11/70–74

11/70 – 74 CPU # 1

CEKBACO

AH-7963C-MC

FICHE 1 OF 1

JAN 1980

digital

MADE IN USA

IDENTIFICATION

PRODUCT CODE: AC-7962C-MC

PRODUCT NAME: CEKBACO PDP-11/70-74MP CPU DIAGNOSTIC PART 1

DATE CREATED: MAY, 1979

MAINTAINER: DIAGNOSTIC ENGINEERING

AUTHORS: DON MONROE
MODIFIED BY: ERNEST PREISIG 12/*/78

CONTENTS

ABSTRACT

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

A. BASIC INSTRUCTION TESTS

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

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

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

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

B. ADVANCED INSTRUCTION AND MISCELLANEOUS LOGIC TESTS

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

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

CONSOLE SWITCH REGISTER OPTIONS. AFTER REPORTING THE ERROR THE PROGRAM CONTINUES ON ITS NORMAL SEQUENCE UNLESS MOD- IFIED 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 SE- QUENCE, 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 _OW), 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.

. REQUIREMENTS

2.1 EQUIPMENT

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

2.2 STORAGE

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

2.3 PRELIMINARY PROGRAMS

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

CEKBB REQUIRES THAT CEKBA RUN

3. LOADING PROCEDURE

3.1 METHOD

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

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE 5.1

4.2 STARTING ADDRESS

200

4.3 PROGRAM AND OPERATOR ACTION

A. CEKBA

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

5.2.1 SPURIOUS ERROR HANDLER

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

5.2.2 SCOPE

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

5.2.3 ERROR

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

5.2.4 TRAP CATCHER

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

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

5.2.5 TRAP

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

5.2.5.1 TYPE ($TYPE)

ROUTINE TO TYPE AN ASCII STRING ON THE TTY.

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

5.2.5.2 TYPEOC ($TYPOC)

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

5.2.5.3 TYPEDS ($TYPDS)

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

5.2.6 POWER DOWN AND UP

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

5.2.7 MONITOR RESTORE (QUIT)

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

5.2.8 CHECK TEST SEQUENCE (SEQENC)

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

5.2.9 PERIPHERAL DETERMINATOR AND INTERRUPT ENABLE ROUTINES

5.2.9.1 PERIPHERAL DETERMINATOR

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

5.2.9.2 INTERRUPT ENABLE ROUTINES

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

5.3 OPERATOR ACTION

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

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

6. ERRORS

6.1 ERROR HALTS AND DESCRIPTION

A. CEKBA

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

B. CEKBB

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

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

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

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

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

SEE SECTION 7.1 FOR NON-STANDARD CONFIGURATION.

6.2 ERROR RECOVERY

A. CEKBA

ERROR RECOVERY IS STRICTLY BY USER INTERVENTION.

B. CEKBB

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

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

7. RESTRICTIONS

7.1 STARTING RESTRICTIONS

A. CEKBA

NONE

B. CEKBB

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

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

7.2 OPERATING RESTRICTIONS

A. CEKBA

NONE

B. CEKBB

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

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

8. MISCELLANEOUS

8.1 EXECUTION TIME

A. CEKBA

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

B. CEKBB

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

8.2 STACK POINTER

STACK IS INITIALLY SET TO 1100.

8.3 PASS COUNT

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

A. CEKBA

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

B. CEKBB

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

8.4 ITERATIONS

A. CEKBA

NONE

B. CEKBB

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

8.5 SPECIAL REGISTERS

A. CEKBA

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 PRECEEDING THE INSTRUCTION UNDER TEST (IUT).

B. CEKBB

ONLY TESTS 1 THRU 20 CONTAIN SYNC INSTRUCTIONS.

8.8 CACHE CONTROL

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

9. PROGRAM DESCRIPTION

9.1 CEKBAB - DIAGNOSTIC ENHANCED TO TEST THE SOB INSTRUCTION MORE THOROUGHLY.

CEKBAC - PROGRAM ENHANCED TO HANDLE A MAIN MEMORY TIME OUT WHEN THE SYSTEM SIZE REGISTER IS GREATER THAN ACTUAL PHYSICAL MEMORY.DIAGNOSTIC MADE 11/74 COMPATABLE.

