

PDP11-70

11/70 CPU #2
CEKBBF0

AH-7968F-MC
FICHE 1 OF 2

SEP 1980
COPYRIGHT © 75, 80
MADE IN USA



PDP11-70

11/70 CPU #2
CEKBBF0

AH-7968F-MC
FICHE 2 OF 2

SEP 1980
COPYRIGHT © 75, 80
MADE IN USA



.REM !

IDENTIFICATION

PRODUCT CODE: AC-7966F-MC
PRODUCT NAME: CEKBBFO 11/70 CPU #2
DATE : MAY, 1980
MAINTAINER: DIAGNOSTIC ENGINEERING

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

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

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

COPYRIGHT (C) 1975, 1980 BY DIGITAL EQUIPMENT CORPORATION

CONTENTS

1. ABSTRACT
2. REQUIREMENTS
 - 2.1 EQUIPMENT
 - 2.2 STORAGE
 - 2.3 PRELIMINARY PROGRAMS
3. LOADING PROCEDURE
 - 3.1 METHOD
4. STARTING PROCEDURE
 - 4.1 CONTROL SWITCH SETTINGS
 - 4.2 STARTING ADDRESS
 - 4.3 PROGRAM AND OPERATOR ACTION
5. OPERATING PROCEDURE
 - 5.1 OPERATIONAL SWITCH SETTINGS
 - 5.2 SUBROUTINE ABSTRACTS
 - 5.3 OPERATOR ACTION
6. ERRORS
 - 6.1 ERROR HALTS AND DESCRIPTION
 - 6.2 ERROR RECOVERY
7. RESTRICTIONS
 - 7.1 STARTING RESTRICTIONS
 - 7.2 OPERATING RESTRICTIONS
8. MISCELLANEOUS
 - 8.1 EXECUTION TIME
 - 8.2 STACK POINTER
 - 8.3 PASS COUNT
 - 8.4 ITERATIONS
 - 8.5 SPECIAL REGISTERS
 - 8.6 T BIT TRAPPING
 - 8.7 OSCILLOSCOPE SYNC POINTS
 - 8.8 CACHE CONTROL
9. PROGRAM DESCRIPTION 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 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 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 CPU WITH OPERATORS CONSOLE
LA30 OR EQUIVALENT TERMINAL
NOTE: THIS DIAGNOSTIC SUPPORTS THE PDP-11/74, AN EXPERIMENTAL, IN-HOUSE PROCESSOR.

2.2 STORAGE

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

2.3 PRELIMINARY PROGRAMS

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

CEKBB REQUIRES THAT CEKBA RUN

3. LOADING PROCEDURE

3.1 METHOD

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

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE 5.1

4.2 STARTING ADDRESS

200

4.3 PROGRAM AND OPERATOR ACTION

A. CEKBA

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

B. CEKBB

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

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

A. CEKBA

NONE

B. CEKBB

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

THE SWITCH SETTINGS ARE:

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

5.2 SUBROUTINE ABSTRACTS

A. CEKBA

SEE 5.2.4 AND 5.2.5

B. CEKBB

5.2.1 SPURIOUS ERROR HANDLER

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

5.2.2 SCOPE

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

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. RESTRICTIONS

7.1 STARTING RESTRICTIONS

A. CEKBA

NONE

B. CEKBB

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

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

7.2 OPERATING RESTRICTIONS

A. CEKBA

NONE

B. CEKBB

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

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

8. MISCELLANEOUS

8.1 EXECUTION TIME

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

8.7 OSCILLOSCOPE SYNC POINTS

A. CEKBA

BEGINNING WITH TEST 24 EACH TEST HAS AN OSCILLOSCOPE SYNC

INSTRUCTION. THE ADDRESS OF THE CONDITION CODE ROM STATE (44) IS IN THE PROCESSOR MICROBREAK REGISTER (ADDRESS 17777770). THIS WILL CAUSE PIN AE1 (SLOT 10) ON THE BACKPLANE TO GO HIGH EACH TIME A CONDITION CODE (OR NOP) INSTRUCTION IS EXECUTED. THEREFORE, IF THE OSCILLOSCOPE EXTERNAL SYNC IS CONNECTED TO THIS PIN AND THE SYNC SELECT PUT ON EXTERNAL THE OSCILLOSCOPE WILL BE SYNCHRONIZED WITH THE INSTRUCTION IMMEDIATELY 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 CACH 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) .
- CEKBBE - CHGE1 AND CHGE2 ADDED TO ELIMINATE TTY INTERRUPTS FROM OCCURRING DURING PRIORITY ARBITRATION LOGIC TESTS. (TEST 47)
- CEKBBF - DOCUMENTATION CHANGES ONLY.

DOCUMENT

CEKBBFO 11/70 CPU #2

COPYRIGHT 1975,1980
DIGITAL EQUIPMENT CORPORATION
MAYNARD, MASS. 01754

TABLE OF CONTENTS

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

TABLE OF CONTENTS

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

17 COPYRIGHT (C) 1975,1980
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 REGISTERS
227 USER 'I' PAGE ADDRESS REGISTERS
238 USER 'D' PAGE ADDRESS REGISTERS

249 SUPERVISOR 'I' PAGE DESCRIPTOR REGISTERS
260 SUPERVISOR 'D' PAGE DESCRIPTOR REGISTERS
271 SUPERVISOR 'I' PAGE ADDRESS REGISTERS
282 SUPERVISOR 'D' PAGE ADDRESS REGISTERS
293 KERNEL 'I' PAGE DESCRIPTOR REGISTERS
304 KERNEL 'D' PAGE DESCRIPTOR REGISTERS
315 KERNEL 'I' PAGE ADDRESS REGISTERS
326 KERNEL 'D' PAGE ADDRESS REGISTERS

340 *****
UNIBUS MAP REGISTER DEFINITIONS

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

434 *****
TRAP CATCHER

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

441 *****
STARTING ADDRESS(ES)

447 *****
ACT11 HOOKS

449 THE FOLLOWING LOCATIONS ARE SETUP TO BE USED WITH ACT11
LOCATION 46 WILL CONTAIN THE ADDRESS OF THE LOGICAL
END OF THE PROGRAM.
LOCATION 52 IS USED TO SPECIFY PROGRAM OPERATING REQUIREMENTS
AND/OR RESTRICTIONS. THIS IS ACCOMPLISHED BY SETTING VARIOUS BITS

TO A ONE OR A ZERO. THE BITS USED AND THERE MEANING ARE:

BIT 15=1 PROGRAM SHOULD BE POWER FAILED WHILE RUNNING
=0 NO POWER FAIL DESIRED

BIT 14=1 PROGRAM RUN TIME IS MEMORY SIZE DEPENDENT
=0 RUN TIME IS NOT MEMORY SIZE DEPENDENT

BITS 13-0 MUST BE ZERO'S

473 *****
COMMON TAGS

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

547 *****
ERROR POINTER TABLE

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

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

1560 TEST 1 SPL

IF FORK A FAILS EXECUTION WILL GO TO ONE OF 3 STATES:
RSD.02,SVC.70, OR D30.00.
RSD.02 WILL CAUSE A TRAP TO LOCATION 10.
SVC.70 WILL HANG THE PROCESSOR IN THE PAUSE STATE. THIS
WILL ONLY OCCUR IF RACF E17(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 MULTITPLIED 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 MULTIPLICAND WILL BE MULTIPLIED BY ZERO.
IF GRAJ SC=0 IS NOT GETTING TO RACK E50(C1) AND RACK E50
IS BAD (INPUT C1 FAILED LOW) THE MULTIPLICAND WILL ONLY
BE MULTIPLIED BY BIT 0 OF THE MULTIPLIER.
IF RACK E50(C1) FAILS HIGH THE PROCESSOR WILL HANG UP IN
STATE MUL.20(266).
IF THE INSTRUCTION FAILS TO MULTIPLY CORRECTLY AND ONE
OF THE ABOVE CONDITIONS CANNOT BE DETERMINED THEN THE
FAILURE COULD BE IN THE INSTRUCTION DECODE ROM.

IF THE CC'S COME UP BAD THEN THE FAILURE COULD BE IN
STATES MUL.40,50, OR 60, OR IN THE CONDITION CODE ROM.
ROM FLOW-50,102,266/246,226/206,310

2222 TEST 13 MUL*DM1

FORK A SHOULD NOT FAIL.
FORK B WILL FAIL TO RSD.00 IF THE R(MUL:ASHC*MFP) FIELD OF
THE INSTRUCTION DECODE ROM IS BAD.

IF STATE MUL.00 FAILS THE RESULT WILL BE BAD.
ROM FLOW-1,175,60,102,266/246,226/206,310

2254 TEST 14 DIV*DM0

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 (Z2(0)*LEFT SAVE (1)) IS BAD.

SECTION 7

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

SECTION 8

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

SECTION 9

THE NEXT SECTION DIVIDES -5 BY 2 TO ENSURE THAT THE REMAINDER IS STORED AS A NEGATIVE NUMBER.

SECTION 10

THE NEXT SECTION DIVIDES -5 BY -2 TO TEST STATES DVC.20,DVC.40, & DVC.60

SECTION 11

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

SECTION 12

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

2584 TEST 15 MTP*DMO

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

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

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

ROM FLOW-45,151,146,205

2627 TEST 16 MTP*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)*DM0 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 :05*07 DOES NOT
GO HIGH OR DOES NOT GET THRU TO THE ALU A TRAP TO 4 WILL OCCUR.
IF THE INTERRUPT DOESN'T OCCUR AN ATTEMPT IS MADE TO ISOLATE THE
FAILURE.

2900 TEST 24 PIR LEVEL 2 INTERRUPT

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

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

2933 TEST 25 PIR LEVEL 3 INTERRUPT

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

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

2966 TEST 26 PIR LEVEL 4 INTERRUPT

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

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

3000 TEST 27 PIR LEVEL 5 INTERRUPT

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

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

3035 TEST 30 PIR LEVEL 6 INTERRUPT

IF BEN13 FAILS EXECUTION WILL GO TO BRK.20.

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

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

3077 TEST 31 PIR LEVEL 7 INTERRUPT

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

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

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

3112 TEST 32 UNIBUS TIMEOUT

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

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

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

A TEST IS THEN MADE TO ENSURE THAT TMCC PRIORITY CLEAR GOES LOW.

3188

PERIPHERAL DETERMINATOR & INTERRUPT ENABLE ROUTINES

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

3191 SUBROUTINE IN A LOCATION.

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

3306 TEST 33 BR LEVEL 4 INTERRUPT

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

3370 TEST 34 BR LEVEL 5 INTERRUPT

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

3403 TEST 35 BR LEVEL 6 INTERRUPT

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

3436 TEST 36 YELLOW ZONE TRAP

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

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

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

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

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

3527 TEST 37 ROM FIELD CHECK OF PC MANIPULATOR STATES

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

3618 TEST 40 RED ZONE TRAP

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

3623

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

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

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

3726 TEST 41 BIT TEST OF STACK LIMIT REGISTER

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

3731

IF SCCE SL ADRS DOES NOT GET TO TMCD OR IF TMCD E28 OR E14 IS BAD THE BR WILL BE SELECTED. THE PB REGISTER IS LOADED WITH 200 SO IF TMCD LO BYTE EN DOES NOT GO LOW AN ERROR WILL BE DETECTED.

3782 TEST 42 SL REGISTER COMPARATOR TEST 1

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 DAT1" CANNOT BE TESTED.
THIS SIGNAL COMES UP WHEN A TRAP VECTOR IS READ IN FROM THE BUS.

4002 TEST 44 ILLEGAL INSTRUCTIONS

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

4039 TEST 45 T BIT TRAP

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

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

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

4058 TEST 46 T BIT TRAP AND RTT

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

4174 TEST 47 PRIORITY ARBITRATION

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

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

THE FOLLOWING IS A TABLE OF CONTENTS OF THIS TEST:

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

4646 TEST 50 GPR SET 1 SELECT TEST

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

- 4649 IT THEN ENSURES THAT GRAC GDREG SET 1 AND GSREG SET 1 GOES HIGH FOR THE MUX SELECTS LL, LH, AND HL. MUX SELECT HH WILL BE TESTED IN SUPERVISOR MODE.
- 4748 TEST 51 REGISTER SET 1 STUCK BIT TEST
THIS TEST ENSURES THAT ALL BITS IN GPR'S R10 THRU R15 WORK
- 4807 TEST 52 PSW HIGH BYTE BIT TEST
THIS TEST ENSURES THAT THE PRESENT AND PREVIOUS MODE BITS OF THE PSW CAN BE SET AND CLEARED AND THAT THEY ARE NOT STUCK TOGETHER.
- 4837 TEST 53 SP SELECTION TEST IN SUPER AND USER MODE
THIS TEST ENSURES THAT THE CORRECT STACK POINTERS ARE SELECTED IN SUPERVISOR AND USER MODE
- 4895 TEST 54 SUPER AND USER SP BIT TEST
THIS TEST ENSURES THAT THE SUPERVISOR AND USER STACK POINTERS DON'T HAVE ANY STUCK BITS.
- 4933 TEST 55 MTP*DMO*DF6*PREVIOUS MODE(SUPER*USER)
THIS TEST ENSURES THAT THE CORRECT SP'S ARE SELECTED WHEN EXECUTING A MTP WITH DIFFERENT PREVIOUS MODE BITS SELECTED.
- 4991 TEST 56 MFP*DMO*DF6*PREVIOUS MODE SUPER
THE ONLY POSSIBLE WAY THIS TEST CAN FAIL IS THAT IRCC DMO (MFP+MTP) DOES NOT GO HIGH ON MFP. THIS WILL ONLY HAPPEN IF THE IR DECODE ROM HAS A BAD FIELD (R(MFP+MTP)).
- 5023 TEST 57 UPAD 7 IN USER MODE
THIS TEST ENSURES THAT A UPAD 7(OCCURS IN RTI) CAUSES THE USER STACK POINTER TO BE USED TO FETCH THE NEW PS AND PC.
- 5027
IF PDRD PS14(1) DOES NOT GET TO THE GSAM(ON GRAC) THE TEST WILL BLOW UP.
- 5074 TEST 60 SPL*SUPERVISOR MODE
THIS TEST ENSURES THAT SPL DOES NOT LOAD THE PSW IN SUPER+USER MODE.
- 5090 TEST 61 PSW CLOCKING TEST
THIS TEST ENSURES THAT ALL THE BITS IN THE PSW GET LOADED WHEN THE FOLLOWING SIGNALS ARE TRUE:
1) LOAD PS*KERNEL MODE, AND 2) LOAD PS*KERNEL DATI.

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

FOLLOWING IS A TABLE TO DESCRIBE THE PSW FOR EACH SECTION

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

5196 TEST 62 ILLEGAL HALT
 THIS TEST ENSURES THAT A HALT IN SUPER OR USER MODE WILL TRAP TO LOCATION 4.
 IF BEN6 FAILS EXECUTION WOULD GO TO FET.04 WHICH WOULD CAUSE THE HALT TO LOOK LIKE A NOP.
 IF TMCE SET CONF GOES LOW THE PROCESSOR WILL HALT AT LOCALOCATION 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)

6011 TEST 65 OPERATOR INTERVENTION TEST
 THIS TEST ENSURES THAT THE RESET AND WAIT FLOWS PUT RO AND THE PC IN THE LIGHTS. THE TEST IS ONLY EXECUTED ON PASS 1 AND CAN BE DISABLED ALTOGETHER WITH SWIICH 0.
 THE RESET LOOP IS STOPED BY CHANGING THE POSITION OF SWITCH 7.
 THE WAIT IS EXITED BY TYPING A CHARACTER ON THE TERMINAL.

6074 *****
 END OF PASS ROUTINE

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

6146 *****
SPURIOUS ERROR HANDLER

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

6211 *****
SCOPE HANDLER ROUTINE

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

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

6277 *****
ERROR HANDLER ROUTINE

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

6328 *****
ERROR MESSAGE TYPEOUT ROUTINE

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

6371 *****
STACK LIMIT TEST TYPE OUT ROUTINE

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

6489 *****
MONITOR RESTORE ROUTINE

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

6527 *****
CHECK TEST SEQUENCE ROUTINE

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

6588 *****
TYPE ROUTINE

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

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

CALL:
1) USING A TRAP INSTRUCTION
TYPE ,MESADR ;:MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
OR
TYPE
MESADR
2) USING A JSR INSTRUCTION
MOV PS,-(SP) ;:PUSH PROCESSOR STATUS WORD ON THE STACK
JSR PC,\$TYPE ;:CALL TYPE ROUTINE
MESADDR ;:FIRST ADRESS OF MESSAGE

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

6663 THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
OCTAL (ASCII) NUMBER AND TYPE IT.
\$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
CALL:

MOV NUM,-(SP) ;:NUMBER TO BE TYPED
TYPOS ;:CALL FOR TYPEOUT

.BYTE N ;:N=1 TO 6 FOR NUMBER OF DIGITS TO TYPE
.BYTE M ;:M=1 OR 0
;:1=TYPE LEADING ZEROS
;:0=SUPPRESS LEADING ZEROS

\$TYPON----ENTER HERE TO TYPE OUT WITH THE SAME PARAMETERS AS THE LAST
\$TYPOS OR \$TYPOC

CALL:
MOV NUM,-(SP) ;:NUMBER TO BE TYPED
TYPON ;:CALL FOR TYPEOUT

\$TYPOC---ENTER HERE FOR TYPEOUT OF A 16 BIT NUMBER

CALL:
MOV NUM,-(SP) ;:NUMBER TO BE TYPED
TYPOC ;:CALL FOR TYPEOUT

6739

CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

6741 THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
REPLACED WITH SPACES.

CALL:
MOV NUM,-(SP) ;:PUT THE BINARY NUMBER ON THE STACK
TYPDS ;:GO TO THE ROUTINE

6807

TRAP DECODER

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

6822

TRAP TABLE

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

6837

POWER DOWN AND UP ROUTINES

1696
1697
1698
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1710
1711

000001
177400

.TITLE CEKBBFO 11/70 CPU #2
:*COPYRIGHT (C) 1975,1980
:*DIGITAL EQUIPMENT CORP.
:*MAYNARD, MASS. 01754
:*
:*
:*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
:*PACKAGE (MAINDEC-11-DZQAC-A5).
:*
\$TN=1 \$SWR=177400

1712
1713
1714
1715
1716
1717
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1729
1730
1731
1732
1733
1734

.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

```

1735
1736          .SBTTL BASIC DEFINITIONS
1737
1738          ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
1739          001100 STACK= 1100          ;;FIRST ADDRESS OF THE STACK
1740          001100 KERSTK= STACK        ;;KERNEL STACK
1741          000700 SUPSTK= STACK-200    ;;SUPERVISOR STACK
1742          000600 USESTK= STACK-300    ;;USER STACK
1743          .EQUIV EMT,ERROR            ;;BASIC DEFINITION OF ERROR CALL
1744          .EQUIV IOT,SCOPE            ;;BASIC DEFINITION OF SCOPE CALL
1745          177776 PS= 177776          ;;PROCESSOR STATUS WORD
1746          .EQUIV PS,PSW
1747          177774 STKLMT= 177774       ;;STACK LIMIT REGISTER
1748          177772 PIRQ= 177772        ;;PROGRAM INTERRUPT REQUEST REGISTER
1749          177570 SWR= 177570         ;;SWITCH REGISTER
1750          177570 DISPLAY=SWR
1751
1752          ;*MISCELLANEOUS DEFINITIONS
1753          000011 HT= 11                ;;CODE FOR HORIZONTAL TAB
1754          000012 LF= 12                ;;CODE LINE FEED
1755          000015 CR= 15                ;;CODE CARRIAGE RETURN
1756          000200 CRLF= 200            ;;CODE FOR CARRIAGE RETURN-LINE FEED
1757
1758          ;*GENERAL PURPOSE REGISTER DEFINITIONS
1759          000000 R0= %0                ;;GENERAL REGISTER
1760          000001 R1= %1                ;;GENERAL REGISTER
1761          000002 R2= %2                ;;GENERAL REGISTER
1762          000003 R3= %3                ;;GENERAL REGISTER
1763          000004 R4= %4                ;;GENERAL REGISTER
1764          000005 R5= %5                ;;GENERAL REGISTER
1765          000006 R6= %6                ;;GENERAL REGISTER
1766          000007 R7= %7                ;;GENERAL REGISTER
1767          .EQUIV R0,R10              ;;GENERAL REGISTER
1768          .EQUIV R1,R11              ;;GENERAL REGISTER
1769          .EQUIV R2,R12              ;;GENERAL REGISTER
1770          .EQUIV R3,R13              ;;GENERAL REGISTER
1771          .EQUIV R4,R14              ;;GENERAL REGISTER
1772          .EQUIV R5,R15              ;;GENERAL REGISTER
1773          000006 SP=%6                ;;STACK POINTER
1774          .EQUIV SP,KSP              ;;KERNEL STACK POINTER
1775          .EQUIV SP,SSP              ;;SUPERVISOR STACK POINTER
1776          .EQUIV SP,USP              ;;USER STACK POINTER
1777          000007 PC=%7                ;;PROGRAM COUNTER
1778
1779          ;*PRIORITY LEVEL DEFINITIONS
1780          000000 PR0= 0                ;;PRIORITY LEVEL 0
1781          000040 PR1= 40                ;;PRIORITY LEVEL 1
1782          000100 PR2= 100              ;;PRIORITY LEVEL 2
1783          000140 PR3= 140              ;;PRIORITY LEVEL 3
1784          000200 PR4= 200              ;;PRIORITY LEVEL 4
1785          000240 PR5= 240              ;;PRIORITY LEVEL 5
1786          000300 PR6= 300              ;;PRIORITY LEVEL 6
1787          000340 PR7= 340              ;;PRIORITY LEVEL 7
1788
1789          ;*"SWITCH REGISTER" SWITCH DEFINITIONS
1790          100000 SW15= 100000
    
```

1791	040000	SW14=	40000
1792	020000	SW13=	20000
1793	010000	SW12=	10000
1794	004000	SW11=	4000
1795	002000	SW10=	2000
1796	001000	SW09=	1000
1797	000400	SW08=	400
1798	000200	SW07=	200
1799	000100	SW06=	100
1800	000040	SW05=	40
1801	000020	SW04=	20
1802	000010	SW03=	10
1803	000004	SW02=	4
1804	000002	SW01=	2
1805	000001	SW00=	1
1806		.EQUIV	SW09,SW9
1807		.EQUIV	SW08,SW8
1808		.EQUIV	SW07,SW7
1809		.EQUIV	SW06,SW6
1810		.EQUIV	SW05,SW5
1811		.EQUIV	SW04,SW4
1812		.EQUIV	SW03,SW3
1813		.EQUIV	SW02,SW2
1814		.EQUIV	SW01,SW1
1815		.EQUIV	SW00,SW0

;*DATA BIT DEFINITIONS (BIT00 TO BIT15)

1817			
1818	100000	BIT15=	100000
1819	040000	BIT14=	40000
1820	020000	BIT13=	20000
1821	010000	BIT12=	10000
1822	004000	BIT11=	4000
1823	002000	BIT10=	2000
1824	001000	BIT09=	1000
1825	000400	BIT08=	400
1826	000200	BIT07=	200
1827	000100	BIT06=	100
1828	000040	BIT05=	40
1829	000020	BIT04=	20
1830	000010	BIT03=	10
1831	000004	BIT02=	4
1832	000002	BIT01=	2
1833	000001	BIT00=	1
1834		.EQUIV	BIT09,BIT9
1835		.EQUIV	BIT08,BIT8
1836		.EQUIV	BIT07,BIT7
1837		.EQUIV	BIT06,BIT6
1838		.EQUIV	BIT05,BIT5
1839		.EQUIV	BIT04,BIT4
1840		.EQUIV	BIT03,BIT3
1841		.EQUIV	BIT02,BIT2
1842		.EQUIV	BIT01,BIT1
1843		.EQUIV	BIT00,BIT0

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

1846 000004

1847	000010	RESVEC= 10	::RESERVED AND ILLEGAL INSTRUCTIONS
1848	000014	TBITVEC=14	::"T" BIT
1849	000014	TRTVEC= 14	::TRACE TRAP
1850	000014	BPTVEC= 14	::BREAKPOINT TRAP (BPT)
1851	000020	IOTVEC= 20	::INPUT/OUTPUT TRAP (IOT) **SCOPE**
1852	000024	PWRVEC= 24	::POWER FAIL
1853	000030	EMTVEC= 30	::EMULATOR TRAP (EMT) **ERROR**
1854	000034	TRAPVEC=34	::"TRAP" TRAP
1855	000060	TKVEC= 60	::TTY KEYBOARD VECTOR
1856	000064	TPVEC= 64	::TTY PRINTER VECTOR
1857	000114	CACHVEC=114	::CACHE ERROR INTERRUPT VECTOR
1858	000240	PIRQVEC=240	::PROGRAM INTERRUPT REQUEST VECTOR
1859	000250	MMVEC= 250	::MEMORY MANAGEMENT VECTOR

.SBTTL CACHE REGISTER DEFINITIONS

1864	177740	LOADRS = 177740	::LOWER 16 BITS OF ADDRESS THAT CAUSED ERROR
1865	177742	HIADRS = 177742	::UPPER SIX BITS OF ADDRESS THAT CAUSED ERROR
1866	177744	MEMERR = 177744	::CACHE ERROR REGISTER
1867	177746	CONTRL = 177746	::MEMORY CONTROL REGISTER
1868	177750	MAINT = 177750	::MEMORY MAINTENENCE REGISTER
1869	177752	HITMIS = 177752	::HIT MISS REGISTER "1" IMPLIES HIT IN CACHE

.SBTTL CPU REGISTER DEFINITIONS

1875	177760	SIZELO = 177760	::MEMORY SIZE REGISTER NUMBER TO PUT INTO A PAR
1876			::TO GET TO THE LAST 32 WORDS OF MEMORY
1877	177762	SIZEHI = 177762	::HIGH SIZE REGISTER, RESERVED FOR FUTURE USE
1878			::CURRENTLY ALL ZERO
1879	177764	SYSTID = 177764	::SYSTEM ID REGISTER
1880	177766	CPUERR = 177766	::CPU ERROR REGISTER HOLDS CONDITION THAT CAUSED
1881			::THE TRAP TO ERRVEC (000004)

.SBTTL MEMORY MANAGEMENT DEFINITIONS

;*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES

1891	177572	MMR0= 177572
1892	177574	MMR1= 177574
1893	177576	MMR2= 177576
1894	172516	MMR3= 172516
1895		.EQUIV MMR0,SR0
1896		.EQUIV MMR1,SR1
1897		.EQUIV MMR2,SR2
1898		.EQUIV MMR3,SR3

;*USER "I" PAGE DESCRIPTOR REGISTERS

1902	177600	UIPDRO= 177600
------	--------	----------------

1903	177602	UIPDR1= 177602
1904	177604	UIPDR2= 177604
1905	177606	UIPDR3= 177606
1906	177610	UIPDR4= 177610
1907	177612	UIPDR5= 177612
1908	177614	UIPDR6= 177614
1909	177616	UIPDR7= 177616
1910		
1911		;*USER 'D' PAGE DESCRIPTOR REGISTORS
1912		
1913	177620	UDPDR0= 177620
1914	177622	UDPDR1= 177622
1915	177624	UDPDR2= 177624
1916	177626	UDPDR3= 177626
1917	177630	UDPDR4= 177630
1918	177632	UDPDR5= 177632
1919	177634	UDPDR6= 177634
1920	177636	UDPDR7= 177636
1921		
1922		;*USER 'I' PAGE ADDRESS REGISTERS
1923		
1924	177640	UIPAR0= 177640
1925	177642	UIPAR1= 177642
1926	177644	UIPAR2= 177644
1927	177646	UIPAR3= 177646
1928	177650	UIPAR4= 177650
1929	177652	UIPAR5= 177652
1930	177654	UIPAR6= 177654
1931	177656	UIPAR7= 177656
1932		
1933		;*USER 'D' PAGE ADDRESS REGISTERS
1934		
1935	177660	UDPAR0= 177660
1936	177662	UDPAR1= 177662
1937	177664	UDPAR2= 177664
1938	177666	UDPAR3= 177666
1939	177670	UDPAR4= 177670
1940	177672	UDPAR5= 177672
1941	177674	UDPAR6= 177674
1942	177676	UDPAR7= 177676
1943		
1944		;*SUPERVISOR 'I' PAGE DESCRIPTOR REGISTERS
1945		
1946	172200	SIPDR0= 172200
1947	172202	SIPDR1= 172202
1948	172204	SIPDR2= 172204
1949	172206	SIPDR3= 172206
1950	172210	SIPDR4= 172210
1951	172212	SIPDR5= 172212
1952	172214	SIPDR6= 172214
1953	172216	SIPDR7= 172216
1954		
1955		;*SUPERVISOR 'D' PAGE DESCRIPTOR REGISTERS
1956		
1957	172220	SDPDR0= 172220
1958	172222	SDPDR1= 172222

1959	172224	SDPDR2= 172224
1960	172226	SDPDR3= 172226
1961	172230	SDPDR4= 172230
1962	172232	SDPDR5= 172232
1963	172234	SDPDR6= 172234
1964	172236	SDPDR7= 172236
1965		
1966		;*SUPERVISOR 'I' PAGE ADDRESS REGISTERS
1967		
1968	172240	SIPAR0= 172240
1969	172242	SIPAR1= 172242
1970	172244	SIPAR2= 172244
1971	172246	SIPAR3= 172246
1972	172250	SIPAR4= 172250
1973	172252	SIPAR5= 172252
1974	172254	SIPAR6= 172254
1975	172256	SIPAR7= 172256
1976		
1977		;*SUPERVISOR 'D' PAGE ADDRESS REGISTERS
1978		
1979	172260	SDPAR0= 172260
1980	172262	SDPAR1= 172262
1981	172264	SDPAR2= 172264
1982	172266	SDPAR3= 172266
1983	172270	SDPAR4= 172270
1984	172272	SDPAR5= 172272
1985	172274	SDPAR6= 172274
1986	172276	SDPAR7= 172276
1987		
1988		;*KERNEL 'I' PAGE DESCRIPTOR REGISTERS
1989		
1990	172300	KIPDR0= 172300
1991	172302	KIPDR1= 172302
1992	172304	KIPDR2= 172304
1993	172306	KIPDR3= 172306
1994	172310	KIPDR4= 172310
1995	172312	KIPDR5= 172312
1996	172314	KIPDR6= 172314
1997	172316	KIPDR7= 172316
1998		
1999		;*KERNEL 'D' PAGE DESCRIPTOR REGISTERS
2000		
2001	172320	KDPDR0= 172320
2002	172322	KDPDR1= 172322
2003	172324	KDPDR2= 172324
2004	172326	KDPDR3= 172326
2005	172330	KDPDR4= 172330
2006	172332	KDPDR5= 172332
2007	172334	KDPDR6= 172334
2008	172336	KDPDR7= 172336
2009		
2010		;*KERNEL 'I' PAGE ADDRESS REGISTERS
2011		
2012	172340	KIPAR0= 172340
2013	172342	KIPAR1= 172342
2014	172344	KIPAR2= 172344

2015	172346	KIPAR3= 172346
2016	172350	KIPAR4= 172350
2017	172352	KIPAR5= 172352
2018	172354	KIPAR6= 172354
2019	172356	KIPAR7= 172356

2020
2021 ;*KERNEL 'D' PAGE ADDRESS REGISTERS

2022		
2023	172360	KDPAR0= 172360
2024	172362	KDPAR1= 172362
2025	172364	KDPAR2= 172364
2026	172366	KDPAR3= 172366
2027	172370	KDPAR4= 172370
2028	172372	KDPAR5= 172372
2029	172374	KDPAR6= 172374
2030	172376	KDPAR7= 172376

2031
2032
2033
2034
2035 .SBTTL UNIBUS MAP REGISTER DEFINITIONS

2036
2037
2038 ;*THE LOWER 16 BITS OF THE MAP REGISTERS ARE LABELED 'MAPLXX'
2039 ;*THE UPPER 6 BITS OF THE MAP REGISTERS ARE LABELED 'MAPHXX'

2040		
2041		
2042	170200	MAPL00 = 170200
2043	170202	MAPH00 = 170202
2044	170204	MAPL01 = 170204
2045	170206	MAPH01 = 170206
2046	170210	MAPL02 = 170210
2047	170212	MAPH02 = 170212
2048	170214	MAPL03 = 170214
2049	170216	MAPH03 = 170216
2050	170220	MAPL04 = 170220
2051	170222	MAPH04 = 170222
2052	170224	MAPL05 = 170224
2053	170226	MAPH05 = 170226
2054	170230	MAPL06 = 170230
2055	170232	MAPH06 = 170232
2056	170234	MAPL07 = 170234
2057	170236	MAPH07 = 170236
2058	170240	MAPL10 = 170240
2059	170242	MAPH10 = 170242
2060	170244	MAPL11 = 170244
2061	170246	MAPH11 = 170246
2062	170250	MAPL12 = 170250
2063	170252	MAPH12 = 170252
2064	170254	MAPL13 = 170254
2065	170256	MAPH13 = 170256
2066	170260	MAPL14 = 170260
2067	170262	MAPH14 = 170262
2068	170264	MAPL15 = 170264
2069	170266	MAPH15 = 170266
2070	170270	MAPL16 = 170270

2071	170272	MAPH16 = 170272
2072	170274	MAPL17 = 170274
2073	170276	MAPH17 = 170276
2074	170300	MAPL20 = 170300
2075	170302	MAPH20 = 170302
2076	170304	MAPL21 = 170304
2077	170306	MAPH21 = 170306
2078	170310	MAPL22 = 170310
2079	170312	MAPH22 = 170312
2080	170314	MAPL23 = 170314
2081	170316	MAPH23 = 170316
2082	170320	MAPL24 = 170320
2083	170320	MAPH24 = 170320
2084	170324	MAPL25 = 170324
2085	170326	MAPH25 = 170326
2086	170330	MAPL26 = 170330
2087	170332	MAPH26 = 170332
2088	170334	MAPL27 = 170334
2089	170336	MAPH27 = 170336
2090	170340	MAPL30 = 170340
2091	170342	MAPH30 = 170342
2092	170344	MAPL31 = 170344
2093	170346	MAPH31 = 170346
2094	170350	MAPL32 = 170350
2095	170352	MAPH32 = 170352
2096	170354	MAPL33 = 170354
2097	170356	MAPH33 = 170356
2098	170360	MAPL34 = 170360
2099	170362	MAPH34 = 170362
2100	170364	MAPL35 = 170364
2101	170366	MAPH35 = 170366
2102	170370	MAPL36 = 170370
2103	170372	MAPH36 = 170372
2104	170374	MAPL37 = 170374
2105	170376	MAPH37 = 170376
2106		.EQUIV MAPL00,MAPL0
2107		.EQUIV MAPH00,MAPH0
2108		.EQUIV MAPL01,MAPL1
2109		.EQUIV MAPH01,MAPH1
2110		.EQUIV MAPL02,MAPL2
2111		.EQUIV MAPH02,MAPH2
2112		.EQUIV MAPL03,MAPL3
2113		.EQUIV MAPH03,MAPH3
2114		.EQUIV MAPL04,MAPL4
2115		.EQUIV MAPH04,MAPH4
2116		.EQUIV MAPL05,MAPL5
2117		.EQUIV MAPH05,MAPH5
2118		.EQUIV MAPL06,MAPL6
2119		.EQUIV MAPH06,MAPH6
2120		.EQUIV MAPL07,MAPL7
2121		.EQUIV MAPH07,MAPH7
2122		
2123		
2124		
2125		
2126	172544	PLKC=172544

```

2127
2128
2129          .SBTTL TRAP CATCHER
2130
2131          000000          .=0
2132          ;*ALL UNUSED LOCATIONS FROM 4 - 776 CONTAIN A ".+2,HALT"
2133          ;*SEQUENCE TO CATCH ILLEGAL TRAPS AND INTERRUPTS
2134          ;*LOCATION 0 CONTAINS 0 TO CATCH IMPROPERLY LOADED VECTORS
2135
2136          .SBTTL STARTING ADDRESS(ES)
2137          000200          .=200
2138
2139 000200 000137 004742          JMP @#START          ;;JUMP TO STARTING ADDRESS OF PROGRAM
2140          ;;*****
2141
2142          .SBTTL          ACT11 HOOKS
2143
2144          ;*THE FOLLOWING LOCATIONS ARE SETUP TO BE USED WITH ACT11
2145          ;*
2146          ;*LOCATION 46 WILL CONTAIN THE ADDRESS OF THE LOGICAL
2147          ;*END OF THE PROGRAM.
2148          ;*LOCATION 52 IS USED TO SPECIFY PROGRAM OPERATING REQUIREMENTS
2149          ;*AND/OR RESTRICTIONS. THIS IS ACCOMPLISHED BY SETTING VARIOUS BITS
2150          ;*TO A ONE OR A ZERO. THE BITS USED AND THERE MEANING ARE:
2151          ;*
2152          ;*          BIT 15=1 PROGRAM SHOULD BE POWER FAILED WHILE RUNNING
2153          ;*          =0 NO POWER FAIL DESIRED
2154          ;*
2155          ;*          BIT 14=1 PROGRAM RUN TIME IS MEMORY SIZE DEPENDENT
2156          ;*          =0 RUN TIME IS NOT MEMORY SIZE DEPENDENT
2157          ;*
2158          ;*          BITS 13-0 MUST BE ZERO'S
2159
2160          000204          $SVPC=.          ;;SAVE LOCATION COUNTER
2161          000046          .=46          ;;SET LOCATION COUNTER
2162 000046 032554          .WORD $ENDAD          ;;SET LOC.46 TO ADDRESS $ENDAD
2163          000052          .=52          ;;SET LOCATION COUNTER
2164 000052 000000          .WORD 0          ;;SET LOC.52 TO ZERO
2165          000204          .= $SVPC          ;; RESTORE LOCATION COUNTER
    
```

```

2166 ;:*****
2167
2168 .SBTTL COMMON TAGS
2169
2170 ;*THIS TABLE CONTAINS VARIOUS COMMON STORAGE LOCATIONS
2171 ;*USED IN THE PROGRAM.
2172
2173         001100         .=1100
2174
2175 001100 $CMTAG:           ;; START OF COMMON TAGS
2176 001100 000000 $PASS: .WORD 0           ;; CONTAINS PASS COUNT
2177 001102 000 $TSTNM: .BYTE 0           ;; CONTAINS THE TEST NUMBER
2178 001103 000 $ERFLG: .BYTE 0           ;; CONTAINS ERROR FLAG
2179 001104 000000 $ICNT: .WORD 0           ;; CONTAINS SUBTEST ITERATION COUNT
2180 001106 000000 $LPADR: .WORD 0           ;; CONTAINS SCOPE LOOP
2181 001110 000000 $LPERR: .WORD 0           ;; CONTAINS SCOPE RETURN FOR ERRORS
2182 001112 000000 $ERTTL: .WORD 0           ;; CONTAINS TOTAL ERRORS DETECTED
2183 001114 000 $ITEMB: .BYTE 0           ;; CONTAINS ITEM CONTROL BYTE
2184 001115 001 $ERMAX: .BYTE 1           ;; CONTAINS MAX. ERRORS PER TEST
2185 001116 000000 $ERRPC: .WORD 0           ;; CONTAINS PC OF LAST ERROR INSTRUCTION
2186 001120 000000 $GDADR: .WORD 0           ;; CONTAINS OF 'GOOD' DATA
2187 001122 000000 $BDADR: .WORD 0           ;; CONTAINS OF 'BAD' DATA
2188 001124 000000 $GDDAT: .WORD 0           ;; CONTAINS 'GOOD' DATA
2189 001126 000000 $BDDAT: .WORD 0           ;; CONTAINS 'BAD' DATA
2190 001130 000000 000000 000000 $TKS: .WORD 0,0,0           ;; RESERVED--NOT TO BE USED
2191 001136 177560 $TKB: 177560           ;; TTY KBD STATUS
2192 001140 177562 $TPS: 177564           ;; TTY KBD BUFFER
2193 001142 177564 $TPB: 177566           ;; TTY PRINTER STATUS REG.
2194 001144 177566 $NULL: .BYTE 0           ;; TTY PRINTER BUFFER REG.
2195 001146 000 $FILLS: .BYTE 2           ;; CONTAINS NULL CHARACTER FOR FILLS
2196 001147 002 $FILLC: .BYTE 12          ;; CONTAINS # OF FILLER CHARACTERS REQUIRED
2197 001150 012 $TPFLG: .BYTE 0           ;; INSERT FILL CHARS. AFTER A 'LINE FEED'
2198 001151 000 $REGAD: .WORD 0           ;; "TERMINAL AVAILABLE" FLAG (BIT<07>=0=YES)
2199 001152 000000
2200
2201 001154 000000 $REG0: .WORD 0           ;; CONTAINS THE FROM
2202 001156 000000 $REG1: .WORD 0           ;; WHICH ($REG0) WAS OBTAINED
2203 001160 000000 $REG2: .WORD 0           ;; CONTAINS (($REGAD)+0)
2204 001162 000000 $TMP0: .WORD 0           ;; CONTAINS (($REGAD)+2)
2205 001164 000000 $TMP1: .WORD 0           ;; CONTAINS (($REGAD)+4)
2206 001166 000000 $TMP2: .WORD 0           ;; USER DEFINED
2207 001170 000000 $TMP3: .WORD 0           ;; USER DEFINED
2208 001172 000000 $TIMES: 0           ;; USER DEFINED
2209 001174 000000 $ESCAPE: 0           ;; USER DEFINED
2210 001176 177607 000377 $BELL: .ASCIZ <207><377><377> ;; MAX. NUMBER OF ITERATIONS
2211 001202 077 $QUES: .ASCII /?/           ;; ESCAPE ON ERROR
2212 001203 015 $CRLF: .ASCII <15>           ;; CODE FOR BELL
2213 001204 000012 $LF: .ASCIZ <12>           ;; QUESTION MARK
2214 001206 000000 $ERPSW: .WORD 0           ;; CARRIAGE RETURN
2215 001210 000000 $$TSTNM: .WORD 0           ;; LINE FEED
2216 001212 000100 $PR2: .WORD PR2           ;; ERROR PSW
2217 001214 000240 $PR5: .WORD PR5           ;; TEST NUMBER STORAGE
2218 001216 000000 $EPIRQ: .WORD 0           ;; PRIORITY LEVEL 2
2219 001220 000000 E1STKLM: .WORD 0           ;; PRIORITY LEVEL 5
2220 001222 000000 E2STKLM: .WORD 0           ;; ERROR PIRQ
2221 001224 000000 INTER5: .WORD 0           ;; STACK LIMIT REGISTER ERROR 1
                                           ;; STACK LIMIT REGISTER ERROR 2
                                           ;; ADDRESS OF BR5 INTER SUBROUTINE
    
```

2222	001226	000000	INT5VEC: .WORD	0	:BR 5 INTERRUPT VECTOR
2223	001230	000000	INT5ST: .WORD	0	:BR 5 STATUS REG
2224	001232	000000	INTER6: .WORD	0	:ADDRESS OF BR6 INTERRUPT SUBROUTINE
2225	001234	000000	INT6VEC: .WORD	0	:BR 6 INTERRUPT VECTOR
2226	001236	000000	INT6ST: .WORD	0	:BR 6 STATUS REG
2227	001240	172040	RSCS1: .WORD	172040	:ADDRESS OF RS STATUS REGISTER
2228	001242	000204	RSVEC: .WORD	204	:ADDRESS OF RS VECTOR
2229	001244	176700	RPCS1: .WORD	176700	:ADDRESS OF RP STATUS REGISTER
2230	001246	000254	RPVEC: .WORD	254	:ADDRESS OF RP VECTOR
2231	001250	177404	RKCS1: .WORD	177404	:ADDRESS OF RK STATUS REGISTER
2232	001252	000220	RKVEC: .WORD	220	:ADDRESS OF RK VECTOR
2233	001254	172440	TMCS1: .WORD	172440	:ADDRESS OF TM STATUS REG
2234	001256	000224	TMVEC: .WORD	224	:ADDRESS OF TM VECTOR
2235	001260	177546	LKSTAT: .WORD	177546	:ADDRESS OF LINE CLOCK STATUS REGISTER
2236	001262	000100	LKVEC: .WORD	100	:ADDRESS OF LINE CLOCK VECTOR
2237	001264	172540	PLKSTAT: .WORD	172540	:ADDRESS OF PROG LINE CLOCK STATUS REG
2238	001266	000104	PLKVEC: .WORD	104	:ADDRESS OF PROG LINE CLOCK VECTOR
2239	001270	000000	NEXTTST: .WORD	0	:ADDRESS OF NEXT TEST
2240	001272	000	KB11E: .BYTE	0	:KB-11E/EM WITHOUT MP CACHE
2241	001273	000	KB11EM: .BYTE	0	:KB-11E/EM WITH MP MODS
2242					
2243					
2244	000007		:OPCODE FOR MFPT INSTRUCTION (AVAILABLE ON KB11-E AND KB11-EM ONLY)		
			MFPT=7		

```

2245 ;:*****
2246
2247 .SBTTL ERROR POINTER TABLE
2248
2249 ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
2250 ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
2251 ;*LOCATION $ITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
2252 ;*NOTE1: IF $ITEMB IS 0 THE ONLY PERTINENT DATA IS ($ERRPC).
2253 ;*NOTE2: EACH ITEM IN THE TABLE CONTAINS 4 POINTERS EXPLAINED AS FOLLOWS:
2254
2255 ;* EM ;;POINTS TO THE ERROR MESSAGE
2256 ;* DH ;;POINTS TO THE DATA HEADER
2257 ;* DT ;;POINTS TO THE DATA
2258 ;* DF ;;POINTS TO THE DATA FORMAT
2259
2260
2261 001274 $ERRTB:
2262
2263 001274 036530 ITEM1: EM1 ;EITHER SPL FAILED OR BITS STUCK
2264 001276 036576 DH1 ;ERRORPC SPL 5 PSW SPL 2 PSW TEST NUMBER
2265 ; EXPECT ACTUAL EXPECT ACTUAL
2266 001300 036716 DT1 ;$ERRPC,$PR5,$ERPSW,$PR2,$STMP0
2267 001302 036734 ITEM2: EM2 ;PR5 LOADS OK BUT PR2 DOESN'T
2268 001304 036576 DH1
2269 001306 036716 DT1
2270 001310 036755 ITEM3: EM3 ;PR2 LOADS OK BUT PR5 DOESN'T
2271 001312 036576 DH1
2272 001314 036716 DT1
2273 001316 037002 ITEM4: EM4 ;FORK A FAILED TO D30.00
2274 001320 037045 DH4 ;ERRORPC TEST NUMBER
2275 001322 037072 DT4 ;$ERRPC,$$STSTNM
2276 001324 037100 ITEM5: EM5 ;FORK A FAILED TO RSD.02
2277 001326 037045 DH4
2278 001330 037072 DT4
2279 001332 037166 ITEM6: EM6 ;RESET DID NOT WORK
2280 001334 037045 DH4
2281 001336 037072 DT4
2282 001340 037221 ITEM7: EM7 ;FORK A FAILED INTO TRP.02
2283 001342 037045 DH4
2284 001344 037072 DT4
2285 001346 037235 ITEM10: EM10 ;FORK A FAILED INTO WAT.00
2286 001350 037045 DH4
2287 001352 037072 DT4
2288 001354 037251 ITEM11: EM11 ;FORK A FAILED TO MTP.00
2289 001356 037045 DH4
2290 001360 037072 DT4
2291 001362 037310 ITEM12: EM12 ;FORK A FAILED TO MFP.80
2292 001364 037045 DH4
2293 001366 037072 DT4
2294 001370 037347 ITEM13: EM13 ;PCR DID NOT LOAD FROM R5
2295 001372 037045 DH4
2296 001374 037072 DT4
2297 001376 037377 ITEM14: EM14 ;MARK DID NOT LOAD SP PROPERLY
2298 001400 037427 DH14 ;ERRORPC SP TEST NUMBER
2299 ; EXPECT ACTUAL
2300 001402 037510 DT14 ;$ERRPC,$REG1,$REG0,$$STSTNM
    
```

2301	001404	037522	ITEM15: EM15	;R5 DID NOT LOAD PROPERLY
2302	001406	037552	DH15	;ERRORPC R5 TEST NUMBER
2303	001410	037510	DT14	
2304	001412	037633	ITEM16: EM16	;FORK A FAILED TO RSD.00
2305	001414	037045	DH4	
2306	001416	037072	DT4	
2307	001420	037667	ITEM17: EM17	;FORK A FAILED TO D67.01
2308	001422	037045	DH4	
2309	001424	037072	DT4	
2310	001426	037744	ITEM20: EM20	;R1 SHIFTED RIGHT INSTEAD OF LEFT
2311	001430	037045	DH4	
2312	001432	037072	DT4	
2313	001434	040035	ITEM21: EM21	;R1 DID NOT SHIFT
2314	001436	037045	DH4	
2315	001440	037072	DT4	
2316	001442	040055	ITEM22: EM22	;R1 SHIFTED BUT CARRY DID NOT SET
2317	001444	037045	DH4	
2318	001446	037072	DT4	
2319	001450	040152	ITEM23: EM23	;R1 SHIFTED LEFT INSTEAD OF RIGHT
2320	001452	037045	DH4	
2321	001454	037072	DT4	
2322	001456	040242	ITEM24: EM24	;SHIFT RIGHT DID NOT SIGN FILL
2323	001460	037045	DH4	
2324	001462	037072	DT4	
2325	001464	040275	ITEM25: EM25	;R1 SHIFTED BUT DON'T KNOW WHERE
2326	001466	040327	DH25	;ERRORPC R5 TEST NUMBER
2327				; EXPECT ACTUAL
2328	001470	040402	DT25	;\$ERRPC,\$TMP1,\$REG1,\$\$TSTNM
2329	001472	040414	ITEM26: EM26	;SHIFT OK BUT CARRY DID NOT LOAD
2330	001474	037045	DH4	
2331	001476	037072	DT4	
2332	001500	040454	ITEM27: EM27	;ASH.20 DID NOT LOAD CC'S CORRECTLY
2333	001502	041440	DH42	;ERRORPC PSW TEST NUMBER
2334				; EXPECT ACTUAL
2335	001504	041516	DT42	;\$ERRPC,\$TMP0,\$ERPSW,\$\$TSTNM
2336	001506	040516	ITEM30: EM30	;R1 SHIFTED WITH A SHIFT COUNT OF 0
2337	001510	037045	DH4	
2338	001512	037072	DT4	
2339	001514	040554	ITEM31: EM31	;ASH.40 DID NOT LOAD CC'S CORRECTLY
2340	001516	041440	DH42	
2341	001520	041516	DT42	
2342	001522	040616	ITEM32: EM32	;FORK A FAILED TO RSD.00
2343	001524	037045	DH4	
2344	001526	037072	DT4	
2345	001530	040662	ITEM33: EM33	;STATE ASH.00 FAILED
2346	001532	037045	DH4	
2347	001534	037072	DT4	
2348	001536	040700	ITEM34: EM34	;FORK B FAILED INTO MUL.00
2349	001540	037045	DH4	
2350	001542	037072	DT4	
2351	001544	040733	ITEM35: EM35	;FORK B FAILED TO RSD.00
2352	001546	037045	DH4	
2353	001550	037072	DT4	
2354	001552	041051	ITEM36: EM36	;FORK A FAILED TO RSD.00
2355	001554	037045	DH4	
2356	001556	037072	DT4	

Line No	Address	Item	Description
2357	001560	041134	ITEM37: EM37
2358	001562	037045	DH4
2359	001564	037072	DT4
2360	001566	041204	ITEM40: EM40
2361	001570	037045	DH4
2362	001572	037072	DT4
2363	001574	041254	ITEM41: EM41
2364	001576	041317	DH41
2365			
2366	001600	041376	DT41
2367	001602	041410	ITEM42: EM42
2368	001604	041440	DH42
2369			
2370	001606	041516	DT42
2371	001610	041530	ITEM43: EM43
2372	001612	041562	DH43
2373			
2374	001614	041672	DT43
2375	001616	041710	ITEM44: EM44
2376	001620	041562	DH43
2377	001622	041672	DT43
2378	001624	041754	ITEM45: EM45
2379	001626	041440	DH42
2380	001630	041516	DT42
2381	001632	040275	ITEM46: EM25
2382	001634	041562	DH43
2383	001636	041672	DT43
2384	001640	042132	ITEM47: EM47
2385	001642	041440	DH42
2386	001644	041516	DT42
2387	001646	042157	ITEM50: EM50
2388	001650	041562	DH43
2389	001652	041672	DT43
2390	001654	042211	ITEM51: EM51
2391	001656	042251	DH51
2392			
2393	001660	042410	DT51
2394	001662	042432	ITEM52: EM52
2395	001664	042251	DH51
2396	001666	042410	DT51
2397	001670	042505	ITEM53: EM53
2398	001672	042251	DH51
2399	001674	042410	DT51
2400	001676	042540	ITEM54: EM54
2401	001700	041562	DH43
2402	001702	041672	DT43
2403	001704	042556	ITEM55: EM55
2404	001706	037045	DH4
2405	001710	037072	DT4
2406	001712	042634	ITEM56: EM56
2407	001714	041562	DH43
2408	001716	041672	DT43
2409	001720	042705	ITEM57: EM57
2410	001722	041562	DH43
2411	001724	041672	DT43
2412	001726	042755	ITEM60: EM60

```

;RACE E45 BAD (AFIRO4(1)*MUL:ASHC+MFP))
;RACE E33 BAD (AFIRO5(1)*(MUL:ASHC+MFP))
;R0 DID NOT SIGN FILL ON RIGHT SHIFT
;ERROR PC      R0      TEST NUMBER
;      EXPECT ACTUAL
;$ERRPC,$TMP0,$REG0,$STSTNM
;BAD CC'S ON RIGHT SHIFT
;ERRORPC      PSW      TEST NUMBER
;      EXPECT ACTUAL
;$ERRPC,$TMP0,$ERPSW,$STSTNM
;R0<0> DID NOT GO TO R1<15>
;ERRORPC      R0      R1      TEST NUMBER
;      EXPECT ACTUAL EXPECT ACTUAL
;$ERRPC,$TMP0,$REG0,$TMP1,$REG1,$STSTNM
;R0 DID NOT SHIFT LEFT PROPERLY
;BAD CC'S ON LEFT SHIFT
;R1 DID NOT SHIFT LEFT PROPERLY
;BAD CC'S ON NO SHIFT
;R1 DID NOT ROTATE PROPERLY
;BITS STUCK IN SC (52 PATTERN)
;ERRORPC      R0      R1      C BIT
;      EXPECT ACTUAL EXPECT ACTUAL EXPECT ACTUAL
;$ERRPC,$TMP0,$REG0,$TMP1,$REG1,$TMP2,$ERPSW,$STSTNM
;BITS STUCK IN SC (25 PATTERN)
;IRCB B FORK MUX INPUT B3 NOT GOING LOW
;STATE ASC.00 FAILED
;FORK A FAILED TO RSD.00
;EITHER GRAD DROC STUCK H OR RACK E64 BAD
;EITHER GRAD DROO STUCK LOW OR RACK E64 BAD
;EITHER GRAJ SC=0 NOT GETTING TO RACK E50
    
```

2413	001730	041562		DH43		
2414	001732	041672		DT43		;OR RACK E50 BAD
2415	001734	043056	ITEM61:	EM61		;INSTRUCTION FAILED TO LOAD R0 & R1 CORRECTLY
2416	001736	041562		DH43		;ON POSITIVE
2417	001740	041672		DT43		
2418	001742	043134	ITEM62:	EM62		;BAD CC'S
2419	001744	041440		DH42		
2420	001746	041516		DT42		
2421	001750	043176	ITEM63:	EM63		;R0 DID NOT LOAD ON NEG.
2422	001752	041562		DH43		
2423	001754	041672		DT43		
2424	001756	043236	ITEM64:	EM64		;R1 DID NOT LOAD ON NEG.
2425	001760	041562		DH43		
2426	001762	041672		DT43		
2427	001764	043276	ITEM65:	EM65		;BAD CC'S DUE TO STATE MUL.50
2428	001766	041440		DH42		
2429	001770	041516		DT42		
2430	001772	043333	ITEM66:	EM66		;C DID NOT SET ON OVERFLOW
2431	001774	037045		DH4		
2432	001776	037072		DT4		
2433	002000	043364	ITEM67:	EM67		;C DID NOT SET ON UNDERFLOW
2434	002002	037045		DH4		
2435	002004	037072		DT4		
2436	002006	043416	ITEM70:	EM70		;STATE MUL.00 FAILED
2437	002010	041562		DH43		
2438	002012	041672		DT43		
2439	002014	042556	ITEM71:	EM55		;BAD FIELD IN IR DECODE ROM
2440	002016	037045		DH4		
2441	002020	037072		DT4		
2442	002022	043434	ITEM72:	EM72		;STATE ASH.30 DID NOT LOAD CC'S CORRECTLY
2443	002024	041440		DH42		
2444	002026	041516		DT42		
2445	002030	043476	ITEM73:	EM73		;STATE ASH.41 FAILED
2446	002032	040327		DH25		
2447	002034	040402		DT25		
2448	002036	043514	ITEM74:	EM74		;BAD CC'S DUE TO STATE ASH.41
2449	002040	041440		DH42		
2450	002042	041516		DT42		
2451	002044	043556	ITEM75:	EM75		;BEN2 FAILED
2452	002046	037045		DH4		
2453	002050	037072		DT4		
2454	002052	043643	ITEM76:	EM76		;BAD CC'S DUE TO DVE.00
2455	002054	041440		DH42		
2456	002056	041516		DT42		
2457	002060	043666	ITEM77:	EM77		;BAD CC'S DUE TO DVC.70
2458	002062	041440		DH42		
2459	002064	041516		DT42		
2460	002066	043711	ITE100:	EM100		;BEN16 FAILED
2461	002070	037045		DH4		
2462	002072	037072		DT4		
2463	002074	043741	ITE101:	EM101		;BEN4 FAILED
2464	002076	037045		DH4		
2465	002100	037072		DT4		
2466	002102	044065	ITE102:	EM102		;QUOTIENT OK REMAINDER BAD
2467	002104	041562		DH43		
2468	002106	041672		DT43		

2469	002110	044137	ITE103:	EM103	:BEN3 FAILED
2470	002112	037045		DH4	
2471	002114	037072		DT4	
2472	002116	044166	ITE104:	EM104	:BEN04 STUCK TO DIV SUB
2473	002120	037045		DH4	
2474	002122	037072		DT4	
2475	002124	044256	ITE105:	EM105	:CAN'T DETERMINE CAUSE OF FAILURE
2476	002126	041562		DH43	
2477	002130	041672		DT43	
2478	002132	044307	ITE106:	EM106	:BEN16 FAILED TO DIV.50
2479	002134	037045		DH4	
2480	002136	037072		DT4	
2481	002140	044256	ITE107:	EM105	:EM107 SAME AS EM105
2482	002142	041562		DH43	
2483	002144	041672		DT43	
2484	002146	044377	ITE110:	EM110	:BEN04*-N FAILED
2485	002150	037045		DH4	
2486	002152	037072		DT4	
2487	002154	044467	ITE111:	EM111	:BEN02 FAILED
2488	002156	037045		DH4	
2489	002160	037072		DT4	
2490	002162	044517	ITE112:	EM112	:BEN05 FAILED
2491	002164	037045		DH4	
2492	002166	037072		DT4	
2493	002170	044637	ITE113:	EM113	:BEN16 FAILED (RACK E50(B0))
2494	002172	037045		DH4	
2495	002174	037072		DT4	
2496	002176	044666	ITE114:	EM114	:BEN16 FAILED (RACK E64(B0))
2497	002200	037045		DH4	
2498	002202	037072		DT4	
2499	002204	044715	ITE115:	EM115	:ROM STATE FAILED
2500	002206	041562		DH43	
2501	002210	041672		DT43	
2502	002212	044065	ITE116:	EM102	:QUOTIENT OK, REMAINDER BAD
2503	002214	041562		DH43	
2504	002216	041672		DT43	
2505	002220	044763	ITE117:	EM117	:BAD CC'S IN DVC.90 OR RACK E63 BAD
2506	002222	041440		DH42	
2507	002224	041516		DT42	
2508	002226	045040	ITE120:	EM120	:BEN05 DIV QUIT (N(0)*SR15(0)) DID NOT
2509	002230	037045		DH4	:GO LOW OR RACK E63(C0) STUCK HIGH
2510	002232	037072		DT4	
2511	002234	045127	ITE121:	EM121	:CC'S BAD DUE TO DIV.30 OR DVE.20
2512	002236	041440		DH42	
2513	002240	041516		DT42	
2514	002242	045177	ITE122:	EM122	:GRAJ E5 BAD
2515	002244	037045		DH4	
2516	002246	037072		DT4	
2517	002250	045237	ITE123:	EM123	:RACK E63(D0) STUCK LOW
2518	002252	037045		DH4	
2519	002254	037072		DT4	
2520	002256	045266	ITE124:	EM124	:CC'S DID NOT LOAD PROPERLY
2521	002260	041440		DH42	
2522	002262	041516		DT42	
2523	002264	045320	ITE125:	EM125	:EITHER GRAJ E5(N(1)*SR15(1)) BAD
2524	002266	041562		DH43	:OR ROM STATE BAD

2525	002270	041672		
2526	002272	044065	ITE126:	DT43
2527	002274	041562		;QUOT. OK REMAINDER BAD
2528	002276	041672		DH43
2529	002300	045373	ITE127:	DT43
2530	002302	041562		;QUOT. BAD, REMAINDER OK
2531	002304	041672		DH43
2532	002306	045445	ITE130:	DT43
2533	002310	041562		;QUOTIENT BAD
2534	002312	041672		DH43
2535	002314	044065	ITE131:	DT43
2536	002316	041562		;QUOT. OK, REMAINDER BAD
2537	002320	041672		DH43
2538	002322	045445	ITE132:	DT43
2539	002324	041562		;QUOT. BAD
2540	002326	041672		DH43
2541	002330	044065	ITE133:	DT43
2542	002332	041562		;QUOT. OK, REMAIN. BAD
2543	002334	041672		DH43
2544	002336	045506	ITE134:	DT43
2545	002340	041440		;BAD CC'S ON DIVISION OVERFLOW
2546	002342	041516		DH42
2547	002344	045555	ITE135:	DT42
2548	002346	041317		;RO DID NOT LOAD CORRECTLY
2549	002350	041376		DH41
2550	002352	045564	ITE136:	DT41
2551	002354	037045		;THE SP DID NOT INCREMENT
2552	002356	037072		DH4
2553	002360	045266	ITE137:	DT4
2554	002362	041440		;CC'S DID NOT LOAD CORRECTLY
2555	002364	041516		DH42
2556	002366	045617	ITE140:	DT42
2557	002370	037045		;FORK A FAILED
2558	002372	037072		DH4
2559	002374	045654	ITE141:	DT4
2560	002376	037045		;STATE MTP.10 FAILED
2561	002400	037072		DH4
2562	002402	045713	ITE142:	DT4
2563	002404	037427		;SP LOADED INCORRECTLY
2564	002406	037510		DH14
2565	002410	045741	ITE143:	DT14
2566	002412	037045		;STATE MTP.00 DID NOT PUT PCB IN DR
2567	002414	037072		DH4
2568	002416	045775	ITE144:	DT4
2569	002420	037045		;FORK A FAILED
2570	002422	037072		DH4
2571	002424	046024	ITE145:	DT4
2572	002426	037045		;STATE MFP.10 DID NOT DEC. THE SP
2573	002430	037072		DH4
2574	002432	046063	ITE146:	DT4
2575	002434	037045		;RO DID NOT GET PUT ON THE STACK
2576	002436	037072		DH4
2577	002440	046122	ITE147:	DT4
2578	002442	041440		;BAD CC'S
2579	002444	041516		DH42
2580	002446	045617	ITE150:	DT42
				;RACF X/CLASS DOES NOT GO HIGH

2581	002450	037045	DH4	
2582	002452	037072	DT4	
2583	002454	046153	ITE151: EM151	;STATE MFP.00 IS BAD
2584	002456	037045	DH4	
2585	002460	037072	DT4	
2586	002462	046166	ITE152: EM152	;BAD CC'S
2587	002464	041440	DH42	
2588	002466	041516	DT42	
2589	002470	037633	ITE153: EM16	;RACE X/CLASS DOES NOT GO HIGH
2590	002472	037045	DH4	
2591	002474	037072	DT4	
2592	002476	042556	ITE154: EM55	;R(NUL:ASHC+MFP) FIELD OF IR ROM BAD
2593	002500	037045	DH4	
2594	002502	037072	DT4	
2595	002504	046215	ITE155: EM155	;STATE TRP.01 FAILED
2596	002506	037045	DH4	
2597	002510	037072	DT4	
2598	002512	046246	ITE156: EM156	;TRAP VECTOR DECODE FAILED
2599	002514	037045	DH4	
2600	002516	037072	DT4	
2601	002520	046366	ITE157: EM157	;STATE TRP.01 FAILED
2602	002522	037045	DH4	
2603	002524	037072	DT4	
2604	002526	046417	ITE160: EM160	;STATE TRP.01 FAILED
2605	002530	037045	DH4	
2606	002532	037072	DT4	
2607	002534	046450	ITE161: EM161	;TRAP VECTOR DECODE FAILED
2608	002536	037045	DH4	
2609	002540	037072	DT4	
2610	002542	046545	ITE162: EM162	;STATE TRP.01 FAILED
2611	002544	037045	DH4	
2612	002546	037072	DT4	
2613	002550	046576	ITE163: EM163	;BIT FAILED IN PIRQ REG
2614	002552	046625	DH163	;ERRORPC PIRQ TEST NUMBER
2615				; EXPECT ACTUAL
2616	002554	046704	ITE164: EM164	;SERRPC,\$TMPO,\$EPIRQ,\$\$TSTNM
2617	002556	046716	DH4	;STATE TST.10 HAS BAD BEN FIELD
2618	002560	037045	DT4	
2619	002562	037072	ITE165: EM165	;HONOR PIR 1 DOES NOT GO LOW
2620	002564	046747	DH4	
2621	002566	037045	DT4	
2622	002570	037072	ITE166: EM166	;PIR 6 WORKS BUT 4 & 1 DON'T
2623	002572	047031	DH4	
2624	002574	037045	DT4	
2625	002576	037072	ITE167: EM167	;TMCA ABOVE BR7 MIGHT BE STUCK LOW
2626	002600	047136	DH4	
2627	002602	037045	DT4	
2628	002604	037072	ITE170: EM170	;TMCE BRQ CLDCK MIGH BE STUCH LOW
2629	002606	047200	DH4	
2630	002610	037045	DT4	
2631	002612	037072	ITE171: EM171	;BEN 13 FAILED TO PUP.00
2632	002614	047242	DH4	
2633	002616	037045	DT4	
2634	002620	037072	ITE172: EM172	;BEN 13 FAILED TO SER.00
2635	002622	047363	DH4	
2636	002624	037045	DT4	

2637	002626	037072		
2638	002630	042003	ITE173:	DT4
2639	002632	042036		EM46
2640	002634	042120		DH46
2641	002636	047636	ITE174:	DT46
2642	002640	037045		EM174
2643	002642	037072		DH4
2644	002644	047720	ITE175:	DT4
2645	002646	037045		EM175
2646	002650	037072		DH4
2647	002652	047735	ITE176:	DT4
2648	002654	050127		EM176
2649	002656	050160		DH201
2650	002660	047777	ITE177:	DT201
2651	002662	037045		EM177
2652	002664	037072		DH4
2653	002666	047720	ITE200:	DT4
2654	002670	037045		EM175
2655	002672	037072		DH4
2656	002674	050061	ITE201:	DT4
2657	002676	050127		EM201
2658	002700	050160		DH201
2659	002702	050170	ITE202:	DT201
2660	002704	037045		EM202
2661	002706	037072		DH4
2662	002710	047720	ITE203:	DT4
2663	002712	037045		EM175
2664	002714	037072		DH4
2665	002716	050252	ITE204:	DT4
2666	002720	050127		EM204
2667	002722	050160		DH201
2668	002724	050320	ITE205:	DT201
2669	002726	037045		EM205
2670	002730	037072		DH4
2671	002732	047720	ITE206:	DT4
2672	002734	037045		EM175
2673	002736	037072		DH4
2674	002740	050403	ITE207:	DT4
2675	002742	050127		EM207
2676	002744	050160		DH201
2677	002746	050451	ITE210:	DT201
2678	002750	037045		EM210
2679	002752	037072		DH4
2680	002754	050517	ITE211:	DT4
2681	002756	037045		EM211
2682	002760	037072		DH4
2683	002762	050600	ITE212:	DT4
2684	002764	037045		EM212
2685	002766	037072		DH4
2686	002770	050644	ITE213:	DT4
2687	002772	050127		EM213
2688	002774	050160		DH201
2689	002776	050712	ITE214:	DT201
2690	003000	037045		EM214
2691	003002	037072		DH4
2692	003004	047720	ITE215:	DT4
				EM175

;MFPT FAILED TO LOAD R0 CORRECTLY

;PIR 2 DID NOT INTERRUPT

;BEN 13 FAILED

;LEVEL 1 INT. WHEN CPU LEVEL 1 ENABLED
 ;ERRORPC PIRQ TEST NUMBER
 ;\$ERRPC,\$EPIRQ,\$\$TSTNM
 ;PIR 3 DID NOT INTERRUPT

;BEN 13 FAILED

;LEVEL 2 WHEN CPU LEVEL 2 ENABLED

;PIR 4 DID NOT INTERRUPT

;BEN 13 FAILED

;LEVEL 3 WHEN CPU LEVEL 3 ENABLED

;PIR 5 DID NOT INTERRUPT

;BEN 13 FAILED

;LEVEL 4 INTERRUPT WHEN NOT SUPPOSE TO

;FAILURE AFTER TMCB E70

;FAILURE IN TMCB E70 OR BEFORE

;BEN 13 FAILED

;LEVEL 5 INTERRUPT WHEN NOT SUPPOSE TO

;LEVEL 7 DID NOT INTERRUPT

;BEN 13 FAILED

2693	003006	037045	DH4	
2694	003010	037072	DT4	
2695	003012	050774	ITE216: EM216	;LEVEL 6 INTERRUPT WHEN NOT SUPPOSE TO
2696	003014	050127	DH201	
2697	003016	050160	DT201	
2698	003020	051042	ITE217: EM217	;NO TIMEOUT ON DATI
2699	003022	037045	DH4	
2700	003024	037072	DT4	
2701	003026	051076	ITE220: EM220	;BEN 13 FAILED
2702	003030	037045	DH4	
2703	003032	037072	DT4	
2704	003034	051157	ITE221: EM221	;NO TIMEOUT ON DATO
2705	003036	037045	DH4	
2706	003040	037072	DT4	
2707	003042	051213	ITE222: EM222	;BR 4 INTERRUPTS WHEN CPU AT LEVEL 7
2708	003044	037045	DH4	
2709	003046	037072	DT4	
2710	003050	051273	ITE223: EM223	;BOTH BR 4 & BR 6 FAILED
2711	003052	037045	DH4	
2712	003054	037072	DT4	
2713	003056	051314	ITE224: EM224	;BR 4 FAILED
2714	003060	037045	DH4	
2715	003062	037072	DT4	
2716	003064	051463	ITE225: EM225	;BR 4 FAILED BUT BR 6 OK
2717	003066	037045	DH4	
2718	003070	037072	DT4	
2719	003072	051574	ITE226: EM226	;BR 5 INTERRUPT FAILED
2720	003074	037045	DH4	
2721	003076	037072	DT4	
2722	003100	051654	ITE227: EM227	;BR 6 INTERRUPT FAILED
2723	003102	037045	DH4	
2724	003104	037072	DT4	
2725	003106	051735	ITE230: EM230	;YEL ZONE TRAP FAILED
2726	003110	037045	DH4	
2727	003112	037072	DT4	
2728	003114	000000	ITE231: 0	;DELETED
2729	003116	037045	DH4	
2730	003120	037072	DT4	
2731	003122	052222	ITE232: EM232	;JSR WITH BAD STACK FAILED
2732	003124	037045	DH4	
2733	003126	037072	DT4	
2734	003130	052274	ITE233: EM233	;STACK LIMIT REG DID NOT DISABLE YEL ZONE
2735	003132	037045	DH4	
2736	003134	037072	DT4	
2737	003136	052371	ITE234: EM234	;TMCC KERNAL R6 DID NOT DISABLE YEL ZONE
2738	003140	037045	DH4	
2739	003142	037072	DT4	
2740	003144	052476	ITE235: EM235	;RED ZONE REFERENCE FAILED
2741	003146	037045	DH4	
2742	003150	037072	DT4	
2743	003152	000000	ITE236: 0	;DELETED
2744	003154	037045	DH4	
2745	003156	037072	DT4	
2746	003160	052643	ITE237: EM237	;BEN 13 FAILED TO PUP.00
2747	003162	037045	DH4	
2748	003164	037072	DT4	

2749	003166	052736	ITE240:	EM240	:BEN 13 FAILED TO BRK.80
2750	003170	037045		DH4	
2751	003172	037072		DT4	
2752	003174	053040	ITE241:	EM241	:NO RED ZONE ON STACK OVERFLOW
2753	003176	037045		DH4	
2754	003200	037072		DT4	
2755	003202	053151	ITE242:	EM242	:NO RED ZONE WHEN SL REG>BADDR
2756	003204	037045		DH4	
2757	003206	037072		DT4	
2758	003210	053305	ITE243:	EM243	:52400 PATTERN FAILED IN SL REG
2759	003212	053355		DH243	:ERRORPC SL REG TEST NUMBER
2760					: EXPECT ACTUAL
2761	003214	053440		DT243	:SERRPC,\$TMP0,E2STKLMT,\$\$TSTNM
2762	003216	053452	ITE244:	EM244	:125000 PATTERN FAILED IN SL REG
2763	003220	053355		DH243	
2764	003222	053522		DT244	:SERRPC,\$TMP0,E1STKLMT,\$\$TSTNM
2765	003224	053534	ITE245:	EM245	:BR SELECTED INSTEAD OF SL
2766	003226	037045		DH4	
2767	003230	037072		DT4	
2768	003232	053627	ITE246:	EM246	:SL AND PB BOTH SELECTED
2769	003234	037045		DH4	
2770	003236	037072		DT4	
2771	003240	053750	ITE247:	EM247	:PSW SELECTED INSTEAD OF SL
2772	003242	037045		DH4	
2773	003244	037072		DT4	
2774	003246	054061	ITE250:	EM250	:WHAT HAPPENED?
2775	003250	054106		DH250	:BOTH PATTERNS FAILED
2776	003252	054224		DT250	
2777	003254	054242	ITE251:	EM251	:YEL ZONE IN RED REGION (SP=330)
2778	003256	037045		DH4	
2779	003260	037072		DT4	
2780	003262	054407	ITE252:	EM252	:YEL ZONE IN RED REGION (SP=240)
2781	003264	037045		DH4	
2782	003266	037072		DT4	
2783	003270	054457	ITE253:	EM253	:YEL ZONE IN RED REGION (SP=140)
2784	003272	037045		DH4	
2785	003274	037072		DT4	
2786	003276	054527	ITE254:	EM254	:RED ZONE IN YELLOW REGION (SP=376)
2787	003300	037045		DH4	
2788	003302	037072		DT4	
2789	003304	054617	ITE255:	EM255	:CPU ERR REG BIT 4 DOES NOT SET
2790	003306	054676		DH255	
2791	003310	041376		DT41	
2792	003312	054757	ITE256:	EM256	:CPU ERR REG DOES NOT CLEAR
2793	003314	037045		DH4	
2794	003316	037072		DT4	
2795	003320	055014	ITE257:	EM257	:CPU ERROR BIT 3 DOES NOT SET
2796	003322	054676		DH255	
2797	003324	041376		DT41	
2798	003326	055065	ITE260:	EM260	:CPU ERROR BIT 3 DOES NOT CLEAR
2799	003330	037045		DH4	
2800	003332	037072		DT4	
2801	003334	055130	ITE261:	EM261	:CPU ERROR BIT 2 DOES NOT SET
2802	003336	054676		DH255	
2803	003340	041376		DT41	
2804	003342	055202	ITE262:	EM262	:CPU ERROR BIT 2 DOES NOT CLEAR

2805	003344	037045	DH4	
2806	003346	037072	DT4	
2807	003350	055244	ITE263: EM263	
2808	003352	055433	DH263	
2809	003354	000000	0	
2810	003356	056056	ITE264: EM264	:GOING TO NEXT TEST
2811	003360	000000	0	
2812	003362	000000	0	
2813	003364	056101	ITE265: EM265	:NEITHER - BYIN NOR DATI CAUSED ODD ADDR.
2814	003366	037045	DH4	
2815	003370	037072	DT4	
2816	003372	056417	ITE266: EM266	:DATI TRAPPED BUT - BYIN DIDN'T
2817	003374	037045	DH4	
2818	003376	037072	DT4	
2819	003400	056535	ITE267: EM267	:BOTH DATI & DATO FAILED
2820	003402	037045	DH4	
2821	003404	037072	DT4	
2822	003406	056615	ITE270: EM270	:DATO WORKS BUT DATI FAILED
2823	003410	037045	DH4	
2824	003412	037072	DT4	
2825	003414	056713	ITE271: EM271	:NO TRAP ON DATO
2826	003416	037045	DH4	
2827	003420	037072	DT4	
2828	003422	057043	ITE272: EM272	:NO TRAP ON SM357*SRC1 DATI
2829	003424	037045	DH4	
2830	003426	037072	DT4	
2831	003430	057102	ITE273: EM273	:ODD ADDR BIT IN CPU ERROR FAILED
2832	003432	037045	DH4	
2833	003434	037072	DT4	
2834	003436	057150	ITE274: EM274	:NO TRAP ON DATO
2835	003440	037045	DH4	
2836	003442	037072	DT4	
2837	003444	057170	ITE275: EM275	:T BIT TRAP FAILED
2838	003446	037045	DH4	
2839	003450	037072	DT4	
2840	003452	057540	ITE276: EM276	:T BIT NEVER SET
2841	003454	037045	DH4	
2842	003456	037072	DT4	
2843	003460	057571	ITE277: EM277	:PS<08> DID NOT DISABLE T TRAP (KB-11E/EM ONLY)
2844	003462	037045	DH4	
2845	003464	037072	DT4	
2846	003466	057645	ITE300: EM300	:TRAP VECTOR DECODE FAILED
2847	003470	037045	DH4	
2848	003472	037072	DT4	
2849	003474	057734	ITE301: EM301	:RTT DID NOT DISABLE T BIT
2850	003476	037045	DH4	
2851	003500	037072	DT4	
2852	003502	060021	ITE302: EM302	:PIR 1 DID NOT DISABLE ON BR4
2853	003504	037045	DH4	
2854	003506	037072	DT4	
2855	003510	060066	ITE303: EM303	:PIR 1 CAME THRU ON YELLOW ZONE
2856	003512	037045	DH4	
2857	003514	037072	DT4	
2858	003516	060217	ITE304: EM304	:PIR 2 CAME THRU ON YELLOW ZONE
2859	003520	037045	DH4	
2860	003522	037072	DT4	

