582	054475	124	041515	020104	EM253: .ASCIIZ /TMCD YEL ZONE DOESN'T GO LOW ON ADR 140/
7583	054502	042531	020114	047532	
7584	054510	042516	042040	042517	
7585	054516	047123	052047	043440	
7586	054524	020117	047514	020127	
7587	054532	047117	040440	051104	
7588	054540	030440	030064	000	
7589	054545	124	041515	020104	EM254: .ASCIIZ /TMCD YEL ZONE DOESN'T GO HIGH OR DIDN'T GET THRU TO E31/
7590	054552	042531	020114	047532	
7591	054560	042516	042040	042517	
7592	054566	047123	052047	043440	
7593	054574	020117	044510	044107	
7594	054602	047440	020122	044504	
7595	054610	047104	052047	043440	
7596	054616	052105	052040	051110	
7597	054624	020125	047524	042440	
7598	054632	030463	000		
7599	054635	125	044516	052502	EM255: .ASCIIZ /UNIBUS TIMEOUT BIT IN CPU ERROR REG DIDN'T SET/
7600	054642	020123	044524	042515	
7601	054650	052517	020124	044502	
7602	054656	020124	047111	041440	
7603	054664	052520	042440	051122	
7604	054672	051117	051040	043505	
7605	054700	042040	042111	023516	
7606	054706	020124	042523	000124	
7607	054714	051105	047522	050122	DH255: .ASCII /ERRORPC CPUERR REG TST NUM/<CRLF>
7608	054722	020103	020040	041440	
7609	054730	052520	051105	020122	
7610	054736	042522	020107	020040	
7611	054744	051524	020124	052516	

7612	054752	100115			
7613	054754	020011	054105	042520	.ASCIZ / EXPECT ACTUAL/
7614	054762	052103	020040	041501	
7615	054770	052524	046101	000	
7616	054775	103	052520	042440	EM256: .ASCIZ /CPU ERRPROR REG DIDN'T CLEAR/
7617	055002	051122	051120	051117	
7618	055010	051040	043505	042040	
7619	055016	042111	023516	020124	
7620	055024	046103	040505	000122	
7621	055032	042531	020114	047532	EM257: .ASCIZ /YEL ZONE BIT IN CPU ERROR REG DIDN'T SET/
7622	055040	042516	041040	052111	
7623	055046	044440	020116	050103	
7624	055054	020125	051105	047522	
7625	055062	020122	042522	020107	
7626	055070	044504	047104	052047	
7627	055076	051440	052105	000	
7628	055103	124	041515	020104	EM260: .ASCIZ /TMCD E4(4) NOT GOING LOW OR E4 BAD/
7629	055110	032105	032050	020051	
7630	055116	047516	020124	047507	
7631	055124	047111	020107	047514	
7632	055132	020127	051117	042440	
7633	055140	020064	040502	000104	
7634	055146	042522	042101	055040	EM261: .ASCIZ /READ ZONE BIT IN CPU ERROR REG DIDN'T SET/
7635	055154	047117	020105	044502	
7636	055162	020124	047111	041440	
7637	055170	052520	042440	051122	
7638	055176	051117	051040	043505	
7639	055204	042040	042111	023516	
7640	055212	020124	042523	000124	
7641	055220	046524	042103	042440	EM262: .ASCIZ /TMCD E15(6) OR E18(14) OR E18 BAD/
7642	055226	032461	033050	020051	
7643	055234	051117	042440	034061	
7644	055242	030450	024464	047440	
7645	055250	020122	030505	020070	
7646	055256	040502	000104		
7647	055262	047506	046114	053517	EM263: .ASCII /FOLLOWING IS A LIST OF THE STACK LIMIT REG/<CRLF>
7648	055270	047111	020107	051511	
7649	055276	040440	046040	051511	
7650	055304	020124	043117	052040	
7651	055312	042510	051440	040524	
7652	055320	045503	046040	046511	
7653	055326	052111	051040	043505	
7654	055334	200			
7655	055335	046	051440	020120	.ASCII /& SP VALUES THAT CAUSED AN ERROR. THEY ARE/<CRLF>
7656	055342	040526	052514	051505	
7657	055350	052040	040510	020124	
7658	055356	040503	051525	042105	
7659	055364	040440	020116	051105	
7660	055372	047522	027122	052040	
7661	055400	042510	020131	051101	
7662	055406	100105			
7663	055410	051107	052517	042520	.ASCIZ /GROUPED ACCORDING TO ERROR TYPES/
7664	055416	020104	041501	047503	
7665	055424	042122	047111	020107	
7666	055432	047524	042440	051122	
7667	055440	051117	052040	050131	

```
7668 055446 051505 000
7669 055451 105 051122 051117 DH263: .ASCIIZ /ERRORPC TEST NUMBER/<CRLF>
7670 055456 041520 052040 051505
7671 055464 020124 052516 041115
7672 055472 051105 000200
7673 055476 051411 040524 045503 DH263A: .ASCII / STACK LIMIT REGISTER STACK POINTER/<CRLF>
7674 055504 046040 046511 052111
7675 055512 051040 043505 051511
7676 055520 042524 004522 020011
7677 055526 020040 020040 020040
7678 055534 052123 041501 020113
7679 055542 047520 047111 042524
7680 055550 100122
7681 055552 041517 040524 020114 .ASCII /OCTAL 15 14 13 12 11 10 9 8 OCTAL/
7682 055560 020040 030440 020065
7683 055566 032061 030440 020063
7684 055574 031061 030440 020061
7685 055602 030061 020040 020071
7686 055610 034040 020040 020040
7687 055616 020040 041517 040524
7688 055624 114
7689 055625 040 020040 030440 .ASCIIZ / 15 14 13 12 11 10 9 8/<CRLF>
7690 055632 020065 032061 030440
7691 055640 020063 031061 030440
7692 055646 020061 030061 020040
7693 055654 020071 034040 000200
7694 055662 042522 020104 051124 DH263B: .ASCIIZ /RED TRAP ON YEL ADR/
7695 055670 050101 047440 020116
7696 055676 042531 020114 042101
7697 055704 000122
7698 055706 042522 020104 051124 DH263C: .ASCIIZ /RED TRAP ON LEGAL ADR/
7699 055714 050101 047440 020116
7700 055722 042514 040507 020114
7701 055730 042101 000122
7702 055734 042531 046114 053517 DH263D: .ASCIIZ /YELLOW TRAP ON RED ADR/
7703 055742 052040 040522 020120
7704 055750 047117 051040 042105
7705 055756 040440 051104 000
7706 055763 131 046105 052040 DH263E: .ASCIIZ /YEL TRAP ON LEGAL ADR/
7707 055770 040522 020120 047117
7708 055776 046040 043505 046101
7709 056004 040440 051104 000
7710 056011 116 020117 051124 DH263F: .ASCIIZ /NO TRAP ON RED ADR/
7711 056016 050101 047440 020116
7712 056024 042522 020104 042101
7713 056032 000122
7714 056034 047516 052040 040522 DH263G: .ASCIIZ /NO TRAP ON YEL ADR/
7715 056042 020120 047117 054440
7716 056050 046105 040440 051104
7717 056056 000
7718 056060
7719 056060 055662 INDEX: .EVEN
7720 056062 055706 DH263B
7721 056064 055734 DH263C
7722 056066 055763 DH263D
7723 056070 056011 DH263E
DH263F
```

7724	056072	056034				DH263G
7725	056074	047507	047111	020107	EM264:	.ASCIZ /GOING TO NEXT TEST/
7726	056102	047524	047040	054105		
7727	056110	020124	042524	052123		
7728	056116	000				
7729						
7730	056117	116	052117	035105	EM265:	.ASCII /NOTE: IF NONE OF THE ODD ADR ERRORS TRAP/<CRLF>
7731	056124	044440	020106	047516		
7732	056132	042516	047440	020106		
7733	056140	044124	020105	042117		
7734	056146	020104	042101	020122		
7735	056154	051105	047522	051522		
7736	056162	052040	040522	100120		
7737	056170	020040	020040	020040	.ASCII /	THEN EITHER TMCC ODD ADRS ERR NOT GETTING TO/<CRLF>
7738	056176	044124	047105	042440		
7739	056204	052111	042510	020122		
7740	056212	046524	041503	047440		
7741	056220	042104	040440	051104		
7742	056226	020123	051105	020122		
7743	056234	047516	020124	042507		
7744	056242	052124	047111	020107		
7745	056250	047524	200			
7746	056253	040	020040	020040	.ASCII /	TMCC BUS ERROR OR DAPB BAMX00 NOT/<CRLF>
7747	056260	052040	041515	020103		
7748	056266	052502	020123	051105		
7749	056274	047522	020122	051117		
7750	056302	042040	050101	020102		
7751	056310	040502	054115	030060		
7752	056316	047040	052117	200		
7753	056323	040	020040	020040	.ASCII /	GETTING TO TMCC E7 AS A HIGH/<CRLF><CRLF>
7754	056330	043440	052105	044524		
7755	056336	043516	052040	020117		
7756	056344	046524	041503	042440		
7757	056352	020067	051501	040440		
7758	056360	044040	043511	100110		
7759	056366	200				
7760	056367	116	044505	044124	.ASCIZ /NEITHER -BYIN NOR DATI CAUSED A TRAP/<CRLF>	
7761	056374	051105	026440	054502		
7762	056402	047111	047040	051117		
7763	056410	042040	052101	020111		
7764	056416	040503	051525	042105		
7765	056424	040440	052040	040522		
7766	056432	100120	000			
7767	056435	111	041522	020104	EM266:	.ASCII /IRCD BYIN DOESN'T GET TO TMCC E12 AS A HIGH/<CRLF>
7768	056442	054502	047111	042040		
7769	056450	042517	047123	052047		
7770	056456	043440	052105	052040		
7771	056464	020117	046524	041503		
7772	056472	042440	031061	040440		
7773	056500	020123	020101	044510		
7774	056506	044107	200			
7775	056511	117	020122	046524	.ASCIZ /OR TMCC E12(9,8) BAD OR E7(2) BAD/	
7776	056516	041503	042440	031061		
7777	056524	034450	034054	020051		
7778	056532	040502	020104	051117		
7779	056540	042440	024067	024462		