DOCUMENT
****
PDP 11/70-74MP CPU DIAGNOSTIC PART 1
****

TABLE OF CONTENTS
****

TABLE OF CONTENTS
****

```
                17      COPYRIGHT (C) JULY 21,1975
                        DIGITAL EQUIPMENT CORP.
                        MAYNARD, MASS. 01754

                        PROGRAM BY DONALD W. MONROE

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


                ;*****************************************************************
        28      ; BASIC DEFINITIONS
                ;*****************************************************************

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

                44      MISCELLANEOUS DEFINITIONS

                50      GENERAL PURPOSE REGISTER DEFINITIONS

                71      PRIORITY LEVEL DEFINITIONS

                81      "SWITCH REGISTER" SWITCH DEFINITIONS

                109     DATA BIT DEFINITIONS (BIT00 TO BIT15)

                137     BASIC "CPU" TRAP VECTOR ADDRESSES

                ;*****************************************************************
        153     ; CACHE   REGISTER DEFINITIONS
                ;*****************************************************************

                ;*****************************************************************
        164     ; CPU REGISTER DEFINITIONS
                ;*****************************************************************

                ;*****************************************************************
        178     ; MEMORY MANAGEMENT DEFINITIONS
                ;*****************************************************************

                181     MEMORY MANAGEMENT STATUS REGISTER ADDRESSES

                192     USER "I" PAGE DESCRIPTOR REGISTERS

                203     USER "D" PAGE DESCRIPTOR REGISTORS

                214     USER "I" PAGE ADDRESS REGISTERS

                225     USER "D" PAGE ADDRESS REGISTERS
```

```
                236     SUPERVISOR ''I'' PAGE DESCRIPTOR REGISTERS

                247     SUPERVISOR 'D'' PAGE DESCRIPTOR REGISTERS

                258     SUPERVISOR ''I'' PAGE ADDRESS REGISTERS

                269     SUPERVISOR 'D'' PAGE ADDRESS REGISTERS

                280     KERNEL ''I'' PAGE DESCRIPTOR REGISTERS

                291     KERNEL 'D'' PAGE DESCRIPTOR REGISTERS

                302     KERNEL ''I'' PAGE ADDRESS REGISTERS

                313     KERNEL 'D'' PAGE ADDRESS REGISTERS

                ;*******************************************************************
        317     ; UNIBUS MAP REGISTER DEFINITIONS
                ;*******************************************************************

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

                ;*******************************************************************
        409     ; TRAP CATCHER
                ;*******************************************************************

                422     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

                ;*******************************************************************
        426     ; STARTING ADDRESS(ES)
                ;*******************************************************************

                ;*******************************************************************
        432     ; ACT11 HOOKS
                ;*******************************************************************

                434     THE FOLLOWING LOCATIONS ARE SETUP TO BE USED WITH ACT11

                        LOCATION 46 WILL CONTAIN THE ADDRESS OF THE LOCICAL
                        END OF THE PROGRAM.
                        LOCATION 52 IS USED TO SPECIFY PROGRAM OPERATING REQUIREMENTS
                        AND/OR RESTRICTIONS. THIS IS ACCOMPLISHED BY SETTING VARIOUS BITS
                        TO A ONE OR A ZERO. THE BITS USED AND THERE MEANING ARE:

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

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

                                BITS 13-0  MUST BE ZERO'S
```

```
458     COMMON TAGS
        ***********************************************************************

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

        ***********************************************************************
506     ERROR POINTER TABLE
        ***********************************************************************

508     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:

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

522             FOLLOWING IS AN INDEX OF THE POSSIBLE FAILURES THAT
                CAUSE A TRAP TO LOCATIONS 4, 10, 14, OR 24 OR THAT
                CAUSE THE PROCESSOR TO HANG (H). NGT-NOT GETTING THRU

            TEST NUMBER  EFFECT                     CAUSE

                 25              10                 RACH NEG.B*DM0 NOT GOING HIGH
                 25              24                 RACH A2 RAB00 NOT GOING LOW
                 25              H                  RACH E59(6) NOT GOING LOW
                                                    OR NGT RACL RADR07
                 26              H                  RACL RADR00 NOT GOING LOW
                 26              4                  EITHER IRCC SM357 STUCK H
                                                    OR RACL E70 BAD
                 26              14                 IRCC C0 RAB03 NOT GOING H OR
                                                    RACL RADR03 INPUT STUCK LOW
                 26              4                  SRC CONST ADDED 1 OR 3

                 27              H                  RACK RADR01 NOT GOING LOW

                 31              10                 RACK A0 RAB01 NOT GOING LOW OR
                                                    NGT RACL RADR01
                 31              4                  RACK BRCAB05 NOT GOING LOW OR
                                                    NGT PACL RADR05
                                                    1$ ON TOP OF STACK
                 31              4                  DST CONST ADDED 1 OR 3
                 32              10                 RACE A0 RAB00 DOES NOT GO LOW

                 33              10                 RACE A0 RAB02 DOES NOT GO LOW

                 34              10                 RACE E44 IS BAD
                 34              10                 IRCB K/ LASS STUXK LOW
                 34              H                  GRAB OBD(*) STUCK H OR NGT RACL E71
```