2861	003524	060304	ITE305:	EM305	;PIR 3 CAME THRU ON YELLOW ZONE
2862	003526	037045		DH4	
2863	003530	037072		DT4	
2864	003532	060372	ITE306:	EM306	;BR4 CAME THRU ON PIR 5
2865	003534	037045		DH4	
2866	003536	037072		DT4	
2867	003540	060514	ITE307:	EM307	;BR4 CAME THRU ON PIR 5
2868	003542	037045		DH4	
2869	003544	037072		DT4	
2870	003546	060567	ITE310:	EM310	;BR4 CAME THRU ON BR 5
2871	003550	037045		DH4	
2872	003552	037072		DT4	
2873	003554	060641	ITE311:	EM311	;BR4 CAME IN ON PIR 6
2874	003556	037045		DH4	
2875	003560	037072		DT4	
2876	003562	060712	ITE312:	EM312	;BR4 CAME IN ON PIR 7
2877	003564	037045		DH4	
2878	003566	037072		DT4	
2879	003570	061036	ITE313:	EM313	;PIR 4 CAME IN ON BR5
2880	003572	037045		DH4	
2881	003574	037072		DT4	
2882	003576	061106	ITE314:	EM314	;PIR 4 CAME IN ON BR6
2883	003600	037045		DH4	
2884	003602	037072		DT4	
2885	003604	061217	ITE315:	EM315	;PIR 4 CAME THRU ON SL YELLOW
2886	003606	037045		DH4	
2887	003610	037072		DT4	
2888	003612	061341	ITE316:	EM316	;BR5 CAME IN ON PIR 5
2889	003614	037045		DH4	
2890	003616	037072		DT4	
2891	003620	061454	ITE317:	EM317	;BR5 CAME IN ON PIR 6
2892	003622	037045		DH4	
2893	003624	037072		DT4	
2894	003626	061474	ITE320:	EM320	;BR5 CAME IN ON PIR 7
2895	003630	037045		DH4	
2896	003632	037072		DT4	
2897	003634	061612	ITE321:	EM321	;PIR 5 CAME IN ON BR6
2898	003636	037045		DH4	
2899	003640	037072		DT4	
2900	003642	061662	ITE322:	EM322	;PIR 5 CAME IN ON SL YELLOW
2901	003644	037045		DH4	
2902	003646	037072		DT4	
2903	003650	061737	ITE323:	EM323	;BR6 CAME IN ON PIR 6
2904	003652	037045		DH4	
2905	003654	037072		DT4	
2906	003656	062047	ITE324:	EM324	;BR6 CAME IN ON PIR 7
2907	003660	037045		DH4	
2908	003662	037072		DT4	
2909	003664	062130	ITE325:	EM325	;PIR 6 CAME IN ON SL YELLOW
2910	003666	037045		DH4	
2911	003670	037072		DT4	
2912	003672	062205	ITE326:	EM326	;PIR 7 CAME IN ON SL YELLOW
2913	003674	037045		DH4	
2914	003676	037072		DT4	
2915	003700	062266	ITE327:	EM327	;PSW BIT 11 DID NOT SET
2916	003702	037045		DH4	

2917	003704	037072		DT4	
2918	003706	062431	ITE330:	EM330	;R12 CLEARED R2
2919	003710	037045		DH4	
2920	003712	037072		DT4	
2921	003714	062542	ITE331:	EM331	;R2 SRC WAS AFFECTED BY CLR R12
2922	003716	037045		DH4	
2923	003720	037072		DT4	
2924	003722	062655	ITE332:	EM332	;R2 DST WAS AFFECTED BY (R12)+
2925	003724	037045		DH4	
2926	003726	037072		DT4	
2927	003730	062736	ITE333:	EM333	;R2 SRC WAS AFFECTED BY (R12)+
2928	003732	037045		DH4	
2929	003734	037072		DT4	
2930	003736	062771	ITE334:	EM334	;R15 DST AFTER UPAD 2
2931	003740	037045		DH4	
2932	003742	037072		DT4	
2933	003744	063024	ITE335:	EM335	;R15 SRC DOES NOT SELECT ON UPAD 2
2934	003746	037045		DH4	
2935	003750	037072		DT4	
2936	003752	063061	ITE336:	EM336	;PDRD PS11 DOES NOT GET TO GRAC
2937	003754	037045		DH4	
2938	003756	037072		DT4	
2939	003760	063130	ITE337:	EM337	;BAD BITS IN GPR SET 1 SRC
2940	003762	063175		DH337	;ERRORPC PATTERN TESTNUMBER
2941	003764	063232		DT337	;\$ERRPC,\$TMP1,\$\$TSTNM
2942	003766	063242	ITE340:	EM340	;BAD BITS IN GPR SET 1 DST
2943	003770	063175		DH337	
2944	003772	063232		DT337	
2945	003774	063314	ITE341:	EM341	;GRAB DST SET 1 DOES NOT GO LOW ON R14
2946	003776	037045		DH4	
2947	004000	037072		DT4	
2948	004002	063361	ITE342:	EM342	;GRAB SRC SET 1 DOES NOT GO LOW ON R14
2949	004004	037045		DH4	
2950	004006	037072		DT4	
2951	004010	063426	ITE343:	EM343	;50000 PATTERN FAILED IN PSW
2952	004012	041440		DH42	
2953	004014	041516		DT42	
2954	004016	063462	ITE344:	EM344	;164000 PATTERN FAILED IN PSW
2955	004020	041440		DH42	
2956	004022	041516		DT42	
2957	004024	063517	IRE345:	EM345	;PSW HIGH BYTE DID NOT CLEAR
2958	004026	041440		DH42	
2959	004030	041516		DT42	
2960	004032	063552	ITE346:	EM346	;SUPER SP DOES NOT SELECT ON UPAD 5
2961	004034	037045		DH4	
2962	004036	037072		DT4	
2963	004040	063632	ITE347:	EM347	;SUPER SP DOES NOT SELECT ON UPAD 0
2964	004042	037045		DH4	
2965	004044	037072		DT4	
2966	004046	063712	ITE350:	EM350	;USER SP DOES NOT SELECT ON UPAD 5
2967	004050	037045		DH4	
2968	004052	037072		DT4	
2969	004054	063767	ITE351:	EM351	;USER SP DOES NOT SELECT ON UPAD 0
2970	004056	037045		DH4	
2971	004060	037072		DT4	
2972	004062	064044	ITE352:	EM352	;EITHER USER OR SUPER SP DST FAILED

2973	004064	063175		DH337	
2974	004066	063232		DT337	
2975	004070	064111	ITE353:	EM353	;EITHER USER OR SUPER SP SRC FAILED
2976	004072	063175		DH337	
2977	004074	063232		DT337	
2978	004076	064156	ITE354:	EM354	;KSP SRC CHANGED ON MTP
2979	004100	037045		DH4	
2980	004102	037072		DT4	
2981	004104	064210	ITE355:	EM355	;KSP SRC & DST CHANGED ON MTP
2982	004106	037045		DH4	
2983	004110	037072		DT4	
2984	004112	064267	ITE356:	EM356	;KSP DST CHANGED ON MTP
2985	004114	037045		DH4	
2986	004116	037072		DT4	
2987	004120	064322	ITE357:	EM357	;SSP DID NOT LOAD PROPERLY ON MTPD
2988	004122	064413		DH357	;ERRORPC SSP TEST NUMBER
2989					; EXPECT ACTUAL
2990	004124	037510		DT14	
2991	004126	064465	ITE360:	EM360	;USER SP DID NOT LOAD ON MTP
2992	004130	037045		DH4	
2993	004132	037072		DT4	
2994	004134	064544	ITE361:	EM361	;BAD FIELD IN IR DECODE ROM
2995	004136	037045		DH4	
2996	004140	037072		DT4	
2997	004142	064614	ITE362:	EM362	;SSP WAS READ AND USP WAS WRITTEN
2998	004144	037045		DH4	
2999	004146	037072		DT4	
3000	004150	064651	ITE363:	EM363	;SSP WAS READ AND WRITTEN
3001	004152	037045		DH4	
3002	004154	037072		DT4	
3003	004156	064721	ITE364:	EM364	;CAN'T DETERMINE WHAT HAPPENED
3004	004160	037045		DH4	;SSP WAS READ BUT THE WRITE FAILED
3005	004162	037072		DT4	
3006	004164	065003	ITE365:	EM365	;USP WAS READ BUT SSP WAS WRITTEN
3007	004166	037045		DH4	
3008	004170	037072		DT4	
3009	004172	065040	ITE366:	EM366	;USP WAS READ BUT REG 7 WAS WRITTEN
3010	004174	037045		DH4	
3011	004176	037072		DT4	
3012	004200	065074	ITE367:	EM367	;SPL WORKED IN SUPERVISOR MODE
3013	004202	041440		DH42	
3014	004204	041516		DT42	
3015	004206	065210	ITE370:	EM370	;BIT FAILED TO SET IN PSW
3016	004210	041440		DH42	
3017	004212	041516		DT42	
3018	004214	065335	ITE371:	EM371	;BIT FAILED TO CLEAR IN PSW
3019	004216	041440		DH42	
3020	004220	041516		DT42	
3021	004222	065442	ITE372:	EM372	;BITS <13:11> DID NOT PRESET
3022	004224	041440		DH42	
3023	004226	041516		DT42	
3024	004230	065572	ITE373:	EM373	;IOT DID NOT CHANGE PSW CORRECTLY
3025	004232	041440		DH42	
3026	004234	041516		DT42	
3027	004236	065704	ITE374:	EM374	;PREVIOUS MODE BITS DID NOT CLEAR
3028	004240	041440		DH42	

3029	004242	041516			
3030	004244	065774	ITE375:	EM375	;NO KT ABORT
3031	004246	037045		DH4	
3032	004250	037072		DT4	
3033	004252	066147	ITE376:	EM376	;PS<08> FAILED TO SET
3034	004254	037045		DH4	
3035	004256	037072		DT4	
3036	004260	000000	ITE377:	0	
3037	004262	000000		0	
3038	004264	000000		0	
3039	004266	066177	ITE400:	EM400	;KT ABORT TRAPPED TO 4
3040	004270	037045		DH4	
3041	004272	037072		DT4	
3042	004274	066300	ITE401:	EM401	;KT ABORT TRAPPED TO 10
3043	004276	037045		DH4	
3044	004300	037072		DT4	
3045	004302	066345	ITE402:	EM402	;KT ABORT TRAPPED TO 240
3046	004304	037045		DH4	
3047	004306	037072		DT4	
3048	004310	066407	ITE403:	EM403	;KT TRAP DID NOT WORK
3049	004312	037045		DH4	
3050	004314	037072		DT4	
3051	004316	000000	ITE404:	0	;DELETED
3052	004320	037045		DH4	
3053	004322	037072		DT4	
3054	004324	066513	ITE405:	EM405	;NO KT TRAP ON SOURCE OPERAND
3055	004326	037045		DH4	
3056	004330	037072		DT4	
3057	004332	066631	ITE406:	EM406	;NO ABORT ON NEXM
3058	004334	037045		DH4	
3059	004336	037072		DT4	
3060	004340	000000	ITE407:	0	;DELETED
3061	004342	037045		DH4	
3062	004344	037072		DT4	
3063	004346	066761	ITE410:	EM410	;NEXM BIT DID NOT SET IN CPUERR REG
3064	004350	054676		DH255	
3065	004352	041376		DT41	
3066	004354	067064	ITE411:	EM411	;NEXM BIT DID NOT CLEAR IN CPUERR REG
3067	004356	037045		DH4	
3068	004360	037072		DT4	
3069	004362	067133	ITE412:	EM412	;KT ABORT ON NEXM (KB11-B/C)
3070	004364	037045		DH4	
3071	004366	037072		DT4	
3072	004370	067210	ITE413:	EM413	;KT ABORT ON SL RED
3073	004372	037045		DH4	
3074	004374	037072		DT4	
3075	004376	067263	ITE414:	EM414	;KT ABORT ON ODD ADDRESS
3076	004400	037045		DH4	
3077	004402	037072		DT4	
3078	004404	067344	ITE415:	EM415	;KT ABORT FAILED TO OVER-RIDE NEXM (KB11-E/EM)
3079	004406	037045		DH4	
3080	004410	037072		DT4	
3081	004412	067427	ITE416:	EM416	;TMCE CACHE BEND DID NOT GO
3082	004414	037045		DH4	;HIGH ON KT ABORT
3083	004416	037072		DT4	
3084	004420	067502	ITE417:	EM417	;ILLEGAL HALT DID NOT TRAP

3085	004422	037045	DH4		
3086	004424	037072	DT4		
3087	004426	067573	ITE420:	EM420	;CPU ERROR REG BIT 5 DID NOT SET
3088	004430	054676		DH255	
3089	004432	041376		DT41	
3090	004434	067650	ITE421:	EM421	;BEN 6 FAILED ON PS RESTORE
3091	004436	037045		DH4	
3092	004440	037072		DT4	
3093	004442	067741	ITE422:	EM422	;NO PE ABORT
3094	004444	037045		DH4	
3095	004446	037072		DT4	
3096	004450	070151	ITE423:	EM423	;PE ABORTED TO 4
3097	004452	037045		DH4	
3098	004454	037072		DT4	
3099	004456	070243	ITE424:	EM424	;PE ABORTED TO 14
3100	004460	037045		DH4	
3101	004462	037072		DT4	
3102	004464	070322	ITE425:	EM425	;PE ABORTED TO 104
3103	004466	037045		DH4	
3104	004470	037072		DT4	
3105	004472	070341	ITE426:	EM426	;NO PE TRAP
3106	004474	037045		DH4	
3107	004476	037072		DT4	
3108	004500	070425	ITE427:	EM427	;PE TRAP, TRAPPED TO
3109	004502	037045		DH4	;WRONG VECTOR
3110	004504	037072		DT4	
3111	004506	070531	ITE430:	EM430	;PIR 6 CAME IN ON MEM MGT TRAP
3112	004510	037045		DH4	
3113	004512	037072		DT4	
3114	004514	070574	ITE431:	EM431	;PIR 3 CAME IN ON MEM MGT TRAP
3115	004516	037045		DH4	
3116	004520	037072		DT4	
3117	004522	070640	ITE432:	EM432	;YEL ZONE CAME IN ON PE TRAP
3118	004524	037045		DH4	
3119	004526	037072		DT4	
3120	004530	070724	ITE433:	EM433	;MEM MGT TRAP CAME IN ON PE
3121	004532	037045		DH4	
3122	004534	037072		DT4	
3123	004536	000000	ITE434:	0	;DELETED
3124	004540	037045		DH4	
3125	004542	037072		DT4	
3126	004544	070772	ITE435:	EM435	;TMCC PRIORITY CLEAR DID NOT WORK
3127	004546	037045		DH4	
3128	004550	037072		DT4	
3129	004552	071112	ITE436:	EM436	;UNIBUS PE ABORT DID NOT HAPPEN
3130	004554	037045		DH4	
3131					
3132	004556	037072		DT4	
3133	004560	071155	ITE437:	EM437	;UNIBUS PE TRAPPED TO 0
3134	004562	037045		DH4	
3135	004564	037072		DT4	
3136					
3137	004566	071225	ITE440:	EM440	;WAIT INSTRUCTION FAILED
3138	004570	037045		DH4	
3139	004572	037072		DT4	
3140	004574	071300	ITE441:	EM441	;T BIT INTERRUPTED WAIT

3141	004576	037045							
3142	004600	037072			DH4				
3143	004602	071351	ITE442:	EM442	DT4				;PIRQ DID NOT INTERRUPT WAIT
3144	004604	037045			DH4				
3145	004606	037072			DT4				
3146	004610	071426	ITE443:	EM443	DH4				;STATE S13.00 FAILED
3147	004612	037045			DT4				
3148	004614	037072			DH4				
3149	004616	071444	ITE444:	EM444	DT4				;STATE S45.00 FAILED
3150	004620	037045			DH4				
3151	004622	037072			DT4				
3152	004624	071462	ITE445:	EM445	DH4				;STATE MTP.10 FAILED
3153	004626	037045			DT4				
3154	004630	037072			DH4				
3155	004632	071500	ITE446:	EM446	DT4				;STATE NEG.00 FAILED
3156	004634	037045			DH4				
3157	004636	037072			DT4				
3158	004640	071516	ITE447:	EM447	DH4				;STATE D45.00 FAILED
3159	004642	037045			DT4				
3160	004644	037072			DH4				
3161	004646	071534	ITE450:	EM450	DT4				;STATE D45.90 FAILED
3162	004650	037045			DH4				
3163	004652	037072			DT4				
3164	004654	071552	ITE451:	EM451	DH4				;STATE D45.00 FAILED
3165	004656	037045			DT4				
3166	004660	037072			DH4				
3167	004662	071570	ITE452:	EM452	DT4				;STATE D45.01 FAILED
3168	004664	037045			DH4				
3169	004666	037072			DT4				
3170	004670	071606	ITE453:	EM453	DH4				;RESERVED INSTRUCTION TRAP FAILED
3171	004672	037045			DT4				
3172	004674	037072			DH4				
3173	004676	071654	ITE454:	EM454	DT4				;TMCB PIRQ DOES NOT GO LOW
3174	004700	037045			DH4				
3175	004702	037072			DT4				
3176	004704	071741	ITE455:	EM455	DH4				;BEN06 FAILED
3177	004706	037045			DT4				
3178	004710	037072			DH4				
3179	004712	072057	ITE456:	EM456	DT4				;ODD ADDRESS FAILED ON DATI & DATO
3180	004714	037045			DH4				
3181	004716	037072			DT4				
3182	004720	072133	ITE457:	EM457	DH4				;PSW BIT 11 DOES NOT CLEAR
3183	004722	037045			DT4				
3184	004724	037072			DH4				
3185	004726	072164	ITE460:	EM460	DT4				;PSW CHANGED ON RESET
3186	004730	041440			DH42				
3187	004732	041516			DT42				
3188	004734	072230	ITE461:	EM461	DH42				;PSW CHANGES ON RESET IN SUPER MODE
3189	004736	041440			DT42				
3190	004740	041516			DH4				
3191	004742	012737	000014	177746	START: MOV	#14,@#CONTRL			;FORCE MISSES IN CACHE
3192	004750	005037	170200		CLR	@#MAPLO			;SETUP MAP 0 AND 1 TO PHYSICAL CORE
3193	004754	005037	170202		CLR	@#MAPHO			
3194	004760	012737	020000	170204	MOV	#20000,@#MAPL1			
3195	004766	005037	170206		CLR	@#MAPH1			
3196	004772	012737	000340	177776	MOV	#340,@#PS			::LOCK OUT ALL INTERRUPTS

```

3197 005000 012706 001100      MOV    #SCMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
3198 005004 005026              CLR    (R6)+          ;;CLEAR MEMORY LOCATION
3199 005006 022706 001136      CMP    #STKS,R6      ;;DONE?
3200 005012 001374              BNE    -6            ;;LOOP BACK IF NO
3201 005014 012706 001100      MOV    #STACK,SP     ;;SETUP THE STACK POINTER
3202 005020 012737 033252 000020  MOV    #SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
3203 005026 012737 000340 000022  MOV    #340,@#IOTVEC+2 ;;LEVEL 7
3204 005034 012737 033520 000030  MOV    #ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
3205 005042 012737 000340 000032  MOV    #340,@#EMTVEC+2 ;;LEVEL 7
3206 005050 012737 036216 000034  MOV    #STRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
3207 005056 012737 000340 000036  MOV    #340,@#TRAPVEC+2;LEVEL 7
3208 005064 012737 036250 000024  MOV    #SPWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
3209 005072 012737 000340 000026  MOV    #340,@#PWRVEC+2 ;;LEVEL 7
3210 005100 016767 025266 025256  MOV    #ENDCT,$EOPCT  ;;SETUP END-OF-PROGRAM COUNTER
3211 005106 005067 174060              CLR    $TIMES        ;;INITIALIZE NUMBER OF ITERATIONS
3212 005112 005067 174056              CLR    $ESCAPE       ;;CLEAR THE ESCAPE ON ERROR ADDRESS
3213 005116 112767 000001 173771  MOV    #1,$ERMAX      ;;ALLOW ONE ERROR PER TEST
3214 005124 012737 032622 000014  MOV    #SRTN,@#TBITVEC ;;SET 'T' BIT VECTOR TO SRTN
3215 005132 012737 000340 000016  MOV    #340,@#TBITVEC+2 ;;LEVEL 7
3216 005140 012767 000002 025454  MOV    #RTI,$RTRN     ;;SET $RTRN TO A RTI
3217 005146 012737 005174 000010  MOV    #65$,@#RESVEC  ;;TRY TO DO A RTT
3218 005154 005046              CLR    -(SP)         ;;DUMMY PS
3219 005156 012746 005164              MOV    #64$,-(SP)    ;;AND PC
3220 005162 000006              RTT                    ;;TRY THE RTT
3221 005164 012767 000006 025430 64$:  MOV    #RTT,$RTRN    ;;RTT IS LEGAL--SET $RTRN TO A RTT
3222 005172 000402              BR     66$
3223 005174 062706 000010 65$:  ADD    #10,SP        ;;RTT ILLEGAL--CLEAN OFF THE STACK
3224 005200 012737 000012 000010 66$:  MOV    #RESVEC+2,@#RESVEC ;;RESTORE TRAP CATCHER
3225 005206 005067 025416              CLR    $TBIT         ;;CLEAR 'T' BIT SWITCH
3226 005212 012767 005212 173666  MOV    #.,$LPADR      ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
3227 005220 012767 005220 173662  MOV    #.,$LPERR      ;;SETUP THE ERROR LOOP ADDRESS
3228
3229
3230 005226 005227 177777      ;;*****
3231 005232 001023              INC    #-1           ;;FIRST TIME?
3232 005234 022737 032554 000042  BNE    67$          ;;BRANCH IF NO
3233 005242 001417              CMP    #SENDAD,@#42  ;;ACT-11?
3234 005244 104400 005252              BEQ    67$          ;;BRANCH IF YES
3235 005250 000414              TYPE   ,68$         ;;TYPE ASCIZ STRING
3236
3237 005302      ;;68$: .ASCIZ <CRLF>"CEKBBFO 11/70 CPU #2"<CRLF>
3238 005302 005227 177777      67$:  INC    #-1           ;;FIRST TIME?
3239 005306 001000              BNE    100$         ;;BR IF NO
3240 005310
3241
3242      ;;*****
3243 005310 005737 000042      ;;SAVE THE MONITOR IF INITIAL LOAD
3244 005314 001003      LOOP: TST    @#42     ;;RUNNING UNDER XXDP CHAIN?
3245 005316 005227 177777      BNE    3$          ;;BRANCH IF YES
3246 005322 001010      INC    #-1         ;;INITIAL LOAD?
3247 005324 012700 002734 3$:  BNE    2$          ;;BRANCH IF NO
3248 005330 012701 073262      MOV    #^D1500,R0   ;;SETUP SOB COUNT
3249 005334 012702 160000      MOV    #SUBTAB+2,R1 ;;GET END ADDRESS OF PROGRAM
3250 005340 014221      MOV    #160000,R2   ;;GET END ADDRESS OF MONITOR
3251 005342 077002      1$:  MOV    -(R2),(R1)+  ;;SAVE 1500 DECIMAL WORDS
3252 005344 012737 034626 000060 2$:  SOB    R0,1$
      MOV    #QUIT,@#TKVEC ;;SETUP KEYBOARD VECTOR
    
```


3253 005352 012737 000340 000062
3254 005360 005077 173554
3255 005364 152777 000100 173544
3256 005372 012737 032636 000004
3257 005400 012737 000340 000006
3258 005406 012737 033042 000114
3259 005414 012737 000340 000116
3260 005422 005037 177766
3261 005426 012737 177777 177744
3262 005434 005767 173440
3263 005440 001402
3264 005442 005037 177746
3265
3266
3267
3268
3269
3270
3271
3272
3273
3274
3275
3276
3277
3278
3279
3280
3281
3282
3283
3284
3285
3286
3287
3288 005446 000004
3289 005450 012767 005630 173516
3290 005456 012767 005630 173604
3291 005464 012706 001100
3292 005470 005005
3293 005472 012737 005612 000010
3294 005500 012737 000113 177770
3295 005506 000235
3296 005510 013767 177776 173470
3297 005516 042767 177437 173462
3298 005524 022767 000240 173454
3299 005532 001401
3300 005534 005205
3301 005536 012737 000044 177770 2\$:
3302 005544 000240
3303 005546 000232
3304 005550 013767 177776 173404
3305 005556 042767 177437 173376
3306 005564 022767 000100 173370
3307 005572 001404
3308 005574 005705

MOV #PR7,@#TKVEC+2 ;SETUP KEYBOARD PSW
CLR @#TKB ;ENSURE BUFFER CLEAR
BISB #BIT6,@#TKS ;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(AFIR04(1)*AFIR05(0)*(RTS:CCOP))
* IS BAD.
* D30.00 WILL CAUSE A TRAP TO LOCATION 10 AFTER STATE D10.60.
* THIS FAILURE CAN BE DIFFERENTIATED FROM THE FIRST BY TESTING
* THE REGISTER ASSOCIATED WITH BITS <2:0> OF THE OP CODE TO
* SEE IF IT WAS INCREMENTED.
*
* IF BOTH LEVELS 2 AND 5 COME UP AS LEVEL 4, PDRD E31(1)
* COULD BE BAD.
*
* ONCE IT IS DETERMINED THAT THE INSTRUCTION WORKS A BIT TEST IS
* MADE ON THE PSW <7:5>.

* ROM FLOW-43,361

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
MOV #44,@#177770 ;SETUP MICROPROCESSOR BREAK REG
NOP ;SYNC INSTRUCTION
SPL 2 ;TEST TO SEE IF BITS 5&7 CLEAR & BIT 6 SETS
MOV @#PSW,\$TMP0 ;SAVE PSW
BIC #177437,\$TMP0 ;MASK OFF THE PRIORITY
CMP #PR2,\$TMP0 ;DID BIT 6 SET & 5&7 CLEAR?
BEQ 3\$;BRANCH IF YES
TST R5 ;DID SPL 5 FAIL?

```

3309 005576 001401          BEQ      4$          ;BRANCH IF NO
3310 005600 104001          ERROR    1          ;EITHER SPL DOES NOT LOAD PSW OR BITS STUCK
3311 005602 104002          4$:  ERROR    2          ;PR5 OK BUT PR2 BAD
3312 005604 005705          3$:  TST      R5          ;DID SPL 5 FAIL?
3313 005606 001410          BEQ      TST2        ;:BRANCH IF NO
3314 005610 104003          ERROR    3          ;PR2 OK BUT PR5 FAILED
3315 005612 012737 000012 000010 1$:  MOV      #12,@#RESVEC ;RESTORE RESERVEC VECTOR ADDRESS
3316 005620 005705          TST      R5          ;DID R5 GET INCREMENTED?
3317 005622 001401          BEQ      5$          ;BRANCH OF NO
3318 005624 104004          ERROR    4          ;FORK A FAILED TO D30.00
3319 005626 104005          5$:  ERROR    5          ;FORK A FAILED TO RSD.02
3320
3321 :*****
3322 :*TEST 2          RESET
3323 :*
3324 :*          IF FORK A FAILS EXECUTION WILL EITHER GO TO WAT.00 OR TRP.02.
3325 :*          THE LA30 PRINTER WILL BE STARTED SUCH THAT A FAILURE INTO
3326 :*          WAT.00 WILL RECOVER.
3327 :*
3328 :*          IF TRP.01 IS ENTERED A TRAP SEQUENCE WILL EXECUTE
3329 :*          WITH A TRAP VECTOR OF 4.
3330 :*
3331 :*          ROM FLOW-15,255,374
3332 :*****
3333 TST2: SCOPE
3334 005630 000004          MOV      #^D1980,$ICNT ;SET ITERATION COUNT
3335 005632 012767 003674 173244  MOV      #TST3,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
3336 005640 012767 006124 173326  MOV      #TST3,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3337 005646 012767 006124 173414  MOV      @#PSW,$TMP3    ;SAVE PSW
3338 005654 013767 177776 173306  MOV      #4$,$LPADR     ;SETUP LOOP ADR
3339 005662 012767 005712 173216  MOV      #4$,$LPERR     ;SETUP ERROR LOOP
3340 005670 012767 005712 173212  MOV      #2$,@#TPVEC    ;PUT ADDRESS OF 2$ IN PRINTER INTERRUPT VEC
3341 005676 012737 006114 000064  MOV      #1$,@#ERRVEC   ;PUT ADDRESS OF 1$ IN LOCATION 4
3342 005704 012737 006104 000004  4$:  MOV      #STACK,SP    ;INITIALIZE THE SP
3343 005712 012706 001100          ;TO CATCH A FAILURE TO THE TRP.02 STATE
3344 005716 000230          SPL      0            ;SET CPU AT 0
3345 005720 012777 000015 173216  MOV      #15,@$TPB      ;SEND A CR TO THE PRINTER
3346 005726 105777 173210  7$:  TSTB     @$TPS        ;WAIT FOR CR TO FINISH
3347 005732 100375          BPL      7$          ;INCASE TP DOUBLE BUFFERED
3348 005734 012777 000101 173202  MOV      #101,@$TPB     ;SEND AN "A"
3349 005742 052777 000100 173172  BIS      #BIT6,@$TPS    ;SET THE INTERRUPT FLAG
3350 005750 012737 034157 177776  MOV      #34157,@#PSW   ;SET AS MANY BITS AS POSSIBLE IN PSW
3351 005756 000240          NOP                    ;SYNC POINT
3352 005760 000005          RESET                 ;EXECUTE INSTRUCTION UNDER TEST
3353 005762 013767 177776 173216  MOV      @#PSW,$ERPSW   ;SAVE PSW
3354 005770 017700 173146          MOV      @$TPS,R0      ;GET THE PRINTER STATUS
3355 005774 042777 000100 173142  BIC      #BIT6,@$TPB    ;CLEAR INTERRUPT FLAG INCASE RESET FAILED
3356 006002 032700 000100          BIT      #BIT6,R0      ;DID INTERRUPT FLAG CLEAR?
3357 006006 001401          BEQ      3$          ;BRANCH IF YES
3358 006010 104006          ERROR    6          ;RESET DID NOT WORK
3359 006012 105777 173124  3$:  TSTB     @$TPS        ;WAIT FOR CHARACTER TO FINISH
3360 006016 100375          BPL      3$
3361 006020 012737 032636 000004  MOV      #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
3362 006026 012767 034157 173126  MOV      #34157,$TMP0   ;SAVE EXPECTED VALUE
3363 006034 032737 000020 177776  BIT      #BIT4,@#PSW    ;IS T BIT ON?
3364 006042 001407          BEQ      5$          ;BRANCH IF NO
3365 006044 012767 034177 173110  MOV      #34177,$TMP0   ;SAVE EXPECTED VALUE
    
```

```

3365 006052 022767 034177 173126      CMP      #34177,$ERPSW  ;DID PSW COME OUT OK?
3366 006060 000403                    BR        6$
3367 006062 022767 034157 173116 5$:    CMP      #34157,$ERPSW  ;DID PSW CHANGE?
3368 006070                    6$:
3369 006070 001415                    BEQ      TST3          ;;BRANCH IF NO
3370 006072 012767 034157 173062      MOV      #34157,$TMPO  ;SAVE EXPECTED VALUE
3371 006100 104377                    ERROR    377          ;PSW CHANGED ON RESET
3372 006102 000460                    460
3373 006104 042777 000100 173030 1$:    BIC      #BIT6,@$TPS   ;CLEAR PRINTER INTERRUPT FLAG
3374 006112 104007                    ERROR    7           ;FORK A FAILED INTO TRP.02
3375 006114 042777 000100 173020 2$:    BIC      #BIT6,@$TPS   ;CLEAR PRINTER INTERRUPT FLAG
3376 006122 104010                    ERROR    10          ;FORK A FAILED TO WAT.00
3377
3378
3379
3380
3381
3382
3383
3384
3385
3386
3387
3388
3389
3390
3391
3392 006124 000004
3393 006126 152777 000100 173002
3394 006134 012767 006222 172744
3395 006142 012767 006222 172740
3396 006150 013767 177776 173012
3397 006156 032737 000020 177776
3398 006164 001416
3399 006166 012746 000340
3400 006172 012746 006222
3401 006176 000006
3402 006200 012767 006406 172766
3403 006206 012767 006406 173054
3404 006214 012737 006356 000010
3405 006222 012706 001100 9$:
3406 006226 012746 100000
3407 006232 012766 100000 177776
3408 006240 012705 006276
3409 006244 000240
3410 006246 006401
3411 006250 022701 006250 8$:
3412 006254 001401
3413 006256 104011 5$:
3414 006260 013701 001074 3$:
3415 006264 100401
3416 006266 104012
3417 006270 012706 001076 4$:
3418 006274 104013
3419 006276 020627 006254 1$:
3420 006302 001410
    
```

 *TEST 3 MARK

FORK A CAN FAIL INTO ONE OF THE FOLLOWING:
 RSD.00, MFP.80, MTP.00, SVC.70, OR D67.01.
 STATE RSD.00 WILL CAUSE A TRAP TO LOCATION 10.
 MFP.80 WILL EXECUTE AN MFP INSTRUCTION.
 MTP.00 WILL EXECUTE AN MTP INSTRUCTION EXCEPT FOR THE ALU.
 SVC.70 WILL HANG THE PROCESSOR IN THE PAUSE CONDITION.
 THIS WILL ONLY HAPPEN IF RACF E8 IS BAD.
 D67.01 WILL STEP THE PCB AND TRAP TO LOCATION 10 AFTER STATE D10.60.

ROM FLOW-47,252,235,234

 TST3: SCOPE

```

BISB      #BIT6,@$TKS   ;RESTORE INTER FLAG AFTER RESET
MOV       #9$,$LPADR    ;SETUP LOOP ADDRESS
MOV       #9$,$LPERR    ;SETUP ERROR LOOP
MOV       @#PSW,$TMP3   ;SAVE PSW
BIT       #BIT4,@#PSW   ;IS T BIT ON?
BEQ       9$           ;BRANCH IF NO
MOV       #PR7,-(SP)    ;PUT NEW PSW ON STACK
MOV       #9$,-(SP)    ;PUT RETURN ADDR ON STACK
RTT
MOV       #TST4,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
MOV       #TST4,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
MOV       #2$,@#RESVEC  ;SETUP LOC 10 TO TRAP TO THIS ROUTINE
9$: MOV     #STACK,SP    ;INITIALIZE STACK
MOV       #BIT15,-(SP)  ;SET SIGN BIT ON STACK
MOV       #BIT15,-2(SP) ;SET SIGN BIT AT LOCATION 1074
MOV       #1$,R5       ;SETUP R5 FOR MARK INSTRUCTION
NOP
MARK      1           ;SYNC POINT
8$: CMP     #8$,R1     ;EXECUTE INSTRUCTION UNDER TEST
BEQ       3$         ;DID MTP OCCUR?
5$: ERROR  11         ;BRANCH IF NO
3$: MOV     @#1074,R1  ;FORK A FAILED TO MTP
BMI       4$         ;DID MFP OCCUR?
4$: ERROR  12         ;BRANCH IF NO
4$: MOV     #1076,SP   ;FORK A FAILED TO MFP
13: ERROR  13         ;INITIALIZE SP INCASE MARK CHANGED IT
1$: CMP     SP,#5$-2   ;PCB DID NOT LOAD FROM R5
BEQ       6$         ;DID SP GET LOADED PROPERLY?
6$:
    
```

```

3421 006304 010667 172644      MOV      SP,$REG0      ;SAVE SP FOR TYPEOUT
3422 006310 012767 006256 172640  MOV      #5,$REG1      ;STORE ADDRESS OF 5$ FOR TYPEOUT
3423 006316 012706 001100      MOV      #STACK,SP     ;REINITIALIZE SP
3424 006322 104014      ERROR    14           ;MARK DID NOT LOAD SP PROPERLY
3425 006324 012706 001100      6$: MOV      #STACK,SP     ;RESTORE THE SP
3426 006330 020527 006250      CMP      R5,#8$       ;DID R5 GET LOADED PROPERLY?
3427 006334 001424      BEQ      TST4         ;:BRANCH IF YES
3428 006336 010567 172612      MOV      R5,$REG0     ;SAVE R5 FOR TYPEOUT
3429 006342 013767 006254 172606  MOV      @#5$-2,$REG1 ;STORE ADDRESS OF 5$-2 FOR TYPEOUT
3430 006350 012706 001076      MOV      #1076,SP     ;RESTORE THE SP
3431 006354 104015      ERROR    15           ;R5 DID NOT LOAD PROPERLY
3432 006356 012737 000012 000010 2$: MOV      #12,@#RESVEC ;RESTORE RESERVED INSTR VECTOR
3433 006364 021627 006250      CMP      (SP),#8$     ;DID PCB GET INCREMENTED BY D67.01?
3434 006370 001003      BNE      7$          ;BRANCH IF NO
3435 006372 012706 001100      MOV      #STACK,SP     ;RESTORE THE SP
3436 006376 104016      ERROR    16           ;FORK A FAILED TO RSD.00
3437 006400 012706 001100      7$: MOV      #STACK,SP     ;RESTORE THE SP
3438 006404 104017      ERROR    17           ;FORK A FAILED TO D67.01
3439
3440      ;*****
3441      ;*TEST 4      ASH*DMO
3442      ;*
3443      ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
3444      ;*
3445      ;*      IF GRAJ SC05 L DOES NOT GO LOW OR DOES NOT GET THRU TO
3446      ;*      RACK BRCAB04 L A SHIFT RIGHT WILL OCCUR.
3447      ;*      IF THE SHIFT COUNTER DOES NOT SHIFT OR GRAJ SC=0 DOES NOT GO
3448      ;*      LOW THE PROCESSOR WILL HANG UP IN STATE ASH.41.
3449      ;*
3450      ;*      ROM FLOW-52,305,257,166 LEFT SHIFT
3451      ;*      52,305,277      RIGHT SHIFT
3452      ;*****
3452 006406 000004      TST4: SCOPE
3453 006410 012767 007150 172652  MOV      #TST5,NEXTTST ;SAVE ADDRESS OF NEXT TEST
3454 006416 012767 006462 172462  MOV      #19$,$LPADR   ;SETUP LOOP ADR
3455 006424 012767 006462 172456  MOV      #19$,$LPERR   ;SETUP ERROR LOOP
3456 006432 032767 000020 172530  BIT      #BIT4,$TMP3   ;WAS T BIT ON?
3457 006440 001405      BEQ      18$          ;BRANCH IF NO
3458 006442 012746 000360      MOV      #360,-(SP)    ;PUT NEW PSW ON STACK
3459 006446 012746 006454      MOV      #18$,-(SP)   ;PUT RETURN ADR ON STACK
3460 006452 000006      RTT
3461
3462      ;ARITHMETIC LEFT SHIFT
3463 006454 012737 007036 000010 18$: MOV      #1$,@#RESVEC ;SET UP FOR FORK A FAILURE
3464 006462 012706 001100      19$: MOV      #STACK,SP ;INITIALIZE THE SP
3465 006466 012700 177701      MOV      #177701,R0   ;PUT SHIFT COUNT IN R0 (+1)
3466 006472 012701 100000      MOV      #BIT15,R1    ;SET BIT 15 IN REGISTER TO BE SHIFTED
3467 006476 000240      NOP
3468 006500 072100      ASH      R0,R1        ;EXECUTE INSTRUCTION UNDER TEST
3469 006502 103414      BCS      2$          ;BRANCH IF SHIFT WORKED
3470 006504 020127 140000      CMP      R1,#140000   ;DID SHIFT GO IN WRONG DIRECTION?
3471 006510 001002      BNE      3$          ;BRANCH IF NO
3472 006512 104020      ERROR    20          ;SHIFTED RIGHT INSTEAD OF LEFT
3473 006514 000425      BR       9$
3474 006516 020127 100000      3$: CMP      R1,#BIT15 ;DID R1 SHIFT AT ALL?
3475 006522 001002      BNE      4$          ;BRANCH IF YES
3476 006524 104021      ERROR    21          ;R1 DID NOT SHIFT
    
```

```

3477 006526 000420          BR      9$
3478 006530 104022          4$:   ERROR  22          ;R1 SHIFTED BUT CARRY DID NOT SET
3479                                     ;SHIFT COUNTER COULD BE STUCK
3480 006532 000416          BR      9$
3481 006534 102003          2$:   BVC     10$          ;BRANCH IF V DID NOT SET
3482 006536 100402          BMI     10$          ;BRANCH IF N DID NOT CLEAR
3483 006540 001001          BNE     10$          ;BRANCH IF Z DID NOT SET
3484 006542 000412          BR      9$          ;CC'S OK
3485 006544 013767 177776 172434 10$:   MOV     @#PSW,$ERPSW ;SAVE PSW
3486 006552 042767 177760 172426 BIC     #177760,$ERPSW ;MASK OFF CC'S
3487 006560 012767 000007 172374 MOV     #7,$TMP0      ;PUT EXPECTED CC'S IN STORAGE
3488 006566 104031          ERROR   31          ;STATE ASH.40 DID NOT LOAD CC'S CORRECTLY
3489                                     ;*****
3490                                     ;ARITHMETIC RIGHT SHIFT TEST
3491 006570 012767 006576 172312 9$:   MOV     #64,$LPERR   ;SETUP ERROR LOOP
3492 006576 012701 100001          64$:   MOV     #100001,R1 ;SETUP R1 FOR RIGH SHIFT
3493 006602 012700 000077          MOV     #77,R0       ;PUT SHIFT COUNT IN RO (-1)
3494 006606 005002          CLR     R2           ;ENSURE R2 CLEAR
3495 006610 000240          NOP                    ;SYNC POINT
3496 006612 072100          ASH     RO,R1        ;EXECUTE RIGHT SHIFT
3497 006614 005502          ADL     R2           ;SAVE CARRY IN R2
3498 006616 020127 140000          CMP     R1,#140000   ;DID R1 SHIFT IN RIGHT DIRECTION?
3499 006622 001417          BEQ     5$           ;BRANCH IF YES
3500 006624 005701          TST     R1           ;DID R1 SHIFT LEFT INSTEAD OF RIGHT?
3501 006626 001002          BNE     6$           ;BRANCH IF NO
3502 006630 104023          ERROR   23          ;R1 SHIFTED WRONG DIRECTION
3503 006632 000416          BR      8$
3504 006634 022701 040000          6$:   CMP     #40000,R1   ;DID SHIFT FAIL TO SIGN FILL?
3505 006640 001001          BNE     7$           ;BRANCH IF NO
3506 006642 104024          ERROR   24          ;SHIFT RIGHT DID NOT SIGN FILL
3507 006644 010167 172306          7$:   MOV     R1,$REG1     ;SAVE R1 FOR TYPE OUT
3508 006650 012767 140000 172306 MOV     #140000,$TMP1 ;PUT EXPECTED VALUE IN STORAGE
3509 006656 104025          ERROR   25          ;R1 SHIFTED, BUT DON'T KNOW WHERE
3510 006660 000403          BR      8$
3511 006662 005702          5$:   TST     R2           ;DID CARRY GET SET ON SHIFT?
3512 006664 001001          BNE     8$           ;BRANCH IF YES
3513 006666 104026          ERROR   26          ;SHIFT OK BUT CARRY DID NOT LOAD
3514                                     ;*****
3515                                     ;CONDITION CODE LOAD TEST(STATE ASH.30)
3516 006670 012767 006676 172212 8$:   MOV     #63,$LPERR   ;SETUP ERROR LOOP
3517 006676 012701 100000          63$:   MOV     #BIT15,R1   ;SETUP RO TO TEST CC'S IN STATE ASH.30
3518                                     ;RO HAS SHIFT COUNT (-1)
3519 006702 000250          CLN                    ;SET UP CC'S TO
3520 006704 000266          +SEV!SEZ             ;COMPLIMENT OF EXPECTED RESULT
3521                                     ;AND OSCILLOSCOPE SYNC POINT
3522 006706 072100          ASH     RO,R1        ;EXECUTE THE SHIFT
3523 006710 013767 177776 172270 MOV     @#PSW,$ERPSW ;SAVE PSW
3524 006716 042767 177760 172262 BIC     #177760,$ERPSW ;MASK OFF THE CC'S
3525 006724 022767 000010 172254 CMP     #10,$ERPSW   ;DID CC'S COME OUT OK?
3526 006732 001404          BEQ     12$          ;BRANCH IF YES
3527 006734 012767 000010 172220 MOV     #10,$TMP0     ;PUT EXPECTED VALUE IN STORAGE
3528 006742 104072          ERROR   72          ;STATE ASH.30 DID NOT LOAD CC'S CORRECTLY
3529                                     ;*****
3530                                     ;SHIFT COUNT=0
3531 006744 012767 006752 172136 12$:   MOV     #62,$LPERR   ;SETUP ERROR LOOP
3532 006752 012701 100000          62$:   MOV     #BIT15,R1   ;SET SIGN BIT IN REGISTER TO BE SHIFTED
    
```

```

3533 006756 005000          CLR    R0          ;SET SHIFT COUNT TO ZERO
3534 006760 000263          +SEC!SEV         ;SETUP CC'S TO COMPLIMENT
3535 006762 000250          CLN             ;OF EXPECTED VALUE
3536                          ;AND OSCILLOSCOPE SYNC POINT
3537 006764 072100          ASH    R0,R1     ;EXECUTE INSTRUCTION
3538 006766 013767 177776 172212  MOV    @#PSW,$ERPSW ;SAVE PSW
3539 006774 042767 177760 172204  BIC    #177760,$ERPSW ;MASK OFF THE CC'S
3540 007002 022767 000010 172176  CMP    #10,$ERPSW   ;DID CC'S COME OUT OK?
3541 007010 001416          BEQ    16$       ;BRANCH IF YES
3542 007012 022701 100000      13$:  CMP    #BIT15,R1   ;DID R1 SHIFT?
3543 007016 001005          BNE    15$       ;BRANCH IF YES
3544 007020 012767 000010 172134  MOV    #10,$TMP0   ;SAVE EXPECTED VALUE
3545 007026 104027          ERROR  27       ;STATE ASH.20 DID NOT LOAD CC'S CORRECTLY
3546 007030 000406          BR     16$
3547 007032 104030      15$:  ERROR  30         ;R1 SHIFTED WHEN NOT SUPPOSE TO
3548 007034 000404          BR     16$
3549 007036 012737 000012 000010  1$:  MOV    #12,@#RESVEC ;RESTORE RESERVED VECTOR
3550 007044 104032          ERROR  32       ;FORK A FAILED
3551                          ;*****
3552                          ;CONDITION CODE TEST OF STATE ASH.41
3553 007046 012767 007054 172034  16$:  MOV    #61$,$LPERR ;SETUP ERROR LOOP
3554 007054 012700 000002      61$:  MOV    #2,R0       ;PUT SHIFT COUNT IN R0
3555                          ;TO TEST STATE ASH.41
3556 007060 012701 060000          MOV    #60000,R1  ;SETUP R1 FOR SHIFT LEFT
3557 007064 000274          +SEN!SEZ       ;SETUP CC'S TO COMPLEMENT
3558 007066 000243          +CLV!CLC       ;OF EXPECTED VALUE
3559                          ;AND OSCILLOSCOPE SYNC POINT
3560 007070 072100          ASH    R0,R1     ;EXECUTE INSTRUCTION UNDER TEST
3561 007072 013767 177776 172106  MOV    @#PSW,$ERPSW ;SAVE PSW
3562 007100 020127 100000          CMP    R1,#BIT15  ;DID R1 SHIFT CORRECTLY?
3563 007104 001406          BEQ    17$       ;BRANCH IF YES
3564 007106 010167 172044          MOV    R1,$REG1   ;SAVE R1 FOR TYPEOUT
3565 007112 012767 100000 172044  MOV    #BIT15,$TMP1 ;STORE EXPECTED VALUE
3566 007120 104073          ERROR  73       ;STATE ASH.41 FAILED
3567 007122 042767 177760 172056  17$:  BIC    #177760,$ERPSW ;MASK OFF CC'S
3568 007130 022767 000013 172050  CMP    #13,$ERPSW  ;DID CC'S LOAD CORRECTLY?
3569 007136 001404          BEQ    TST5      ;BRANCH IF YES
3570 007140 012767 000013 172014  MOV    #13,$TMP0   ;STORE EXPECTED VALUE
3571 007146 104074          ERROR  74       ;BAD CC'S DUE TO ASH.41
3572                          ;*****
3573                          ;*TEST 5          ASH*DM1
3574                          ;*
3575                          ;* IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
3576                          ;*
3577                          ;* IF FORK B FAILS EXECUTION WILL EITHER GO TO RSD.00 OR
3578                          ;* MUL.00.
3579                          ;* MUL.00 WILL ONLY BE ENTERED IF EITHER B FORK MUX INPUT B2
3580                          ;* DOES NOT GO HIGH OR THE MUX IS BAD.
3581                          ;*
3582                          ;* ROM FLOW-1,175,62,52,305,257
3583                          ;*****
3584 007150 000004          TST5:  SCOPE
3585 007152 012767 007264 172014  MOV    #TST6,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
3586 007160 012767 007264 172102  MOV    #TST6,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3587 007166 012706 001100          MOV    #STACK,SP   ;INITIALIZE THE STACK
3588 007172 012737 007232 000010  MOV    #1$,@#RESVEC ;SETUP RESERVED VECTOR
    
```

```

3589 007200 012700 001164      MOV    #STMP1,R0      ;PUT ADDRESS OF SHIFT COUNT IN R0
3590 007204 012710 000001      MOV    #1,(R0)       ;SET SHIFT COUNT TO ONE
3591 007210 012701 100000      MOV    #BIT15,R1     ;SETUP REGISTER TO BE SHIFTED
3592 007214 000240              NOP                  ;OSCILLOSCOPE SYNC POINT
3593 007216 072110              ASH    (R0),R1       ;EXECUTE INSTRUCTION UNDER TEST
3594 007220 103421              BCS    TST6          ;:TEST OK, GO TO NEXT TEST
3595 007222 005701              TST    R1            ;DID MULTIPLY OCCUR?
3596 007224 100401              BMI    2$           ;BRANCH IF YES
3597 007226 104033              ERROR  33           ;STATE ASH.00 FAILED
3598 007230 104034              ERROR  34           ;FORK B FAILED INTO MUL.00
3599 007232 012737 007246 000010 1$:  MOV    #3$,@#RESVEC ;SETUP RESVEC
3600 007240 005000              CLR    R0            ;PUT EVEN ADDRESS IN R0
3601 007242 000240              NOP                  ;OSCILLOSCOPE SYNC POINT
3602 007244 072120              ASH    (R0)+,R1     ;EXECUTE DM2
3603 007246 012737 000012 000010 3$:  MOV    #12,@#RESVEC ;RESTORE RESERVED VECTOR
3604 007254 005700              TST    R0            ;DID R0 AUTO INCREMENT?
3605 007256 001401              BEQ    4$           ;BRANCH IF NO
3606 007260 104035              ERROR  35           ;FORK B FAILED
3607 007262 104036              ERROR  36           ;FORK A FAILED
3608
3609      ;*****
3610      ;*TEST 6      ASH*DM2
3611      ;*
3612      ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
3613      ;*
3614      ;*      ALL OTHER LOGIC HAS BEEN TESTED
3615      ;*
3616      ;*      ROM FLOW-2,175,62,52,305
3617      ;*****
3617 007264 000004      TST6:  SCOPE
3618 007266 012767 007344 171700      MOV    #TST7,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
3619 007274 012767 007344 171766      MOV    #TST7,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3620 007302 012706 001100              MOV    #STACK,SP    ;INITIALIZE THE SP
3621 007306 012737 007334 000010      MOV    #1$,@#RESVEC ;PUT ADDRESS OF 1$ IN RESERVED VECTOR
3622 007314 012700 001164              MOV    #STMP1,R0    ;PUT ADDRESS OF SHIFT COUNT IN R0
3623 007320 005010              CLR    (R0)         ;PUT SHIFT COUNT OF ZERO IN $TMP1
3624 007322 072120              ASH    (R0)+,R1     ;EXECUTE INSTRUCTION UNDER TEST
3625 007324 012737 000012 000010      MOV    #12,@#RESVEC ;RESTORE RESVEC
3626 007332 000404              BR     TST7         ;:GO TO NEXT TEST
3627 007334 012737 000012 000010 1$:  MOV    #12,@#RESVEC ;RESTORE RESERVED VECTOR
3628 007342 104037              ERROR  37           ;FORK A FAILED
3629
3630      ;*****
3631      ;*TEST 7      ASH*DM4
3632      ;*
3633      ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
3634      ;*
3635      ;*      ALL OTHER LOGIC HAS BEEN TESTED.
3636      ;*
3637      ;*      ROM FLOW-4,122,177,62,52,305
3638      ;*****
3638 007344 000004      TST7:  SCOPE
3639 007346 012767 007430 171620      MOV    #TST10,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
3640 007354 012767 007430 171706      MOV    #TST10,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3641 007362 012706 001100              MOV    #STACK,SP    ;INITIALIZE THE SP
3642 007366 012737 007420 000010      MOV    #1$,@#RESVEC ;PUT ADDRESS OF 1$ IN RESERVED VECTOR
3643 007374 012700 001166              MOV    #STMP2,R0    ;PUT ADDRESS OF SHIFT COUNT+2 IN R0
3644 007400 005060 177776              CLR    -2(R0)       ;PUT SHIFT COUNT IN $TMP1
    
```



```

3757 010100 012702 000052      MOV      #52,R2      ;PUT SHIFT COUNT (-26) IN R2
3758 010104 005001      CLR      R1          ;ENSURE R0 CLEAR
3759 010106 005067 171050      CLR      $TMP0      ;PUT EXPECTED VALUES OF
3760 010112 005067 171046      CLR      $TMP1      ;R0 & R1 IN STORAGE
3761 010116 012767 000001 171042      MOV      #1,$TMP2    ;C BIT EXPECTED
3762 010124 000240      NOP                    ;OSCILLOSCOPE SYNC POINT
3763 010126 073002      ASHC     R2,R0        ;EXECUTE SHIFT
3764 010130 013767 177776 171050      MOV      @#PSW,$ERPSW ;SAVE PSW
3765 010136 103410      BCS     16$          ;BRANCH IF CORRECT SHIFT
3766 010140 042767 177776 171040      BIC      #177776,$ERPSW ;MASK OFF C BIT
3767 010146 010067 171002      MOV      R0,$REG0
3768 010152 010167 171000      MOV      R1,$REG1
3769 010156 104051      ERROR   51          ;STUCK BITS IN SC
3770 010160 012767 010166 170722 16$:      MOV      #60$,$LPERR ;SETUP ERROR LOOP
3771 010166 012701 004000 60$:      MOV      #4000,R1    ;SET UP R1 FOR LEFT SHIFT
3772 010172 005000      CLR      R0          ;ENSURE R0 CLEAR
3773 010174 012702 000025      MOV      #25,R2      ;PUT SHIFT COUNT (+25) IN R2
3774 010200 000240      NOP                    ;OSCILLOSCOPE SYNC POINT
3775 010202 073002      ASHC     R2,R0        ;EXECUTE SHIFT
3776 010204 013767 177776 170774      MOV      @#PSW,$ERPSW ;SAVE PSW
3777 010212 042767 177776 170766      BIC      #177776,$ERPSW ;MASK OFF C BIT
3778 010220 103414      BCS     TST11        ;GO TO NEXT TEST IF CORRECT SHIFT
3779 010222 012767 000000 170724      MOV      #R0,$REG0   ;SAVE R0 FOR TYPEOUT
3780 010230 012767 000001 170720      MOV      #R1,$REG1   ;SAVE R1 FOR TYPEOUT
3781 010236 005067 170720      CLR      $TMP0      ;SAVE EXPECTED
3782 010242 012767 000001 170714      MOV      #1,$TMP1    ;VALUES
3783 010250 104052      ERROR   52          ;STUCK BITS IN SC
3784
3785 *****
3786 :*TEST 11      ASHC*DM1
3787 :*
3788 :*      THE ONLY POSSIBLE FAILURES ARE FORK B OR STATE ASC.00.
3789 :*
3790 :*      IF FORK B FAILS EXECUTION WILL EITHER GO TO RSD.00 OR
3791 :*      ASH.00.
3792 :*      ASH.00 WOULD PERFORM AN ASH INSTRUCTION INSTEAD OF AN ASHC.
3793 :*
3794 :*      ROM FLOW-1,175,63,53,306,267,227
3795 *****
3796 TST11: SCOPE
3796 010252 000004      MOV      #TST12,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
3797 010254 012767 010412 170712      MOV      #TST12,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3798 010262 012767 010412 171000      MOV      #STACK,SP    ;INITIALIZE THE SP
3799 010270 012706 001100      MOV      #1$,@#RESVEC ;SETUP RESERVED VECTOR
3800 010274 012737 010402 000010      MOV      #1$,@#RESVEC ;SETUP RESERVED VECTOR
3800 010302 012702 001166      MOV      #1$,@#RESVEC ;SETUP RESERVED VECTOR
3801 010306 012700 000001      MOV      #1,$R0       ;SETUP R0 FOR RIGHT SHIFT
3802 010312 005001      CLR      R1          ;SETUP R1 FOR RIGHT SHIFT
3803 010314 012712 000077      MOV      #77,(R2)     ;PUT SHIFT COUNT (-1) IN $TMP2
3804 010320 005067 170636      CLR      $TMP0      ;PUT EXPECTED VALUES OF
3805 010324 012767 100000 170632      MOV      #BIT15,$TMP1 ;R0 & R1 IN STORAGE
3806 010332 000240      NOP                    ;OSCILLOSCOPE SYNC POINT
3807 010334 073012      ASHC     (R2),R0      ;EXECUTE INSTRUCTION UNDER TEST
3808 010336 103012      BCC     2$          ;BRANCH IF FORK B DID NOT FAIL
3809 010340 013767 177776 170640      MOV      @#PSW,$ERPSW ;SAVE PSW FOR TYPEOUT
3810 010346 005067 170610      CLR      $TMP0      ;SAVE EXPECTED VALUE
3811 010352 010067 170576      MOV      R0,$REG0
3812 010356 010167 170574      MOV      R1,$REG1
    
```

```

3813 010362 104053          ERROR 53          ;FORK B FAILED TO ASH.00
3814 010364 005701          2$:  TST  R1          ;DID R1 SIGN BIT SET?
3815 010366 100411          BMI  TST12         ;:BRANCH IF YES
3816 010370 010067 170560    MOV  R0,$REG0      ;SAVE R0 & R1
3817 010374 010167 170556    MOV  R1,$REG1      ;FOR TYPEOUT
3818 010400 104054          ERROR 54          ;STATE ASC.00 FAILED
3819 010402 012737 000012 000010 1$:  MOV  #12,@#RESVEC  ;RESTORE RESVEC
3820 010410 104055          ERROR 55          ;FORK A FAILED TO RSD.00
3821
3822          ;*****
3823          ;TEST 12      MUL*DMO
3824          ;
3825          ;   FORK A SHOULD NOT FAIL.
3826          ;   THE FOLLOWING WOULD BE BEN11 FAILURES:
3827          ;   IF EITHER GRAD DROO IS STUCK HIGH OR NOT GETTING THRU TO
3828          ;   RACK E64(C1) OR RACK E64 IS BAD (INPUT C1 FAILED HIGH)
3829          ;   THE MULTIPICAND WILL BE MULTITPLIED BY 177777.
3830          ;   IF EITHER GRAD DROO IS STUCK LOW OR NOT GETTING THRU TO
3831          ;   RACK E64(C1) OR RACK E64 IS BAD (INPUT C1 FAILED LOW)
3832          ;   THE MULTIPLICAND WILL BE MULTIPLIED BY ZERO.
3833          ;   IF GRAJ SC=0 IS NOT GETTING TO RACK E50(C1) AND RACK E50
3834          ;   IS BAD (INPUT C1 FAILED LOW) THE MULTIPLICAND WILL ONLY
3835          ;   BE MULTIPLIED BY BIT 0 OF THE MULTIPLIER.
3836          ;   IF RACK E50(C1) FAILS HIGH THE PROCESSOR WILL HANG UP IN
3837          ;   STATE MUL.20(266).
3838          ;   IF THE INSTRUCTION FAILS TO MULTIPLY CORRECTLY AND ONE
3839          ;   OF THE ABOVE CONDITIONS CANNOT BE DETERMINED THEN THE
3840          ;   FAILURE COULD BE IN THE INSTRUCTION DECODE ROM.
3841          ;
3842          ;   IF THE CC'S COME UP BAD THEN THE FAILURE COULD BE IN
3843          ;   STATES MUL.40,50, OR 60, OR IN THE CONDITION CODE ROM.
3844          ;
3845          ;   ROM FLOW-50,102,266/246,226/206,310
3846          ;*****
3846 010412 000004          TST12: SCOPE
3847 010414 012767 010754 170646    MOV  #TST13,NEXTTST ;SAVE ADDRESS OF NEXT TEST
3848
3849          ;MULTIPLY 3 TIMES 2
3850 010422 012700 000002          MOV  #2,R0          ;PUT MULTIPLICAND IN R0
3851 010426 012702 000003          MOV  #3,R2          ;PUT MULTIPLIER IN R2
3852 010432 005001          CLR  R1             ;ENSURE R1 CLEAR
3853 010434 005067 170522          CLR  $TMP0          ;PUT EXPECTED VALUE OF
3854 010440 012767 000006 170516    MOV  #6,$TMP1       ;R0 & R1 IN STORAGE
3855 010446 000277          SCC                ;MULTIPLY SHOULD CLEAR ALL OF THEM
3856          ;AND OSCILLOSCOPE SYNC POINT
3857 010450 070002          MUL  R2,R0          ;EXECUTE INSTRUCTION UNDER TEST
3858 010452 013767 177776 170526    MOV  @#PSW,$ERPSW  ;SAVE PSW
3859 010460 010067 170470          MOV  R0,$REG0      ;SAVE R0 & R1
3860 010464 010167 170466          MOV  R1,$REG1      ;FOR TYPEOUT
3861 010470 020127 000006          CMP  R1,#6          ;DID R1 GET LOW PRODUCT
3862 010474 001002          BNE  1$            ;BRANCH IF NO
3863 010476 005700          TST  R0             ;DID R0 GET UPPER PRODUCT?
3864 010500 001423          BEQ  2$            ;BRANCH IF YES
3865 010502 020127 177776          1$:  CMP  R1,#177776  ;DID R0 GET MULTIPLIED BY 177777?
3866 010506 001002          BNE  3$            ;BRANCH IF NO
3867 010510 104056          ERROR 56          ;EITHER GRAD DROO STUCK H OR RACK E64 BAD
3868 010512 000430          BR   8$
    
```

```

3869 010514 005701      3$:   TST      R1          ;DID R1 STAY ZERO?
3870 010516 001401      BEQ      4$          ;BRANCH IF YES
3871 010520 000404      BR       5$          ;CONTINUE ERROR ANALYSIS
3872 010522 005700      4$:   TST      R0          ;DID R1 & R0 GO TO ZERO?
3873 010524 001002      BNE     5$          ;BRANCH IF NO
3874 010526 104057      ERROR   57          ;EITHER GRAD DROO STUCK L OR RACK E64 BAD
3875 010530 000421      BR      8$
3876 010532 020127 100000  5$:   CMP      R1,#BIT15    ;DID R0 ONLY GET MULTIPLIED BY BIT 0?
3877 010536 001002      BNE     6$          ;BRANCH IF NO
3878 010540 104060      ERROR   60          ;RACH E50(C1) FAILED LOW
3879 010542 000414      BR      8$
3880 010544 104061      6$:   ERROR   61          ;CAN'T DETERMINE WHAT HAPPENED
3881 010546 000412      BR      8$
3882 010550 042767 177760 170430 2$:   BIC     #177760,$ERPSW ;MASK OFF THE CC'S
3883 010556 022767 000000 170422  CMP     #0,$ERPSW
3884 010564 001403      BEQ     8$          ;BRANCH IF YES
3885 010566 005067 170370  7$:   CLR     $TMPO        ;PUT EXPECTED PSW IN $TMO
3886 010572 104062      ERROR   62          ;BAD CC'S
3887
3888 ;:*****
3889 ;THE FOLLOWING SECTION TESTS STATE MUL.50 (-1 TIMES 3)
3889 010574 012767 010602 170306 8$:   MOV     #64,$LPERR    ;SETUP ERROR LOOP
3890 010602 012700 177777  64$:  MOV     #177777,R0    ;PUT -1 (MULTIPLICAND) IN R0
3891 ;:
3892 010606 000277      SCC
3893 010610 000250      CLN
3894 ;:
3895 010612 070002      MUL     R2,R0        ;AND OSCILLOSCOPE SYNC POINT
3896 010614 013767 177776 170364  MOV     @#PSW,$ERPSW  ;EXECUTE INSTRUCTION UNDER TEST
3897 010622 010067 170326  MOV     R0,$REG0     ;SAVE PSW
3898 010626 010167 170324  MOV     R1,$REG1     ;SAVE R0 & R1
3899 010632 012767 177777 170322  MOV     #-1,$TMPO    ;FOR TYPEOUT
3900 010640 012767 177775 170316  MOV     #-3,$TMP1    ;SAVE EXPECTED
3901 010646 020027 177777  MOV     #0,#177777   ;VALUES
3902 010652 001402      CMP     R0,#177777   ;DID R0 LOAD CORRECTLY?
3903 010654 104063      BEQ     9$          ;BRANCH IF YES
3904 010656 000420      ERROR   63          ;R0 DID NOT LOAD CORRECTLY
3905 010660 020127 177775  9$:   BR      12$
3906 010664 001402      CMP     R1,#177775   ;DID R1 LOAD PROPERLY?
3907 010666 104064      BEQ     10$         ;BRANCH IF YES
3908 010670 000413      ERROR   64          ;R1 DID NOT LOAD CORRECTLY.
3909 010672 042767 177760 170306 10$:  BR      12$
3910 010700 022767 000010 170300  BIC     #177760,$ERPSW ;MASK OFF THE CC'S
3911 010706 001404      CMP     #10,$ERPSW  ;DID THE CC'S LOAD PROPERLY?
3912 010710 012767 000010 170244 11$:  BEQ     12$         ;BRANCH IF YES
3913 010716 104065      MOV     #10,$TMPO   ;PUT EXPECTED PSW IN $TMO
3914 ;:
3915 ;:*****
3916 ;THE FOLLOWING VERIFIES THAT C SETS IF MULTIPLICATION OVERFLOWS OR UNDER FLOWS
3916 010720 012767 010726 170162 12$:  MOV     #63,$LPERR    ;SETUP ERROR LOOP
3917 010726 012700 077777  63$:  MOV     #77777,R0    ;PUT LARGEST POSITIVE NO. IN R0
3918 ;:
3919 010732 000240      NOP
3920 010734 070002      MUL     R2,R0        ;OSCILLOSCOPE SYNC POINT
3921 010736 103401      BCS     13$         ;EXECUTE INSTRUCTION
3922 010740 104066      ERROR   66          ;BRANCH IF C SET
3923 010742 012700 100000 13$:  MOV     #BIT15,R0    ;C DID NOT SET ON OVERFLOW
3924 ;:

```

```

3925 010746 070002          MUL      R2,R0          ;EXECUTE INSTRUCTION
3926 010750 103401          BCS     TST13          ;;BRANCH IF C SET
3927 010752 104067          ERROR   67            ;C DID NOT SET ON UNDERFLOW
3928
3929
3930
3931
3932
3933
3934
3935
3936
3937
3938
    ;*****
    ;*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
    ;*****
    
```

```

3939 010754 000004          TST13: SCOPE
3940 010756 012767 011070 170210      MOV     #TST14,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
3941 010764 012767 011070 170276      MOV     #TST14,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
3942 010772 012706 001100          MOV     #STACK,SP      ;INITIALIZE THE SP
3943 010776 012737 011060 000010      MOV     #1$,@#RESVEC   ;SETUP RESVEC
3944 011004 012701 000002          MOV     #2,R1          ;PUT MULTIPLICAND IN R1
3945 011010 012702 001166          MOV     #STMP2,R2      ;PUT ADDRESS OF MULTIPLIER IN R2
3946 011014 012712 000002          MOV     #2,(R2)        ;PUT MULTIPLIER (+2) IN $TMP2
3947 011020 005000          CLR     R0             ;ENSURE R0 CLEAR
3948 011022 005067 170134          CLR     $TMP0         ;STORE EXPECTED VALUE
3949 011026 012767 000004 170130      MOV     #4,$TMP1      ;OF R0 & R1
3950 011034 000240          NOP                   ;OSCILLOSCOPE SYNC POINT
3951 011036 070112          MUL     (R2),R1        ;EXECUTE INSTRUCTION UNDER TEST
3952 011040 020127 000004          CMP     R1,#4          ;DID MULTIPLY WORK?
3953 011044 001411          BEQ     TST14          ;;BRANCH IF YES
3954 011046 010067 170102          MOV     R0,$REG0      ;SAVE R0 & R1
3955 011052 010167 170100          MOV     R1,$REG1      ;FOR TYPEOUT
3956 011056 104070          ERROR   70            ;STATE MUL.00 FAILED
3957 011060 012737 000012 000010 1$: MOV     #12,@#RESVEC   ;RESTORE RESVEC
3958 011066 104071          ERROR   71            ;FORK B FAILED
3959
    
```

```

3960
3961
3962
3963
3964
3965
3966
3967
3968
3969
3970
3971
3972
3973
3974
3975
3976
3977
3978
3979
3980
    ;*****
    ;*TEST 14          DIV*DMO
    ;*
    ;*   FORK A SHOULD NOT FAIL.
    ;*   SECTION 1
    ;*   THE FIRST SECTION HAS A ZERO DIVISOR. IF STATE DIV.00
    ;*   FAILS OR IRCF Z2(1) DOES NOT GET TO RACK E49 OR RACK
    ;*   E49(B1) IS STUCK LOW EXECUTION WILL GO FROM DIV.10 TO
    ;*   DIV.20. THIS WILL CAUSE THE DIVIDE TO ABORT (GO TO DVE.20)
    ;*   AFTER STATE DIV.60 AND THE C BIT WILL BE CLEAR.
    ;*   SECTION 2
    ;*   THE NEXT SECTION DIVIDES 4 BY 2. IF RACK E49(B1) IS STUCK
    ;*   HIGH THE ALGORITHM WILL ABORT THINKING THAT THE DIVISOR
    ;*   IS ZERO. IF BEN04 FAILS THE DIVIDE WILL ABORT THINKING
    ;*   THAT THE DIVIDEND IS THE MOST NEGATIVE NUMBER.
    ;*   IF BEN16 FAILS THE DIVIDE WILL COMPLETE BUT R1 WILL CONTAIN
    ;*   155776. IF BEN05 FAILS THE ALGORITHM WILL ABORT THRU STATE
    ;*   DVE.20. IF BEN04 (AFTER DIV.70) FAILS R0 WILL END UP WITH 177777
    ;*   AND R1 WILL HAVE 177774. IF BEN03 FAILS R0 WILL END UP WITH
    ;*   20 AND R1 WILL HAVE 177774. IF BEN16 FAILS AFTER DVC.00 R0
    ;*
    ;*****
    
```

```

3981 : * WILL HAVE 2 BUT R1 WILL HAVE 177776.
3982 : *
3983 : * SECTION 3
3984 : * THE NEXT SECTION DIVIDES 6 BY 2 TO TEST BEN16*DR0(1).
3985 : * A FAILURE WILL LEAVE THE REMAINDER (R1)=2 INSTEAD OF ZERO.
3986 : *
3987 : * SECTION 4
3988 : * THE NEXT SECTION DIVIDES 4 BY -2 TO TEST BEN15*SR15(1).
3989 : * IF THIS FAILS R0 WILL CONTAIN 27777 AND R1 WILL CONTAIN 4.
3990 : *
3991 : * SECTION 5
3992 : * THE NEXT SECTION DIVIDES 1177777 BY 1 TO TEST BEN05*DIV QUIT.
3993 : * IF THIS FAILS R0 WILL CONTAIN 177777.
3994 : *
3995 : * SECTION 6
3996 : * THE NEXT SECTION DIVIDES 1000000 B2 -2 TO TEST BEN05*DIV QUIT.
3997 : * THIS SECTION WILL ONLY FAIL IF GRAJ E5 (Z2(0)*LEFT SAVE (1)) IS BAD.
3998 : *
3999 : * SECTION 7
4000 : * THE NEXT SECTION DIVIDES 100000000000 BY 2 TO TEST
4001 : * BEN04*NEGATIVE DIVIDEND.
4002 : *
4003 : * SECTION 8
4004 : * THE NEXT SECTION DIVIDED 177776 177777 BY -1 TO TEST BEN05*DIV QUIT.
4005 : * THIS TEST WILL ONLY FAIL IF GRAJ E5(N(1)*SR15(1)) IS BAD.
4006 : *
4007 : * SECTION 9
4008 : * THE NEXT SECTION DIVIDES -5 BY 2 TO ENSURE THAT THE REMAINDER IS
4009 : * STORED AS A NEGATIVE NUMBER.
4010 : *
4011 : * SECTION 10
4012 : * THE NEXT SECTION DIVIDES -5 BY -2 TO TEST STATES DVC.20,DVC.40, & DVC.60
4013 : *
4014 : * SECTION 11
4015 : * THE NEXT SECTION DIVIDES -2**16 BY 2**14 TO TEST STATE DVN.20
4016 : *
4017 : * SECTION 12
4018 : * THE NEXT SECTION DIVIDES 100 000200 BY -177 TO TEST STATES
4019 : * DVD.00 AND DVD.10.
4020 : *
    
```

```

4021 011070 000004
4022 011072 012767 012550 170170
4023
4024 011100 005000
4025 011102 012701 000004
4026 011106 005002
4027 011110 000270
4028
4029 011112 071002
4030 011114 013767 177776 170064
4031 011122 042767 177760 170056
4032 011130 022767 000007 170050
4033 011136 001412
4034 011140 022767 000002 170040
4035 011146 001002
4036 011150 104075
    
```

```

*****
TST14: SCOPE
MOV #TST15,NEXTTST ;SAVE ADDRESS OF NEXT TEST
;DIVISOR=0 SECTION (ALSO TESTS CONDITION CODE ROM FOR DIV)
CLR R0 ;PUT DIVIDEND IN
MOV #4,R1 ;R0 AND R1
CLR R2 ;MAKE DIVISOR=0
SEN ;ENSURE N SET
;AND OSCILLOSCOPE SYNC POINT
DIV R2,R0 ;EXECUTE INSTRUCTION UNDER TEST
MOV @#PSW,$ERPSW ;SAVE PSW
BIC #177760,$ERPSW ;MASK OFF THE CC'S
CMP #7,$ERPSW ;DID THE CC'S COME OUT OK?
BEQ 2$ ;BRANCH IF YES
CMP #2,$ERPSW ;DID INSTR GO THROUGH DVE.20?
BNE 3$ ;BRANCH IF NO
ERROR 75 ;BEN02 FAILED
    
```

```

4037 011152 000404          BR      2$
4038 011154 012767 000007 170000 3$:  MOV     #7,$TMP0      ;SAVE EXPECTED VALUE
4039 011162 104076          ERROR   76            ;CC'S BAD IN STATE DVE.00
4040                                     ;*****
4041                                     ;SECTION TWO (ALSO CHECKS CC LOAD OF STATE DVC.70)
4042 011164 012767 011172 167716 2$:  MOV     #64,$LPERR    ;SETUP ERROR LOOP
4043 011172 012702 000002          64$:  MOV     #2,R2        ;PUT DIVISOR IN R2 (R0 & R1 HOLD DIVIDEND)
4044 011176 000277          SCC
4045                                     ;ENSURE CC'S SET
4046 011200 071002          DIV     R2,R0        ;AND OSCILLOSCOPE SYNC POINT
4047 011202 013767 177776 167776  MOV     @#PSW,$ERPSW  ;EXECUTE INSTRUCTION UNDER TEST
4048 011210 020027 000002          CMP     R0,#2        ;SAVE PSW
4049 011214 001042          BNE     4$          ;DID QUOTIENT COME OUT CORRECT?
4050 011216 005701          TST    R1           ;BRANCH IF NO
4051 011220 001013          BNE     5$          ;DID REMAINDER COME OUT CORRECT?
4052 011222 042767 177760 167756  BIC     #177760,$ERPSW ;BRANCH IF NO
4053 011230 022767 000000 167750  CMP     #0,$ERPSW    ;MASK OFF CC'S
4054 011236 001513          BEQ     7$          ;ARE CC'S CORRECT?
4055 011240 005067 167716          6$:  CLR     $TMP0        ;BRANCH IF YES
4056 011244 104077          ERROR   77            ;SAVE EXPECTED VALUE
4057 011246 000507          BR      7$          ;BAD CC'S DUE TO STATE DVC.70
4058 011250 012767 000002 167704  5$:  MOV     #2,$TMP0      ;SAVE EXPECTED
4059 011256 005067 167702          CLR     $TMP1        ;VALUES
4060 011262 020127 177776          CMP     R1,#177776   ;DID BEN16*DR0(0) FAIL?
4061 011266 001002          BNE     8$          ;BRANCH IF NO
4062 011270 104100          ERROR   100         ;RACK E50(B0) STUCK HIGH
4063 011272 000475          BR      7$
4064 011274 020127 000004          8$:  CMP     R1,#4        ;DID BEN04 FAIL?
4065 011300 001002          BNE     20$         ;BRANCH IF NO
4066 011302 104101          ERROR   101         ;BEN04 FAILED
4067 011304 000470          BR      7$
4068 011306 010067 167642          20$: MOV     R0,$REG0     ;SAVE R0 &
4069 011312 010167 167640          MOV     R1,$REG1     ;R1 FOR TYPEOUT
4070 011316 104102          ERROR   102         ;QUOTIENT OK BUT REMAINDER BAD
4071 011320 000462          BR      7$
4072 011322 012767 000002 167632  4$:  MOV     #2,$TMP0      ;SAVE EXPECTED
4073 011330 005067 167630          CLR     $TMP1        ;VALUES
4074 011334 022700 000020          CMP     #20,R0       ;DID BEN03 FAIL?
4075 011340 001002          BNE     9$          ;BRANCH IF NO
4076 011342 104103          ERROR   103         ;RACK E49(A1) STUCK LOW
4077 011344 000450          BR      7$
4078 011346 022700 077777          9$:  CMP     #77777,R0    ;DID BEN04 (-DIV SUB) FAIL?
4079 011352 001002          BNE     10$        ;BRANCH IF NO
4080 011354 104104          ERROR   104         ;BEN04 STUCK TO DIV SUB
4081 011356 000443          BR      7$
4082 011360 022700 177777          10$: CMP     #-1,R0       ;DID R0 GET A -1?
4083 011364 001002          BNE     13$        ;BRANCH IF NO
4084 011366 104110          ERROR   110         ;BEN04 (-N) FAILED
4085 011370 000436          BR      7$
4086 011372 020027 000000          13$: CMP     R0,#0        ;DID R0 STAY 0?
4087 011376 001406          BEQ     11$        ;BRANCH IF YES
4088 011400 010067 167550          MOV     R0,$REG0     ;SAVE R0 & R1
4089 011404 010167 167546          MOV     R1,$REG1     ;FOR TYPEOUT
4090 011410 104105          ERROR   105         ;INSTRUCTION FAILED
4091 011412 000425          BR      7$
4092 011414 020127 155776          11$: CMP     R1,#155776   ;DID BEN16*SR15(0) FAIL?
    
```

```

4093 011420 001002          BNE      12$          ;BRANCH IF NO
4094 011422 104106          ERROR   106          ;BEN16 FAILED TO DIV.50
4095 011424 000420          BR       7$
4096 011426 020127 000004 12$:    CMP      R1,#4        ;DID R1 CHANGE?
4097 011432 001406          BEQ     14$          ;BRANCH IF NO
4098 011434 010067 167514  MOV     RO,$REGO     ;SAVE RO & R1
4099 011440 010167 167512  MOV     R1,$REG1    ;FOR TYPEOUT
4100 011444 104107          ERROR   107          ;CAN'T DETERMINE FAILURE
4101 011446 000407          BR       7$
4102 011450 032767 000001 167530 14$:    BIT     #BIT0,$ERPSW ;DID BEN02 FAIL?
4103 011456 001402          BEQ     15$          ;BRANCH IF NO
4104 011460 104111          ERROR   111          ;BEN02 FAILED
4105 011462 000401          BR       7$
4106 011464 104112          15$:    ERROR   112          ;BEN05 FAILED
4107
4108
4109 011466 012767 011474 167414 7$:    MOV     #63$,$LPERR ;SETUP ERROR LOOP
4110 011474 005000          63$:    CLR     RO          ;PUT DIVIDEND IN RO
4111 011476 012701 000006  MOV     #6,R1        ;PUT DIVIDEND IN R1 (DIVISOR=2 IN R2)
4112 011502 000240          NOP
4113 011504 071002          DIV     R2,RO        ;OSCILLOSCOPE SYNC POINT
4114 011506 020127 000002  CMP     R1,#2        ;EXECUTE INSTRUCTION UNDER TEST
4115 011512 001001          BNE     16$          ;DID BEN16*DR0(1) FAIL?
4116 011514 104113          ERROR   113          ;BRANCH IF NO
4117
4118
4119 011516 012767 011524 167364 16$:    MOV     #62$,$LPERR ;RACK E50(B0) STUCK LOW
4120 011524 005000          62$:    CLR     RO          ;SETUP ERROR LOOP
4121 011526 012701 000004  MOV     #4,R1        ;PUT DIVIDEND IN RO & R1
4122 011532 012702 177776  MOV     #-2,R2       ;PUT DIVISOR IN R2
4123 011536 012767 177776 167416  MOV     #-2,$TMP0    ;SAVE EXPECTED
4124 011544 005067 167414  CLR     $TMP1        ;VALUES
4125 011550 000240          NOP
4126 011552 071002          DIV     R2,RO        ;OSCILLOSCOPE SYNC POINT
4127 011554 013767 177776 167424  MOV     @#PSW,$ERPSW ;EXECUTE INSTRUCTION UNDER TEST
4128 011562 020027 177776  MOV     RO,#177776  ;SAVE CC'S
4129 011566 001413          CMP     RO,#177776  ;DID DIVIDE WORK?
4130 011570 020027 027777  BEQ     17$          ;BRANCH IF YES
4131 011574 001002          CMP     RO,#27777  ;DID BEN16*SR15(1) FAIL?
4132 011576 104114          BNE     18$          ;BRANCH IF NO
4133 011600 000432          ERROR   114          ;RACK E64(B0) STUCK LOW
4134 011602 010067 167346 18$:    BR       21$
4135 011606 010167 167344  MOV     RO,$REGO     ;SAVE RO & R1 FOR
4136 011612 104115          MOV     R1,$REG1    ;TYPEOUT
4137
4138 011614 000424          ERROR   115          ;EITHER STATE DVC.20 OR DVC.40 OR DVC.80
4139 011616 020127 000000 17$:    BR       21$          ;OR DVC.90 FAILED
4140 011622 001406          CMP     R1,#0        ;DID REMAINDER COME OUT CORRECT?
4141 011624 010067 167324  BEQ     19$          ;BRANCH IF YES
4142 011630 010167 167322  MOV     RO,$REGO     ;SAVE RO & R1
4143 011634 104116          MOV     R1,$REG1    ;FOR TYPEOUT
4144 011636 000413          ERROR   116          ;QUOTIENT OK, REMAINDER BAD
4145 011640 042767 177760 167340 19$:    BR       21$
4146 011646 022767 000010 167332  BIC     #177760,$ERPSW ;MASK OF CC'S
4147 011654 001404          CMP     #10,$ERPSW  ;DID CC'S COME OUT CORRECT?
4148 011656 012767 000010 167276  BEQ     21$          ;CONTINUE IF YES
                                MOV     #10,$TMP0    ;SAVE EXPECTED VALUE
    
```



```
4149 011664 104117          ERROR 117          ;BAD CC'S DUE TO STATE DVC.90
4150          ;:*****
4151          ;SECTION FIVE (DIV QUIT CAUSED BY N(0)*SR15(0))
4152 011666 012767 011674 167214 21$: MOV #61$, $LPERR ;SETUP ERROR LOOP
4153 011674 012700 000002 61$: MOV #2, R0 ;SETUP R0 AND R1
4154 011700 005001          CLR R1 ;TO CAUSE DIV QUIT ABORT
4155 011702 012702 000001          MOV #1, R2 ;PU. DIVISOR IN R2
4156 011706 000277          SCC ;SETUP CC'S TO COMPLIMENT
4157 011710 000242          CLV ;OF EXPECTED VALUE
4158          ;AND OSCILLOSCOPE SYNC POINT
4159 011712 071002          DIV R2, R0 ;EXECUTE INSTRUCTION UNDER TEST
4160 011714 013767 177776 167264 MOV @#PSW, $ERPSW ;SAVE CC'S
4161 011722 020027 000002          CMP R0, #2 ;DID DIVIDE ABORT?
4162 011726 001402          BEQ 22$ ;BRANCH IF YES
4163 011730 104120          ERROR 120 ;DIV QUIT DID NOT GO LOW
4164 011732 000413          BR 23$
4165 011734 042767 177760 167244 22$: BIC #177760, $ERPSW ;MASK OFF CC'S
4166 011742 022767 000002 167236          CMP #2, $ERPSW ;DID CC'S COME OUT OK?
4167 011750 001404          BEQ 23$ ;BRANCH IF YES
4168 011752 012767 000002 167202          MOV #2, $TMPO ;STORE EXPECTED CC'S FOR TYPEOUT
4169 011760 104121          ERROR 121 ;CC'S BAD DUE TO EITHER DIV.30 OR DVE.20
4170          ;:*****
4171          ;SECTION SIX (DIV QUIT CAUSED BY SHFTR=0)
4172 011762 012767 011770 167120 23$: MOV #60$, $LPERR ;SETUP ERROR LOOP
4173 011770 012700 000002 60$: MOV #2, R0 ;SETUP R0 & R1 TO
4174 011774 005001          CLR R1 ;CAUSE DIV QUIT TO ABORT
4175 011776 012702 177776          MOV #-2, R2 ;SETUP DIVISOR (-2)
4176 012002 000240          NOP ;OSCILLOSCOPE SYNC POINT
4177 012004 071002          DIV R2, R0 ;EXECUTE INSTRUCTION UNDER TEST
4178 012006 022700 000002          CMP #2, R0 ;DID DIVIDE ABORT?
4179 012012 001401          BEQ 24$ ;BRANCH IF YES
4180 012014 104122          ERROR 122 ;GRAJ E5 IS BAD (Z2(0)*LEFT SAVE (1))
4181          ;:*****
4182          ;SECTION SEVEN
4183 012016 012767 012024 167064 24$: MOV #57$, $LPERR ;SETUP ERROR LOOP
4184 012024 012700 100000 57$: MOV #BIT15, R0 ;MAKE DIVIDEND
4185 012030 005001          CLR R1 ;MOST NEGATIVE NUMBER
4186 012032 012702 000001          MOV #1, R2 ;SETUP DIVISOR
4187 012036 000257          CCC ;SET CC'S TO COMPLIMENT OF EXPECTED
4188 012040 000264          SEZ ;OSCILLOSCOPE SYNC POINT
4189 012042 071002          DIV R2, R0 ;EXECUTE INSTRUCTION UNDER TEST
4190 012044 013767 177776 167134 MOV @#PSW, $ERPSW ;SAVE PSW
4191 012052 020027 100000          CMP R0, #BIT15 ;DID DIVIDE ABORT?
4192 012056 001402          BEQ 25$ ;BRANCH IF YES
4193 012060 104123          ERROR 123 ;BEN04*N FAILED
4194          ;
4195 012062 000413          BR 26$
4196 012064 042767 177760 167114 25$: BIC #177760, $ERPSW ;MASK OFF CC'S
4197 012072 022767 000016 167106          CMP #16, $ERPSW ;DID CC'S LOAD PROPERLY?
4198 012100 001404          BEQ 26$ ;BRANCH IF YES
4199 012102 012767 000016 167052          MOV #16, $TMPO ;STORE EXPECTED VALUE
4200 012110 104124          ERROR 124 ;CC'S DID NOT LOAD PROPERLY
4201          ;:*****
4202          ;SECTION EIGHT (DIV QUIT CAUSED BY N(1)*SR15(1))
4203 012112 012767 012120 166770 26$: MOV #56$, $LPERR ;SETUP ERROR LOOP
4204 012120 012700 177776 56$: MOV #177776, R0 ;SETUP R0 & R1 TO
```

```

4205 012124 012701 177777      MOV      #177777,R1      ;CAUSE DIV QUIT TO ABORT
4206 012130 012702 177777      MOV      #177777,R2      ;SETUP DIVISOR (-1)
4207 012134 000240      NOP                      ;OSCILLOSCOPE SYNC POINT
4208 012136 071002      DIV      R2,R0           ;EXECUTE INSTRUCTION UNDER TEST
4209 012140 020027 000001      CMP      R0,#1          ;DID DIVIDE ABORT LEAVE R0=1
4210 012144 001401      BEQ      27$            ;BRANCH IF YES
4211 012146 000403      BR       35$
4212 012150 020127 000001      27$:  CMP      R1,#1          ;DID DIVIDE ABORT?
4213 012154 001413      BEQ      28$            ;BRANCH IF YES
4214 012156 012767 000001 166776 35$:  MOV      #1,$TMP0        ;STORE EXPECTED VALUE
4215 012164 010067 166764      MOV      R0,$REG0        ;SAVE R0
4216 012170 012767 000001 166766      MOV      #1,$TMP1
4217 012176 010167 166754      MOV      R1,$REG1        ;SAVE R1
4218 012202 104125      ERROR    125            ;EITHER GRAJ E5 BAD OR MICROSTATE BAD
4219
4220      ;*****
4220      ;SECTION NINE
4221 012204 012767 012212 166676 28$:  MOV      #55$,$LPERR     ;SETUP ERROR LOOP
4222 012212 012700 177777      55$:  MOV      #-1,R0         ;PUT DIVIDEND IN
4223 012216 012701 177773      MOV      #-5,R1         ;R0 & R1
4224 012222 012702 000002      MOV      #2,R2         ;PUT DIVISOR IN R2
4225 012226 012767 177776 166726      MOV      #-2,$TMP0      ;SAVE EXPECTED VALUE
4226 012234 012767 177777 166722      MOV      #-1,$TMP1      ;SAVE EXPECTED VALUE
4227 012242 000240      NOP                      ;OSCILLOSCOPE SYNC POINT
4228 012244 071002      DIV      R2,R0           ;EXECUTE INSTRUCTION UNDER TEST
4229 012246 010067 166702      MOV      R0,$REG0        ;SAVE R0
4230 012252 010167 166700      MOV      R1,$REG1        ;SAVE R1
4231 012256 020127 177777      CMP      R1,#-1         ;IS REMAINDER CORRECT?
4232 012262 001402      BEQ      29$            ;BRANCH IF YES
4233 012264 104126      ERROR    126            ;REMAINDER BAD
4234 012266 000404      BR       30$
4235 012270 022700 177776      29$:  CMP      #-2,R0         ;IS QUOTIENT CORRECT?
4236 012274 001401      BEQ      30$            ;BRANCH IF YES
4237 012276 104127      ERROR    127            ;QUOTIENT IS INCORRECT
4238
4239      ;*****
4239      ;SECTION TEN
4240 012300 012767 012306 166602 30$:  MOV      #54$,$LPERR     ;SETUP ERROR LOOP
4241 012306 012700 177777      54$:  MOV      #-1,R0         ;SETUP
4242 012312 012701 177773      MOV      #-5,R1         ;DIVIDEND
4243 012316 012702 177776      MOV      #-2,R2         ;AND DIVISOR
4244 012322 012767 000002 166632      MOV      #2,$TMP0      ;SAVE EXPECTED VALUES
4245 012330 012767 177777 166626      MOV      #-1,$TMP1      ;OF R0 & R1
4246 012336 000240      NOP                      ;OSCILLOSCOPE SYNC POINT
4247 012340 071002      DIV      R2,R0           ;EXECUTE INSTRUCTION UNDER TEST
4248 012342 010067 166606      MOV      R0,$REG0        ;SAVE R0
4249 012346 010167 166604      MOV      R1,$REG1        ;SAVE R1
4250 012352 020027 000002      CMP      R0,#2          ;IS QUOTIENT CORRECT?
4251 012356 001402      BEQ      31$            ;BRANCH IF YES
4252 012360 104130      ERROR    130            ;QUOTIENT BAD
4253 012362 000404      BR       32$
4254 012364 020127 177777      31$:  CMP      R1,#-1         ;IS REMAINDER CORRECT
4255 012370 001401      BEQ      32$            ;BRANCH IF YES
4256 012372 104131      ERROR    131            ;REMAINDER INCORRECT (QUOT. OK)
4257
4258      ;*****
4258      ;SECTION ELEVEN
4259 012374 012767 012402 166506 32$:  MOV      #53$,$LPERR     ;SETUP ERROR LOOP
4260 012402 012700 177776      53$:  MOV      #177776,R0     ;SETUP DIVIDEND
    
```

```

4261 012406 005001          CLR      R1          ;AND DIVISOR TO GET A
4262 012410 012702 040000  MOV      #40000,R2  ;QUOT OF -10 & REMAINDER=0
4263 012414 012767 177770 166540  MOV      #-10,$TMP0 ;SAVE EXPECTED VALUE
4264 012422 005067 166536          CLR      $TMP1      ;SAVE EXPECTED VALUE
4265 012426 000240          NOP                    ;OSCILLOSCOPE SYNC POINT
4266 012430 071002          DIV      R2,R0       ;EXECUTE INSTRUCTION UNDER TEST
4267 012432 010067 166516  MOV      R0,$REG0    ;SAVE R0
4268 012436 010167 166514  MOV      R1,$REG1    ;SAVE R1
4269 012442 020027 177770  CMP      R0,#-10     ;IS QUOTIENT CORRECT?
4270 012446 001402          BEQ      33$         ;BRANCH IF YES
4271 012450 104132          ERROR   132         ;QUOTIENT IS BAD
4272 012452 000404          BR       34$         ;
4273 012454 020127 000000 33$:  CMP      R1,#0       ;IS REMAINDER CORRECT?
4274 012460 001401          BEQ      34$         ;BRANCH IF YES
4275 012462 104133          ERROR   133         ;REMAINDER BAD (QUOT. OK)
4276
4277 *****
4278 012464 012767 012472 166416 34$:  MOV      #52$,$LPERR ;SETUP ERROR LOOP
4279 012472 012700 000100 52$:  MOV      #100,R0     ;SETUP DIVIDEND
4280 012476 012701 000200          MOV      #200,R1    ;AND DIVISOR TO GENERATE
4281 012502 012702 177601          MOV      #-177,R2   ;DIVISION OVERFLOW
4282 012506 000277          SCC                    ;
4283 012510 000242          CLV                    ;AND OSCILLOSCOPE SYNC POINT
4284 012512 071002          DIV      R2,R0       ;EXECUTE INSTRUCTION UNDER TEST
4285 012514 013767 177776 166464  MOV      @#PSW,$ERPSW ;SAVE PSW
4286 012522 042767 177760 166456  BIC      #177760,$ERPSW ;MASK OFF CC'S
4287 012530 022767 000002 166450  CMP      #2,$ERPSW   ;ARE CC'S CORRECT?
4288 012536 001404          BEQ      TST15       ;:TEST OK, GO TO NEXT TEST
4289 012540 012767 000002 166414  MOV      #2,$TMP0    ;SAVE EXPECTED
4290 012546 104134          ERROR   134         ;BAD CC'S ON DIVISION OVERFLOW
4291 *****
4292 *TEST 15      MTP*DMO
4293 *
4294 *      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
4295 *
4296 *      THE ONLY OTHER POSSIBLE FAILURES WOULD BE IN ROM STATES
4297 *      MTP.00 OR MTP.10.
4298 *
4299 *      NOTE: THIS TEST ONLY TESTS THE CPU FUNCTIONS OF THIS INSTRUCTION.
4300 *      THE MEMORY MANAGEMENT TEST VERIFIES THE INTERMODE TRANSFER.
4301 *      AS FAR AS THE CPU IS CONCERNED THERE IS NO DIFFERENCE
4302 *      BETWEEN MTPI AND MTPD.
4303 *
4304 *      ROM FLOW-45,151,146,205
4305 *****
4306 012550 000004          TST15:  SCOPE
4307 012552 012767 012716 166414  MOV      #TST16,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
4308 012560 012767 012716 166502  MOV      #TST16,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
4309 012566 012737 012706 000010  MOV      #1$,@#RESVEC  ;SETUP RESVEC
4310 012574 005037 177776          CLR      @#PSW       ;ENSURE PREVIOUS MODE KERNAL
4311 012600 012706 001074          MOV      #1074,SP    ;SETUP THE SP
4312 012604 012716 100000          MOV      #BIT15,(SP) ;SET SIGN BIT ON STACK
4313 012610 005000          CLR      R0          ;ENSURE R0 CLEAR
4314 012612 000277          SCC                    ;SET CC'S TO COMPLIMENT
4315 012614 000250          CLN                    ;OF EXPECTED VALUE (EXCEPT FOR C)
4316          ;AND OSCILLOSCOPE SYNC POINT
    
```

4317	012616	006600				MTP1	R0	:EXECUTE INSTRUCTION UNDER TEST
4318	012620	013767	177776	166360		MOV	@#PSW,\$ERPSW	:SAVE PSW
4319	012626	010067	166322			MOV	R0,\$REG0	:SAVE R0 & SEC CC'S
4320	012632	012767	100000	166322		MOV	#BIT15,\$TMPO	:SAVE EXPECTED VALUE
4321	012640	100403				BMI	2\$:BRANCH IF R0 LOADED
4322	012642	012700	100000			MOV	#BIT15,R0	:SAVE EXPECTED VALUE
4323	012646	104135				ERROR	135	:R0 DID NOT LOAD
4324	012650	022706	001076		2\$:	CMP	#1076,SP	:DID SP INCREMENT?
4325	012654	001401				BEQ	3\$:BRANCH IF YES
4326	012656	104136				ERROR	136	:SP DID NOT INCREMENT
4327	012660	042767	177760	166320	3\$:	BIC	#177760,\$ERPSW	:MASK OFF CC'S
4328	012666	022767	000011	166312		CMP	#11,\$ERPSW	:DID CC'S COME OUT CORRECT?
4329	012674	001410				BEQ	TST16	:BRANCH IF YES
4330	012676	012767	000011	166256		MOV	#11,\$TMPO	:SAVE EXPECTED VALUE
4331	012704	104137				ERROR	137	:CC'S BAD (CC ROM)
4332	012706	012737	000012	000010	1\$:	MOV	#12,@#RESVEC	:RESTORE RESVEC
4333	012714	104140				ERROR	140	:FORK A FAILED
4334						:*****		
4335						:*TEST 16 MTP*DM1		
4336						:*		
4337						:* IF FORK A FAILS EXECUTION WILL GO TO RSD.00.		
4338						:* THIS WILL ONLY HAPPEN IF RACF E20(4) IS STUCK HIGH.		
4339						:*		
4340						:* THIS TEST ENSURES STATE MTP.10 RELOADS THE		
4341						:* DR IF THE DESTINATION IS R6 AND THAT IT PUTS THE PC IN THE		
4342						:* DR IF THE DESTINATION FIELD IS R7.		
4343						:*		
4344						:* ROM FLOW-45,151,146,111,155,312		
4345						:*****		
4346	012716	000004				TST16:	SCOPE	
4347	012720	012767	013070	166342		MOV	#TST17,NEXTTST	:SAVE ADDRESS OF NEXT TEST
4348	012726	012737	013060	000010		MOV	#1\$,@#RESVEC	:SETUP RESVEC
4349	012734	012706	001072			MOV	#1072,SP	:SETUP THE SP
4350	012740	012716	100000			MOV	#BIT15,(SP)	:SET THE SIGN BIT ON THE STACK
4351	012744	005066	000002			CLR	2(SP)	:ENSURE 1074 IS CLEAR
4352	012750	000240				NOP		:OSCILLOSCOPE SYNC POINT
4353	012752	006616				MTP1	(SP)	:EXECUTE INSTRUCTION UNDER TEST
4354	012754	005737	001074			TST	@#1074	:DID 1074 GET SIGN BIT SET?
4355	012760	100413				BMI	2\$:BRANCH IF YES
4356	012762	022706	001074			CMP	#1074,SP	:DID MTP.10 FAIL TO RELOAD THE DR?
4357	012766	001002				BNE	3\$:BRANCH IF NO
4358	012770	104141				ERROR	141	:STATE MTP.10 FAILED
4359	012772	000406				BR	2\$	
4360	012774	010667	166154		3\$:	MOV	SP,\$REG0	:SAVE SP
4361	013000	012767	001074	166150		MOV	#1074,\$REG1	:SAVE EXPECTED VALUE
4362	013006	104142				ERROR	142	:DON'T KNOW WHAT HAPPENED
4363	013010	012767	013016	166072	2\$:	MOV	#64\$,\$LPERR	:SETUP ERROR LOOP
4364	013016	012706	001074		64\$:	MOV	#1074,SP	:SETUP THE SP
4365	013022	000240				NOP		:OSCILLOSCOPE SYNC POINT
4366	013024	006617				MTP1	(PC)	:EXECUTE INSTRUCTION UNDER TEST
4367	013026	000000			4\$:	.WORD	0	
4368	013030	005767	177772			TST	4\$:DID 4\$ GET BIT15 SET?
4369	013034	100403				BMI	5\$:BRANCH IF YES
4370	013036	005067	177764			CLR	4\$:RESTORE 4\$
4371	013042	104143				ERROR	143	:STATE MTP.10 DID NOT PUT PCB IN DR
4372	013044	005067	177756		5\$:	CLR	4\$:RESTORE 4\$

```

4373 013050 012737 000012 000010      MOV    #12,@#RESVEC    ;RESTORE RESVEC
4374 013056 000404                BR     TST17           ;;GO TO NEXT TEST
4375 013060 012737 000012 000010 1$:  MOV    #12,@#RESVEC    ;RESTORE RESVEC
4376 013066 104144                ERROR  144            ;FORK A FAILED
4377                                     ;*****
4378                                     ;*TEST 17          MFP*DM0
4379                                     ;*
4380                                     ;*   IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
4381                                     ;*   IF ANYTHING ELSE FAILS, THEN A ROM STATE IS BAD.
4382                                     ;*
4383                                     ;*   ROM FLOW-46,304,250,222,300
4384                                     ;*
4385 013070 000004                TST17: SCOPE
4386 013072 012767 013256 166074      MOV    #TST20,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
4387 013100 012767 013256 166162      MOV    #TST20,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
4388 013106 012767 013142 165772      MOV    #4$,$LPADR     ;SET UP LOOP
4389 013114 012767 013142 165766      MOV    #4$,$LPERR     ;SETUP ERROR LOOP
4390 013122 013767 177776 166040      MOV    @#PSW,$TMP3    ;SAVE PSW
4391 013130 012746 000340      MOV    #PR7,-(SP)     ;PUT NEW PSW ON STACK
4392 013134 012746 013142      MOV    #4$,-(SP)     ;PUT RETURN ADDR ON STACK
4393 013140 000006                RTT                    ;TURN T BIT OFF
4394 013142 012737 013246 000010 4$:  MOV    #1$,@#RESVEC   ;SETUP RESVEC
4395 013150 012706 001076      MOV    #1076,SP       ;SETUP THE SP
4396 013154 005016                CLR    (SP)           ;ENSURE 1076 CLEAR
4397 013156 005066 177776      CLR    -2(SP)         ;ENSURE 1074 CLEAR
4398 013162 012700 100000      MOV    #BIT15,R0      ;SET THE SIGN BIT IN R0
4399 013166 000277                SCC                    ;SETUP CC'S TO COMPLIMENT
4400 013170 000250                CLN                    ;OF EXPECTED VALUE (EXCEPT FOR C)
4401                                     ;AND OSCILLOSCOPE SYNC POINT
4402 013172 006500                MFPI   R0             ;EXECUTE INSTRUCTION UNDER TEST
4403 013174 013767 177776 166004      MOV    @#PSW,$ERPSW   ;SAVE THE PSW
4404 013202 022706 001074      CMP    #1074,SP       ;DID THE SP DECREMENT?
4405 013206 001401                BEQ    2$             ;BRANCH IF YES
4406 013210 104145                ERROR  145           ;STATE MFP.10 DID NOT DEC. THE SP
4407 013212 005716                2$:  TST    (SP)        ;DID R0 GET PUT ON THE STACK?
4408 013214 100401                BMI    3$             ;BRANCH IF YES
4409 013216 104146                ERROR  146           ;R0 DID NOT GET PUT ON THE STACK
4410 013220 042767 177760 165760 3$:  BIC    #177760,$ERPSW ;MASK OFF THE CC'S
4411 013226 022767 000011 165752      CMP    #11,$ERPSW    ;DID THE CC'S SET CORRECTLY?
4412 013234 001410                BEQ    TST20         ;;BRANCH IF YES
4413 013236 012767 000011 165716      MOV    #11,$TMP0     ;SAVE EXPECTED VALUE
4414 013244 104147                ERROR  147           ;INCORRECT CC'S
4415 013246 012737 000012 000010 1$:  MOV    #12,@#RESVEC   ;RESTORE RESVEC
4416 013254 104150                ERROR  150           ;FORK A FAILED
4417                                     ;*****
4418                                     ;*TEST 20          MFP*DM2
4419                                     ;*
4420                                     ;*   IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
4421                                     ;*   IF FORK B FAILS EXECUTION WILL ALSO GO TO RSD.00 BUT THE DR
4422                                     ;*   WILL HAVE BEEN INCREMENTED. IF ANYTHING ELSE FAILS THEN
4423                                     ;*   STATE MFP.00 IS BAD.
4424                                     ;*
4425                                     ;*   ROM FLOW-2,175,66,250,222,300
4426                                     ;*
4427 013256 000004                TST20: SCOPE
4428 013260 012767 013416 165706      MOV    #TST21,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
    
```

```

4429 013266 012767 013416 165774      MOV      #TST21,NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
4430 013274 012737 013376 000010      MOV      #1$,@#RESVEC   ;SETUP RESVEC
4431 013302 012706 001076      MOV      #1076,SP       ;SETUP THE SP
4432 013306 005016      CLR      (SP)           ;ENSURE WORDS ON
4433 013310 005066 177776      CLR      -2(SP)         ;STACK CLEAR
4434 013314 012700 001164      MOV      #$TMP1,R0      ;PUT ADDRESS OF $TMP1 IN R0
4435 013320 012710 100000      MOV      #BIT15,(R0)    ;SET THE SIGN BIT IN $TMP1
4436 013324 000277      SCC      ;SETUP CC'S TO COMPLIMENT OF
4437 013326 000250      CLN      ;EXPECTED VALUE (EXCEPT FOR C)
4438                                     ;AND OSCILLOSCOPE SYNC POINT
4439 013330 006520      MFPI     (R0)+          ;EXECUTE INSTRUCTION UNDER TEST
4440 013332 013767 177776 165646      MOV      @#PSW,$ERPSW   ;SAVE PSW
4441 013340 022706 001074      CMP      #1074,SP       ;DID THE SP DECREMENT?
4442 013344 001401      BEQ      2$             ;BRANCH IF YES
4443 013346 104151      ERROR    151            ;STATE MFP.00 IS BAD
4444 013350 042767 177760 165630 2$:    BIC      #177760,$ERPSW ;MASK OFF THE CC'S
4445 013356 022767 000011 165622      CMP      #11,$ERPSW     ;DID THE CC'S SET CORRECTLY?
4446 013364 001414      BEQ      TST21          ;BRANCH IF YES
4447 013366 012767 000011 165566      MOV      #11,$TMP0      ;SAVE EXPECTED VALUE
4448 013374 104152      ERROR    152            ;INCORRECT CC'S DUE TO STATE MFP.00
4449 013376 012737 000012 000010 1$:    MOV      #12,@#RESVEC   ;RESTORE RESVEC
4450 013404 022700 001166      CMP      #$TMP2,R0      ;DID R0 INCREMENT?
4451 013410 001401      BEQ      3$             ;BRANCH IF YES
4452 013412 104153      ERROR    153            ;FORK A FAILED
4453 013414 104154      3$:     ERROR    154            ;FORK B FAILED
4454                                     ;*****
4455                                     ;*TEST 21      BPT
4456                                     ;*
4457                                     ;*   FORK A SHOULD NOT FAIL.
4458                                     ;*   IF THE TRAP VECTOR LOGIC FAILS THE TRAP VECTOR WOULD
4459                                     ;*   COME OUT TO BE 4.
4460                                     ;*   THE ONLY OTHER FAILURE WOULD BE TRP.00.
4461                                     ;*   IF THIS STATE FAILS TO LOAD THE DR THE TRAP VECTOR WILL
4462                                     ;*   BE WHATEVER IS IN R3. IF IT FAILS TO LOAD THE BR THE OLD
4463                                     ;*   PS WILL FAIL TO BE STACKED.
4464                                     ;*****
4465 013416 000004      TST21:  SCOPE
4466 013420 012767 013532 165546      MOV      #TST22,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
4467 013426 012767 013532 165634      MOV      #TST22,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
4468 013434 012737 013472 000014 4$:    MOV      #1$,@#BPTVEC   ;SETUP BPT VECTOR
4469 013442 012706 001100      MOV      #STACK,SP      ;SETUP THE SP
4470 013446 012737 013512 000004      MOV      #2$,@#ERRVEC   ;SETUP LOCATION 4
4471 013454 012737 013514 000024      MOV      #3$,@#24       ;SETUP LOCATION 24
4472 013462 012703 000024      MOV      #24,R3         ;SETUP R3
4473 013466 000237      SPL      7              ;SETUP PSW
4474 013470 000003      BPT      ;EXECUTE INSTRUCTION UNDER TEST
4475 013472 042737 177437 001076 1$:    BIC      #177437,@#1076 ;MASK OFF PRIORITIES OF OLD PS
4476 013500 022737 000340 001076      CMP      #340,@#1076    ;DID OLD PS GET STACKED?
4477 013506 001403      BEQ      5$             ;BRANCH IF YES
4478 013510 104160      ERROR    160            ;STATE TRP.00 FAILED
4479 013512 104161      2$:     ERROR    161            ;TRAP VECTOR CAME UP BAD
4480 013514 104162      3$:     ERROR    162            ;STATE TRP.00 FAILED
4481 013516 012737 032636 000004 5$:    MOV      #CPUSPUR,@#ERRVEC ;RESTORE ERR VEC
4482 013524 012737 032622 000014      MOV      #$RTRN,@#TBITVEC ;RESTORE T BIT VECTOR
4483                                     ;*****
4484

```

4485
4486
4487
4488
4489
4490
4491
4492
4493
4494
4495
4496
4497
4498
4499
4500
4501
4502
4503
4504
4505
4506
4507
4508
4509
4510
4511
4512
4513
4514
4515
4516
4517
4518
4519
4520
4521
4522
4523
4524
4525
4526
4527
4528
4529
4530
4531
4532
4533
4534
4535
4536
4537
4538
4539
4540

```

; *ALL THE LOGIC FOR (JMP+JSR)*DMO HAS BEEN TESTED.
; *****
; *****
; *TEST 22          BIT TEST OF PIRQ REGISTER
; *
; *   IF ONE OF THE BLOCK LEVELS IS STUCK HIGH OR TMCB
; *   PS07(0)'S STUCK HIGH OR PDRD PRIORITY=0 IS STUCK HIGH,
; *   A PIRQ TRAP LOOP WILL OCCUR WHEN THAT PIR LEVEL IS ENABLED.
; *
; *   A COUNT PATTERN IS THEN RUN THRU THE REGISTER TO ENSURE THAT
; *   THE ENCODER FUNCTIONS PROPERLY.
; *****

```

```

TST22: SCOPE
MOV     #TST23,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
MOV     #TST23,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
MOV     #STACK,SP      ;INITIALIZE THE SP
MOV     #4,$,SLPADR    ;SETUP LOOP ADR
MOV     #4,$,SLPERR    ;SETUP ERROR LOOP
BIT     #BIT4,$TMP3    ;WAS T BIT ON?
BEQ     4$             ;BRANCH IF NO
MOV     #360,-(SP)     ;PUT NEW PSW ON STACK
MOV     #4,$,-(SP)    ;PUT RETURN ADDR ON STACK
RTT                    ;TURN T BIT ON
4$:     SPL            7 ;SET THE CPU PRIORITY AT 7.
CLR     $TMP0          ;SETUP COMPARISON LOCATION
CLR     @#PIRQ         ;CLEAR PIRQ REGISTER
CMP     $TMP0,@#PIRQ  ;DID PIRQ CLEAR?
BNE     1$             ;BRANCH IF NO
MOV     #177,R0        ;SETUP ITERATION COUNT
MOV     #1042,$TMP0   ;SETUP COMPARISON LOCATION
MOV     #2,R1          ;SETUP R1
2$:     ADD     #1000,@#PIRQ ;START COUNT PATTERN
CMP     $TMP0,@#PIRQ  ;DID REGISTER SET CORRECT?
BNE     1$             ;BRANCH IF NO
CMPB   R1,@#PIRQ+1   ;IS PIRQ READY TO GO TO NEXT LEVEL?
BNE     3$             ;BRANCH IF NO
ADD     #42,$TMP0     ;INCREMENT ENCODED VALUES IN TEST LOC.
INC     R1             ;SETUP R1 FOR ROTATE
ROL     R1             ;SET R1 TO NEXT CHECK LEVEL
3$:     ADD     #1000,$TMP0 ;INC. PIRQ LEVEL IN TEST LOCATION
SOB    R0,2$         ;CONTINUE COUNT
CLR     @#PIRQ        ;ENSURE PIRQ CLEAR
BR     TST23         ;;GO TO NEXT TEST
1$:     MOV     @#PIRQ,$EPIRQ ;SAVE PIRQ FOR TIMEOUT
ERROR  163          ;PIRQ REG. FAILED
; *****

```

```

; *TEST 23          PIR LEVEL 1 INTERRUPT
; *
; *   IF BEN13 FAILS EXECUTION WOULD GO TO ONE OF THE FOLLOWING:
; *   PUP.00, BRK.20, OR SER.00.
; *   PUP.00 WOULD START THE POWER UP ROUTINE.
; *   BRK.20 WOULD CAUSE A TRAP TO ZERO.
; *   SER.00 WOULD PUT 4 IN THE SP AND PERFORM A RED ZONE TRAP.
; *   IF TMCB PIRQ DOES NOT GET TO DAPE OR IF DAPE TV05*07 DOES NOT
; *   GO HIGH OR DOES NOT GET THRU TO THE ALU A TRAP TO 4 WILL OCCUR.
; *

```

```

4541 ;* IF THE INTERRUPT DOESN'T OCCUR AN ATTEMPT IS MADE TO ISOLATE THE
4542 ;* FAILURE.
4543 ;*****
4544 TST23: SCOPE
4545 MOV #TST24,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
4546 MOV #TST24,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
4547 MOV #10$,@#PWRVEC ;SETUP
4548 MOV #12$,@#0 ;SETUP LOCATION 0
4549 MOV #PR7,@#2 ;PUT PRIORITY 7 IN 2
4550 MOV #11$,@#4 ;FAILURE TRAP VECTORS
4551 MOV #1$,@#PIRQVEC ;SETUP PIRQ VECTOR
4552 BIT #BIT4,@#PSW ;IS T BIT ON?
4553 BEQ 13$ ;BRANCH IF NO
4554 MOV #360,@#PIRQVEC+2 ;SET T BIT IN PIRQ VECTOR
4555 BR 14$ ;
4556 13$: MOV #PR7,@#PIRQVEC+2 ;SETUP PIRQ VECTOR PSW
4557 14$: MOV #STACK,SP ;SETUP THE SP
4558 SPL 0 ;SET PRIORITY LEVEL AT ZERO
4559 BIS #BIT9,@#PIRQ ;SET LEVEL ONE
4560 CLR @#PIRQ ;CLEAR LEVEL ONE
4561 ;TRY TO GET INTERRUPT AT FET.01 INSTEAD OF TST.10.
4562 MOV #2$,@#PIRQVEC ;SETUP PIRQ VEC
4563 MOV #BIT9,@#PIRQ ;SET LEVEL ONE
4564 CLR @#PIRQ
4565 BR 3$ ;BRANCH IF NO INTERRUPT
4566 2$: CLR @#PIRQ ;CLEAR LEVEL 1
4567 ERROR 164 ;STATE TST.10 HAS BAD BEN FIELD
4568 ;TRY PIR LEVEL 4
4569 3$: MOV #4$,@#PIRQVEC ;SETUP PIRQ VECTOR
4570 BIS #BIT12,@#PIRQ ;SET LEVEL 4
4571 CLR @#PIRQ
4572 BR 5$ ;BRANCH IF NO INTERRUPT
4573 4$: CLR @#PIRQ ;CLEAR LEVEL 4
4574 ERROR 165 ;EITHER TMCB E62(1) IS BAD OR SOMETHING
4575 ;IN THE HONOR PIR 1 LOGIC IS BAD.
4576 ;TRY PIR 6
4577 5$: MOV #6$,@#PIRQVEC ;SETUP PIRQ VECTOR
4578 BIS #BIT14,@#PIRQ ;SET LEVEL 6
4579 CLR @#PIRQ
4580 BR 7$ ;BRAHCN IF NO INTERRUPT
4581 6$: CLR @#PIRQ ;
4582 ERROR 166 ;EITHER TMCB E51(9) OR E55(10-11) OR
4583 ;E62 IS BAD OR TMCA INH BELOW BR6 IS
4584 ;STUCK LOW
4585 ;TRY PIR 7
4586 7$: MOV #8$,@#PIRQVEC ;SETUP PIRQ VECTOR
4587 BIS #BIT15,@#PIRQ ;SET LEVEL 7
4588 CLR @#PIRQ ;CLEAR LEVEL 7
4589 BR 9$ ;BRANCH IF NO INTERRUPT
4590 8$: CLR @#PIRQ ;
4591 ERROR 167 ;TMCA ABOVE BR7 MIGHT BE STUCK LOW
4592 9$: ERROR 170 ;TMCE BRQ CLOCK MIGHT BE STUCK LOW
4593 1$: CLR @#PIRQ ;CLEAR LEVEL 1
4594 MOV #CPUSPUR,@#ERRVEC ;RESTORE ERR VEC
4595 MOV #SPWRDN,@#PWRVEC ;RESTORE THE POWER VECTOR
4596 BR TST24 ;GO TO NEXT TEST
    
```



```
4597 014244 005037 177772 10$: CLR @#PIRQ ;CLEAR LEVEL 1
4598 014250 104171 ERROR 171 ;BEN 13 FAILED
4599 014252 005037 177772 11$: CLR @#PIRQ
4600 014256 012737 032636 000004 MOV #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
4601 014264 012706 001100 MOV #STACK,SP ;RESTORE THE SP
4602 014270 005037 177766 CLR @#CPUERR ;CLEAR ERROR REG
4603 014274 104172 ERROR 172 ;BEN 13 FAILED OR TRAP VECTOR FAILED
4604 014276 005037 177772 12$: CLR @#PIRQ ;CLEAR LEVEL 1
4605 014302 104377 ERROR 377 ;EITHER TMCB E53 IS NOT GOING HIGH
4606 014304 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.

```
4616 014306 000004 TST24: SCOPE
4617 014310 012767 014444 164752 MOV #TST25,NEXTTST ;SAVE ADDRESS OF NEXT TEST
4618 014316 012706 001100 MOV #STACK,SP ;SETUP THE SP
4619 014322 012737 014356 000000 MOV #1$,@#0 ;SETUP LOC. 0 TO CATCH BEN 13 FAILURE
4620 014330 012737 014372 000240 MOV #2$,@#PIRQVEC ;SETUP PIRQ VECTOR
4621 014336 000231 SPL 1 ;SET CPU PRIORITY TO 1
4622 014340 052737 002000 177772 BIS #BIT10,@#PIRQ ;SET LEVEL 2
4623 014346 005037 177772 CLR @#PIRQ ;CLEAR LEVEL 2
4624 014352 104174 ERROR 174 ;PIR 2 DID NOT INTERRUPT
4625 014354 000406 BR 2$
4626 014356 005037 177772 1$: CLR @#PIRQ ;CLEAR LEVEL 2
4627 014362 012737 032636 000004 MOV #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
4628 014370 104175 ERROR 175 ;BEN 13 FAILED
4629 ;ENSURE PIR09 & 10 NOT SHORTED
4630 014372 012767 014400 164510 2$: MOV #64$, $LPERR ;SETUP ERROR LOOP
4631 014400 005037 177772 64$: CLR @#PIRQ ;CLEAR LEVEL 2
4632 014404 012737 014430 000240 MOV #3$,@#PIRQVEC ;SETUP VECTOR
4633 014412 000231 SPL 1 ;SET LEVEL 1
4634 014414 052737 001000 177772 BIS #BIT9,@#PIRQ ;SET PIR 1
4635 014422 005037 177772 CLR @#PIRQ ;CLEAR PIR 1
4636 014426 000406 BR TST25 ;GO TO NEXT TEST
4637 014430 013767 177772 164560 3$: MOV @#PIRQ,$EPIRQ ;SAVE PIRQ FOR TYPEOUT
4638 014436 005037 177772 CLR @#PIRQ ;CLEAR LEVEL 1
4639 014442 104176 ERROR 176 ;PIR09 & 10 SHORTED
```

*TEST 25 PIR LEVEL 3 INTERRUPT

* IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
* THIS WILL ONLY HAPPEN IF TMCB E63(3) IS BAD.
* IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(3)
* IS BAD OR TMCB HONOR PIR3 IS BEING HELD HIGH.

```
4649 014444 000004 TST25: SCOPE
4650 014446 012767 014602 164614 MOV #TST26,NEXTTST ;SAVE ADDRESS OF NEXT TEST
4651 014454 012706 001076 MOV #1076,SP ;INITIALIZE THE SP
4652 014460 012737 014530 000240 MOV #1$,@#PIRQVEC ;SETUP PIRQ VECTOR
```

```

4653 014466 012737 014514 000000      MOV     #2$,@#0      ;SETUP LOCATION 0
4654 014474 000232                    SPL     2            ;SET CPU AT LEVEL 2
4655 014476 052737 004000 177772      BIS     #BIT11,@#PIRQ ;SET LEVEL 3 PIR
4656 014504 005037 177772      CLR     @#PIRQ      ;CLEAR LEVEL 3
4657 014510 104177      ERROR  177         ;INTERRUPT FAILED
4658 014512 000406                    BR     1$
4659 014514 005037 177772      CLR     @#PIRQ      ;CLEAR LEVEL 3
4660 014520 012737 032636 000004 2$:    MOV     #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
4661 014526 104200      ERROR  200         ;BEN 13 FAILED
4662 014530 012767 014536 164352 1$:    MOV     #64$, $LPERR   ;SETUP ERROR LOOP
4663 014536 005037 177772      CLR     @#PIRQ      ;CLEAR LEVEL 3
4664 014542 012737 014566 000240 64$:   MOV     #3$,@#PIRQVEC ;SETUP PIRQ VECTOR
4665 014550 000232                    SPL     2            ;SET CPU PRIORITY AT 2
4666 014552 052737 002000 177772      BIS     #BIT10,@#PIRQ ;ENABLE PIR2
4667 014560 005037 177772      CLR     @#PIRQ      ;CLEAR LEVEL 2
4668 014564 000406                    BR     TST26        ;GO TO NEXT TEST
4669 014566 013767 177772 164422 3$:   MOV     @#PIRQ,$EPIRQ  ;SAVE PIRQ
4670 014574 005037 177772      CLR     @#PIRQ      ;CLEAR IT
4671 014600 104201      ERROR  201         ;LEVEL 2 INTERRUPT WHEN CPU LEVEL
4672                                     ;2 ENABLED.
4673                                     ;*****
4674                                     ;*TEST 26      PIR LEVEL 4 INTERRUPT
4675                                     ;*
4676                                     ;*      IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
4677                                     ;*      THIS WILL ONLY HAPPEN IF TMCB E63(5) IS BAD.
4678                                     ;*
4679                                     ;*      IF THE INTERRUPT DOESN'T OCCUR THEN EITHER TMCB E62(5)
4680                                     ;*      IS BAD OR TMCA HONOR PIR 4 IS BEING HELD HIGH.
4681                                     ;*****
4682 014602 000004      TST26: SCOPE
4683 014604 012767 014752 164456      MOV     #TST27,NEXTTST ;SAVE ADDRESS OF NEXT TEST
4684 014612 012706 001100      MOV     #STACK,SP     ;INITIALIZE THE SP
4685 014616 012737 014670 000240      MOV     #1$,@#PIRQVEC ;SETUP PIRQ VEC
4686 014624 012737 014654 000000      MOV     #2$,@#0      ;SETUP LOCATION 0
4687 014632 000233                    SPL     3            ;SET CPU AT 3
4688 014634 052737 010000 177772      BIS     #BIT12,@#PIRQ ;ENABLE PIR 4
4689 014642 012737 032636 000004      MOV     #CPUSPUR,@#ERRVEC
4690 014650 104202      ERROR  202         ;INTERRUPT DID NOT OCCUR
4691 014652 000406                    BR     1$
4692 014654 005037 177772      CLR     @#PIRQ      ;CLEAR LEVEL 4
4693 014660 012737 032636 000004 2$:    MOV     #CPUSPUR,@#ERRVEC
4694 014666 104203      ERROR  203         ;BEN 13 FAILED
4695 014670 012767 014676 164212 1$:    MOV     #64$, $LPERR   ;SETUP ERROR LOOP
4696 014676 005037 177772      CLR     @#PIRQ      ;CLEAR LEVEL 4
4697 014702 012737 032636 000004 64$:   MOV     #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
4698 014710 012737 014736 000240      MOV     #3$,@#PIRQVEC ;SETUP PIRQ VECTOR
4699 014716 000233                    SPL     3            ;SET CPU AT LEVEL 3
4700 014720 052737 004000 177772      BIS     #BIT11,@#PIRQ ;SET LEVEL 3
4701 014726 005037 177772      CLR     @#PIRQ      ;CLEAR LEVEL 3
4702 014732 000237                    SPL     7
4703 014734 000406                    BR     TST27        ;GO TO NEXT TEST
4704 014736 013767 177772 164252 3$:   MOV     @#PIRQ,$EPIRQ  ;SAVE PIRQ
4705 014744 005037 177772      CLR     @#PIRQ      ;CLEAR IT
4706 014750 104204      ERROR  204         ;LEVEL 3 INTERRUPT WHEN CPU LEVEL 3 ENABLED
4707                                     ;*****
4708                                     ;*TEST 27      PIR LEVEL 5 INTERRUPT
    
```

4709
 4710
 4711
 4712
 4713
 4714
 4715
 4716 014752 000004
 4717 014754 012767 015126 164306
 4718 014762 012706 001100
 4719 014766 012737 015044 000240
 4720 014774 012737 015030 000000
 4721 015002 000234
 4722 015004 052737 020000 177772
 4723 015012 005037 177772
 4724 015016 012737 032636 000004
 4725 015024 104205
 4726 015026 000406
 4727 015030 005037 177772
 4728 015034 012737 032636 000004
 4729 015042 104206
 4730 015044 012767 015052 164036
 4731 015052 012737 032636 000004
 4732 015060 005037 177772
 4733 015064 012737 015112 000240
 4734 015072 000234
 4735 015074 012737 010000 177772
 4736 015102 005037 177772
 4737 015106 000237
 4738 015110 000406
 4739 015112 013767 177772 164076
 4740 015120 005037 177772
 4741 015124 104207

```

    ;*
    ;* 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.
    ;*
    ;*****
TST27: SCOPE
    MOV #TST30,NEXTTST ;SAVE ADDRESS OF NEXT TEST
    MOV #STACK,SP ;INITIALIZE THE SP
    MOV #1$,@#PIRQVEC ;SETUP THE PIRQ VECTOR
    MOV #2$,@#0 ;SETUP LOCATION 0
    SPL 4 ;SET CPU ST LEVEL 4
    BIS #BIT13,@#PIRQ ;SET LEVEL 5
    CLR @#PIRQ ;CLEAR LEVEL 5
    MOV #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
    ERROR 205 ;INTERRUPT DID NOT OCCUR
    BR 1$
2$: CLR @#PIRQ ;CLEAR LEVEL 5
    MOV #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
    ERROR 206 ;BEN 13 FAILED
1$: MOV #64$, $LPERR ;SETUP ERROR LOOP
64$: MOV #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
    CLR @#PIRQ ;CLEAR LEVEL 5
    MOV #3$,@#PIRQVEC ;SETUP PIRQ VECTOR
    SPL 4 ;SET CPU AT LEVEL 4
    MOV #BIT12,@#PIRQ ;SET LEVEL 4
    CLR @#PIRQ ;CLEAR LEVEL 4
    SPL 7
    BR TST30 ;;GO TO NEXT TEST
3$: MOV @#PIRQ,$EPIRQ ;SAVE PIRQ
    CLR @#PIRQ ;CLEAR LEVELS
    ERROR 207 ;LEVEL 4 INT. WHEN CPU LEVEL 4 ENABLED
    ;*****

```

4742
 4743
 4744
 4745
 4746
 4747
 4748
 4749
 4750
 4751
 4752
 4753 015126 000004
 4754 015130 012767 015326 164132
 4755 015136 012706 001100
 4756 015142 012737 015244 000240
 4757 015150 012737 015230 000000
 4758 015156 000235
 4759 015160 052737 040000 177772
 4760 015166 012737 032636 000004
 4761 015174 012737 015220 000240
 4762 015202 012737 100000 177772
 4763 015210 005037 177772
 4764 015214 104210

```

    ;*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).
    ;*
    ;*****
TST30: SCOPE
    MOV #TST31,NEXTTST ;SAVE ADDRESS OF NEXT TEST
    MOV #STACK,SP ;INITIALIZE THE SP
    MOV #1$,@#PIRQVEC ;SETUP THE PIRQ VECTOR
    MOV #2$,@#0 ;SETUP LOCATION 0
    SPL 5 ;SET THE CPU AT LEVEL 5
    BIS #BIT14,@#PIRQ ;SET LEVEL 6
    MOV #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
    MOV #3$,@#PIRQVEC ;SETUP THE PIRQ VECTOR
    MOV #BIT15,@#PIRQ ;SET LEVEL 7
    CLR @#PIRQ ;CLEAR LEVEL 7
    ERROR 210 ;FAILURE IS AFTER TMCB E70

```

```

4765 015216 000412          BR      1$
4766 015220 005037 177772  3$:    CLR      @#PIRQ          ;CLEAR LEVEL 7
4767 015224 104211          ERROR   211          ;FAILURE IS IN TMCB E70 OR BEFORE
4768 015226 000406          BR      1$
4769 015230 012737 032636 000004 2$:    MOV      #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
4770 015236 005037 177772          CLR      @#PIRQ          ;CLEAR LEVEL 6
4771 015242 104212          ERROR   212          ;BEN 13 FAILED
4772 015244 012767 015252 163636 1$:    MOV      #64$,$LPERR      ;SETUP ERROR LOOP
4773 015252 005037 177772 64$:   CLR      @#PIRQ          ;CLEAR LEVEL 6
4774 015256 012737 032636 000004  MOV      #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
4775 015264 012737 015312 000240  MOV      #4$,@#PIRQVEC    ;SETUP PIRQ VECTOR
4776 015272 000235          SPL     5              ;SET PRIORITY AT 5
4777 015274 012737 020000 177772  MOV      #BIT13,@#PIRQ    ;SET LEVEL 5
4778 015302 005037 177772          CLR      @#PIRQ          ;CLEAR LEVEL 5
4779 015306 000237          SPL     7
4780 015310 000406          BR      TST31          ;;GO TO NEXT TEST
4781 015312 013767 177772 163676 4$:    MOV      @#PIRQ,$EPIRQ    ;SAVE PIRQ
4782 015320 005037 177772          CLR      @#PIRQ          ;CLEAR LEVEL 5
4783 015324 104213          ERROR   213          ;LEVEL 5 INTERRUPT WHEN CPU 5 ENABLED
4784
4785 ;*****
4786 ;*TEST 31          PIR LEVEL 7 INTERRUPT
4787 ;*
4788 ;*          IF BEN 13 FAILS EXECUTION WILL GO TO BRK.20.
4789 ;*          THIS WILL ONLY HAPPEN IF TMCB E63(6) IS BAD.
4790 ;*
4791 ;*          IF THE INTERRUPT DOES NOT OCCUR THEN EITHER TMCB E70(6)
4792 ;*          IS BAD OR TMCA HONOR PIR7 IS BEING HELD HIGH.
4793 ;*****
4793 015326 000004          TST31: SCOPE
4794 015330 012767 015500 163732  MOV      #TST32,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
4795 015336 012706 001100          MOV      #STACK,SP      ;INITIALIZE THE SP
4796 015342 012737 015420 000240  MOV      #1$,@#PIRQVEC   ;SETUP THE PIRQ VECTOR
4797 015350 012737 015404 000000  MOV      #2$,@#0         ;SETUP LOCATION 0
4798 015356 000236          SPL     6              ;SET PRIORITY AT LEVEL 6
4799 015360 052737 100000 177772  BIS      #BIT15,@#PIRQ    ;SET LEVEL 7
4800 015366 005037 177772          CLR      @#PIRQ          ;CLEAR LEVEL 7
4801 015372 012737 032636 000004  MOV      #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
4802 015400 104214          ERROR   214          ;LEVEL 7 DID NOT INTERRUPT
4803 015402 000406          BR      1$
4804 015404 005037 177772  2$:    CLR      @#PIRQ          ;CLEAR LEVEL 7
4805 015410 012737 032636 000004  MOV      #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
4806 015416 104215          ERROR   215          ;BEN 13 FAILED
4807 015420 012767 015426 163462 1$:    MOV      #64$,$LPERR      ;SETUP ERROR LOOP
4808 015426 012737 032636 000004 64$:   MOV      #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
4809 015434 012737 015464 000240  MOV      #3$,@#PIRQVEC   ;SETUP PIRQ VECTOR
4810 015442 005037 177772          CLR      @#PIRQ          ;CLEAR LEVEL 7
4811 015446 000236          SPL     6              ;SET LEVEL 6 IN CPU
4812 015450 012737 040000 177772  MOV      #BIT14,@#PIRQ    ;SET PIR 6
4813 015456 005037 177772          CLR      @#PIRQ          ;CLEAR PIR 6
4814 015462 000406          BR      TST32          ;;GO TO NEXT TEST
4815 015464 013767 177772 163524 3$:    MOV      @#PIRQ,$EPIRQ    ;SAVE PIRQ
4816 015472 005037 177772          CLR      @#PIRQ          ;CLEAR LEVEL 6
4817 015476 104216          ERROR   216          ;LEVEL 6 INT. WHEN CPU AT 6
4818
4819 ;*****
4820 ;*TEST 32          UNIBUS TIMEOUT
    
```

```

4821
4822
4823
4824
4825
4826
4827
4828
4829
4830
4831 015500 000004
4832 015502 012767 016620 163560
4833 015510 012737 015602 000004
4834 015516 032737 000020 177776
4835 015524 001404
4836 015526 012737 000360 000006
4837 015534 000403
4838 015536 012737 000340 000006
4839 015544 012767 015552 163336
4840 015552 012706 001100
4841
4842
4843 015556 013700 160000
4844
4845 015562 032737 000020 177766
4846 015570 001002
4847 015572 104217
4848 015574 000423
4849 015576 104220
4850 015600 000421
4851 015602 022716 160000
4852 015606 001003
4853 015610 104377
4854
4855 015612 000455
4856 015614 000413
4857 015616 012737 015644 000004
4858 015624 012767 015632 163256
4859 015632 012706 001100
4860
4861
4862 015636 010037 160000
4863 015642 104221
4864
4865
4866 015644 000237
4867 015646 012767 015716 163234
4868 015654 012737 015736 000004
4869 015662 012737 015744 000240
4870 015670 032737 000020 177776
4871 015676 001404
4872 015700 012737 000360 000242
4873 015706 000403
4874 015710 012737 000340 000006
4875 015716 012706 001100
4876 015722 052737 100000 177772

:
:
: IF TMCC ABORT DOES NOT GO HIGH OR DOES NOT GET TO RACA
: OR IF RACA ZAP DOES NOT GO LOW THE PROCESSOR WILL NOT TRAP TO 4.
:
: IF BEN06 FAILS THE STACKED PC WILL BE 160000 INSTEAD OF 1$-2.
:
: IF BEN 13 FAILS EITHER TMCC AERF(1) L IS NOT GOING LOW
: OR TMCB E53(11) IS BAD.
: A TEST IS THEN MADE TO ENSURE THAT TMCC PRIORITY CLEAR GOES LOW.
:*****
TST32: SCOPE
MOV #TST33,NEXTTST ;SAVE ADDRESS OF NEXT TEST
MOV #2$,@#ERRVEC ;SETUP TIMEOUT VECTOR
BIT #BIT4,@#PSW ;IS T BIT ON?
BEQ 19$ ;BRANCH IF NO
MOV #360,@#ERRVEC+2 ;SETUP ERROR VEC FOR T BIT
BR 17$ ;
19$: MOV #PR7,@#ERRVEC+2 ;SETUP ERROR VEC WITHOUT T BIT
17$: MOV #12$, $LPERR ;SETUP ERROR LOOP
12$: MOV #STACK,SP ;INITIALIZE THE SP
:
:EXECUTE A TIMEOUT ON A DATI
MOV @#160000,RO ;EXECUTE TIMEOUT ON DATI
:FAILURE-DID NOT TIMEOUT OR AERF DID NOT GO LOW
BIT #BIT4,@#CPUERR ;DID TIMEOUT FLAG SET?
BNE 1$ ;BRANCH IF YES
ERROR 217 ;DID NOT TIMEOUT
BR 7$ ;
1$: ERROR 220 ;BEN 13 FAILED
BR 7$ ;
2$: CMP #160000,(SP) ;DID 160000 GET STACKED?
BNE 16$ ;BRANCH IF NO
ERROR 377 ;EITHER PS RESTORE STUCK HIGH
;OR RACK E63(B0) BAD
455
BR 7$
16$: MOV #7$,@#ERRVEC ;SETUP TIMEOUT VECTOR
MOV #13$, $LPERR ;SETUP ERROR LOOP
13$: MOV #STACK,SP ;INITIALIZE THE SP
:
:EXECUTE A TIMEOUT ON A DATO
MOV RO,@#160000 ;EXECUTE TIMEOUT ON DATO
ERROR 221 ;DID NOT TIMEOUT
: PRIORITY CLEAR ON ABORT
: ENSURES PIR VECTOR DOES NOT COME IN ON ABORT
7$: SPL 7 ;ENSURE CPU AT LEVEL 7
MOV #15$, $LPERR ;SETUP THE ERROR LOOP
MOV #11$,@#ERRVEC ;SETUP ERRVEC
MOV #9$,@#PIRQVEC ;SETUP PIRQ VECTOR
BIT #BIT4,@#PSW ;IS T BIT ON?
BEQ 18$ ;BRANCH IF NO
MOV #360,@#PIRQVEC+2 ;SETUP PIRQ PSW
BR 15$ ;
18$: MOV #PR7,@#ERRVEC+2 ;SETUP NEW PSW
15$: MOV #STACK,SP ;INITIALIZE THE SP
BIS #BIT15,@#PIRQ ;SET PIR LEVEL 7

```

```

4877 015730 000236          SPL      6
4878 015732 005237 160000    INC      @#160000      ;EXECUTE REFERENCE TO BAD ADDRESS
4879 015736 005037 177772    11$:    CLR      @#PIRQ      ;CLEAR PIRQ REGISTER
4880 015742 000404          BR       10$          ;SKIP OVER ERROR CALL
4881 015744 005037 177772    9$:     CLR      @#PIRQ      ;CLEAR PIRQ REG
4882 015750 104377          ERROR    377          ;TMCC PRIORITY CLEAR DID NOT GO LOW
4883 015752 000435          435
4884 015754 012737 032636 000004 10$:    MOV      #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
4885          ;CPU ERROR REGISTER BIT 4 TEST
4886 015762 022737 000020 177766    CMP      #BIT4,@#CPUERR ;DID CPU ERROR REG TIMEOUT BIT SET?
4887 015770 001407          BEQ      5$          ;BRANCH IF YES
4888 015772 013767 177766 163154    MOV      @#CPUERR,$REGO  ;SAVE REG FOR TYPEOUT
4889 016000 012767 000020 163154    MOV      #BIT4,$TMPO    ;SAVE EXPECTED VALUE
4890 016006 104255          ERROR    255          ;CPU ERROR REG FAILED TO SET
4891 016010 005037 177766    5$:     CLR      @#CPUERR      ;CLEAR OUT BIT 4
4892 016014 022737 000000 177766    CMP      #0,@#CPUERR    ;DID REGISTER CLEAR?
4893 016022 001401          BEQ      PDINTE       ;;BRANCH IF YES
4894 016024 104256          ERROR    256          ;CPU ERROR REG DOES NOT CLEAR
4895          ;*****
4896          .SBTTL PERIPHERAL DETERMINATOR & INTERRUPT ENABLE ROUTINES
4897          ;* THIS SECTION OF CODE TRYS TO FIND A DEVICE ON BR5 AND BR6.
4898          ;* WHEN IT FINDS A DEVICE IT PUTS THE ADDRESS OF THAT DEVICES
4899          ;* SUBROUTINE IN A LOCATION.
4900          ;*
4901          ;* THE CODE TO INITIATE AN INTERRUPT SEQUENCE ON CERTAIN
4902          ;* DEVICES IS ALSO HERE. WHEN A TEST REQUIRES AN INTERRUPT ON
4903          ;* A CERTAIN LEVEL IT LOOKS AT INTERX TO DETERMINE IF A
4904          ;* DEVICE IS AVAILABLE AND IF SO DOES A JSR TO THAT DEVICES
4905          ;* INTERRUPT ENABLE ROUTINE.
4906          ;*****
4907          PDINTE:
4908 016026 012767 016620 163140    MOV      #TST33,$ESCAPE  AVE START ADDRESS OF NEXT TEST
4909 016034 012767 016620 163226    MOV      #TST33,NEXTTST  ;SAVE START ADDRESS OF NEXT TEST
4910 016042 005767 163032          TST      $PASS          ;IS THIS FIRST PASS?
4911 016046 001004          BNE      1$          ;BRANCH IF NO
4912 016050 032737 040000 177570    BIT      #SW14,@#SWR     ;IS SWITCH 14 ON?
4913 016056 001402          BEQ      2$          ;BRANCH IF NO
4914 016060 000177 163204    1$:     JMP      @NEXTTST       ;GO TO NEXT TEST
4915 016064 012706 001100    2$:     MOV      #STACK,SP     ;INITIALIZE THE SP
4916 016070 012737 016104 000004    MOV      #3$,@#ERRVEC    ;SETUP ERROR VECTOR
4917 016076 005777 163136          TST      @RSCS1         ;IS RS AVAILABLE?
4918 016102 000430          BR       6$          ;YES
4919 016104 012737 016120 000004 3$:     MOV      #4$,@#ERRVEC    ;SETUP ERROR VECTOR
4920 016112 005777 163126          TST      @RPCS1         ;IS RP AVAILABLE?
4921 016116 000431          BR       7$          ;YES
4922 016120 012737 016134 000004 4$:     MOV      #5$,@#ERRVEC    ;SETUP ERROR VECTOR
4923 016126 005777 163122          TST      @TMCS1         ;IS TM AVAILABLE?
4924 016132 000432          BR       8$          ;YES
4925 016134 012737 016244 000004 5$:     MOV      #10$,@#ERRVEC   ;SETUP ERROR VECTOR
4926 016142 005777 163102          TST      @RKCS1         ;IS RK AVAILABLE?
4927 016146 016767 163076 163054    MOV      RKCS1,INT5ST    ;YES,SAVE RK STATUS
4928 016154 016767 163072 163044    MOV      RKVEC,INT5VEC   ;SAVE RK VECTOR
4929 016162 000424          BR       9$          ;EXIT
4930 016164 016767 163050 163036 6$:     MOV      RSCS1,INT5ST    ;SAVE RS STATUS
4931 016172 016767 163044 163026    MOV      RSVEC,INT5VEC   ;SAVE RS VECTOR
4932 016200 000415          BR       9$          ;EXIT
    
```

C 8

CEKBBFO 11/70 CPU #2 MACY11 30A(1052) 16-APR-80 09:11 PAGE 94
 CEKBBF.P11 07-APR-80 13:15 PERIPHERAL DETERMINATOR & INTERRUPT ENABLE ROUTINES SEQ 0093

```

4933 016202 016767 163036 163020 7$:  MOV   RPCS1,INT5ST  ;SAVE RP STATUS
4934 016210 016767 163032 163010      MOV   RPVEC,INT5VEC ;SAVE RP VECTOR
4935 016216 000406      BR    9$           ;EXIT
4936 016220 016767 163030 163002 8$:  MOV   TMCS1,INT5ST  ;SAVE TM STATUS
4937 016226 016767 163024 162772      MOV   TMVEC,INT5VEC ;SAVE TM VECTOR
4938 016234 012767 016434 162762 9$:  MOV   #INT5SU,INTER5 ;SAVE ADDRESS OF INTER 5 SUBROUTINE
4939 016242 000406      BR    LEVE6        ;GO CHECK LEVEL 6
4940 016244 032737 000440 177570 10$: BIT   #440,@#SWR    ;SWITCH 8 OR 5 ON?
4941 016252 001002      BNE   LEVE6        ;BRANCH IF YES
4942 016254 104400 072402      TYPE  ,EM710
4943
4944      ;FIND OUT WHAT IS AVAILABLE ON LEVEL 6
4945 016260 012737 016330 000004 LEVE6: MOV   #1$,@#ERRVEC ;SETUP LOCATION 4
4946 016266 005777 162766      TST   @LKSTAT      ;IS LINE CLOCK AVAILABLE?
4947 016272 012767 016500 162732      MOV   #$KW11L,INTER6 ;PUT ADDRESS OF KW11-L SUBROUTINE IN STORAGE
4948 016300 016767 162756 162726      MOV   LKVEC,INT6VEC ;SAVE BR 6 INTERR VEC
4949 016306 016767 162746 162722      MOV   LKSTAT,INT6ST ;SAVE ADDRESS OF BR 6 STATUS
4950 016314 012737 032636 000004      MOV   #CPUSPUR,@#ERRVEC
4951 016322 005037 177766      CLR   @#CPUERR     ;ENSURE TIMEOUT BIT CLEAR
4952 016326 000534      BR    TST33        ;:PERIPHERAL DETERMINATOR FINISHED
4953 016330 104400 073071 1$:  TYPE  ,EM725
4954
4955      ;LINE CLOCK NOT AVAILABLE, SEE IF PROGRAMMABLE CLOCK AVAILABLE
4956 016334 012737 016404 000004      MOV   #2$,@#ERRVEC ;SETUP LOCATION 4
4957 016342 005777 162716      TST   @PLKSTAT     ;IS PROGRAMMABLE AVAILABLE?
4958 016346 012767 016544 162656      MOV   #$KW11P,INTER6 ;YES, PUT ADDRESS OF KW11-P SUBROUTINE IN STORAGE
4959 016354 016767 162706 162652      MOV   PLKVEC,INT6VEC ;SAVE BR 6 INTERR VEC
4960 016362 016767 162676 162646      MOV   PLKSTAT,INT6ST ;SAVE ADDRESS OF BR 6 STATUS
4961 016370 005037 177766      CLR   @#CPUERR     ;ENSURE TIMEOUT BIT CLEAR
4962 016374 012737 032636 000004      MOV   #CPUSPUR,@#ERRVEC ;RESTORE ERROR VECTOR
4963 016402 000506      BR    TST33        ;:PERIPHERAL DETERMINATOR FINISHED
4964 016404 012737 032636 000004 2$:  MOV   #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
4965 016412 005037 177766      CLR   @#CPUERR     ;ENSURE TIMEOUT BIT CLEAR
4966 016416 032737 000500 177570      BIT   #500,@#SWR   ;SWITCH 6 OR 8 ON?
4967 016424 001002      BNE   3$          ;BRANCH IF YES
4968 016426 104400 072422      TYPE  ,EM711      ;TYPE MESSAGE(NO DEVICE AVAILABLE)
4969 016432
4970 016432 000472 3$:  BR    TST33        ;:PERIPHERAL DETERMINATOR FINISHED
4971
4972      ;THIS CODE SETS UP THE DEVICE ON LEVEL 5 TO INTERRUPT
4973      ;AND IS CALLED BY A JSR PC,@INTER5
4973 016434 011600 INT5SU: MOV   (SP),R0    ;SAVE RETURN PC
4974 016436 012777 016470 162562      MOV   #ENDBR5,@INT5VEC ;SETUP LEVEL 5 VECTOR
4975 016444 005001      CLR   R1          ;SETUP WAIT COUNTER
4976 016446 012777 000311 162554      MOV   #311,@INT5ST   ;SET LEVEL 5 INTERRUPT
4977 016454 005201 2$:  INC   R1          ;WAIT FOR
4978 016456 001376      BNE   2$          ;INTERRUPT
4979 016460 005077 162544      CLR   @INT5ST      ;CLEAR INTERRUPT FLAG
4980 016464 062700 000002      ADD   #2,R0        ;ADJUST RETURN PC
4981 016470 005077 162534 ENDBR5: CLR   @INT5ST  ;CLEAR LEVEL 5 STATUS
4982 016474 010046      MOV   R0,-(SP)    ;PUT RETURN PC ON STACK
4983 016476 000207      RTS   PC          ;RETURN
4984
4985      ;THIS CODE SETS UP THE KW11-L TO INTERRUPT AND IS CALLED BY A JSR PC,@INTER6
4986 016500 011600 $KW11L: MOV   (SP),R0    ;SAVE THE RETURN PC
4987 016502 012777 016536 162524      MOV   #1$,@INT6VEC  ;SETUP INTERRUPT VECTOR
4988 016510 005001      CLR   R1          ;SETUP COUNTER

```

```
4989 016512 012777 000100 162516      MOV    #BIT6,@INT6ST ;SET INTERRUPT ENABLE BIT & CLR MONITOR BIT
4990 016520 005201                2$:   INC    R1             ;WAIT FOR
4991 016522 001376                BNE   2$             ;INTERRUPT
4992 016524 005077 162506      CLR   @INT6ST       ;CLEAR INTERRUPT BIT
4993 016530 062700 000002      ADD   #2,R0         ;ADJUST RETURN PC
4994 016534 000427                BR    $ENDBR6        ;RETURN
4995 016536 005077 162474      1$:   CLR   @INT6ST       ;CLEAR INTERRUPT FLAG
4996 016542 000424                BR    $ENDBR6        ;RETURN
```

```
4997
4998
4999 016544 011600                ;THIS CODE SETS UP THE KW11-P TO INTERRUPT AND IS CALLED BY A JSR PC,@INTER6
5000 016546 012777 016610 162460 $KW11P: MOV (SP),R0 ;SAVE THE RETURN PC
5001 016554 012737 000001 172544      MOV   #1$,@INT6VEC ;SETUP THE INTERRUPT VECTOR
5002 016562 005001                MOV   #1,@#PLKC     ;SET THE COUNTER FOR ONE COUNT
5003 016564 012777 000105 162444      CLR   R1            ;SETUP THE WAIT COUNTER
5004 016572 005201                2$:   MOV   #105,@INT6ST ;START COUNTER
5005 016574 001376                INC   R1             ;WAIT FOR
5006 016576 005077 162434      BNE   2$             ;INTERRUPT
5007 016602 062700 000002      CLR   @INT6ST       ;CLEAR INTERRUPT BIT
5008 016606 000402                ADD   #2,R0         ;ADJUST R0 FOR RETURN
5009 016610 005077 162422      1$:   BR    $ENDBR6        ;RETURN
5010 016614 010046      $ENDBR6: MOV R0,-(SP) ;PUT RETURN PC ON STACK.
5011 016616 000207                RTS    PC            ;RETURN
```

```
5012
5013
5014
5015
5016
5017
5018
5019
5020
5021
5022
5023
5024
5025
5026
5027
5028
5029
5030
5031
5032
5033
5034
5035
5036
5037
5038
5039
5040
5041
5042
5043
5044
```

*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.