7780	056546	041040	042101	000	
7781	056553	104	050101	020102	EM267: .ASCIZ /DAPB BAMX00 DOESN'T GET TO TMCC E7(4) OR E7 BAD/
7782	056560	040502	054115	030060	
7783	056566	042040	042517	047123	
7784	056574	052047	043440	052105	
7785	056602	052040	020117	046524	
7786	056610	041503	042440	024067	
7787	056616	024464	047440	020122	
7788	056624	033505	041040	042101	
7789	056632	000			
7790	056633	122	041501	020103	EM270: .ASCII /RACC UBSC00 DOESN'T GET TO TMCC E5(13) AS/<CRLF>
7791	056640	041125	041523	030060	
7792	056646	042040	042517	047123	
7793	056654	052047	043440	052105	
7794	056662	052040	020117	046524	
7795	056670	041503	042440	024065	
7796	056676	031461	020051	051501	
7797	056704	200			
7798	056705	101	046040	053517	.ASCIZ /A LOW OR E5(13) BAD/
7799	056712	047440	020122	032505	
7800	056720	030450	024463	041040	
7801	056726	042101	000		
7802	056731	122	041501	020103	EM271: .ASCII /RACC UBSC02 DOESN'T GET TO TMCC E12(5) AS/<CRLF>
7803	056736	041125	041523	031060	
7804	056744	042040	042517	047123	
7805	056752	052047	043440	052105	
7806	056760	052040	020117	046524	
7807	056766	041503	042440	031061	
7808	056774	032450	020051	051501	
7809	057002	200			
7810	057003	101	044040	043511	.ASCIZ /A HIGH OR E12(6) DOESN'T GO LOW OR E5(12) BAD/
7811	057010	020110	051117	042440	
7812	057016	031061	033050	020051	
7813	057024	047504	051505	023516	
7814	057032	020124	047507	046040	
7815	057040	053517	047440	020122	
7816	057046	032505	030450	024462	
7817	057054	041040	042101	000	
7818	057061	123	031515	033465	EM272: .ASCIZ /SM357*SRC1 DATI FAILED TO TRAP/
7819	057066	051452	041522	020061	
7820	057074	040504	044524	043040	
7821	057102	044501	042514	020104	
7822	057110	047524	052040	040522	
7823	057116	000120			
7824	057120	042117	020104	042101	EM273: .ASCIZ /ODD ADR BIT IN CPUERR REG DOESN'T SET/
7825	057126	020122	044502	020124	
7826	057134	047111	041440	052520	
7827	057142	051105	020122	042522	
7828	057150	020107	047504	051505	
7829	057156	023516	020124	042523	
7830	057164	000124			
7831	057166	047516	052040	040522	EM274: .ASCIZ /NO TRAP ON DATO/
7832	057174	020120	047117	042040	
7833	057202	052101	000117		
7834	057206	051520	032060	030450	EM275: .ASCII /PS04(1) DOESN'T GET TO TMCB E74(9) AS A HIGH/<CRLF>
7835	057214	020051	047504	051505	

7836	057222	023516	020124	042507
7837	057230	020124	047524	052040
7838	057236	041515	020102	033505
7839	057244	024064	024471	040440
7840	057252	020123	020101	044510
7841	057260	044107	200	
7842	057263	117	020122	052111
7843	057270	042040	042517	047123
7844	057276	052047	043440	052105
7845	057304	052040	020117	032505
7846	057312	024061	030061	020051
7847	057320	051501	040440	046040
7848	057326	053517	047440	020122
7849	057334	032505	020061	040502
7850	057342	100104		
7851	057344	051117	044440	041522
7852	057352	020104	052122	020124
7853	057360	047504	051505	023516
7854	057366	020124	042507	020124
7855	057374	047524	052040	041515
7856	057402	020102	033505	024064
7857	057410	030461	020051	051501
7858	057416	040440	044040	043511
7859	057424	100110		
7860	057426	051117	052040	041515
7861	057434	020102	047510	047516
7862	057442	020122	020124	044504
7863	057450	047104	052047	043440
7864	057456	020117	047514	020127
7865	057464	051117	052040	041515
7866	057472	020102	047524	020113
7867	057500	044504	100104	
7868	057504	047516	020124	047507
7869	057512	046040	053517	047440
7870	057520	020122	052111	042040
7871	057526	042111	023516	020124
7872	057534	042507	020124	044124
7873	057542	052522	052040	041515
7874	057550	020102	032505	000063
7875	057556	044502	020124	020064
7876	057564	047111	050040	053523
7877	057572	042040	042517	047123
7878	057600	052047	051440	052105
7879	057606	000		
7880	057607	123	052105	044524
7881	057614	043516	050040	036123
7882	057622	034060	020076	040506
7883	057630	046111	042105	052040
7884	057636	020117	044504	040523
7885	057644	046102	020105	020124
7886	057652	044502	020124	051124
7887	057660	050101	000	
7888	057663	124	041515	020102
7889	057670	047524	020113	047504
7890	057676	051505	023516	020124
7891	057704	042507	020124	047524

.ASCII /OR IT DOESN'T GET TO E51(10) AS A LOW OR E51 BAD/<CRLF>

.ASCII /OR IRCD RTT DOESN'T GET TO TMCB E74(11) AS A HIGH/<CRLF>

.ASCII /OR TMCB HONOR T DIDN'T GO LOW OR TMCB TOK DID/<CRLF>

.ASCIZ /NOT GO LOW OR IT DIDN'T GET THRU TMCB E53/

EM276: .ASCIZ /BIT 4 IN PSW DOESN'T SET/

EM277: .ASCIZ /SETTING PS<08> FAILED TO DISABLE T BIT TRAP/

EM300: .ASCIZ /TMCB TOK DOESN'T GET TO DAPE E7(11) AS A LOW OR E7 BAD/

7892	057712	042040	050101	020105
7893	057720	033505	030450	024461
7894	057726	040440	020123	020101
7895	057734	047514	020127	051117
7896	057742	042440	020067	040502
7897	057750	000104		
7898	057752	051111	042103	051040
7899	057760	052124	042040	042517
7900	057766	047123	052047	043440
7901	057774	052105	052040	020117
7902	060002	046524	041103	042440
7903	060010	032067	040440	020123
7904	060016	020101	047514	100127
7905	060024	051117	042440	032067
7906	060032	041040	042101	000
7907	060037	124	041515	020102
7908	060044	032505	024070	024465
7909	060052	042040	042111	023516
7910	060060	020124	047507	046040
7911	060066	053517	047440	020122
7912	060074	033105	020064	040502
7913	060102	000104		
7914	060104	046524	040503	040440
7915	060112	047502	042526	041040
7916	060120	033522	042040	042517
7917	060126	047123	052047	043440
7918	060134	020117	047514	020127
7919	060142	047117	040440	054440
7920	060150	046105	055040	047117
7921	060156	020105	051117	200
7922	060163	111	020124	047504
7923	060170	051505	023516	020124
7924	060176	042507	020124	047524
7925	060204	052040	041515	020102
7926	060212	034105	024064	031461
7927	060220	020051	051117	042440
7928	060226	032070	041040	042101
7929	060234	000		
7930	060235	124	041515	020101
7931	060242	041101	053117	020105
7932	060250	051102	020067	047504
7933	060256	051505	023516	020124
7934	060264	042507	020124	047524
7935	060272	052040	041515	020102
7936	060300	034105	024064	024461
7937	060306	047440	020122	034105
7938	060314	020064	040502	000104
7939	060322	046524	040503	040440
7940	060330	047502	042526	041040
7941	060336	033522	042040	042517
7942	060344	047123	052047	043440
7943	060352	052105	052040	020117
7944	060360	046524	041103	042440
7945	060366	033467	030450	024463
7946	060374	047440	020122	033505
7947	060402	020067	040502	000104