	34	H	GRAD DRMX00 STUCK H OR GRAB E50 BAD
	35	10	RACE BIN*SM0 H DID NOT GO HIGH
	35	10	IR DECODE ROM WORD BAD
558			
	36	10	RACE BIN*SM0 FAILED
	36	10	IR DECODE ROM WORD BAD
	37	10	RACE E45 BAD
	40	10	RACE A0 RAB00 DOES NOT GO LOW
	40	10	IRCB(JMP+JSR) IS STUCK LOW
	40	H	IRCB FJ CLASS IS STUCK HIGH
	41	10	RACE E45 IS BAD
	42	10	RACE E33 IS BAD
	43	10	RACH J/CLASS NOT GOING HIGH 5$-2 ON TOP OF STACK
	43	10	EITHER RACE E10 BAD OR RACE BIN NOT GOING H 5$ ON TOP OF STACK
	44	10	RACJ AFIR 14(1) NGT RACE E42
	44	10	IRCB E38(6) STUCK HIGH
	45	H	GRAB OBD(0) STUCK H OR NGT RACK E51
	46	10	RACE E33 BAD
	54	10	RACF E3 DOES NOT GO HIGH
	56	10	PART PCLASS FIELD BAD IN IR DECODE ROM
	60	10	PART PCLASS FIELD BAD IN IR DECODE ROM
	62	10	IRCC C0 RAB03 STUCK H OR NGT RACL RADR03 OR FORK C MUX INPUT B0 STUCK H
	65	10	B FORK MUX SELECT STUCK LOW
	65	H	IRCB B0 RAB04 NGT RADR05
	65	H	B FORK MUX INPUT B0 STUCK H OR IRCC B0 RAB00 STUCK H
	65	4	B FORK MUX INPUT B3 OR IRCB B0 RAB00 STUCK L
	65	H	IRCB E46(10) STUCK L-MICRO ADR 170
	67	10	RACE E35(1) BAD
	70	10	B FORK MUX STROBE STUCK L (CHIP FAILURE)
	75	10	RACF JMP+JSR+SWAB NOT GOING HIGH

```
                    75              10          IRCB E63 BAD-R5 CONTAINS 'T67+2''

                    105             4           RACF (HALT:OP CODE 7) DOES NOT GO HIGH
                    105             4           RACE E7 BAD-ODD ADR BIT SET IN ERROR REG


613                 THE FOLLOWING FIVE CONDITION CODE AND BRANCH TESTS ARE A
                    FUNCTION OF RACK F TRUE 1. SECTION 1 OF EACH TEST IS
                    DEPENDENT ON TRUE 1 NOT GOING HIGH, WHILE SECTION 2 IS
                    DEPENDENT ON TRUE ONE GOING HIGH.


620     TEST 1      CCC*BRANCH THRU FET.13

                    THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 1:
                       SECTION 1
                            BCS     E58(13,12)      L,H
                            BMI     E59(10,11,9)    H,L,H
                            BVS     E48(5,3,4)      H,L,H
                            BLOS    E59(4,3,5)      H,L,H

642     TEST 2      SEC*BRANCH THRU FET.13 AND FET.11

                    THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 1:
                       SECTION 1
                            BMI     E58(13,12)      H,L
                            BVS     E58(13,12)      H,L
                       SECTION 2
                            BCC     E58(13,12)      H,H

667     TEST 3      SEV*BRANCH THRU FET.13 AND FET.11

                    THE FOLLOWING IS A LIST OF PATTERNS PUT ON THE GATES OF TRUE 1:
                       SECTION 1
                            BCS     E48(5,3,4)      L,H,H
                            BMI     E48(5,3,4)      H,H,L
                       SECTION 2
                            BVC     E48(5,3,4)      H,H,H

692     TEST 4      SEZ*BRANCH THRU FET.13 AND FET.11

                    THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 1:
                       SECTION 1
                            BCS     E59(4,3,5)      H,H,L
                            BMI     E59(4,3,5)      L,H,H
                       SECTION 2
                            BHI     E59(4,3,5)      H,H,H

717     TEST 5      SEN*BRANCH THRU FET.13 AND FET.11

                    THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 1:
```

```
720                 SECTION 1
                        BLOS    E59(10,11,9)    H,H,L
                        BVS     E59(10,11,9)    L,H,H
                    SECTION 2
                        BPL     E59(10,11,9)    H,H,H

742             THE FOLLOWING SEVEN TESTS ARE A FUNCTION OF RACF TRUE 2.
            SECTION 1 OF EACH TEST IS DEPENDENT ON TRUE 2 NOT GOING HIGH