```
5020 016620 000004                TST33: SCOPE
5021 016622 012767 017114 162440      MOV   #TST34,NEXTTST ;SAVE ADDRESS OF NEXT TEST
5022 016630 012767 003706 162246      MOV   #^D1990,$ICNT ;ADJUST ITERATION COUNT
5023 016636 012767 016644 162242      MOV   #14,$LPADR    ;SETUP LOOP ADR
5024 016644 032737 000400 177570      14$:  BIT   #SW8,@#SWR    ;SWITCH 8 ON?
5025 016652 001012                BNE   8$             ;BRANCH IF YES
5026 016654 032737 000020 177570      BIT   #SW4,@#SWR    ;IS SWITCH 4 ON?
5027 016662 001406                BEQ   8$             ;BRANCH IF NO
5028 016664 005767 162210      TST   $PASS         ;IS THIS FIRST PASS?
5029 016670 001002                BNE   9$             ;BRANCH IF NO
5030 016672 104400 072442      TYPE  ,EM712        ;TYPE MESSAGE(SKIPPING TEST)
5031 016676                9$:
5032 016676 000506                BR    TST34         ;;GO TO NEXT TEST
5033 016700 012706 001100      8$:   MOV   #STACK,SP     ;INITIALIZE THE SP
5034 016704 012737 016760 000064      MOV   #1$,@#TPVEC   ;SETUP THE TERMINAL INTERRUPT VECTOR
5035 016712 032737 000020 177776      BIT   #BIT4,@#PSW   ;IS T BIT ON?
5036 016720 001404                BEQ   10$           ;BRANCH IF NO
5037 016722 012737 000360 000066      MOV   #360,@#TPVEC+2 ;SETUP TPVEC PSW
5038 016730 000403                BR    11$
5039 016732 012737 000340 000066      10$:  MOV   #PR7,@#TPVEC+2 ;PUT PRIORITY 7 IN NEW PSW
5040 016740 000237                11$:  SPL   7             ;SET CPU AT LEVEL 7
5041 016742 005000                CLR   R0            ;SETUP R0
5042 016744 052777 000100 162170      BIS   #100,@$TPS    ;GET INTERRUPT ON BR 4
5043 016752 005200                2$:   INC   R0            ;WAIT AND SEE
5044 016754 001376                BNE   2$            ;IF INTERRUPT OCCURS
```



```

5045 016756 000403          BR      3$          ;NO INTERRUPT
5046 016760 005077 162156 1$: CLR    @STPS      ;CLEAR INTERRUPT ENABLE
5047 016764 104222          ERROR  222         ;EITHER TMCB PS07(0) IS NOT GETTING TO
5048                                     ;TMCB E77 OR E77 IS BAD
5049 016766          3$:
5050 016766 012767 017114 162200 MOV    #TST34,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
5051 016774 012767 017114 162266 MOV    #TST34,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
5052 017002 012737 017110 000064 MOV    #4$,@#TPVEC    ;SETUP THE INTERRUPT VECTOR
5053 017010 005000          CLR    R0           ;SETUP R0
5054 017012 000233          SPL    3             ;SET CPU AT LEVEL 3
5055 017014 052777 000100 162120 BIS    #100,@STPS     ;GET A BR 4
5056 017022 005200 5$: INC    R0           ;WAIT FOR
5057 017024 001376          BNE    5$          ;INTERRUPT
5058 017026 005077 162110 CLR    @STPS        ;CLEAR TP INTERRUPT BIT
5059                                     ;BR 4 INTERRUPT FAILED. IS THERE A BR 6 DEVICE AVAILABLE?
5060 017032 005767 162174 TST    INTER6       ;TEST LEVEL 6
5061 017036 001422          BEQ    7$          ;
5062 017040 016700 162170 MOV    INT6VEC,R0    ;GET ADDR OF INTER 6 VECTOR
5063 017044 062700 000002 ADD    #2,R0         ;ADJUST TO PSW VEC
5064 017050 032737 000020 177776 BIT    #BIT4,@#PSW   ;IS T BIT ON?
5065 017056 001403          BEQ    12$         ;BRANCH IF NO
5066 017060 012710 000360 MOV    #360,(R0)     ;SETUP PSW
5067 017064 000402          BR     13$         ;
5068 017066 012710 000340 12$: MOV   #PR7,(R0)   ;SETUP PSW NO T BIT
5069 017072 000235 13$: SPL    5             ;SET CPU AT 5
5070 017074 004777 162132 JSR    PC,@INTER6    ;EXECUTE INTERRUPT
5071 017100 000402          BR     6$          ;RETURN HERE IF 6 INTERRUPTS
5072 017102 104223          ERROR  223         ;BOTH BR 4 AND BR 6 FAILED
5073 017104 104224 7$: ERROR  224         ;BR 4 FAILED
5074 017106 104225 6$: ERROR  225         ;BR 4 FAILED BUT BR 6 OK
5075 017110 005077 162026 4$: CLR    @STPS        ;CLEAR PRINTER INTERRUPT FLAG
    
```

```

*****
*TEST 34 BR LEVEL 5 INTERRUPT
*
* THE ONLY POSSIBLE FAILURE IS THAT TMCA HONOR BR 5
* DOES NOT GO LOW OR TMCB E62(6) IS BAD.
*****
    
```

```

5083 017114 000004 TST34: SCOPE
5084 017116 012767 017244 162050 MOV    #TST35,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
5085 017124 012767 017244 162136 MOV    #TST35,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
5086 017132 032737 000400 177570 BIT    #SW8,@#SWR    ;SWITCH 8 ON?
5087 017140 001012          BNE    1$          ;BRANCH IF YES
5088 017142 032737 000040 177570 BIT    #SW5,@#SWR    ;IS SWITCH 5 UP?
5089 017150 001406          BEQ    1$          ;BRANCH IF NO
5090 017152 005767 161722 TST    $PASS        ;IS THIS FIRST PASS?
5091 017156 001002          BNE    3$          ;BRANCH IF NO
5092 017160 104400 072467 TYPE    ,EM713       ;TYPE MESSAGE(TEST BEING SKIPPED)
5093 017164          3$:
5094 017164 000427          BR     TST35       ;;GO TO NEXT TEST
5095 017166 005767 162032 1$: TST    INTER5       ;IS THERE A DEVICE AVAILABLE?
5096 017172 001424          BEQ    TST35       ;;BRANCH IF NO
5097 017174 012706 001100 2$: MOV    #STACK,SP    ;INITIALIZE THE SP
5098 017200 016700 162022 MOV    INT5VEC,R0    ;GET ADDR OF BR 5 VECTOR
5099 017204 062700 000002 ADD    #2,R0         ;ADJUST TO PSW ADDR
5100 017210 032737 000020 177776 BIT    #BIT4,@#PSW   ;IS T BIT ON?
    
```

```

5101 017216 001403          BEQ      5$          ;BRANCH IF NO
5102 017220 012710 000360  MOV     #360,(R0)    ;PUT NEW PSW IN VECTOR
5103 017224 000402          BR       6$          ;
5104 017226 012710 000340  5$:    MOV     #PR7,(R0) ;PUT NEW PSW IN VEC NO T BIT
5105 017232 000234 6$:    SPL     4          ;SET CPU AT LEVEL 4
5106 017234 004777 161764  JSR     PC,@INTERS  ;EXECUTE LEVEL 5 INTERRUPT
5107 017240 000401          BR       TST35       ;GO TO NEXT TEST
5108 017242 104226          ERROR   226         ;BR 5 DID NOT INTERRUPT
5109

```

```

5110
5111      ;*****
5112      ;*TEST 35      BR LEVEL 6 INTERRUPT
5113      ;*
5114      ;*      THE ONLY POSSIBLE FAILURE IS THAT TMCA HONOR BR 6 DOES NOT GO LOW
5115      ;*      OR TMCB E62(12) IS BAD.
5116      ;*****

```

```

5116 017244 000004          TST35: SCOPE
5117 017246 012767 017374 161720  MOV     #TST36,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
5118 017254 012767 017374 162006  MOV     #TST36,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
5119 017262 032737 000400 177570  BIT     #SW8,@#SWR     ;SWITCH 8 ON?
5120 017270 001012          BNE     1$          ;BRANCH IF YES
5121 017272 032737 000100 177570  BIT     #SW6,@#SWR     ;IS SWITCH 6 UP?
5122 017300 001406          BEQ     1$          ;BRANCH IF NO
5123 017302 005767 161572  TST     $PASS         ;IS THIS FIRST PASS?
5124 017306 001002          BNE     3$          ;BRANCH IF NO
5125 017310 104400 072514  TYPE   ,EM715        ;TYPE MESSAGE(TEST BEING SKIPPED)
5126 017314          3$:
5127 017314 000427          BR      TST36        ;;GO TO NEXT TEST
5128 017316 005767 161710  1$:    TST     INTER6       ;IS THERE A DEVICE AVAILABLE?
5129 017322 001424          BEQ     TST36        ;;BRANCH IF NO
5130 017324 012706 001100  2$:    MOV     #STACK,SP    ;INITIALIZE THE SP
5131 017330 016700 161700  MOV     INT6VEC,R0    ;GET ADR OF BR 6 VECTOR
5132 017334 062700 000002  ADD     #2,R0         ;ADJUST TO PSW ADDR
5133 017340 032737 000020 177776  BIT     #BIT4,@#PSW   ;IS T BIT ON?
5134 017346 001403          BEQ     5$          ;BRANCH IF NO
5135 017350 012710 000360  MOV     #360,(R0)    ;SETUP NEW PSW
5136 017354 000402          BR      6$          ;
5137 017356 012710 000340  5$:    MOV     #PR7,(R0)    ;SETUP NEW PSW
5138 017362 000235 6$:    SPL     5          ;SET CPU AT LEVEL 5
5139 017364 004777 161642  JSR     PC,@INTER6   ;EXECUTE LEVEL 6 INTERRUPT
5140 017370 000401          BR      TST36        ;;GO TO NEXT TEST
5141 017372 104227          ERROR   227         ;BR 6 DID NOT INTERRUPT
5142

```

```

5143      ;*****
5144      ;*TEST 36      YELLOW ZONE TRAP
5145      ;*
5146      ;*      A YELLOW ZONE IS FIRST ATTEMPTED WITH THE SP AT 376.
5147      ;*      IF BEN 13 FAILS THE TRAP WILL NOT OCCUR.
5148      ;*
5149      ;*      IF THE PROCESSOR FAILS TO TRAP EITHER TMCD SL YEL IS
5150      ;*      NOT GOING HIGH OR TMCA HONOR SLY IS NOT GOING LOW
5151      ;*      OR TMCB E70(3) IS BAD OR BEN13 FAILED.
5152      ;*
5153      ;*      IF TMCC PRIORITY CLEAR DOES NOT GO LOW THE PROCESSOR WILL HANG
5154      ;*      UP IN A RED ZONE TRAP LOOP.
5155      ;*
5156      ;*      A JSR WITH A BAD SP IS THEN EXECUTED TO ENSURE TMCC

```

```

5157      ;*      KERNAL R6 GOES HIGH WHEN ENABLED BY "STACK REFERENCE * KERNAL MODE".
5158      ;*
5159      ;*      IF THE TRAP WORKS TESTS WILL BE PERFORMED TO ENSURE ALL THE
5160      ;*      APPROPRIATE CONDITIONS DISABLE THE TRAP EXCEPT
5161      ;*      THE PRIORITY ARBITRATOR.
5162      ;*
5163      ;*****
5163 017374 000004 TST36: SCOPE
5164 017376 012767 017442 161502      MOV      #11$, $LPADR      ;SETUP LP ADR
5165 017404 012767 017442 161476      MOV      #11$, $LPERR     ;SETUP ERROR LOOP
5166 017412 013767 177776 161550      MOV      @#PSW, $TMP3     ;SAVE PSW
5167 017420 032737 000020 177776      BIT      #BIT4, @#PSW     ;IS T BIT ON?
5168 017426 001405                      BEQ      11$              ;BRANCH IF NO
5169 017430 012746 000340                      MOV      #PR7, -(SP)      ;PUT NEW PSW ON STACK
5170 017434 012746 017442                      MOV      #11$, -(SP)      ;PUT RETURN ADR ON STACK
5171 017440 000006                      RTT                          ;TURN T BIT OFF
5172 017442 012737 000340 000006 11$:  MOV      #PR7, @#ERRVEC+2  ;RESTORE ERR VEC PSW
5173 017450 012737 017506 000004      MOV      #1$, @#ERRVEC    ;SETUP ERRVEC
5174 017456 012767 017464 161424      MOV      #2$, $LPERR     ;SETUP ERROR LOOP
5175 017464 012706 000376 2$:      MOV      #376, SP        ;SETUP THE SP
5176 017470 012700 177777      MOV      #-1, R0         ;SETUP R0
5177 017474 010016                      MOV      R0, (SP)        ;EXECUTE INSTRUCTION UNDER TEST
5178 017476 012706 001100                      MOV      #STACK, SP     ;REINITIALIZE THE SP
5179 017502 104230                      ERROR    230            ;YELLOW ZONE DID NOT OCCUR
5180 017504 000407                      BR      8$              ;
5181 017506 022737 177777 000376 1$:  CMP      #-1, @#376      ;DID RED ZONE OCCUR?
5182 017514 001403                      BEQ      8$              ;BRANCH IF NO
5183 017516 012706 001100                      MOV      #STACK, SP     ;RESTORE SP
5184 017522 104254                      ERROR    254            ;RED ZONE IN YELLOW REGION
5185
5186      ;JSR WITH A BAD SP
5187 017524 012737 017564 000004 8$:  MOV      #6$, @#ERRVEC    ;SETUP ERRVEC
5188 017532 012767 017540 161350      MOV      #63$, $LPERR    ;SETUP THE ERROR LOOP
5189 017540 012706 000376 63$:  MOV      #376, SP        ;SETUP THE SP
5190 017544 004767 000000                      JSR      PC, 7$         ;EXECUTE INSTRUCTION UNDER TEST
5191 017550 012706 001100 7$:  MOV      #STACK, SP     ;RESET THE SP
5192 017554 012737 032636 000004      MOV      #CPUSPUR, @#ERRVEC ;RESTORE ERRVEC
5193 017562 104232                      ERROR    232            ;TMCC KERNAL R6 DID NOT GO HIGH ON JSR
5194
5195      ;DISABLE WITH STACK LIMIT REGISTER
5196 017564 012737 017612 000004 6$:  MOV      #3$, @#ERRVEC    ;SETUP ERRVEC
5197 017572 012767 017600 161310      MOV      #62$, $LPERR    ;SETUP ERROR LOOP
5198 017600 012706 000740 62$:  MOV      #740, SP        ;SETUP THE SP
5199 017604 013716 000742                      MOV      @#742, (SP)     ;EXECUTE INSTRUCTION UNDER TEST
5200 017610 000406                      BR      4$              ;GO TO NEXT SECTION
5201 017612 012737 032636 000004 3$:  MOV      #CPUSPUR, @#ERRVEC ;RESTORE ERRVEC
5202 017620 012706 001100                      MOV      #STACK, SP     ;RESTORE THE SP
5203 017624 104233                      ERROR    233            ;PDRK STACK LIMIT DID NOT DISABLE TRAP
5204
5205      ;DISABLE WITH TMCC KERNAL R6
5206      ;DISABLE KERNAL R6 WITH DAT1
5207 017626 012737 017652 000004 4$:  MOV      #5$, @#ERRVEC    ;SETUP ERRVEC
5208 017634 012767 017642 161246      MOV      #61$, $LPERR    ;SETUP THE ERROR LOOP
5209 017642 012706 000376 61$:  MOV      #376, SP        ;SETUP THE SP
5210 017646 011606                      MOV      (SP), SP       ;EXECUTE INSTRUCTION UNDER TEST
5211 017650 000415                      BR      9$              ;GO TO NEXT SECTION
5212 017652 012706 001100 5$:  MOV      #STACK, SP     ;RESTORE THE SP
    
```

```

5213 017656 012737 032636 000004      MOV    #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5214 017664 104234      ERROR  234                ;TMCC KERNAL R6 DID NOT DISABLE ON DATI
5215 017666 000406      BR     9$
5216
5217      ;USED FOR ERROR LOOP IF BIT 3 IN ERROR REG
5218      ;DOES NOT SET
5219 017670 012706 000376      60$:  MOV    #376,SP          ;SETUP THE SP
5220 017674 012737 017704 000004      MOV    #9$,@#ERRVEC      ;SETUP ERROR VECTOR
5221 017702 005016      CLR    (SP)              ;CAUSE TRAP
5222 017704 012767 017670 161176      9$:  MOV    #60$, $LPERR      ;SETUP ERROR LOOP
5223 017712 012706 001100      MOV    #STACK,SP        ;RESTORE THE SP
5224 017716 022737 000010 177766      CMP    #BIT3,@#CPUERR    ;DID YEL ZONE BIT SET?
5225 017724 001407      BEQ    10$              ;BRANCH IF YES
5226 017726 013767 177766 161220      MOV    @#CPUERR,$REGO    ;SAVE FOR TYPEOUT
5227 017734 012767 000010 161220      MOV    #BIT3,$TMP0      ;SAVE EXPECTED VALUE
5228
5229 017742 104257      ERROR  257                ;YEL ZONE BIT DID NOT SET IN CPUERR
5230 017744 012767 017752 161136      10$: MOV    #57$, $LPERR     ;SETUP ERROR LOOP
5231 017752 005037 177766      57$: CLR    @#CPUERR      ;CLEAR YEL ZONE BIT
5232 017756 005737 177766      TST    @#CPUERR          ;DID IT CLEAR?
5233 017762 001401      BEQ    TST37            ;BRANCH IF YES
5234 017764 104260      ERROR  260                ;BIT3 DID NOV CLEAR IN CPUERR
5235
5236      ;*****
5237      ;*TEST 37          ROM FIELD CHECK OF PC MANIPULATOR STATES
5238      ;*
5239      ;* THIS TEST EXECUTES THE MACHINE STATES THAT MANIPULATE
5240      ;* THE PC TO ENSURE THAT THE PCB ROM FIELD OR DRX ROM FIELD
5241      ;* OR SRX ROM FIELD OF THESE STATES IS FUNCTIONAL.
5242      ;* THESE STATES ARE S13.00, S45.00, MTP.10, D45.80, D45.90,
5243      ;* D45.00, AND D45.01.
5244      ;*****
5244 017766 000004      TST37: SCOPE
5245 017770 012767 020020 161110      MOV    #18$, $LPADR      ;SETUP LOOP ADR
5246 017776 032767 000020 161164      BIT    #BIT4,$TMP3      ;WAS T BIT ON?
5247 020004 001405      BEQ    18$              ;BRANCH IF NO
5248 020006 012746 000360      MOV    #360,-(SP)        ;PUT NEW PSW ON STACK
5249 020012 012746 020020      MOV    #18$,-(SP)        ;PUT RETURN ADDR ON STACK
5250 020016 000006      RTT                    ;TURN T BIT ON
5251
5252 020020 012767 020026 161062      ;BIN*SM1*SF7*DM0(S13.00)
5253 020026 011700      18$:  MOV    #64$, $LPERR   ;SETUP ERROR LOOP
5254 020030 020027 020027      64$:  MOV    (PC),R0        ;EXECUTE INSTRUCTION UNDER TEST
5255 020034 001402      CMP    R0,#20027         ;DID TEST WORK
5256 020036 104377      BEQ    1$              ;BRANCH IF YES
5257 020040 000443      ERROR  377              ;STATE S13.00 FAILED
5258      443
5259 020042 012767 020050 161040      ;BIN*SM4*SF7*DM2*DF7 (S45.00)
5260 020050 012767 024727 000000      1$:  MOV    #63$, $LPERR   ;SETUP ERROR LOOP
5261 020056 024727      63$:  MOV    #24727,2$     ;PUT INSTRUCTION IN 2$
5262 020060 000240      2$:  CMP    -(PC),(PC)+    ;EXECUTE INSTRUCTION UNDER TEST
5263 020062 001402      NOP                    ;USED TO BE SAFE
5264 020064 104377      BEQ    3$              ;BRANCH IF TEST OK
5265 020066 000444      ERROR  377              ;STATE S45.00 FAILED
5266      444
5267 020070 012767 020076 161012      ;MTP*DM2*DF7 (MTP.10)
5268 020076 012706 001100      3$:  MOV    #62$, $LPERR   ;SETUP ERROR LOOP
5268      62$:  MOV    #STACK,SP    ;INITIALIZE THE SP
    
```

```

5269 020102 012746 177777      MOV      #-1,-(SP)      ;SET 1076 TO ALL ONES
5270 020106 005067 000002      CLR      4$           ;ENSURE 4$ CLEAR
5271 020112 006627      MTP1    (PC)+        ;EXECUTE INSTRUCTION UNDER TEST
5272 020114 000000      4$:      .WORD      0
5273 020116 022767 177777 177770      CMP      #-1,4$      ;DID INSTRUCTION WORK?
5274 020124 001402      BEQ     7$           ;BRANCH IF YES
5275 020126 T04377      ERROR   377        ;STATE MTP.10 FAILED
5276 020130 000445      445
5277      ;BIN*SM2*SF7*DM4*DF7 (D45.80)
5278 020132 012767 020140 160750      7$:      MOV      #61$,$LPERR  ;SETUP ERROR LOOP
5279 020140 012747      61$:     MOV      (PC)+,-(PC) ;EXECUTE INSTRUCTION UNDER TEST
5280 020142 000402      BR      10$        ;WILL EXECUTE THIS IF INSTR. WORKS
5281 020144 T04377      ERROR   377        ;STATE D45.80 FAILED
5282 020146 000447      447
5283      ;BIN*SM1*SFO*DM4*DF7 (D45.90)
5284 020150 012767 020156 160732      10$:    MOV      #60$,$LPERR  ;SETUP ERROR LOOP
5285 020156 012767 141047 000024      60$:    MOV      #141047,11$ ;SETUP INSTRUCTION TO EXECUTE
5286 020164 012700 001162      MOV     #$TMP0,R0   ;PUT ADDRESS OF $TMP0 IN R0
5287 020170 005200      INC     R0
5288 020172 112710 000002      MOV     #BIT1,(R0)  ;SET BIT1 IN $TMP1 HIGH BYTE
5289 020176 012705 001166      MOV     #$TMP2,R5   ;SETUP R5
5290 020202 012767 001000 160754      MOV     #BIT9,$TMP1 ;SETUP $TMP1
5291 020210 141047      11$:    BICB   (R0),-(PC)  ;EXECUTE INSTRUCTION UNDER TEST
5292 020212 005767 160746      TST     $TMP1      ;DID $TMP1 GET CLEARED?
5293 020216 001402      BEQ     12$        ;BRANCH IF YES
5294 020220 T04377      ERROR   377        ;STATE D45.00 FAILED
5295 020222 000450      450
5296      ;DAC*DM4*DF7 (D45.00)
5297 020224 012767 020232 160656      12$:    MOV      #57$,$LPERR  ;SETUP ERROR LOOP
5298 020232 012767 140047 000016      57$:    MOV      #140047,13$ ;SETUP INSTRUCTION TO EXECUTE
5299 020240 012700 000002      MOV     #BIT1,R0   ;SETUP R0 TO CHANGE DR FROM 7 TO 5
5300 020244 012705 001165      MOV     #$TMP1+1,R5 ;PUT ADDRESS OF $TMP1 HIGH BYTE IN R5
5301 020250 012767 000002 160706      MOV     #BIT1,$TMP1 ;SET UP $TMP1 SO INSTRUCTION CLEARS IT
5302 020256 140047      13$:    BICB   R0,-(PC)   ;EXECUTE INSTRUCTION UNDER TEST
5303 020260 005767 160700      TST     $TMP1      ;DID $TMP1 CLEAR?
5304 020264 001402      BEQ     14$        ;BRANCH IF YES
5305 020266 T04377      ERROR   377        ;STATE D45.90 FAILED
5306 020270 000451      451
5307      ;DAC*DM5*DF7 (D45.01)
5308 020272 012767 020300 160610      14$:    MOV      #56$,$LPERR  ;SETUP ERROR LOOP
5309 020300 012767 130057 000052      56$:    MOV      #130057,15$ ;SETUP INSTRUCTION TO EXECUTE
5310 020306 032737 000020 177776      BIT     #BIT4,@#PSW ;IS T BIT ON?
5311 020314 001404      BEQ     20$        ;BRANCH IF NO
5312 020316 012737 000360 000066      MOV     #360,@#TPVEC+2 ;SETUP TP VEC PSW
5313 020324 000403      BR      17$
5314 020326 012737 000340 000066      20$:    MOV     #PR7,@#TPVEC+2
5315 020334 012737 020374 000064      17$:    MOV     #16$,@#TPVEC ;SETUP BR4 INTERRUPT VECTOR
5316 020342 000233      SPL     3           ;SET PROCESSOR PRIORITY BELOW BR4
5317 020344 152777 000100 160570      BISB   #BIT6,@$TPS  ;SET INTERRUPT BIT
5318 020352 012777 000015 160564      MOV     #15,@$TPB   ;SEND CHARACTER TO PRINTER
5319 020360 130057      15$:    BITB   R0,@-(PC)   ;EXECUTE INSTRUCTION UNDER TEST
5320 020362 142777 000100 160552      BICB   #BIT6,@$TPS ;WILL EXECUTE IF STATE FAILS
5321 020370 T04377      ERROR   377        ;STATE D45.01 FAILED
5322 020372 000452      452
5323 020374 142777 000100 160540      16$:    BICB   #BIT6,@$TPS ;TEST OK, CLEAR INTERRUPT BIT
5324      ;CONTINUE
    
```

5325
5326
5327
5328
5329
5330
5331
5332
5333
5334
5335
5336
5337
5338
5339
5340
5341
5342
5343
5344
5345
5346
5347
5348
5349
5350
5351
5352
5353
5354
5355
5356
5357
5358
5359
5360
5361
5362
5363
5364
5365
5366
5367
5368
5369
5370
5371
5372
5373
5374
5375
5376
5377
5378
5379
5380

```
*****  
*TEST 40 RED ZONE TRAP  
*  
* A RED ZONE TRAP IS FIRST ATTEMPTED WITH THE SP AT 336.  
* IF BEN13 FAILS EXECUTION WILL GO TO EITHER BRK.80 OR BRK.20  
* OR PUP.00.  
* BRK.80 WILL CAUSE THE OLD PSW AND PC TO BE STACKED ON THE  
* OLD STACK INSTEAD OF LOCATIONS 2 AND 0.  
* BRK.20 WILL MAKE IT LOOK LIKE THE RED ZONE FAILED.  
* PUP.00 WILL CAUSE A TRAP TO LOCATION 24.  
*  
* IF THE PROCESSOR FAILS TO TRAP EITHER TMCD SL RED IS  
* NOT GOING LOW OR TMCC ABORT IS NOT GOING LOW.  
*  
* IF UBCB ABORT RESTART FAILS TO GO LOW OR E10(13)  
* IS BAD THE PROCESSOR WILL HANG IN THE PAUSE STATE.  
*****  
TST40: SCOPE  
MOV #TST41,NEXTTST ;SAVE ADDRESS OF NEXT TEST  
MOV #14$, $LPADR ;SETUP LOOP ADR  
MOV @#PSW,$TMP3 ;SAVE PSW  
BIT #BIT4,@#PSW ;IS T BIT ON?  
BEQ 14$ ;BRANCH IF NO  
MOV #PR7,-(SP) ;PUT NEW PSW ON STACK  
MOV #14$,-(SP) ;PUT RETURN ADR ON STACK  
RTT ;TURN T BIT OFF  
14$: MOV #PR7,@#ERRVEC+2 ;RESTORE ERRVEC PSW  
MOV #64$, $LPERR ;SETUP ERROR LOOP  
MOV #3$, @#PWRVEC ;SETUP LOCATION 24  
64$: MOV #1$, @#ERRVEC ;SETUP ERRVEC  
MOV #336,SP ;SET THE SP TO RED ZONE  
MOV #-1,R0 ;SETUP R0  
MOV R0,(SP) ;EXECUTE THE TRAP INSTRUCTION  
6$: MOV #STACK,SP ;RESET THE SP  
MOV #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC  
ERROR 235 ;RED ZONE REFERENCE FAILED TO TRAP  
BR 8$  
3$: MOV #STACK,SP ;RESET THE SP  
MOV #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4  
ERROR 237 ;BEN 13 FAILED TO PUP.00  
BR 8$  
1$: CMP @#0,#6$ ;DID BEN 13 FAIL?  
BEQ 7$ ;BRANCH IF NO  
MOV #STACK,SP ;RESET THE SP  
MOV #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC  
ERROR 240 ;BEN 13 FAILED TO BRK.80  
BR 8$  
7$: CMP #-1,@#336 ;DID YEL ZONE OCCUR?  
BNE 8$ ;BRANCH IF NO  
CLR @#336 ;SETUP FOR LOOPING  
ERROR 251 ;YEL ZONE IN RED REGION  
*  
*TEST TO ENSURE PSW REFERENCE VIA THE SP CAUSES A RED ZONE TRAP  
8$: MOV #63$, $LPERR ;SETUP ERROR LOOP  
63$: MOV #4$, @#ERRVEC ;SETUP ERRVEC
```

```

5381 020626 012706 177776      MOV      #PSW,SP      ;PUT ADDRESS OF PSW IN SP
5382 020632 005016              CLR      (SP)         ;EXECUTE THE TRAP CAUSING INSTRUCTION
5383 020634 012706 001076      MOV      #1076,SP    ;RESET THE SP
5384 020640 012737 032636 0C0004  MOV      #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5385 020646 104241              ERROR    241          ;NO RED ZONE ON STACK OVERFLOW
5386
5387      ;TEST TO ENSURE SL REG GREATER THAN BADRR CAUSES A RED ZONE
5388 020650 012767 020656 160232 4$:      MOV      #62$, $LPERR ;SETUP ERROR LOOP
5389 020656 012737 020720 000004 62$:      MOV      #5$, @#ERRVEC ;SETUP RESVEC
5390 020664 012737 000400 177774      MOV      #400, @#STKLMT ;SET STACK LIMIT REGISTER TO 400
5391 020672 012706 000336      MOV      #336, SP    ;SET THE SP
5392 020676 005016              CLR      (SP)         ;EXECUTE THE TRAP CAUSING INSTRUCTION
5393 020700 012706 001100      MOV      #STACK, SP  ;RESET THE SP
5394 020704 005037 177774      CLR      @#STKLMT    ;ENSURE SL REG. CLEAR
5395 020710 012737 032636 000004  MOV      #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5396 020716 104242              ERROR    242          ;NO RED ZONE WHEN SL REG>BUS ADDR
5397
5398      ;TEST OF TMCD YEL ZONE (E15)
5399 020720 012767 020732 160162 5$:      MOV      #61$, $LPERR ;SETUP ERROR LOOP
5400 020726 005037 177774      CLR      @#STKLMT    ;ENSURE SL CLEAR
5401 020732 012737 020752 000004 61$:      MOV      #9$, @#ERRVEC ;SETUP ERRVEC
5402 020740 012706 000240      MOV      #240, SP    ;SETUP THE SP
5403 020744 012700 177777      MOV      #-1, R0     ;SETUP R0
5404 020750 010016              MOV      R0, (SP)    ;EXECUTE THE TRAP CAUSING INSTRUCTION
5405 020752 022737 177777 000240 9$:      CMP      #-1, @#240  ;DID YEL ZONE OCCUR?
5406 020760 001010              BNE     10$         ;BRANCH IF NO
5407 020762 012706 001100      MOV      #STACK, SP ;RESTORE THE SP
5408 020766 012737 032636 000004  MOV      #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5409 020774 005037 000240      CLR      @#240      ;FOR LOOPING
5410 021000 104252              ERROR    252          ;TMCD YEL ZONE DID NOT GO LOW
5411 021002 012767 021010 160100 10$:      MOV      #60$, $LPERR ;SETUP ERROR LOOP
5412 021010 012737 021024 000004 60$:      MOV      #11$, @#ERRVEC ;SETUP ERRVEC
5413 021016 012706 000140      MOV      #140, SP   ;SETUP THE SP
5414 021022 010016              MOV      R0, (SP)   ;EXECUTE THE TRAP CAUSING INSTR.
5415 021024 022737 177777 000140 11$:      CMP      #-1, @#140 ;DID YEL ZONE OCCUR?
5416 021032 001010              BNE     12$         ;BRANCH IF NO
5417 021034 012706 001100      MOV      #STACK, SP ;RESTORE SP
5418 021040 012737 032636 000004  MOV      #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5419 021046 005037 000140      CLR      @#140      ;FOR LOOPING
5420 021052 104253              ERROR    253          ;TMCD YEL ZONE DID NOT GO LOW
5421 021054 012706 001100      MOV      #STACK, SP ;RESTORE SP
5422 021060 012737 032636 000004  MOV      #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5423 021066 022737 000004 177766      CMP      #BIT2, @#CPUERR ;DID RED ZONE BIT IN CPU ERROR SET?
5424 021074 001407              BEQ     13$         ;BRANCH IF YES
5425 021076 013767 177766 160050      MOV      @#CPUERR, $REGO ;SAVE FOR TYPEOUT
5426 021104 012767 000004 160050      MOV      #BIT2, $TMPO ;SAVE EXPECTED VALUE
5427 021112 104261              ERROR    261          ;RED ZONE BIT IN CPU ERROR DID NOT SET
5428 021114 012767 021122 157766 13$:      MOV      #57$, $LPERR ;SETUP ERROR LOOP
5429 021122 005037 177766 57$:      CLR      @#CPUERR   ;CLEAR RED ZONE BIT
5430 021126 005737 177766      TST     @#CPUERR   ;DID REG CLEAR?
5431 021132 001401              BEQ     TST41       ;BRANCH IF YES
5432 021134 104262              ERROR    262          ;RED ZONE BIT DID NOT CLEAR
5433
5434      ;*****
5435      ;*TEST 41 BIT TEST OF STACK LIMIT REGISTER
5436      ;*
```

```

5437      ;*      FIRST A 125252 AND 52525 PATTERN IS PUT IN THE REGISTER TO ENSURE
5438      ;*      THAT THE REGISTER DOESN'T HAVE ANY STUCK BITS AND THAT
5439      ;*      THE DMUX SELECT AND INPUT LINES WORK.
5440      ;*
5441      ;*
5442      ;*      IF SCCE SL ADRS DOES NOT GET TO TMCD OR IF TMCD E28 OR E14
5443      ;*      IS BAD THE BR WILL BE SELECTED.  THE PB REGISTER IS LOADED
5444      ;*      WITH 200 SO IF TMCD LO BYTE EN DOES NOT GO LOW AN
5445      ;*      ERROR WILL BE DETECTED.
5446      ;*      *****
5446 021136 000004 TST41: SCOPE
5447 021140 012767 021402 160026 MOV #TST42,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
5448 021146 012767 021402 160114 MOV #TST42,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
5449 021154 012737 036250 000024 MOV #SPWRDN,@#PWRVEC ;RESTORE POWER VECTOR
5450 021162 005005 CLR R5 ;CLEAR R5
5451 021164 012737 125252 177774 MOV #125252,@#STKLMT ;PUT PATTERN IN STACK LIMIT REG
5452 021172 012737 000200 177770 MOV #200,@#177770 ;PUT 200 IN PB REGISTER
5453 021200 012700 125000 MOV #125000,R0 ;SETUP R0 TO LOOK LIKE STK LIMIT
5454 021204 000237 SPL 7 ;PUT PRIORITY BITS IN KNOWN CONFIGURATION
5455 021206 020037 177774 CMP R0,@#STKLMT ;EXECUTE TEST ON STKLMT REG.
5456 021212 001404 BEQ 1$ ;BRANCH IF TEST OK
5457 021214 005205 INC R5 ;INCREMENT TEST FAIL INDICATOR
5458 021216 013767 177774 157774 MOV @#STKLMT,E1STKLM ;SAVE ERROR VALUE
5459 021224 012737 052400 177774 1$: MOV #52400,@#STKLMT ;PUT COMPLEMENT PATTERN REG.
5460 021232 012700 052400 MOV #52400,R0 ;PUT IN R0
5461 021236 020037 177774 CMP R0,@#STKLMT ;EXECUTE TEST ON REG.
5462 021242 001415 BEQ 2$ ;BRANCH IF TEST OK
5463 021244 005705 TST R5 ;DID FIRST TEST FAIL?
5464 021246 001023 BNE 3$ ;BRANCH IF YES
5465 021250 013767 177774 157744 MOV @#STKLMT,E2STKLM ;SAVE ERROR VALUE
5466 021256 012767 052400 157676 MOV #52400,$TMPO ;SAVE EXPECTED VALUE
5467 021264 005037 177774 CLR @#STKLMT ;CLEAR THE REG
5468 021270 012706 001100 MOV #STACK,SP ;RESTORE THE SP
5469 021274 104243 ERROR 243 ;52400 PATTERN FAILED
5470 021276 005705 2$: TST R5 ;DID FIRST TEST FAIL?
5471 021300 001436 BEQ 4$ ;BRANCH IF NO
5472 021302 012767 125000 157652 MOV #125000,$TMPO ;SAVE EXPECTED VALUE
5473 021310 005037 177774 CLR @#STKLMT ;CLEAR REG.
5474 021314 104244 ERROR 244 ;125252 PATTERN FAILED
5475 021316 005037 177774 3$: CLR @#STKLMT ;CLEAR STACK LIMIT REG
5476 021322 026727 157672 013767 CMP E1STKLM,#13767 ;DID BR GET SELECTED ON STACK LIMIT REF.?
5477 021330 001001 BNE 5$ ;BRANCH IF NO
5478 021332 104245 ERROR 245 ;BR SELECTED BY DMUX
5479 021334 026727 157660 125200 5$: CMP E1STKLM,#125200 ;DID PB GET GATED ALSO?
5480 021342 001001 BNE 6$ ;BRANCH IF NO
5481 021344 104246 ERROR 246 ;TMCD LO BYTE EN DOES NOT GO LOW
5482 021346 022767 000340 157644 6$: CMP #340,E1STKLM ;DID PS GET SELECTED?
5483 021354 001001 BNE 7$ ;BRANCH IF NO
5484 021356 104247 ERROR 247 ;D MUX SELECTED PSW
5485 021360 012767 125000 157574 7$: MOV #125000,$TMPO
5486 021366 012767 052400 157570 MOV #52400,$TMP1
5487 021374 104250 ERROR 250 ;BOTH PATTERNS FAILED BUT DON'T KNOW WHY
5488 021376 005037 177774 4$: CLR @#STKLMT ;CLEAR THE STACK LIMIT REG.
5489
5490      ;*      *****
5491      ;*      TEST 42 SL REGISTER COMPARATOR TEST 1
5492      ;*
    
```



```

5493      : * THIS TEST RUNS A HIGH BYTE COUNT PATTERN THRU THE BUS
5494      : * ADDRESS MUX FOR EACH PATTERN OF THE STACK LIMIT REGISTER.
5495      : * FOR EACH PATTERN OF ADDRESSES THERE WILL BE ONE YEL TRAP
5496      : * AT THE ADDRESS CORRESPONDING TO THE SL REG+340 AND A RED
5497      : * TRAP AT EVERY ADDRESS BELOW THIS.
5498      : * THIS TEST ONLY TESTS ADDRESSES UP TO THE I/O PAGE.
5499      : * THE I/O PAGE ADDRESSES WILL BE TESTED SEPARATELY WITH
5500      : * MEMORY MANAGEMENT ENABLED AND THE I/O PAGE MAPPED INTO RESIDENT
5501      : * MEMORY
5502      :
5503      : * THE FOLLOWING ARE THE TYPES OF ERRORS THAT CAN OCCUR IN THIS TEST:
5504      : *
5505      : *          TYPE          DESCRIPTION
5506      : *          0             RED ZONE TRAP ON YELLOW ZONE ADDRESS
5507      : *          2             RED ZONE TRAP ON LEGAL ADDRESS
5508      : *          4             YELLOW ZONE TRAP ON RED ZONE ADDRESS
5509      : *          6             YELLOW ZONE TRAP ON LEGAL ADDRESS
5510      : *          10            NO TRAP ON RED ZONE ADDRESS
5511      : *          12            NO TRAP ON YELLOW ZONE ADDRESS
5512      : *
5513      : * THE LOW BYTE ADDRESS IN THE STACK POINTER IS ALWAYS
5514      : * 340 AND WILL NOT BE TYPED ON AN ERROR.
5515      : * *****
5515      021402 000004 TST42: SCOPE
5516      : * *****
5517      : * NOTE: IF THE LOOP ON ERROR SWITCH IS UP (SWITCH 9) THE TEST WILL
5518      : * LOOP ON THE FIRST ERROR WITH NO ERROR TYPEOUT. OTHERWISE ALL
5519      : * ERRORS WILL BE RECORDED IN A TABLE AND TYPED OUT AT THE
5520      : * END OF THE TEST.
5521      : * IF SWITCH 3 (DISABLE MEMORY MANAGEMENT TESTS) IS NOT ON,
5522      : * THIS TEST IS SKIPPED AND TEST 70 WILL EXECUTE.
5523      : * *****
5524      021404 012767 022012 157562      MOV      #TST43,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
5525      021412 012767 022012 157650      MOV      #TST43,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
5526      021420 012767 003706 157456      MOV      #^D1990,$ICNT ;SETUP ITERATION COUNT
5527      021426 012767 021434 157452      MOV      #13$,$LPADR ;SETUP LOOP ADDRESS
5528      021434 032737 000010 177570      13$: BIT   #BIT3,@#SWR ;IS SWITCH 3 ON?
5529      021442 001002                BNE     11$ ;BRANCH IF YES
5530      021444 000177 157524                JMP     @ $ESCAPE ;GO TO NEXT TEST
5531      021450 012737 021560 000004      11$: MOV   #1$,@#ERRVEC ;SETUP ERRVEC
5532      021456 005067 157504                CLR     $TMP2 ;INITIALIZE ERROR OVERFLOW FLAG
5533      021462 012703 000340                MOV     #340,R3 ;SET SOB COUNT FOR SL REGISTER
5534      021466 012700 120000                MOV     #120000,R0 ;INITIALIZE ERROR DATA POINTER
5535      021472 005060 000002                CLR     2(R0) ;INITIALIZE ERROR DATA BUFFER
5536      021476 012701 000340                MOV     #340,R1 ;INITIALIZE YELLOW ZONE ADDRESS(TRAP CASE)
5537      021502 012704 000344                MOV     #344,R4 ;SETUP YELLOW ZONE ADDR(NO TRAP)
5538      021506 012702 000340      2$: MOV   #340,R2 ;SET SOB COUNT FOR SP
5539      021512 012705 000344                MOV     #344,R5 ;INITIALIZE SECONDARY STORAGE FOR SP
5540      021516 016567 177776 157436      3$: MOV   -2(R5),$TMP0 ;SAVE WORDS AT
5541      021524 016567 177774 157432      MOV   -4(R5),$TMP1 ;STACK UNDER TEST
5542      021532 010506                MOV     R5,R6 ;SET THE SP
5543      021534 011616                MOV     (R6),(R6) ;EXECUTE TEST INSTRUCTION
5544      : NO TRAP. DETERMINE IF THIS IS CORRECT.
5545      021536 020604                CMP     R6,R4 ;IS ADDRESS > YELL ZONE BOUNDRY?
5546      021540 101066                BHI     5$ ;BRANCH IF YES
5547      021542 001403                BEQ     6$ ;BRANCH IF ADDRESS = YELL ZONE BOUNDRY
5548      : NO TRAP ADDRESS IS LESS THAN YELLOW ZONE BOUNDRY
    
```

```

5549 021544 052710 000010          BIS    #BIT3,(R0)    ;SET ERROR TYPE IN DATA BUFFER
5550 021550 000442          BR     10$          ;GO RECORD DATA
5551          ;NO TRAP EQUALS YELLOW ZONE BOUNDRY
5552 021552 052710 000012      6$:    BIS    #12,(R0)    ;SET ERROR TYPE IN DATA BUFFER
5553 021556 000437          BR     10$          ;GO RECORD DATA
5554          ;
5555          ;GOT A TRAP. RESTORE AND DETERMINE IF IT IS CORRECT.
5556 021560 016765 157376 177776  1$:    MOV    $TMP0,-2(R5) ;RESTORE WORDS AT
5557 021566 016765 157372 177774  MOV    $TMP1,-4(R5) ;OLD STACK UNDER TEST
5558 021574 032737 000004 177766  BIT    #BIT2,@#CPUERR ;WAS IT A RED ZONE?
5559 021602 001406          BEQ    8$          ;BRANCH IF NO
5560 021604 020106          CMP    R1,R6      ;IS ADDRESS < YELL ZONE BOUNDRY?
5561 021606 101043          BHI    5$          ;BRANCH IF YES
5562 021610 001422          BEQ    10$         ;BRANCH IF ADDRESS = YELL ZONE BOUNDRY
5563          ;RED ZONE TRAP ON LEGAL ADDRESS
5564 021612 052710 000002      BIS    #BIT1,(R0)    ;SET ERROR TYPE IN DATA BUFFER
5565 021616 000417          BR     10$          ;GO RECORD DATA
5566          ;NOT A RED ZONE. IS IT A YELLOW ZONE?
5567 021620 032737 000010 177766  8$:    BIT    #BIT3,@#CPUERR ;IS THIS A YELLOW ZONE TRAP?
5568 021626 001002          BNE    12$         ;BRANCH IF NO
5569 021630 000167 011002          JMP    CPUSPUR     ;GO TO SPURIOUS ROUTINE
5570 021634 020106          12$:  CMP    R1,R6      ;IS ADDRESS = YELL ZONE BOUNDRY?
5571 021636 001427          BEQ    5$          ;BRANCH IF YES
5572 021640 101003          BHI    9$          ;BRANCH IF ADDRESS IS < YELL ZONE BOUNDRY
5573          ;YELLOW ZONE TRAP ON LEGAL ADDRESS
5574 021642 052710 000006      BIS    #6,(R0)      ;SET ERROR TYPE IN DATA BUFFER
5575 021646 000403          BR     10$          ;GO RECORD DATA
5576          ;YELLOW ZONE TRAP ON RED ZONE ADDRESS
5577 021650 052710 000004      9$:    BIS    #BIT2,(R0)    ;SET ERROR TYPE IN DATA BUFFER
5578 021654 000400          BR     10$          ;GO RECORD DATA
5579          ;
5580          ;RECORD ERROR DATA
5581 021656 032737 001000 177570  10$:  BIT    #BIT9,@#SWR   ;IS LOOP ON ERROR ENABLED?
5582 021664 001322          BNE    4$          ;BRANCH IF YES
5583 021666 005767 157274          TST    $TMP2      ;HAS ERROR BUFFER OVERFLOWED?
5584 021672 001011          BNE    5$          ;BRANCH IF YES
5585 021674 005200          INC    R0         ;SET POINTER TO HIGH BYTE
5586 021676 113720 177775          MOV    @#STKLM+1,(R0)+ ;SAVE ERROR STACK LIMIT
5587 021702 010620          MOV    R6,(R0)+   ;SAVE ERROR SP
5588 021704 020027 157774          CMP    R0,#157774 ;IS BUFFER AT PAGE 7?
5589 021710 001002          BNE    5$          ;BRANCH IF NO
5590 021712 005267 157250          INC    $TMP2      ;SET BUFFER OVERFLOW FLAG
5591          ;
5592 021716 062705 000400      5$:    ADD    #400,R5     ;GO TO NEXT STACK ADDRESS
5593 021722 005037 177766          CLR    @#CPUERR   ;CLEAR ERROR REG
5594 021726 000240          NOP
5595 021730 005302          DEC    R2         ;REPLACES A
5596 021732 001271          BNE    3$          ;SOB INSTRUCTION
5597 021734 062737 000400 177774  ADD    #400,@#STKLM ;GO TO NEXT SL ADDRESS
5598 021742 062701 000400          ADD    #400,R1    ;SET NEXT YELLOW ZONE ADDRESS
5599 021746 000240          NOP
5600 021750 062704 000400          ADD    #400,R4    ;SET NEXT YELLOW ZONE ADDR(NO TRAP)
5601 021754 005303          DEC    R3         ;THIS REPLACES
5602 021756 001253          BNE    2$          ;A SOB
5603          ;
5604          ;DONE WITH TEST. WAS THERE AN ERROR?
    
```

```

5605 021760 005037 177774          CLR    @#STKLMT      ;RESET THE SL REG
5606 021764 012706 001100          MOV    #STACK,SP    ;AND SP
5607 021770 012737 032636 000004    MOV    #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5608 021776 005737 120002          TST    @#120002     ;WAS THERE AN ERROR?
5609 022002 001403          BEQ    TST43        ;:BRANCH IF NO
5610 022004 010067 157144          MOV    R0,$REGO     ;SAVE ERROR DATA POINTER
5611 022010 104263          ERROR  263         ;STACK LIMIT COMPARATORS FAILED
5612
5613 .....
5614 *TEST 43          ODD ADDRESS ERROR
5615
5616 *          BEN 13 SHOULD NOT FAIL.
5617 *          IF THE PROCESSOR FAILS TO TRAP IN ALL SECTIONS EITHER TMCC ODD
5618 *          ADRS ERR IS NOT GOING LOW OR TMCC BUS ERROR IS NOT GOING LOW.
5619
5620 *          EACH TYPE OF ODD ADDRESS ERROR IS TESTED INDIVIDUALLY TO
5621 *          ALLOW MAXIMUM ISOLATION.
5622 *          NOTE:AN ODD ADDRESS ON "KERNEL DATI" CANNOT BE TESTED.
5623 *          THIS SIGNAL COMES UP WHEN A TRAP VECTOR IS READ IN FROM THE BUS.
5624 .....
5624 022012 000004          TST43: SCOPE
5625 022014 012767 022446 157246    MOV    #TST44,NEXTTST ;SAVE ADDRESS OF NEXT TEST
5626 022022 005005          CLR    R5           ;INITIALIZE ERROR COUNT
5627
5628 *NON BYTE REFERENCE ON DATIP
5629 022024 012706 001100          MOV    #STACK,SP    ;INITIALIZE THE SP
5630 022030 012737 022110 000004    MOV    #1$,@#ERRVEC ;SETUP ERRVEC
5631 022036 012700 001163          MOV    #STMPO+1,R0  ;PUT ODD ADDRESS IN R0
5632 022042 012767 177777 157112    MOV    #-1,$TMP0    ;PUT -1 IN $TMP0 TO SEE IF TEST CHANGES IT
5633 022050 005210          INC    (R0)         ;EXECUTE ODD ADDRESS INSTRUCTION
5634
5634 022052 012737 022076 000004    ;TRAP DID NOT OCCUR. TRY A DATI.
5635 022060 110030          MOV    #2$,@#ERRVEC ;SETUP ERRVEC
5636 022062 012737 032636 000004    MOV    R0,@(R0)+    ;EXECUTE DATI TO CAUSE ODD ADDR
5637 022070 005205          MOV    #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5638 022072 104265          INC    R5           ;SET ERROR COUNT
5639 022074 000423          ERROR  265         ;NEITHER - BYIN OR DATI CAUSE ODD ADDR
5640 022076 012737 032636 000004    2$:  MOV    #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5641 022104 104266          ERROR  266         ;DATI TRAPPED BUT -BYIN DIDN'T
5642 022106 000416          BR    3$
5643
5644 *NON BYTE REFERENCE WORKED. NOW TRY DATI.
5645 022110 005037 177766          1$:  CLR    @#CPUERR     ;CLEAR ERROR REG
5646 022114 012706 001100          MOV    #STACK,SP    ;INITIALIZE THE SP
5647 022120 012767 022110 156762    MOV    #1$, $LPERR  ;CHANGE LPERR ADDRESS TO THIS SECTION
5648 022126 012737 022144 000004    MOV    #3$,@#ERRVEC ;SETUP ERRVEC
5649 022134 012700 001163          MOV    #STMPO+1,R0  ;PUT ODD ADDR IN R0
5650 022140 110030          MOV    R0,@(R0)+    ;EXECUTE ODD ADDRESS INSTRUCTION
5651 022142 000404          BR    13$
5652 022144 032737 000100 177766    3$:  BIT    #BIT6,@#CPUERR ;DID ODD ADDR BIT SET?
5653 022152 001037          BNE    12$         ;BRANCH IF YES
5654
5654 *TRAP FAILED OR ERROR REG FAILED. TRY A DATO. (REVERSES INPUTS TO TMCC E5(12,13))
5655 022154 005037 177766          13$: CLR    @#CPUERR     ;CLEAR ODD ADDR BIT
5656 022160 012706 001077          MOV    #STACK-1,SP  ;MAKE SP AN ODD ADDRESS
5657 022164 012737 022220 000004    MOV    #4$,@#ERRVEC ;SETUP ERRVEC
5658 022172 012737 022202 000030    MOV    #5$,@#EMTVEC ;SETUP EMT VECTOR
5659 022200 104000          EMT    0           ;EXECUTE DATO TO SP
5660 022202 012706 001100          5$:  MOV    #STACK,SP  ;RESTORE SP
    
```

```

5661 022206 012737 032636 000004      MOV    #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5662 022214 104267                ERROR  267                ;BOTH DATI AND DATO FAILED
5663 022216 000446                BR     6$
5664 022220 012706 001100                4$:  MOV    #STACK,SP      ;RESTORE THE SP
5665 022224 012737 032636 000004      MOV    #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5666 022232 032737 000100 177766      BIT    #BIT6,@#CPUERR   ;DID ODD ADDR BIT SET?
5667 022240 001401                BEQ    14$                ;BRANCH IF NO
5668 022242 104270                ERROR  270                ;DATO WORKS BUT DATI FAILED
5669 022244 104377                14$: ERROR  377
5670 022246 000456                456
5671 022250 000431                BR     6$
5672
5673                ;EITHER DATI WORKED OR -BYIN AND DATI FAILED. NOW TRY DATO.
5674 022252 012767 022144 156630      12$: MOV    #3$,$LPERR    ;CHANGE LPERR ADDRESS TO THIS SECTION
5675 022260 012706 001077                MOV    #STACK-1,SP      ;MAKE SP AN ODD ADDRESS
5676 022264 012737 022334 000004      MOV    #6$,@#ERRVEC    ;SETUP ERRVEC
5677 022272 012737 022302 000030      MOV    #7$,@#EMTVEC    ;SETUP EMTVEC TO CATCH A FAILURE
5678 022300 104000                EMT    0                ;EXECUTE DATO TO ODD ADDRESS
5679                ;TRAP FAILED
5680 022302 012706 001076                7$:  MOV    #1076,SP      ;RESET SP
5681 022306 012737 032636 000004      MOV    #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5682 022314 012737 033520 000030      MOV    # $ERROR,@#EMTVEC ;RESTORE EMTVEC
5683 022322 005705                TST    R5
5684 022324 001002                BNE    11$
5685 022326 104271                ERROR  271                ;NO TRAP ON DATO BUT DATI & -BYIN OK
5686 022330 000401                BR     6$
5687 022332 104274                11$: ERROR  274                ;NO TRAPS
5688
5689                ;EITHER DATO WORKED OR -BYIN AND DATI AND DATO FAILED OR DATO
5690                ;FAILED BUT -BYIN AND DATI OK. TRY SM357 AND SRC1 DATI.
5691 022334 012737 033520 000030      6$:  MOV    # $ERROR,@#EMTVEC ;RESTORE EMT VEC
5692 022342 012767 022350 156540      MOV    #8$,$LPERR      ;CHANGE LOOP ERROR ADDR TO THIS SECT.
5693 022350 012706 001076                8$:  MOV    #1076,SP      ;RESET SP
5694 022354 012737 022422 000004      MOV    #9$,@#ERRVEC    ;SETUP ERRVEC
5695 022362 012700 001163                MOV    # $TMP0+1,R0     ;PUT ODD ADDRESS IN R0
5696 022366 012767 001164 156566      MOV    # $TMP1,$TMP0    ;PUT EVEN ADDRESS IN $TMP0
5697 022374 113001                MOVB   @(R0)+,R1        ;EXECUTE INSTRUCTION TO CAUSE TRAP
5698 022376 012737 032636 000004      MOV    #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5699 022404 012767 022446 156562      MOV    #TST44,$ESCAPE   ;SAVE START ADDRESS OF NEXT TEST
5700 022412 012767 022446 156650      MOV    #TST44,NEXTTST   ;SAVE START ADDRESS OF NEXT TEST
5701 022420 104272                ERROR  272                ;SM357*SRC1 DATI FAILED TO TRAP
5702
5703                ;SM357*SRC1 DATI WORKS CHECK CPU ERROR REG.
5704 022422 012737 032636 000004      9$:  MOV    #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
5705 022430 032737 000100 177766      BIT    #BIT6,@#CPUERR   ;IS ODD ADDRESS BIT SET?
5706 022436 001001                BNE    10$                ;BRANCH IF YES
5707 022440 104273                ERROR  273                ;CPU ERROR REG BIT DOES NOT SET
5708 022442 005037 177766                10$: CLR    @#CPUERR      ;CLEAR ERROR REG.
5709                ;CONTINUE
5710
5711                ;*****
5712                ;*TEST 44      ILLEGAL INSTRUCTIONS
5713                ;*      THIS TEST ENSURES THAT ILLEGAL OP CODES TRAP TO LOCATION 10.
5714                ;*      ONLY THOSE OP CODES THAT HAVE A SINGLE BIT THAT
5715                ;*      DISTINGUISHES THEM FROM A LEGAL INSTRUCTION WILL BE TESTED.
5716                ;*****
5716 022446 000004      TST44: SCOPE
    
```

```
5717 022450 012767 023112 156612      MOV      #KBTST,NEXTTST ;SAVE ADDRESS OF NEXT TEST
5718 022456 012767 022512 156422      MOV      #31$, $LPADR  ;SETUP LOOP ADR
5719 022464 032737 000020 177776      BIT      #BIT4, @#PSW  ;IS T BIT ON?
5720 022472 001404                BEQ      30$           ;BRANCH IF NO
5721 022474 012737 000360 000012      MOV      #360, @#RESVEC+2 ;SETUP RESVEC PSW
5722 022502 000403                BR       31$         ;GO DO TEST
5723                ;SECTION 1-10 THRU 77
5724 022504 012737 000340 000012 30$:      MOV      #PR7, @#RESVEC+2 ;RESTORE RESVEC PSW
5725 022512 012767 022526 156370 31$:      MOV      #1$, $LPERR   ;SETUP ERROR LOOP
5726 022520 012737 022544 000010      MOV      #2$, @#RESVEC  ;SETUP RESVEC
5727 022526 012706 001100                1$:      MOV      #STACK, SP   ;INITIALIZE THE SP
5728 022532 012746 022540                MOV      #3$, -(SP)    ;PUT ADDRESS OF 3$ ON STACK
5729 022536 000010                10      ;EXECUTE OP CODE 10
5730                ;FAILURE-RTT OCCURED
5731 022540 104377                3$:      ERROR   377        ;OP CODE 10 FAILED TO TRAP
5732 022542 000453                453
5733 022544 012767 022560 156336 2$:      MOV      #4$, $LPERR   ;SETUP ERROR LOOP
5734 022552 012737 022566 000010      MOV      #5$, @#RESVEC  ;SETUP RESVEC
5735 022560 000015                4$:      15                ;EXECUTE OP CODE 15
5736                ;FAILURE
5737 022562 104377                ERROR   377        ;OP CODE 15 FAILED TO TRAP
5738 022564 000453                453
5739 022566 012767 022602 156314 5$:      MOV      #6$, $LPERR   ;SETUP ERROR LOOP
5740 022574 012737 022610 000010      MOV      #7$, @#RESVEC  ;SETUP RESVEC
5741 022602 000025                6$:      25                ;EXECUTE OP CODE 25
5742                ;FAILURE
5743 022604 104377                ERROR   377        ;OP CODE 25 FAILED TO TRAP
5744 022606 000453                453
5745 022610 012767 022624 156272 7$:      MOV      #8$, $LPERR   ;SETUP ERROR LOOP
5746 022616 012737 022632 000010      MOV      #9$, @#RESVEC  ;SETUP RESVEC
5747 022624 000045                8$:      45                ;EXECUTE OP CODE 45
5748                ;FAILURE
5749 022626 104377                ERROR   377        ;OP CODE 45 FAILED TO TRAP
5750 022630 000453                453
5751                ;
5752                ;*****
5753                ;SECTION 2-210 THRU 227
5754 022632 012767 022646 156250 9$:      MOV      #10$, $LPERR  ;SETUP ERROR LOOP
5755 022640 012737 022664 000010      MOV      #11$, @#RESVEC ;SETUP RESVEC
5756 022646 012706 001100                10$:     MOV      #STACK, SP   ;INITIALIZE THE SP
5757 022652 012700 022660                MOV      #12$, R0     ;SETUP R0 INCASE RTS
5758 022656 000210                210     ;EXECUTE OP CODE 210
5759                ;FAILURE-RTS EXECUTED
5760 022660 104377                12$:     ERROR   377        ;OP CODE 210 FAILED TO TRAP
5761 022662 000453                453
5762 022664 012767 022700 156216 11$:     MOV      #13$, $LPERR  ;SETUP ERROR LOOP
5763 022672 012737 022716 000010      MOV      #14$, @#RESVEC ;SETUP RESVEC
5764 022700 012706 001100                13$:     MOV      #STACK, SP   ;INITIALIZE THE SP
5765 022704 012700 022712                MOV      #15$, R0     ;SETUP R0 TO CATCH RTS
5766 022710 000220                220     ;EXECUTE OP CODE 220
5767                ;FAILURE-RTS EXECUTED
5768 022712 104377                15$:     ERROR   377        ;OP CODE 220 FAILED TO TRAP
5769 022714 000453                453
5770                ;
5771                ;*****
5772                ;SECTION 3-7000 THRU 7777
```

```
5773 022716 012767 022732 156164 14$: MOV #27$, $LPERR ; SETUP ERROR LOOP
5774 022724 012737 022740 000010 MOV #16$, @#RESVEC ; SETUP RESVEC
5775 022732 007000 27$: 7000 ; EXECUTE OP CODE 7000
5776 ; FAILURE
5777 022734 104377 ERROR 377 ; OP CODE 7000 FAILED TO TRAP
5778 022736 000453 453
5779
5780
5781 ; *****
5782 022740 012767 022754 156142 16$: MOV #17$, $LPERR ; SETUP ERROR LOOP
5783 022746 012737 022762 000010 MOV #18$, @#RESVEC ; SETUP RESVEC
5784 022754 075000 17$: 75000 ; EXECUTE OP CODE 75000
5785 ; FAILURE
5786 022756 104377 ERROR 377 ; OP CODE 75000 FAILED TO TRAP
5787 022760 000453 453
5788 022762 012767 022776 156120 18$: MOV #19$, $LPERR ; SETUP ERROR LOOP
5789 022770 012737 023004 000010 MOV #20$, @#RESVEC ; SETUP RESVEC
5790 022776 076000 19$: 76000 ; EXECUTE OP CODE 76000
5791 ; FAILURE
5792 023000 104377 ERROR 377 ; OP CODE 76000 FAILED TO TRAP
5793 023002 000453 453
5794
5795 ; *****
5796 ; SECTION 5-106400 THRU 106477
5797 023004 012767 023020 156076 20$: MOV #21$, $LPERR ; SETUP ERROR LOOP
5798 023012 012737 023026 000010 MOV #22$, @#RESVEC ; SETUP RESVEC
5799 023020 106400 21$: 106400 ; EXECUTE OP CODE 106400
5800 ; FAILURE
5801 023022 104377 ERROR 377 ; OP CODE 106400 FAILED TO TRAP
5802 023024 000453 453
5803
5804 ; *****
5805 ; SECTION 6-106700 THRU 107777
5806 023026 012767 023042 156054 22$: MOV #23$, $LPERR ; SETUP ERROR LOOP
5807 023034 012737 023054 000010 MOV #24$, @#RESVEC ; SETUP RESVEC
5808 023042 012706 001100 23$: MOV #STACK, SP ; INITIALIZE THE SP
5809 023046 106700 106700 ; EXECUTE OP CODE 106700
5810 ; FAILURE
5811 023050 104377 ERROR 377 ; OP CODE 106700 FAILED TO TRAP
5812 023052 000453 453
5813 023054 012767 023070 156026 24$: MOV #25$, $LPERR ; SETUP ERROR LOOP
5814 023062 012737 023076 000010 MOV #26$, @#RESVEC ; SETUP RESVEC
5815 023070 107000 25$: 107000 ; EXECUTE OP CODE 107000
5816 ; FAILURE
5817 023072 104377 ERROR 377 ; OP CODE 107000 FAILED TO TRAP
5818 023074 000453 453
5819 023076 012737 000012 000010 26$: MOV #12, @#RESVEC ; RESTORE RESVEC
5820 023104 012737 000340 000012 MOV #PR7, @#RESVEC+2 ; RESTORE RESVEC PSW
5821 ; CONTINUE
5822
5823 ; *****
```

```
5824
5825 ; THIS ROUTINE IS USED TO DETERMINE WHICH CPU WE ARE RUNNING ON FOR THE NEXT
5826 ; TEST. THE ROUTINE EXECUTES AN MFPT INSTRUCTION WHICH ONLY EXISTS ON A KB11-E
5827 ; OR KB11-EM. ON A KB11-B/C OR KB11-CM THE MFPT WILL TRAP. AFTER DETERMINING
5828 ; WHICH CPU IS BEING RUN ON A MESSAGE WILL BE PRINTED. THIS TEST COULD NOT
```

```

5829 ; BE RUN BEFORE THE PREVIOUS RESERVED INSTRUCTION TEST.
5830 ;
5831 023112 005227 177777 KBTST: INC #-1 ;FIRST TIME?
5832 023116 001032 BNE KBDONE ;BRANCH IF NOT
5833 023120 005037 001272 CLR @#KB11E ;CLEAR KB11E AND KB11EM FLAGS
5834 023124 012737 023144 000010 MOV #MFPTTR,@#RESVEC ;SET UP TRAP ADDRESS FOR MFPT AT RESERV VECTOR
5835 023132 000007 MFPT ;EXECUTE MFPT. WILL TRAP ON 1170 (KB11B/C) OR
5836 ;KB11-CM
5837 023134 012737 000001 001272 MOV #1,@#KB11E ;HERE IF KB11E OR KB11EM. SET FLAG
5838 023142 000403 BR ENDKB ;DONE DETERMINING WHICH CPU
5839
5840 MFPTTR:
5841 023144 012716 023152 MOV #ENDKB,(SP) ;HERE IF MFPT TRAPPED.
5842 023150 000002 RTI ;SET UP RETURN ADDRESS FOR RTI
5843 023152 012737 000012 000010 ENDKB: MOV #12,@#RESVEC ;RETURN
;RESTORE RESERVE VECTOR
5844
5845 023160 104400 036410 TYPE ,MSG1 ;<15><12>CPU UNDER TEST FOUND TO BE A
5846 023164 005737 001272 TST @#KB11E ;IS THIS A KB11-E OR KB11-EM?
5847 023170 001003 BNE 101$ ;BR IF EITHER ONE
5848 023172 104400 036447 TYPE ,MSG3 ;KB11-B/C OR KB11-CM<15><12>
5849 023176 000402 BR KBDONE ;SKIP OTHER MESSAGE
5850 023200 104400 036514 101$: TYPE ,MSG5 ;KB11-E OR KB11EM<15><12>
5851 023204 KBDONE:
5852 ;*****
5853 ;*****
5854 ;*TEST 45 T BIT TRAP
5855 ;*
5856 ;* IF BEN 13 FAILS EXECUTION WOULD GO TO BRK.20.
5857 ;* THIS WOULD LOOK LIKE THE TRAP DIDN'T OCCUR.
5858 ;*
5859 ;* IF THE TRAP DOESN'T OCCUR THEN EITHER PDRD PS04(1) DOES NOT GET
5860 ;* TO TMCB AS A HIGH OR IT DOES NOT GET TO TMCB E51(10)
5861 ;* AS A LOW OR E51 IS BAD OR IRCD RTT DOES NOT GET TO TMCB AS A HIGH.
5862 ;*
5863 ;* THIS TEST ALSO CHECKS THAT PS<08> SET WILL INHIBIT A T BIT TRAP IF
5864 ;* THIS IS A KB11-E OR KB11-EM.
5865 ;*****
5866 023204 000004 TST45: SCOPE
5867 023206 012767 023454 155760 MOV #TST46,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
5868 023214 012767 023454 156046 MOV #TST46,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
5869 ;*****
5870 ;THIS CODE TURNS THE T BIT OFF IF IT IS ON.
5871 023222 012746 000340 MOV #PR7,-(SP) ;PUT PRIORITY LEVEL 7 ON SP
5872 023226 012746 023242 MOV #1$,-(SP) ;PUT RETURN ADDRESS ON SP
5873 023232 013767 177776 155730 MOV @#PSW,$TMP3 ;SAVE PSW
5874 023240 000006 RTI ;TURN OFF T BIT
5875 ;*****
5876 023242 012767 023250 155640 1$: MOV #2$,$LPERR ;SETUP LOOP ADDRESS
5877 023250 012706 001100 2$: MOV #1100,SP ;INITIALIZE THE SP
5878 023254 012737 023340 000014 MOV #3$,@#TBITVEC ;SET UP T BIT VECTOR
5879 023262 012737 023336 000010 MOV #8$,@#RESVEC ;SETUP RESVEC
5880 023270 012746 000360 MOV #360,-(SP) ;PUT NEW PSW ON STACK (ENABLE T BIT)
5881 023274 012746 023302 MOV #5$,-(SP) ;PUT RETURN LOCATION ON STACK
5882 023300 000002 RTI ;TURN ON T BIT
5883 023302 012746 000340 5$: MOV #PR7,-(SP) ;SETUP STACK TO
5884 023306 012746 023322 MOV #6$,-(SP) ;TURN OFF T BIT
    
```

```
5885 023312 013767 177776 155666      MOV    @#PSW,$ERPSW      ;SAVE PSW
5886 023320 000006      RTT                      ;TURN T BIT OFF
5887 023322 032767 000020 155656 6$:   BIT    #BIT4,$ERPSW     ;DID T BIT SET?
5888 023330 001401      BEQ    7$                ;BRANCH IF NO
5889 023332 104275      ERROR  275              ;T BIT TRAP FAILED
5890 023334 104276      7$:   ERROR  276          ;T BIT NEVER SET
5891 023336 104300      8$:   ERROR  300          ;TRAP VECTOR DECODE FAILED
5892 023340 005767 155726      3$:   TST    KB11E        ;KB11-E OR KB11-EM?
5893 023344 001443      BEQ    11$              ;DONE IF NOT
5894 023346 012746 000340      MOV    #PR7,-(SP)       ;PUT PRIORITY LEVEL 7 ON SP
5895 023352 012746 023366      MOV    #15$,-(SP)      ;PUT RETURN ADDRESS ON SP
5896 023356 013767 177776 155604      MOV    @#PSW,$TMP3     ;SAVE PSW
5897 023364 000006      RTT                      ;TURN OFF T BIT
5898 023366 012767 023374 155514 15$:  MOV    #12$,$LPERR     ;SET UP LOOP ADDRESS
5899 023374 012706 001100 12$:  MOV    #STACK,SP       ;INIT STACK POINTER
5900 023400 012737 023420 000014      MOV    #10$,@#TBITVEC  ;SET UP T BIT VECTOR
5901 023406 012746 000760      MOV    #760,-(SP)      ;NEW PSW ON STACK, ENABLE T BIT AND
5902                                ;SET PS<08> (SUSPEND BIT)
5903 023412 012746 023454      MOV    #11$,-(SP)      ;RETURN LOCATION ON STACK. SHOULD GO TO 11$
5904                                ;INSTEAD OF 10$ ON RTI
5905                                ;TURN ON T BIT AND PS<08>
5906 023420 012746 000340 10$:  MOV    #PR7,-(SP)      ;SETUP STACK TO
5907 023424 012746 023440      MOV    #13$,-(SP)      ;TURN OFF T BIT
5908 023430 013767 177776 155550      MOV    @#PSW,$ERPSW     ;SAVE PSW
5909 023436 000006      RTT                      ;TURN T BIT OFF
5910 023440 032767 000400 155540 13$:  BIT    #BIT8,$ERPSW     ;DID PS<08> SET?
5911 023446 001401      BEQ    14$              ;BR IF NOT
5912 023450 104277      ERROR  277              ;SETTING PS<08> FAILED TO DISABLE T-TRAP
5913 023452 104376      14$:  ERROR  376          ;PS<08> FAILED TO SET
5914 023454      11$:  ;CONTINUE
5915                                ;*****
5916                                ;*TEST 46      T BIT TRAP AND RTI
5917                                ;*
5918                                ;*   IF THE INSTRUCTION AFTER THE RTT DOES NOT GET EXECUTED THEN
5919                                ;*   EITHER IRCD RTT DOES NOT GO LOW OR IT DOES NOT GET TO
5920                                ;*   TMCB E74(11) OR TMCB E74 IS BAD.
5921                                ;*****
5922 023454 000004      TST46: SCOPE
5923 023456 012767 023526 155604      MOV    #TST47,NEXTTST  ;SAVE ADDRESS OF NEXT TEST
5924 023464 005000      CLR    R0              ;ENSURE R0 CLEAR
5925 023466 012706 001100      MOV    #STACK,SP       ;INITIALIZE THE SP
5926 023472 012746 000360      MOV    #360,-(SP)      ;PUT NEW PSW ON STACK (ENABLE T BIT)
5927 023476 012746 023512      MOV    #1$,-(SP)       ;PUT PC ON STACK
5928 023502 012737 023516 000014      MOV    #2$,@#TBITVEC  ;SETUP T BIT VECTOR
5929 023510 000006      RTT                      ;TURN T BIT ON WITH RTT
5930 023512 012700 000001 1$:   MOV    #1,R0            ;THIS SHOULD EXECUTE
5931 023516 022700 000001 2$:   CMP    #1,R0            ;DID MOV INSTRUCTION EXECUTE?
5932 023522 001401      BEQ    TST47            ;BRANCH IF YES
5933 023524 104301      ERROR  301            ;RTT DID NOT DISABLE T BIT
5934                                ;*****
5935                                ;*TEST 47      PRIORITY ARBITRATION
5936                                ;*
5937                                ;*   THIS TEST ASSURES THAT EACH NECESSARY INPUT TO AN HONOR FLAG
5938                                ;*   CAN DISABLE THAT FLAG.
5939                                ;*
5940                                ;*   EACH SECTION WILL PERFORM A SETUP SO THAT A TIGHT ERROR LOOP
```


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

```

*****
5967 023526 000004 TST47: SCOPE
5968 023530 012767 026350 155532 MOV #TST50,NEXTTST ;SAVE ADDRESS OF NEXT TEST
5969 023536 012767 003706 155340 CHGE1: MOV #^D1990,$ICNT ;SETUP ITERATION COUNT
5970 023544 012767 023560 155334 CHGE2: MOV #103$,$LPADR ;SETUP LOOP ADDRESS
5971 023552 012737 032622 000014 MOV #SRTN,@#TBITVEC ;RESTORE T BIT VEC
5972 023560 005005 103$: CLR R5 ;CLEAR BR5 DISABLE FLAG
5973 023562 005004 CLR R4 ;CLEAR BR6 DISABLE FLAG
5974 023564 005003 CLR R3 ;CLEAR BR4 DISABLE FLAG
5975 023566 032737 000020 001170 BIT #BIT4,@#STMP3 ;WAS T BIT ON?
5976 023574 001417 BEQ 71$ ;BRANCH IF NO
5977 023576 012737 000360 000066 MOV #360,@#TPVEC+2
5978 023604 012737 000360 000242 MOV #360,@#PIRQVEC+2
5979 023612 012737 000360 000006 MOV #360,@#ERRVEC+2
5980 023620 012746 000360 MOV #360,-(SP) ;SET UP STACK PSW
5981 023624 012746 023656 MOV #72$,-(SP) ;PUT RETURN ADDRESS ON STACK
5982 023630 000006 RTT ;TURN T BIT ON
5983 023632 000411 BR 72$
5984 023634 012737 000340 000066 71$: MOV #PR7,@#TPVEC+2 ;PUT PRIORITY 7 IN PRINTER VECTOR
5985 023642 012737 000340 000006 MOV #PR7,@#ERRVEC+2 ;RESTORE ERRVEC PSW
5986 023650 012737 000340 000242 MOV #PR7,@#PIRQVEC+2 ;PUT PRIORITY 7 IN PIRQ VECTOR
5987 023656 032737 000400 177570 72$: BIT #SW8,@#SWR ;SWITCH 8 ON?
5988 023664 001006 BNE 100$ ;BRANCH IF YES
5989 023666 032737 000040 177570 BIT #SW5,@#SWR ;IS BR 5 TESTING DISABLED?
5990 023674 001402 BEQ 100$ ;BRANCH IF NO
5991 023676 005205 102$: INC R5 ;SET BR 5 DISABLE FLAG
5992 023700 000403 BR 101$ ;CONTINUE
5993 023702 005767 155316 100$: TST INTER5 ;IS THERE A BR 5 DEVICE?
5994 023706 001773 BEQ 102$ ;BRANCH IF NO
*****

```

5995
 5996 :SECTION 1 - BR4 AND PIR1

```

5997 023710 032737 000400 177570 101$: BIT #SW8,@#SWR ;SWITCH 8 ON?
5998 023716 001006 BNE 68$ ;BRANCH IF YES
5999 023720 032737 000020 177570 BIT #SW4,@#SWR ;IS BR4 TESTING DISABLED?
6000 023726 001402 BEQ 68$ ;BRANCH IF NO
6001 023730 005203 INC R3 ;SET BR4 FLAG
6002 023732 000432 BR 2$ ;GO TO NEXT SECTION
6003 023734 012767 023762 155146 68$: MOV #1$,$LPERR ;SETUP ERROR LOOP
6004 023742 012737 024020 000064 MOV #2$,@#TPVEC ;SETUP PRINTER VECTOR
6005 023750 012737 024006 000240 MOV #3$,@#PIRQVEC ;SETUP PIRQ VECTOR
6006 ;***** CHGE1 *****
6007 023756 005077 155154 CLR @#TKS ;CON'T ALLOW KEYBD INTERRUPTS
6008 ;*****
6009 023762 012706 001076 1$: MOV #1076,SP ;INITIALIZE THE SP
6010 023766 000237 SPL 7 ;ENSURE CPU AT LEVEL 7
6011 023770 052737 001000 177772 BIS #BIT9,@#PIRQ ;SET PIR LEVEL 1
6012 023776 004767 002212 JSR PC,LEVEL4 ;GO GET BR4
6013 024002 000230 SPL 0 ;SET CPU AT ZERO
6014 024004 000240 NOP ;USED WITH SPL
6015 ;FAILURE PIR CAME THRU
6016 024006 005037 177772 3$: CLR @#PIRQ ;CLEAR THE PIRQ
6017 024012 005077 155124 CLR @#TPS ;CLEAR THE PRINTER FLAG
6018 024016 104302 ERROR 302 ;PIR 1 DID NOT DISABLE ON BR
6019 ;*****
6020 ;SECTION 2 - PIR 1 AND SL YELLOW
6021 024020 005077 155116 2$: CLR @#TPS ;CLEAR PRINTER INT FLAG
6022 024024 012767 024052 155056 MOV #4$,$LPERR ;SETUP ERROR LOOP
6023 024032 012737 024110 000004 MOV #5$,@#ERRVEC ;SETUP LOCATION 4
6024 024040 012737 024070 000240 MOV #6$,@#PIRQVEC ;SETUP PIRQ VECTOR
6025 024046 005037 177772 CLR @#PIRQ ;CLEAR LEVEL 1
6026 024052 012706 000376 4$: MOV #376,SP ;SETUP THE SP
6027 024056 052737 001000 177772 BIS #BIT9,@#PIRQ ;SET LEVEL 1
6028 024064 000230 SPL 0 ;SET CPU AT ZERO
6029 024066 011616 MOV (SP),(SP) ;EXECUTE YELL ZONE INSTR
6030 ;FAILURE, PIR CAME THRU
6031 024070 012706 001100 6$: MOV #STACK,SP ;RESET SP
6032 024074 012737 032636 000004 MOV #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
6033 024102 005037 177772 CLR @#PIRQ ;CLEAR LEVEL 1
6034 024106 104303 ERROR 303 ;PIR CAME THRU ON YELLOW ZONE
6035 ;*****
6036 ;SECTION 3 - PIR 2 AND SL YELLOW
6037 ;*****
6038 024110 005037 177772 5$: CLR @#PIRQ ;CLEAR LEVEL 1
6039 024114 012767 024136 154766 MOV #7$,$LPERR ;SETUP ERROR LOOP
6040 024122 012737 024174 000004 MOV #8$,@#ERRVEC ;SETUP LOCATION 4
6041 024130 012737 024154 000240 MOV #9$,@#PIRQVEC ;SETUP PIRQ VECTOR
6042 024136 012706 000376 7$: MOV #376,SP ;SETUP THE SP
6043 024142 052737 002000 177772 BIS #BIT10,@#PIRQ ;SET LEVEL 2
6044 024150 000231 SPL 1 ;SET CPU AT LEVEL 1
6045 024152 011616 MOV (SP),(SP) ;EXECUTE TRAP CAUSING INSTR
6046 ;FAILURE, PIR CAME THRU
6047 024154 012706 001100 9$: MOV #STACK,SP ;RESET SP
6048 024160 012737 032636 000004 MOV #CPUSPUR,@#ERRVEC ;RESET ERRVEC
6049 024166 005037 177772 CLR @#PIRQ ;CLEAR LEVEL 2
6050 024172 104304 ERROR 304 ;PIR 2 CAME THRU ON YELLOW ZONE
6051 ;*****
6052 ;*****
    