EM301: .ASCII /IRCD RTT DOESN'T GET TO TMCB E74 AS A LOW/<CRLF>

.ASCIZ /OR E74 BAD/

EM302: .ASCIZ /TMCB E58(5) DIDN'T GO LOW OR E64 BAD/

EM303: .ASCII /TMCA ABOVE BR7 DOESN'T GO LOW ON A YEL ZONE OR/<CRLF>

.ASCIZ /IT DOESN'T GET TO TMCB E84(13) OR E84 BAD/

EM304: .ASCIZ /TMCA ABOVE BR7 DOESN'T GET TO TMCB E84(1) OR E84 BAD/

EM305: .ASCIZ /TMCA ABOVE BR7 DOESN'T GET TO TMCB E77(13) OR E77 BAD/

7948	060410	046524	040503	041040	EM306: .ASCII /TMCA BLOCK BR4 DOESN'T GO LOW ON PIR4 OR/<CRLF>
7949	060416	047514	045503	041040	
7950	060424	032122	042040	042517	
7951	060432	047123	052047	043440	
7952	060440	020117	047514	020127	
7953	060446	047117	050040	051111	
7954	060454	020064	051117	200	
7955	060461	111	020124	047504	.ASCIZ /IT DOESN'T GET TO TMCB E77(5) OR E77 BAD/
7956	060466	051505	023516	020124	
7957	060474	042507	020124	047524	
7958	060502	052040	041515	020102	
7959	060510	033505	024067	024465	
7960	060516	047440	020122	033505	
7961	060524	020067	040502	000104	
7962	060532	046524	040503	044440	EM307: .ASCIZ /TMCA INH BELOW PIR4 DOESN'T GO LOW ON PIR5/
7963	060540	044116	041040	046105	
7964	060546	053517	050040	051111	
7965	060554	020064	047504	051505	
7966	060562	023516	020124	047507	
7967	060570	046040	053517	047440	
7968	060576	020116	044520	032522	
7969	060604	000			
7970	060605	124	041515	020101	EM310: .ASCIZ /TMCA INH BELOW PIR4 DOESN'T GO LOW ON BR5/
7971	060612	047111	020110	042502	
7972	060620	047514	020127	044520	
7973	060626	032122	042040	042517	
7974	060634	047123	052047	043440	
7975	060642	020117	047514	020127	
7976	060650	047117	041040	032522	
7977	060656	000			
7978	060657	124	041515	020101	EM311: .ASCIZ /TMCA BLOCK LEVEL 4 DIDN'T GO LOW ON PIR6/
7979	060664	046102	041517	020113	
7980	060672	042514	042526	020114	
7981	060700	020064	044504	047104	
7982	060706	052047	043440	020117	
7983	060714	047514	020127	047117	
7984	060722	050040	051111	000066	
7985	060730	046524	040503	040440	EM312: .ASCII /TMCA ABOVE BR7 DOESN'T GO LOW(ON PIR 7) OR/<CRLF>
7986	060736	047502	042526	041040	
7987	060744	033522	042040	042517	
7988	060752	047123	052047	043440	
7989	060760	020117	047514	024127	
7990	060766	047117	050040	051111	
7991	060774	033440	020051	051117	
7992	061002	200			
7993	061003	111	020124	047504	.ASCIZ /IT DOESN'T GET TO TMCB E77(1) OR E77 BAD/
7994	061010	051505	023516	020124	
7995	061016	042507	020124	047524	
7996	061024	052040	041515	020102	
7997	061032	033505	024067	024461	
7998	061040	047440	020122	033505	
7999	061046	020067	040502	000104	
8000	061054	041125	042103	042440	EM313: .ASCIZ /UBCD EXT BRQ DIDN'T GO LOW ON HONOR BR5/
8001	061062	052130	041040	050522	
8002	061070	042040	042111	023516	
8003	061076	020124	047507	046040	

8004	061104	053517	047440	020116	
8005	061112	047510	047516	020122	
8006	061120	051102	000065		
8007	061124	047111	020110	042502	EM314: .ASCII /INH BELOW BR6 DOESN'T GO LOW(ON BR6) OR IT DOES/<CRLF>
8008	061132	047514	020127	051102	
8009	061140	020066	047504	051505	
8010	061146	023516	020124	047507	
8011	061154	046040	053517	047450	
8012	061162	020116	051102	024466	
8013	061170	047440	020122	052111	
8014	061176	042040	042517	100123	
8015	061204	047516	020124	042507	.ASCIZ /NOT GET THRU TMCA E82(6)/
8016	061212	020124	044124	052522	
8017	061220	052040	041515	020101	
8018	061226	034105	024062	024466	
8019	061234	000			
8020	061235	124	041515	020101	EM315: .ASCII /TMCA E67(8) DOESN'T GO LOW (ON SL YEL) OR IT IS NOT/<CRLF>
8021	061242	033105	024067	024470	
8022	061250	042040	042517	047123	
8023	061256	052047	043440	020117	
8024	061264	047514	020127	047450	
8025	061272	020116	046123	054440	
8026	061300	046105	020051	051117	
8027	061306	044440	020124	051511	
8028	061314	047040	052117	200	
8029	061321	107	052105	044524	.ASCIZ /GETTING TO E60(12) OR E60 BAD/
8030	061326	043516	052040	020117	
8031	061334	033105	024060	031061	
8032	061342	020051	051117	042440	
8033	061350	030066	041040	042101	
8034	061356	000			
8035	061357	124	041515	020101	EM316: .ASCIZ /TMCA E81(12) DOESN'T GO LOW(ON PIR5) OR IT DOES/<CRLF>
8036	061364	034105	024061	031061	
8037	061372	020051	047504	051505	
8038	061400	023516	020124	047507	
8039	061406	046040	053517	047450	
8040	061414	020116	044520	032522	
8041	061422	020051	051117	044440	
8042	061430	020124	047504	051505	
8043	061436	000200			
8044	061440	042507	020124	047524	.ASCIZ /GET TO E83(10) OR E83 BAD/
8045	061446	042440	031470	030450	
8046	061454	024460	047440	020122	
8047	061462	034105	020063	040502	
8048	061470	000104			
8049	061472	046524	040503	042440	EM317: .ASCIZ /TMCA E81(2) BAD/
8050	061500	030470	031050	020051	
8051	061506	040502	000104		
8052	061512	046524	040503	042440	EM320: .ASCII /TMCA E67(8) DOESN'T GO LOW(ON PIR 7) OR IT DOES/<CRLF>
8053	061520	033466	034050	020051	
8054	061526	047504	051505	023516	
8055	061534	020124	047507	046040	
8056	061542	053517	047450	020116	
8057	061550	044520	020122	024467	
8058	061556	047440	020122	052111	
8059	061564	042040	042517	100123	

8060	061572	047516	020124	042507		.ASCIIZ /NOT GET TO E83(13) OR E83 BAD/
8061	061600	020124	047524	042440		
8062	061606	031470	030450	024463		
8063	061614	047440	020122	034105		
8064	061622	020063	040502	000104		
8065	061630	041125	042103	042440	EM321:	.ASCIIZ /UBCD EXT BRQ DIDN'T GO LOW ON HONOR BR6/
8066	061636	052130	041040	050522		
8067	061644	042040	042111	023516		
8068	061652	020124	047507	046040		
8069	061660	053517	047440	020116		
8070	061666	047510	047516	020122		
8071	061674	051102	000066			
8072	061700	046524	040503	042440	EM322:	.ASCIIZ /TMCA E67(8) DOESN'T GET TO E83(1) OR E83 BAD/
8073	061706	033466	034050	020051		
8074	061714	047504	051505	023516		
8075	061722	020124	042507	020124		
8076	061730	047524	042440	031470		
8077	061736	030450	020051	051117		
8078	061744	042440	031470	041040		
8079	061752	042101	000			
8080	061755	124	041515	020101	EM323:	.ASCII /TMCA E81(8) IS NOT GOING LOW OR IT IS NOT/<CRLF>
8081	061762	034105	024061	024470		
8082	061770	044440	020123	047516		
8083	061776	020124	047507	047111		
8084	062004	020107	047514	020127		
8085	062012	051117	044440	020124		
8086	062020	051511	047040	052117		
8087	062026	200				
8088	062027	107	052105	044524		.ASCIIZ /GETTING TO E76(12) OR E76 BAD/
8089	062034	043516	052040	020117		
8090	062042	033505	024066	031061		
8091	062050	020051	051117	042440		
8092	062056	033067	041040	042101		
8093	062064	000				
8094	062065	124	041515	020101	EM324:	.ASCIIZ /TMCA E67(8) NOT GETTING TO E76(13) OR E76 IS BAD/
8095	062072	033105	024067	024470		
8096	062100	047040	052117	043440		
8097	062106	052105	044524	043516		
8098	062114	052040	020117	033505		
8099	062122	024066	031461	020051		
8100	062130	051117	042440	033067		
8101	062136	044440	020123	040502		
8102	062144	000104				
8103	062146	046524	040503	042440	EM325:	.ASCIIZ /TMCA E67(8) DOESN'T GET TO E76(1) OR E76 BAD/
8104	062154	033466	034050	020051		
8105	062162	047504	051505	023516		
8106	062170	020124	042507	020124		
8107	062176	047524	042440	033067		
8108	062204	030450	020051	051117		
8109	062212	042440	033067	041040		
8110	062220	042101	000			
8111	062223	124	041515	020101	EM326:	.ASCIIZ /TMCA E67(6) NOT GETTING TO E69(12) OR E69 IS BAD/
8112	062230	033105	024067	024466		
8113	062236	047040	052117	043440		
8114	062244	052105	044524	043516		
8115	062252	052040	020117	033105		

8116 062260 024071 031061 020051
8117 062266 051117 042440 034466
8118 062274 044440 020123 040502
8119 062302 000104
8120 062304 051520 020127 044502
8121 062312 020124 030461 042040
8122 062320 042517 047123 052047
8123 062326 051440 052105 047440
8124 062334 020122 046524 043103
8125 062342 041440 045514 044040
8126 062350 020111 051520 042040
8127 062356 042517 100123
8128 062362 047516 020124 047507
8129 062370 046040 053517 047440
8130 062376 020122 044502 020124
8131 062404 030461 042040 042517
8132 062412 047123 052047 043440
8133 062420 052105 052040 020117
8134 062426 051117 052040 051110
8135 062434 020125 044124 020105
8136 062442 046504 054125 000
8137 062447 107 040522 020103
8138 062454 042107 042522 020107
8139 062462 042523 020124 020061
8140 062470 047504 051505 023516
8141 062476 020124 047507 044040
8142 062504 043511 020110 047450
8143 062512 020116 040520 020104
8144 062520 024465 047440 100122
8145 062526 047504 051505 023516
8146 062534 020124 042507 020124
8147 062542 047524 052040 042510
8148 062550 043440 020104 042522
8149 062556 000107
8150 062560 051107 041501 043440
8151 062566 051123 043505 051440
8152 062574 052105 030440 042040
8153 062602 042517 047123 052047
8154 062610 043440 020117 044510
8155 062616 044107 047450 020116
8156 062624 040520 020104 024465
8157 062632 047440 100122
8158 062636 052111 042040 042517
8159 062644 047123 052047 043440
8160 062652 052105 052040 020117
8161 062660 044124 020105 051507
8162 062666 051040 043505 000
8163 062673 107 040522 020102
8164 062700 051123 020103 042523
8165 062706 020124 020061 047504
8166 062714 051505 023516 020124
8167 062722 047507 046040 053517
8168 062730 047440 020122 051107
8169 062736 041501 042440 031462
8170 062744 033050 020051 040502
8171 062752 000104