            WHILE SECTION 2 IS DEPENDENT ON TRUE 2 GOING HIGH.

            TEST 6  BRANCHES THRU FET.13

                    THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
                       SECTION 1
                            BLE     E58(2,1)H,L     F59(13,1,2)L,H,H        E48(1,13,2)H,H,L
                            BLT     E59(13,1,2)L,H,H        E48(1,13,2)H,H,L        E58(10,9)L,H
                            BEQ     E58(2,1)H,L     E58(10,9)H,L

764         TEST 7  BRANCH THRU FET.13 AND FET.12

                    THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
                       SECTION 1
                            BEQ     E48(1,13,2)     L,H,H
                       SECTION 2
                            BGT     E48(1,13,2)     H,H,H

788         TEST 10 BRANCH THRU FET.13 AND FET.12

                    THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
                       SECTION 1
                            BLT     E58(2,1)        L,H
                       SECTION 2
                            BNE     E58(2,1)        H,H

811         TEST 11 BRANCH THRU FET.13 AND FET.12

                    THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
                       SECTION 1
                            BEQ     E59(13,1,2)     H,H,L
                       SECTION 2
                            BGE     E59(13,1,2)     H,H,H

834         TEST 12 BRANCHES THRU FET.13 AND FET.12

                    THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
                       SECTION 1
                            BEQ     E58(2,1)H,L     E58(10,9)H,L
                            BLT     E59(13,1,2)H,L,H        E48(1,13,2)H,L,H        E58(10,9)L,H
```

```
850     TEST 13 UNIARY AND BINARY (SM0)

                THE FOLLOWING TEST TESTS ALL THE E/CLASS
                INSTRUCTIONS WITH A DESTINATION MODE OF 0
                AND DESTINATION FIELD OF NOT 7. THIS CLASS CONSISTS
                OF ALL THE UNIARY (EXCEPT NEG) INSTRUCTIONS AND
                ALL THE BINARY INSTRUCTIONS WITH A SOURCE MODE
                OF 0.

1143    TEST 14 REGISTER SELECTION TEST

                THIS TEST ENSURES THAT THE 6 ADDRESS LINES INTO
                THE GENERAL PURPOSE REGISTERS (GPR) ARE NOT STUCK.
                THE LABELS OF THE ADDRESS LINES ARE:
                        GSAX    GENERAL SOURCE ADDRESS LINE
                        GDAX    GENERAL DESTINATION ADDRESS LINE
                WHERE X STANDS FOR LINE 0,1, OR 2.
                THE CLASSES OF ERRORS DESCRIBED IN THIS TEST

1152            ARE DEFINED AS FOLLOWS:
                        CLASS A=GDAX OK
                                GSAX STUCK
                        CLASS B=GSAX OK
                                GDAX STUCK
                        CLASS C=GSAX STUCK
                                GDAX STUCK

1248    TEST 15 GPR1 STUCK BIT TEST

                LOADS GPR1 WITH ZEROS AND ONES AND COMPARES R1 SOURCE AND
                DESTINATIONS WITH R0. IF THE COMPARISON FAILS A BIT IS STUCK.

1274    TEST 16 GPR2 STUCK BIT TEST

                LOADS GPR2 WITH ZEROS AND ONES AND COMPARES R2 SOURCE AND
                DESTINATION WITH R0.

1300    TEST 17 GPR3 STUCK BIT TEST

                LOADS GPR3 WITH ZEROS AND ONES AND COMPARES
                R3 SOURCE AND DESTINATION WITH R0.

1326    TEST 20 GPR4 STUCK BIT TEST

                LOADS GPR4 WITH ZEROS AND ONES AND COMPARES
                R4 SOURCE AND DESTINATION WITH R0.

1352    TEST 21 GPR5 STUCK BIT TEST

                LOADS R5 WITH ZEROS AND ONES AND COMPARES
                R5 SOURCE AND DESTINATION WITH R0.
```