```

```

6053      :SECTION 4 - PIR 3 AND YEL ZONE
6054 024174 005037 177772      8$: CLR @#PIRQ ;CLEAR LEVEL 2
6055 024200 012767 024222 154702 MOV #10$, $LPERR ;SETUP ERROR LOOP
6056 024206 012737 024260 000004 MOV #11$, @#ERRVEC ;SETUP ERR VECTOR
6057 024214 012737 024240 000240 MOV #12$, @#PIRQVEC ;SETUP PIRQ VECTOR
6058 024222 012706 000376      10$: MOV #376, SP ;SETUP THE SP
6059 024226 052737 004000 177772 BIS #BIT11, @#PIRQ ;SET LEVEL 3
6060 024234 000232      SPL 2 ;SET CPU AT LEVEL 2
6061 024236 011616      MOV (SP), (SP) ;EXECUTE TRAP CAUSING INSTR
6062      :FAILURE, PIR CAME THRU
6063 024240 012706 001100      12$: MOV #STACK, SP ;RESET SP
6064 024244 012737 032636 000004 MOV #CPUSPUR, @#ERRVEC ;RESET LOCATION 4
6065 024252 005037 177772      CLR @#PIRQ ;CLEAR LEVEL 3
6066 024256 104305      ERROR 305 ;PIR 3 CAME THRU ON YELLOW ZONE
6067      :*****
6068      :SECTION 5- PIR4 AND BR4
6069 024260 012706 001100      11$: MOV #STACK, SP ;RESTORE THE SP
6070 024264 005703      TST R3 ;IS BR4 TESTING DISABLED?
6071 024266 001402      BEQ 70$ ;BRANCH IF NO
6072 024270 000167 000410      JMP 29$ ;SKIP BR4 SECTIONS
6073 024274 012767 024316 154606 70$: MOV #13$, $LPERR ;SETUP ERROR LOOP
6074 024302 012737 024352 000240 MOV #15$, @#PIRQVEC ;SETUP PIRQ VEC
6075 024310 012737 024340 000064 MOV #14$, @#TPVEC ;SETUP TPVEC
6076 024316 012706 001100      13$: MOV #STACK, SP ;INITIALIZE THE SP
6077 024322 012737 010000 177772 MOV #BIT12, @#PIRQ ;SET PIR LEVEL 4
6078 024330 004767 001660      JSR PC, LEVEL4 ;GO GET BR 4
6079 024334 000233      SPL 3 ;LOWER CPU
6080 024336 000240      NOP ;REQUIRED BECAUSE OF SPL
6081      :FAILURE, BR4 CAME THRU
6082 024340 005077 154576      14$: CLR @STPS ;CLEAR PRINTER INT FLAG
6083 024344 005037 177772      CLR @#PIRQ ;CLEAR LEVEL 4
6084 024350 104306      ERROR 306 ;BR4 CAME THRU ON PIR4
6085      :*****
6086      :SECTION 6- PIR 5 AND BR4
6087      15$: MOV #16$, $LPERR ;SETUP ERROR LOOP
6088 024352 012767 024374 154530 MOV #17$, @#TPVEC ;SETUP BR4 VEC
6089 024360 012737 024416 000064 MOV #18$, @#PIRQVEC ;SETUP PIRQ VEC
6090 024366 012737 024430 000240      16$: MOV #STACK, SP ;INITIALIZE THE SP
6091 024374 012706 001100      MOV #BIT13, @#PIRQ ;SET PIR LEVEL 5
6092 024400 012737 020000 177772 JSR PC, LEVEL4 ;GO GET BR4
6093 024406 004767 001602      SPL 3 ;SET CPU AT 3
6094 024412 000233      NOP ;REQUIRED BECAUSE OF SPL
6095 024414 000240      :FAILURE, BR4 CAME THRU
6096      17$: CLR @STPS ;CLEAR BR4 INT FLAG
6097 024416 005077 154520      CLR @#PIRQ ;CLEAR PIR5
6098 024422 005037 177772      ERROR 307 ;BR4 CAME THRU ON PIR5
6099 024426 104307
6100      :*****
6101      :SECTION 7- BR5 AND BR4
6102      18$: CLR @#PIRQ ;CLEAR PIR LEVEL 5
6103      CLR @STPS ;CLEAR PRINTER INT FLAG
6104 024430 005037 177772      TST R5 ;IS BR 5 TESTING DISABLED?
6105 024434 005077 154502      BNE 23$ ;BRANCH IF YES
6106 024440 005705      20$: MOV #21$, $LPERR ;SETUP ERROR LOOP
6107 024442 001040
6108 024444 012767 024512 154436
    
```

```

6109 024452 012737 024532 000064      MOV      #22$,@#TPVEC      ;SETUP BR4 VECTOR
6110 024460 016700 154542      MOV      INT5VEC,RO      ;GET BR5 VECTOR
6111 024464 012720 024544      MOV      #23$, (R0)+     ;PUT ADDRESS OF 23$ IN VECTOR
6112 024470 032737 000020 177776    BIT      #BIT4,@#PSW     ;IS T BIT ON?
6113 024476 001403      BEQ      73$             ;BRANCH IF NO
6114 024500 012710 000360      MOV      #360, (R0)     ;SETUP VECTOR PSW
6115 024504 000402      BR       21$
6116 024506 012710 000340      73$:  MOV      #PR7, (R0)   ;PUT PR7 IN VECTOR+2
6117 024512 012706 001100      21$:  MOV      #STACK,SP   ;INITIALIZE THE SP
6118 024516 004767 001502      JSR     PC,LEVEL5      ;GO GET BR5
6119 024522 004767 001466      JSR     PC,LEVEL4      ;GO GET BR4
6120 024526 000233      SPL     3              ;SET CPU AT LEVEL 3
6121 024530 000240      NOP                    ;REQUIRED BECAUSE OF SPL
6122      ;FAILURE, BR4 CAME IN
6123 024532 004767 001534      22$:  JSR     PC,KILBR5    ;GET RID OF BR 5
6124 024536 005077 154400      CLR     @#STPS        ;CLEAR BR4 INT ENABLE
6125 024542 104310      ERROR   310          ;BR4 CAME THRU ON BR5
6126
6127      ;
6128      ;*****
6129 024544 004767 001522      ;SECTION 8- PIR6 AND BR4
6130 024550 012767 024572 154332    23$:  JSR     PC,KILBR5    ;GET RID OF BR 5
6131 024556 012737 024614 000064      MOV     #24$, $LPERR    ;SETUP ERROR LOOP
6132 024564 012737 024626 000240      MOV     #25$,@#TPVEC    ;SETUP BR4 INTERRUPT VECTOR.
6133 024572 012706 001076      MOV     #26$,@#PIRQVEC  ;SETUP PIRQ VECTOR
6134 024576 012737 040000 177772    24$:  MOV     #1076,SP       ;INITIALIZE THE SP
6135 024604 004767 001404      MOV     #BIT14,@#PIRQ   ;SET PIR LEVEL 6
6136 024610 000233      JSR     PC,LEVEL4      ;GO GET BR4
6137 024612 000240      SPL     3              ;SET CPU AT LEVEL 3
6138      NOP                    ;REQUIRED BECAUSE OF SPL
6139 024614 005037 177772      ;FAILURE, BR4 CAME IN
6140 024620 005077 154316    25$:  CLR     @#PIRQ        ;CLEAR PIR LEVEL 6
6141 024624 104311      CLR     @#STPS        ;CLEAR PRINTER INTER FLAG
6142      ERROR   311          ;BR4 CAME IN ON PIR6
6143      ;
6144      ;*****
6145 024626 012767 024650 154254    ;SECTION 9- PIR 7 AND BR4
6146 024634 012737 024672 000064    26$:  MOV     #27$, $LPERR    ;SETUP ERROR LOOP
6147 024642 012737 024704 000240      MOV     #28$,@#TPVEC    ;SETUP BR4 VECTOR
6148 024650 012706 001076      MOV     #29$,@#PIRQVEC  ;SETUP PIRQ VECTOR
6149 024654 012737 100000 177772    27$:  MOV     #1076,SP       ;SETUP THE SP
6150 024662 004767 001326      MOV     #BIT15,@#PIRQ   ;SET PIR LEVEL 7
6151 024666 000233      JSR     PC,LEVEL4      ;GO GET BR4
6152 024670 000240      SPL     3              ;LOWER CPU TO 3
6153      NOP                    ;REQUIRED BECAUSE OF SPL
6154 024672 005077 154244    ;FAILURE, BR4 CAME IN
6155 024676 005037 177772    28$:  CLR     @#STPS        ;CLEAR PRINTER INTERR FLAG
6156 024702 104312      CLR     @#PIRQ        ;CLEAR PIR LEVEL 7
6157      ERROR   312          ;BR4 CAME IN ON PIR7
6158      ;
6159      ;*****
6160 024704 005077 154232    ;SECTION 10- BR5 AND PIR4
6161 024710 005037 177772    29$:  CLR     @#STPS        ;CLEAR BR4 INTERR FLAG
6162 024714 005705      CLR     @#PIRQ        ;CLEAR PIRQ LEVEL 7
6163 024716 001041      TST     R5             ;IS THERE A BR5 DEVICE?
6164 024720 012767 024766 154162    BNE     32$           ;BRANCH TO SECTION 11 IF NO
      MOV     #30$, $LPERR ;SETUP ERROR LOOP
    
```

```

6165 024726 012737 025010 000240      MOV      #31$,@#PIRQVEC ;SETUP PIRQ VECTOR
6166 024734 016700 154266      MOV      INT5VEC,R0    ;GET ADDR OF BR 5 VECTOR
6167 024740 012720 025022      MOV      #32$, (R0)+  ;SETUP BR5 VECTOR
6168 024744 032737 000020 177776      BIT      #BIT4,@#PSW  ;IS T BIT ON?
6169 024752 001403      BEQ      74$          ;BRANCH IF NO
6170 024754 012710 000360      MOV      #360,(R0)
6171 024760 000462      BR       75$
6172 024762 012710 000340      74$: MOV      #PR7,(R0)
6173 024766 012706 001100      30$: MOV      #STACK,SP ;SETUP THE SP
6174 024772 012737 010000 177772      MOV      #BIT12,@#PIRQ ;SET PIR4
6175 025000 004767 001220      JSR      PC,LEVEL5    ;GO GET BR5
6176 025004 000233      SPL      3            ;LOWER CPU TO LEVEL 3
6177 025006 000240      NOP                    ;REQUIRED BECAUSE OF SPL
6178      ;FAILURE, PIR4 CAME IN
6179 025010 004767 001256      31$: JSR      PC,KILBR5 ;GET RID OF BR 5
6180 025014 005037 177772      CLR      @#PIRQ      ;CLEAR PIR LEVEL 4
6181 025020 104313      ERROR    313        ;PIR4 CAME IN ON BR5
6182
6183      ;
6184      ;*****
6185      ;SECTION 11- BR6 AND PIR4
6185 025022 005037 177772      32$: CLR      @#PIRQ      ;CLEAR PIR LEVEL 4
6186 025026 004767 001240      JSR      PC,KILBR5    ;GET RID OF BR 5
6187 025032 032737 000400 177570      BIT      #SW8,@#SWR  ;SWITCH 8 ON?
6188 025040 001004      BNE      80$         ;BRANCH IF YES
6189 025042 032737 000100 177570      BIT      #SW6,@#SWR  ;IS BR6 TESTING DISABLED?
6190 025050 001003      BNE      33$         ;BRANCH IF YES
6191 025052 005767 154154      80$: TST      INTER6   ;IS THERE A BR6 DEVICE?
6192 025056 001002      BNE      34$         ;BRANCH IF YES
6193 025060 005204      33$: INC      R4        ;SET BR 6 DISABLE FLAG
6194 025062 000441      BR       37$         ;GO TO SECTION 12
6195 025064 012767 025132 154016 34$: MOV      #35$, $LPERR ;SETUP ERROR LOOP
6196 025072 012737 025154 000240      MOV      #36$,@#PIRQVEC ;SETUP PIR VECTOR
6197 025100 016701 154130      MOV      INT6VEC,R1  ;GET BR 6 VECTOR
6198 025104 012721 025166      MOV      #37$, (R1)+ ;PU ADDRESS OF 37$ IN VECTOR
6199 025110 032737 000020 177776      BIT      #BIT4,@#PSW  ;IS T BIT ON?
6200 025116 001403      BEQ      75$         ;BRANCH IF NO
6201 025120 012711 000360      MOV      #360,(R1)
6202 025124 000402      BR       35$
6203 025126 012711 000340      75$: MOV      #PR7,(R1) ;PUT PRIORITY OF 7 IN VECTOR+2
6204 025132 012706 001100      35$: MOV      #STACK,SP ;INITIALIZE THE SP
6205 025136 012737 010000 177772      MOV      #BIT12,@#PIRQ ;SET PIR LEVEL 4
6206 025144 004767 001064      JSR      PC,LEVEL6    ;GO GET BR6
6207 025150 000233      SPL      3            ;LOWER CPU TO LEVEL 3
6208 025152 000240      NOP                    ;REQUIRED BECAUSE OF SPL
6209      ;FAILURE, PIR4 CAME IN
6210 025154 005077 154056      36$: CLR      @INT6ST   ;CLEAR BR6 INTERR FLAG
6211 025160 005037 177772      CLR      @#PIRQ      ;CLEAR PIR LEVEL 4
6212 025164 104314      ERROR    314        ;PIR4 CAME IN ON BR6
6213
6214      ;
6215      ;*****
6216      ;SECTION 12- PIR4 AND STACK LIMIT YELLOW
6216 025166 005077 154044      37$: CLR      @INT6ST   ;CLEAR BR6 INTERR. FLAG
6217 025172 005037 177772      CLR      @#PIRQ      ;CLEAR LEVEL 4
6218 025176 012767 025220 153704      MOV      #38$, $LPERR ;SETUP ERROR LOOP
6219 025204 012737 025236 000240      MOV      #39$,@#PIRQVEC ;SETUP PIRQ VECTOR
6220 025212 012737 025256 000004      MOV      #40$,@#ERRVEC ;SETUP YELL ZONE VECTOR
    
```

```

6221 025220 012706 000376      38$:  MOV    #376,SP          ;SETUP THE SP
6222 025224 052737 010000 177772  BIS    #BIT12,@#PIRQ    ;SET LEVEL 4
6223 025232 000233          SPL    3                ;SET CPU AT LEVEL 3
6224 025234 011616          MOV    (SP),(SP)        ;EXECUTE TRAP CAUSING INSTR
6225          ;FAILURE, PIR 4 CAME THRU
6226 025236 012706 001100      39$:  MOV    #STACK,SP      ;RESET THE SP
6227 025242 012737 032636 000004  MOV    #CPUSPUR,@#ERRVEC ;RESTORE ERR VEC
6228 025250 005037 177772          CLR    @#PIRQ          ;CLEAR PIR LEVEL 4
6229 025254 104315          ERROR  315            ;PIR4 CAME IN ON SL YELLOW
6230
6231          ;*****
6232          ;SECTION 13- BR5 AND PIR5
6233 025256 012706 001100      40$:  MOV    #STACK,SP      ;RESTORE THE SP
6234 025262 012737 032636 000004  MOV    #CPUSPUR,@#ERRVEC ;RESTORE LOCATION 4
6235 025270 005037 177772          CLR    @#PIRQ          ;CLEAR PIR 4
6236 025274 005705          TST    R5              ;IS BR5 DISABLED?
6237 025276 001105          BNE    49$             ;BRANCH IF YES TO SECTION 16
6238 025300 012767 025322 153602  MOV    #41$,$LPERR     ;SETUP ERROR LOOP
6239 025306 012777 025344 153712  MOV    #42$,@#INT5VEC  ;SETUP BR5 VECTOR
6240 025314 012737 025356 000240  MOV    #43$,@#PIRQVEC  ;SETUP PIRQ VECTOR
6241
6242 025322 012706 001076      41$:  MOV    #1076,SP        ;INITIALIZE THE SP
6243 025326 012737 020000 177772  MOV    #BIT13,@#PIRQ    ;SET PIR LEVEL 5
6244 025334 004767 000664          JSR    PC,LEVEL5       ;GO GET BR5
6245
6246          SPL    4                ;LOWER CPU TO LEVEL 4
6247 025342 000240          NOP                    ;REQUIRED BECAUSE OF SPL
6248          ;FAILURE, BR5 CAME IN
6249 025344 004767 000722      42$:  JSR    PC,KILBR5       ;GET RID OF BR 5
6250 025350 005037 177772          CLR    @#PIRQ          ;CLEAR PIR5
6251 025354 104316          ERROR  316            ;BR5 CAME IN ON PIR5
6252
6253          ;*****
6254          ;SECTION 14- BR5 AND PIR6
6255 025356 012767 025400 153524  43$:  MOV    #44$,$LPERR     ;SETUP ERROR LOOP
6256 025364 012777 025422 153634  MOV    #45$,@#INT5VEC  ;SETUP BR5 VEC
6257 025372 012737 025434 000240  MOV    #46$,@#PIRQVEC  ;SETUP PIRQ VECTOR
6258 025400 012706 001076      44$:  MOV    #1076,SP        ;INITIALIZE THE SP
6259 025404 012737 040000 177772  MOV    #BIT14,@#PIRQ    ;SET PIR LEVEL 6
6260 025412 004767 000606          JSR    PC,LEVEL5       ;GO GET BR5
6261 025416 000234          SPL    4                ;SET CPU AT LEVEL 4
6262 025420 000240          NOP                    ;REQUIRED BECAUSE OF SPL
6263          ;FAILURE, BR5 CAME IN
6264 025422 004767 000644      45$:  JSR    PC,KILBR5       ;GET RID OF BR5
6265 025426 005037 177772          CLR    @#PIRQ          ;CLEAR PIR LEVEL 6
6266 025432 104317          ERROR  317            ;BR5 CAME IN ON PIR6
6267
6268          ;*****
6269          ;SECTION 15- BR5 AND PIR7
6270 025434 012767 025456 153446  46$:  MOV    #47$,$LPERR     ;SETUP ERROR LOOP
6271 025442 012777 025500 153556  MOV    #48$,@#INT5VEC  ;SETUP BR5 VECTOR
6272 025450 012737 025512 000240  MOV    #49$,@#PIRQVEC  ;SETUP PIRQ VECTOR
6273 025456 012706 001076      47$:  MOV    #1076,SP        ;INITIALIZE THE SP
6274 025462 012737 100000 177772  MOV    #BIT15,@#PIRQ    ;SET PIR LEVEL 7
6275 025470 004767 000530          JSR    PC,LEVEL5       ;GO GET BR5
6276 025474 000234          SPL    4                ;ALLOW BRQ
    
```

```

6277 025476 000240                NOP                ;REQUIRED BECAUSE OF SPL
6278                                ;FAILURE, BR5 CAME IN
6279 025500 004767 000566        48$: JSR          PC,KILBR5    ;GET RID OF BR5
6280 025504 005037 177772        CLR          @#PIRQ          ;CLEAR PIR LEVEL 7
6281 025510 104320                ERROR          320          ;BR5 CAME IN ON PIR7
6282
6283                                ;*****
6284                                ;SECTION 16- PIR5 AND BR6
6285 025512 004767 000554        49$: JSR          PC,KILBR5    ;GET RID OF BR 5
6286 025516 005704                TST          R4              ;IS BR6 TESTING DISABLED?
6287 025520 001030                BNE          52$             ;BRANCH IF YES TO SECTION 17
6288 025522 012767 025544 153360  MOV          #50$, $LPERR     ;SETUP ERROR LOOP
6289 025530 012737 025570 000240  MOV          #51$, @#PIRQVEC  ;SETUP PIR VECTOR
6290 025536 012777 025602 153470  MOV          #52$, @INT6VEC   ;SETUP BR6 VECTOR
6291 025544 012706 001076        50$: MOV          #1076, SP     ;INITIALIZE THE SP
6292 025550 012737 020000 177772  MOV          #BIT13, @#PIRQ   ;SET IR LEVEL 5
6293 025556 004767 000452        JSR          PC,LEVEL6       ;GO GET BR6
6294 025562 000234                SPL          4               ;ALLOW BRQ
6295 025564 000240                NOP                ;REQUIRED BECAUSE OF SPL
6296                                ;FAILURE, PIR 5 CAME IN
6297 025566 000000                HALT                ;DEBUG ONLY
6298 025570 005077 153442        51$: CLR          @INT6ST      ;CLEAR BR6 INTERR FLAG
6299 025574 005037 177772        CLR          @#PIRQ          ;CLEAR PIR LEVEL 5
6300 025600 104321                ERROR          321          ;PIR 5 CAME IN ON BR6
6301
6302                                ;*****
6303                                ;SECTION 17- PIR 5 AND STACK LIMIT YELLOW
6304 025602 005077 153430        52$: CLR          @INT6ST      ;CLEAR BR6 INTERR FLAG
6305 025606 012767 025630 153274  MOV          #53$, $LPERR     ;SETUP ERROR LOOP
6306 025614 012737 025646 000240  MOV          #54$, @#PIRQVEC  ;SETUP PIRQVEC
6307 025622 012737 025666 000004  MOV          #55$, @#ERRVEC   ;SETUP LOCATION 4
6308 025630 012706 000376        53$: MOV          #376, SP     ;SETUP THE SP
6309 025634 052737 020000 177772  BIS          #BIT13, @#PIRQ   ;SET LEVEL 5
6310 025642 000234                SPL          4               ;SET CPU AT 4
6311 025644 011616                MOV          (SP), (SP)      ;EXECUTE YEL ZONE INSTR
6312                                ;FAILURE, PIR5 CAME IN ON SL YELLOW
6313 025646 005037 177772        54$: CLR          @#PIRQ          ;CLEAR LEVEL 5
6314 025652 012737 032636 000004  MOV          #CPUSPUR, @#ERRVEC ;RESTORE ERRVEC
6315 025660 012706 001100        MOV          #STACK, SP     ;RESET THE SP
6316 025664 104322                ERROR          322          ;PIR5 CAME IN ON SL YELLOW
6317
6318                                ;*****
6319                                ;SECTION 18- BR6 AND PIR6
6320 025666 012706 001100        55$: MOV          #STACK, SP   ;RESTORE THE SP
6321 025672 005704                TST          R4              ;IS BR6 TESTING DISABLED?
6322 025674 001056                BNE          61$             ;BRANCH IF YES TO SECTION 20
6323 025676 012767 025720 153204  MOV          #56$, $LPERR     ;SETUP ERROR LOOP
6324 025704 012777 025742 153322  MOV          #57$, @INT6VEC   ;SETUP BR6 INTERR VECTOR
6325 025712 012737 025754 000240  MOV          #58$, @#PIRQVEC  ;SETUP PIR VECTOR
6326 025720 012706 001076        56$: MOV          #1076, SP     ;SETUP THE SP
6327 025724 012737 040000 177772  MOV          #BIT14, @#PIRQ   ;SET PIR LEVEL 6
6328 025732 004767 000276        JSR          PC,LEVEL6       ;GO GET BR6
6329 025736 000235                SPL          5               ;LOWER CPU TO 5
6330 025740 000240                NOP                ;REQUIRED BECAUSE OF SPL
6331                                ;FAILURE, BR6 CAME IN ON PIR6
6332 025742 005077 153270        57$: CLR          @INT6ST      ;CLEAR BR6 INTERR FLAG
    
```

```

6333 025746 005037 177772          CLR    @#PIRQ          ;CLEAR LEVEL 6
6334 025752 104323          ERROR  323            ;BR6 CAME IN ON PIR6
6335
6336
6337
6338 025754 012767 025776 153126  ;SECTION 19- BR6 AND PIR7
6339 025762 012777 026020 153244 58$:  MOV    #59$, $LPERR    ;SETUP ERROR LOOP
6340 025770 012737 026032 000240    MOV    #60$, @INT6VEC    ;SETUP BR6 VECTOR
6341 025776 012706 001076    MOV    #61$, @#PIRQVEC   ;SETUP PIRQ VECTOR
6342 026002 012737 100000 177772 59$:  MOV    #1076, SP        ;SETUP THE SP
6343 026010 004767 000220    MOV    #BIT15, @#PIRQ    ;SET PIR 7
6344 026014 000235    JSR    PC, LEVEL6        ;GO GET BR6
6345 026016 000240    SPL                    ;LOWER CPU TO 5
6346                                NOP                        ;REQUIRED BECAUSE OF SPL
6347 026020 005077 153212  ;FAILURE, BR6 CAME IN ON PIR7
6348 026024 005037 177772 60$:  CLR    @INT6ST          ;CLEAR BR6 INTERR FLAG
6349 026030 104324          CLR    @#PIRQ          ;CLEAR LEVEL 7
6350                                ERROR  324            ;BR6 CAME IN ON PIR7
6351
6352
6353 026032 012767 026054 153050  ;SECTION 20- PIR6 AND SL YELLOW
6354 026040 012737 026072 000240 61$:  MOV    #62$, $LPERR    ;SETUP LOOP ADDRESS
6355 026046 012737 026104 000004    MOV    #63$, @#PIRQVEC   ;SETUP PIRQ VECTOR
6356 026054 012706 000376    MOV    #64$, @#ERRVEC    ;SETUP LOCATION 4
6357 026060 052737 040000 177772 62$:  MOV    #376, SP         ;SETUP THE SP
6358 026066 000235    BIS    #BIT14, @#PIRQ    ;SET LEVEL 6
6359 026070 011616    SPL    5                ;SET CPU AT LEVEL 5
6360                                MOV    (SP), (SP)        ;EXECUTE YELL ZONE INSTR
6361 026072 005037 177772  ;FAILURE, PIR 6 CAME IN ON SL YELLOW
6362 026076 012706 001100 63$:  CLR    @#PIRQ          ;CLEAR PIR 6
6363 026102 104325    MOV    #STACK, SP       ;RESTORE THE SP
6364                                ERROR  325            ;PIR 6 CAME IN ON SL YELLOW
6365
6366
6367 026104 005077 153126  ;SECTION 21-PIR7 AND STACK LIMIT YELLOW
6368 026110 012767 026132 152772 64$:  CLR    @INT6ST          ;ENSURE BR6 FLAG CLEAR
6369 026116 012737 026150 000240    MOV    #65$, $LPERR    ;SETUP ERROR LOOP
6370 026124 012737 026162 000004    MOV    #66$, @#PIRQVEC   ;SETUP PIRQ VECTOR
6371 026132 012706 000376    MOV    #67$, @#ERRVEC    ;SETUP THE ERROR VECTOR
6372 026136 052737 100000 177772 65$:  MOV    #376, SP         ;SETUP THE SP
6373 026144 000236    BIS    #BIT15, @#PIRQ    ;SET LEVEL 7
6374 026146 011616    SPL    6                ;SET UP AT 6
6375                                MOV    (SP), (SP)        ;CAUSE YELL ZONE
6376 026150 005037 177772  ;FAILURE, PIR 7 CAME IN ON SL YELLOW
6377 026154 012706 001100 66$:  CLR    @#PIRQ          ;CLEAR LEVEL 7
6378 026160 104326    MOV    #STACK, SP       ;RESTORE THE SP
6379                                ERROR  326            ;PIR7 CAME IN ON SL YELLOW
6380
6381 026162 012706 001100  ;END OF TEST
6382 026166 005037 177772 67$:  MOV    #STACK, SP       ;RESET THE SP
6383 026172 012737 032636 000004    CLR    @#PIRQ          ;CLEAR LEVEL 7
6384 026200 005037 177766    MOV    #CPUSPUR, @#ERRVEC ;RESTORE LOCATION 4
6385                                CLR    @#CPUERR         ;CLEAR OUT ERROR REG
6386 026204 012777 000100 152724  ;***** CHGE2 *****
6387                                MOV    #BIT6, @ $TKS     ;ALLOW KEYBD INTERRUPTS
6388                                ;*****
6388 026212 000456    BR     TST50            ;CONTINUE
    
```



```

6389
6390
6391 026214 052777 000100 152720 ;ROUTINES TO GET BUS REQUESTS
6392 026222 000445 LEVEL4: BIS #100,@$TPS ;GET A BR 4
6393 026224 012777 000311 152776 BR WAIT ;GO WAIT
6394 026232 000441 LEVEL5: MOV #311,@INT5ST ;START BR 5 INTERRUPT
6395 026234 026727 152774 000100 BR WAIT ;GO WAIT
6396 026242 001004 LEVEL6: CMP INT6VEC,#100 ;IS BR6 DEVICE KW11-L?
6397 026244 012777 000100 152764 BNE 1$ ;BRANCH IF NO
6398 026252 000431 MOV #BIT6,@INT6ST ;SET INTERR BIT
6399 026254 012737 000001 172544 1$: MOV #1,@#PLKC ;GO WAIT
6400 026262 012777 000105 152746 MOV #105,@INT6ST ;SET KW11-P COUNTER
6401 026270 000422 BR WAIT ;SET INTERR BIT
6402 026272 012777 026330 152726 KILBR5: MOV #2$,@INT5VEC ;SET UP INTER 5 VEC
6403 026300 042777 000100 152722 BIC #BIT6,@INT5ST ;ENSURE INTERRUPT FLAG CLEAR
6404 026306 005037 177772 CLR @#PIRQ ;ENSURE PIRQ REG CLEAR
6405 026312 005077 152624 CLR @$TPS
6406 026316 000230 SPL 0 ;LET BR 5 COME IN
6407 026320 005000 CLR R0
6408 026322 005200 3$: INC R0
6409 026324 001376 BNE 3$
6410 026326 000406 BR ENDGET ;DON'T CHANGE STACK
6411 026330 062706 000004 2$: ADD #4,SP ;RESTORE THE SP
6412 026334 000403 BR ENDGET
6413 026336 005000 WAIT: CLR R0 ;WAIT FOR
6414 026340 005200 2$: INC R0 ;THE INTERRUPT
6415 026342 001376 BNE 2$ ;TO COME IN
6416 026344 000237 ENDGET: SPL 7
6417 026346 000207 RTS PC ;RETURN
6418
6419 ;*****
6420 ;*TEST 50 GPR SET 1 SELECT TEST
6421 ;*
6422 ;* THIS TEST FIRST ENSURES THAT PSW BIT 11 SETS AND CLEARS.
6423 ;* IT THEN ENSURES THAT GRAC GDREG SET 1 AND GSREG SET 1 GOES
6424 ;* HIGH FOR THE MUX SELECTS LL,LH,AND HL.
6425 ;* MUX SELECT HH WILL BE TESTED IN SUPERVISOR MODE.
6426 ;*****
6426 026350 000004 TST50: SCOPE
6427 026352 012767 027000 152614 MOV #TST51,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
6428 026360 012767 027000 152702 MOV #TST51,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
6429 026366 052737 004000 177776 BIS #BIT11,@#PSW ;SET REG SET SELECT BIT
6430 026374 032737 004000 177776 BIT #BIT11,@#PSW ;DID BIT 11 SET?
6431 026402 001004 BNE 20$ ;BRANCH IF YES
6432 026404 042737 004000 177776 BIC #BIT11,@#PSW
6433 026412 104327 ERROR 327 ;EITHER PSW BIT 11 DOES NOT SET OR TMCF
6434 ;CLK HI PS DOES NOT GO LOW OR PSW BIT
6435 ;11 DOES NOT GET TO OR THRU THE DMUX.
6436 026414 105037 177777 20$: CLRB @#PSW+1 ;CLEAR BIT 11
6437 026420 032737 004000 177776 BIT #BIT11,@#PSW ;DID BIT 11 CLEAR?
6438 026426 001402 BEQ 1$ ;BRANCH IF YES
6439 026430 104377 ERROR 377
6440 026432 000457 457 ;PSW BIT 11 DID NOT CLEAR
6441
6442 ;*****
6442 ;UPAD 5 TEST
6443 026434 005067 152534 1$: CLR $ESCAPE
6444 026440 012767 026446 152442 MOV #10$,$LPERR
    
```

```

6445 026446 012702 177777 10$: MOV #-1,R2 ;SETUP R2
6446 026452 052737 004000 177776 BIS #BIT11,@#PSW ;SELECT REG SET 1
6447 026460 005002 CLR R12 ;CLEAR R12
6448 026462 042737 004000 177776 BIC #BIT11,@#PSW ;GO BACK TO REG SET 0
6449 026470 005702 TST R2 ;DID R2 CLEAR?
6450 026472 001002 BNE 2$ ;BRANCH IF NO
6451 026474 104330 ERROR 330 ;CLEAR R10 ACTUALLY CLEARED R2
6452 026476 000403 BR 3$
6453 026500 020202 2$: CMP R2,R2 ;WAS R2 SOURCE AFFECTED BY R12?
6454 026502 001401 BEQ 3$ ;BRANCH IF NO
6455 026504 104331 ERROR 331 ;R2 SRC WAS AFFECTED BY CLR R12
6456
6457 ;:*****
6458 026506 012767 026514 152374 ;UPAD 0 TEST
6459 026514 005002 3$: MOV #11$,$LPERR
6460 026516 052737 004000 177776 11$: CLR R2 ;SETUP R2
6461 026524 012202 BIS #BIT11,@#PSW ;GO TO REG SET 1
6462 026526 042737 004000 177776 MOV (R12)+,R12 ;EXECUTE A UPAD 0 INSTRUCTION
6463 026534 005702 BIC #BIT11,@#PSW ;COME BACK TO SET 0.
6464 026536 001402 TST R2 ;DID DESTINATION CHANGE
6465 026540 104332 BEQ 4$ ;BRANCH IF NO
6466 026542 000403 ERROR 332 ;R2 DST GOT CHANGED ON (R12)+
6467 026544 020202 BR 5$
6468 026546 001401 4$: CMP R2,R2 ;DID SRC R2 GET CHANGED?
6469 026550 104333 BEQ 5$ ;BRANCH IF NO
6470 ERROR 333 ;R2 SRC GOT CHANGED ON (R12)+
6471 ;:*****
6472 026552 012767 026560 152330 ;UPAD 2 TEST
6473 026560 012705 026642 5$: MOV #12$,$LPERR
6474 026564 052737 004000 177776 12$: MOV #9$,R5 ;SETUP R5
6475 026572 012705 026616 BIS #BIT11,@#PSW ;GO TO SET 1
6476 026576 020527 026616 MOV #6$,R15 ;SETUP R15
6477 026602 001401 CMP R15,#6$ ;IS THERE STUCK BITS IN R15 SRC?
6478 026604 000475 BEQ 7$ ;BRANCH IF NO
6479 026606 020505 7$: BR TST51 ;:CAN'T DO THIS TEST, GO TO STUCK BIT TEST
6480 026610 001401 CMP R15,R15 ;IS THERE STUCK BITS IN R15 DST?
6481 026612 000472 BEQ 8$ ;BRANCH IF NO
6482 026614 006400 BR TST51 ;:CAN'T DO THIS TEST, GO TO STUCK BIT TEST
6483 026616 026705 177774 8$: MARK 0 ;EXECUTE A UPAD 2 INSTRUCTION
6484 026622 001425 6$: CMP 6$,R15 ;DID R15 DST GET LOADED?
6485 026624 042737 004000 177776 BEQ 16$ ;BRANCH IF YES
6486 026632 012706 001100 BIC #BIT11,@#PSW ;GO BACK TO SET 0
6487 026636 104334 MOV #STACK,SP ;RESTORE THE SP
6488 026640 000416 ERROR 334 ;R15 DST BAD AFTER UPAD2
6489 026642 022705 026642 9$: BR 16$
6490 026646 001407 CMP #9$,R15 ;IS DST ALSO BAD?
6491 026650 042737 004000 177776 BEQ 15$ ;BRANCH IF YES
6492 026656 012706 001100 BIC #BIT11,@#PSW
6493 026662 104335 MOV #STACK,SP ;RESTORE THE SP
6494 026664 000404 ERROR 335 ;R15 SRC DOES NOT SELECT ON UPAD2
6495 026666 042737 004000 177776 15$: BR 16$
6496 026674 104336 BIC #BIT11,@#PSW
6497 ERROR 336 ;PDRD PS11 DOES NOT GET TO GRAC
6498 ;:*****
6499 026676 012706 001100 ;TEST GRAB E19(2 & 3)
6500 026702 012767 026710 152200 16$: MOV #STACK,SP ;RESET THE SP
MOV #14$,$LPERR
    
```

```

6501 026710 042737 004000 177776 14$: BIC #BIT11,@#PSW ;GO BACK TO SET 0
6502 026716 005004 CLR R4 ;ENSURE R4 CLEAR
6503 026720 052737 004000 177776 BIS #BIT11,@#PSW ;GO TO SET 1
6504 026726 005204 INC R14 ;INCREMENT R14
6505 026730 042737 004000 177776 BIC #BIT11,@#PSW ;GO BACK TO SET 0
6506 026736 005704 TST R4 ;WAS R4 AFFECTED?
6507 026740 001401 BEQ 17$ ;BRANCH IF NO
6508 026742 104341 ERROR 341 ;GRAB DST SET 1 DID NOT GO LOW ON R14
6509
6510 ;*****
6511 026744 012767 026752 152136 17$: MOV #13$, $LPERR ;TEST GRAB E18(2 & 3)
6512 026752 052737 004000 177776 13$: BIS #BIT11,@#PSW ;GO TO SET 1
6513 026760 005004 CLR R14 ;ENSURE EVEN ADDRESS IN R14
6514 026762 012404 MOV (R14)+,R14 ;EXECUTE A UPAD 0
6515 026764 042737 004000 177776 BIC #BIT11,@#PSW ;GO BACK TO SET 0
6516 026772 005704 TST R4 ;WAS R4 AFFECTED?
6517 026774 001401 BEQ TST51 ;BRANCH IF NO
6518 026776 104342 ERROR 342 ;GRAB SRC SET 1 DID NOT GO LOW ON R14
6519
6520 ;*****
6521 ;*TEST 51 REGISTER SET 1 STUCK BIT TEST
6522 ;*
6523 ;* THIS TEST ENSURES THAT ALL BITS IN GPR'S R10 THRU R15 WORK
6524 ;*****
6525 027000 000004 TST51: SCOPE
6526 027002 012767 027214 152164 MOV #TST52,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
6527 027010 012767 027214 152252 MOV #TST52,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
6528 027016 005067 152140 CLR $TMP0 ;INITIALIZE PASS COUNT
6529 027022 052737 004000 177776 BIS #BIT11,@#PSW ;GO TO REG SET 1
6530 027030 012737 4$: MOV (PC)+,@(PC)+ ;SAVE EXPECTED VALUE
6531 027032 125252 5$: .WORD 125252
6532 027034 001164 .WORD $TMP1 ;ADDRESS OF $TEMP1
6533 027036 012700 MOV (PC)+,R10 ;PUT PATTERN IN R10
6534 027040 125252 6$: .WORD 125252
6535 027042 022700 CMP (PC)+,R10 ;IS R10 DST OK?
6536 027044 125252 7$: .WORD 125252
6537 027046 001040 BNE 1$ ;BRANCH IF NO
6538 027050 020027 CMP R10,(PC)+ ;IS R10 SRC OK?
6539 027052 125252 8$: .WORD 125252
6540 027054 001034 BNE 2$ ;BRANCH IF NO
6541 027056 010001 MOV R10,R11
6542 027060 020001 CMP R10,R11 ;IS R11 DST OK?
6543 027062 001032 BNE 1$ ;BRANCH IF NO
6544 027064 020100 CMP R11,R10 ;IS R11 SRC OK?
6545 027066 001027 BNE 2$ ;BRANCH IF NO
6546 027070 010102 MOV R11,R12
6547 027072 020002 CMP R10,R12 ;IS R12 DST OK?
6548 027074 001025 BNE 1$ ;BRANCH IF NO
6549 027076 020200 CMP R12,R10 ;IS R12 SRC OK?
6550 027100 001022 BNE 2$ ;BRANCH IF NO
6551 027102 010003 MOV R10,R13
6552 027104 020003 CMP R10,R13 ;IS R13 DST OK
6553 027106 001020 BNE 1$ ;BRANCH IF NO
6554 027110 020300 CMP R13,R10 ;IS R13 SRC OK?
6555 027112 001015 BNE 2$ ;BRANCH IF NO
6556 027114 010004 MOV R10,R14
    
```

6557	027116	020004				CMP	R10,R14		:IS R14 DST OK?
6558	027120	001013				BNE	1\$:BRANCH IF NO
6559	027122	020400				CMP	R14,R10		:IS R14 SRC OK?
6560	027124	001010				BNE	2\$:BRANCH IF NO
6561	027126	010005				MOV	R10,R15		:
6562	027130	020005				CMP	R10,R15		:IS R15 DST OK
6563	027132	001006				BNE	1\$:BRANCH IF NO
6564	027134	020500				CMP	R15,R10		:IS R15 SRC OK
6565	027136	001410				BEQ	3\$:BRANCH IF YES
6566	027140	042737	004000	177776		BIC	#BIT11,@#PSW		
6567	027146	104337			2\$:	ERROR	337		:BAD BITS IN GPR SET 1 SRC
6568	027150	042737	004000	177776	1\$:	BIC	#BIT11,@#PSW		
6569	027156	104340				ERROR	340		:BAD BITS IN GPR SET 1 DST
6570	027160	005767	151776		3\$:	TST	\$TMP0		:IS THIS FIRST PASS?
6571	027164	001013				BNE	TST52		:BRANCH IF NO
6572	027166	005267	151770			INC	\$TMP0		:SET PASS COUNT
6573	027172	005167	177634			COM	5\$:GO TO
6574	027176	005167	177636			COM	6\$:COMPLIMENT
6575	027202	005167	177636			COM	7\$:PATTERN
6576	027206	005167	177640			COM	8\$		
6577	027212	000706				BR	4\$:GO TEST COMPLIMENT PATTERN

```

6578
6579
6580
6581
6582
6583
6584
6585
6586
6587
6588
6589
6590
6591
6592
6593
6594
6595
6596
6597
6598
6599
6600
6601
6602
6603
6604
6605
6606
6607
6608
6609
6610
6611
6612

```

```

:*****
:*TEST 52          PSW HIGH BYTE BIT TEST
:*

```

```

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

```

```

TST52: SCOPE
MOV      #TST53,NEXTTST ;SAVE ADDRESS OF NEXT TEST
MOVB    #120,@#PSW+1    ;SET TO SUPER & PREVIOUS SUPER
CMPB    #120,@#PSW+1    ;IS IT OK?
BEQ     1$              ;BRANCH IF YES
MOV     #50000,$TMP0    ;SAVE EXPECTED VALUE
MOV     @#PSW,$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

```

```

6613 ;* SELECTED IN SUPERVISOR AND USER MODE
6614 ;:*****
6615 027364 000004 TST53: SCOPE
6616 027366 012767 027634 151674 MOV #TST54,NEXTTST ;SAVE ADDRESS OF NEXT TEST
6617 ;UPAD 5-SSP
6618 027374 012767 027402 151506 MOV #5$,$LPERR ;SETUP ERROR LOOP
6619 027402 012706 001100 5$: MOV #STACK,KSP ;SETUP THE KERNAL SP
6620 027406 052737 040000 177776 BIS #BIT14,@#PSW ;GO TO SUPER MODE
6621 027414 005006 CLR SSP ;CLEAR THE SUPER SP (UPAD 5)
6622 027416 042737 040000 177776 BIC #BIT14,@#PSW ;GO BACK TO KERNEL MODE
6623 027424 005706 TST KSP ;DID THE KSP CHANGE?
6624 027426 001003 BNE 1$ ;BRANCH IF NO
6625 027430 012706 001100 MOV #STACK,KSP ;RESTORE THE KSP
6626 027434 104346 ERROR 346 ;SUPER SP DOES NOT SELECT ON UPAD 5
6627 ;
6628 ;UPAD 0-SSP
6629 027436 012767 027444 151444 1$: MOV #6$,$LPERR ;SETUP ERROR LOOP
6630 027444 012706 001100 6$: MOV #STACK,KSP ;INITIALIZE THE SP
6631 027450 052737 040000 177776 BIS #BIT14,@#PSW ;GO TO SUPER MODE
6632 027456 012706 001776 MOV #1776,SSP ;SETUP SUPER SP
6633 027462 012606 MOV (SSP)+,SSP ;EXECUTE A UPAD 0 TYPE OPERATION
6634 027464 042737 040000 177776 BIC #BIT14,@#PSW ;GO BACK TO KERNEL
6635 027472 020627 001100 CMP KSP,#STACK ;DID KSP CHANGE?
6636 027476 001403 BEQ 2$ ;BRANCH IF NO
6637 027500 012706 001100 MOV #STACK,KSP ;RESTORE THE KSP
6638 027504 104347 ERROR 347 ;SUPER SP DOES NOT SELECTON UPAD 0
6639 ;
6640 ;UPAD 5-USP
6641 027506 012767 027514 151374 2$: MOV #7$,$LPERR ;SETUP ERROR LOOP
6642 027514 052737 040000 177776 7$: BIS #BIT14,@#PSW ;GO TO SUPER MODE
6643 027522 012706 177777 MOV #-1,SSP ;SET THE SSP TO A NON-ZERO NUMBER
6644 027526 052737 100000 177776 BIS #BIT15,@#PSW ;GO TO USER MODE
6645 027534 005006 CLR USP ;CLEAR R17
6646 027536 042737 100000 177776 BIC #BIT15,@#PSW ;GO BACK TO SUPER MODE
6647 027544 005706 TST SSP ;DID SSP CLEAR?
6648 027546 001003 BNE 3$ ;BRANCH IF NO
6649 027550 105037 177777 CLRB @#PSW+1 ;GO BACK TO KERNEL
6650 027554 104350 ERROR 350 ;USER SP DOES NOT SELECT ON UPAD 5
6651 ;
6652 ;UPAD 0-USP
6653 027556 012767 027564 151324 3$: MOV #8$,$LPERR ;SETUP ERROR LOOP
6654 027564 052737 040000 177776 8$: BIS #BIT14,@#PSW ;GO TO SUPER MODE
6655 027572 005006 CLR SSP ;ENSURE SSP CLEAR
6656 027574 052737 100000 177776 BIS #BIT15,@#PSW ;GO TO USER MODE
6657 027602 012706 001776 MOV #1776,USP ;SET USP TO EVEN NUMBER
6658 027606 012606 MOV (USP)+,USP ;EXECUTE A UPAD 0 TYPE INSTURCTION
6659 027610 042737 100000 177776 BIC #BIT15,@#PSW ;GO BACK TO SUPER
6660 027616 005706 TST SSP ;DID SSP CHANGE?
6661 027620 001403 BEQ 4$ ;BRANCH IF NO
6662 027622 105037 177777 CLRB @#PSW+1 ;GO BACK TO KERNEL
6663 027626 104351 ERROR 351 ;USER SP DOES NOT SELECT ON UPAD 0
6664 027630 105037 177777 4$: CLRB @#PSW+1 ;GO BACK TO KERNAL
6665 ;CONTINUE
6666 ;:*****
6667 ;*TEST 54 SUPER AND USER SP BIT TEST
6668 ;:*****
    
```

```

6669
6670
6671
6672
6673 027634 000004
6674 027636 012767 030000 151330
6675 027644 012767 030000 151416
6676 027652 005067 151306
6677 027656 052737 040000 177776
6678 027664 012706
6679 027666 052525
6680 027670 022706
6681 027672 052525
6682 027674 001403
6683 027676 105037 177777
6684 027702 104352
6685 027704 020627
6686 027706 052525
6687 027710 001403
6688 027712 105037 177777
6689 027716 104353
6690 027720 005737 177776
6691 027724 100404
6692 027726 052737 100000 177776
6693 027734 000753
6694 027736 005767 151222
6695 027742 001014
6696 027744 005167 177716
6697 027750 005167 177716
6698 027754 005167 177726
6699 027760 042737 100000 177776
6700 027766 005267 151172
6701 027772 000734
6702 027774 105037 177777
6703
6704
6705
6706
6707
6708
6709
6710
6711 030000 000004
6712 030002 012767 030234 151260
6713
6714
6715 030010 012706 001100
6716 030014 012746 177777
6717 030020 005000
6718 030022 052737 010000 177776
6719 030030 106606
6720 030032 022706 001100
6721 030036 001401
6722 030040 005200
6723 030042 020627 001100
6724 030046 001410
    
```

```

;*
;* THIS TEST ENSURES THAT THE SUPERVISOR AND USER STACK POINTERS
;* DON'T HAVE ANY STUCK BITS.
;*****
TST54: SCOPE
MOV #TST55,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
MOV #TST55,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
CLR $TMP1 ;CLEAR LOOP COUNT
BIS #BIT14,@#PSW ;GO TO SUPER MODE
4$: MOV (PC)+,SP ;PUT PATTERN IN SP
6$: .WORD 52525 ;DATA PATTERN
CMP (PC)+,SP ;IS DST OK?
7$: .WORD 52525 ;DATA PATTERN
BEQ 1$ ;BRANCH IF YES
CLRB @#PSW+1 ;GO BACK TO KERNEL
ERROR 352 ;DST FAILED
1$: CMP SP,(PC)+ ;IS SRC OK?
8$: .WORD 52525 ;DATA PATTERN
BEQ 2$ ;BRANCH IF YES
CLRB @#PSW+1 ;GO BACK TO KERNEL
ERROR 353 ;SRC FAILED
2$: TST @#PSW ;IS BIT 15 SET?
BMI 3$ ;BRANCH IF YES
BIS #BIT15,@#PSW ;GO TO USER MODE
BR 4$ ;GO CHECK USER STACK POINTER
3$: TST $TMP1 ;IS THIS SECOND PASS?
BNE 5$ ;BRANCH IF YES
COM 6$ ;CHANGE TEST
COM 7$ ;TO COMPLIMENT
COM 8$ ;PATTERN
BIC #BIT15,@#PSW ;GO BACK TO SUPER MODE
INC $TMP1 ;SET PASS COUNT
BR 4$ ;GO TEST COMPLIMENT PATTERN
5$: CLRB @#PSW+1 ;GO BACK TO KERNAL
;CONTINUE
;*****
;*TEST 55 MTP*DMO*DF6*PREVIOUS MODE(SUPER*USER)
;*
;* THIS TEST ENSURES THAT THE CORRECT SP'S ARE SELECTED WHEN EXECUTING
;* A MTP WITH DIFFERENT PREVIOUS MODE BITS SELECTED.
;*****
TST55: SCOPE
MOV #TST56,NEXTTST ;SAVE ADDRESS OF NEXT TEST
;SECTION 1-POP THE KSP AND PUT IT IN THE SSP
MOV #STACK,KSP ;SETUP THE KERNEL SP
MOV #-1,-(KSP) ;PUT -1 ON THE STACK
CLR R0 ;CLEAR ERROR COUNT
BIS #BIT12,@#PSW ;SET PREVIOUS MODE SUPER
MTPD SP ;EXECUTE INSTRUCTION UNDER TEST
CMP #STACK,KSP ;DID THE KSP DST COME OUT OK?
BEQ 2$ ;BRANCH IF YES
INC R0 ;SET THE ERROR COUNT
2$: CMP KSP,#STACK ;DID THE KSP SRC COME OUT OK?
BEQ 3$ ;BRANCH IF YES
    
```

```

6725 030050 012706 001100      MOV      #STACK,KSP      ;RESTORE THE SP
6726 030054 005700      TST      R0              ;DID DST ALSO FAIL?
6727 030056 001002      BNE      4$              ;BRANCH IF YES
6728 030060 104354      ERROR   354              ;KSP SRC WAS CHANGED
6729 030062 000431      BR       9$              ;
6730 030064 104355      4$:     ERROR   355              ;BOTH KSP SRC AND DST CHANGED
6731 030066 000427      BR       9$              ;
6732 030070 005700      3$:     TST      R0              ;DID KSP DST CHANGE
6733 030072 001404      BEQ      5$              ;BRANCH IF NO
6734 030074 012706 001100      MOV      #STACK,KSP      ;RESTORE THE SP
6735 030100 104356      ERROR   356              ;KSP DST WAS CHANGED
6736 030102 000421      BR       9$              ;
6737 030104 052737 040000 177776 5$:     BIS      #BIT14,@#PSW     ;GO TO SUPER MODE
6738 030112 022706 177777      CMP      #-1,SSP         ;DID SSP LOAD OK?
6739 030116 001412      BEQ      6$              ;BRANCH IF YES
6740 030120 010667 151030      MOV      SSP,$REG0       ;SAVE THE SSP
6741 030124 005006      CLR      SSP             ;FOR LOOPING
6742 030126 042737 040000 177776      BIC      #BIT14,@#PSW     ;GO BACK TO KERNEL
6743 030134 012767 177777 151014      MOV      #-1,$REG1       ;SAVE EXPECTED VALUE
6744 030142 104357      ERROR   357              ;SSP DID NOT LOAD PROPERLY
6745
6746      ;SECTION 2-POP THE KSP AND PUT IT IN THE USP
6747 030144 005006      6$:     CLR      SSP         ;ENSURE SSP CLEAR FOR ITERATIONS
6748 030146 042737 040000 177776 9$:     BIC      #BIT14,@#PSW     ;GO BACK TO KERNEL
6749 030154 012767 030162 150726      MOV      #7$,$LPERR      ;SET UP ERROR LOOP
6750 030162 012706 001100      7$:     MOV      #STACK,KSP     ;SETUP THE SP
6751 030166 012746 177777      MOV      #-1,-(KSP)       ;PUT -1 ON THE KERNEL STACK
6752 030172 052737 030000 177776      BIS      #30000,@#PSW     ;SET PREVIOUS MODE USER
6753 030200 106606      MTPD     SP              ;EXECUTE INSTRUCTION UNDER TEST
6754 030202 052737 140000 177776      BIS      #140000,@#PSW    ;GO TO USER MODE
6755 030210 020627 177777      CMP      USP,#-1         ;DID USER SP GET LOADED?
6756 030214 001404      BEQ      8$              ;BRANCH IF YES
6757 030216 005006      CLR      USP             ;CLEAR USER SP
6758 030220 105037 177777      CLRB    @#PSW+1         ;GO BACK TO KERNEL
6759 030224 104360      ERROR   360              ;USER SP DID NOT LOAD ON MTP
6760 030226 005006      8$:     CLR      USP         ;ENSURE USP CLEAR, IF ITERATING
6761 030230 105037 177777      CLRB    @#PSW+1         ;GO BACK TO KERNEL
6762
6763      ;CONTINUE
6764
6765      ;*****
6766      ;*TEST 56      MFP*DMO*DF6*PREVIOUS MODE SUPER
6767
6768      ;*      THE ONLY POSSIBLE WAY THIS TEST CAN FAIL IS THAT
6769      ;*      IRCC DMO (MFP+MTP) DOES NOT GO HIGH ON MFP.
6770      ;*      THIS WILL ONLY HAPPEN IF THE IR DECODE ROM HAS
6771      ;*      A BAD FIELD (R(MFP+MTP)).
6772      ;*      *****
6773      ;TST56:  SCOPE
6774      MOV      #TST57,$ESCAPE ;SAVE START ADDRESS OF NEXT TEST
6775      MOV      #TST57,NEXTTST ;SAVE START ADDRESS OF NEXT TEST
6776
6777      ;PUSH THE SSP ONTO THE KERNEL STACK
6778      MOV      #1$,$LPADR    ;SETUP LOOP ADR
6779      MOV      #1$,$LPERR    ;SETUP ERROR LOOP
6780      MOV      @#PSW,$TMP3    ;SAVE PSW
6781      BIT      #BIT4,@#PSW    ;IS T BIT ON?
6782      BEQ      1$            ;BRANCH IF NO
    
```

```

6781 030304 012746 000340      MOV      #PR7,-(SP)      ;PUT NEW PSW ON STACK
6782 030310 012746 030316      MOV      #1$,-(SP)     ;PUT RETURN ADDR ON STACK
6783 030314 000006                RIT                    ;TURN T BIT OFF
6784 030316 052737 040000 177776 1$:  BIS      #BIT14,@#PSW   ;GO TO SUPER MODE
6785 030324 005006                CLR      SSP           ;CLEAR THE SUPER SP
6786 030326 105037 177777      CLRB    @#PSW+1        ;GO BACK TO KERNEL
6787 030332 012706 001100      MOV      #STACK,KSP    ;SETUP THE KSP
6788 030336 012766 177777 177776  MOV      #-1,-2(KSP)    ;PUT KNOWN VALUE ON STACK
6789 030344 052737 010000 177776  BIS      #BIT12,@#PSW   ;SET PREVIOUS MODE SUPER
6790 030352 106506                MFPD    SP             ;EXECUTE INSTRUCTION UNDER TEST
6791 030354 005716                TST     (KSP)          ;DID STACK GET SUPER SP?
6792 030356 001401                BEQ     TST57          ;;BRANCH IF YES
6793 030360 104361                ERROR   361           ;IR DECODE ROM BAD
6794
6795
6796
6797
6798
6799
6800
6801
6802
6803

```

```

:*****
:*TEST 57          UPAD 7 IN USER MODE
:*
:*      THIS TEST ENSURES THAT A UPAD 7(OCCURS IN RTI) CAUSES THE USER
:*      STACK POINTER TO BE USED TO FETCH THE NEW PS AND PC.
:*
:*      IF PDRD PS14(1) DOES NOT GET TO THE GSAM(ON GRAC)
:*      THE TEST WILL BLOW UP.
:*****

```

```

6804 030362 000004                TST57: SCOPE
6805 030364 012767 030572 150676  MOV      #TST60,NEXTTST ;SAVE ADDRESS OF NEXT TEST
6806 030372 012706 001100      MOV      #STACK,KSP    ;SETUP THE KERNAL SP
6807 030376 052737 040000 177776  BIS      #BIT14,@#PSW   ;GO TO SUPER MODE
6808 030404 012706 001064      MOV      #1064,SSP     ;SETUP THE SSP
6809 030410 012716 030450      MOV      #1$,(SSP)     ;PUT ADDRESS OF 1$ ON SUPER STACK
6810 030414 012766 140340 000002  MOV      #140340,2(SSP) ;PUT NEW PSW ON SUPER STACK
6811 030422 052737 100000 177776  BIS      #BIT15,@#PSW   ;GO TO USER MODE
6812 030430 012706 001060      MOV      #1060,USP     ;SETUP THE USP
6813 030434 012716 030520      MOV      #2$,(USP)     ;PUT ADDRESS OF 2$ ON USER STACK
6814 030440 012766 140340 000002  MOV      #140340,2(USP) ;PUT NEW PSW ON USER STACK
6815 030446 000002                RTI                    ;EXECUTE INSTRUCTION THAT USES UPAD 7
6816                ;FAILURE, SUPER STACK POINTER WAS READ
6817 030450 022706 001070 1$:  CMP      #1070,USP     ;DID USP DST GET WRITTEN?
6818 030454 001004                BNE     3$            ;BRANCH IF NO
6819 030456 105037 177777      CLRB    @#PSW+1        ;GO BACK TO KERNEL
6820 030462 104362                ERROR   362           ;SSP WAS READ AND USP WAS WRITTEN
6821 030464 000415                BR      2$            ;
6822 030466 042737 100000 177776 3$:  BIC      #BIT15,@#PSW   ;GO TO SUPER MODE
6823 030474 022706 001070      CMP      #1070,SSP     ;WAS THE SSP WRITTEN>
6824 030500 001004                BNE     4$            ;BRANCH IF NO
6825 030502 105037 177777      CLRB    @#PSW+1        ;GO BACK TO KERNEL
6826 030506 104363                ERROR   363           ;SSP WAS READ AND WRITTEN
6827 030510 000403                BR      2$            ;
6828 030512 105037 177777 4$:  CLRB    @#PSW+1        ;GO BACK TO KERNEL
6829 030516 104364                ERROR   364           ;DON'T KNOW WHAT HAPPENED
6830                ;READ OK,NOW CHECK WRITE
6831 030520 052737 140000 177776 2$:  BIS      #140000,@#PSW ;SETUP PSW
6832 030526 022706 001064      CMP      #1064,USP     ;DID THE USP GET WRITTEN?
6833 030532 001415                BEQ     5$            ;BRANCH IF YES
6834 030534 042737 100000 177776  BIC      #BIT15,@#PSW   ;GO TO SUPER MODE
6835 030542 022706 001064      CMP      #1064,SSP     ;DID THE SSP GET WRITTEN?
6836 030546 001004                BNE     6$            ;BRANCH IF NO

```


6837 030550 105037 177777
 6838 030554 104365
 6839 030556 000403
 6840 030560 105037 177777
 6841 030564 104366
 6842 030566 105037 177777
 6843
 6844
 6845
 6846
 6847
 6848
 6849
 6850
 6851 030572 000004
 6852 030574 000237
 6853 030576 052737 040000 177776
 6854 030604 000230
 6855 030606 042737 040000 177776
 6856 030614 013767 177776 150364
 6857 030622 042767 177437 150356
 6858 030630 022767 000340 150350
 6859 030636 001404
 6860 030640 012767 000340 150314
 6861 030646 104367
 6862
 6863
 6864
 6865
 6866
 6867
 6868
 6869
 6870
 6871
 6872
 6873
 6874
 6875
 6876
 6877
 6878
 6879
 6880
 6881 030650 000004
 6882 030652 013767 177776 150304
 6883 030660 012767 174000 150274
 6884 030666 005767 150400
 6885 030672 001403
 6886 030674 052767 000400 150260
 6887 030702
 6888 030702 012767 031414 150360
 6889 030710 005067 150272
 6890
 6891
 6892 030714 012767 030722 150166

```

        CLRB    @#PSW+1      ;GO TO KERNEL MODE
        ERROR   365          ;USP WAS READ BUT SSP WAS WRITTEN
        BR      5$
6$:    CLRB    @#PSW+1      ;GO TO KERNEL MODE
        ERROR   366          ;USP WAS READ BUT REG 7 WAS WRITTEN
5$:    CLRB    @#PSW+1      ;GO TO KERNEL
        ;CONTINUE

;*****
;*TEST 60      SPL*SUPERVISOR MODE
;*
;*          THIS TEST ENSURES THAT SPL DOES NOT LOAD THE PSW IN SUPER+USER MODE.
;*****
TST60:  SCOPE
        SPL     7            ;SET CPU AT LEVEL 7
        BIS     #BIT14,@#PSW ;GO TO SUPER MODE
        SPL     0            ;TRY AND CLEAR PRIORITIES
        BIC     #BIT14,@#PSW ;GO BACK TO KERNEL
        MOV     @#PSW,$ERPSW ;SAVE PSW
        BIC     #^C<PR7>,$ERPSW ;MASK OFF PRIORITIES
        CMP     #PR7,$ERPSW  ;DID SPL CLR PRIORITIES
        BEQ     TST61        ;:BRANCH IF NO
        MOV     #PR7,$TMP0   ;SAVE EXPECTED VALUE
        ERROR   367          ;SPL WORKED IN SUPER MODE
;*****
;*TEST 61      PSW CLOCKING TEST
;*
;*          THIS TEST ENSURES THAT ALL THE BITS IN THE PSW GET LOADED WHEN THE
;*          FOLLOWING SIGNALS ARE TRUE:
;*          1) LOAD PS*KERNEL MODE, AND 2) LOAD PS*KERNEL DATI.
;*
;*          IT ALSO ENSURES THAT THE PRESET LOGIC ON BITS 11, 12, 13,
;*          14, AND 15 FUNCTIONS PROPERLY.
;*
;*          FOLLOWING IS A TABLE TO DESCRIBE THE PSW FOR EACH SECTION
;*
;          SECTION PSW AT START      PSW ON STK(OR VECTOR)      EXPEC PSW
;*          1          000XXX          174XXX          174XXX (175XXX IF KB11-E/EM)
;*          2          174XXX          000XXX          174XXX
;*          3          040XXX          134XXX          174XXX
;*          4          144XXX          000XXX          030XXX
;*          5          030XXX          000XXX          000XXX
;*****
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 $ERPSW CLEAR
;*****
;*RTI IN KERNEL MODE WITH NEW PSW<15:11>174 (175 IN KB11-E) AND OLD PSW<15:11>000
        MOV     #11$,$LPERR   ;SETUP ERROR LOOP
    
```

```
6893 030722 012706 001072      11$:  MOV      #1072,SP      ;SETUP THE SP
6894 030726 012716 030742      MOV      #1$, (SP)      ;PUT ADDRESS OF 1$ ON THE STACK
6895 030732 012766 177757 000002  MOV      #177757,2(SP)  ;SET ALL BITS IN NEW PSW EXCEPT T
6896 030740 000002      RTI                      ;EXECUTE INSTRUCTION
6897 030742 005767 150324      1$:  TST      KB11E        ;IS THIS A KB11-E OE KB11-EM PROCESSOR?
6898 030746 001405      BEQ      20$            ;BR IF NOT
6899 030750 122737 000371 177777  CMPB     #371,@#PSW+1   ;NEW PSW LOAD OK?
6900 030756 001413      BEQ      2$            ;BR IF YES
6901 030760 000404      BR       21$          ;OTHERWISE REPORT ERROR
6902 030762 122737 000370 177777 20$:  CMPB     #370,@#PSW+1   ;DID NEW PSW LOAD OK?
6903 030770 001406      BEQ      2$            ;BRANCH IF YES
6904 030772 113767 177777 150207 21$:  MOVB     @#PSW+1,$ERPSW+1 ;SAVE ERROR PSW
6905 031000 105037 177777      CLRB     @#PSW+1      ;GO BACK TO KERNEL
6906 031004 104370      ERROR    370          ;A HIGH BYTE BIT DID NOT SET IN PSW
6907                                     ;:*****
6908                                     ;RTI IN USER MODE WITH NEW PSW<15:11>000 AND OLD PSW<15:11>174
6909 031006 012767 174000 150146 2$:  MOV      #174000,$TMPO  ;SAVE EXPECTED VALUE OF PSW
6910 031014 012767 031022 150066      MOV      #12$, $LPERR  ;SETUP ERROR LOOP
6911 031022 152737 000370 177777 12$:  BISB     #370,@#PSW+1   ;SETUP THE PSW
6912 031030 012706 001072      MOV      #1072,USP     ;SETUP USER SP
6913 031034 012716 031046      MOV      #3$, (USP)    ;PUT ADDRESS OF 3$ ON STACK
6914 031040 005066 000002      CLR      2(USP)        ;PUT NEW PSW ON STACK
6915 031044 000002      RTI                      ;EXECUTE INSTRUCTION
6916 031046 122737 000370 177777 3$:  CMPB     #370,@#PSW+1   ;DID PSW STAY THE SAME?
6917 031054 001406      BEQ      4$            ;BRANCH IF YES
6918 031056 113767 177777 150123  MOVB     @#PSW+1,$ERPSW+1 ;SAVE PSW
6919 031064 105037 177777      CLRB     @#PSW+1      ;GO BACK TO KERNEL
6920 031070 104371      ERROR    371          ;A HIGH BYTE BIT CLEARED IN PSW
6921                                     ;:*****
6922                                     ;RTI IN SUPERVISOR MODE WITH NEW PSW<15:11>134 AND OLD PSW<15:11>040
6923                                     ;CHECKS PRESETS ON BITS 11, 12, 13, AND 15
6924 031072 012767 031100 150010 4$:  MOV      #13$, $LPERR  ;SETUP ERROR LOOP
6925 031100 012737 040340 177776 13$:  MOV      #40340,@#PSW  ;GO TO SUPER AND CLEAR PREVIOUS MODE AND REG BITS
6926 031106 012706 001072      MOV      #1072,$SSP    ;SET THE SSP
6927 031112 012716 031126      MOV      #5$, ($SSP)   ;PUT ADDRESS OF 5$ ON STACK
6928 031116 012766 134000 000002  MOV      #134000,2($SSP) ;PUT NEW PSW ON STACK
6929 031124 000002      RTI                      ;EXECUTE INSTRUCTION
6930 031126 122737 000370 177777 5$:  CMPB     #370,@#PSW+1   ;DID ALL BITS SET?
6931 031134 001406      BEQ      10$           ;BRANCH IF YES
6932 031136 113767 177777 150043  MOVB     @#PSW+1,$ERPSW+1 ;SAVE PSW HIGH BYTE
6933 031144 105037 177777      CLRB     @#PSW+1      ;GO BACK TO KERNEL
6934 031150 104372      ERROR    372          ;BITS <15,13:11> DID NOT PRESET
6935                                     ;:*****
6936                                     ;IOT IN USER MODE WITH NEW PSW<15:11>000 AND OLD PSW<15:11>144
6937 031152 012767 031160 147730 10$:  MOV      #14$, $LPERR  ;SETUP ERROR LOOP
6938 031160 112737 000310 177777 14$:  MOVB     #310,@#PSW+1   ;SETUP PSW
6939 031166 012737 031204 000020      MOV      #6$, @#IOTVEC ;PUT ADDRESS OF 6$ IN IOT VECTOR
6940 031174 012737 000340 000022      MOV      #PR7,@#IOTVEC+2 ;PUT NEW PSW IN IOT VECTOR+2
6941 031202 000004      IOT                      ;EXECUTE INSTRUCTION
6942 031204 122737 000060 177777 6$:  CMPB     #60,@#PSW+1   ;DID NEW PSW COME UP OK?
6943 031212 001411      BEQ      7$            ;BRANCH IF YES
6944 031214 113767 177777 147765  MOVB     @#PSW+1,$ERPSW+1 ;SAVE PSW
6945 031222 012767 030000 147732      MOV      #30000,$TMPO  ;SAVE EXPECTED VALUE
6946 031230 105037 177777      CLRB     @#PSW+1      ;MAKE SURE KERNEL MODE
6947 031234 104373      ERROR    373          ;PSW HIGH BYTE WRONG
6948                                     ;:*****
```

```

6949 ;IOT IN KERNEL MODE WITH NEW PSW<15:11>000 AND OLD PSW<15:11>030
6950 031236 012767 031244 147644 7$: MOV #15$, $LPERR ;SETUP ERROR LOOP
6951 031244 112737 000060 177777 15$: MOVB #60, @#PSW+1 ;SETUP PSW
6952 031252 012737 031270 000020 MOV #8$, @#IOTVEC ;SETUP IOT VECTOR
6953 031260 012737 030340 000022 MOV #30340, @#IOTVEC+2 ;SETUP NEW PSW
6954 031266 000004 IOT ;EXECUTE INSTRUCTION
6955 031270 105737 177777 8$: TSTB @#PSW+1 ;IS PSW HIGH BYTE CLEAR?
6956 031274 001430 BEQ 9$ ;BRANCH IF YES
6957 031276 113767 177777 147703 MOVB @#PSW+1, $ERPSW+1 ;SAVE PSW
6958 031304 005067 147652 CLR $TMPO ;SAVE EXPECTED VALUE
6959 031310 104374 ERROR 374 ;PREVIOUS MODE BITS DID NOT CLEAR
6960 ;*****
6961 ;RESET IN SUPER MODE
6962 031312 012737 074357 177776 MOV #74357, @#PSW ;SETUP PSW
6963 031320 000005 RESET ;EXECUTE INSTRUCTION
6964 031322 013767 177776 147656 MOV @#PSW, $ERPSW ;SAVE PSW
6965 031330 026727 147652 074340 CMP $ERPSW, #74340 ;WAS PSW OK?
6966 031336 001412 BEQ 16$ ;BRANCH IF YES
6967 031340 012767 074340 147614 MOV #74340, $TMPO ;SAVE EXPECTED VALUE
6968 031346 005037 177776 CLR @#PSW
6969 031352 104377 ERROR 377 ;RESET AFFECTED BITS IN PSW
6970 031354 000461 461
6971 031356 012737 033252 000020 9$: MOV $$SCOPE, @#IOTVEC ;RESTORE IOTVEC
6972 031364 032767 000020 147572 16$: BIT #BIT4, $TMP1 ;WAS T BIT ON?
6973 031372 001410 BEQ TST62 ;BRANCH IF NO
6974 031374 012706 001074 MOV #1074, SP ;SETUP THE STACK
6975 031400 016716 147664 MOV NEXTTST, (SP) ;PUT ADDRESS OF NEXT TEST ON STACK
6976 031404 012766 000360 000002 MOV #360, 2(SP) ;SET T BIT ON STACK
6977 031412 000002 RTI ;TURN T BIT ON
6978 ;*****
6979 ;*TEST 62 ILLEGAL HALT
6980 ;* THIS TEST ENSURES THAT A HALT IN SUPER OR USER MODE WILL TRAP TO
6981 ;* LOCATION 4.
6982 ;*
6983 ;* IF BEN6 FAILS EXECUTION WOULD GO TO FET.04 WHICH WOULD
6984 ;* CAUSE THE HALT TO LOOK LIKE A NOP.
6985 ;* IF TMCE SET CONF GOES LOW THE PROCESSOR WILL HALT AT LOCALLOCATION 1$.
6986 ;*
6987 ;* THE CPU ERROR REGISTER BIT 7 IS ALSO TESTED HERE.
6988 ;*****
6989 031414 000004 TST62: SCOPE
6990 031416 012767 031530 147644 MOV #TST63, NEXTTST ;SAVE ADDRESS OF NEXT TEST
6991 031424 012706 001100 MOV #STACK, SP ;SETUP THE SP
6992 031430 012737 031456 000004 MOV #1$, @#ERRVEC ;SETUP ERRVEC
6993 031436 052737 040000 177776 BIS #BIT14, @#PSW ;GO TO SUPER MODE
6994 031444 000000 HALT ;EXECUTE INSTRUCTION UNDER TEST
6995 ;FAILURE, NO TRAP
6996 031446 105037 177777 CLRB @#PSW+1 ;GO BACK TO KERNEL
6997 031452 104377 ERROR 377 ;ILLEGAL HALT DID NOT
6998 031454 000417 417 ;TRAP TO 4
6999 031456 032737 000200 177766 1$: BIT #BIT7, @#CPUERR ;IS ERROR REG OK?
7000 031464 001010 BNE 2$ ;BRANCH IF YES
7001 031466 013767 177766 147460 MOV @#CPUERR, $REGO ;SAVE FOR TYPOUT
7002 031474 012767 000200 147460 MOV #200, $TMPO ;SAVE EXPECTED VALUE
7003 031502 104377 ERROR 377 ;BIT 7 DID NOT SET
7004 031504 000420 420
    
```

```

7005 031506 005037 177766      2$: CLR @#CPUERR ;CLEAR BIT 7
7006 031512 005737 177766      TST @#CPUERR ;DID BIT 7 CLEAR?
7007 031516 001401              BEQ 3$ ;BRANCH IF YES
7008 031520 104256              ERROR 256 ;BIT 7 DID NOT CLEAR
7009 031522 012737 032636 000004 3$: MOV #CPUSPUR,@#ERRVEC ;RESTORE ERRVEC
7010                                     ;CONTINUE
7011                                     ;*****
7012                                     ;*TEST 63 WAIT
7013                                     ;*
7014                                     ;* THIS TEST ENSURES THAT THE WAIT INSTRUCTION WORKS PROPERLY.
7015                                     ;* IT FIRST EXECUTES WITH A LEVEL 4 INTERRUPT.
7016                                     ;* THEN THE T BIT IS ENABLED TO ENSURE THAT THE INTERRUPT
7017                                     ;* OCCURS AND NOT THE T BIT TRAP.
7018                                     ;*****
7019 031530 000004      TST63: SCOPE
7020 031532 012767 032024 147530      MOV #TST64,NEXTTST ;SAVE ADDRESS OF NEXT TEST
7021 031540 012767 003706 147336      MOV #^D1990,$ICNT ;ADJUST ITERATION COUNT
7022 031546 012767 031554 147332      MOV #10$,$LPADR ;SETUP LOOP ADR
7023 031554 012737 031624 000064 10$: MOV #2$,@#TPVEC ;SETUP TELEPRINTER VECTOR
7024 031562 012767 031570 147320      MOV #1$,$LPERR ;SETUP ERROR LOOP
7025 031570 012706 001100      1$: MOV #STACK,SP ;INITIALIZE THE SP
7026 031574 000233      SPL 3
7027 031576 012777 000100 147336      MOV #BIT6,@$TPS ;SET PRINTER INTERRUPT FLAG
7028 031604 012777 000015 147332      MOV #15,@$TPB ;SEND CHARACTER
7029 031612 000001      WAIT ;EXECUTE INSTRUCTION UNDER TEST
7030                                     ;FAILURE
7031 031614 005077 147322      CLR @$TPS ;CLEAR INTERR ENABLE BIT
7032 031620 104377      ERROR 377 ;WAIT INSTRUCTION DID NOT
7033 031622 000440      440 ;WAIT FOR INTERRUPT
7034                                     ;TEST OK, NO TRY WITH T BIT
7035 031624 012767 031646 147256 2$: MOV #3$,$LPERR ;SETUP ERROR LOOP
7036 031632 012737 031702 000014      MOV #4$,@#TBITVEC ;SETUP T BIT VECTOR
7037 031640 012737 031724 000064      MOV #5$,@#TPVEC ;SETUP TELEPRINTER VECTOR
7038 031646 012706 001100      3$: MOV #STACK,SP ;INITIALIZE THE SP
7039 031652 012746 000020      MOV #20,-(SP) ;PUT SP ON STACK TO TURN ON T BIT
7040 031656 012746 031700      MOV #6$,-(SP) ;PUT RETURN ADDRESS ON STACK
7041 031662 052777 000100 147252      BIS #BIT6,@$TPS ;SET PRINTER INTERRUPT BIT
7042 031670 012777 000015 147246      MOV #15,@$TPB ;SEND CHARACTER
7043 031676 000006      RTT ;TURN T BIT ON
7044 031700 000001      6$: WAIT ;EXECUTE INSTRUCTION UNDER TEST
7045                                     ;FAILURE, T BIT CAME IN
7046 031702 005077 147234      4$: CLR @$TPS ;CLEAR PRINTER STATUS
7047 031706 012706 001100      MOV #STACK,SP ;RESTORE THE SP
7048 031712 012737 032622 000014      MOV #RTRN,@#TBITVEC ;RESTORE T BIT VECTOR
7049 031720 104377      ERROR 377 ;T BIT CAUSED WAIT TO
7050 031722 000441      441 ;QUIT
7051                                     ;TEST OK, NOW TRY PIRQ
7052 031724 012737 032622 000014 5$: MOV #RTRN,@#TBITVEC ;RESTORE T BIT VECTOR
7053 031732 012767 031754 147150      MOV #7$,$LPERR ;SETUP ERROR LOOP
7054 031740 012737 032000 000064      MOV #8$,@#TPVEC ;SETUP TP VECTOR
7055 031746 012737 032014 000240      MOV #9$,@#PIRQVEC ;SETUP PIRQ VECTOR
7056 031754 012706 001100      7$: MOV #STACK,SP ;SETUP THE SP
7057 031760 012737 100000 177772      MOV #BIT15,@#PIRQ ;SET PIR LEVEL 7
7058 031766 012777 000015 147150      MOV #15,@$TPB ;SEND CHARACTER
7059 031774 000233      SPL 3 ;LOWER CPU
7060 031776 000001      WAIT ;EXECUTE INSTRUCTION UNDER TEST

```

7061
7062 032000 005077 147136
7063 032004 005037 177772
7064 032010 104377
7065 032012 000442
7066
7067 032014 005077 147122
7068 032020 005037 177772
7069
7070
7071
7072
7073
7074
7075
7076
7077
7078
7079
7080
7081
7082
7083