EM327: .ASCII /PSW BIT 11 DOESN'T SET OR TMCF CLK HI PS DOES/<<CRLF>

.ASCIZ /NOT GO LOW OR BIT 11 DOESN'T GET TO OR THRU THE DMUX/

EM330: .ASCII /GRAC GDREG SET 1 DOESN'T GO HIGH (ON PAD 5) OR/<<CRLF>

.ASCIZ /DOESN'T GET TO THE GD REG/

EM331: .ASCII /GRAC GSREG SET 1 DOESN'T GO HIGH(ON PAD 5) OR/<<CRLF>

.ASCIZ /IT DOESN'T GET TO THE GS REG/

EM332: .ASCIZ /GRAB SRC SET 1 DOESN'T GO LOW OR GRAC E23(6) BAD/

8172	062754	051107	041501	042440	EM333:	.ASCIZ	/GRAC E24(6) DOESN'T GO LOW/
8173	062762	032062	033050	020051			
8174	062770	047504	051505	023516			
8175	062776	020124	047507	046040			
8176	063004	053517	000				
8177	063007	107	040522	020103	EM334:	.ASCIZ	/GRAC E23(4) DOESN'T GO LOW/
8178	063014	031105	024063	024464			
8179	063022	042040	042517	047123			
8180	063030	052047	043440	020117			
8181	063036	047514	000127				
8182	063042	051107	041501	042440	EM335:	.ASCIZ	/GRAC E23(4) IS NOT GOING LOW/
8183	063050	031462	032050	020051			
8184	063056	051511	047040	052117			
8185	063064	043440	044517	043516			
8186	063072	046040	053517	000			
8187	063077	120	051104	020104	EM336:	.ASCIZ	/PDRD PS11 DOESN'T GET TO GRAC AS A LOW/
8188	063104	051520	030461	042040			
8189	063112	042517	047123	052047			
8190	063120	043440	052105	052040			
8191	063126	020117	051107	041501			
8192	063134	040440	020123	020101			
8193	063142	047514	000127				
8194	063146	042522	044507	052123	EM337:	.ASCIZ	/REGISTER SET 1 SOURCE HAS STUCK BITS/
8195	063154	051105	051440	052105			
8196	063162	030440	051440	052517			
8197	063170	041522	020105	040510			
8198	063176	020123	052123	041525			
8199	063204	020113	044502	051524			
8200	063212	000					
8201	063213	105	051122	051117	DH337:	.ASCIZ	/ERRORPC PATTERN TEST NUMBER/
8202	063220	041520	050040	052101			
8203	063226	042524	047122	052040			
8204	063234	051505	020124	052516			
8205	063242	041115	051105	000			
8206		063250					
8207	063250	001116	001164	001210	DT337:	.EVEN .WORD	\$ERRPL,\$TMP1,\$STSTNM,0
8208	063256	000000					
8209	063260	042522	044507	052123	EM340:	.ASCIZ	/REGISTER SET 1 DESTINATION HAS STUCK BITS/
8210	063266	051105	051440	052105			
8211	063274	030440	042040	051505			
8212	063302	044524	040516	044524			
8213	063310	047117	044040	051501			
8214	063316	051440	052524	045503			
8215	063324	041040	052111	000123			
8216	063332	051107	041101	042040	EM341:	.ASCIZ	/GRAB DST SET 1 DOESN'T GO LOW ON R14/
8217	063340	052123	051440	052105			
8218	063346	030440	042040	042517			
8219	063354	047123	052047	043440			
8220	063362	020117	047514	020127			
8221	063370	047117	051040	032061			
8222	063376	000					
8223	063377	107	040522	020102	EM342:	.ASCIZ	/GRAB SRC SET 1 DOESN'T GO LOW ON R14/
8224	063404	051123	020103	042523			
8225	063412	020124	020061	047504			
8226	063420	051505	023516	020124			
8227	063426	047507	046040	053517			

8228	063434	047440	020116	030522	
8229	063442	000064			
8230	063444	030065	030060	020060	EM343: .ASCIZ /50000 PATTERN FAILED IN PSW/
8231	063452	040520	052124	051105	
8232	063460	020116	040506	046111	
8233	063466	042105	044440	020116	
8234	063474	051520	000127		
8235	063500	033061	030064	030060	EM344: .ASCIZ /164000 PATTERN FAILED IN PSW/
8236	063506	050040	052101	042524	
8237	063514	047122	043040	044501	
8238	063522	042514	020104	047111	
8239	063530	050040	053523	000	
8240	063535	120	053523	044040	EM345: .ASCIZ /PSW HIGH BYTE DIDN'T CLEAR/
8241	063542	043511	020110	054502	
8242	063550	042524	042040	042111	
8243	063556	023516	020124	046103	
8244	063564	040505	000122		
8245	063570	051107	041101	042040	EM346: .ASCIZ /GRAB DST SET 1 DOESN'T GO LOW ON DF6*SUPER MODE/
8246	063576	052123	051440	052105	
8247	063604	030440	042040	045517	
8248	063612	047123	052047	043440	
8249	063620	020117	047514	020127	
8250	063626	047117	042040	033106	
8251	063634	051452	050125	051105	
8252	063642	046440	042117	000105	
8253	063650	051107	041101	051440	EM347: .ASCIZ /GRAB SRC SET 1 DOESN'T GO LOW ON SF6*SUPER MODE/
8254	063656	041522	051440	052105	
8255	063664	030440	042040	042517	
8256	063672	047123	052047	043440	
8257	063700	020117	047514	020127	
8258	063706	047117	051440	033106	
8259	063714	051452	050125	051105	
8260	063722	046440	042117	000105	
8261	063730	051107	041501	042440	EM350: .ASCIZ /GRAC E35(8) DOESN'T GO HIGH ON DF6*USER MODE/
8262	063736	032463	034050	020051	
8263	063744	047504	051505	023516	
8264	063752	020124	047507	044040	
8265	063760	043511	020110	047117	
8266	063766	042040	033106	052452	
8267	063774	042523	020122	047515	
8268	064002	042504	000		
8269	064005	107	040522	020103	EM351: .ASCIZ /GRAC E15(6) DOESN'T GO HIGH ON SF6*USER MODE/
8270	064012	030505	024065	024466	
8271	064020	042040	042517	047123	
8272	064026	052047	043440	020117	
8273	064034	044510	044107	047440	
8274	064042	020116	043123	025066	
8275	064050	051525	051105	046440	
8276	064056	042117	000105		
8277	064062	051525	051105	047440	EM352: .ASCIZ /USER OR SUPER SP DST FAILED BIT TEST/
8278	064070	020122	052523	042520	
8279	064076	020122	050123	042040	
8280	064104	052123	043040	044501	
8281	064112	042514	020104	044502	
8282	064120	020124	042524	052123	
8283	064126	000			

8284	064127	125	042523	020122	EM353: .ASCIZ /USER OR SUPER SP SRC FAILED BIT TEST/
8285	064134	051117	051440	050125	
8286	064142	051105	051440	020120	
8287	064150	051123	020103	040506	
8288	064156	046111	042105	041040	
8289	064164	052111	052040	051505	
8290	064172	000124			
8291	064174	051107	041501	042440	EM354: .ASCIZ /GRAC E20(4) NOT GOING LOW/
8292	064202	030062	032050	020051	
8293	064210	047516	020124	047507	
8294	064216	047111	020107	047514	
8295	064224	000127			
8296	064226	051107	041501	042440	EM355: .ASCIZ /GRAC E20(12) NOT GOING LOW ON MTP*DMO*DF6*PS12/
8297	064234	030062	030450	024462	
8298	064242	047040	052117	043440	
8299	064250	044517	043516	046040	
8300	064256	053517	047440	020116	
8301	064264	052115	025120	046504	
8302	064272	025060	043104	025066	
8303	064300	051520	031061	000	
8304	064305	107	040522	020103	EM356: .ASCIZ /GRAC E20(11) NOT GOING LOW/
8305	064312	031105	024060	030461	
8306	064320	020051	047516	020124	
8307	064326	047507	047111	020107	
8308	064334	047514	000127		
8309	064340	051523	020120	040502	EM357: .ASCII /SSP BAD ON MTP EITHER GRAC GRWF/<CRLF>
8310	064346	020104	047117	046440	
8311	064354	050124	042440	052111	
8312	064362	042510	020122	051107	
8313	064370	041501	043440	053522	
8314	064376	100105			
8315	064400	044510	020102	051117	.ASCIZ /HIB OR LOB NOT GOING LOW/
8316	064406	046040	041117	047040	
8317	064414	052117	043440	044517	
8318	064422	043516	046040	053517	
8319	064430	000			
8320	064431	105	051122	051117	DH357: .ASCII /ERRORPC SSP TST NUM/<CRLF>
8321	064436	041520	020040	020040	
8322	064444	020040	051523	004520	
8323	064452	051524	020124	052516	
8324	064460	100115			
8325	064462	020011	054105	042520	.ASCIZ / EXPECT ACTUAL/
8326	064470	052103	020040	041501	
8327	064476	052524	046101	000	
8328	064503	107	040522	020103	EM360: .ASCIZ /GRAC E35(8) DIDN'T GO HIGH ON MTP*DMO*DF6*PS13/
8329	064510	031505	024065	024470	
8330	064516	042040	042111	023516	
8331	064524	020124	047507	044040	
8332	064532	043511	020110	047117	
8333	064540	046440	050124	042052	
8334	064546	030115	042052	033106	
8335	064554	050052	030523	000063	
8336	064562	051111	041503	042040	EM361: .ASCIZ /IRCC DMO(MFP+MTP) NOT GOING HIGH ON MFP/
8337	064570	030115	046450	050106	
8338	064576	046453	050124	020051	
8339	064604	047516	020124	047507	