1378 TEST 22 GPR6 STUCK BIT TEST

LOADS R6 WITH ZEROS AND ONES AND COMPARES
R6'SOURCE AND DESTINATION WITH R0.

1404 TEST 23 GPR SHORTED BIT TEST

TEST IF GPR'S 1 THRU 6 HAVE TWO BITS TIED TOGETHER

NOTE: R0 IS CONSIDERED 'HARDCORE''

****

1509

****

1511 TEST 24 ONE MICROSTATE (E/CLASS*DM0*DF7)

THIS TEST EXECUTES AN ADD INSTRUCTION WITH SM0,DM0, AND DF7.
IF THE TEST FAILS THE SAME FLOW (EXC. 90) IS
TRIED WITH A PENDING BRQ (T BIT TRAP) INSTEAD OF A DF7. IF THIS TEST FAILS
A FORK A FAILURE IS REPORTED. IF IT PASSES, A TEST 1 FAILURE IS REPORTED.

ROM FLOW-30

****

1559

****

1561 TEST 25 TWO MICROSTATES (NEG*DM0)

THIS TEST EXECUTES A NEGATE INSTRUCTION WITH DM0.

IF FORK A FAILS EXECUTION WOULD GO TO EITHER RSD.00, ZAP.00,
OR FOP.00.
FOP.00 WILL CAUSE THE PROCESSOR TO HANG.
RSD.00 WOULD CAUSE A TRAP TO LOCATION 10. THIS WOULD ONLY HAPPEN
IF RACH NEG.B*DM0 DID NOT GO HIGH.
ZAP.00 WOULD CAUSE A TRAP TO LOCATION 24. THIS WOULD ONLY HAPPEN
IF RACH A2 RAB00 DID NOT GO LOW.