```
:FAILURE, PIRQ DID NOT INTERRUPT
8$: CLR @STPS ;CLEAR TP STATUS
    CLR @NPIRQ ;CLEAR PIR LEVEL 7
    ERROR 377 ;PIRQ DID NOT CAUSE
        442 ;INTERRUPT

:TEST OK
9$: CLR @STPS ;CLEAR INTERRUPT FLAG
    CLR @NPIRQ ;CLEAR PIR LEVEL 7
        ;CONTINUE
```

```
::*****
:*TEST 64 CHECK MFPT INSTRUCTION (KB-11E/EM ONLY)
:* THE MFPT INSTRUCTION IS NOT AVAILABLE ON THE 11/70 BUT
:* IF THIS IS AN KB-11E/EM THIS TEST IS RUN. MFPT RETURNS
:* DATA TO R0 IN THE FOLLOWING FORMAT:
:* BIT 0 - 1 INDICATES 11/44 CPU
:* BIT 1 - 1 INDICATES KB-11E/EM CPU (SHOULD ALWAYS COME UP IN THIS TEST)
:* BIT 8 - 1 INDICATES CISP PRESENT
:* BIT 9 - 1 INDICATES FP PRESENT
:*
:* THIS TEST MASKS OFF BITS 8 AND 9 AND ONLY CHECKS THAT BITS 0 AND 1
:* COME BACK CORRECTLY AND THAT THE OTHER BITS ARE CLEAR.
```

7084 032024 000004
7085 032026 012767 032076 147234
7086 032034 005737 001272
7087 032040 001002
7088 032042 000167 000030
7089 032046
7090 032046 012703 000002
7091 032052 000007
7092 032054 042700 001400
7093 032060 020003
7094 032062 001405
7095 032064 010337 001164
7096 032070 010037 001162
7097 032074 104173
7098 032076
7099 032076

```
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

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:

7100
7101
7102
7103
7104
7105
7106
7107
7108
7109
7110
7111
7112

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

7113 032076 000004
7114 032100 012767 032336 147162
7115 032106 005767 146766
7116 032112 001006

```
TST65: SCOPE
      MOV #SEOP,NEXTTST ;SAVE ADDRESS OF SEOP
      TST $PASS ;IS THIS FIRST PASS?
      BNE 1$ ;BRANCH IF NO
```

```

7117 032114 032737 000001 177570      BIT      #SW0,@#SWR      ;IS SWITCH 0 ON?
7118 032122 001403                BEQ      2$            ;BRANCH IF NO
7119 032124 104400 072541                TYPE     ,EM717       ;TYPE MESSAGE(SKIPPING TEST)
7120 032130 000502                1$: BR      $EOP        ;EXIT
7121 032132 005737 000042                2$: TST     @#42       ;IS MONITOR ACT11?
7122 032136 001077                BNE     $EOP        ;BRANCH IF YES
7123 032140 012700 166667                MOV     #166667,R0    ;SETUP R0
7124 032144 104400 072564                TYPE     ,EM720       ;TYPE MESSAGE
7125 032150 032737 000200 177570      BIT      #SW7,@#SWR    ;IS SWITCH 7 ON?
7126 032156 001416                BEQ     3$            ;BRANCH IF NO
7127 032160 012767 032202 146774      MOV     #4$+2,$TMPO   ;SAVE PC
7128 032166 016746 146770                MOV     $TMPO,-(SP)   ;;SAVE $TMPO FOR TYPEOUT
7129                                ;;TYPE ADDRESS
7130 032172 104402                TYPOC   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7131 032174 104400 072723                TYPE     ,EM721       ;TYPE MESSAGE
7132 032200 000005                4$: RESET  ;;EXECUTE INSTRUCTION
7133 032202 032737 000200 177570      BIT      #SW7,@#SWR    ;HAS SWITCH 7 CHANGED?
7134 032210 001416                BEQ     5$            ;BRANCH IF YES
7135 032212 000772                BR      4$            ;LOOP
7136 032214 012767 032236 146740      3$: MOV     #6$+2,$TMPO ;SAVE PC
7137 032222 016746 146734                MOV     $TMPO,-(SP)   ;;SAVE $TMPO FOR TYPEOUT
7138                                ;;TYPE ADDRESS
7139 032226 104402                TYPOC   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7140 032230 104400 072723                TYPE     ,EM721       ;TYPE MESSAGE
7141 032234 000005                6$: RESET  ;;EXECUTE INSTRUCTION
7142 032236 032737 000200 177570      BIT      #SW7,@#SWR    ;HAS SWITCH 7 CHANGED?
7143 032244 001773                BEQ     6$            ;BRANCH IF NO
7144                                ;
7145                                ;WAIT TEST
7146 032246 104400 072564                5$: TYPE     ,EM720       ;TYPE MESSAGE
7147 032252 012767 032316 146702      MOV     #7$,$TMPO     ;SAVE PC
7148 032260 016746 146676                MOV     $TMPO,-(SP)   ;;SAVE $TMPO FOR TYPEOUT
7149                                ;;TYPE ADDRESS
7150 032264 104402                TYPOC   ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7151 032266 104400 072761                TYPE     ,EM722       ;TYPE MESSAGE
7152 032272 005077 146642                CLR     @TKB         ;CLEAR KEYBOARD BUFFER
7153 032276 012737 032316 #000060      MOV     #7$,@TKVEC    ;SETUP KEYBOARD VECTOR
7154 032304 052777 000100 146624      BIS     #BIT6,@TKS    ;SET INTERRUPT ENABLE BIT
7155 032312 000233                SPL     3             ;LOWER PRIORITY FOR INTERRUPT
7156 032314 000001                WAIT    ;;EXECUTE INSTRUCTION
7157 032316 012737 034626 000060      7$: MOV     #QUIT,@TKVEC ;RESTORE TKVECTOR
7158 032324 005077 146610                CLR     @TKB         ;ENSURE BUFFER CLEAR
7159 032330 152777 000100 146600      BISB   #BIT6,@TKS    ;SET INTERRUPT ENABLE FLAG
7160                                ;CONTINUE
7161                                ;*****
7162                                ;
7163                                ;.SBTTL  END OF PASS ROUTINE
7164                                ;
7165                                ;*INCREMENT THE PASS NUMBER ($PASS)
7166                                ;*INDICATE END-OF-PROGRAM AFTER 1 PASSES THRU THE PROGRAM
7167                                ;*TYPE "END PASS #XXXXX TOTAL NUMBER OF ERRORS SINCE LAST REPORT YYYYY"
7168                                ;*WHERE XXXXX AND YYYYY ARE DECIMAL NUMBERS
7169                                ;*IF SW12=1 INHIBIT TRACE TRAP
7170                                ;*IF THERES A MONITOR GO TO IT
7171                                ;*IF THERE ISN'T JUMP TO LOOP
7172
    
```

```

7173 032336          $EOP:
7174 032336 000004          SCOPE
7175 032340 005067 146536          CLR $STNM          ;;ZERO THE TEST NUMBER
7176 032344 005067 146622          CLR $TIMES          ;;ZERO THE NUMBER OF ITERATIONS
7177 032350 005267 146524          INC $PASS          ;;INCREMENT THE PASS NUMBER
7178 032354 042767 100000 146516          BIC #100000,$PASS  ;;DON'T ALLOW A NEG. NUMBER
7179 032362 005327          DEC (PC)+          ;;LOOP?
7180 032364 000001          $EOPCT: .WORD 1
7181 032366 003076          BGT $DOAGN          ;;YES
7182 032370 012737          MOV (PC)+,@(PC)+  ;;RESTORE COUNTER
7183 032372 000001          $ENDCT: .WORD 1
7184 032374 032364          $EOPCT
7185 032376 104400 032404          TYPE ,65$          ;;TYPE ASCIZ STRING
7186 032402 000407          BR 64$            ;;GET OVER THE ASCIZ
7187          ;;65$: .ASCIZ <12><15>/END PASS #/
7188 032422          64$:
7189 032422 016746 146452          MOV $PASS,-(SP)   ;;SAVE $PASS FOR TYPEOUT
7190          ;;TYPE PASS NUMBER
7191 032426 104410          TYPDS          ;;GO TYPE--DECIMAL ASCII WITH SIGN
7192 032430 104400 032436          TYPE ,67$          ;;TYPE ASCIZ STRING
7193 032434 000421          BR 66$            ;;GET OVER THE ASCIZ
7194          ;;67$: .ASCIZ / TOTAL ERRORS SINCE LAST REPORT /
7195 032500          66$:
7196 032500 016746 146406          MOV $ERTTL,-(SP)  ;;SAVE $ERTTL FOR TYPEOUT
7197          ;;TOTAL NUMBER OF ERRORS
7198 032504 104410          TYPDS          ;;GO TYPE--DECIMAL ASCII WITH SIGN
7199 032506 104400 001203          TYPE , $CRLF          ;;TYPE CARRIAGE RETURN, LINE FEED
7200 032512 005067 146374          CLR $ERTTL          ;;CLEAR ERROR TOTAL
7201 032516 013700 000042          $GET42: MOV @#42,R0  ;;GET MONITOR ADDRESS
7202 032522 001420          BEQ $DOAGN          ;;BRANCH IF NO MONITOR
7203 032524 005046          CLR -(SP)          ;;INSURE THE 'T' BIT IS CLEAR
7204 032526 012746 032534          MOV #$CLR.T,-(SP) ;;SETUP FOR AN RTI OR RTT
7205 032532 000433          BR $RTRN          ;;GO DO AN RTI OR RTT TO LOAD THE PSW
7206          ;;WITH A CLEARED 'T' BIT
7207 032534          $CLR.T:
7208 032534 012703 000001          MOV #1,R3          ;;MAKE R3=1
7209 032540 004767 002120          JSR PC,CHAINQ      ;;GO RESTORE MONITOR
7210 032544 013700 000042          MOV @#42,R0          ;;INSURE R0 CONTAINS THE MONITORS
7211 032550 001405          BEQ $DOAGN          ;;RETURN ADDRESS
7212 032552 000005          RESET          ;;CLEAR THE WORLD
7213 032554 004710          $ENDAD: JSR PC,(R0)  ;;GO TO MONITOR
7214 032556 000240          NOP          ;;SAVE ROOM
7215 032560 000240          NOP          ;;FOR
7216 032562 000240          NOP          ;;ACT11
7217 032564          $DOAGN:
7218 032564 013746 177776          MOV @#PS,-(SP)     ;;PUT THE PS ON THE STACK AND
7219 032570 042716 000020          BIC #20,(SP)       ;;CLEAR THE 'T' BIT
7220 032574 032737 010000 177570          BIT #BIT12,@#SWR   ;;RUN WITH TRACE TRAP?
7221 032602 001005          BNE 1$            ;;BR IF NO
7222 032604 005167 000020          COM $TBIT          ;;IS IT TIME FOR TRACE TRAP
7223 032610 100402          BMI 1$            ;;BR IF NO
7224 032612 052716 000020          BIS #20,(SP)       ;;SET TRACE TRAP
7225 032616 012746 032624          1$: MOV #$LOOP,-(SP) ;;JUMP TO START OF TEST
7226 032622 000002          $RTRN: RTI          ;;RETURN--THIS IS CHANGED TO
7227          ;;AN 'RTT' IF 'RTT' IS A LEGAL
7228          ;;INSTRUCTION
    
```

```

7229 032624          $LOOP:
7230 032624 000137 005310      JMP      @#LOOP          ;;RETURN
7231 032630 000000          $TBIT: .WORD 0          ;;'T' BIT STATE INDICATOR
7232 032632 377 377 000 $ENULL: .BYTE -1,-1,0  ;;NULL CHARACTER STRING
7233          .EVEN
7234          ;;*****
7235          .SBTTL SPURIOUS ERROR HANDLER
7236          ;; THIS ROUTINE IS ENTERED BY AN UNEXPECTED TRAP TO 4 OR 114.
7237          ;; IF SWITCH 13 IS OFF, AN ERROR MESSAGE, THE ERROR REGISTER,
7238          ;; THE ERROR PC, AND THE TEST NUMBER ARE TYPED OUT.
7239          ;; IF SWITCH 13 IS ON, ONLY THE ERROR MESSAGE WILL BE TYPED.
7240          ;;*****
7241 032636 011667 146254      CPUSPUR:MOV      (SP), $ERRPC          ;SAVE THE ERROR PC
7242 032642 012706 001100      MOV      #STACK, SP          ;RESTORE THE SP
7243 032646 104400 032654      TYPE     ,65$              ;;TYPE ASCIZ STRING
7244 032652 000414          BR      64$                ;;GET OVER THE ASCIZ
7245          ;;65$: .ASCIZ /UNEXPECTED TRAP TO 4/<15><12>
7246 032704          64$:
7247 032704 032737 020000 177570      BIT      #BIT13, @#SWR      ;IS INHIBIT ERROR TYPEOUT ON?
7248 032712 001045          BNE      1$                ;BRANCH IF YES
7249 032714 104400 032722      TYPE     ,67$              ;;TYPE ASCIZ STRING
7250 032720 000417          BR      66$                ;;GET OVER THE ASCIZ
7251          ;;67$: .ASCIZ /ERRORPC TSTNUM CPUERR REG/<15><12>
7252 032760          66$:
7253 032760 013767 177766 146174      MOV      @#CPUERR, $TMPO     ;GET CPU ERROR REG
7254 032766 116767 146110 146214      MOVB     $TSTNM, $$TSTNM    ;GET TEST NUMBER
7255 032774 016746 146116      MOV      $ERRPC, -(SP)      ;;SAVE $ERRPC FOR TYPEOUT
7256          ;;TYPE ERROR PC
7257 033000 104402          TYPCC   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
7258 033002 104400 034622      TYPE     ,TWOSP
7259 033006 016746 146176      MOV      $$TSTNM, -(SP)     ;;SAVE $$TSTNM FOR TYPEOUT
7260          ;;TYPE TEST NUMBER
7261 033012 104402          TYPCC   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
7262 033014 104400 034622      TYPE     ,TWOSP
7263 033020 016746 146136      MOV      $TMPO, -(SP)      ;;SAVE $TMPO FOR TYPEOUT
7264          ;;TYPE ERROR REGISTER
7265 033024 104402          TYPCC   ;GO TYPE--OCTAL ASCII(ALL DIGITS)
7266 033026 005037 177766 146134      1$: CLR      @#CPUERR          ;CLEAR CPU ERROR REG
7267 033032 016767 146232          MOV      NEXTTST, $ESCAPE    ;SETUP ESCAPE ADDRESS
7268 033040 104264          ERROR   264                ;MAKE THE ERROR CALL TO SYSMAC
7269 033042 011667 146050      CACHSPU:MOV     (SP), $ERRPC  ;SAVE ERROR PC
7270 033046 012706 001100      MOV      #STACK, SP        ;RESTORE STACK
7271 033052 104400 033060      TYPE     ,65$              ;;TYPE ASCIZ STRING
7272 033056 000415          BR      64$                ;;GET OVER THE ASCIZ
7273          ;;65$: .ASCIZ /UNEXPECTED TRAP TO 114/<15><12>
7274 033112          64$:
7275 033112 032737 020000 177570      BIT      #BIT13, @#SWR      ;IS SWITCH 13 ON?
7276 033120 001045          BNE      1$                ;BRANCH IF YES
7277 033122 104400 033130      TYPE     ,67$              ;;TYPE ASCIZ STRING
7278 033126 000417          BR      66$                ;;GET OVER THE ASCIZ
7279          ;;67$: .ASCIZ /ERRORPC TSTNUM MEMERR REG/<15><12>
7280 033166          66$:
7281 033166 013767 177744 145766      MOV      @#MEMERR, $TMPO     ;SAVE MEMORY ERROR REG
7282 033174 116767 145702 146006      MOVB     $TSTNM, $$TSTNM    ;SAVE TEST NUMBER
7283 033202 016746 145710      MOV      $ERRPC, -(SP)      ;;SAVE $ERRPC FOR TYPEOUT
7284          ;;TYPE ERROR PC
    
```



```

7285 033206 104402          TYP0C          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7286 033210 104400 034622  TYPE          TWOSP
7287 033214 016746 145770  MOV          $$TSTNM,-(SP) ;;SAVE $$TSTNM FOR TYPEOUT
7288                                     ;;TYPE TEST NUMBER
7289 033220 104402          TYP0C          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7290 033222 104400 034622  TYPE          TWOSP
7291 033226 016746 145730  MOV          $TMPO,-(SP)  ;;SAVE $TMPO FOR TYPEOUT
7292                                     ;;TYPE MEM ERROR REG
7293 033232 104402          TYP0C          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7294 033234 013737 177744 177744 1$: MOV          @#MEMERR,@#MEMERR ;CLEAR MEMERR REG.
7295 033242 016767 146022 145724  MOV          NEXTTST,$ESCAPE ;SETUP ESCAPE ADDRESS
7296 033250 104264          ERROR          264
7297                                     ;;*****
7298
7299 .SBTTL SCOPE HANDLER ROUTINE
7300
7301 ;;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
7302 ;;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<7:0>)
7303 ;;*AND LOAD THE ERROR FLAG ($ERFLG) INTO DISPLAY<15:08>
7304 ;;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7305 ;;*SW14=1 LOOP ON TEST
7306 ;;*SW11=1 INHIBIT ITERATIONS
7307 ;;*SW09=1 LOOP ON ERROR
7308 ;;*SW08=1 LOOP ON TEST IN SWR<7:0>
7309 ;;*CALL
7310 ;;* SCOPE          ;;:SCOPE=10T
7311
7312 033252          $SCOPE:
7313 033252 006137 177570          ROL          @#SWR          ;;LOOP ON PRESENT TEST?
7314 033256 100511          BMI          $OVER          ;;YES IF SW14=1
7315                                     ;#####START OF CODE FOR THE XOR TESTER#####
7316 033260 000416          $XTSTR: BR          6$          ;;IF RUNNING ON THE "XOR" TESTER CHANGE
7317                                     ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
7318 033262 013746 000004          MOV          @#ERRVEC,-(SP) ;;SAVE THE CONTENTS OF THE ERROR VECTOR
7319 033266 012737 033306 000004  MOV          #5$,@#ERRVEC ;;SET FOR TIMEOUT
7320 033274 005737 177060          TST          @#177060      ;;TIME OUT ON XOR?
7321 033300 012637 000004          MOV          (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
7322 033304 000463          BR          $$VLAD          ;;GO TO THE NEXT TEST
7323 033306 022626          5$: CMP          (SP)+,(SP)+ ;;CLEAR THE STACK AFTER A TIME OUT
7324 033310 012637 000004          MOV          (SP)+,@#ERRVEC ;;RESTORE THE ERROR VECTOR
7325 033314 000423          BR          7$          ;;LOOP ON THE PRESENT TEST
7326 033316          6$:;#####END OF CODE FOR THE XOR TESTER#####
7327 033316 032737 000400 177570  BIT          #BIT08,@#SWR ;;LOOP ON SPEC. TEST?
7328 033324 001404          BEQ          2$          ;;BR IF NO
7329 033326 123767 177570 145546  CMPB         @#SWR,$TSTNM ;;ON THE RIGHT TEST? SWR<7:0>
7330 033334 001462          BEQ          $OVER          ;;BR IF YES
7331 033336 105767 145541          2$: TSTB         $ERFLG      ;;HAS AN ERROR OCCURRED?
7332 033342 001421          BEQ          3$          ;;BR IF NO
7333 033344 126767 145545 145531  CMPB         $ERMAX,$ERFLG ;;MAX. ERRORS FOR THIS TEST OCCURRED?
7334 033352 101015          BHI          3$          ;;BR IF NO
7335 033354 032737 001000 177570  BIT          #BIT09,@#SWR ;;LOOP ON ERROR?
7336 033362 001404          BEQ          4$          ;;BR IF NO
7337 033364 016767 145520 145514  7$: MOV          $LPERR,$LPADR ;;SET LOOP ADDRESS TO LAST SCOPE
7338 033372 000443          BR          $OVER
7339 033374 105067 145503          4$: CLRB         $ERFLG      ;;ZERO THE ERROR FLAG
7340 033400 005067 145566          CLR          $TIMES      ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
    
```

```

7341 033404 000415          BR      1$          ;;ESCAPE TO THE NEXT TEST
7342 033406 032737 004000 177570 3$:  BIT      #BIT11,@#SWR  ;;INHIBIT ITERATIONS?
7343 033414 001011          BNE     1$          ;;BR IF YES
7344 033416 005767 145456    TST     $PASS      ;;IF FIRST PASS OF PROGRAM
7345 033422 001406          BEQ     1$          ;;          INHIBIT ITERATIONS
7346 033424 005267 145454    INC     $ICNT      ;;INCREMENT ITERATION COUNT
7347 033430 026767 145536 145446  CMP     $TIMES,$ICNT ;;CHECK THE NUMBER OF ITERATIONS MADE
7348 033436 002021          BGE     $OVER      ;;BR IF MORE ITERATION REQUIRED
7349 033440 012767 000001 145436 1$:  MOV     #1,$ICNT   ;;REINITIALIZE THE ITERATION COUNTER
7350 033446 016767 000044 145516  MOV     $MXCNT,$TIMES ;;SET NUMBER OF ITERATIONS TO DO
7351 033454 105267 145422  $SVLAD: INCB    $STNM  ;;COUNT TEST NUMBERS
7352 033460 011667 145422    MOV     (SP),$LPADR ;;SAVE SCOPE LOOP ADDRESS
7353 033464 011667 145420    MOV     (SP),$LPERR ;;SAVE ERROR LOOP ADDRESS
7354 033470 005067 145500    CLR     $ESCAPE    ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
7355 033474 112767 000001 145413  MOVB    #1,$ERMAX   ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
7356 033502 016737 145374 177570 $OVER: MOV     $STNM,@#DISPLAY ;;DISPLAY TEST NUMBER
7357 033510 016716 145372    MOV     $LPADR,(SP) ;;FUDGE RETURN ADDRESS
7358 033514 000002          RTI          ;;FIXES PS
7359 033516 003720          $MXCNT: 2000.    ;;MAX. NUMBER OF ITERATIONS
7360          ;;*****
7361          .SBTTL  ERROR HANDLER ROUTINE
7362          ;*THIS ROUTINE WILL INCREMENT THE ERROR FLAG AND THE ERROR COUNT,
7363          ;*SAVE THE ERROR ITEM NUMBER AND THE ADDRESS OF THE ERROR CALL
7364          ;*AND GO TO ETYPDM ON ERROR
7365          ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
7366          ;*SW15=1      HALT ON ERROR
7367          ;*          HALT CAN OCCUR BEFORE AND AFTER THE ERROR TYPEOUT
7368          ;*SW13=1     INHIBIT ERROR TYPEOUTS
7369          ;*SW10=1    BELL ON ERROR
7370          ;*SW09=1    LOOP ON ERROR
7371          ;*CALL
7372          ;*          ERROR  N          ;;ERROR=EMT AND N=ERROR ITEM NUMBER
7373          $ERROR:
7374          7$:  MOVB    $STNM,$$STNM  ;GET TEST NUMBER FOR TYPE OUT
7375          INCB    $ERFLG          ;;SET THE ERROR FLAG
7376          BEQ     7$             ;;DON'T LET THE FLAG GO TO ZERO
7377          MOV     $STNM,@#DISPLAY ;;DISPLAY TEST NUMBER AND ERROR FLAG
7378          TST     @#SWR          ;;HALT ON ERROR = 1?
7379          BPL     8$             ;;BRANCH IF NO
7380          HALT                    ;;YES--HALT
7381          8$:  BIT      #BIT10,@#SWR  ;;BELL ON ERROR?
7382          BEQ     1$             ;;NO - SKIP
7383          TYPE    , $BELL        ;;RING BELL
7384          1$:  INC     $ERTTL      ;;COUNT THE NUMBER OF ERRORS
7385          MOV     (SP),$ERRPC    ;;GET ADDRESS OF ERROR INSTRUCTION
7386          SUB     #2,$ERRPC
7387          MOVB    @ $ERRPC,$ITEMB ;;STRIP AND SAVE THE ERROR ITEM CODE
7388          BIT     #BIT13,@#SWR  ;;SKIP TYPEOUT IF SET
7389          BNE     2$             ;;SKIP TYPEOUTS
7390          JSR     PC,ETYPDM      ;;GO TO USER ERROR ROUTINE
7391          TYPE    , $CRLF
7392          2$:  TST     @#SWR          ;;HALT ON ERROR
7393          BPL     9$             ;;SKIP IF CONTINUE
    
```

```

7397 033640 000000          HALT                ;;HALT ON ERROR!
7398 033642 022767 032554 144172 9$:  CMP          #SENDAD,42      ;;ACT-11?
7399 033650 001001          BNE          3$              ;;BRANCH IF NO
7400 033652 000000          HALT                ;;YES
7401 033654 032737 001000 177570 3$:  BIT          #BIT09,@#SWR    ;;LOOP ON ERROR SWITCH SET?
7402 033662 001402          BEQ          4$              ;;BR IF NO
7403 033664 016716 145220          MOV          $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
7404 033670 005767 145300          4$:  TST          $ESCAPE      ;;CHECK FOR AN ESCAPE ADDRESS
7405 033674 001402          BEQ          5$              ;;BR IF NONE
7406 033676 016716 145272          MOV          $ESCAPE,(SP)   ;;FUDGE RETURN ADDRESS FOR ESCAPE
7407 033702          5$:
7408 033702 000002          RTI                    ;;RETURN
7409
7410          ;;*****
7411          .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE
7412          ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
7413          ;*ERROR IS TO BE REPORTED. IT THEN OBTAINS, FROM THE "ERROR TABLE" ($ERRTB),
7414          ;*AND REPORTS THE APPROPRIATE INFORMATION CONCERNING THE ERROR.
7415          ;;*****
7415 033704 026727 145300 000042  ETYFDM:  CMP          $$TSTNM,#42      ;IS THIS TEST 42?
7416 033712 001464          BEQ          TYP5LE         ;BRANCH IF YES
7417 033714 026727 145270 000070  CMP          $$TSTNM,#70      ;IS THIS TEST 70?
7418 033722 001460          BEQ          TYP5LE         ;BRANCH IF YES
7419 033724 104400 001203          TYPE          , $CRLF       ;; "CARRIAGE RETURN" & "LINE FEED"
7420 033730 005000          CLR          RO            ;; PICKUP THE ITEM INDEX
7421 033732 153700 001114          BISB         @#$ITEMB,RO
7422 033736 126727 145152 000377  CMPB         $ITEMB,#377     ;IS MESSAGE NUMBER OVER 377
7423 033744 001006          BNE          1$              ;BRANCH IF NO
7424 033746 016600 000002          MOV          2(SP),RO        ;GET RETURN PC FROM STACK
7425 033752 011000          MOV          (RO),RO         ;GET MESSAGE NUMBER
7426 033754 062766 000002 000002  ADD          #2,2(SP)         ;CORRECT PC
7427 033762 005300          1$:  DEC          RO            ;;ADJUST THE INDEX SO THAT IT WILL
7428 033764 010001          MOV          RO,R1
7429 033766 070127 000006          MUL          #6,R1           ;;WORK FOR THE ERROR TABLE
7430 033772 010100          MOV          R1,RO
7431 033774 062700 001274          ADD          #$ERRTB,RO      ;;FORM TABLE POINTER
7432 034000 012067 000004          MOV          (RO)+,2$        ;;PICKUP "ERROR MESSAGE" POINTER
7433 034004 001404          BEQ          3$              ;;SKIP TYPEOUT IF NO POINTER
7434 034006 104400          TYPE          ;;TYPE THE "ERROR MESSAGE"
7435 034010 000000          2$:  .WORD          0           ;; "ERROR MESSAGE" POINTER GOES HERE
7436 034012 104400 001203          TYPE          , $CRLF       ;; "CARRIAGE RETURN" & "LINE FEED"
7437 034016 012067 000004          3$:  MOV          (RO)+,4$        ;;PICKUP "DATA HEADER" POINTER
7438 034022 001404          BEQ          5$              ;;SKIP TYPEOUT IF 0
7439 034024 104400          TYPE          ;;TYPE THE "DATA HEADER"
7440 034026 000000          4$:  .WORD          0           ;; "DATA HEADER" POINTER GOES HERE
7441 034030 104400 001203          TYPE          , $CRLF       ;; "CARRIAGE RETURN" & "LINE FEED"
7442 034034 011000          5$:  MOV          (RO),RO        ;;PICKUP "DATA TABLE" POINTER
7443 034036 001003          BNE          7$              ;;GO TYPE THE DATA
7444 034040 104400 001203          6$:  TYPE          , $CRLF       ;; "CARRIAGE RETURN" & "LINE FEED"
7445 034044 000207          RTS          PC             ;;RETURN
7446 034046          7$:
7447 034046 013046          MOV          @(RO)+,-(SP)    ;;SAVE @(RO)+ FOR TYPEOUT
7448 034050 104402          TYPOC         ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7449 034052 005710          TST          (RO)           ;;IS THERE ANOTHER NUMBER?
7450 034054 001771          BEQ          6$              ;;BR IF NO
7451 034056 104400 034622          TYPE          ,TWOSP        ;;TYPE TWO(2) SPACES
7452 034062 000771          BR          7$              ;;LOOP
    
```

```

7453
7454
7455
7456
7457
7458 034064 104400
7459 034066 055244
7460 034070 104400
7461 034072 055433
7462 034074 016727 145016
7463 034100 000000
7464 034102 016746 177772
7465
7466 034106 104402
7467 034110 104400
7468 034112 034622
7469 034114 016727 145070
7470 034120 000000
7471 034122 016746 177772
7472
7473 034126 104402
7474 034130 104400
7475 034132 001203
7476
7477 034134 104400
7478 034136 055460
7479 034140 005067 000020
7480
7481 034144 005001
7482 034146 005067 000134
7483 034152 005067 000174
7484 034156 012700 120000
7485 034162 122710
7486 034164 000001
7487 034166 001417
7488 034170 062700 000004
7489 034174 020067 144754
7490 034200 001401
7491 034202 000767
7492 034204 000241
7493 034206 062767 000002 177750
7494 034214 026727 177744 000014
7495 034222 001350
7496 034224 000467
7497
7498 034226 005701
7499 034230 001014
7500 034232 016701 177726
7501 034236 062701 056042
7502 034242 011167 000006
7503 034246 104400 034604
7504 034252 104400
7505 034254 000000
7506 034256 104400
7507 034260 001203
7508

;*****
;SBTTL STACK LIMIT TEST TYPE OUT ROUTINE
;* THIS ROUTINE TYPES THE ADDRESS AND STACK LIMIT REGISTER
;*VALUES THAT FAILED IN TEST 153 OR 201 IN OCTAL AND BINARY.
;*****
TYPSE: TYPE ;TYPE
EM263 ;ERROR MESSAGE
TYPE ;TYPE
DH263 ;DATA HEADER
MOV $ERRPC,(PC)+ ;GET ERROR PC
1$: .WORD ;ERROR PC
MOV 1$,-(SP) ;;SAVE 1$ FOR TYPEOUT
;TYPE IT
;GO TYPE--OCTAL ASCII(ALL DIGITS)
;TYPE TWO SPACES
TYPYC
TYPE
TWOSP
MOV $$TSTNM,(PC)+ ;GET TEST NUMBER
2$: .WORD 0 ;TEST NUMBER
MOV 2$,-(SP) ;;SAVE 2$ FOR TYPEOUT
;TYPE IT
;GO TYPE--OCTAL ASCII(ALL DIGITS)
;TYPE CR LF
TYPE
DH263A ;TYPE DATA HEADER
CLR $TYPEE ;INITIALIZE $TYPEE
STYPE: CLR R1 ;ENSURE R1 CLEAR
CLR SLDATA
CLR SPDATA
MOV #120000,R0 ;GET ERROR DATA BUFFER POINTER
BUFFDP: CMPB (PC)+,(R0) ;IS THIS CORRECT TYPE?
$TYPEE: .WORD 1
BEQ TYPEHE ;BRANCH IF YES, GO TYPE IT
ADD #4,R0 ;STEP R0 TO NEXT ENTRY
NEXTEN: CMP R0,$REGO ;LAST ENTRY?
BEQ 2$ ;BRANCH IF YES
BR BUFFDP ;GO CHECK NEXT ENTRY
2$: CLC ;ENSURE C CLEAR
ADD #2,$TYPEE ;SELECT NEXT ERROR TYPE
CMP $TYPEE,#14 ;IS TYPE ROUTINE DONE?
BNE STYPE ;BRANCH IF NO
BR $$DONE ;RETURN
;OUTPUT THE TYPE HEADER IF FIRST ERROR OF THIS TYPE
TYPEHE: TST R1 ;FIRST ERROR OF THIS TYPE?
BNE 4$ ;BRANCH IF NO
MOV $TYPEE,R1 ;GET TYPE NUMBER
ADD #INDEX,R1 ;GET ADDRESS OF TYPE HEADER ADDRESS
MOV (R1),3$ ;GET ADDRESS OF HEADER
TYPE ;TYPE SPACES
TYPE ;GO TYPE THE HEADER
3$: .WORD ;ADDRESS OF HEADER
TYPE ;TYPE A CRLF
$CRLF
;OUTPUT THE DATA
    
```

```

7509 034262 126067 000001 000017 4$:  CMPB 1(R0),SLDATA+1 ;IS SL DATA SAME AS LAST SL DATA?
7510 034270 001005 BNE 8$ ;BRANCH IF NO
7511 034272 104400 TYPE ;TYPE 38 SPACES
7512 034274 034562 SP38
7513 034276 062700 000002 ADD #2,R0 ;STEP R0 TO SP DATA
7514 034302 000422 BR SPDATA-2 ;GO TO SP DATA OUTPUT
7515 034304 012027 8$: MOV (R0)+,(PC)+ ;GET SL DATA
7516 034306 000000 SLDATA: .WORD ;
7517 034310 042767 000377 177770 BIC #377,SLDATA ;MASK HIGH BYTE
7518 034316 016746 177764 MOV SLDATA,-(SP) ;SAVE SLDATA FOR TYPEOUT
7519 ;TYPE IT
7520 034322 104402 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
7521 034324 016746 177756 MOV SLDATA,-(SP) ;PUT DATA ON STACK
7522 034330 004767 000150 JSR PC,TYPEBN ;GO TYPE BINARY
7523 034334 021067 000012 CMP (R0),SPDATA ;IS SP DATA SAME AS LAST SP DATA?
7524 034340 001003 BNE 9$ ;BRANCH IF NO
7525 034342 062700 000002 ADD #2,R0 ;STEP R0 TO NEXT SL DATA
7526 034346 000413 BR TERM ;GO TERMINATE LINE
7527 034350 012027 9$: MOV (R0)+,(PC)+ ;GET SP ERROR DATA
7528 034352 000000 SPDATA: .WORD ;
7529 034354 104400 034620 TYPE ,FOURSP ;TYPE FOUR SPACES
7530 034360 016746 177766 MOV SPDATA,-(SP) ;SAVE SPDATA FOR TYPEOUT
7531 ;GO TYPE IT IN OCTAL
7532 034364 104402 TYPOC ;GO TYPE--OCTAL ASCII(ALL DIGITS)
7533 034366 016746 177760 MOV SPDATA,-(SP) ;PUT DATA ON STACK
7534 034372 004767 000106 JSR PC,TYPEBN ;GO TYPE BINARY
7535 034376 104400 TERM: TYPE ;TYPE A CR LF
7536 034400 001203 $CRLF
7537 034402 000674 BR NEXTEN ;GO CHECK NEXT ENTRY
7538 034404 026727 144544 157774 $$DONE: CMP $REGO,#157774 ;DID BUFFER OVERFLOW?
7539 034412 001033 BNE 1$ ;BRANCH IF NO
7540 034414 104400 034422 TYPE ,65$ ;TYPE ASCII STRING
7541 034420 000430 BR 64$ ;GET OVER THE ASCII
7542 ;:65$: .ASCIIZ <15><12>/PROBABLY MORE ERRORS BUT BUFFER OVERFLOWED/<15><12>
7543 034502 64$:
7544 034502 000207 1$: RTS PC ;RETURN TO $ERROR
7545
7546 034504 016602 000002 TYPEBN: MOV 2(SP),R2 ;GET NUMBER TO BE TYPED
7547 034510 104400 TYPE ;TYPE FOUR SPACES
7548 034512 034620 FOURSP
7549 034514 012703 000010 MOV #10,R3 ;SETUP SOB COUNT
7550 034520 112767 000060 000030 3$: MOVB #60,BINARY ;PUT ASCII 0 IN LOCATION BINARY
7551 034526 006102 ROL R2 ;GET BIT TO BE TYPED
7552 034530 103005 BCC 1$ ;BRANCH IF IT IS 0
7553 034532 005567 000020 ADC BINARY ;MAKE IT ASCII
7554
7555 034536 104400 034556 TYPE ,BINARY ;GO TYPE IT
7556 034542 000402 BR 2$ ;GO TO END OF LOOP
7557 034544 104400 034621 1$: TYPE ,THRESP ;BIT WAS 0 TYPE 3 SPACES
7558 034550 077315 2$: SOB R3,3$
7559 034552 012616 MOV (SP)+,(SP) ;READ JUST STACK
7560 034554 000207 RTS PC ;RETURN
7561 034556 000 040 040 BINARY: .BYTE 0,40,40,0 ;STORAGE FOR ASCII CHARACTERS & TERMINATOR
7562 034561 000
7563 034562 020040 020040 020040 SP38: .ASCII / /
7564 034570 020040 020040 020040
    
```

```

7565 034576 020040 020040 020040
7566 034604 020040 020040 020040 SP16: .ASCII / /
7567 034612 020040 020040 040
7568 034617 040 FIVESP: .ASCII / /
7569 034620 040 FOURSP: .ASCII / /
7570 034621 040 THRESP: .ASCII / /
7571 034622 020040 000 TWOSP: .ASCIZ / /
7572 034626 .EVEN
7573 ::*****
7574 .SBTTL MONITOR RESTORE ROUTINE
7575 ;*THIS ROUTINE IS ENTERED BY TYPING A CHARACTER ON THE KEYBOARD
7576 ;*IF THE CHARACTER IS NOT A CTRL C EXECUTION IS RETURNED TO THE
7577 ;*TEST FOLLOWING THE ONE THAT WAS INTERRUPTED.
7578 ;*IF IT IS A CTRL C THE MONITOR IS RESTORED AND THE
7579 ;*PROCESSOR HALTS.
7580 034626 005003 QUIT: CLR R3 ;NOT ENTERED THROUGH XXDP CHAIN
7581 034630 017700 144304 MOV @STKB,R0 ;GET CHARACTER
7582 034634 042700 000200 BIC #BIT7,R0 ;GET RID OF PARITY BIT
7583 034640 022700 000003 CMP #3,R0 ;WAS IT A CONTROL C?
7584 034644 001041 BNE DUMMY ;BRANCH IF NO
7585 034646 000005 RESET ;CLEAR THE WORLD
7586 034650 104400 034656 TYPE ,65$ ;;TYPE ASCIZ STRING
7587 034654 000403 BR 64$ ;;GET OVER THE ASCIZ
7588 ;;65$: .ASCIZ /^C/<15><12>
7589 64$:
7590 034664 012700 002734 CHAINQ: MOV #^D1500,R0 ;SETUP SOB COUNT
7591 034670 012701 073262 MOV #SUBTAB+2,R1 ;GET END ADDRESS OF PROGRAM
7592 034674 012702 160000 MOV #160000,R2 ;GET TOP ADDRESS OF MONITOR
7593 034700 012142 1$: MOV (R1)+,-(R2) ;RESTORE THE MONITOR
7594 034702 077002 SOB R0,1$
7595 034704 005303 DEC R3
7596 034706 001001 BNE 2$
7597 034710 000207 RTS PC
7598 034712
7599 034712 104400 03,720 2$: TYPE ,65$ ;;TYPE ASCIZ STRING
7600 034716 000413 BR 64$ ;;GET OVER THE ASCIZ
7601 ;;65$: .ASCIZ /MONITOR RESTORED/<15><12><15><12>
7602 64$:
7603 034746 000000 HALT
7604 034750 005077 144164 DUMMY: CLR @STKB ;CLEAR KEYBOARD BUFFER
7605 034754 152777 000100 144154 BISB #BIT6,@STKS ;RESET INTERRUPT ENABLE BIT
7606 034762 104400 034770 TYPE ,65$ ;;TYPE ASCIZ STRING
7607 034766 000417 BR 64$ ;;GET OVER THE ASCIZ
7608 ;;65$: .ASCIZ <15><12>/TYPE A CTRL C TO QUIT!!!!/<15><12>
7609 64$:
7610 035026 000177 144236 JMP @NEXTTST ;RETURN
7611 ::*****
7612 .SBTTL CHECK TEST SEQUENCE ROUTINE
7613 ;*THIS ROUTINE IS CALLED IN THE SCOPE ROUTINE. IT VERIFYS
7614 ;*THAT A TEST HAS NOT BEEN SKIPPED.
7615 035032 016605 000002 SEQENC: MOV 2(SP),R5 ;GET ADDRESS OF SCOPE+2
7616 035036 162705 000002 SUB #2,R5 ;GET ADDRESS OF SCOPE CALL
7617 035042 005004 CLR R4 ;ENSURE R4 CLEAR
7618 035044 116704 144032 MOVB $TSTNM,R4 ;GET TEST NUMBER
7619 035050 006104 ROL R4 ;ADJUST
7620 035052 026405 073106 CMP ADRTAB(R4),R5 ;HAS TEST BEEN SKIPPED?
    
```

```

7621 035056 001523          BEQ     1$          ;BRANCH IF NO
7622 035060 006004          ROR     R4
7623 035062 005304          DEC     R4
7624 035064 010467 144064    MOV     R4,$REGO   ;SAVE PREVIOUS TST NUM FOR TYP0UT
7625
7626          ;FIND OUT WHICH TEST THIS IS.
7627 035070 012767 000002 144064    MOV     #2,$TMP0   ;SET BUFFER HEADER POINTER
7628 035076 012767 000176 144060    MOV     #176,$TMP1 ;SET BUFFER END POINTER
7629 035104 012703 000100          MOV     #100,R3    ;SET R3 AT MIDDLE OF BUFFER
7630 035110 026305 073106    4$:    CMP     ADRTAB(R3),R5 ;IS IT THIS TEST?
7631 035114 001426          BEQ     2$          ;BRANCH IF YES
7632 035116 101014          BHI     3$          ;MOVE UP TABLE
7633          ;CORRECT TEST IS FURTHER DOWN TABLE
7634 035120 010367 144036    MOV     R3,$TMP0   ;UPDATE HEADER POINTER
7635 035124 016702 144034    MOV     $TMP1,R2   ;GET END POINTER
7636 035130 166702 144026    SUB     $TMP0,R2   ;FIND RANGE OF REMAINING TABLE
7637 035134 000241          CLC
7638 035136 006002          ROR     R2          ;FIND MIDPOINT OF RANGE
7639 035140 060203          ADD     R2,R3      ;MAKE R3 MID POINT OF REMAINING TABLE
7640 035142 042703 000001    5$:    BIC     #BIT0,R3   ;ENSURE EVEN ADDRESS
7641 035146 000760          BR      4$          ;GO CHECK AGAIN
7642
7643          ;CORRECT TEST IS FURTHER UP THE TABLE
7644 035150 010367 144010    3$:    MOV     R3,$TMP1   ;UPDATE END POINTER
7645 035154 010302          MOV     R3,R2      ;GET END POINTER
7646 035156 166702 144000    SUB     $TMP0,R2   ;FIND RANGE OF REMAINING TABLE
7647 035162 000241          CLC
7648 035164 006002          ROR     R2          ;FIND MID POINT OF RANGE
7649 035166 160203          SUB     R2,R3      ;MAKE R3 MIDPOINT OF REMAINING TABLE
7650 035170 000764          BR      5$          ;MAKE EVEN ADDRESS AND CHECK AGAIN
7651
7652          ;FOUND THE CORRECT TEST
7653 035172 000241    2$:    CLC
7654 035174 006003          ROR     R3          ;GET TEST NUMBER
7655 035176 110367 143700    MOV     R3,$STNM   ;UPDATE TEST NUMBER
7656 035202 010367 143750    MOV     R3,$REG1   ;SAVE FOR TYPEOUT
7657 035206 104400 035214    TYPE   ,65$        ;:TYPE ASCIZ STRING
7658 035212 000404          BR      64$        ;:GET OVER THE ASCIZ
7659          ;;65$: .ASCIZ /TEST /
7660          64$:
7661 035224 016746 143724    MOV     $REG0,-(SP) ;;SAVE $REG0 FOR TYPEOUT
7662 035230 104402          TYPOC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7663 035232 104400 035240    TYPE   ,67$        ;:TYPE ASCIZ STRING
7664 035236 000426          BR      66$        ;:GET OVER THE ASCIZ
7665          ;;67$: .ASCIZ / FAILED, CAUSING ECECUTION TO GO TO TEST /
7666          66$:
7667 035314 016746 143636    MOV     $REG1,-(SP) ;;SAVE $REG1 FOR TYPEOUT
7668 035320 104402          TYPOC          ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
7669 035322 104400 001203    TYPE   ,$CRLF      ;
7670 035326 000207    1$:    RTS     PC          ;RETURN
7671          ;;*****
7672
7673          .SBTTL TYPE ROUTINE
7674
7675          ;*ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
7676          ;*THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
    
```

```

7677 ;*NOTE1:          $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
7678 ;*NOTE2:          $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
7679 ;*NOTE3:          $FILLC CONTAINS THE CHARACTER TO FILL AFTER.
7680 ;*
7681 ;*CALL:
7682 ;*1) USING A TRAP INSTRUCTION
7683 ;*      TYPE      ,MESADR          ;;MESADR IS FIRST ADDRESS OF AN ASCIZ STRING
7684 ;*OR
7685 ;*      TYPE
7686 ;*      MESADR
7687 ;*
7688 ;*2) USING A JSR INSTRUCTION
7689 ;*      MOV      PS,-(SP)          ;;PUSH PROCESSOR STATUS WOPD ON THE STACK
7690 ;*      JSR      PC,$TYPE          ;;CALL TYPE ROUTINE
7691 ;*      MESADDR          ;;FIRST ADRESS OF MESSAGE
7692
7693 035330 105767 143615 $TYPE: TSTB $TPFLG          ;;IS THERE A TERMINAL?
7694 035334 100002      BPL 1$          ;;BR IF YES
7695 035336 000000      HALT          ;;HALT HERE IF NO TERMINAL
7696 035340 000407      BR 3$          ;;LEAVE
7697 035342 010046 1$: MOV RO,-(SP)          ;;SAVE RO
7698 035344 017600 000002 MOV @2(SP),RO          ;;GET ADDRESS OF ASCIZ STRING
7699 035350 112046 2$: MOVB (RO)+,-(SP)          ;;PUSH CHARACTER TO BE TYPED ONTO STACK
7700 035352 001005      BNE 4$          ;;BR IF IT ISN'T THE TERMINATOR
7701 035354 005726      TST (SP)+          ;;IF TERMINATOR POP IT OFF THE STACK
7702 035356 012600      MOV (SP)+,RO          ;;RESTORE RO
7703 035360 062716 000002 3$: ADD #2,(SP)          ;;ADJUST RETURN PC
7704 035364 000002      RTI          ;;RETURN
7705 035366 122716 000011 4$: CMPB #HT,(SP)          ;;BRANCH IF <HT>
7706 035372 001426      BEQ 8$
7707 035374 122716 000200      CMPB #CRLF,(SP)          ;;BRANCH IF NOT
7708 035400 001004      BNE 5$
7709 035402 005726      TST (SP)+          ;;POP <CR><LF> EQUIV
7710 035404 104400 001203      TYPE , $CRLF
7711 035410 000757      BR 2$          ;;GET NEXT CHARACTER
7712 035412 004767 000056 5$: JSR PC,$TYPEPC          ;;GO TYPE THIS CHARACTER
7713 035416 126726 143526 6$: CMPB $FILLC,(SP)+          ;;IS IT TIME FOR FILLER CHARS.?
7714 035422 001352      BNE 2$          ;;IF NO GO GET NEXT CHAR.
7715 035424 016746 143516      MOV $NULL,-(SP)          ;;GET # OF FILLER CHARS. NEEDED
7716 ;*AND THE NULL CHAR.
7717 035430 105366 000001 7$: DECB 1(SP)          ;;DOES A NULL NEED TO BE TYPED?
7718 035434 002770      BLT 6$          ;;BR IF NO--GO POP THE NULL OFF OF STACK
7719 035436 004767 000032      JSR PC,$TYPEPC          ;;GO TYPE A NULL
7720 035442 105367 000072      DECB $CHARCNT          ;;DON'T COUNT THE NULL AS A CHARACTER
7721 035446 000770      BR 7$          ;;LOOP
7722
7723 ;;HORIZONTAL TAB PROCESSOR
7724
7725 035450 112716 000040 8$: MOVB #' ,(SP)          ;;REPLACE TAB WITH SPACE
7726 035454 004767 000014 9$: JSR PC,$TYPEPC          ;;TYPE A SPACE
7727 035460 132767 000007 000052 BITB #7,$CHARCNT          ;;BRANCH IF NOT AT
7728 035466 001372      BNE 9$          ;;TAB STOP
7729 035470 005726      TST (SP)+          ;;POP SPACE OFF STACK
7730 035472 000726      BR 2$          ;;GET NEXT CHARACTER
7731 035474 105777 143442 $TYPEPC: TSTB @ $TPS          ;;WAIT UNTIL PRINTER IS READY
7732 035500 100375      BPL $TYPEPC
    
```



```

7733 035502 116677 000002 143434      MOVB 2(SP),@STPB      ;;LOAD CHAR TO BE TYPED INTO DATA REG.
7734 035510 122766 000015 000002      CMPB #CR,2(SP)      ;;BRANCH IF
7735 035516 001003          BNE 1$              ;;NOT <CR>
7736 035520 105067 000014          CLRB $CHARCNT      ;;
7737 035524 000406          BR $TYPEX          ;;EXIT
7738 035526 122766 000012 000002 1$:  CMPB #LF,2(SP)      ;;BRANCH IF
7739 035534 001402          BEQ $TYPEX         ;;<LF>
7740 035536 105227          INCB (PC)+         ;;INC SPACE
7741 035540 000000          $CHARCNT:.WORD 0  ;;COUNT
7742 035542 000207          $TYPEX: RTS      PC
7743
7744
7745
7746
7747
7748
7749
7750
7751
7752
7753
7754
7755
7756
7757
7758
7759
7760
7761
7762
7763
7764
7765
7766
7767
7768
7769

```

```

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

```

```

7770 035544 017646 000000          $TYPOS: MOV @ (SP),-(SP)  ;;PICKUP THE MODE
7771 035550 116667 000001 000211  MOVB 1(SP),$OFILL  ;;LOAD ZERO FILL SWITCH
7772 035556 112667 000207          MOVB (SP)+,$OMODE+1 ;;NUMBER OF DIGITS TO TYPE
7773 035562 062716 000002          ADD #2,(SP)        ;;ADJUST RETURN ADDRESS
7774 035566 000406          BR $TYPON
7775 035570 112767 000001 000171  $TYPOC: MOVB #1,$OFILL  ;;SET THE ZERO FILL SWITCH
7776 035576 112767 000006 000165  MOVB #6,$OMODE+1  ;;SET FOR SIX(6) DIGITS
7777 035604 112767 000005 000154  $TYPON: MOVB #5,$OCNT  ;;SET THE ITERATION COUNT
7778 035612 010346          MOV R3,-(SP)      ;;SAVE R3
7779 035614 010446          MOV R4,-(SP)      ;;SAVE R4
7780 035616 010546          MOV R5,-(SP)      ;;SAVE R5
7781 035620 116704 000145          MOVB $OMODE+1,R4  ;;GET THE NUMBER OF DIGITS TO TYPE
7782 035624 005404          NEG R4
7783 035626 062704 000006          ADD #6,R4         ;;SUBTRACT IT FOR MAX. ALLOWED
7784 035632 110467 000132          MOVB R4,$OMODE   ;;SAVE IT FOR USE
7785 035636 116704 000125          MOVB $OFILL,R4   ;;GET THE ZERO FILL SWITCH
7786 035642 016605 000012          MOV 12(SP),R5    ;;PICKUP THE INPUT NUMBER
7787 035646 005003          CLR R3           ;;CLEAR THE OUTPUT WORD
7788 035650 006105          1$: ROL R5      ;;ROTATE MSB INTO 'C'

```

```

7789 035652 000404
7790 035654 006105      2$:  BR      3$      ;;GO DO MSB
7791 035656 006105      ROL      R5      ;;FORM THIS DIGIT
7792 035660 006105      ROL      R5
7793 035662 010503      MOV      R5,R3
7794 035664 006103      3$:  ROL      R3      ;;GET LSB OF THIS DIGIT
7795 035666 105367 000076  DECB    $OMODE   ;;TYPE THIS DIGIT?
7796 035672 100016      BPL      7$      ;;BR IF NO
7797 035674 042703 177770  BIC     #177770,R3 ;;GET RID OF JUNK
7798 035700 001002      BNE     4$      ;;TEST FOR 0
7799 035702 005704      TST     R4      ;;SUPPRESS THIS 0?
7800 035704 001403      BEQ     5$      ;;BR IF YES
7801 035706 005204      4$:  INC     R4      ;;DON'T SUPPRESS ANYMORE 0'S
7802 035710 052703 000060  BIS     #'0,R3   ;;MAKE THIS DIGIT ASCII
7803 035714 052703 000040  5$:  BIS     #' ,R3  ;;MAKE ASCII IF NOT ALREADY
7804 035720 110367 000040  MOV     R3,8$   ;;SAVE FOR TYPING
7805 035724 104400 035764  TYPE    ,8$     ;;GO TYPE THIS DIGIT
7806 035730 105367 000032  7$:  DECB    $OCNT   ;;COUNT BY 1
7807 035734 003347      BGT     2$      ;;BR IF MORE TO DO
7808 035736 002402      BLT     6$      ;;BR IF DONE
7809 035740 005204      INC     R4      ;;INSURE LAST DIGIT ISN'T A BLANK
7810 035742 000744      BR      2$      ;;GO DO THE LAST DIGIT
7811 035744 012605      6$:  MOV     (SP)+,R5 ;;RESTORE R5
7812 035746 012604      MOV     (SP)+,R4 ;;RESTORE R4
7813 035750 012603      MOV     (SP)+,R3 ;;RESTORE R3
7814 035752 016666 000002 000004  MOV     2(SP),4(SP) ;;SET THE STACK FOR RETURNING
7815 035760 012616      MOV     (SP)+,(SP)
7816 035762 000002      RTI     ;;RETURN
7817 035764 000      8$:  .BYTE  0      ;;STORAGE FOR ASCII DIGIT
7818 035765 000      .BYTE  0      ;;TERMINATOR FOR TYPE ROUTINE
7819 035766 000      $OCNT: .BYTE  0  ;;OCTAL DIGIT COUNTER
7820 035767 000      $OFILL: .BYTE 0  ;;ZERO FILL SWITCH
7821 035770 000000      $OMODE: .WORD  0 ;;NUMBER OF DIGITS TO TYPE

```

.SBTTL CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

```

;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
;*REPLACED WITH SPACES.

```

```

;*CALL:
;*      MOV     NUM,-(SP)      ;;PUT THE BINARY NUMBER ON THE STACK
;*      TYPDS      ;;GO TO THE ROUTINE

```

```

$TYPDS:
MOV     R0,-(SP)      ;;PUSH R0 ON STACK
MOV     R1,-(SP)      ;;PUSH R1 ON STACK
MOV     R2,-(SP)      ;;PUSH R2 ON STACK
MOV     R3,-(SP)      ;;PUSH R3 ON STACK
MOV     R5,-(SP)      ;;PUSH R5 ON STACK
MOV     #20200,-(SP)   ;;SET BLANK SWITCH AND SIGN
MOV     20(SP),R5     ;;GET THE INPUT NUMBER
BPL     1$            ;;BR IF INPUT IS POS.
NEG     R5            ;;MAKE THE BINARY NUMBER POS.

```

```

7835 035772
7836 035772 010046
7837 035774 010146
7838 035776 010246
7839 036000 010346
7840 036002 010546
7841 036004 012746 020200
7842 036010 016605 000020
7843 036014 100004
7844 036016 005405

```

```

7845 036020 112766 000055 000001      MOVB    #'-,1(SP)      ;;MAKE THE ASCII NUMBER NEG.
7846 036026 005000                    CLR     R0              ;;ZERO THE CONSTANTS INDEX
7847 036030 012703 036206              MOV     #$DBLK,R3      ;;SETUP THE OUTPUT POINTER
7848 036034 112723 000040              MOVB   #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
7849 036040 005002                    CLR     R2              ;;CLEAR THE BCD NUMBER
7850 036042 016001 036176              MOV     $DTBL(R0),R1   ;;GET THE CONSTANT
7851 036046 160105                    SUB     R1,R5           ;;FORM THIS BCD DIGIT
7852 036050 002402                    BLT    4$              ;;BR IF DONE
7853 036052 005202                    INC     R2              ;;INCREASE THE BCD DIGIT BY 1
7854 036054 000774                    BR     3$              ;;
7855 036056 060105                    4$:    ADD     R1,R5           ;;ADD BACK THE CONSTANT
7856 036060 005702                    TST    R2              ;;CHECK IF BCD DIGIT=0
7857 036062 001002                    BNE    5$              ;;FALL THROUGH IF 0
7858 036064 105716                    TSTB   (SP)            ;;STILL DOING LEADING 0'S?
7859 036066 100407                    BMI    7$              ;;BR IF YES
7860 036070 106316                    5$:    ASLB   (SP)            ;;MSD?
7861 036072 103003                    BCC    6$              ;;BR IF NO
7862 036074 116663 000001 177777      MOVB   1(SP),-1(R3)    ;;YES--SET THE SIGN
7863 036102 052702 000060              6$:    BIS    #'0,R2       ;;MAKE THE BCD DIGIT ASCII
7864 036106 052702 000040              7$:    BIS    #' ,R2       ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
7865 036112 110223                    MOVB   R2,(R3)+       ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
7866 036114 005720                    TST    (R0)+          ;;JUST INCREMENTING
7867 036116 020027 000010              CMP     R0,#10        ;;CHECK THE TABLE INDEX
7868 036122 002746                    BLT    2$              ;;GO DO THE NEXT DIGIT
7869 036124 003002                    BGT    8$              ;;GO TO EXIT
7870 036126 010502                    MOV     R5,R2         ;;GET THE LSD
7871 036130 000764                    BR     6$              ;;GO CHANGE TO ASCII
7872 036132 105726                    8$:    TSTB   (SP)+       ;;WAS THE LSD THE FIRST NON-ZERO?
7873 036134 100003                    BPL    9$              ;;BR IF NO
7874 036136 116663 177777 177776      MOVB   -1(SP),-2(R3)  ;;YES--SET THE SIGN FOR TYPING
7875 036144 105013                    9$:    CLRB   (R3)         ;;SET THE TERMINATOR
7876 036146 012605                    MOV     (SP)+,R5      ;;POP STACK INTO R5
7877 036150 012603                    MOV     (SP)+,R3      ;;POP STACK INTO R3
7878 036152 012602                    MOV     (SP)+,R2      ;;POP STACK INTO R2
7879 036154 012601                    MOV     (SP)+,R1      ;;POP STACK INTO R1
7880 036156 012600                    MOV     (SP)+,R0      ;;POP STACK INTO R0
7881 036160 104400 036206              TYPE    , $DBLK        ;;NOW TYPE THE NUMBER
7882 036164 016666 000002 000004      MOV     2(SP),4(SP)   ;;ADJUST THE STACK
7883 036172 012616                    MOV     (SP)+,(SP)
7884 036174 000002                    RTI                    ;;RETURN TO USER
7885 036176 023420                    $DTBL: 10000.
7886 036200 001750                    1000.
7887 036202 000144                    100.
7888 036204 000012                    10.
7889 036206 000004                    $DBLK: .BLKW 4
7890
7891
7892
7893
7894
7895
7896
7897
7898
7899 036216 010046                    $TRAP: MOV    R0,-(SP)  ;;SAVE R0
7900 036220 016600 000002                    MOV     2(SP),R0      ;;GET TRAP ADDRESS
    
```

```

.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.
    
```

```

7901 036224 005740          TST      -(R0)          ;;BACKUP BY 2
7902 036226 111000          MOV     (R0),R0        ;;GET RIGHT BYTE OF TRAP
7903 036230 016000 036236  MOV     $TRPAD(R0),R0  ;;INDEX TO TABLE
7904 036234 000200          RTS      R0           ;;GO TO ROUTINE
7905
7906
7907
7908
7909
7910
7911
7912
7913
7914 036236
7915 036236 035330
7916 036240 035570
7917 036242 035544
7918 036244 035604
7919 036246 035772
7920
7921
7922
7923
7924
7925 036250 012737 036372 000024
7926 036256 012737 000340 000026
7927 036264 010046
7928 036266 010146
7929 036270 010246
7930 036272 010346
7931 036274 010446
7932 036276 010546
7933 036300 010667 000072
7934 036304 012737 036316 000024
7935 036312 000000
7936 036314 000776
7937
7938
7939 036316 016706 000054
7940 036322 005067 000050
7941 036326 005267 000044
7942 036332 001375
7943 036334 012605
7944 036336 012604
7945 036340 012603
7946 036342 012602
7947 036344 012601
7948 036346 012600
7949 036350 012737 036250 000024
7950 036356 012737 000340 000026
7951 036364 104400
7952 036366 036400
7953 036370 000002
7954 036372 000000
7955 036374 000776
7956 036376 000000
    
```

.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

```

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

:POWER UP ROUTINE

```

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

```

7957 036400 005015 047520 042527 $POWER: .ASCIZ <15><12>'POWER'
7958 036406 000122
7959
7960 036410 041600 052520 052440 MSG1: .EVEN
7961 036416 042116 051105 052040 .ASCIZ<CRLF> 'CPU UNDER TEST FOUND TO BE A '
7962 036424 051505 020124 047506
7963 036432 047125 020104 047524
7964 036440 041040 020105 020101
7965 036446 000
7966 036447 113 030502 026461 MSG3: .ASCIZ 'KB11-B/C OR KB11-CM
7967 036454 027502 020103 051117 ''<CRLF>
7968 036462 045440 030502 026461
7969 036470 046503 020040 020040
7970 036476 020040 020040 020040
7971 036504 020040 020040 020040
7972 036512 000200
7973 036514 020040 000040 MSG5: .ASCIZ '' ''
7974 036520 047200 052117 035105 MSG6: .ASCIZ <CRLF>'NOTE:''
7975 036526 000
7976 036530 .EVEN
7977
7978 036530 050123 020114 040506 EM1: .ASCIZ /SPL FAILED OR PSW PRIORITY BITS STUCK/
7979 036536 046111 042105 047440
7980 036544 020122 051520 020127
7981 036552 051120 047511 044522
7982 036560 054524 041040 052111
7983 036566 020123 052123 041525
7984 036574 000113
7985 036576 051105 047522 020122 DH1: .ASCII /ERROR PC SPL 5 PSW SPL 2 PSW TST NUM/<CRLF>
7986 036604 041520 020040 051440
7987 036612 046120 032440 050040
7988 036620 053523 020040 020040
7989 036626 020040 051440 046120
7990 036634 031040 050040 053523
7991 036642 020040 052040 052123
7992 036650 047040 046525 200
7993 036655 011 042440 050130 .ASCIZ / EXPECT ACTUAL EXPECT ACTUAL/
7994 036662 041505 020124 041501
7995 036670 052524 046101 020040
7996 036676 042440 050130 041505
7997 036704 020124 041501 052524
7998 036712 046101 000
7999 036716
8000 036716 001116 001214 001206 DT1: .EVEN
8001 036724 001212 001162 001210 .WORD $ERRPC,$PR5,$ERPSW,$PR2,$TMP0,$$TSTNM,0
8002 036732 000000
8003 036734 051120 032440 047440 EM2: .ASCIZ /PR 5 OK PR 2 BAD/
8004 036742 020113 051120 031040
8005 036750 041040 042101 000
8006 036755 120 020122 020062 EM3: .ASCIZ /PR 2 OK BUT PR 5 BAD/
8007 036762 045517 041040 052125
8008 036770 050040 020122 020065
8009 036776 040502 000104
8010 037002 040522 043103 042440 EM4: .ASCIZ /RACF E3 BAD(NOT GOING HIGH ON SPL)/
8011 037010 020063 040502 024104
8012 037016 047516 020124 047507
    
```

8013	037024	047111	020107	044510		
8014	037032	044107	047440	020116		
8015	037040	050123	024514	000		
8016	037045	105	051122	051117	DH4:	.ASCIZ /ERRORPC TEST NUMBER/
8017	037052	041520	052040	051505		
8018	037060	020124	052516	041115		
8019	037066	051105	000			
8020		037072				
8021	037072	001116	001210	000000	DT4:	.EVEN
8022	037100	040522	043103	042440	EM5:	.WORD \$ERRPC,\$\$TSTNM,0
8023	037106	020065	040502	024104		.ASCIZ /RACF E5 BAD(AFIRO4(1)*AFIRO3(1)*AFIRO5(1)*(RTS:CCOP))/
8024	037114	043101	051111	032060		
8025	037122	030450	025051	043101		
8026	037130	051111	031460	030450		
8027	037136	025051	043101	051111		
8028	037144	032460	030450	025051		
8029	037152	051050	051524	041472		
8030	037160	047503	024520	000051		
8031	037166	042522	042523	020124	EM6:	.ASCIZ /RESET DIDN'T SEND OUT INIT/
8032	037174	044504	047104	052047		
8033	037202	051440	047105	020104		
8034	037210	052517	020124	047111		
8035	037216	052111	000			
8036	037221	122	041501	020106	EM7:	.ASCIZ /RACF E5 BAD/
8037	037226	032505	041040	042101		
8038	037234	000				
8039	037235	122	041501	020106	EM10:	.ASCIZ /RACF E8 BAD/
8040	037242	034105	041040	042101		
8041	037250	000				
8042	037251	122	041501	020106	EM11:	.ASCIZ "RACF E17 BAD(AFIRO7(0)*X/CLASS"
8043	037256	030505	020067	040502		
8044	037264	024104	043101	051111		
8045	037272	033460	030050	025051		
8046	037300	027530	046103	051501		
8047	037306	000123				
8048	037310	040522	043103	042440	EM12:	.ASCIZ "RACF E5 BAD(AFIRO6(0)*X/CLASS)"
8049	037316	020065	040502	024104		
8050	037324	043101	051111	033060		
8051	037332	030050	025051	027530		
8052	037340	046103	051501	024523		
8053	037346	000				
8054	037347	120	041103	042040	EM13:	.ASCIZ /PCB DIDN'T LOAD FROM R5/
8055	037354	042111	023516	020124		
8056	037362	047514	042101	043040		
8057	037370	047522	020115	032522		
8058	037376	000				
8059	037377	123	020120	044504	EM14:	.ASCIZ /SP DIDN'T LOAD PROPERLY/
8060	037404	047104	052047	046040		
8061	037412	040517	020104	051120		
8062	037420	050117	051105	054514		
8063	037426	000				
8064	037427	105	051122	051117	DH14:	.ASCII /ERRORPC SP TST NUM/<CRLF>
8065	037434	041520	020040	020040		
8066	037442	020040	051440	004520		
8067	037450	052040	052123	047040		
8068	037456	046525	200			

Line	Address	Hex	Hex	Hex	Label	Description
8069	037461	040	020040	020040		.ASCIZ / EXPECT ACTUAL/
8070	037466	020040	042440	050130		
8071	037474	041505	020124	040440		
8072	037502	052103	040525	000114		
8073						
8074	037510	001116	001156	001154	DT14:	.EVEN \$ERRPC,\$REG1,\$REG0,\$\$TSTNM,0
8075	037516	001210	000000			
8076	037522	032522	042040	042111	EM15:	.ASCIZ /R5 DIDN'T LOAD PROPERLY/
8077	037530	023516	020124	047514		
8078	037536	042101	050040	047522		
8079	037544	042520	046122	000131		
8080	037552	051105	047522	050122	DH15:	.ASCII /ERRORPC R5 TST NUM/<CRLF>
8081	037560	020103	020040	020040		
8082	037566	020040	032522	020040		
8083	037574	020040	020040	020040		
8084	037602	051524	020124	052516		
8085	037610	100115				
8086	037612	020011	054105	042520		.ASCIZ / EXPECT ACTUAL/
8087	037620	052103	020040	041501		
8088	037626	052524	046101	000		
8089	037633	122	041501	020106	EM16:	.ASCIZ "RACF X/CLASS DIDN'T GO HIGH"
8090	037640	027530	046103	051501		
8091	037646	020123	044504	047104		
8092	037654	052047	043440	020117		
8093	037662	044510	044107	000		
8094	037667	122	041501	020106	EM17:	.ASCIZ /RACF E34 BAD OR NOT GETTING THRU RACL RADR05/
8095	037674	031505	020064	040502		
8096	037702	020104	051117	047040		
8097	037710	052117	043440	052105		
8098	037716	044524	043516	052040		
8099	037724	051110	020125	040522		
8100	037732	046103	051040	042101		
8101	037740	030122	000065			
8102	037744	051107	045101	051440	EM20:	.ASCIZ /GRAJ SC05 L DOESN'T GET THRU TO RACK BRCAB04 L AS A HIGH/
8103	037752	030103	020065	020114		
8104	037760	047504	051505	023516		
8105	037766	020124	042507	020124		
8106	037774	044124	052522	052040		
8107	040002	020117	040522	045503		
8108	040010	041040	041522	041101		
8109	040016	032060	046040	040440		
8110	040024	020123	020101	044510		
8111	040032	044107	000			
8112	040035	122	020061	044504	EM21:	.ASCIZ /R1 DIDN'T SHIFT/
8113	040042	047104	052047	051440		
8114	040050	044510	052106	000		
8115	040055	122	020061	044123	EM22:	.ASCII /R1 SHIFTED BUT CARRY DIDN'T SET/<CRLF>
8116	040062	043111	042524	020104		
8117	040070	052502	020124	040503		
8118	040076	051122	020131	044504		
8119	040104	047104	052047	051440		
8120	040112	052105	200			
8121	040115	123	044510	052106		.ASCIZ /SHIFT COUNTER COULD BE STUCK/
8122	040122	041440	052517	052116		
8123	040130	051105	041440	052517		
8124	040136	042114	041040	020105		

8125	040144	052123	041525	000113	
8126	040152	051107	045101	051440	EM23: .ASCIZ /GRAJ SC05 L DOESN'T GET THRU TO RACK BRCAB04 L AS A LOW/
8127	040160	030103	020065	020114	
8128	040166	047504	051505	023516	
8129	040174	020124	042507	020124	
8130	040202	044124	052522	052040	
8131	040210	020117	040522	045503	
8132	040216	041040	041522	041101	
8133	040224	032060	046040	040440	
8134	040232	020123	020101	047514	
8135	040240	000127			
8136	040242	051501	020110	044522	EM24: .ASCIZ /ASH RIGHT DIDN'T SIGN FILL/
8137	040250	044107	020124	044504	
8138	040256	047104	052047	051440	
8139	040264	043511	020116	044506	
8140	040272	046114	000		
8141	040275	122	020061	044504	EM25: .ASCIZ /R1 DIDN'T SHIFT CORRECTLY/
8142	040302	047104	052047	051440	
8143	040310	044510	052106	041440	
8144	040316	051117	042522	052103	
8145	040324	054514	000		
8146	040327	105	051122	051117	DH25: .ASCII /ERRORPC R1 TST NUM/<CRLF>
8147	040334	041520	020040	020040	
8148	040342	020040	051040	004461	
8149	040350	052040	052123	047040	
8150	040356	046525	200		
8151	040361	011	042440	050130	.ASCIZ / EXPECT ACTUAL/
8152	040366	041505	020124	040440	
8153	040374	052103	040525	000114	
8154					.EVEN
8155	040402	001116	001164	001156	DT25: .WORD \$ERRPC,\$TMP1,\$REG1,\$\$TSTNM,0
8156	040410	001210	000000		
8157	040414	030522	051440	044510	EM26: .ASCIZ /R1 SHIFTED BUT CARRY DIDN'T SET/
8158	040422	052106	042105	041040	
8159	040430	052125	041440	051101	
8160	040436	054522	042040	042111	
8161	040444	023516	020124	042523	
8162	040452	000124			
8163	040454	051501	027110	030062	EM27: .ASCIZ /ASH.20 DIDN'T LOAD CC'S CORRECTLY/
8164	040462	042040	042111	023516	
8165	040470	020124	047514	042101	
8166	040476	041440	023503	020123	
8167	040504	047503	051122	041505	
8168	040512	046124	000131		
8169	040516	030522	051440	044510	EM30: .ASCIZ /R1 SHIFTED WHEN SHIFT COUNT=0/
8170	040524	052106	042105	053440	
8171	040532	042510	020116	044123	
8172	040540	043111	020124	047503	
8173	040546	047125	036524	000060	
8174	040554	051501	027110	030064	EM31: .ASCIZ /ASH.40 DIDN'T LOAD CC'S CORRECTLY/
8175	040562	042040	042111	023516	
8176	040570	020124	047514	042101	
8177	040576	041440	023503	020123	
8178	040604	047503	051122	041505	
8179	040612	046124	000131		
8180	040616	040522	043103	052440	EM32: .ASCIZ "RACF U/CLASS DOESN'T GO HIGH ON ASH"

8181	040624	041457	040514	051523	
8182	040632	042040	042517	047123	
8183	040640	052047	043440	020117	
8184	040646	044510	044107	047440	
8185	040654	020116	051501	000110	
8186	040662	051501	027110	030060	EM33: .ASCIZ /ASH.00 FAILED/
8187	040670	043040	044501	042514	
8188	040676	000104			
8189	040700	051111	041103	042440	EM34: .ASCIZ /IRCB E35(B2) NOT GOING LOW/
8190	040706	032463	041050	024462	
8191	040714	047040	052117	043440	
8192	040722	044517	043516	046040	
8193	040730	053517	000		
8194	040733	111	041522	020102	EM35: .ASCII /IRCB (MUL:ASHC)+MFP L DIDN'T GO LOW OR IRCB E37 BAD/<CRLF>
8195	040740	046450	046125	040472	
8196	040746	044123	024503	046453	
8197	040754	050106	046040	042040	
8198	040762	042111	023516	020124	
8199	040770	047507	046040	053517	
8200	040776	047440	020122	051111	
8201	041004	041103	042440	033463	
8202	041012	041040	042101	200	
8203	041017	117	020122	051111	.ASCIZ /OR IRCB E35(B1) STUCK LOW/
8204	041024	041103	042440	032463	
8205	041032	041050	024461	051440	
8206	041040	052524	045503	046040	
8207	041046	053517	000		
8208	041051	122	041501	020105	EM36: .ASCIZ /RACE (MUL:ASHC+MFP) DIDN'T GO HIGH OR RACE E44 BAD/
8209	041056	046450	046125	040472	
8210	041064	044123	025503	043115	
8211	041072	024520	042040	042111	
8212	041100	023516	020124	047507	
8213	041106	044040	043511	020110	
8214	041114	051117	051040	041501	
8215	041122	020105	032105	020064	
8216	041130	040502	000104		
8217	041134	040522	042503	042440	EM37: .ASCIZ /RACE E45 BAD (AFIRO4(1)*(MUL:ASHC+MFP))/
8218	041142	032464	041040	042101	
8219	041150	024040	043101	051111	
8220	041156	032060	030450	025051	
8221	041164	046450	046125	040472	
8222	041172	044123	025503	043115	
8223	041200	024520	000051		
8224	041204	040522	042503	042440	EM40: .ASCIZ /RACE E33 BAD (AFIRO5(1)*(MUL:ASHC+MFP))/
8225	041212	031463	041040	042101	
8226	041220	024040	043101	051111	
8227	041226	032460	030450	025051	
8228	041234	046450	046125	040472	
8229	041242	044123	025503	043115	
8230	041250	024520	000051		
8231	041254	030122	042040	042111	EM41: .ASCIZ /RO DIDN'T SIGN FILL ON RIGHT SHIFT/
8232	041262	023516	020124	044523	
8233	041270	047107	043040	046111	
8234	041276	020114	047117	051040	
8235	041304	043511	052110	051440	
8236	041312	044510	052106	000	