8340	064612	047111	020107	044510	
8341	064620	044107	047440	020116	
8342	064626	043115	000120		
8343	064632	051107	041501	042440	EM362: .ASCIZ /GRAC E24(13) DOESN'T GO HIGH/
8344	064640	032062	030450	024463	
8345	064646	042040	042517	047123	
8346	064654	052047	043440	020117	
8347	064662	044510	044107	000	
8348	064667	120	051104	020104	EM363: .ASCIZ /PDRD PS15 DOESN'T GET TO GRAC AS A HIGH/
8349	064674	051520	032461	042040	
8350	064702	042517	047123	052047	
8351	064710	043440	052105	052040	
8352	064716	020117	051107	041501	
8353	064724	040440	020123	020101	
8354	064732	044510	044107	000	
8355	064737	123	050123	053440	EM364: .ASCIZ /SSP WAS READ BUT NEITHER USP NOR SSP WERE WRITTEN/
8356	064744	051501	051040	040505	
8357	064752	020104	052502	020124	
8358	064760	042516	052111	042510	
8359	064766	020122	051525	020120	
8360	064774	047516	020122	051523	
8361	065002	020120	042527	042522	
8362	065010	053440	044522	052124	
8363	065016	047105	000		
8364	065021	107	040522	020103	EM365: .ASCIZ /GRAC E23(13) DOESN'T GO HIGH/
8365	065026	031105	024063	031461	
8366	065034	020051	047504	051505	
8367	065042	023516	020124	047507	
8368	065050	044040	043511	000110	
8369	065056	051107	041501	042440	EM366: .ASCIZ /GRAC E23(3) DOESN'T GO HIGH/
8370	065064	031462	031450	020051	
8371	065072	047504	051505	023516	
8372	065100	020124	047507	044040	
8373	065106	043511	000110		
8374	065112	042120	042122	050040	EM367: .ASCII /PDRD PS14(0) DOESN'T GET TO PDRD E73(13)/<CRLF>
8375	065120	030523	024064	024460	
8376	065126	042040	042517	047123	
8377	065134	052047	043440	052105	
8378	065142	052040	020117	042120	
8379	065150	042122	042440	031467	
8380	065156	030450	024463	200	
8381	065163	101	020123	020101	.ASCIZ /AS A HIGH OR E73(12) NOT GOING LOW/
8382	065170	044510	044107	047440	
8383	065176	020122	033505	024063	
8384	065204	031061	020051	047516	
8385	065212	020124	047507	047111	
8386	065220	020107	047514	000127	
8387	065226	020101	051520	020127	EM370: .ASCII /A PSW HIGH BYTE BIT(S) DIDN'T SET ON LOAD PS 8/<CRLF>
8388	065234	044510	044107	041040	
8389	065242	052131	020105	044502	
8390	065250	024124	024523	042040	
8391	065256	042111	023516	020124	
8392	065264	042523	020124	047117	
8393	065272	046040	040517	020104	
8394	065300	051520	023040	200	
8395	065305	113	051105	042516	.ASCIZ /KERNEL MODE WITH A BR VALUE OF 174000/

8396	065312	020114	047515	042504
8397	065320	053440	052111	020110
8398	065326	020101	051102	053040
8399	065334	046101	042525	047440
8400	065342	020106	033461	030064
8401	065350	030060	000	
8402	065353	101	041040	052111
8403	065360	051450	020051	047111
8404	065366	052040	042510	050040
8405	065374	053523	041440	042514
8406	065402	051101	042105	047440
8407	065410	020116	047101	051040
8408	065416	044524	044440	020116
8409	065424	051525	051105	046440
8410	065432	042117	100105	
8411	065436	020046	020101	051102
8412	065444	053040	046101	042525
8413	065452	047440	020106	000060
8414	065460	020101	044502	024124
8415	065466	024523	047440	020106
8416	065474	051520	020127	030474
8417	065502	026065	031461	030472
8418	065510	037061	042040	042111
8419	065516	023516	020124	051120
8420	065524	051505	052105	047440
8421	065532	020116	047101	051040
8422	065540	044524	200	
8423	065543	111	020116	052523
8424	065550	042520	020122	047515
8425	065556	042504	023040	040440
8426	065564	041040	020122	040526
8427	065572	052514	020105	043117
8428	065600	030440	032063	030060
8429	065606	000060		
8430	065610	020101	044502	024124
8431	065616	024523	044440	020116
8432	065624	051520	020127	044510
8433	065632	041107	041040	052131
8434	065640	020105	040506	046111
8435	065646	042105	047440	020116
8436	065654	047101	044440	052117
8437	065662	044440	020116	051525
8438	065670	051105	200	
8439	065673	115	042117	020105
8440	065700	020046	020101	051102
8441	065706	053040	046101	042525
8442	065714	047440	020106	000060
8443	065722	020101	051120	053105
8444	065730	047511	051525	046440
8445	065736	042117	020105	044502
8446	065744	024124	024523	042040
8447	065752	042111	023516	020124
8448	065760	046103	040505	020122
8449	065766	047117	040440	020116
8450	065774	047511	020124	047111
8451	066002	045440	051105	042516

EM371: .ASCII /A BIT(S) IN THE PSW CLEARED ON AN RTI IN USER MODE/<CRLF>

.ASCIZ /& A BR VALUE OF 0/

EM372: .ASCII /A BIT(S) OF PSW <15,13:11> DIDN'T PRESET ON AN RTI/<CRLF>

.ASCIZ /IN SUPER MODE & A BR VALUE OF 134000/

EM373: .ASCII /A BIT(S) IN PSW HIGH BYTE FAILED ON AN IOT IN USER/<CRLF>

.ASCIZ /MODE & A BR VALUE OF 0/

EM374: .ASCIZ /A PREVIOUS MODE BIT(S) DIDN'T CLEAR ON AN IOT IN KERNEL/

8452	066010	000114			
8453	066012	051523	041522	045440	EM375: .ASCII /SSRC KT ABORT FLG DOESN'T GET TO TMCC E34(10) OR/<CRLF>
8454	066020	020124	041101	051117	
8455	066026	020124	046106	020107	
8456	066034	047504	051505	023516	
8457	066042	020124	042507	020124	
8458	066050	047524	052040	041515	
8459	066056	020103	031505	024064	
8460	066064	030061	020051	051117	
8461	066072	200			
8462	066073	105	032063	030450	.ASCIZ /E34(10) BAD OR TMCC AERF(1) DOESN'T GO HIGH ON A KT ABORT/
8463	066100	024460	041040	042101	
8464	066106	047440	020122	046524	
8465	066114	041503	040440	051105	
8466	066122	024106	024461	042040	
8467	066130	042517	047123	052047	
8468	066136	043440	020117	044510	
8469	066144	044107	047440	020116	
8470	066152	020101	052113	040440	
8471	066160	047502	052122	000	
8472	066165	120	053523	041040	EM376: .ASCIZ /PSW BIT 8 FAILED TO SET/
8473	066172	052111	034040	043040	
8474	066200	044501	042514	020104	
8475	066206	047524	051440	052105	
8476	066214	000			
8477	066215	124	041515	020102	EM400: .ASCII /TMCB SEGT DOESN'T GO LOW ON A KT ABORT OR/<CRLF>
8478	066222	042523	052107	042040	
8479	066230	042517	047123	052047	
8480	066236	043440	020117	047514	
8481	066244	020127	047117	040440	
8482	066252	045440	020124	041101	
8483	066260	051117	020124	051117	
8484	066266	200			
8485	066267	111	020124	047504	.ASCIZ /IT DOESN'T GET TO DAPE/
8486	066274	051505	023516	020124	
8487	066302	042507	020124	047524	
8488	066310	042040	050101	000105	
8489	066316	040504	042520	052040	EM401: .ASCIZ /DAPE TV05*07 DOESN'T GO HIGH ON SEGT/
8490	066324	030126	025065	033460	
8491	066332	042040	042517	047123	
8492	066340	052047	043440	020117	
8493	066346	044510	044107	047440	
8494	066354	020116	042523	052107	
8495	066362	000			
8496	066363	104	050101	020105	EM402: .ASCIZ /DAPE TV03 DOESN'T GO HIGH ON SEGT/
8497	066370	053124	031460	042040	
8498	066376	042517	047123	052047	
8499	066404	043440	020117	044510	
8500	066412	044107	047440	020116	
8501	066420	042523	052107	000	
8502	066425	124	041515	020101	EM403: .ASCII /TMCA SEG+CON+PAR DOESN'T GO LOW OR IT DOES/<CRLF>
8503	066432	042523	025507	047503	
8504	066440	025516	040520	020122	
8505	066446	047504	051505	023516	
8506	066454	020124	047507	046040	
8507	066462	053517	047440	020122	

8508	066470	052111	042040	042517	
8509	066476	100123			
8510	066500	047516	020124	042507	.ASCIIZ /NOT GET THRU TMCB E70(2)/
8511	066506	020124	044124	052522	
8512	066514	052040	041515	020102	
8513	066522	033505	024060	024462	
8514	066530	000			
8515	066531	124	041515	020101	EM405: .ASCII /TMCA SEGTF DOESN'T GET TO E44 OR TMCE PAUSES H/<15><12>
8516	066536	042523	052107	020106	
8517	066544	047504	051505	023516	
8518	066552	020124	042507	020124	
8519	066560	047524	042440	032064	
8520	066566	047440	020122	046524	
8521	066574	042503	050040	052501	
8522	066602	042523	020123	006510	
8523	066610	012			
8524	066611	104	042517	047123	.ASCIIZ /DOESN'T GET TO E44 OR E44 BAD/
8525	066616	052047	043440	052105	
8526	066624	052040	020117	032105	
8527	066632	020064	051117	042440	
8528	066640	032064	041040	042101	
8529	066646	000			
8530	066647	124	041515	020103	EM406: .ASCII /TMCC NEXM DOESN'T GO LOW OR IT DOESN'T GET THRU E34/<<CRLF>
8531	066654	042516	046530	042040	
8532	066662	042517	047123	052047	
8533	066670	043440	020117	047514	
8534	066676	020127	051117	044440	
8535	066704	020124	047504	051505	
8536	066712	023516	020124	042507	
8537	066720	020124	044124	052522	
8538	066726	042440	032063	200	
8539	066733	117	020122	052111	.ASCIIZ /OR IT DOESN'T GET TO E14 OR E40 BAD/
8540	066740	042040	042517	047123	
8541	066746	052047	043440	052105	
8542	066754	052040	020117	030505	
8543	066762	020064	051117	042440	
8544	066770	030064	041040	042101	
8545	066776	000			
8546	066777	116	054105	020115	EM410: .ASCII /NEXM BIT DIDN'T SET IN CPU ERROR REG/<CRLF>
8547	067004	044502	020124	044504	
8548	067012	047104	052047	051440	
8549	067020	052105	044440	020116	
8550	067026	050103	020125	051105	
8551	067034	047522	020122	042522	
8552	067042	100107			
8553	067044	051117	052040	041515	.ASCIIZ /OR TMCC ABORT DOESN'T GO HIGH/
8554	067052	020103	041101	051117	
8555	067060	020124	047504	051505	
8556	067066	023516	020124	047507	
8557	067074	044040	043511	000110	
8558	067102	042516	046530	041040	EM411: .ASCIIZ /NEXM BIT DIDN'T CLEAR IN CPU ERROR REG/
8559	067110	052111	042040	042111	
8560	067116	023516	020124	046103	
8561	067124	040505	020122	047111	
8562	067132	041440	052520	042440	
8563	067140	051122	051117	051040	