8237	041317	105	051122	051117	DH41:	.ASCII	/ERRORPC	RO	TST NUM/<CRLF>
8238	041324	041520	020040	020040					
8239	041332	020040	020040	030122					
8240	041340	020011	052040	052123					
8241	041346	047040	046525	200					
8242	041353	011	020040	054105		.ASCIZ	/	EXPECT	ACTUAL/
8243	041360	042520	052103	020040					
8244	041366	041501	052524	046101					
8245	041374	000							
8246		041376							
8247	041376	001116	001162	001154	DT41:	.EVEN			
8248	041404	001210	000000			.WORD	\$ERRPC,\$TMPO,\$REGO,\$\$TSTNM,0		
8249	041410	040502	020104	041503	EM42:	.ASCIZ	/BAD CC'S ON RIGHT SHIFT/		
8250	041416	051447	047440	020116					
8251	041424	044522	044107	020124					
8252	041432	044123	043111	000124					
8253	041440	051105	047522	050122	DH42:	.ASCII	/ERRORPC	PSW	TST NUM/<CRLF>
8254	041446	020103	020040	020040					
8255	041454	020040	051520	004527					
8256	041462	020040	051524	020124					
8257	041470	052516	100115						
8258	041474	020011	054105	042520		.ASCIZ	/	EXPECT	ACTUAL/
8259	041502	052103	020040	041501					
8260	041510	052524	046101	000					
8261		041516							
8262	041516	001116	001162	001206	DT42:	.EVEN			
8263	041524	001210	000000			.WORD	\$ERRPC,\$TMPO,\$ERPSW,\$\$TSTNM,0		
8264	041530	030122	030074	020076	EM43:	.ASCIZ	/RO<0> DIDN'T GO TO R1<15>/		
8265	041536	044504	047104	052047					
8266	041544	043440	020117	047524					
8267	041552	051040	036061	032461					
8268	041560	000076							
8269	041562	051105	047522	050122	DH43:	.ASCII	/ERRORPC	RO	R1 TST NUM/<CRLF>
8270	041570	020103	020040	020040					
8271	041576	020040	030122	020011					
8272	041604	020040	020040	020040					
8273	041612	051040	004461	020040					
8274	041620	051524	020124	052516					
8275	041626	100115							
8276	041630	020011	054105	042520		.ASCIZ	/	EXPECT	ACTUAL EXPECT ACTUAL/
8277	041636	052103	020040	041501					
8278	041644	052524	046101	020040					
8279	041652	042440	050130	041505					
8280	041660	020124	040440	052103					
8281	041666	040525	000114						
8282									
8283	041672	001116	001162	001154	DT43:	.EVEN			
8284	041700	001164	001156	001210		.WORD	\$ERRPC,\$TMPO,\$REGO,\$TMP1,\$REG1,\$\$TSTNM,0		
8285	041706	000000							
8286	041710	030122	042040	042111	EM44:	.ASCIZ	/RO DIDN'T GET SHIFTED LEFT PROPERLY/		
8287	041716	023516	020124	042507					
8288	041724	020124	044123	043111					
8289	041732	042524	020104	042514					
8290	041740	052106	050040	047522					
8291	041746	042520	046122	000131					
8292	041754	040502	020104	041503	EM45:	.ASCIZ	/BAD CC'S ON LEFT SHIFT/		

8349	042432	044502	020124	052123	EM52:	.ASCIZ	/BIT STUCK IN SHIFT COUNTER WITH 25 PATTERN/
8350	042440	041525	020113	047111			
8351	042446	051440	044510	052106			
8352	042454	041440	052517	052116			
8353	042462	051105	053440	052111			
8354	042470	020110	032462	050040			
8355	042476	052101	042524	047122			
8356	042504	000					
8357	042505	111	041522	020102	EM53:	.ASCIZ	/IRCB E35(B3) NOT GOING LOW/
8358	042512	031505	024065	031502			
8359	042520	020051	047516	020124			
8360	042526	047507	047111	020107			
8361	042534	047514	000127				
8362	042540	051501	027103	030060	EM54:	.ASCIZ	/ASC.00 FAILED/
8363	042546	043040	044501	042514			
8364	042554	000104					
8365	042556	024122	052515	035114	EM55:	.ASCIZ	/R(MUL:ASHC+MFP) FIELD IN INSTR DECODE ROM BAD/
8366	042564	051501	041510	046453			
8367	042572	050106	020051	044506			
8368	042600	046105	020104	047111			
8369	042606	044440	051516	051124			
8370	042614	042040	041505	042117			
8371	042622	020105	047522	020115			
8372	042630	040502	000104				
8373	042634	051107	042101	042040	EM56:	.ASCIZ	/GRAD DROO STUCK HIGH OR RACK E64(C1) BAD/
8374	042642	030122	020060	052123			
8375	042650	041525	020113	044510			
8376	042656	044107	047440	020122			
8377	042664	040522	045503	042440			
8378	042672	032066	041450	024461			
8379	042700	041040	042101	000			
8380	042705	107	040522	020104	EM57:	.ASCIZ	/GRAD DROO STUCK LOW OR RACK E64(C1) BAD/
8381	042712	051104	030060	051440			
8382	042720	052524	045503	046040			
8383	042726	053517	047440	020122			
8384	042734	040522	045503	042440			
8385	042742	032066	041450	024461			
8386	042750	041040	042101	000			
8387	042755	107	040522	020112	EM60:	.ASCII	/GRAJ SC=0 L NOT GETTING TO RACK E50(C1) OR/<CRLF>
8388	042762	041523	030075	046040			
8389	042770	047040	052117	043440			
8390	042776	052105	044524	043516			
8391	043004	052040	020117	040522			
8392	043012	045503	042440	030065			
8393	043020	041450	024461	047440			
8394	043026	100122					
8395	043030	042440	030065	041040		.ASCIZ	/ E50 BAD (FAILED LOW)/
8396	043036	042101	024040	040506			
8397	043044	046111	042105	046040			
8398	043052	053517	000051				
8399	043056	030122	047440	020122	EM61:	.ASCIZ	/R0 OR R1 OR BOTH BAD ON POSITIVE MULTIPLICAND/
8400	043064	030522	047440	020122			
8401	043072	047502	044124	041040			
8402	043100	042101	047440	020116			
8403	043106	047520	044523	044524			
8404	043114	042526	046440	046125			

8405	043122	044524	046120	041511	
8406	043130	047101	000104		
8407	043134	040502	020104	041503	EM62: .ASCIZ /BAD CC'S ON POSITIVE MULTIPLICAND/
8408	043142	051447	047440	020116	
8409	043150	047520	044523	044524	
8410	043156	042526	046440	046125	
8411	043164	044524	046120	041511	
8412	043172	047101	000104		
8413	043176	030122	041040	042101	EM63: .ASCIZ /R0 BAD ON NEGATIVE MULTIPLICAND/
8414	043204	047440	020116	042516	
8415	043212	040507	044524	042526	
8416	043220	046440	046125	044524	
8417	043226	046120	041511	047101	
8418	043234	000104			
8419	043236	030522	041040	042101	EM64: .ASCIZ /R1 BAD ON NEGATIVE MULTIPLICAND/
8420	043244	047440	020116	042516	
8421	043252	040507	044524	042526	
8422	043260	046440	046125	044524	
8423	043266	046120	041511	047101	
8424	043274	000104			
8425	043276	040502	020104	041503	EM65: .ASCIZ /BAD CC'S DUE TO STATE MUL.50/
8426	043304	051447	042040	042525	
8427	043312	052040	020117	052123	
8428	043320	052101	020105	052515	
8429	043326	027114	030065	000	
8430	043333	103	042040	042111	EM66: .ASCIZ /C DIDN'T SET ON OVERFLOW/
8431	043340	023516	020124	042523	
8432	043346	020124	047117	047440	
8433	043354	042526	043122	047514	
8434	043362	000127			
8435	043364	020103	044504	047104	EM67: .ASCIZ /C DIDN'T SET ON UNDERFLOW/
8436	043372	052047	051440	052105	
8437	043400	047440	020116	047125	
8438	043406	042504	043122	047514	
8439	043414	000127			
8440	043416	052515	027114	030060	EM70: .ASCIZ /MUL.00 FAILED/
8441	043424	043040	044501	042514	
8442	043432	000104			
8443	043434	051501	027110	030063	EM72: .ASCIZ /ASH.30 DIDN'T LOAD CC'S CORRECTLY/
8444	043442	042040	042111	023516	
8445	043450	020124	047514	042101	
8446	043456	041440	023503	020123	
8447	043464	047503	051122	041505	
8448	043472	046124	000131		
8449	043476	051501	027110	030464	EM73: .ASCIZ /ASH.41 FAILED/
8450	043504	043040	044501	042514	
8451	043512	000104			
8452	043514	051501	027110	030464	EM74: .ASCIZ /ASH.41 DIDN'T LOAD CC'S CORRECTLY/
8453	043522	042040	042111	023516	
8454	043530	020124	047514	042101	
8455	043536	041440	023503	020123	
8456	043544	047503	051122	041505	
8457	043552	046124	000131		
8458	043556	051111	043103	055040	EM75: .ASCIZ /IRCF Z2(1) DOESN'T GO HIGH OR RACK E49(B1) STUCK LOW/
8459	043564	024062	024461	042040	
8460	043572	042517	047123	052047	

8461	043600	043440	020117	044510	
8462	043606	044107	047440	020122	
8463	043614	040522	045503	042440	
8464	043622	034464	041050	024461	
8465	043630	051440	052524	045503	
8466	043636	046040	053517	000	
8467	043643	103	023503	020123	EM76: .ASCIZ /CC'S BAD IN DVE.00/
8468	043650	040502	020104	047111	
8469	043656	042040	042526	030056	
8470	043664	000060			
8471	043666	041503	051447	041040	EM77: .ASCIZ /CC'S BAD IN DVC.70/
8472	043674	042101	044440	020116	
8473	043702	053104	027103	030067	
8474	043710	000			
8475	043711	122	041501	020113	EM100: .ASCIZ /RACK E50(B0) STUCK HIGH/
8476	043716	032505	024060	030102	
8477	043724	020051	052123	041525	
8478	043732	020113	044510	044107	
8479	043740	000			
8480					
8481	043741	107	040522	020112	EM101: .ASCIZ /GRAJ DIV SUB L NOT GOING LOW OR NOT GETTING THRU/<CRLF>
8482	043746	044504	020126	052523	
8483	043754	020102	020114	047516	
8484	043762	020124	047507	047111	
8485	043770	020107	047514	020127	
8486	043776	051117	047040	052117	
8487	044004	043440	052105	044524	
8488	044012	043516	052040	051110	
8489	044020	100125	000		
8490	044023	124	020117	040522	.ASCIZ /TO RACK E49 OR E49(D0) STUCK HIGH/
8491	044030	045503	042440	034464	
8492	044036	047440	020122	032105	
8493	044044	024071	030104	020051	
8494	044052	052123	041525	020113	
8495	044060	044510	044107	000	
8496	044065	121	047525	020124	EM102: .ASCIZ /QUOT OK REMAINDER BAD (ROM STATE FAILURE)/
8497	044072	045517	051040	046505	
8498	044100	044501	042116	051105	
8499	044106	041040	042101	024040	
8500	044114	047522	020115	052123	
8501	044122	052101	020105	040506	
8502	044130	046111	051125	024505	
8503	044136	000			
8504	044137	122	041501	020113	EM103: .ASCIZ /RACK E49(A1) STUCK LOW/
8505	044144	032105	024071	030501	
8506	044152	020051	052123	041525	
8507	044160	020113	047514	000127	
8508	044166	051107	045101	042040	EM104: .ASCIZ /GRAJ DIV SUB L NOT GOING HIGH OR RACK E49(D0) STUCK LOW/
8509	044174	053111	051440	041125	
8510	044202	046040	047040	052117	
8511	044210	043440	044517	043516	
8512	044216	044040	043511	020110	
8513	044224	051117	051040	041501	
8514	044232	020113	032105	024071	
8515	044240	030104	020051	052123	
8516	044246	041525	020113	047514	

8517	044254	000127			
8518	044256	052521	047524	042511	EM105: .ASCIZ /QUTOIENT & REMAINDER BAD/
8519	044264	052116	023040	051040	
8520	044272	046505	044501	042116	
8521	044300	051105	041040	042101	
8522	044306	000			
8523	044307	107	040522	020110	EM106: .ASCIZ /GRAH SR15 NOT GETTING TO RACK E64 OR E64(B0) STUCK HIGH/
8524	044314	051123	032461	047040	
8525	044322	052117	043440	052105	
8526	044330	044524	043516	052040	
8527	044336	020117	040522	045503	
8528	044344	042440	032066	047440	
8529	044352	020122	033105	024064	
8530	044360	030102	020051	052123	
8531	044366	041525	020113	044510	
8532	044374	044107	000		
8533	044377	111	041522	020110	EM110: .ASCIZ /IRCH N(1) NOT GETTING TO RACK E63 OR E63(D0) STUCK HIGH/
8534	044404	024116	024461	047040	
8535	044412	052117	043440	052105	
8536	044420	044524	043516	052040	
8537	044426	020117	040522	045503	
8538	044434	042440	031466	047440	
8539	044442	020122	033105	024063	
8540	044450	030104	020051	052123	
8541	044456	041525	020113	044510	
8542	044464	044107	000		
8543	044467	122	041501	020113	EM111: .ASCIZ /RACK E49(B1) STUCK HIGH/
8544	044474	032105	024071	030502	
8545	044502	020051	052123	041525	
8546	044510	020113	044510	044107	
8547	044516	000			
8548	044517	107	040522	020112	EM112: .ASCIZ /GRAJ DIV QUIT L NOT GOING HIGH OR NOT GETTING/<CRLF>
8549	044524	044504	020126	052521	
8550	044532	052111	046040	047040	
8551	044540	052117	043440	044517	
8552	044546	043516	044040	043511	
8553	044554	020110	051117	047040	
8554	044562	052117	043440	052105	
8555	044570	044524	043516	000200	
8556	044576	047524	051040	041501	.ASCIZ /TO RACK E63 OR E63(C0) STUCK LOW/
8557	044604	020113	033105	020063	
8558	044612	051117	042440	031466	
8559	044620	041450	024460	051440	
8560	044626	052524	045503	046040	
8561	044634	053517	000		
8562	044637	122	041501	020113	EM113: .ASCIZ /RACK E50(B0) STUCK LOW/
8563	044644	032505	024060	030102	
8564	044652	020051	052123	041525	
8565	044660	020113	047514	000127	
8566	044666	040522	045503	042440	EM114: .ASCIZ /RACK E64(B0) STUCK LOW/
8567	044674	032066	041050	024460	
8568	044702	051440	052524	045503	
8569	044710	046040	053517	000	
8570	044715	104	041526	031056	EM115: .ASCIZ /DVC.20,DVC.40,DVC.80,OR DVC.90 FAILED/
8571	044722	026060	053104	027103	
8572	044730	030064	042054	041526	

8573	044736	034056	026060	051117	
8574	044744	042040	041526	034456	
8575	044752	020060	040506	046111	
8576	044760	042105	000		
8577	044763	102	042101	041440	EM117: .ASCIZ /BAD CC'S IN DVC.90 OR RACK E63(D0) STUCK LOW/
8578	044770	023503	020123	047111	
8579	044776	042040	041526	034456	
8580	045004	020060	051117	051040	
8581	045012	041501	020113	033105	
8582	045020	024063	030104	020051	
8583	045026	052123	041525	020113	
8584	045034	047514	000127		
8585	045040	051107	045101	042040	EM120: .ASCIZ /GRAJ DIV QUIT DIDN'T GO LOW OR RACK E63(C0) STUCK HIGH/
8586	045046	053111	050440	044525	
8587	045054	020124	044504	047104	
8588	045062	052047	043440	020117	
8589	045070	047514	020127	051117	
8590	045076	051040	041501	020113	
8591	045104	033105	024063	030103	
8592	045112	020051	052123	041525	
8593	045120	020113	044510	044107	
8594	045126	000			
8595	045127	103	023503	020123	EM121: .ASCIZ /CC'S BAD DUE TO EITHER DIV.30 OR DVE.20/
8596	045134	040502	020104	052504	
8597	045142	020105	047524	042440	
8598	045150	052111	042510	020122	
8599	045156	044504	027126	030063	
8600	045164	047440	020122	053104	
8601	045172	027105	030062	000	
8602	045177	107	040522	020112	EM122: .ASCIZ /GRAJ E5 BAD(Z2(0)*LEFT SAVE(1))/
8603	045204	032505	041040	042101	
8604	045212	055050	024062	024460	
8605	045220	046052	043105	020124	
8606	045226	040523	042526	030450	
8607	045234	024451	000		
8608	045237	122	041501	020113	EM123: .ASCIZ /RACK E63(D0) STUCK LOW/
8609	045244	033105	024063	030104	
8610	045252	020051	052123	041525	
8611	045260	020113	047514	000127	
8612	045266	041503	051447	042040	EM124: .ASCIZ /CC'S DIDN'T LOAD PROPERLY/
8613	045274	042111	023516	020124	
8614	045302	047514	042101	050040	
8615	045310	047522	042520	046122	
8616	045316	000131			
8617	045320	051107	045101	042440	EM125: .ASCIZ /GRAJ E5(N(1)*SR15(1)) BAD OR ROM STATE BAD/
8618	045326	024065	024116	024461	
8619	045334	051452	030522	024065	
8620	045342	024461	020051	040502	
8621	045350	020104	051117	051040	
8622	045356	046517	051440	040524	
8623	045364	042524	041040	042101	
8624	045372	000			
8625	045373	121	047525	020124	EM127: .ASCIZ /QUOT BAD REMAINDER OK (ROM STATE FAILURE)/
8626	045400	040502	020104	042522	
8627	045406	040515	047111	042504	
8628	045414	020122	045517	024040	

8629	045422	047522	020115	052123	
8630	045430	052101	020105	040506	
8631	045436	046111	051125	024505	
8632	045444	000			
8633	045445	121	047525	044524	EM130: .ASCIZ /QUOTIENT BAD (ROM STATE FAILURE)/
8634	045452	047105	020124	040502	
8635	045460	020104	051050	046517	
8636	045466	051440	040524	042524	
8637	045474	043040	044501	052514	
8638	045502	042522	000051		
8639	045506	040502	020104	041503	EM134: .ASCIZ /BAD CC'S ON DIV OVERFLOW, STATE DVD.10/
8640	045514	051447	047440	020116	
8641	045522	044504	020126	053117	
8642	045530	051105	046106	053517	
8643	045536	020054	052123	052101	
8644	045544	020105	053104	027104	
8645	045552	030061	000		
8646	045555	122	020060	040502	EM135: .ASCIZ /RO BAD/
8647	045562	000104			
8648	045564	052123	052101	020105	EM136: .ASCIZ /STATE MTP.00 DIDN'T INC SP/
8649	045572	052115	027120	030060	
8650	045600	042040	042111	023516	
8651	045606	020124	047111	020103	
8652	045614	050123	000		
8653	045617	122	041501	020106	EM140: .ASCIZ "RACF X/CLASS DOESN'T GO HIGH"
8654	045624	027530	046103	051501	
8655	045632	020123	047504	051505	
8656	045640	023516	020124	047507	
8657	045646	044040	043511	000110	
8658	045654	052115	027120	030061	EM141: .ASCIZ /MTP.10 FAILED TO RELOAD THE DR/
8659	045662	043040	044501	042514	
8660	045670	020104	047524	051040	
8661	045676	046105	040517	020104	
8662	045704	044124	020105	051104	
8663	045712	000			
8664	045713	123	020120	047514	EM142: .ASCIZ /SP LOADED INCORRECTLY/
8665	045720	042101	042105	044440	
8666	045726	041516	051117	042522	
8667	045734	052103	054514	000	
8668	045741	115	050124	030456	EM143: .ASCIZ /MTP.10 DIDN'T PUT PCB IN DR/
8669	045746	020060	044504	047104	
8670	045754	052047	050040	052125	
8671	045762	050040	041103	044440	
8672	045770	020116	051104	000	
8673	045775	122	041501	020106	EM144: .ASCIZ /RACF E20(4) STUCK HIGH/
8674	046002	031105	024060	024464	
8675	046010	051440	052524	045503	
8676	046016	044040	043511	000110	
8677	046024	043115	027120	030061	EM145: .ASCIZ /MFP.10 DIDN'T DECREMENT THE SP/
8678	046032	042040	042111	023516	
8679	046040	020124	042504	051103	
8680	046046	046505	047105	020124	
8681	046054	044124	020105	050123	
8682	046062	000			
8683	046063	122	020060	044504	EM146: .ASCIZ /RO DIDN'T GET PUT ON THE STACK/
8684	046070	047104	052047	043440	

8685	046076	052105	050040	052125	
8686	046104	047440	020116	044124	
8687	046112	020105	052123	041501	
8688	046120	000113			
8689	046122	040502	020104	041503	EM147: .ASCIZ /BAD CC'S, CC CONTROL ROM/
8690	046130	051447	020054	041503	
8691	046136	041440	047117	051124	
8692	046144	046117	051040	046517	
8693	046152	000			
8694	046153	115	050106	030056	EM151: .ASCIZ /MFP.00 BAD/
8695	046160	020060	040502	000104	
8696	046166	040502	020104	041503	EM152: .ASCIZ /BAD CC'S DUE TO MFP.00/
8697	046174	051447	042040	042525	
8698	046202	052040	020117	043115	
8699	046210	027120	030060	000	
8700	046215	124	050122	030056	EM155: .ASCIZ /TRP.01 FAILED TO LOAD BR/
8701	046222	020061	040506	046111	
8702	046230	042105	052040	020117	
8703	046236	047514	042101	041040	
8704	046244	000122			
8705	046246	051111	042103	044440	EM156: .ASCII /IRCD IOT DOESN'T GO LOW OR DAPE TV04 DOES/<CRLF>
8706	046254	052117	042040	042517	
8707	046262	047123	052047	043440	
8708	046270	020117	047514	020127	
8709	046276	051117	042040	050101	
8710	046304	020105	053124	032060	
8711	046312	042040	042517	100123	
8712	046320	047516	020124	047507	.ASCIZ /NOT GO HIGH OR DOESN'T GET TO THE ALU/
8713	046326	044040	043511	020110	
8714	046334	051117	042040	042517	
8715	046342	047123	052047	043440	
8716	046350	052105	052040	020117	
8717	046356	044124	020105	046101	
8718	046364	000125			
8719	046366	051124	027120	030460	EM157: .ASCIZ /TRP.01 FAILED TO LOAD DR/
8720	046374	043040	044501	042514	
8721	046402	020104	047524	046040	
8722	046410	040517	020104	051104	
8723	046416	000			
8724	046417	124	050122	030056	EM160: .ASCIZ /TRP.00 FAILED TO LOAD BR/
8725	046424	020060	040506	046111	
8726	046432	042105	052040	020117	
8727	046440	047514	042101	041040	
8728	046446	000122			
8729	046450	051111	042103	047440	EM161: .ASCII /IRCD OPCODE3 DOESN'T GO LOW OR DOESN'T/<CRLF>
8730	046456	041520	042117	031505	
8731	046464	042040	042517	047123	
8732	046472	052047	043440	020117	
8733	046500	047514	020127	051117	
8734	046506	042040	042517	047123	
8735	046514	052047	200		
8736	046517	107	052105	052040	.ASCIZ /GET THRU TO DAPE TV03/
8737	046524	051110	020125	047524	
8738	046532	042040	050101	020105	
8739	046540	053124	031460	000	
8740	046545	124	050122	030056	EM162: .ASCIZ /TRP.00 FAILED TO LOAD DR/

8741	046552	020060	040506	046111	
8742	046560	042105	052040	020117	
8743	046566	047514	042101	042040	
8744	046574	000122			
8745	046576	044502	020124	040506	EM163: .ASCIZ /BIT FAILED IN PIRQ REG/
8746	046604	046111	042105	044440	
8747	046612	020116	044520	050522	
8748	046620	051040	043505	000	
8749	046625	105	051122	051117	DH163: .ASCII /ERRORPC PIRQ TST NUM/<CRLF>
8750	046632	041520	020040	020040	
8751	046640	020040	050040	051111	
8752	046646	004521	020040	051524	
8753	046654	020124	052516	100115	
8754	046662	020011	054105	042520	.ASCIZ / EXPECT ACTUAL/
8755	046670	052103	020040	041501	
8756	046676	052524	046101	000	
8757		046704			.EVEN
8758	046704	001116	001162	001216	DT163: .WORD \$ERRPC,\$TMPO,\$EPIRQ,\$\$TSTNM,0
8759	046712	001210	000000		
8760	046716	042506	027124	030060	EM164: .ASCIZ /FET.00 HAD BAD BEN FIELD/
8761	046724	044040	042101	041040	
8762	046732	042101	041040	047105	
8763	046740	043040	042511	042114	
8764	046746	000			
8765	046747	124	041515	020102	EM165: .ASCIZ /TMCB E62(1) BAD OR TMCB HONOR PIR 1 NOT GOING LOW/
8766	046754	033105	024062	024461	
8767	046762	041040	042101	047440	
8768	046770	020122	046524	041103	
8769	046776	044040	047117	051117	
8770	047004	050040	051111	030440	
8771	047012	047040	052117	043440	
8772	047020	044517	043516	046040	
8773	047026	053517	000		
8774	047031	124	041515	020102	EM166: .ASCII /TMCB E51(9) OR E55(10,11) OR E62 BAD OR/<CRLF>
8775	047036	032505	024061	024471	
8776	047044	047440	020122	032505	
8777	047052	024065	030061	030454	
8778	047060	024461	047440	020122	
8779	047066	033105	020062	040502	
8780	047074	020104	051117	200	
8781	047101	124	041515	020101	.ASCIZ /TMCA INH BELOW BR6 STUCK LOW/
8782	047106	047111	020110	042502	
8783	047114	047514	020127	051102	
8784	047122	020066	052123	041525	
8785	047130	020113	047514	000127	
8786	047136	046524	040503	040440	EM167: .ASCIZ /TMCA ABOVE BR7 MIGHT BE STUCK LOW/
8787	047144	047502	042526	041040	
8788	047152	033522	046440	043511	
8789	047160	052110	041040	020105	
8790	047166	052123	041525	020113	
8791	047174	047514	000127		
8792	047200	046524	042503	041040	EM170: .ASCIZ /TMCE BRQ CLOCK MIGHT BE STUCK LOW/
8793	047206	050522	041440	047514	
8794	047214	045503	046440	043511	
8795	047222	052110	041040	020105	
8796	047230	052123	041525	020113	

8797	047236	047514	000127		
8798	047242	046524	041103	050040	EM171: .ASCII /TMCB PF(0)*(SF+TF) NOT GOING HIGH OR NOT/<CRLF>
8799	047250	024106	024460	024052	
8800	047256	043123	052053	024506	
8801	047264	047040	052117	043440	
8802	047272	044517	043516	044040	
8803	047300	043511	020110	051117	
8804	047306	047040	052117	200	
8805	047313	107	052105	044524	.ASCIZ /GETTING TO RACK E50 OR RACK E50(A1) BAD/
8806	047320	043516	052040	020117	
8807	047326	040522	045503	042440	
8808	047334	030065	047440	020122	
8809	047342	040522	045503	042440	
8810	047350	030065	040450	024461	
8811	047356	041040	042101	000	
8812	047363	124	041515	020102	EM172: .ASCII /TMCB PF(0)*(SF+-TF) NOT GOING LOW OR/<CRLF>
8813	047370	043120	030050	025051	
8814	047376	051450	025506	052055	/
8815	047404	024506	047040	052117	
8816	047412	043440	044517	043516	
8817	047420	046040	053517	047440	
8818	047426	100122			
8819	047430	047516	020124	042507	.ASCII /NOT GETTING TO RACK E64 OR RACK E64(A1) BAD/<CRLF>:
8820	047436	052124	047111	020107	
8821	047444	047524	051040	041501	
8822	047452	020113	033105	020064	
8823	047460	051117	051040	041501	
8824	047466	020113	033105	024064	
8825	047474	030501	020051	040502	
8826	047502	100104			
8827	047504	051117	052040	041515	.ASCII /OR TMCB PIRQ NOT GETTING TO DAPE OR DAPE TV05*07/<CRLF>
8828	047512	020102	044520	050522	
8829	047520	047040	052117	043440	
8830	047526	052105	044524	043516	
8831	047534	052040	020117	040504	
8832	047542	042520	047440	020122	
8833	047550	040504	042520	052040	
8834	047556	030126	025065	033460	
8835	047564	200			
8836	047565	116	052117	043440	.ASCIZ /NOT GOING HIGH OR NOT GETTING TO THE ALU/
8837	047572	044517	043516	044040	
8838	047600	043511	020110	051117	
8839	047606	047040	052117	043440	
8840	047614	052105	044524	043516	
8841	047622	052040	020117	044124	
8842	047630	020105	046101	000125	
8843	047636	046524	041103	042440	EM174: .ASCIZ /TMCB E62(2) BAD OR TMCB HONOR PIR 2 NOT GOING LOW/
8844	047644	031066	031050	020051	
8845	047652	040502	020104	051117	
8846	047660	052040	041515	020102	
8847	047666	047510	047516	020122	
8848	047674	044520	020122	020062	
8849	047702	047516	020124	047507	
8850	047710	047111	020107	047514	
8851	047716	000127			
8852	047720	046524	041103	042440	EM175: .ASCIZ /TMCB E63 BAD/

8853	047726	031466	041040	042101	
8854	047734	000			
8855	047735	114	053105	046105	EM176: .ASCIZ /LEVEL 2 INTERRUPT WHEN LEVEL 1 ON/
8856	047742	031040	044440	052116	
8857	047750	051105	052522	052120	
8858	047756	053440	042510	020116	
8859	047764	042514	042526	020114	
8860	047772	020061	047117	000	
8861	047777	124	041515	020102	EM177: .ASCIZ /TMCB E62(3) BAD OR TMCB HONOR PIR 3 NOT GOING LOW/
8862	050004	033105	024062	024463	
8863	050012	041040	042101	047440	
8864	050020	020122	046524	041103	
8865	050026	044040	047117	051117	
8866	050034	050040	051111	031440	
8867	050042	047040	052117	043440	
8868	050050	044517	043516	046040	
8869	050056	053517	000		
8870	050061	114	053105	046105	EM201: .ASCIZ /LEVEL 2 INTERRUPT WHEN CPU LEVEL 2 ON/
8871	050066	031040	044440	052116	
8872	050074	051105	052522	052120	
8873	050102	053440	042510	020116	
8874	050110	050103	020125	042514	
8875	050116	042526	020114	020062	
8876	050124	047117	000		
8877	050127	105	051122	051117	DH201: .ASCIZ /ERRORPC PIRQ TST NUM/
8878	050134	041520	020040	044520	
8879	050142	050522	020040	020040	
8880	050150	051524	020124	052516	
8881	050156	000115			
8882					
8883	050160	001116	001216	001210	DT201: .EVEN .WORD \$ERRPC,\$EPIRQ,\$\$TSTNM,0
8884	050166	000000			
8885	050170	046524	041103	042440	EM202: .ASCIZ /TMCB E62(5) BAD OR TMCA HONOR PIR 4 NOT GOING LOW/
8886	050176	031066	032450	020051	
8887	050204	040502	020104	051117	
8888	050212	052040	041515	020101	
8889	050220	047510	047516	020122	
8890	050226	044520	020122	020064	
8891	050234	047516	020124	047507	
8892	050242	047111	020107	047514	
8893	050250	000127			
8894	050252	042514	042526	020114	EM204: .ASCIZ /LEVEL 3 INTERRUPT WHEN CPU LEVEL 3 ON/
8895	050260	020063	047111	042524	
8896	050266	051122	050125	020124	
8897	050274	044127	047105	041440	
8898	050302	052520	046040	053105	
8899	050310	046105	031440	047440	
8900	050316	000116			
8901	050320	046524	041103	042440	EM205: .ASCIZ /TMCB E62(11) BAD OR TMCA HONOR PIR 5 NOT GOING LOW/
8902	050326	031066	030450	024461	
8903	050334	041040	042101	047440	
8904	050342	020122	046524	040503	
8905	050350	044040	047117	051117	
8906	050356	050040	051111	032440	
8907	050364	047040	052117	043440	
8908	050372	044517	043516	046040	

8909	050400	053517	000		
8910	050403	114	053105	046105	EM207: .ASCIZ /LEVEL 4 INTERRUPT WHEN CPU LEVEL 4 ON/
8911	050410	032040	044440	052116	
8912	050416	051105	052522	052120	
8913	050424	053440	042510	020116	
8914	050432	050103	020125	042514	
8915	050440	042526	020114	020064	
8916	050446	047117	000		
8917	050451	124	041515	020102	EM210: .ASCIZ /TMCB E51(11) BAD OR TMCB E55(8-9) BAD/
8918	050456	032505	024061	030461	
8919	050464	020051	040502	020104	
8920	050472	051117	052040	041515	
8921	050500	020102	032505	024065	
8922	050506	026470	024471	041040	
8923	050514	042101	000		
8924	050517	124	041515	020102	EM211: .ASCIZ /TMCB E70(1) BAD OR TMCA HONOR PIR6 NOT GOING LOW/
8925	050524	033505	024060	024461	
8926	050532	041040	042101	047440	
8927	050540	020122	046524	040503	
8928	050546	044040	047117	051117	
8929	050554	050040	051111	020066	
8930	050562	047516	020124	047507	
8931	050570	047111	020107	047514	
8932	050576	000127			
8933	050600	046524	041103	042440	EM212: .ASCIZ .TMCB E63(12) BAD OR TMCB E61(1) BAD/
8934	050606	031466	030450	024462	
8935	050614	041040	042101	047440	
8936	050622	020122	046524	041103	
8937	050630	042440	030466	030450	
8938	050636	020051	040502	000104	
8939	050644	042514	042526	020114	EM213: .ASCIZ /LEVEL 5 INTERRUPT WHEN CPU LEVEL 5 ON/
8940	050652	020065	047111	042524	
8941	050660	051122	050125	020124	
8942	050666	044127	047105	041440	
8943	050674	052520	046040	053105	
8944	050702	046105	032440	047440	
8945	050710	000116			
8946	050712	046524	041103	042440	EM214: .ASCIZ /TMCB E70(6) BAD OR TMCA HONOR PIR 7 NOT GOING LOW/
8947	050720	030067	033050	020051	
8948	050726	040502	020104	051117	
8949	050734	052040	041515	020101	
8950	050742	047510	047516	020122	
8951	050750	044520	020122	020067	
8952	050756	047516	020124	047507	
8953	050764	047111	020107	047514	
8954	050772	000127			
8955	050774	042514	042526	020114	EM216: .ASCIZ /LEVEL 6 INTERRUPT WHEN CPU LEVEL 6 ON/
8956	051002	020066	047111	042524	
8957	051010	051122	050125	020124	
8958	051016	044127	047105	041440	
8959	051024	052520	046040	053105	
8960	051032	046105	033040	047440	
8961	051040	000116			
8962	051042	044524	042515	052517	EM217: .ASCIZ /TIMEOUT ON DATI DIDN'T WORK/
8963	051050	020124	047117	042040	
8964	051056	052101	020111	044504	

8965	051064	047104	052047	053440	
8966	051072	051117	000113		
8967	051076	046524	041503	040440	EM220: .ASCII /TMCC AERF(1) L NOT GOING LOW/<CRLF>
8968	051104	051105	024106	024461	
8969	051112	046040	047040	052117	
8970	051120	043440	044517	043516	
8971	051126	046040	053517	200	
8972	051133	117	020122	046524	.ASCIZ /OR TMCB E53(11) BAD/
8973	051140	041103	042440	031465	
8974	051146	030450	024461	041040	
8975	051154	042101	000		
8976	051157	124	046511	047505	EM221: .ASCIZ /TIMEOUT ON DATO DIDN'T WORK/
8977	051164	052125	047440	020116	
8978	051172	040504	047524	042040	
8979	051200	042111	023516	020124	
8980	051206	047527	045522	000	
8981	051213	124	041515	020102	EM222: .ASCIZ /TMCB PS07(0) NOT GETTING TO TMCB E77 OR E77 BAD/
8982	051220	051520	033460	030050	
8983	051226	020051	047516	020124	
8984	051234	042507	052124	047111	
8985	051242	020107	047524	052040	
8986	051250	041515	020102	033505	
8987	051256	020067	051117	042440	
8988	051264	033467	041040	042101	
8989	051272	000			
8990	051273	102	032122	023040	EM223: .ASCIZ /BR4 & BR6 FAILED/
8991	051300	041040	033122	043040	
8992	051306	044501	042514	000104	
8993	051314	051102	020064	040506	EM224: .ASCII /BR4 FAILED. EITHER TMCB HONOR BR4 NOT GOING LOW OR/<CRLF>
8994	051322	046111	042105	020056	
8995	051330	044505	044124	051105	
8996	051336	052040	041515	020102	
8997	051344	047510	047516	020122	
8998	051352	051102	020064	047516	
8999	051360	020124	047507	047111	
9000	051366	020107	047514	020127	
9001	051374	051117	200		
9002	051377	124	041515	020102	.ASCIZ /TMCB E62(4) BAD OR INTERRUPT OR BG LOGIC ON UBC BAD/
9003	051404	033105	024062	024464	
9004	051412	041040	042101	047440	
9005	051420	020122	047111	042524	
9006	051426	051122	050125	020124	
9007	051434	051117	041040	020107	
9008	051442	047514	044507	020103	
9009	051450	047117	052440	041502	
9010	051456	041040	042101	000	
9011	051463	102	032122	043040	EM225: .ASCII /BR4 FAILED BR6 OK EITHER TMCB HONOR BR4 NOT GOING/<CRLF>
9012	051470	044501	042514	020104	
9013	051476	051102	020066	045517	
9014	051504	042440	052111	042510	
9015	051512	020122	046524	041103	
9016	051520	044040	047117	051117	
9017	051526	041040	032122	047040	
9018	051534	052117	043440	044517	
9019	051542	043516	200		
9020	051545	114	053517	047440	.ASCIZ /LOW OR TMCB E62(4) BAD/

9021	051552	020122	046524	041103
9022	051560	042440	031066	032050
9023	051566	020051	040502	000104
9024	051574	046524	040503	044040
9025	051602	047117	051117	041040
9026	051610	032522	047040	052117
9027	051616	043440	044517	043516
9028	051624	046040	053517	047440
9029	051632	020122	046524	041103
9030	051640	042440	031066	033050
9031	051646	020051	040502	000104
9032	051654	046524	040503	044040
9033	051662	047117	051117	041040
9034	051670	033122	047040	052117
9035	051676	043440	044517	043516
9036	051704	046040	053517	047440
9037	051712	020122	046524	041103
9038	051720	042440	031066	030450
9039	051726	024462	041040	042101
9040	051734	000		
9041	051735	131	046105	055040
9042	051742	047117	020105	040506
9043	051750	046111	042105	042440
9044	051756	052111	042510	020122
9045	051764	046524	042103	051440
9046	051772	020114	042531	020114
9047	052000	047516	020124	047507
9048	052006	047111	020107	044510
9049	052014	044107	047440	100122
9050	052022	051117	052040	041515
9051	052030	020101	047510	047516
9052	052036	020122	046123	020131
9053	052044	047516	020124	047507
9054	052052	047111	020107	047514
9055	052060	020127	051117	052040
9056	052066	041515	020102	033505
9057	052074	024060	024463	041040
9058	052102	042101	200	
9059	052105	117	020122	042502
9060	052112	030516	020063	040506
9061	052120	046111	042105	020055
9062	052126	044505	044124	051105
9063	052134	052040	041515	020101
9064	052142	047510	047516	020122
9065	052150	046123	020131	047516
9066	052156	020124	042507	052124
9067	052164	047111	100107	
9068	052170	047524	052040	041515
9069	052176	020102	032505	020063
9070	052204	051117	042440	031465
9071	052212	031450	020051	040502
9072	052220	000104		
9073	052222	046524	041503	042440
9074	052230	033061	034050	020051
9075	052236	047516	020124	047507
9076	052244	047111	020107	047514

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/

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/

9077	052252	020127	051117	052040	
9078	052260	041515	020103	031505	
9079	052266	020066	040502	000104	
9080	052274	042120	041522	051440	EM233: .ASCII /PDRC STACK LIMIT NOT GETTING TO TMCD AS A LOW OR/<CRLF>
9081	052302	040524	045503	046040	
9082	052310	046511	052111	047040	
9083	052316	052117	043440	052105	
9084	052324	044524	043516	052040	
9085	052332	020117	046524	042103	
9086	052340	040440	020123	020101	
9087	052346	047514	020127	051117	
9088	052354	200			
9089	052355	124	041515	020104	.ASCIZ /TMCD E8 BAD/
9090	052362	034105	041040	042101	
9091	052370	000			
9092	052371	125	041502	020103	EM234: .ASCII /UBCC DATI NOT GETTING TO TMCC AS A LOW OR EITHER/<CRLF>
9093	052376	040504	044524	047040	
9094	052404	052117	043440	052105	
9095	052412	044524	043516	052040	
9096	052420	020117	046524	041503	
9097	052426	040440	020123	020101	
9098	052434	047514	020127	051117	
9099	052442	042440	052111	042510	
9100	052450	100122			
9101	052452	046524	041503	042440	.ASCIZ /TMCC E30 OR E36 BAD/
9102	052460	030063	047440	020122	
9103	052466	031505	020066	040502	
9104	052474	000104			
9105	052476	046524	042103	051440	EM235: .ASCII /TMCD SL RED NOT GOING LOW OR TMCC ABORT/<CRLF>
9106	052504	020114	042522	020104	
9107	052512	047516	020124	047507	
9108	052520	047111	020107	047514	
9109	052526	020127	051117	052040	
9110	052534	041515	020103	041101	
9111	052542	051117	100124		
9112	052546	047516	020124	047507	.ASCII /NOT GOING LOW /
9113	052554	047111	020107	047514	
9114	052562	020127			
9115	052564	051117	052040	041515	.ASCIZ /OR TMCB E50(6) DIDN'T GO HIGH ON TMCC SERF(1)L/
9116	052572	020102	032505	024060	
9117	052600	024466	042040	042111	
9118	052606	023516	020124	047507	
9119	052614	044040	043511	020110	
9120	052622	047117	052040	041515	
9121	052630	020103	042523	043122	
9122	052636	030450	046051	000	
9123	052643	124	041515	020103	EM237: .ASCII /TMCC SERF(1) NOT GOING LOW OR/<CRLF>
9124	052650	042523	043122	030450	
9125	052656	020051	047516	020124	
9126	052664	047507	047111	020107	
9127	052672	047514	020127	051117	
9128	052700	200			
9129	052701	116	052117	043440	.ASCIZ /NOT GETTING TO TMCB E50(2&1)/
9130	052706	052105	044524	043516	
9131	052714	052040	020117	046524	
9132	052722	041103	042440	030065	

9133	052730	031050	030446	000051	
9134	052736	046524	041103	050040	EM240: .ASCII /TMCB PF(0)*(SF+-TF) NOT GOING HIGH OR/<CRLF>
9135	052744	024106	024460	024052	
9136	052752	043123	026453	043124	
9137	052760	020051	047516	020124	
9138	052766	047507	047111	020107	
9139	052774	044510	044107	047440	
9140	053002	100122			
9141	053004	047516	020124	042507	.ASCIZ /NOT GETTING TO RACK BRCAB04/
9142	053012	052124	047111	020107	
9143	053020	047524	051040	041501	
9144	053026	020113	051102	040503	
9145	053034	030102	000064		
9146	053040	041523	042503	051440	EM241: .ASCII /SCCE STACK OVERFLOW NOT GOING HIGH OR/<CRLF>
9147	053046	040524	045503	047440	
9148	053054	042526	043122	047514	
9149	053062	020127	047516	020124	
9150	053070	047507	047111	020107	
9151	053076	044510	044107	047440	
9152	053104	100122			
9153	053106	047516	020124	042507	.ASCIZ /NOT GETTING TO TMCD E31 OR E31 BAD/
9154	053114	052124	047111	020107	
9155	053122	047524	052040	041515	
9156	053130	020104	031505	020061	
9157	053136	051117	042440	030463	
9158	053144	041040	042101	000	
9159	053151	120	051104	020103	EM242: .ASCII /PDRG RED ZONE NOT GOING HIGH OR/<CRLF>
9160	053156	042522	020104	047532	
9161	053164	042516	047040	052117	
9162	053172	043440	044517	043516	
9163	053200	044040	043511	020110	
9164	053206	051117	200		
9165	053211	116	052117	043440	.ASCIZ /NOT GETTING TO TMCD E31 OR TMCD E31 BAD OR SL REG BIT 0 BAD/
9166	053216	052105	044524	043516	
9167	053224	052040	020117	046524	
9168	053232	042103	042440	030463	
9169	053240	047440	020122	046524	
9170	053246	042103	042440	030463	
9171	053254	041040	042101	047440	
9172	053262	020122	046123	051040	
9173	053270	043505	041040	052111	
9174	053276	030040	041040	042101	
9175	053304	000			
9176	053305	065	032062	030060	EM243: .ASCIZ /52400 PATTERN FAILED, 125000 PATTERN OK/
9177	053312	050040	052101	042524	
9178	053320	047122	043040	044501	
9179	053326	042514	026104	030440	
9180	053334	032462	030060	020060	
9181	053342	040520	052124	051105	
9182	053350	020116	045517	000	
9183	053355	105	051122	051117	DH243: .ASCII /ERRORPC SL REG TST NUM/<CRLF>
9184	053362	041520	020040	020040	
9185	053370	020040	046123	051040	
9186	053376	043505	020040	020040	
9187	053404	020040	051524	020124	
9188	053412	052516	100115		

Address	Hex 1	Hex 2	Hex 3	Hex 4	Hex 5	Text
9189	053416	020011	054105	042520		.ASCIZ / EXPECT ACTUAL/
9190	053424	052103	020040	041501		
9191	053432	052524	046101	000		
9192		053440				.EVEN
9193	053440	001116	001162	001222	DT243:	.WORD \$ERRPC,\$TMPO,E2STKLM,\$\$TSTNM,0
9194	053446	001210	000000			
9195	053452	031061	030065	030060	EM244:	.ASCIZ /125000 PATTERN FAILED 52400 PATTERN OK/
9196	053460	050040	052101	042524		
9197	053466	047122	043040	044501		
9198	053474	042514	020104	031065		
9199	053502	030064	020060	040520		
9200	053510	052124	051105	020116		
9201	053516	045517	000			
9202		053522				.EVEN
9203	053522	001116	001162	001220	DT244:	.WORD \$ERRPC,\$TMPO,E1STKLM,\$\$TSTNM,0
9204	053530	001210	000000			
9205	053534	041523	042503	051440	EM245:	.ASCII /SCCE SL ADDRESS NOT GETTING TO TMCD OR/<CRLF>
9206	053542	020114	042101	051104		
9207	053550	051505	020123	047516		
9208	053556	020124	042507	052124		
9209	053564	047111	020107	047524		
9210	053572	052040	041515	020104		
9211	053600	051117	200			
9212	053603	124	041515	020104		.ASCIZ /TMCD E28 OR E14 BAD/
9213	053610	031105	020070	051117		
9214	053616	042440	032061	041040		
9215	053624	042101	000			
9216	053627	124	041515	020104	EM246:	.ASCII /TMCD LOW BYTE EN DOESN'T GO LOW OR/<CRLF>
9217	053634	047514	020127	054502		
9218	053642	042524	042440	020116		
9219	053650	047504	051505	023516		
9220	053656	020124	047507	046040		
9221	053664	053517	047440	100122		
9222	053672	047516	020124	042507		.ASCIZ /NOT GETTING THRU TO THE DMUX (PDRE) AS A LOW /
9223	053700	052124	047111	020107		
9224	053706	044124	052522	052040		
9225	053714	020117	044124	020105		
9226	053722	046504	054125	024040		
9227	053730	042120	042522	020051		
9228	053736	051501	040440	046040		
9229	053744	053517	000040			
9230	053750	046524	042103	042040	EM247:	.ASCII /TMCD DMX S1 STUCK HIGH OR IT DOESN'T/<CRLF>
9231	053756	054115	051440	020061		
9232	053764	052123	041525	020113		
9233	053772	044510	044107	047440		
9234	054000	020122	052111	042040		
9235	054006	042517	047123	052047		
9236	054014	200				
9237	054015	107	052105	052040		.ASCIZ /GET THRU TO THE DMUX(PDRE) AS A LOW/
9238	054022	051110	020125	047524		
9239	054030	052040	042510	042040		
9240	054036	052515	024130	042120		
9241	054044	042522	020051	051501		
9242	054052	040440	046040	053517		
9243	054060	000				
9244	054061	102	052117	020110	EM250:	.ASCIZ /BOTH PATTERNS FAILED/

9245	054066	040520	052124	051105	
9246	054074	051516	043040	044501	
9247	054102	042514	000104		
9248	054106	051105	047522	050122	DH250: .ASCII /ERRORPC SL REG SL REG TST NUM/<CRLF>
9249	054114	020103	020040	020040	
9250	054122	051440	020114	042522	
9251	054130	004507	020040	020040	
9252	054136	020040	046123	051040	
9253	054144	043505	020040	020040	
9254	054152	052040	052123	047040	
9255	054160	046525	200		
9256	054163	011	042440	050130	.ASCIZ / EXPECT ACTUAL EXPECT ACTUAL/
9257	054170	041505	020124	040440	
9258	054176	052103	040525	020114	
9259	054204	042440	050130	041505	
9260	054212	020124	040440	052103	
9261	054220	040525	000114		
9262					
9263	054224	001116	001162	001220	DT250: .EVEN \$ERRPC,\$TMPO,E1STKLM,\$TMP1,E2STKLM,\$\$TSTNM,0
9264	054232	001164	001222	001210	
9265	054240	000000			
9266	054242	046524	042103	054440	EM251: .ASCII /TMCD YEL ZONE DIDN'T GO LOW OR IT DIDN'T GET THRU TO E31/<CRLF>
9267	054250	046105	055040	047117	
9268	054256	020105	044504	047104	
9269	054264	052047	043440	020117	
9270	054272	047514	020127	051117	
9271	054300	044440	020124	044504	
9272	054306	047104	052047	043440	
9273	054314	052105	052040	051110	
9274	054322	020125	047524	042440	
9275	054330	030463	200		
9276	054333	117	020122	046524	.ASCIZ /OR TMCE CACHE BEND DIDN'T GO HIGH ON SL RED/
9277	054340	042503	041440	041501	
9278	054346	042510	041040	047105	
9279	054354	020104	044504	047104	
9280	054362	052047	043440	020117	
9281	054370	044510	044107	047440	
9282	054376	020116	046123	051040	
9283	054404	042105	000		
9284	054407	124	041515	020104	EM252: .ASCIZ /TMCD YEL ZONE DOESN'T GO LOW ON ADR 240/
9285	054414	042531	020114	047532	
9286	054422	042516	042040	042517	
9287	054430	047123	052047	043440	
9288	054436	020117	047514	020127	
9289	054444	047117	040440	051104	
9290	054452	031040	030064	000	
9291	054457	124	041515	020104	EM253: .ASCIZ /TMCD YEL ZONE DOESN'T GO LOW ON ADR 140/
9292	054464	042531	020114	047532	
9293	054472	042516	042040	042517	
9294	054500	047123	052047	043440	
9295	054506	020117	047514	020127	
9296	054514	047117	040440	051104	
9297	054522	030440	030064	000	
9298	054527	124	041515	020104	EM254: .ASCIZ /TMCD YEL ZONE DOESN'T GO HIGH OR DIDN'T GET THRU TO E31/
9299	054534	042531	020114	047532	
9300	054542	042516	042040	042517	

9301	054550	047123	052047	043440	
9302	054556	020117	044510	044107	
9303	054564	047440	020122	044504	
9304	054572	047104	052047	043440	
9305	054600	052105	052040	051110	
9306	054606	020125	047524	042440	
9307	054614	030463	000		
9308	054617	125	044516	052502	EM255: .ASCIZ /UNIBUS TIMEOUT BIT IN CPU ERROR REG DIDN'T SET/
9309	054624	020123	044524	042515	
9310	054632	052517	020124	044502	
9311	054640	020124	047111	041440	
9312	054646	052520	042440	051122	
9313	054654	051117	051040	043505	
9314	054662	042040	042111	023516	
9315	054670	020124	042523	000124	
9316	054676	051105	047522	050122	DH255: .ASCII /ERRORPC CPUERR REG TST NUM/<CRLF>
9317	054704	020103	020J40	041440	
9318	054712	052520	051105	020122	
9319	054720	042522	020107	020040	
9320	054726	051524	020124	052516	
9321	054734	100115			
9322	054736	020011	054105	042520	.ASCIZ / EXPECT ACTUAL/
9323	054744	052103	020040	041501	
9324	054752	052524	046101	000	
9325	054757	103	052520	042440	EM256: .ASCIZ /CPU ERRPROR REG DIDN'T CLEAR/
9326	054764	051122	051120	051117	
9327	054772	051040	043505	042040	
9328	055000	042111	023516	020124	
9329	055006	046103	040505	000122	
9330	055014	042531	020114	047532	EM257: .ASCIZ /YEL ZONE BIT IN CPU ERROR REG DIDN'T SET/
9331	055022	042516	041040	052111	
9332	055030	044440	020116	050103	
9333	055036	020125	051105	047522	
9334	055044	020122	042522	020107	
9335	055052	044504	047104	052047	
9336	055060	051440	052105	000	
9337	055065	124	041515	020104	EM260: .ASCIZ /TMCD E4(4) NOT GOING LOW OR E4 BAD/
9338	055072	032105	032050	020051	
9339	055100	047516	020124	047507	
9340	055106	047111	020107	047514	
9341	055114	020127	051117	042440	
9342	055122	020064	040502	000104	
9343	055130	042522	042101	055040	EM261: .ASCIZ /READ ZONE BIT IN CPU ERROR REG DIDN'T SET/
9344	055136	047117	020105	044502	
9345	055144	020124	047111	041440	
9346	055152	052520	042440	051122	
9347	055160	051117	051040	043505	
9348	055166	042040	042111	023516	
9349	055174	020124	042523	000124	
9350	055202	046524	042103	042440	EM262: .ASCIZ /TMCD E15(6) OR E18(14) OR E18 BAD/
9351	055210	032461	033050	020051	
9352	055216	051117	042440	034061	
9353	055224	030450	024464	047440	
9354	055232	020122	030505	020070	
9355	055240	040502	000104		
9356	055244	047506	046114	053517	EM263: .ASCII /FOLLOWING IS A LIST OF THE STACK LIMIT REG/<CRLF>

9357 055252 047111 020107 051511
9358 055260 040440 046040 051511
9359 055266 020124 043117 052040
9360 055274 042510 051440 040524
9361 055302 045503 046040 046511
9362 055310 052111 051040 043505
9363 055316 200
9364 055317 046 051440 020120
9365 055324 040526 052514 051505
9366 055332 052040 040510 020124
9367 055340 040503 051525 042105
9368 055346 040440 020116 051105
9369 055354 047522 027122 052040
9370 055362 042510 020131 051101
9371 055370 100105
9372 055372 051107 052517 042520
9373 055400 020104 041501 047503
9374 055406 042122 047111 020107
9375 055414 047524 042440 051122
9376 055422 051117 052040 050131
9377 055430 051505 000
9378 055433 105 051122 051117
9379 055440 041520 052040 051505
9380 055446 020124 052516 041115
9381 055454 051105 000200
9382 055460 051411 040524 045503
9383 055466 046040 046511 052111
9384 055474 051040 043505 051511
9385 055502 042524 004522 020011
9386 055510 020040 020040 020040
9387 055516 052123 041501 020113
9388 055524 047520 047111 042524
9389 055532 100122
9390 055534 041517 040524 020114
9391 055542 020040 030440 020065
9392 055550 032061 030440 020063
9393 055556 031061 030440 020061
9394 055564 030061 020040 020071
9395 055572 034040 020040 020040
9396 055600 020040 041517 040524
9397 055606 114
9398 055607 040 020040 030440
9399 055614 020065 032061 030440
9400 055622 020063 031061 030440
9401 055630 020061 030061 020040
9402 055636 020071 034040 000200
9403 055644 042522 020104 051124
9404 055652 050101 047440 020116
9405 055660 042531 020114 042101
9406 055666 000122
9407 055670 042522 020104 051124
9408 055676 050101 047440 020116
9409 055704 042514 040507 020114
9410 055712 042101 000122
9411 055716 042531 046114 053517
9412 055724 052040 040522 020120

.ASCII /& SP VALUES THAT CAUSED AN ERROR. THEY ARE/<CRLF>
.ASCIZ /GROUPED ACCORDING TO ERROR TYPES/
DH263: .ASCIZ /ERRORPC TEST NUMBER/<CRLF>
DH263A: .ASCII / STACK LIMIT REGISTER STACK POINTER/<CRLF>
.ASCII /OCTAL 15 14 13 12 11 10 9 8 OCTAL/
.ASCIZ / 15 14 13 12 11 10 9 8/<CRLF>
DH263B: .ASCIZ /RED TRAP ON YEL ADR/
DH263C: .ASCIZ /RED TRAP ON LEGAL ADR/
DH263D: .ASCIZ /YELLOW TRAP ON RED ADR/

9413	055732	047117	051040	042105	
9414	055740	040440	051104	000	
9415	055745	131	046105	052040	DH263E: .ASCIZ /YEL TRAP ON LEGAL ADR/
9416	055752	040522	020120	047117	
9417	055760	046040	043505	046101	
9418	055766	040440	051104	000	
9419	055773	116	020117	051124	DH263F: .ASCIZ /NO TRAP ON RED ADR/
9420	056000	050101	047440	020116	
9421	056006	042522	020104	042101	
9422	056014	000122			
9423	056016	047516	052040	040522	DH263G: .ASCIZ /NO TRAP ON YEL ADR/
9424	056024	020120	047117	054440	
9425	056032	046105	040440	051104	
9426	056040	000			
9427		056042			
9428	056042	055644			INDEX: .EVEN
9429	056044	055670			DH263B
9430	056046	055716			DH263C
9431	056050	055745			DH263D
9432	056052	055773			DH263E
9433	056054	056016			DH263F
9434	056056	047507	047111	020107	DH263G
9435	056064	047524	047040	054105	EM264: .ASCIZ /GOING TO NEXT TEST/
9436	056072	020124	042524	052123	
9437	056100	000			
9438					
9439	056101	116	052117	035105	EM265: .ASCII /NOTE: IF NONE OF THE ODD ADR ERRORS TRAP/<CRLF>
9440	056106	044440	020106	047516	
9441	056114	042516	047440	020106	
9442	056122	044124	020105	042117	
9443	056130	020104	042101	020122	
9444	056136	051105	047522	051522	
9445	056144	052040	040522	100120	
9446	056152	020040	020040	020040	.ASCII / THEN EITHER TMCC ODD ADRS ERR NOT GETTING TO/<CRLF>
9447	056160	044124	047105	042440	
9448	056166	052111	042510	020122	
9449	056174	046524	041503	047440	
9450	056202	042104	040440	051104	
9451	056210	020123	051105	020122	
9452	056216	047516	020124	042507	
9453	056224	052124	047111	020107	
9454	056232	047524	200		
9455	056235	040	020040	020040	.ASCII / TMCC BUS ERROR OR DAPB BAMX00 NOT/<CRLF>
9456	056242	052040	041515	020103	
9457	056250	052502	020123	051105	
9458	056256	047522	020122	051117	
9459	056264	042040	050101	020102	
9460	056272	040502	054115	030060	
9461	056300	047040	052117	200	
9462	056305	040	020040	020040	.ASCII / GETTING TO TMCC E7 AS A HIGH/<CRLF><CRLF>
9463	056312	043440	052105	044524	
9464	056320	043516	052040	020117	
9465	056326	046524	041503	042440	
9466	056334	020067	051501	040440	
9467	056342	044040	043511	100110	
9468	056350	200			

9469	056351	116	044505	044124	.ASCIZ /NEITHER -BYIN NOR DATI CAUSED A TRAP/<CRLF>
9470	056356	051105	026440	054502	
9471	056364	047111	047040	051117	
9472	056372	042040	052101	020111	
9473	056400	040503	051525	042105	
9474	056406	040440	052040	040522	
9475	056414	100120	000		
9476	056417	111	041522	020104	EM266: .ASCII /IRCD BYIN DOESN'T GET TO TMCC E12 AS A HIGH/<CRLF>
9477	056424	054502	047111	042040	
9478	056432	042517	047123	052047	
9479	056440	043440	052105	052040	
9480	056446	020117	046524	041503	
9481	056454	042440	031061	040440	
9482	056462	020123	020101	044510	
9483	056470	044107	200		
9484	056473	117	020122	046524	.ASCIZ /OR TMCC E12(9,8) BAD OR E7(2) BAD/
9485	056500	041503	042440	031061	
9486	056506	034450	034054	020051	
9487	056514	040502	020104	051117	
9488	056522	042440	024067	024462	
9489	056530	041040	042101	000	
9490	056535	104	050101	020102	EM267: .ASCIZ /DAPB BAMX00 DOESN'T GET TO TMCC E7(4) OR E7 BAD/
9491	056542	040502	054115	030060	
9492	056550	042040	042517	047123	
9493	056556	052047	043440	052105	
9494	056564	052040	020117	046524	
9495	056572	041503	042440	024067	
9496	056600	024464	047440	020122	
9497	056606	033505	041040	042101	
9498	056614	000			
9499	056615	122	041501	020103	EM270: .ASCII /RACC UBSC00 DOESN'T GET TO TMCC E5(13) AS/<CRLF>
9500	056622	041125	041523	030060	
9501	056630	042040	042517	047123	
9502	056636	052047	043440	052105	
9503	056644	052040	020117	046524	
9504	056652	041503	042440	024065	
9505	056660	031461	020051	051501	
9506	056666	200			
9507	056667	101	046040	053517	.ASCIZ /A LOW OR E5(13) BAD/
9508	056674	047440	020122	032505	
9509	056702	030450	024463	041040	
9510	056710	042101	000		
9511	056713	122	041501	020103	EM271: .ASCII /RACC UBSC02 DOESN'T GET TO TMCC E12(5) AS/<CRLF>
9512	056720	041125	041523	031060	
9513	056726	042040	042517	047123	
9514	056734	052047	043440	052105	
9515	056742	052040	020117	046524	
9516	056750	041503	042440	031061	
9517	056756	032450	020051	051501	
9518	056764	200			
9519	056765	101	044040	043511	.ASCIZ /A HIGH OR E12(6) DOESN'T GO LOW OR E5(12) BAD/
9520	056772	020110	051117	042440	
9521	057000	031061	033050	020051	
9522	057006	047504	051505	023516	
9523	057014	020124	047507	046040	
9524	057022	053517	047440	020122	

9525	057030	032505	030450	024462	
9526	057036	041040	042101	000	
9527	057043	123	031515	033465	EM272: .ASCIZ /SM357*SRC1 DATI FAILED TO TRAP/
9528	057050	051452	041522	020061	
9529	057056	040504	044524	043040	
9530	057064	044501	042514	020104	
9531	057072	047524	052040	040522	
9532	057100	000120			
9533	057102	042117	020104	042101	EM273: .ASCIZ /ODD ADR BIT IN CPUERR REG DOESN'T SET/
9534	057110	020122	044502	020124	
9535	057116	047111	041440	052520	
9536	057124	051105	020122	042522	
9537	057132	020107	047504	051505	
9538	057140	023516	020124	042523	
9539	057146	000124			
9540	057150	047516	052040	040522	EM274: .ASCIZ /NO TRAP ON DATO/
9541	057156	020120	047117	042040	
9542	057164	052101	000117		
9543	057170	051520	032060	030450	EM275: .ASCII /PS04(1) DOESN'T GET TO TMCB E74(9) AS A HIGH/<CRLF>
9544	057176	020051	047504	051505	
9545	057204	023516	020124	042507	
9546	057212	020124	047524	052040	
9547	057220	041515	020102	033505	
9548	057226	024064	024471	040440	
9549	057234	020123	020101	044510	
9550	057242	044107	200		
9551	057245	117	020122	052111	.ASCII /OR IT DOESN'T GET TO E51(10) AS A LOW OR E51 BAD/<CRLF>
9552	057252	042040	042517	047123	
9553	057260	052047	043440	052105	
9554	057266	052040	020117	032505	
9555	057274	024061	030061	020051	
9556	057302	051501	040440	046040	
9557	057310	053517	047440	020122	
9558	057316	032505	020061	040502	
9559	057324	100104			
9560	057326	051117	044440	041522	.ASCII /OR IRCD RTT DOESN'T GET TO TMCB E74(11) AS A HIGH/<CRLF>
9561	057334	020104	052122	020124	
9562	057342	047504	051505	023516	
9563	057350	020124	042507	020124	
9564	057356	047524	052040	041515	
9565	057364	020102	033505	024064	
9566	057372	030461	020051	051501	
9567	057400	040440	044040	043511	
9568	057406	100110			
9569	057410	051117	052040	041515	.ASCII /OR TMCB HONOR T DIDN'T GO LOW OR TMCB TOK DID/<CRLF>
9570	057416	020102	047510	047516	
9571	057424	020122	020124	044504	
9572	057432	047104	052047	043440	
9573	057440	020117	047514	020127	
9574	057446	051117	052040	041515	
9575	057454	020102	047524	020113	
9576	057462	044504	100104		
9577	057466	047516	020124	047507	.ASCIZ /NOT GO LOW OR IT DIDN'T GET THRU TMCB E53/
9578	057474	046040	053517	047440	
9579	057502	020122	052111	042040	
9580	057510	042111	023516	020124	

9581	057516	042507	020124	044124	
9582	057524	052522	052040	041515	
9583	057532	020102	032505	000063	
9584	057540	044502	020124	020064	EM276: .ASCIZ /BIT 4 IN PSW DOESN'T SET/
9585	057546	047111	050040	053523	
9586	057554	042040	042517	047123	
9587	057562	052047	051440	052105	
9588	057570	000			
9589	057571	123	052105	044524	EM277: .ASCIZ /SETTING PS<08> FAILED TO DISABLE T BIT TRAP/
9590	057576	043516	050040	036123	
9591	057604	034060	020076	040506	
9592	057612	046111	042105	052040	
9593	057620	020117	044504	040523	
9594	057626	046102	020105	020124	
9595	057634	044502	020124	051124	
9596	057642	050101	000		
9597	057645	124	041515	020102	EM300: .ASCIZ /TMCB TOK DOESN'T GET TO DAPE E7(11) AS A LOW OR E7 BAD/
9598	057652	047524	020113	047504	
9599	057660	051505	023516	020124	
9600	057666	042507	020124	047524	
9601	057674	042040	050101	020105	
9602	057702	033505	030450	024461	
9603	057710	040440	020123	020101	
9604	057716	047514	020127	051117	
9605	057724	042440	020067	040502	
9606	057732	000104			
9607	057734	051111	042103	051040	EM301: .ASCII /IRCD RTT DOESN'T GET TO TMCB E74 AS A LOW/<CRLF>
9608	057742	052124	042040	042517	
9609	057750	047123	052047	043440	
9610	057756	052105	052040	020117	
9611	057764	046524	041103	042440	
9612	057772	032067	040440	020123	
9613	060000	020101	047514	100127	
9614	060006	051117	042440	032067	.ASCIZ /OR E74 BAD/
9615	060014	041040	042101	000	
9616	060021	124	041515	020102	EM302: .ASCIZ /TMCB E58(5) DIDN'T GO LOW OR E64 BAD/
9617	060026	032505	024070	024465	
9618	060034	042040	042111	023516	
9619	060042	020124	047507	046040	
9620	060050	053517	047440	020122	
9621	060056	033105	020064	040502	
9622	060064	000104			
9623	060066	046524	040503	040440	EM303: .ASCII /TMCA ABOVE BR7 DOESN'T GO LOW ON A YEL ZONE OR/<CRLF>
9624	060074	047502	042526	041040	
9625	060102	033522	042040	042517	
9626	060110	047123	052047	043440	
9627	060116	020117	047514	020127	
9628	060124	047117	040440	054440	
9629	060132	046105	055040	047117	
9630	060140	020105	051117	200	
9631	060145	111	020124	047504	.ASCIZ /IT DOESN'T GET TO TMCB E84(13) OR E84 BAD/
9632	060152	051505	023516	020124	
9633	060160	042507	020124	047524	
9634	060166	052040	041515	020102	
9635	060174	034105	024064	031461	
9636	060202	020051	051117	042440	

9637	060210	032070	041040	042101	
9638	060216	000			
9639	060217	124	041515	020101	EM304: .ASCIZ /TMCA ABOVE BR7 DOESN'T GET TO TMCB E84(1) OR E84 BAD/
9640	060224	041101	053117	020105	
9641	060232	051102	020067	047504	
9642	060240	051505	023516	020124	
9643	060246	042507	020124	047524	
9644	060254	052040	041515	020102	
9645	060262	034105	024064	024461	
9646	060270	047440	020122	034105	
9647	060276	020064	040502	000104	
9648	060304	046524	040503	040440	EM305: .ASCIZ /TMCA ABOVE BR7 DOESN'T GET TO TMCB E77(13) OR E77 BAD/
9649	060312	047502	042526	041040	
9650	060320	033522	042040	042517	
9651	060326	047123	052047	043440	
9652	060334	052105	052040	020117	
9653	060342	046524	041103	042440	
9654	060350	033467	030450	024463	
9655	060356	047440	020122	033505	
9656	060364	020067	040502	000104	
9657	060372	046524	040503	041040	EM306: .ASCII /TMCA BLOCK BR4 DOESN'T GO LOW ON PIR4 OR/<CRLF>
9658	060400	047514	045503	041040	
9659	060406	032122	042040	042517	
9660	060414	047123	052047	043440	
9661	060422	020117	047514	020127	
9662	060430	047117	050040	051111	
9663	060436	020064	051117	200	
9664	060443	111	020124	047504	.ASCIZ /IT DOESN'T GET TO TMCB E77(5) OR E77 BAD/
9665	060450	051505	023516	020124	
9666	060456	042507	020124	047524	
9667	060464	052040	041515	020102	
9668	060472	033505	024067	024465	
9669	060500	047440	020122	033505	
9670	060506	020067	040502	000104	
9671	060514	046524	040503	044440	EM307: .ASCIZ /TMCA INH BELOW PIR4 DOESN'T GO LOW ON PIR5/
9672	060522	044116	041040	046105	
9673	060530	053517	050040	051111	
9674	060536	020064	047504	051505	
9675	060544	023516	020124	047507	
9676	060552	046040	053517	047440	
9677	060560	020116	044520	032522	
9678	060566	000			
9679	060567	124	041515	020101	EM310: .ASCIZ /TMCA INH BELOW PIR4 DOESN'T GO LOW ON BR5/
9680	060574	047111	020110	042502	
9681	060602	047514	020127	044520	
9682	060610	032122	042040	042517	
9683	060616	047123	052047	043440	
9684	060624	020117	047514	020127	
9685	060632	047117	041040	032522	
9686	060640	000			
9687	060641	124	041515	020101	EM311: .ASCIZ /TMCA BLOCK LEVEL 4 DIDN'T GO LOW ON PIR6/
9688	060646	046102	041517	020113	
9689	060654	042514	042526	020114	
9690	060662	020064	044504	047104	
9691	060670	052047	043440	020117	
9692	060676	047514	020127	047117	

9693	060704	050040	051111	000066	
9694	060712	046524	040503	040440	EM312: .ASCII /TMCA ABOVE BR7 DOESN'T GO LOW(ON PIR 7) OR/<CRLF>
9695	060720	047502	042526	041040	
9696	060726	033522	042040	042517	
9697	060734	047123	052047	043440	
9698	060742	020117	047514	024127	
9699	060750	047117	050040	051111	
9700	060756	033440	020051	051117	
9701	060764	200			
9702	060765	111	020124	047504	.ASCIZ /IT DOESN'T GET TO TMCB E77(1) OR E77 BAD/
9703	060772	051505	023516	020124	
9704	061000	042507	020124	047524	
9705	061006	052040	041515	020102	
9706	061014	033505	024067	024461	
9707	061022	047440	020122	033505	
9708	061030	020067	040502	000104	
9709	061036	041125	042103	042440	EM313: .ASCIZ /UBCD EXT BRQ DIDN'T GO LOW ON HONOR BR5/
9710	061044	052130	041040	050522	
9711	061052	042040	042111	023516	
9712	061060	020124	047507	046040	
9713	061066	053517	047440	020116	
9714	061074	047510	047516	020122	
9715	061102	051102	000065		
9716	061106	047111	020110	042502	EM314: .ASCII /INH BELOW BR6 DOESN'T GO LOW(ON BR6) OR IT DOES/<CRLF>
9717	061114	047514	020127	051102	
9718	061122	020066	047504	051505	
9719	061130	023516	020124	047507	
9720	061136	046040	053517	047450	
9721	061144	020116	051102	024466	
9722	061152	047440	020122	052111	
9723	061160	042040	042517	100123	
9724	061166	047516	020124	042507	.ASCIZ /NOT GET THRU TMCA E82(6)/
9725	061174	020124	044124	052522	
9726	061202	052040	041515	020101	
9727	061210	034105	024062	024466	
9728	061216	000			
9729	061217	124	041515	020101	EM315: .ASCII /TMCA E67(8) DOESN'T GO LOW (ON SL YEL) OR IT IS NOT/<CRLF>
9730	061224	033105	024067	024470	
9731	061232	042040	042517	047123	
9732	061240	052047	043440	020117	
9733	061246	047514	020127	047450	
9734	061254	020116	046123	054440	
9735	061262	046105	020051	051117	
9736	061270	044440	020124	051511	
9737	061276	047040	052117	200	
9738	061303	107	052105	044524	.ASCIZ /GETTING TO E60(12) OR E60 BAD/
9739	061310	043516	052040	020117	
9740	061316	033105	024060	031061	
9741	061324	020051	051117	042440	
9742	061332	030066	041040	042101	
9743	061340	000			
9744	061341	124	041515	020101	EM316: .ASCIZ /TMCA E81(12) DOESN'T GO LOW(ON PIR5) OR IT DOES/<CRLF>
9745	061346	034105	024061	031061	
9746	061354	020051	047504	051505	
9747	061362	023516	020124	047507	
9748	061370	046040	053517	047450	

9749	061376	020116	044520	032522	
9750	061404	020051	051117	044440	
9751	061412	020124	047504	051505	
9752	061420	000200			
9753	061422	042507	020124	047524	.ASCIZ /GET TO E83(10) OR E83 BAD/
9754	061430	042440	031470	030450	
9755	061436	024460	047440	020122	
9756	061444	034105	020063	040502	
9757	061452	000104			
9758	061454	046524	040503	042440	EM317: .ASCIZ /TMCA E81(2) BAD/
9759	061462	030470	031050	020051	
9760	061470	040502	000104		
9761	061474	046524	040503	042440	EM320: .ASCII /TMCA E67(8) DOESN'T GO LOW(ON PIR 7) OR IT DOES/<CRLF>
9762	061502	033466	034050	020051	
9763	061510	047504	051505	023516	
9764	061516	020124	047507	046040	
9765	061524	053517	047450	020116	
9766	061532	044520	020122	024467	
9767	061540	047440	020122	052111	
9768	061546	042040	042517	100123	
9769	061554	047516	020124	042507	.ASCIZ /NOT GET TO E83(13) OR E83 BAD/
9770	061562	020124	047524	042440	
9771	061570	031470	030450	024463	
9772	061576	047440	020122	034105	
9773	061604	020063	040502	000104	
9774	061612	041125	042103	042440	EM321: .ASCIZ /UBCD EXT BRQ DIDN'T GO LOW ON HONOR BR6/
9775	061620	052130	041040	050522	
9776	061626	042040	042111	023516	
9777	061634	020124	047507	046040	
9778	061642	053517	047440	020116	
9779	061650	047510	047516	020122	
9780	061656	051102	000066		
9781	061662	046524	040503	042440	EM322: .ASCIZ /TMCA E67(8) DOESN'T GET TO E83(1) OR E83 BAD/
9782	061670	033466	034050	020051	
9783	061676	047504	051505	023516	
9784	061704	020124	042507	020124	
9785	061712	047524	042440	031470	
9786	061720	030450	020051	051117	
9787	061726	042440	031470	041040	
9788	061734	042101	000		
9789	061737	124	041515	020101	EM323: .ASCII /TMCA E81(8) IS NOT GOING LOW OR IT IS NOT/<CRLF>
9790	061744	034105	024061	024470	
9791	061752	044440	020123	047516	
9792	061760	020124	047507	047111	
9793	061766	020107	047514	020127	
9794	061774	051117	044440	020124	
9795	062002	051511	047040	052117	
9796	062010	200			
9797	062011	107	052105	044524	.ASCIZ /GETTING TO E76(12) OR E76 BAD/
9798	062016	043516	052040	020117	
9799	062024	033505	024066	031061	
9800	062032	020051	051117	042440	
9801	062040	033067	041040	042101	
9802	062046	000			
9803	062047	124	041515	020101	EM324: .ASCIZ /TMCA E67(8) NOT GETTING TO E76(13) OR E76 IS BAD/
9804	062054	033105	024067	024470	

9805	062062	047040	052117	043440	
9806	062070	052105	044524	043516	
9807	062076	052040	020117	033505	
9808	062104	024066	031461	020051	
9809	062112	051117	042440	033067	
9810	062120	044440	020123	040502	
9811	062126	000104			
9812	062130	046524	040503	042440	EM325: .ASCIZ /TMCA E67(8) DOESN'T GET TO E76(1) OR E76 BAD/
9813	062136	033466	034050	020051	
9814	062144	047504	051505	023516	
9815	062152	020124	042507	020124	
9816	062160	047524	042440	033067	
9817	062166	030450	020051	051117	
9818	062174	042440	033067	041040	
9819	062202	042101	000		
9820	062205	124	041515	020101	EM326: .ASCIZ /TMCA E67(6) NOT GETTING TO E69(12) OR E69 IS BAD/
9821	062212	033105	024067	024466	
9822	062220	047040	052117	043440	
9823	062226	052105	044524	043516	
9824	062234	052040	020117	033105	
9825	062242	024071	031061	020051	
9826	062250	051117	042440	034466	
9827	062256	044440	020123	040502	
9828	062264	000104			
9829	062266	051520	020127	044502	EM327: .ASCII /PSW BIT 11 DOESN'T SET OR TMCF CLK HI PS DOES/<CRLF>
9830	062274	020124	030461	042040	
9831	062302	042517	047123	052047	
9832	062310	051440	052105	047440	
9833	062316	020122	046524	043103	
9834	062324	041440	045514	044040	
9835	062332	020111	051520	042040	
9836	062340	042517	100123		
9837	062344	047516	020124	047507	.ASCIZ /NOT GO LOW OR BIT 11 DOESN'T GET TO OR THRU THE DMUX/
9838	062352	046040	053517	047440	
9839	062360	020122	044502	020124	
9840	062366	030461	042040	042517	
9841	062374	047123	052047	043440	
9842	062402	052105	052040	020117	
9843	062410	051117	052040	051110	
9844	062416	020125	044124	020105	
9845	062424	046504	054125	000	
9846	062431	107	040522	020103	EM330: .ASCII /GRAC GDREG SET 1 DOESN'T GO HIGH (ON PAD 5) OR/<CRLF>
9847	062436	042107	042522	020107	
9848	062444	042523	020124	020061	
9849	062452	047504	051505	023516	
9850	062460	020124	047507	044040	
9851	062466	043511	020110	047450	
9852	062474	020116	040520	020104	
9853	062502	024465	047440	100122	
9854	062510	047504	051505	023516	.ASCIZ /DOESN'T GET TO THE GD REG/
9855	062516	020124	042507	020124	
9856	062524	047524	052040	042510	
9857	062532	043440	020104	042522	
9858	062540	000107			
9859	062542	051107	041501	043440	EM331: .ASCII /GRAC GSREG SET 1 DOESN'T GO HIGH(ON PAD 5) OR/<CRLF>
9860	062550	051123	043505	051440	

9861	062556	052105	030440	042040	
9862	062564	042517	047123	052047	
9863	062572	043440	020117	044510	
9864	062600	044107	047450	020116	
9865	062606	040520	020104	024465	
9866	062614	047440	100122		
9867	062620	052111	042040	042517	.ASCIZ /IT DOESN'T GET TO THE GS REG/
9868	062626	047123	052047	043440	
9869	062634	052105	052040	020117	
9870	062642	044124	020105	051507	
9871	062650	051040	043505	000	
9872	062655	107	040522	020102	EM332: .ASCIZ /GRAB SRC SET 1 DOESN'T GO LOW OR GRAC E23(6) BAD/
9873	062662	051123	020103	042523	
9874	062670	020124	020061	047504	
9875	062676	051505	023516	020124	
9876	062704	047507	046040	053517	
9877	062712	047440	020122	051107	
9878	062720	041501	042440	031462	
9879	062726	033050	020051	040502	
9880	062734	000104			
9881	062736	051107	041501	042440	EM333: .ASCIZ /GRAC E24(6) DOESN'T GO LOW/
9882	062744	032062	033050	020051	
9883	062752	047504	051505	023516	
9884	062760	020124	047507	046040	
9885	062766	053517	000		
9886	062771	107	040522	020103	EM334: .ASCIZ /GRAC E23(4) DOESN'T GO LOW/
9887	062776	031105	024063	024464	
9888	063004	042040	042517	047123	
9889	063012	052047	043440	020117	
9890	063020	047514	000127		
9891	063024	051107	041501	042440	EM335: .ASCIZ /GRAC E23(4) IS NOT GOING LOW/
9892	063032	031462	032050	020051	
9893	063040	051511	047040	052117	
9894	063046	043440	044517	043516	
9895	063054	046040	053517	000	
9896	063061	120	051104	020104	EM336: .ASCIZ /PDRD PS11 DOESN'T GET TO GRAC AS A LOW/
9897	063066	051520	030461	042040	
9898	063074	042517	047123	052047	
9899	063102	043440	052105	052040	
9900	063110	020117	051107	041501	
9901	063116	040440	020123	020101	
9902	063124	047514	000127		
9903	063130	042522	044507	052123	EM337: .ASCIZ /REGISTER SET 1 SOURCE HAS STUCK BITS/
9904	063136	051105	051440	052105	
9905	063144	030440	051440	052517	
9906	063152	041522	020105	040510	
9907	063160	020123	052123	041525	
9908	063166	020113	044502	051524	
9909	063174	000			
9910	063175	105	051122	051117	DH337: .ASCIZ /ERRORPC PATTERN TEST NUMBER/
9911	063202	041520	050040	052101	
9912	063210	042524	047122	052040	
9913	063216	051505	020124	052516	
9914	063224	041115	051105	000	
9915		063232			.EVEN.
9916	063232	001116	001164	001210	DT337: .WORD \$ERRPC,\$TMP1,\$\$STSTNM,0

9917	063240	000000			
9918	063242	042522	044507	052123	EM340: .ASCIZ /REGISTER SET 1 DESTINATION HAS STUCK BITS/
9919	063250	051105	051440	052105	
9920	063256	030440	042040	051505	
9921	063264	044524	040516	044524	
9922	063272	047117	044040	051501	
9923	063300	051440	052524	045503	
9924	063306	041040	052111	000123	
9925	063314	051107	041101	042040	EM341: .ASCIZ /GRAB DST SET 1 DOESN'T GO LOW ON R14/
9926	063322	052123	051440	052105	
9927	063330	030440	042040	042517	
9928	063336	047123	052047	043440	
9929	063344	020117	047514	020127	
9930	063352	047117	051040	032061	
9931	063360	000			
9932	063361	107	040522	020102	EM342: .ASCIZ /GRAB SRC SET 1 DOESN'T GO LOW ON R14/
9933	063366	051123	020103	042523	
9934	063374	020124	020061	047504	
9935	063402	051505	023516	020124	
9936	063410	047507	046040	053517	
9937	063416	047440	020116	030522	
9938	063424	000064			
9939	063426	030065	030060	020060	EM343: .ASCIZ /50000 PATTERN FAILED IN PSW/
9940	063434	040520	052124	051105	
9941	063442	020116	040506	046111	
9942	063450	042105	044440	020116	
9943	063456	051520	000127		
9944	063462	033061	030064	030060	EM344: .ASCIZ /164000 PATTERN FAILED IN PSW/
9945	063470	050040	052101	042524	
9946	063476	047122	043040	044501	
9947	063504	042514	020104	047111	
9948	063512	050040	053523	000	
9949	063517	120	053523	044040	EM345: .ASCIZ /PSW HIGH BYTE DIDN'T CLEAR/
9950	063524	043511	020110	054502	
9951	063532	042524	042040	042111	
9952	063540	023516	020124	046103	
9953	063546	040505	000122		
9954	063552	051107	041101	042040	EM346: .ASCIZ /GRAB DST SET 1 DOESN'T GO LOW ON DF6*SUPER MODE/
9955	063560	052123	051440	052105	
9956	063566	030440	042040	042517	
9957	063574	047123	052047	043440	
9958	063602	020117	047514	020117	
9959	063610	047117	042040	033106	
9960	063616	051452	050125	051105	
9961	063624	046440	042117	000105	
9962	063632	051107	041101	051440	EM347: .ASCIZ /GRAB SRC SET 1 DOESN'T GO LOW ON SF6*SUPER MODE/
9963	063640	041522	051440	052105	
9964	063646	030440	042040	042517	
9965	063654	047123	052047	043440	
9966	063662	020117	047514	020127	
9967	063670	047117	051440	033106	
9968	063676	051452	050125	051105	
9969	063704	046440	042117	000105	
9970	063712	051107	041501	042440	EM350: .ASCIZ /GRAC E35(8) DOESN'T GO HIGH ON DF6*USER MODE/
9971	063720	032463	034050	020051	
9972	063726	047504	051505	023516	

9973	063734	020124	047507	044040	
9974	063742	043511	020110	047117	
9975	063750	042040	033106	052452	
9976	063756	042523	020122	047515	
9977	063764	042504	000		
9978	063767	107	040522	020103	EM351: .ASCIZ /GRAC E15(6) DOESN'T GO HIGH ON SF6*USER MODE/
9979	063774	030505	024065	024466	
9980	064002	042040	042517	047123	
9981	064010	052047	043440	020117	
9982	064016	044510	044107	047440	
9983	064024	020116	043123	025066	
9984	064032	051525	051105	046440	
9985	064040	042117	000105		
9986	064044	051525	051105	047440	EM352: .ASCIZ /USER OR SUPER SP DST FAILED BIT TEST/
9987	064052	020122	052523	042520	
9988	064060	020122	050123	042040	
9989	064066	052123	043040	044501	
9990	064074	042514	020104	044502	
9991	064102	020124	042524	052123	
9992	064110	000			
9993	064111	125	042523	020122	EM353: .ASCIZ /USER OR SUPER SP SRC FAILED BIT TEST/
9994	064116	051117	051440	050125	
9995	064124	051105	051440	020120	
9996	064132	051123	020103	040506	
9997	064140	046111	042105	041040	
9998	064146	052111	052040	051505	
9999	064154	000124			
10000	064156	051107	041501	042440	EM354: .ASCIZ /GRAC E20(4) NOT GOING LOW/
10001	064164	030062	032050	020051	
10002	064172	047516	020124	047507	
10003	064200	047111	020107	047514	
10004	064206	000127			
10005	064210	051107	041501	042440	EM355: .ASCIZ /GRAC E20(12) NOT GOING LOW ON MTP*DM0*DF6*PS12/
10006	064216	030062	030450	024462	
10007	064224	047040	052117	043440	
10008	064232	044517	043516	046040	
10009	064240	053517	047440	020116	
10010	064246	052115	025120	046504	
10011	064254	025060	043104	025066	
10012	064262	051520	031061	000	
10013	064267	107	040522	020103	EM356: .ASCIZ /GRAC E20(11) NOT GOING LOW/
10014	064274	031105	024060	030461	
10015	054302	020051	047516	020124	
10016	064310	047507	047111	020107	
10017	064316	047514	000127		
10018	064322	051523	020120	040502	EM357: .ASCII /SSP BAD ON MTP EITHER GRAC GRWE/<CRLF>
10019	064330	020104	047117	046440	
10020	064336	050124	042440	052111	
10021	064344	042510	020122	051107	
10022	064352	041501	043440	053522	
10023	064360	100105			
10024	064362	044510	020102	051117	! .ASCIZ /HIB OR LOB NOT GOING LOW/
10025	064370	046040	041117	047040	
10026	064376	052117	043440	044517	
10027	064404	043516	046040	053517	
10028	064412	000			

10029	064413	105	051122	051117	DH357: .ASCII /ERRORPC SSP TST NUM/<CRLF>
10030	064420	041520	020040	020040	
10031	064426	020040	051523	004520	
10032	064434	051524	020124	052516	
10033	064442	100115			
10034	064444	020011	054105	042520	.ASCIZ / EXPECT ACTUAL/
10035	064452	052103	020040	041501	
10036	064460	052524	046101	000	
10037	064465	107	040522	020103	EM360: .ASCIZ /GRAC E35(8) DIDN'T GO HIGH ON MTP*DM0*DF6*PS13/
10038	064472	031505	024065	024470	
10039	064500	042040	042111	023516	
10040	064506	020124	047507	044040	
10041	064514	043511	020110	047117	
10042	064522	046440	050124	042052	
10043	064530	030115	042052	033106	
10044	064536	050052	030523	000063	
10045	064544	051111	041503	042040	EM361: .ASCIZ /IRCC DM0(MFP+MTP) NOT GOING HIGH ON MFP/
10046	064552	030115	046450	050106	
10047	064560	046453	050124	020051	
10048	064566	047516	020124	047507	
10049	064574	047111	020107	044510	
10050	064602	044107	047440	020116	
10051	064610	043115	000120		
10052	064614	051107	041501	042440	EM362: .ASCIZ /GRAC E24(13) DOESN'T GO HIGH/
10053	064622	032062	030450	024463	
10054	064630	042040	042517	047123	
10055	064636	052047	043440	020117	
10056	064644	044510	044107	000	
10057	064651	120	051104	020104	EM363: .ASCIZ /PDRD PS15 DOESN'T GET TO GRAC AS A HIGH/
10058	064656	051520	032461	042040	
10059	064664	042517	047123	052047	
10060	064672	043440	052105	052040	
10061	064700	020117	051107	041501	
10062	064706	040440	020123	020101	
10063	064714	044510	044107	000	
10064	064721	123	050123	053440	EM364: .ASCIZ /SSP WAS READ BUT NEITHER USP NOR SSP WERE WRITTEN/
10065	064726	051501	051040	040505	
10066	064734	020104	052502	020124	
10067	064742	042516	052111	042510	
10068	064750	020122	051525	020120	
10069	064756	047516	020122	051523	
10070	064764	020120	042527	042522	
10071	064772	053440	044522	052124	
10072	065000	047105	000		
10073	065003	107	040522	020103	EM365: .ASCIZ /GRAC E23(13) DOESN'T GO HIGH/
10074	065010	031105	024063	031461	
10075	065016	020051	047504	051505	
10076	065024	023516	020124	047507	
10077	065032	044040	043511	000110	
10078	065040	051107	041501	042440	EM366: .ASCIZ /GRAC E23(3) DOESN'T GO HIGH/
10079	065046	031462	031450	020051	
10080	065054	047504	051505	023516	
10081	065062	020124	047507	044040	
10082	065070	043511	000110		
10083	065074	042120	042122	050040	EM367: .ASCII /PDRD PS14(0) DOESN'T GET TO PDRD E73(13)/<CRLF>
10084	065102	030523	024064	024460	

10085	065110	042040	042517	047123	
10086	065116	052047	043440	052105	
10087	065124	052040	020117	042120	
10088	065132	042122	042440	031467	
10089	065140	030450	024463	200	
10090	065145	101	020123	020101	.ASCIZ /AS A HIGH OR E73(1,2) NOT GOING LOW/
10091	065152	044510	044107	047440	
10092	065160	020122	033505	024063	
10093	065166	031061	020051	047516	
10094	065174	020124	047507	047111	
10095	065202	020107	047514	000127	
10096	065210	020101	051520	020127	EM370: .ASCII /A PSW HIGH BYTE BIT(S) DIDN'T SET ON LOAD PS &/<CRLF>
10097	065216	044510	044107	041040	
10098	065224	052131	020105	044502	
10099	065232	024124	024523	042040	
10100	065240	042111	023516	020124	
10101	065246	042523	020124	047117	
10102	065254	046040	040517	020104	
10103	065262	051520	023040	200	
10104	065267	113	051105	042516	.ASCIZ /KERNEL MODE WITH A BR VALUE OF 174000/
10105	065274	020114	047515	042504	
10106	065302	053440	052111	020110	
10107	065310	020101	051102	053040	
10108	065316	046101	042525	047440	
10109	065324	020106	033461	030064	
10110	065332	030060	000		
10111	065335	101	041040	052111	EM371: .ASCII /A BIT(S) IN THE PSW CLEARED ON AN RTI IN USER MODE/<CRLF>
10112	065342	051450	020051	047111	
10113	065350	052040	042510	050040	
10114	065356	053523	041440	042514	
10115	065364	051101	042105	047440	
10116	065372	020116	047101	051040	
10117	065400	044524	044440	020116	
10118	065406	051525	051105	046440	
10119	065414	042117	100105		
10120	065420	020046	020101	051102	.ASCIZ /& A BR VALUE OF 0/
10121	065426	053040	046101	042525	
10122	065434	047440	020106	000060	
10123	065442	020101	044502	024124	EM372: .ASCII /A BIT(S) OF PSW <15,13:11> DIDN'T PRESET ON AN RTI/<CRLF>
10124	065450	024523	047440	020106	
10125	065456	051520	020127	030474	
10126	065464	026065	031461	030472	
10127	065472	037061	042040	042111	
10128	065500	023516	020124	051120	
10129	065506	051505	052105	047440	
10130	065514	020116	047101	051040	
10131	065522	044524	200		
10132	065525	111	020116	052523	.ASCIZ /IN SUPER MODE & A BR VALUE OF 134000/
10133	065532	042520	020122	047515	
10134	065540	042504	023040	040440	
10135	065546	041040	020122	040526	
10136	065554	052514	020105	043117	
10137	065562	030440	032063	030060	
10138	065570	000060			
10139	065572	020101	044502	024124	EM373: .ASCII /A BIT(S) IN PSW HIGH BYTE FAILED ON AN IOT IN USER/<CRLF>
10140	065600	024523	044440	020116	

10141	065606	051520	020127	044510	
10142	065614	044107	041040	052131	
10143	065622	020105	040506	046111	
10144	065630	042105	047440	020116	
10145	065636	047101	044440	052117	
10146	065644	044440	020116	051525	
10147	065652	051105	200		
10148	065655	115	042117	020105	.ASCIZ /MODE & A BR VALUE OF 0/
10149	065662	020046	020101	051102	
10150	065670	053040	046101	042525	
10151	065676	047440	020106	000060	
10152	065704	020101	051120	053105	EM374: .ASCIZ /A PREVIOUS MODE BIT(S) DIDN'T CLEAR ON AN IOT IN KERNEL/
10153	065712	047511	051525	046440	
10154	065720	042117	020105	044502	
10155	065726	024124	024523	042040	
10156	065734	042111	023516	020124	
10157	065742	046103	040505	020122	
10158	065750	047117	040440	020116	
10159	065756	047511	020124	047111	
10160	065764	045440	051105	042516	
10161	065772	000114			
10162	065774	051523	041522	045440	EM375: .ASCII /SSRC KT ABORT FLG DOESN'T GET TO TMCC E34(10) OR/<CRLF>
10163	066002	020124	041101	051117	
10164	066010	020124	046106	020107	
10165	066016	047504	051505	023516	
10166	066024	020124	042507	020124	
10167	066032	047524	052040	041515	
10168	066040	020103	031505	024064	
10169	066046	030061	020051	051117	
10170	066054	200			
10171	066055	105	032063	030450	.ASCIZ /E34(10) BAD OR TMCC AERF(1) DOESN'T GO HIGH ON A KT ABORT/
10172	066062	024460	041040	042101	
10173	066070	047440	020122	046524	
10174	066076	041503	040440	051105	
10175	066104	024106	024461	042040	
10176	066112	042517	047123	052047	
10177	066120	043440	020117	044510	
10178	066126	044107	047440	020116	
10179	066134	020101	052113	040440	
10180	066142	047502	052122	000	
10181	066147	120	053523	041040	EM376: .ASCIZ /PSW BIT 8 FAILED TO SET/
10182	066154	052111	034040	043040	
10183	066162	044501	042514	020104	
10184	066170	047524	051440	052105	
10185	066176	000			
10186	066177	124	041515	020102	EM400: .ASCII /TMCB SEGT DOESN'T GO LOW ON A KT ABORT OR/<CRLF>
10187	066204	042523	052107	042040	
10188	066212	042517	047123	052047	
10189	066220	043440	020117	047514	
10190	066226	020127	047117	040440	
10191	066234	045440	020124	041101	
10192	066242	051117	020124	051117	
10193	066250	200			
10194	066251	111	020124	047504	.ASCIZ /IT DOESN'T GET TO DAPE/
10195	066256	051505	023516	020124	
10196	066264	042507	020124	047524	

10197	066272	042040	050101	000105	
10198	066300	040504	042520	052040	EM401: .ASCIZ /DAPE TV05*07 DOESN'T GO HIGH ON SEGT/
10199	066306	030126	025065	033460	
10200	066314	042040	042517	047123	
10201	066322	052047	043440	020117	
10202	066330	044510	044107	047440	
10203	066336	020116	042523	052107	
10204	066344	000			
10205	066345	104	050101	020105	EM402: .ASCIZ /DAPE TV03 DOESN'T GO HIGH ON SEGT/
10206	066352	053124	031460	042040	
10207	066360	042517	047123	052047	
10208	066366	043440	020117	044510	
10209	066374	044107	047440	020116	
10210	066402	042523	052107	000	
10211	066407	124	041515	020101	EM403: .ASCII /TMCA SEG+CON+PAR DOESN'T GO LOW OR IT DOES/<CRLF>
10212	066414	042523	025507	047503	
10213	066422	025516	040520	020122	
10214	066430	047504	051505	023516	
10215	066436	020124	047507	046040	
10216	066444	053517	047440	020122	
10217	066452	052111	042040	042517	
10218	066460	100123			
10219	066462	047516	020124	042507	.ASCIZ /NOT GET THRU TMCB E70(2)/
10220	066470	020124	044124	052522	
10221	066476	052040	041515	020102	
10222	066504	033505	024060	024462	
10223	066512	000			
10224	066513	124	041515	020101	EM405: .ASCII /TMCA SEGTF DOESN'T GET TO E44 OR TMCE PAUSES H/<15><12>
10225	066520	042523	052107	020106	
10226	066526	047504	051505	023516	
10227	066534	020124	042507	020124	
10228	066542	047524	042440	032064	
10229	066550	047440	020122	046524	
10230	066556	042503	050040	052501	
10231	066564	042523	020123	006510	
10232	066572	012			
10233	066573	104	042517	047123	.ASCIZ /DOESN'T GET TO E44 OR E44 BAD/
10234	066600	052047	043440	052105	
10235	066606	052040	020117	032105	
10236	066614	020064	051117	042440	
10237	066622	032064	041040	042101	
10238	066630	000			
10239	066631	124	041515	020103	EM406: .ASCII /TMCC NEXM DOESN'T GO LOW OR IT DOESN'T GET THRU E34/<CRLF>
10240	066636	042516	046530	042040	
10241	066644	042517	047123	052047	
10242	066652	043440	020117	047514	
10243	066660	020127	051117	044440	
10244	066666	020124	047504	051505	
10245	066674	023516	020124	042507	
10246	066702	020124	044124	052522	
10247	066710	042440	032063	200	
10248	066715	117	020122	052111	.ASCIZ /OR IT DOESN'T GET TO E14 OR E40 BAD/
10249	066722	042040	042517	047123	
10250	066730	052047	043440	052105	
10251	066736	052040	020117	030505	
10252	066744	020064	051117	042440	

10253	066752	030064	041040	042101	
10254	066760	000			
10255	066761	116	054105	020115	EM410: .ASCII /NEXM BIT DIDN'T SET IN CPU ERROR REG/<CRLF>
10256	066766	044502	020124	044504	
10257	066774	047104	052047	051440	
10258	067002	052105	044440	020116	
10259	067010	050103	020125	051105	
10260	067016	047522	020122	042522	
10261	067024	100107			
10262	067026	051117	052040	041515	.ASCIIZ /OR TMCC ABORT DOESN'T GO HIGH/
10263	067034	020103	041101	051117	
10264	067042	020124	047504	051505	
10265	067050	023516	020124	047507	
10266	067056	044040	043511	000110	
10267	067064	042516	046530	041040	EM411: .ASCIIZ /NEXM BIT DIDN'T CLEAR IN CPU ERROR REG/
10268	067072	052111	042040	042111	
10269	067100	023516	020124	046103	
10270	067106	040505	020122	047111	
10271	067114	041440	052520	042440	
10272	067122	051122	051117	051040	
10273	067130	043505	000		
10274	067133	124	041515	020105	EM412: .ASCIIZ /TMCE KT BEND DOESN'T GO LOW ON TMCC NEXM LOW/
10275	067140	052113	041040	047105	
10276	067146	020104	047504	051505	
10277	067154	023516	020124	047507	
10278	067162	046040	053517	047440	
10279	067170	020116	046524	041503	
10280	067176	047040	054105	020115	
10281	067204	047514	000127		
10282	067210	046524	042503	045440	EM413: .ASCIIZ /TMCE KT BEND DOESN'T GO LOW ON TMCD SL RED/
10283	067216	020124	042502	042116	
10284	067224	042040	042517	047123	
10285	067232	052047	043440	020117	
10286	067240	047514	020127	047117	
10287	067246	052040	041515	020104	
10288	067254	046123	051040	042105	
10289	067262	000			
10290	067263	124	041515	020105	EM414: .ASCIIZ /TMCE KT BEND DOESN'T GO LOW ON TMCC ODD ADRS ERR/
10291	067270	052113	041040	047105	
10292	067276	020104	047504	051505	
10293	067304	023516	020124	047507	
10294	067312	046040	053517	047440	
10295	067320	020116	046524	041503	
10296	067326	047440	042104	040440	
10297	067334	051104	020123	051105	
10298	067342	000122			
10299	067344	052113	040440	047502	EM415: .ASCIIZ ?KT ABORT FAILED TO OVER-RIDE NEXM TRAP (KB11-E/EM)?
10300	067352	052122	043040	044501	
10301	067360	042514	020104	047524	
10302	067366	047440	042526	026522	
10303	067374	044522	042504	047040	
10304	067402	054105	020115	051124	
10305	067410	050101	024040	041113	
10306	067416	030461	042455	042457	
10307	067424	024515	000		
10308	067427	124	041515	020105	EM416: .ASCIIZ /TMCE CACHE BEND DIDN'T GO HIGH ON KT ABORT/

10309	067434	040503	044103	020105	
10310	067442	042502	042116	042040	
10311	067450	042111	023516	020124	
10312	067456	047507	044040	043511	
10313	067464	020110	047117	045440	
10314	067472	020124	041101	051117	
10315	067500	000124			
10316	067502	040504	040520	041040	EM417: .ASCII /DAPA BR14 DOESN'T GET TO RACK E49 AS A LOW/<CRLF>
10317	067510	030522	020064	047504	
10318	067516	051505	023516	020124	
10319	067524	042507	020124	047524	
10320	067532	051040	041501	020113	
10321	067540	032105	020071	051501	
10322	067546	040440	046040	053517	
10323	067554	200			
10324	067555	117	020122	032105	.ASCIZ /OR E49(5) BAD/
10325	067562	024071	024465	041040	
10326	067570	042101	000		
10327	067573	111	046114	043505	EM420: .ASCIZ /ILLEGAL HALT BIT IN CPU ERROR REG DIDN'T SET/
10328	067600	046101	044040	046101	
10329	067606	020124	044502	020124	
10330	067614	047111	041440	052520	
10331	067622	042440	051122	051117	
10332	067630	051040	043505	042040	
10333	067636	042111	023516	020124	
10334	067644	042523	000124		
10335	067650	051523	040522	050040	EM421: .ASCII /SSRA PS RESTORE(1) DOESN'T GET TO RACK E63/<CRLF>
10336	067656	020123	042522	052123	
10337	067664	051117	024105	024461	
10338	067672	042040	042517	047123	
10339	067700	052047	043440	052105	
10340	067706	052040	020117	040522	
10341	067714	045503	042440	031466	
10342	067722	200			
10343	067723	117	020122	033105	.ASCIZ /OR E63(5) BAD/
10344	067730	024063	024465	041040	
10345	067736	042101	000		
10346	067741	125	041502	020102	EM422: .ASCII /UBCB PE ABORT DOESN'T GO LOW OR/<CRLF>
10347	067746	042520	040440	047502	
10348	067754	052122	042040	042517	
10349	067762	047123	052047	043440	
10350	067770	020117	047514	020127	
10351	067776	051117	200		
10352	070001	111	020124	047504	.ASCII /IT DOESN'T GET TO TMCC E33 OR E33 BAD/<CRLF>
10353	070006	051505	023516	020124	
10354	070014	042507	020124	047524	
10355	070022	052040	041515	020103	
10356	070030	031505	020063	051117	
10357	070036	042440	031463	041040	
10358	070044	042101	200		
10359	070047	117	020122	041125	.ASCII /OR UBCB PARITY ERR DOESN'T GET TO TMCB E53/<CRLF>
10360	070054	041103	050040	051101	
10361	070062	052111	020131	051105	
10362	070070	020122	047504	051505	
10363	070076	023516	020124	042507	
10364	070104	020124	047524	052040	

10365	070112	041515	020102	032505	
10366	070120	100063			
10367	070122	051501	040440	046040	.ASCIZ /AS A LOW OR E53(5) BAD/
10368	070130	053517	047440	020122	
10369	070136	032505	024063	024465	
10370	070144	041040	042101	000	
10371	070151	125	041502	020102	EM423: .ASCII /UBCB PARITY ERR DOESN'T GO LOW OR IT DOES/<CRLF>
10372	070156	040520	044522	054524	
10373	070164	042440	051122	042040	
10374	070172	042517	047123	052047	
10375	070200	043440	020117	047514	
10376	070206	020127	051117	044440	
10377	070214	020124	047504	051505	
10378	070222	200			
10379	070223	116	052117	043440	.ASCIZ /NOT GET TO DAPE/
10380	070230	052105	052040	020117	
10381	070236	040504	042520	000	
10382	070243	104	050101	020105	EM424: .ASCIZ /DAPE E11(4) BAD OR TV06 DOESN'T GET TO THE ALU/
10383	070250	030505	024061	024464	
10384	070256	041040	042101	047440	
10385	070264	020122	053124	033060	
10386	070272	042040	042517	047123	
10387	070300	052047	043440	052105	
10388	070306	052040	020117	044124	
10389	070314	020105	046101	000125	
10390	070322	040504	042520	042440	EM425: .ASCIZ /DAPE E7(1) BAD/
10391	070330	024067	024461	041040	
10392	070336	042101	000		
10393	070341	124	041515	020101	EM426: .ASCIZ /TMCA SEG+CON+PAR DOESN'T GO LOW ON CCBJ PARITY TRAP/
10394	070346	042523	025507	047503	
10395	070354	025516	040520	020122	
10396	070362	047504	051505	023516	
10397	070370	020124	047507	046040	
10398	070376	053517	047440	020116	
10399	070404	041503	045102	050040	
10400	070412	051101	052111	020131	
10401	070420	051124	050101	000	
10402	070425	124	041515	020102	EM427: .ASCII /TMCB PART DOESN'T GO LOW OR DOES/<CRLF>
10403	070432	040520	052122	042040	
10404	070440	042517	047123	052047	
10405	070446	043440	020117	047514	
10406	070454	020127	051117	042040	
10407	070462	042517	100123		
10408	070466	047516	020124	042507	.ASCIZ /NOT GET TO UBCB OR UBCB E18(1) BAD/
10409	070474	020124	047524	052440	
10410	070502	041502	020102	051117	
10411	070510	052440	041502	020102	
10412	070516	030505	024070	024461	
10413	070524	041040	042101	000	
10414	070531	124	041515	020101	EM430: .ASCIZ /TMCA E67(8) DOESN'T GO LOW ON MGMT/
10415	070536	033105	024067	024470	
10416	070544	042040	042517	047123	
10417	070552	052047	043440	020117	
10418	070560	047514	020127	047117	
10419	070566	046440	046507	000124	
10420	070574	046524	040503	042440	EM431: .ASCIZ /TMCA E67(12) DOESN'T GO LOW ON MGMT/

10421	070602	033466	030450	024462	
10422	070610	042040	042517	047123	
10423	070616	052047	043440	020117	
10424	070624	047514	020127	047117	
10425	070632	046440	046507	000124	
10426	070640	046524	040503	042440	EM432: .ASCII /TMCA E68(6) DOESN'T GO LOW ON PAR TRP/<CRLF>
10427	070646	034066	033050	020051	
10428	070654	047504	051505	023516	
10429	070662	020124	047507	046040	
10430	070670	053517	047440	020116	
10431	070676	040520	020122	051124	
10432	070704	100120			
10433	070706	051117	042440	032464	.ASCIZ /OR E45(4) BAD/
10434	070714	032050	020051	040502	
10435	070722	000104			
10436	070724	046524	040503	042440	EM433: .ASCIZ /TMCA E68(8) DOESN'T GO LOW ON PAR TRP/
10437	070732	034066	034050	020051	
10438	070740	047504	051505	023516	
10439	070746	020124	047507	046040	
10440	070754	053517	047440	020116	
10441	070762	040520	020122	051124	
10442	070770	000120			
10443	070772	046524	041503	050040	EM435: .ASCII /TMCC PRIORITY CLEAR DIDN'T GO LOW OR DIDN'T/<CRLF>
10444	071000	044522	051117	052111	
10445	071006	020131	046103	040505	
10446	071014	020122	044504	047104	
10447	071022	052047	043440	020117	
10448	071030	047514	020127	051117	
10449	071036	042040	042111	023516	
10450	071044	100124			
10451	071046	042507	020124	044124	.ASCIZ /GET THRU TMCA E43(2) ON ABORT CLEAR/
10452	071054	052522	052040	041515	
10453	071062	020101	032105	024063	
10454	071070	024462	047440	020116	
10455	071076	041101	051117	020124	
10456	071104	046103	040505	000122	
10457	071112	052502	020123	041120	EM436: .ASCIZ /BUS PB DIDN'T GET TO UBCB PE ABORT/
10458	071120	042040	042111	023516	
10459	071126	020124	042507	020124	
10460	071134	047524	052440	041502	
10461	071142	020102	042520	040440	
10462	071150	047502	052122	000	
10463	071155	125	041502	020102	EM437: .ASCIZ /UBCB PARITY ERR DIDN'T GO LOW ON BUS PB/
10464	071162	040520	044522	054524	
10465	071170	042440	051122	042040	
10466	071176	042111	023516	020124	
10467	071204	047507	046040	053517	
10468	071212	047440	020116	052502	
10469	071220	020123	041120	000	
10470	071225	127	044501	020124	EM440: .ASCIZ /WAIT INSTRUCTION DIDN'T WAIT FOR INTERRUPT/
10471	071232	047111	052123	052522	
10472	071240	052103	047511	020116	
10473	071246	044504	047104	052047	
10474	071254	053440	044501	020124	
10475	071262	047506	020122	047111	
10476	071270	042524	051122	050125	

10477	071276	000124				
10478	071300	040527	052111	044440	EM441:	.ASCIZ /WAIT INSTRUCTION FELL THRU ON T BIT TRAP/
10479	071306	051516	051124	041525		
10480	071314	044524	047117	043040		
10481	071322	046105	020114	044124		
10482	071330	052522	047440	020116		
10483	071336	020124	044502	020124		
10484	071344	051124	050101	000		
10485	071351	125	041502	020103	EM442:	.ASCIZ /UBCC (PWR+INTR) DOESN'T GO LOW ON TMCB PIRQ/
10486	071356	050050	051127	025506		
10487	071364	047111	051124	020051		
10488	071372	047504	051505	023516		
10489	071400	020124	047507	046040		
10490	071406	053517	047440	020116		
10491	071414	046524	041103	050040		
10492	071422	051111	000121			
10493	071426	030523	027063	030060	EM443:	.ASCIZ /S13.00 FAILED/
10494	071434	043040	044501	042514		
10495	071442	000104				
10496	071444	032123	027065	030060	EM444:	.ASCIZ /S45.00 FAILED/
10497	071452	043040	044501	042514		
10498	071460	000104				
10499	071462	052115	027120	030061	EM445:	.ASCIZ /MTP.10 FAILED/
10500	071470	043040	044501	042514		
10501	071476	000104				
10502	071500	042516	027107	030071	EM446:	.ASCIZ /NEG.90 FAILED/
10503	071506	043040	044501	042514		
10504	071514	000104				
10505	071516	032104	027065	030070	EM447:	.ASCIZ /D45.80 FAILED/
10506	071524	043040	044501	042514		
10507	071532	000104				
10508	071534	032104	027065	030071	EM450:	.ASCIZ /D45.90 FAILED/
10509	071542	043040	044501	042514		
10510	071550	000104				
10511	071552	032104	027065	030060	EM451:	.ASCIZ /D45.00 FAILED/
10512	071560	043040	044501	042514		
10513	071566	000104				
10514	071570	032104	027065	030460	EM452:	.ASCIZ /D45.01 FAILED/
10515	071576	043040	044501	042514		
10516	071604	000104				
10517	071606	047101	044440	046114	EM453:	.ASCIZ /AN ILLEGAL INSTRUCTION FAILED TO TRAP/
10518	071614	043505	046101	044440		
10519	071622	051516	051124	041525		
10520	071630	044524	047117	043040		
10521	071636	044501	042514	020104		
10522	071644	047524	052040	040522		
10523	071652	000120				
10 24	071654	046524	041103	042440	EM454:	.ASCIZ /TMCB E53 DOESN'T GO HIGH OR TMCB PIRQ DOESN'T GO LOW/
10 25	071662	031465	042040	042517		
10526	071670	047123	052047	043440		
10527	071676	020117	044510	044107		
10528	071704	047440	020122	046524		
10529	071712	041103	050040	051111		
10530	071720	020121	047504	051505		
10531	071726	023516	020124	047507		
10532	071734	046040	053517	000		

10533	071741	123	051123	020101	EM455: .ASCII /SSRA PS RESTORE IS STUCK HIGH OR IT IS NOT/<CRLF>
10534	071746	051520	051040	051505	
10535	071754	047524	042522	044440	
10536	071762	020123	052123	041525	
10537	071770	020113	044510	044107	
10538	071776	047440	020122	052111	
10539	072004	044440	020123	047516	
10540	072012	100124			
10541	072014	042507	052124	047111	.ASCIZ /GETTING THRU RACK E63(B0) AS A LOW/
10542	072022	020107	044124	052522	
10543	072030	051040	041501	020113	
10544	072036	033105	024063	030102	
10545	072044	020051	051501	040440	
10546	072052	046040	053517	000	
10547	072057	124	041515	020103	EM456: .ASCIZ /TMCC E5(11) DOESN'T GO HIGH ON DATI OR DATO/
10548	072064	032505	030450	024461	
10549	072072	042040	042517	047123	
10550	072100	052047	043440	020117	
10551	072106	044510	044107	047440	
10552	072114	020116	040504	044524	
10553	072122	047440	020122	040504	
10554	072130	047524	000		
10555	072133	120	053523	041040	EM457: .ASCIZ /PSW BIT 11 DOESN'T CLEAR/
10556	072140	052111	030440	020061	
10557	072146	047504	051505	023516	
10558	072154	020124	046103	040505	
10559	072162	000122			
10560	072164	051520	020127	044103	EM460: .ASCIZ /PSW CHANGED ON RESET IN KERNEL MODE/
10561	072172	047101	042507	020104	
10562	072200	047117	051040	051505	
10563	072206	052105	044440	020116	
10564	072214	042513	047122	046105	
10565	072222	046440	042117	000105	
10566	072230	051520	020127	044103	EM461: .ASCIZ /PSW CHANGES ON RESET IN SUPER MODE/
10567	072236	047101	042507	020123	
10568	072244	047117	051040	051505	
10569	072252	052105	044440	020116	
10570	072260	052523	042520	020122	
10571	072266	047515	042504	000	
10572	072273	115	046505	046440	EM703: .ASCIZ /MEM MGT TESTS DISABLED/<CRLF>
10573	072300	052107	052040	051505	
10574	072306	051524	042040	051511	
10575	072314	041101	042514	100104	
10576	072322	000			
10577	072323	103	041501	042510	EM704: .ASCIZ /CACHE TESTS DISABLED/<CRLF>
10578	072330	052040	051505	051524	
10579	072336	042040	051511	041101	
10580	072344	042514	100104	000	
10581	072351	125	044516	052502	EM706: .ASCIZ /UNIBUS PE TEST DISABLED/<CRLF>
10582	072356	020123	042520	052040	
10583	072364	051505	020124	044504	
10584	072372	040523	046102	042105	
10585	072400	000200			
10586	072402	047516	041040	020122	EM710: .ASCIZ /NO BR 5 DEVICE/<CRLF>
10587	072410	020065	042504	044526	
10588	072416	042503	000200		

10589	072422	047516	041040	020122	EM711: .ASCIZ /NO BR 6 DEVICE/<CRLF>
10590	072430	020066	042504	044526	
10591	072436	042503	000200		
10592	072442	051102	032040	052040	EM712: .ASCIZ /BR 4 TESTS DISABLED/<CRLF>
10593	072450	051505	051524	042040	
10594	072456	051511	041101	042514	
10595	072464	100104	000		
10596	072467	102	020122	020065	EM713: .ASCIZ /BR 5 TESTS DISABLED/<CRLF>
10597	072474	042524	052123	020123	
10598	072502	044504	040523	046102	
10599	072510	042105	000200		
10600	072514	051102	033040	052040	EM715: .ASCIZ /BR 6 TESTS DISABLED/<CRLF>
10601	072522	051505	051524	042040	
10602	072530	051511	041101	042514	
10603	072536	100104	000		
10604	072541	117	051120	052040	EM717: .ASCIZ /OPR TEST DISABLED/<CRLF>
10605	072546	051505	020124	044504	
10606	072554	040523	046102	042105	
10607	072562	000200			
10608	072564	047514	045517	040440	EM720: .ASCII /LOOK AT THE CONSOLE LIGHTS/<CRLF>
10609	072572	020124	044124	020105	
10610	072600	047503	051516	046117	
10611	072606	020105	044514	044107	
10612	072614	051524	200		
10613	072617	124	042510	042040	.ASCII /THE DATA LIGHTS SHOULD READ 166667/<CRLF>
10614	072624	052101	020101	044514	
10615	072632	044107	051524	051440	
10616	072640	047510	046125	020104	
10617	072646	042522	042101	030440	
10618	072654	033066	033066	100067	
10619	072662	044124	020105	042101	.ASCIZ /THE ADDRESS LIGHTS SHOULD READ /
10620	072670	051104	051505	020123	
10621	072676	044514	044107	051524	
10622	072704	051440	047510	046125	
10623	072712	020104	042522	042101	
10624	072720	020040	000		
10625	072723	200	044103	047101	EM721: .ASCIZ <CRLF>/CHANGE SWITCH 7 TO CONTINUE/<CRLF>
10626	072730	042507	051440	044527	
10627	072736	041524	020110	020067	
10628	072744	047524	041440	047117	
10629	072752	044524	052516	100105	
10630	072760	000			
10631	072761	200	054524	042520	EM722: .ASCIZ <CRLF>/TYPE A CHARACTER TO CONTINUE/<CRLF>
10632	072766	040440	041440	040510	
10633	072774	040522	052103	051105	
10634	073002	052040	020117	047503	
10635	073010	052116	047111	042525	
10636	073016	000200			
10637	073020	047200	020117	040515	EM724: .ASCIZ <CRLF>/NO MAP REGISTERS AVAILABLE FOR TEST 75/<CRLF>
10638	073026	020120	042522	044507	
10639	073034	052123	051105	020123	
10640	073042	053101	044501	040514	
10641	073050	046102	020105	047506	
10642	073056	020122	042524	052123	
10643	073064	033440	100065	000	
10644	073071	116	020117	020114	EM725: .ASCIZ /NO L CLOCK/<CRLF>

10645	073076	046103	041517	100113	
10646	073104	000			
10647		073106			
10648	073106	073106			ADRTAB: .EVEN
10649	073110	005446			.WORD
10650	073112	005630			TST1
10651	073114	006124			TST2
10652	073116	006406			TST3
10653	073120	007150			TST4
10654	073122	007264			TST5
10655	073124	007344			TST6
10656	073126	007430			TST7
10657	073130	010252			TST10
10658	073132	010412			TST11
10659	073134	010754			TST12
10660	073136	011070			TST13
10661	073140	012550			TST14
10662	073142	012716			TST15
10663	073144	013070			TST16
10664	073146	013256			TST17
10665	073150	013416			TST20
10666	073152	013532			TST21
10667	073154	013736			TST22
10668	073156	014306			TST23
10669	073160	014444			TST24
10670	073162	014602			TST25
10671	073164	014752			TST26
10672	073166	015126			TST27
10673	073170	015326			TST30
10674	073172	015500			TST31
10675	073174	016620			TST32
10676	073176	017114			TST33
10677	073200	017244			TST34
10678	073202	017374			TST35
10679	073204	017766			TST36
10680	073206	020402			TST37
10681	073210	021136			TST40
10682	073212	021402			TST41
10683	073214	022012			TST42
10684	073216	022446			TST43
10685	073220	023204			TST44
10686	073222	023454			TST45
10687	073224	023526			TST46
10688	073226	026350			TST47
10689	073230	027000			TST50
10690	073232	027214			TST51
10691	073234	027364			TST52
10692	073236	027634			TST53
10693	073240	030000			TST54
10694	073242	030234			TST55
10695	073244	030362			TST56
10696	073246	030572			TST57
10697	073250	030650			TST60
10698	073252	031414			TST61
10699	073254	031530			TST62
10700	073256	032024			TST63
					TST64

CEKBBFO 11/70 CPU #2 MACY11 30A(1052) 16-APR-80 09:11 PAGE 197 B 16
CEKBBF.P11 07-APR-80 13:15 POWER DOWN AND UP ROUTINES

SEQ 0196

10701 073260 032336
10702 000001

SUBTAB: \$EOP
.END

		9382	9398	9439	9446	9455	9462	9469	9476	9499	9511	9543	9551	9560
		9569	9607	9623	9657	9694	9716	9729	9744	9761	9789	9829	9846	9859
		10018	10029	10083	10096	10111	10123	10139	10162	10186	10211	10239	10255	10316
		10335	10346	10352	10359	10371	10402	10426	10443	10533	10572	10577	10581	10586
		10589	10592	10596	10600	10604	10608	10613	10625	10631	10637	10644		
DH1	036576	2264	2268	2271	7985#									
DH14	037427	2298	2563	8064#										
DH15	037552	2302	8080#											
DH163	046625	2614	8749#											
DH201	050127	2648	2657	2666	2675	2687	2696	8877#						
DH243	053355	2759	2763	9183#										
DH25	040327	2326	2446	8146#										
DH250	054106	2775	9248#											
DH255	054676	2790	2796	2802	3064	3088	9316#							
DH263	055433	2808	7461	9378#										
DH263A	055460	7478	9382#											
DH263B	055644	9403#	9428											
DH263C	055670	9407#	9429											
DH263D	055716	9411#	9430											
DH263E	055745	9415#	9431											
DH263F	055773	9419#	9432											
DH263G	056016	9423#	9433											
DH337	063175	2940	2943	2973	2976	9910#								
DH357	064413	2988	10029#											
DH4	037045	2274	2277	2280	2283	2286	2289	2292	2295	2305	2308	2311	2314	2317
		2320	2323	2330	2337	2343	2346	2349	2352	2355	2358	2361	2404	2431
		2434	2440	2452	2461	2464	2470	2473	2479	2485	2488	2491	2494	2497
		2509	2515	2518	2551	2557	2560	2566	2569	2572	2575	2581	2584	2590
		2593	2596	2599	2602	2605	2608	2611	2618	2621	2624	2627	2630	2633
		2636	2642	2645	2651	2654	2660	2663	2669	2672	2678	2681	2684	2690
		2693	2699	2702	2705	2708	2711	2714	2717	2720	2723	2726	2729	2732
		2735	2738	2741	2744	2747	2750	2753	2756	2766	2769	2772	2778	2781
		2784	2787	2793	2799	2805	2814	2817	2820	2823	2826	2829	2832	2835
		2838	2841	2844	2847	2850	2853	2856	2859	2862	2865	2868	2871	2874
		2877	2880	2883	2886	2889	2892	2895	2898	2901	2904	2907	2910	2913
		2916	2919	2922	2925	2928	2931	2934	2937	2946	2949	2961	2964	2967
		2970	2979	2982	2985	2992	2995	2998	3001	3004	3007	3010	3031	3034
		3040	3043	3046	3049	3052	3055	3058	3061	3067	3070	3073	3076	3079
		3082	3085	3091	3094	3097	3100	3103	3106	3109	3112	3115	3118	3121
		3124	3127	3130	3134	3138	3141	3144	3147	3150	3153	3156	3159	3162
		3165	3168	3171	3174	3177	3180	3183	8016#					
DH41	041317	2364	2548	8237#										
DH42	041440	2333	2340	2368	2379	2385	2419	2428	2443	2449	2455	2458	2506	2512
		2521	2545	2554	2578	2587	2952	2955	2958	3013	3016	3019	3022	3025
		3028	3186	3189	8253#									
DH43	041562	2372	2376	2382	2388	2401	2407	2410	2413	2416	2422	2425	2437	2467
		2476	2482	2500	2503	2524	2527	2530	2533	2536	2539	2542	8269#	
DH46	042036	2639	8301#											
DH51	042251	2391	2395	2398	8328#									
DISPLA=	177570	1750#	7356*	7380*										
DOMFPT	032046	7087	7089#											
DONE7	032076	7088	7094	7098#										
DT1	036716	2266	2269	2272	8000#									
DT14	037510	2300	2303	2564	2990	8074#								
DT163	046704	2616	8758#											
DT201	050160	2649	2658	2667	2676	2688	2697	8883#						

EM127	045373	2529	8625#				
EM13	037347	2294	8054#				
EM130	045445	2532	2538	8633#			
EM134	045506	2544	8639#				
EM135	045555	2547	8646#				
EM136	045564	2550	8648#				
EM14	037377	2297	8059#				
EM140	045617	2556	2580	8653#			
EM141	045654	2559	8658#				
EM142	045713	2562	8664#				
EM143	045741	2565	8668#				
EM144	045775	2568	8673#				
EM145	046024	2571	8677#				
EM146	046063	2574	8683#				
EM147	046122	2577	8689#				
EM15	037522	2301	8076#				
EM151	046153	2583	8694#				
EM152	046166	2586	8696#				
EM155	046215	2595	8700#				
EM156	046246	2598	8705#				
EM157	046366	2601	8719#				
EM16	037633	2304	2589	8089#			
EM160	046417	2604	8724#				
EM161	046450	2607	8729#				
EM162	046545	2610	8740#				
EM163	046576	2613	8745#				
EM164	046716	2617	8760#				
EM165	046747	2620	8765#				
EM166	047031	2623	8774#				
EM167	047136	2626	8786#				
EM17	037667	2307	8094#				
EM170	047200	2629	8792#				
EM171	047242	2632	8798#				
EM172	047363	2635	8812#				
EM174	047636	2641	8843#				
EM175	047720	2644	2653	2662	2671	2692	8852#
EM176	047735	2647	8855#				
EM177	047777	2650	8861#				
EM2	036734	2267	8003#				
EM20	037744	2310	8102#				
EM201	050061	2656	8870#				
EM202	050170	2659	8885#				
EM204	050252	2665	8894#				
EM205	050320	2668	8901#				
EM207	050403	2674	8910#				
EM21	040035	2313	8112#				
EM210	050451	2677	8917#				
EM211	050517	2680	8924#				
EM212	050600	2683	8933#				
EM213	050644	2686	8939#				
EM214	050712	2689	8946#				
EM216	050774	2695	8955#				
EM217	051042	2698	8962#				
EM22	040055	2316	8115#				
EM220	051076	2701	8967#				
EM221	051157	2704	8976#				

EM222	051213	2707	8981#	
EM223	051273	2710	8990#	
EM224	051314	2713	8993#	
EM225	051463	2716	9011#	
EM226	051574	2719	9024#	
EM227	051654	2722	9032#	
EM23	040152	2319	8126#	
EM230	051735	2725	9041#	
EM232	052222	2731	9073#	
EM233	052274	2734	9080#	
EM234	052371	2737	9092#	
EM235	052476	2740	9105#	
EM237	052643	2746	9123#	
EM24	040242	2322	8136#	
EM240	052736	2749	9134#	
EM241	053040	2752	9146#	
EM242	053151	2755	9159#	
EM243	053305	2758	9176#	
EM244	053452	2762	9195#	
EM245	053534	2765	9205#	
EM246	053627	2768	9216#	
EM247	053750	2771	9230#	
EM25	040275	2325	2381	8141#
EM250	054061	2774	9244#	
EM251	054242	2777	9266#	
EM252	054407	2780	9284#	
EM253	054457	2783	9291#	
EM254	054527	2786	9298#	
EM255	054617	2789	9308#	
EM256	054757	2792	9325#	
EM257	055014	2795	9330#	
EM26	040414	2329	8157#	
EM260	055065	2798	9337#	
EM261	055130	2801	9343#	
EM262	055202	2804	9350#	
EM263	055244	2807	7459	9356#
EM264	056056	2810	9434#	
EM265	056101	2813	9439#	
EM266	056417	2816	9476#	
EM267	056535	2819	9490#	
EM27	040454	2332	8163#	
EM270	056615	2822	9499#	
EM271	056713	2825	9511#	
EM272	057043	2828	9527#	
EM273	057102	2831	9533#	
EM274	057150	2834	9540#	
EM275	057170	2837	9543#	
EM276	057540	2840	9584#	
EM277	057571	2843	9589#	
EM3	036755	2270	8006#	
EM30	040516	2336	8169#	
EM300	057645	2846	9597#	
EM301	057734	2849	9607#	
EM302	060021	2852	9616#	
EM303	060066	2855	9623#	
EM304	060217	2858	9639#	

EM305	060304	2861	9648#
EM306	060372	2864	9657#
EM307	060514	2867	9671#
EM31	040554	2339	8174#
EM310	060567	2870	9679#
EM311	060641	2873	9687#
EM312	060712	2876	9694#
EM313	061036	2879	9709#
EM314	061106	2882	9716#
EM315	061217	2885	9729#
EM316	061341	2888	9744#
EM317	061454	2891	9758#
EM32	040616	2342	8180#
EM320	061474	2894	9761#
EM321	061612	2897	9774#
EM322	061662	2900	9781#
EM323	061737	2903	9789#
EM324	062047	2906	9803#
EM325	062130	2909	9812#
EM326	062205	2912	9820#
EM327	062266	2915	9829#
EM33	040662	2345	8186#
EM330	062431	2918	9846#
EM331	062542	2921	9859#
EM332	062655	2924	9872#
EM333	062736	2927	9881#
EM334	062771	2930	9886#
EM335	063024	2933	9891#
EM336	063061	2936	9896#
EM337	063130	2939	9903#
EM34	040700	2348	8189#
EM340	063242	2942	9918#
EM341	063314	2945	9925#
EM342	063361	2948	9932#
EM343	063426	2951	9939#
EM344	063462	2954	9944#
EM345	063517	2957	9949#
EM346	063552	2960	9954#
EM347	063632	2963	9962#
EM35	040733	2351	8194#
EM350	063712	2966	9970#
EM351	063767	2969	9978#
EM352	064044	2972	9986#
EM353	064111	2975	9993#
EM354	064156	2978	10000#
EM355	064210	2981	10005#
EM356	064267	2984	10013#
EM357	064322	2987	10018#
EM36	041051	2354	8208#
EM360	064465	2991	10037#
EM361	064544	2994	10045#
EM362	064614	2997	10052#
EM363	064651	3000	10057#
EM364	064721	3003	10064#
EM365	065003	3006	10073#
EM366	065040	3009	10078#

EM367	065074	3012	10083#
EM37	041134	2357	8217#
EM370	065210	3015	10096#
EM371	065335	3018	10111#
EM372	065442	3021	10123#
EM373	065572	3024	10139#
EM374	065704	3027	10152#
EM375	065774	3030	10162#
EM376	066147	3033	10181#
EM4	037002	2273	8010#
EM40	041204	2360	8224#
EM400	066177	3039	10186#
EM401	066300	3042	10198#
EM402	066345	3045	10205#
EM403	066407	3048	10211#
EM405	066513	3054	10224#
EM406	066631	3057	10239#
EM41	041254	2363	8231#
EM410	066761	3063	10255#
EM411	067064	3066	10267#
EM412	067133	3069	10274#
EM413	067210	3072	10282#
EM414	067263	3075	10290#
EM415	067344	3078	10299#
EM416	067427	3081	10308#
EM417	067502	3084	10316#
EM42	041410	2367	8249#
EM420	067573	3087	10327#
EM421	067650	3090	10335#
EM422	067741	3093	10346#
EM423	070151	3096	10371#
EM424	070243	3099	10382#
EM425	070322	3102	10390#
EM426	070341	3105	10393#
EM427	070425	3108	10402#
EM43	041530	2371	8264#
EM430	070531	3111	10414#
EM431	070574	3114	10420#
EM432	070640	3117	10426#
EM433	070724	3120	10436#
EM435	070772	3126	10443#
EM436	071112	3129	10457#
EM437	071155	3133	10463#
EM44	041710	2375	8286#
EM440	071225	3137	10470#
EM441	071300	3140	10478#
EM442	071351	3143	10485#
EM443	071426	3146	10493#
EM444	071444	3149	10496#
EM445	071462	3152	10499#
EM446	071500	3155	10502#
EM447	071516	3158	10505#
EM45	041754	2378	8292#
EM450	071534	3161	10508#
EM451	071552	3164	10511#
EM452	071570	3167	10514#

EM453	071606	3170	10517#																	
EM454	071654	3173	10524#																	
EM455	071741	3176	10533#																	
EM456	072057	3179	10547#																	
EM457	072133	3182	10555#																	
EM46	042003	2638	8296#																	
EM460	072164	3185	10560#																	
EM461	072230	3188	10566#																	
EM47	042132	2384	8313#																	
EM5	037100	2276	8022#																	
EM50	042157	2387	8317#																	
EM51	042211	2390	8322#																	
EM52	042432	2394	8349#																	
EM53	042505	2397	8357#																	
EM54	042540	2400	8362#																	
EM55	042556	2403	2439	2592	8365#															
EM56	042634	2406	8373#																	
EM57	042705	2409	8380#																	
EM6	037166	2279	8031#																	
EM60	042755	2412	8387#																	
EM61	043056	2415	8399#																	
EM62	043134	2418	8407#																	
EM63	043176	2421	8413#																	
EM64	043236	2424	8419#																	
EM65	043276	2427	8425#																	
EM66	043333	2430	8430#																	
EM67	043364	2433	8435#																	
EM7	037221	2282	8036#																	
EM70	043416	2436	8440#																	
EM703	072273	10572#																		
EM704	072323	10577#																		
EM706	072351	10581#																		
EM710	072402	4942	10586#																	
EM711	072422	4968	10589#																	
EM712	072442	5030	10592#																	
EM713	072467	5092	10596#																	
EM715	072514	5125	10600#																	
EM717	072541	7119	10604#																	
EM72	043434	2442	8443#																	
EM720	072564	7124	7146	10608#																
EM721	072723	7131	7140	10625#																
EM722	072761	7151	10631#																	
EM724	073020	10637#																		
EM725	073071	4953	10644#																	
EM73	043476	2445	8449#																	
EM74	043514	2448	8452#																	
EM75	043556	2451	8458#																	
EM76	043643	2454	8467#																	
EM77	043666	2457	8471#																	
ENDBR5	016470	4974	4981#																	
ENDGET	026344	6410	6412	6416#																
ENDKB	023152	5838	5841	5843#																
ERRVEC=	000004	1846#	3256*	3257*	3340*	3360*	4470*	4481*	4594*	4600*	4627*	4660*	4689*	4693*						
		4697*	4724*	4728*	4731*	4760*	4769*	4774*	4801*	4805*	4808*	4833*	4836*	4838*						
		4857*	4868*	4874*	4884*	4916*	4919*	4922*	4925*	4945*	4950*	4956*	4962*	4964*						
		5172*	5173*	5187*	5192*	5196*	5201*	5207*	5213*	5220*	5352*	5355*	5360*	5364*						

ITEM43	001610	2371#
ITEM44	001616	2375#
ITEM45	001624	2378#
ITEM46	001632	2381#
ITEM47	001640	2384#
ITEM5	001324	2276#
ITEM50	001646	2387#
ITEM51	001654	2390#
ITEM52	001662	2394#
ITEM53	001670	2397#
ITEM54	001676	2400#
ITEM55	001704	2403#
ITEM56	001712	2406#
ITEM57	001720	2409#
ITEM6	001332	2279#
ITEM60	001726	2412#
ITEM61	001734	2415#
ITEM62	001742	2418#
ITEM63	001750	2421#
ITEM64	001756	2424#
ITEM65	001764	2427#
ITEM66	001772	2430#
ITEM67	002000	2433#
ITEM7	001340	2282#
ITEM70	002006	2436#
ITEM71	002014	2439#
ITEM72	002022	2442#
ITEM73	002030	2445#
ITEM74	002036	2448#
ITEM75	002044	2451#
ITEM76	002052	2454#
ITEM77	002060	2457#
ITE100	002066	2460#
ITE101	002074	2463#
ITE102	002102	2466#
ITE103	002110	2469#
ITE104	002116	2472#
ITE105	002124	2475#
ITE106	002132	2478#
ITE107	002140	2481#
ITE110	002146	2484#
ITE111	002154	2487#
ITE112	002162	2490#
ITE113	002170	2493#
ITE114	002176	2496#
ITE115	002204	2499#
ITE116	002212	2502#
ITE117	002220	2505#
ITE120	002226	2508#
ITE121	002234	2511#
ITE122	002242	2514#
ITE123	002250	2517#
ITE124	002256	2520#
ITE125	002264	2523#
ITE126	002272	2526#
ITE127	002300	2529#

ITE130	002306	2532#
ITE131	002314	2535#
ITE132	002322	2538#
ITE133	002330	2541#
ITE134	002336	2544#
ITE135	002344	2547#
ITE136	002352	2550#
ITE137	002360	2553#
ITE140	002366	2556#
ITE141	002374	2559#
ITE142	002402	2562#
ITE143	002410	2565#
ITE144	002416	2568#
ITE145	002424	2571#
ITE146	002432	2574#
ITE147	002440	2577#
ITE150	002446	2580#
ITE151	002454	2583#
ITE152	002462	2586#
ITE153	002470	2589#
ITE154	002476	2592#
ITE155	002504	2595#
ITE156	002512	2598#
ITE157	002520	2601#
ITE160	002526	2604#
ITE161	002534	2607#
ITE162	002542	2610#
ITE163	002550	2613#
ITE164	002556	2617#
ITE165	002564	2620#
ITE166	002572	2623#
ITE167	002600	2626#
ITE170	002606	2629#
ITE171	002614	2632#
ITE172	002622	2635#
ITE173	002630	2638#
ITE174	002636	2641#
ITE175	002644	2644#
ITE176	002652	2647#
ITE177	002660	2650#
ITE200	002666	2653#
ITE201	002674	2656#
ITE202	002702	2659#
ITE203	002710	2662#
ITE204	002716	2665#
ITE205	002724	2668#
ITE206	002732	2671#
ITE207	002740	2674#
ITE210	002746	2677#
ITE211	002754	2680#
ITE212	002762	2683#
ITE213	002770	2686#
ITE214	002776	2689#
ITE215	003004	2692#
ITE216	003012	2695#
ITE217	003020	2698#

ITE220	003026	2701#
ITE221	003034	2704#
ITE222	003042	2707#
ITE223	003050	2710#
ITE224	003056	2713#
ITE225	003064	2716#
ITE226	003072	2719#
ITE227	003100	2722#
ITE230	003106	2725#
ITE231	003114	2728#
ITE232	003122	2731#
ITE233	003130	2734#
ITE234	003136	2737#
ITE235	003144	2740#
ITE236	003152	2743#
ITE237	003160	2746#
ITE240	003166	2749#
ITE241	003174	2752#
ITE242	003202	2755#
ITE243	003210	2758#
ITE244	003216	2762#
ITE245	003224	2765#
ITE246	003232	2768#
ITE247	003240	2771#
ITE250	003246	2774#
ITE251	003254	2777#
ITE252	003262	2780#
ITE253	003270	2783#
ITE254	003276	2786#
ITE255	003304	2789#
ITE256	003312	2792#
ITE257	003320	2795#
ITE260	003326	2798#
ITE261	003334	2801#
ITE262	003342	2804#
ITE263	003350	2807#
ITE264	003356	2810#
ITE265	003364	2813#
ITE266	003372	2816#
ITE267	003400	2819#
ITE270	003406	2822#
ITE271	003414	2825#
ITE272	003422	2828#
ITE273	003430	2831#
ITE274	003436	2834#
ITE275	003444	2837#
ITE276	003452	2840#
ITE277	003460	2843#
ITE300	003466	2846#
ITE301	003474	2849#
ITE302	003502	2852#
ITE303	003510	2855#
ITE304	003516	2858#
ITE305	003524	2861#
ITE306	003532	2864#
ITE307	003540	2867#

ITE310	003546	2870#
ITE311	003554	2873#
ITE312	003562	2876#
ITE313	003570	2879#
ITE314	003576	2882#
ITE315	003604	2885#
ITE316	003612	2888#
ITE317	003620	2891#
ITE320	003626	2894#
ITE321	003634	2897#
ITE322	003642	2900#
ITE323	003650	2903#
ITE324	003656	2906#
ITE325	003664	2909#
ITE326	003672	2912#
ITE327	003700	2915#
ITE330	003706	2918#
ITE331	003714	2921#
ITE332	003722	2924#
ITE333	003730	2927#
ITE334	003736	2930#
ITE335	003744	2933#
ITE336	003752	2936#
ITE337	003760	2939#
ITE340	003766	2942#
ITE341	003774	2945#
ITE342	004002	2948#
ITE343	004010	2951#
ITE344	004016	2954#
ITE346	004032	2960#
ITE347	004040	2963#
ITE350	004046	2966#
ITE351	004054	2969#
ITE352	004062	2972#
ITE353	004070	2975#
ITE354	004076	2978#
ITE355	004104	2981#
ITE356	004112	2984#
ITE357	004120	2987#
ITE360	004126	2991#
ITE361	004134	2994#
ITE362	004142	2997#
ITE363	004150	3000#
ITE364	004156	3003#
ITE365	004164	3006#
ITE366	004172	3009#
ITE367	004200	3012#
ITE370	004206	3015#
ITE371	004214	3018#
ITE372	004222	3021#
ITE373	004230	3024#
ITE374	004236	3027#
ITE375	004244	3030#
ITE376	004252	3033#
ITE377	004260	3036#
ITE400	004266	3039#

ITE401	004274	3042#							
ITE402	004302	3045#							
ITE403	004310	3048#							
ITE404	004316	3051#							
ITE405	004324	3054#							
ITE406	004332	3057#							
ITE407	004340	3060#							
ITE410	004346	3063#							
ITE411	004354	3066#							
ITE412	004362	3069#							
ITE413	004370	3072#							
ITE414	004376	3075#							
ITE415	004404	3078#							
ITE416	004412	3081#							
ITE417	004420	3084#							
ITE420	004426	3087#							
ITE421	004434	3090#							
ITE422	004442	3093#							
ITE423	004450	3096#							
ITE424	004456	3099#							
ITE425	004464	3102#							
ITE426	004472	3105#							
ITE427	004500	3108#							
ITE430	004506	3111#							
ITE431	004514	3114#							
ITE432	004522	3117#							
ITE433	004530	3120#							
ITE434	004536	3123#							
ITE435	004544	3126#							
ITE436	004552	3129#							
ITE437	004560	3133#							
ITE440	004566	3137#							
ITE441	004574	3140#							
ITE442	004602	3143#							
ITE443	004610	3146#							
ITE444	004616	3149#							
ITE445	004624	3152#							
ITE446	004632	3155#							
ITE447	004640	3158#							
ITE450	004646	3161#							
ITE451	004654	3164#							
ITE452	004662	3167#							
ITE453	004670	3170#							
ITE454	004676	3173#							
ITE455	004704	3176#							
ITE456	004712	3179#							
ITE457	004720	3182#							
ITE460	004726	3185#							
ITE461	004734	3188#							
KBDONE	023204	5832	5849	5851#					
KBTST	023112	5717	5831#						
KB11E	001272	2240#	5833*	5837*	5846	5892	6884	6897	7086
KB11EM	001273	2241#							
KDPA0=	172360	2023#							
KDPA1=	172362	2024#							
KDPA2=	172364	2025#							

MAPH15= 170266	2069#	
MAPH16= 170272	2071#	
MAPH17= 170276	2073#	
MAPH2 = 170212	2111#	
MAPH20= 170302	2075#	
MAPH21= 170306	2077#	
MAPH22= 170312	2079#	
MAPH23= 170316	2081#	
MAPH24= 170320	2083#	
MAPH25= 170326	2085#	
MAPH26= 170332	2087#	
MAPH27= 170336	2089#	
MAPH3 = 170216	2113#	
MAPH30= 170342	2091#	
MAPH31= 170346	2093#	
MAPH32= 170352	2095#	
MAPH33= 170356	2097#	
MAPH34= 170362	2099#	
MAPH35= 170366	2101#	
MAPH36= 170372	2103#	
MAPH37= 170376	2105#	
MAPH4 = 170222	2115#	
MAPH5 = 170226	2117#	
MAPH6 = 170232	2119#	
MAPH7 = 170236	2121#	
MAPL0 = 170200	2106#	3192*
MAPL00= 170200	2042#	2106
MAPL01= 170204	2044#	2108
MAPL02= 170210	2046#	2110
MAPL03= 170214	2048#	2112
MAPL04= 170220	2050#	2114
MAPL05= 170224	2052#	2116
MAPL06= 170230	2054#	2118
MAPL07= 170234	2056#	2120
MAPL1 = 170204	2108#	3194*
MAPL10= 170240	2058#	
MAPL11= 170244	2060#	
MAPL12= 170250	2062#	
MAPL13= 170254	2064#	
MAPL14= 170260	2066#	
MAPL15= 170264	2068#	
MAPL16= 170270	2070#	
MAPL17= 170274	2072#	
MAPL2 = 170210	2110#	
MAPL20= 170300	2074#	
MAPL21= 170304	2076#	
MAPL22= 170310	2078#	
MAPL23= 170314	2080#	
MAPL24= 170320	2082#	
MAPL25= 170324	2084#	
MAPL26= 170330	2086#	
MAPL27= 170334	2088#	
MAPL3 = 170214	2112#	
MAPL30= 170340	2090#	
MAPL31= 170344	2092#	
MAPL32= 170350	2094#	

\$LPERR 001110	2181#	3227*	3338*	3395*	3455*	3491*	3516*	3531*	3553*	3697*	3724*	3740*	3755*
	3770*	3889*	3916*	4042*	4109*	4119*	4152*	4172*	4183*	4203*	4221*	4240*	4259*
	4278*	4363*	4389*	4503*	4630*	4662*	4695*	4730*	4772*	4807*	4839*	4858*	4867*
	5165*	5174*	5188*	5197*	5208*	5222*	5230*	5252*	5259*	5267*	5278*	5284*	5297*
	5308*	5353*	5379*	5388*	5399*	5411*	5428*	5647*	5674*	5692*	5725*	5733*	5739*
	5745*	5754*	5762*	5773*	5782*	5788*	5797*	5806*	5813*	5876*	5898*	6003*	6022*
	6039*	6055*	6073*	6088*	6108*	6130*	6145*	6164*	6195*	6218*	6238*	6255*	6270*
	6288*	6305*	6323*	6338*	6353*	6368*	6444*	6458*	6472*	6500*	6511*	6618*	6629*
	6641*	6653*	6749*	6777*	6892*	6910*	6924*	6937*	6950*	7024*	7035*	7053*	7337
	7353*	7359	7403										
\$MXCNT 033516	7350	7359#											
\$NULL 001146	2195#	7715	7744										
\$NWTST= 000001	3266#	3268	3320#	3322	3378#	3380	3439#	3441	3572#	3574	3608#	3610	3629#
	3631	3651#	3653	3784#	3786	3821#	3823	3928#	3930	3960#	3962	4291#	4293
	4334#	4336	4377#	4379	4417#	4419	4454#	4456	4488#	4490	4531#	4533	4607#
	4609	4640#	4642	4673#	4675	4707#	4709	4742#	4744	4784#	4786	4819#	4821
	5013#	5015	5077#	5079	5110#	5112	5143#	5145	5235#	5237	5326#	5328	5434#
	5436	5490#	5492	5612#	5614	5710#	5712	5853#	5855	5915#	5917	5934#	5936
	6418#	6420	6520#	6522	6579#	6581	6609#	6611	6667#	6669	6705#	6707	6763#
	6765	6795#	6797	6846#	6848	6862#	6864	6978#	6980	7011#	7013	7071#	7073
	7100#	7102											
\$SOCNT 035766	7777*	7806*	7819#										
\$SOMODE 035770	7772*	7776*	7781	7784*	7795*	7821#							
\$SOVER 033502	7314	7330	7338	7348	7356#								
\$SPASS 001100	2176#	3262	4910	5028	5090	5123	7115	7177*	7178*	7189	7231	7344	7360
\$SPOWER 036400	7952	7957#											
\$SPR2 001212	2216#	8000											
\$SPR5 001214	2217#	8000											
\$SPWRDN 036250	3208	4595	5449	7925#	7949								
\$SPWRMG 036366	7952#												
\$SPWRUP 036316	7934	7939#											
\$SQUES 001202	2211#	7409											
\$SRDCHR= ***** U	7920												
\$SRDEC= ***** U	7920												
\$RDLIN= ***** U	7920												
\$RDOCT= ***** U	7920												
\$SREGAD 001152	2199#												
\$SREGO 001154	2201#	3421*	3428*	3684*	3692*	3706*	3750*	3767*	3779*	3811*	3816*	3859*	3897*
	3954*	4068*	4088*	4098*	4134*	4141*	4215*	4229*	4248*	4267*	4319*	4360*	4888*
	5226*	5425*	5610*	6740*	7001*	7489	7538	7624*	7661	8074	8247	8283	8346
\$SREG1 001156	2202#	3422*	3429*	3507*	3564*	3693*	3707*	3751*	3768*	3780*	3812*	3817*	3860*
	3898*	3955*	4069*	4089*	4099*	4135*	4142*	4217*	4230*	4249*	4268*	4361*	6743*
	7656*	7667	8074	8155	8283	8346							
\$SREG2 001160	2203#												
\$RTRN 032622	3214	3216*	3221*	4482	5971	7048	7052	7205	7226#				
\$SAVRE= ***** U	7920												
\$SAVR6 036376	7933*	7939	7940*	7941*	7956#								
\$SCOPE 033252	3202	6971	7312#										
\$SETUP= 000037	2128#	3202	3204	3206	3208	3210	3211	3212	3214	3226	3232	7175	7395
\$STUP = 177777	2128#												
\$SVLAD 033454	7322	7351#											
\$SVPC = 000204	2160#	2165											
\$SWR = 177400	1710	1711#	1719	1720	1721	1722	1723	1724	1725	1726	2208	2209	2210
	3211	3212	3214	3226	3227	3289	3333	3393	3453	3585	3618	3639	3666
	3796	3847	3940	4022	4307	4347	4386	4428	4466	4499	4545	4617	4650
	4683	4717	4754	4794	4832	5021	5084	5117	5164	5245	5344	5447	5516

\$XTSTR 033260	7316#													
\$\$DONE 034404	7496	7538#												
\$\$GET4= 000001	7203#	7210#												
\$\$TRP = 000002	7905#	7916	7917	7918	7919	7920								
\$\$TSTN 001210	2215#	7254*	7259	7282*	7287	7377*	7415	7417	7469	8000	8021	8074	8155	
	8247	8262	8283	8311	8346	8758	8883	9193	9203	9263	9916			
\$OFILL 035767	7771*	7775*	7785	7820#										
. = 073262	2131#	2135	2137#	2160	2161#	2163#	2165#	2173#	2214	3200	3226	3227	3237#	
	7188#	7231	7233#	7246#	7252#	7274#	7280#	7359	7360	7409	7543#	7572#	7589#	
	7602#	7660#	7744	7889#	7936	7955	7976#	7999#	8020#	8246#	8261#	8310#	8757#	
	9192#	9202#	9427#	9915#	10647#	10648								

MSG136	5110#	5112																
MSG137	5143#	5145																
MSG140	5326#	5328																
MSG141	5434#	5436																
MSG142	5490#	5492																
MSG143	5612#	5614																
MSG144	5853#	5855																
MSG145	5915#	5917																
MSG146	5934#	5936																
MSG147	6418#	6420																
MSG150	6520#	6522																
MSG151	6579#	6581																
MSG152	6609#	6611																
MSG153	6667#	6669																
MSG154	6705#	6707																
MSG155	6763#	6765																
MSG156	6795#	6797																
MSG157	6846#	6848																
MSG160	6862#	6864																
MSG173	7100#	7102																
MS137A	5235#	5237																
MS145A	5710#	5712																
MS160A	6978#	6980																
MS160B	7011#	7013																
MULT	1#	2125#																
MYTAGS	1696#	2214																
NEWST	1#	1699#	2125#	3266	3320	3378	3439	3572	3608	3629	3651	3784	3821	3928	3960			
	4291	4334	4377	4417	4454	4488	4531	4607	4640	4673	4707	4742	4784	4819	5013			
	5077	5110	5143	5235	5326	5434	5490	5612	5710	5853	5915	5934	6418	6520	6579			
	6609	6667	6705	6763	6795	6846	6862	6978	7011	7071	7100							
POP	1#	2125#	7876	7943														
PUSH	1#	2125#	7835	7927														
SAVEAD	1696#	3289	3334	3402	3585	3618	3639	3796	3940	4307	4386	4428	4466	4499	4545			
	4907	5049	5084	5117	5447	5524	5699	5867	6427	6526	6674	6772						
SCOPE	1744#	3288	3332	3392	3452	3584	3617	3638	3665	3795	3846	3939	4021	4306	4346			
	4385	4427	4465	4498	4544	4616	4649	4682	4716	4753	4793	4831	5020	5083	5116			
	5163	5244	5343	5446	5515	5624	5716	5866	5922	5967	6426	6525	6585	6615	6673			
	6711	6771	6804	6851	6881	6989	7019	7084	7113	7174								
SETTRA	7906#	7916	7917	7918	7919													
SETUP	1#	2125#	3196															
SKIP	1#	2125#	3263	3313	3368	3427	3569	3594	3626	3648	3778	3815	3926	3953	4288			
	4329	4374	4412	4446	4528	4596	4636	4668	4703	4738	4780	4814	4893	4952	4963			
	4969	5031	5093	5096	5107	5126	5129	5140	5233	5431	5609	5932	6388	6478	6481			
	6517	6571	6603	6792	6859	6973												
SLASH	1#	2125#																
SPACE	2125#																	
STARS	1#	1699#	2125#	2140	2166	2245	3229	3241	3266	3287	3320	3331	3378	3391	3439			
	3451	3489	3514	3529	3551	3572	3583	3608	3616	3629	3637	3651	3664	3695	3722			
	3738	3753	3784	3794	3821	3845	3887	3914	3928	3938	3960	4020	4040	4107	4117			
	4150	4170	4181	4201	4219	4238	4257	4276	4291	4305	4334	4345	4377	4384	4417			
	4426	4454	4464	4484	4486	4488	4497	4531	4543	4607	4615	4640	4648	4673	4681			
	4707	4715	4742	4752	4784	4792	4819	4830	4895	4906	5013	5019	5077	5082	5110			
	5115	5143	5162	5235	5243	5326	5342	5434	5445	5490	5514	5516	5523	5612	5623			
	5710	5715	5752	5771	5780	5795	5804	5823	5852	5853	5865	5869	5875	5915	5921			
	5934	5966	5995	6019	6036	6052	6067	6086	6102	6127	6143	6158	6183	6214	6231			
	6253	6268	6283	6302	6318	6336	6351	6365	6418	6425	6441	6456	6470	6497	6509			

.\$SUPR	1#			
.\$STRAP	1#	1699#	7297#	7890
.\$TYPB	1#			
.\$STYPD	1#	1699#	7297#	7822
.\$STYPE	1#	1699#	7671	
.\$STYPO	1#	1699#	7297#	7744
.\$1170	1#	1699#	1735	

. ABS. 073262 000

ERRORS DETECTED: 0

DSKZ:CEKBBF.BIN,DSKZ:CEKBBF.LST/CRF/SOL/NL:TOC=DSKZ:CEKBBF.SML,DSKZ:CEKBBF.P11
RUN-TIME: 67 100 10 SECONDS
RUN-TIME RATIO: 959/179=5.3
CORE USED: 35K (69 PAGES)