8564	067146	043505	000		
8565	067151	124	041515	020105	EM412: .ASCIZ /TMCE KT BEND DOESN'T GO LOW ON TMCC NEXM LOW/
8566	067156	052113	041040	047105	
8567	067164	020104	047504	051505	
8568	067172	023516	020124	047507	
8569	067200	046040	053517	047440	
8570	067206	020116	046524	041503	
8571	067214	047040	054105	020115	
8572	067222	047514	000127		
8573	067226	046524	042503	045440	EM413: .ASCIZ /TMCE KT BEND DOESN'T GO LOW ON TMCD SL RED/
8574	067234	020124	042502	042116	
8575	067242	042040	042517	047123	
8576	067250	052047	043440	020117	
8577	067256	047514	020127	047117	
8578	067264	052040	041515	020104	
8579	067272	046123	051040	042105	
8580	067300	000			
8581	067301	124	041515	020105	EM414: .ASCIZ /TMCE KT BEND DOESN'T GO LOW ON TMCC ODD ADRS ERR/
8582	067306	052113	041040	047105	
8583	067314	020104	047504	051505	
8584	067322	023516	020124	047507	
8585	067330	046040	053517	047440	
8586	067336	020116	046524	041503	
8587	067344	047440	042104	040440	
8588	067352	051104	020123	051105	
8589	067360	000122			
8590	067362	052113	040440	047502	EM415: .ASCIZ ?KT ABORT FAILED TO OVER-RIDE NEXM TRAP (KB11-E/EM)?
8591	067370	052122	043040	044501	
8592	067376	042514	020104	047524	
8593	067404	047440	042526	026522	
8594	067412	044522	042504	047040	
8595	067420	054105	020115	051124	
8596	067426	050101	024040	041113	
8597	067434	030461	042455	042457	
8598	067442	024515	000		
8599	067445	124	041515	020105	EM416: .ASCIZ /TMCE CACHE BEND DIDN'T GO HIGH ON KT ABORT/
8600	067452	040503	044103	020105	
8601	067460	042502	042116	042040	
8602	067466	042111	023516	020124	
8603	067474	047507	044040	043511	
8604	067502	020110	047117	045440	
8605	067510	020124	041101	051117	
8606	067516	000124			
8607	067520	040504	040520	041040	EM417: .ASCII /DAPA BR14 DOESN'T GET TO RACK E49 AS A LOW/<CRLF>
8608	067526	030522	020064	047504	
8609	067534	051505	023516	020124	
8610	067542	042507	020124	047524	
8611	067550	051040	041501	020113	
8612	067556	032105	020071	051501	
8613	067564	040440	046040	053517	
8614	067572	200			
8615	067573	117	020122	032105	.ASCIZ /OR E49(5) BAD/
8616	067600	024071	024465	041040	
8617	067606	042101	000		
8618	067611	111	046114	043505	EM420: .ASCIZ /ILLEGAL HALT BIT IN CPU ERROR REG DIDN'T SET/
8619	067616	046101	044040	046101	

8620	067624	020124	044502	020124	
8621	067632	047111	041440	052520	
8622	067640	042440	051122	051117	
8623	067646	051040	043505	042040	
8624	067654	042111	023516	020124	
8625	067662	042523	000124		
8626	067666	051523	040522	050040	EM421: .ASCII /SSRA PS RESTORE(1) DOESN'T GET TO RACK E63/<CRLF>
8627	067674	020123	042522	052123	
8628	067702	051117	024105	024461	
8629	067710	042040	042517	047123	
8630	067716	052047	043440	052105	
8631	067724	052040	020117	040522	
8632	067732	045503	042440	031466	
8633	067740	200			
8634	067741	117	020122	033105	.ASCIZ /OR E63(5) BAD/
8635	067746	024063	024465	041040	
8636	067754	042101	000		
8637	067757	125	041502	020102	EM422: .ASCII /UBCB PE ABORT DOESN'T GO LOW OR/<CRLF>
8638	067764	042520	040440	047502	
8639	067772	052122	042040	042517	
8640	070000	047123	052047	043440	
8641	070006	020117	047514	020127	
8642	070014	051117	200		
8643	070017	111	020124	047504	.ASCII /IT DOESN'T GET TO TMCC E33 OR E33 BAD/<CRLF>
8644	070024	051505	023516	020124	
8645	070032	042507	020124	047524	
8646	070040	052040	041515	020103	
8647	070046	031505	020063	051117	
8648	070054	042440	031463	041040	
8649	070062	042101	200		
8650	070065	117	020122	041125	.ASCII /OR UBCB PARITY ERR DOESN'T GET TO TMCB E53/<CRLF>
8651	070072	041103	050040	051101	
8652	070100	052111	020131	051105	
8653	070106	020122	047504	051505	
8654	070114	023516	020124	042507	
8655	070122	020124	047524	052040	
8656	070130	041515	020102	032505	
8657	070136	100063			
8658	070140	051501	040440	046040	.ASCIZ /AS A LOW OR E53(5) BAD/
8659	070146	053517	047440	020122	
8660	070154	032505	024063	024465	
8661	070162	041040	042101	000	
8662	070167	125	041502	020102	EM423: .ASCII /UBCB PARITY ERR DOESN'T GO LOW OR IT DOES/<CRLF>
8663	070174	040520	044522	054524	
8664	070202	042440	051122	042040	
8665	070210	042517	047123	052047	
8666	070216	043440	020117	047514	
8667	070224	020127	051117	044440	
8668	070232	020124	047504	051505	
8669	070240	200			
8670	070241	116	052117	043440	.ASCIZ /NOT GET TO DAPE/
8671	070246	052105	052040	020117	
8672	070254	040504	042520	000	
8673	070261	104	050101	020105	EM424: .ASCIZ /DAPE E11(4) BAD OR TV06 DOESN'T GET TO THE ALU/
8674	070266	030505	024061	024464	
8675	070274	041040	042101	047440	

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

8732	071000	040520	020122	051124	
8733	071006	000120			
8734	071010	046524	041503	050040	EM435: .ASCII /TMCC PRIORITY CLEAR DIDN'T GO LOW OR DIDN'T/<CRLF>
8735	071016	044522	051117	052111	
8736	071024	020131	046103	040505	
8737	071032	020122	044504	047104	
8738	071040	052047	043440	020117	
8739	071046	047514	020127	051117	
8740	071054	042040	042111	023516	
8741	071062	100124			
8742	071064	042507	020124	044124	.ASCIZ /GET THRU TMCA E43(2) ON ABORT CLEAR/
8743	071072	052522	052040	041515	
8744	071100	020101	032105	024063	
8745	071106	024462	047440	020116	
8746	071114	041101	051117	020124	
8747	071122	046103	040505	000122	
8748	071130	052502	020123	041120	EM436: .ASCIZ /BUS PB DIDN'T GET TO UBCB PE ABORT/
8749	071136	042040	042111	023516	
8750	071144	020124	042507	020124	
8751	071152	047524	052440	041502	
8752	071160	020102	042520	040440	
8753	071166	047502	052122	000	
8754	071173	125	041502	020102	EM437: .ASCIZ /UBCB PARITY ERR DIDN'T GO LOW ON BUS PB/
8755	071200	040520	044522	054524	
8756	071206	042440	051122	042040	
8757	071214	042111	023516	020124	
8758	071222	047507	046040	053517	
8759	071230	047440	020116	052502	
8760	071236	020123	041120	000	
8761	071243	127	044501	020124	EM440: .ASCIZ /WAIT INSTRUCTION DIDN'T WAIT FOR INTERRUPT/
8762	071250	047111	052123	052522	
8763	071256	052103	047511	020116	
8764	071264	044504	047104	052047	
8765	071272	053440	044501	020124	
8766	071300	047506	020122	047111	
8767	071306	042524	051122	050125	
8768	071314	000124			
8769	071316	040527	052111	044440	EM441: .ASCIZ /WAIT INSTRUCTION FELL THRU ON T BIT TRAP/
8770	071324	051516	051124	041525	
8771	071332	044524	047117	043040	
8772	071340	046105	020114	044124	
8773	071346	052522	047440	020116	
8774	071354	020124	044502	020124	
8775	071362	051124	050101	000	
8776	071367	125	041502	020103	EM442: .ASCIZ /UBCC (PWR+INTR) DOESN'T GO LOW ON TMCB PIRQ/
8777	071374	050050	051127	025506	
8778	071402	047111	051124	020051	
8779	071410	047504	051505	023516	
8780	071416	020124	047507	046040	
8781	071424	053517	047440	020116	
8782	071432	046524	041103	050040	
8783	071440	051111	000121		
8784	071444	030523	027063	030060	EM443: .ASCIZ /S13.00 FAILED/
8785	071452	043040	044501	042514	
8786	071460	000104			
8787	071462	032123	027065	030060	EM444: .ASCIZ /S45.00 FAILED/

8788	071470	043040	044501	042514	
8789	071476	000104			
8790	071500	052115	027120	030061	EM445: .ASCIZ /MTP.10 FAILED/
8791	071506	043040	044501	042514	
8792	071514	000104			
8793	071516	042516	027107	030071	EM446: .ASCIZ /NEG.90 FAILED/
8794	071524	043040	044501	042514	
8795	071532	000104			
8796	071534	032104	027065	030070	EM447: .ASCIZ /D45.80 FAILED/
8797	071542	043040	044501	042514	
8798	071550	000104			
8799	071552	032104	027065	030071	EM450: .ASCIZ /D45.90 FAILED/
8800	071560	043040	044501	042514	
8801	071566	000104			
8802	071570	032104	027065	030060	EM451: .ASCIZ /D45.00 FAILED/
8803	071576	043040	044501	042514	
8804	071604	000104			
8805	071606	032104	027065	030460	EM452: .ASCIZ /D45.01 FAILED/
8806	071614	043040	044501	042514	
8807	071622	000104			
8808	071624	047101	044440	046114	EM453: .ASCIZ /AN ILLEGAL INSTRUCTION FAILED TO TRAP/
8809	071632	043505	046101	044440	
8810	071640	051516	051124	041525	
8811	071646	044524	047117	043040	
8812	071654	044501	042514	020104	
8813	071662	047524	052040	040522	
8814	071670	000120			
8815	071672	046524	041103	042440	EM454: .ASCIZ /TMCB E53 DOESN'T GO HIGH OR TMCB PIRQ DOESN'T GO LOW/
8816	071700	031465	042040	042517	
8817	071706	047123	052047	043440	
8818	071714	020117	044510	044107	
8819	071722	047440	020122	046524	
8820	071730	041103	050040	051111	
8821	071736	020121	047504	051505	
8822	071744	023516	020124	047507	
8823	071752	046040	053517	000	
8824	071757	123	051123	020101	EM455: .ASCII /SSRA PS RESTORE IS STUCK HIGH OR IT IS NOT/<CRLF>
8825	071764	051520	051040	051505	
8826	071772	047524	042522	044440	
8827	072000	020123	052123	041525	
8828	072006	020113	044510	044107	
8829	072014	047440	020122	052111	
8830	072022	044440	020123	047516	
8831	072030	100124			
8832	072032	042507	052124	047111	.ASCIZ /GETTING THRU RACK E63(B0) AS A LOW/
8833	072040	020107	044124	052522	
8834	072046	051040	041501	020113	
8835	072054	033105	024063	030102	
8836	072062	020051	051501	040440	
8837	072070	046040	053517	000	
8838	072075	124	041515	020103	EM456: .ASCIZ /TMCC E5(11) DOESN'T GO HIGH ON DATI OR DATO/
8839	072102	032505	030450	024461	
8840	072110	042040	042517	047123	
8841	072116	052047	043440	020117	
8842	072124	044510	044107	047440	
8843	072132	020116	040504	044524	

8844	072140	047440	020122	040504	
8845	072146	047524	000		
8846	072151	120	053523	041040	EM457: .ASCIZ /PSW BIT 11 DOESN'T CLEAR/
8847	072156	052111	030440	020061	
8848	072164	047504	051505	023516	
8849	072172	020124	046103	040505	
8850	072200	000122			
8851	072202	051520	020127	044103	EM460: .ASCIZ /PSW CHANGED ON RESET IN KERNEL MODE/
8852	072210	047101	042507	020104	
8853	072216	047117	051040	051505	
8854	072224	052105	044440	020116	
8855	072232	042513	047122	046105	
8856	072240	046440	042117	000105	
8857	072246	051520	020127	044103	EM461: .ASCIZ /PSW CHANGES ON RESET IN SUPER MODE/
8858	072254	047101	042507	020123	
8859	072262	047117	051040	051505	
8860	072270	052105	044440	020116	
8861	072276	052523	042520	020122	
8862	072304	047515	042504	000	
8863	072311	115	046505	046440	EM703: .ASCIZ /MEM MGT TESTS DISABLED/<CRLF>
8864	072316	052107	052040	051505	
8865	072324	051524	042040	051511	
8866	072332	041101	042514	100104	
8867	072340	000			
8868	072341	103	041501	042510	EM704: .ASCIZ /CACHE TESTS DISABLED/<CRLF>
8869	072346	052040	051505	051524	
8870	072354	042040	051511	041101	
8871	072362	042514	100104	000	
8872	072367	125	044516	052502	EM706: .ASCIZ /UNIBUS PE TEST DISABLED/<CRLF>
8873	072374	020123	042520	052040	
8874	072402	051505	020124	044504	
8875	072410	040523	046102	042105	
8876	072416	000200			
8877	072420	047516	041040	020122	EM710: .ASCIZ /NO BR 5 DEVICE/<CRLF>
8878	072426	020065	042504	044526	
8879	072434	042503	000200		
8880	072440	047516	041040	020122	EM711: .ASCIZ /NO BR 6 DEVICE/<CRLF>
8881	072446	020066	042504	044526	
8882	072454	042503	000200		
8883	072460	051102	032040	052040	EM712: .ASCIZ /BR 4 TESTS DISABLED/<CRLF>
8884	072466	051505	051524	042040	
8885	072474	051511	041101	042514	
8886	072502	100104	000		
8887	072505	102	020122	020065	EM713: .ASCIZ /BR 5 TESTS DISABLED/<CRLF>
8888	072512	042524	052123	020123	
8889	072520	044504	040523	046102	
8890	072526	042105	000200		
8891	072532	051102	033040	052040	EM715: .ASCIZ /BR 6 TESTS DISABLED/<CRLF>
8892	072540	051505	051524	042040	
8893	072546	051511	041101	042514	
8894	072554	100104	000		
8895	072557	117	051120	052040	EM717: .ASCIZ /OPR TEST DISABLED/<CRLF>
8896	072564	051505	020124	044504	
8897	072572	040523	046102	042105	
8898	072600	000200			
8899	072602	047514	045517	040440	EM720: .ASCII /LOOK AT THE CONSOLE LIGHTS/<CRLF>

8900	072610	020124	044124	020105	
8901	072616	047503	051516	046117	
8902	072624	020105	044514	044107	
8903	072632	051524	200		
8904	072635	124	042510	042040	.ASCII /THE DATA LIGHTS SHOULD READ 166667/<CRLF>
8905	072642	052101	020101	044514	
8906	072650	044107	051524	051440	
8907	072656	047510	046125	020104	
8908	072664	042522	042101	030440	
8909	072672	033066	033066	100067	
8910	072700	044124	020105	042101	.ASCIIZ /THE ADDRESS LIGHTS SHOULD READ /
8911	072706	051104	051505	020123	
8912	072714	044514	044107	051524	
8913	072722	051440	047510	046125	
8914	072730	020104	042522	042101	
8915	072736	020040	000		
8916	072741	200	044103	047101	EM721: .ASCIIZ <CRLF>/CHANGE SWITCH 7 TO CONTINUE/<CRLF>
8917	072746	042507	051440	044527	
8918	072754	041524	020110	020067	
8919	072762	047524	041440	047117	
8920	072770	044524	052516	100105	
8921	072776	000			
8922	072777	200	054524	042520	EM722: .ASCIIZ <CRLF>/TYPE A CHARACTER TO CONTINUE/<CRLF>
8923	073004	040440	041440	040510	
8924	073012	040522	052103	051105	
8925	073020	052040	020117	047503	
8926	073026	052116	047111	042525	
8927	073034	000200			
8928	073036	047200	020117	040515	EM724: .ASCIIZ <CRLF>/NO MAP REGISTERS AVAILABLE FOR TEST 75/<CRLF>
8929	073044	020120	042522	044507	
8930	073052	052123	051105	020123	
8931	073060	053101	044501	040514	
8932	073066	046102	020105	047506	
8933	073074	020122	042524	052123	
8934	073102	033440	100065	000	
8935	073107	116	020117	020114	EM725: .ASCIIZ /NO L CLOCK/<CRLF>
8936	073114	046103	041517	100113	
8937	073122	000			
8938		073124			.EVEN
8939	073124	073124			ADRTAB: .WORD .
8940	073126	005476			TST1
8941	073130	005660			TST2
8942	073132	006154			TST3
8943	073134	006436			TST4
8944	073136	007200			TST5
8945	073140	007314			TST6
8946	073142	007374			TST7
8947	073144	007460			TST10
8948	073146	010302			TST11
8949	073150	010442			TST12
8950	073152	011004			TST13
8951	073154	011120			TST14
8952	073156	012600			TST15
8953	073160	012746			TST16
8954	073162	013120			TST17
8955	073164	013306			TST20

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

SUBTAB: \$EOP
.END

ADRTAB	073124	5912	5922	8939#															
BINARY	034574	5842*	5845*	5847	5853#														
BIT0 =	000001	149#	1974	2004	2406	5932													
BIT00 =	000001	139#	149																
BIT01 =	000002	138#	148																
BIT02 =	000004	137#	147																
BIT03 =	000010	136#	146																
BIT04 =	000020	135#	145																
BIT05 =	000040	134#	144																
BIT06 =	000100	133#	143																
BIT07 =	000200	132#	142																
BIT08 =	000400	131#	141	5621															
BIT09 =	001000	130#	140	5629	5695														
BIT1 =	000002	148#	3590	3601	3603	3866													
BIT10 =	002000	129#	2925	2969	4342	5678													
BIT11 =	004000	128#	2958	3003	4358	4723	4724	4726	4731	4740	4742	4754	4756	4768					
		4779	4785	4789	4795	4797	4799	4806	4809	4823	4860	4862	5636						
BIT12 =	010000	127#	2873	2991	3038	4376	4473	4504	4521	5012	5083	5514							
BIT13 =	020000	126#	3025	3080	4391	4541	4589	4606	5541	5569	5685								
BIT14 =	040000	125#	2881	3062	3115	4433	4556	4624	4654	4914	4916	4925	4928	4936					
		4948	4971	5031	5036	5042	5078	5101	5147	5149	5287								
BIT15 =	100000	124#	1710	1711	1770	1778	1821	1836	1846	1866	1869	1895	2002	2031					
		2045	2109	2180	2227	2488	2495	2615	2623	2625	2653	2701	2738	2890					
		3065	3102	3179	4448	4571	4639	4669	4938	4940	4950	4953	4986	4993					
		5105	5116	5128	5351														
BIT2 =	000004	147#	3725	3728	3860	3879													
BIT3 =	000010	146#	3527	3530	3830	3851	3869												
BIT4 =	000020	145#	1666	1701	1760	2807	2855	3137	3148	3173	3189	3192	3338	3367					
		3403	3436	3470	3548	3612	3649	4021	4189	4277	4411	4467	4498	5073					
		5266																	
BIT5 =	000040	144#																	
BIT6 =	000100	143#	1559	1652	1658	1659	1677	1679	1697	3292	3619	3622	3625	3954					
		3968	4007	4691	4697	5321	5335	5448	5453	5897									
BIT7 =	000200	142#	5293	5874															
BIT8 =	000400	141#	4212																
BIT9 =	001000	140#	2862	2866	2937	3592	3883	4310	4326										
BPTVEC =	000014	156#	2771*																
BUFFDP	034200	5777#	5783																
CACHSP	033060	1562	5563#																
CACHVE	000114	163#	1562*	1563*															
CHAINQ	034702	5503	5882#																
CONTRL =	177746	173#	1495*	1568*															
CPUErr =	177766	186#	1564*	2905*	3148	3189	3191	3194*	3195	3254*	3264*	3268*	3527	3529					
		3533*	3534	3725	3727	3731*	3732	3860	3869	3895*	3947*	3954	3957*	3968					
		4007	4010*	4681*	5293	5295	5299*	5300	5547	5560*									
CPUSPU	032654	1560	1664	2784	2897	2903	2930	2963	2992	2996	3000	3027	3031	3034					
		3063	3072	3077	3104	3108	3111	3187	3253	3265	3267	3495	3504	3516					
		3662	3666	3672	3686	3697	3710	3720	3724	3871	3909	3938	3942	3963					
		3967	3983	4000	4006	4351	4347	4363	4526	4533	4611	4680	5303	5535#					
CR	000015	61#	6026	6036															
CRLF =	000200	62#	1541	5999	6036	6252	6258	6266	6277	6356	6372	6407	6438	6486					
		6529	6545	6561	6593	6620	6679	6772	6839	6996	7020	7040	7065	7089					
		7103	7110	7118	7258	7284	7302	7332	7341	7350	7371	7383	7396	7414					
		7425	7437	7450	7474	7496	7507	7521	7539	7557	7607	7647	7655	7669					
		7673	7689	7730	7737	7746	7753	7760	7767	7790	7802	7834	7842	7851					
		7860	7898	7914	7948	7985	8007	8020	8035	8052	8080	8120	8137	8150					

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

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

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

ITE222	003042	1013#
ITE223	003050	1016#
ITE224	003056	1019#
ITE225	003064	1022#
ITE226	003072	1025#
ITE227	003100	1028#
ITE230	003106	1031#
ITE231	003114	1034#
ITE232	003122	1037#
ITE233	003130	1040#
ITE234	003136	1043#
ITE235	003144	1046#
ITE236	003152	1049#
ITE237	003160	1052#
ITE240	003166	1055#
ITE241	003174	1058#
ITE242	003202	1061#
ITE243	003210	1064#
ITE244	003216	1068#
ITE245	003224	1071#
ITE246	003232	1074#
ITE247	003240	1077#
ITE250	003246	1080#
ITE251	003254	1083#
ITE252	003262	1086#
ITE253	003270	1089#
ITE254	003276	1092#
ITE255	003304	1095#
ITE256	003312	1098#
ITE257	003320	1101#
ITE260	003326	1104#
ITE261	003334	1107#
ITE262	003342	1110#
ITE263	003350	1113#
ITE264	003356	1116#
ITE265	003364	1119#
ITE266	003372	1122#
ITE267	003400	1125#
ITE270	003406	1128#
ITE271	003414	113 #
ITE272	003422	1134#
ITE273	003430	1137#
ITE274	003436	1140#
ITE275	003444	1143#
ITE276	003452	1146#
ITE277	003460	1149#
ITE300	003466	1152#
ITE301	003474	1155#
ITE302	003502	1158#
ITE303	003510	1161#
ITE304	003516	1164#
ITE305	003524	1167#
ITE306	003532	1170#
ITE307	003540	1173#
ITE310	003546	1176#
ITE311	003554	1179#

PC
CE
\$1
\$2
\$3
\$C
.

\$BDADR	001122	493#																	
\$BDDAT	001126	495#																	
\$BELL	001176	516#	5680	5703															
\$CHARC	035556	6012*	6019	6028*	6033#														
\$CLR.T	032552	5498	5501#																
\$CMTAG	001100	481#	1500	1501	1508	1514	1515	1516											
\$CM1 =	000003	507#	508#	509#	510#														
\$CM2 =	000006	507#	508#	509#	510#														
\$CM3 =	000003	505#	507																
\$CM4 =	000004	510#	511#	512#	513#	514#													
\$CRLF	001203	518#	5493	5688	5703	5713	5730	5735	5738	5769	5799	5828	5961	6002					
		6036																	
\$DBLK	036224	6139	6173	618#															
\$DOAGN	032602	5475	5496	5505	5511#														
\$DTBL	036214	6142	6177#																
\$ENDAD	032572	468	1536	5507#	5692														
\$ENDBR	016644	3297	3299	3311	3313#														
\$ENDCT	032410	1514	5477#																
\$ENULL	032650	5526#																	
\$EOP	032354	5408	5414	5416	5467#	8992													
\$EOPCT	032402	1514*	5474#	5478															
\$EPIRQ	001216	524#	2832*	2940*	2972*	3007*	3042*	3084*	3118*	7049	7174								
\$ERFLG	001103	484#	5597	5625	5627	5633*	5654	5672*	5703										
\$ERMAX	001115	490#	1517*	5627	5649*	5654													
\$ERPSW	001206	520#	1600*	1601*	1602	1656*	1669	1671	1789*	1790*	1827*	1828*	1829	1842*					
		1843*	1844	1865*	1871*	1872	1979*	1980*	1981	2008*	2009*	2014	2036*	2038*					
		2039	2068*	2070*	2080*	2081*	2113*	2162*	2186*	2187	2200*	2213*	2214	2334*					
		2335*	2336	2338	2351*	2356*	2357	2406	2431*	2449*	2450	2464*	2469*	2470					
		2494*	2499*	2500	2588*	2589*	2590	2621*	2630*	2631	2706*	2713*	2714	2743*					
		2747*	2748	4187*	4189	4210*	4212	4885*	4886*	4892*	4893*	4899*	4900*	5150*					
		5151*	5152	5183*	5198*	5212*	5226*	5238*	5251*	5258*	5259	6292	6554	6638					
\$ERROR	033536	1508	3984	3993	5670#														
\$ERRPC	001116	491#	5535*	5549	5563*	5577	5682*	5683*	5684	5703	5756	6292	6313	6366					
		6447	6539	6554	6575	6603	6638	7049	7174	7484	7494	7554	8207						
		567#	5725																
\$ERRTB	001274	488#	5490	5494*	5681*	5703													
\$ERTTL	001112	515#	1516*	1593*	1638*	1706*	1889*	1922*	1943*	2100*	2244*	2610*	2689*	2731*					
\$ESCAF	001174	2769*	2802*	2848*	3211*	3353*	3387*	3420*	3749*	3826*	3832	4001*	4169*	4721*					
		4737*	4820*	4968*	5066*	5561*	5589*	5648*	5698	5700	5703								
		503#	6005	6036															
\$FILLC	001150	502#	6036																
\$FILLS	001147	492#																	
\$GDADR	001120	494#																	
\$GDDAT	001124	5495#																	
\$GET42	032534	1#	16	17															
\$HD =	000001	485#	1637*	3325*	3828*	4271*	5315*	5640*	5641	5643*	5653								
\$ICNT	001104	6217	6246#																
\$ILLUP	036410	489#	5684*	5703	5715	5716													
\$ITEMB	001114	3250	3289#																
\$KW11L	016530	3261	3302#																
\$KW11P	016574	519#	5703	6036															
\$LF	001204	5519	5523#																
\$LOOP	032642	486#	1530*	1641*	1698*	1758*	2691*	2805*	3326*	3467*	3547*	3647*	3829*	4020*					
\$LPADR	001106	4272*	5070*	5316*	5631*	5646*	5651	5653											
		487#	1531*	1642*	1699*	1759*	1795*	1820*	1835*	1857*	2001*	2028*	2044*	2059*					
\$LPERR	001110	2074*	2193*	2220*	2346*	2413*	2423*	2456*	2476*	2487*	2506*	2524*	2543*	2562*					

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

SSGET4= 000001	5497#	5504#											
SSTRP = 000002	6197#	6208	6209	6210	6211	6212							
SSTSTN 001210	521#	5548*	5553	5576*	5581	5671*	5709	5711	5763	6292	6313	6366	6447
	6539	6554	6575	6603	6638	7049	7174	7484	7494	7554	8207		
SOFILL 036005	6063*	6067*	6077	6112#									
= 073300	437#	441	443#	466	467#	469#	471#	479#	520	1504	1530	1531	1541#
	5482#	5525	5527#	5540#	5546#	5568#	5574#	5653	5654	5703	5835#	5864#	5881#
	5894#	5952#	6036	6181#	6228	6247	6268#	6291#	6312#	6538#	6553#	6602#	7048#
	7483#	7493#	7718#	8206#	8938#	8939							

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

.\$SUPR	1#			
.\$STRAP	1#	4#	5591#	6182
.\$TYPB	1#			
.\$TYPD	1#	4#	5591#	6114
.\$TYPE	1#	4#	5963	
.\$TYPO	1#	4#	5591#	6036
.\$170	1#	4#	41	

. ABS. 073300 000

ERRORS DETECTED: 0

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