ROM FLOW-301,210

```
                ;*********************************************************************
        1616
                ;*********************************************************************

                1618    TEST 26 THREE MICROSTATES (BIN*SM1*DM0*-DF7*SR0(0)

                                IF FORK A FAILS EXECUTION WILL GO TO EITHER EXEC.80 OR D12.00.
                                EXC.80 WILL HANG THE PROCESSOR IN THE PAUSE STATE AT MICRO ADDRESS 343.
                                THIS WILL ONLY HAPPEN IF RACL RADR00 IS NOT GOING LOW DUE
                                TO RACF A1 RAB00 (AFIR59(1)*[-BIN+SM01]*U/CLASS).
                                D12.00 WOULD MOV THE PC TO LOCATION 0.

                                IF FORK C FAILS EXECUTION WOULD GO FROM S13.10 TO D00.80 OR
                                D45.01 OR S13.20 OR D12.00 OR JSR.10 OR ASC.80 OR RTI.50 OR ASH.20 OR FOP.50.
                                D00.80 WOULD SWAP THE BYTES OF THE SOURCE OPERAND BEFORE
                                PUTTING THEM IN R5.
                                D45.01 WOULD MOVE THE PC TO LOCATION 0.
                                S13.20 WILL EXECUTE A SM3 (AND NO AUTO INC) INSTR.  THIS WILL CAUSE
                                AN ODD ADDRESS TRAP SINCE LOCATION POSERR CONTAINS AN ODD WORD.
                                JSR.10 WOULD PUSH THE ADDR OF POSERR ONTO THE STACK.
                                ASC.80 WILL HALT AT 8$.
                                RTI.50 WILL CAUSE 1004XX TO BE PLACED IN THE PS WORD
                                AND THE PROCESSOR WILL TRAP TO LOCATION 14.
                                FOP.50 WILL ?????.

                1638            ASH.20 WOULD CAUSE A BAD CC.
                                IF THE SRC CONST FAILS IN STATE S13.00 AND ADDS 1 OR 3, AN ODD
                                ADDRESS TRAP WILL OCCUR.
                                IF THE SRC CONSTANT ADDS 2, THE ERROR AT 6$ WILL REPORT THE FAILURE.

                                ROM FLOW-21,27,205

                1700    TEST 27 THREE MICROSTATES (BIN*SM2*DM0*-DF7*SR0(0)

                                THIS TEST IS THE SAME AS THE PREVIOUS TEST EXCEPT FOR THE
                                SM. A WORD AND BYTE INSTRUCTION IS EXECUTED TO VERIFY THAT STATE
                                S13.01 ADDS THE CORRECT SOURCE CONSTANT.

                                IF FORK A FAILS EXECUTION WILL GO TO EXC.80.
                                EXC.80 WILL HANG THE PROCESSOR IN THE PAUSE STATE AT MICRO ADDRESS 343.
                                THIS WILL ONLY HAPPEN IF RACL RADR01
                                IS NOT GOING LOW DUE TO RACF A1 RAB01 (AFIR10(1)*U/CLASS).

                                ROM FLOW-22,27,205

                1752    TEST 30 ALU CARRY FUNCTIONAL TEST

                                THIS TEST DOES A COMPLETE CHECK OF THE ALU CARRY FUNCTIONS
                                THE FIRST SECTION ENSURES THAT ALL THE INPUT AND OUTPUT LINES ARE
                                OK AND THE REST OF THE TEST ENSURES THAT THE CARRY LOGIC IS OK.
                                FOLLOWING ARE THE LOGIC EQUATIONS FOR A 74S181 AND 74S182 THAT
                                DICTATED THE PATTERNS USED IN EACH SECTION:
                                    74S181
                                        G=A3*B3+A2*B2*(A3+B3)+A1*B1*(A2+B2)*(A3+B3)+A0*B0*(A1+B1)*(A2+B2)*(A3+B3
```

```
                        P=(A3+B3)*(A2+B2)*(A1+B )*(A0+BC)
                        COUT=G+P*CIN
                    74S182
                        CX=G0+P0*CIN
                        CY=G1+P1*G0+P1*P0*CIN
                        CZ=G2+P2*G1+P2*P1*G0+P2*P1*P0*CIN

1882    TEST 31 THREE MICROSTATES (DAC*DM2*0/CLASS)

                IF FORK A FAILS EXECUTION WILL GO TO RSD.00. THIS WILL ONLY HAPPEN
                IF RACE A0 RAB01 DOES NOT GO LOW OR DOES NOT GET THRU
                TO RACL RADR01.  THIS WILL CAUSE A TRAP TO LOCATION 10.

                IF BEN15 FAILS EXECUTION WILL GO TO D45.80.
                THIS WILL CAUSE A TRAP TO 4 WITH THE ADDRESS OF 1$ ON THE STACK.

                IF THE DESTINATION CONSTANT FAILS(ADDS 1 OR 3) IN STATE D12.60
                AN ODD ADDRESS TRAP WILL OCCUR.
                IF THE DST CONST ADDS 0, THE ERROR AT 3$ WILL REPORT THE FAILURE.
                IF THE DESTINATION IS NOT LOADED WITH THE SOURCE, THE
                ERROR AFTER 5$ WILL REPORT THE FAILURE.

                ROM FLOW-2,155,312

1943    TEST 32 THREE MICROSTATES (DAC*DM1*0/CLASS)

                THIS TEST IS THE SAME AS THE PREVIOUS TEST EXCEPT FOR THE DM.
                IF FORK A FAILS, EXECUTION WILL GO TO RSD.00.
                THIS WILL CAUSE A TRAP TO LOCATION 10 WITH AN ODD ADDRESS ERROR.
                THIS WILL ONLY HAPPEN IF RACE A0 RAB00 DOES NOT GO LOW OR
                DOES NOT GET TO RACL.
                EITHER E44 OR E6 IS BAD.

                IF THE INSTRUCTION FAILS TO MOVE R5 TO THE PC INDIRECT,
                THE ERROR AFTER 1$ WILL REPORT THE FAILURE.

                ROM FLOW-1,155,312

1976    TEST 33 THREE MICROSTATES (DAC*DM4*0/CLASS)

                IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
                THIS WILL CAUSE A TRAP TO LOCATION 10. THIS WILL ONLY HAPPEN IF
                RACE A0 RAB02 IS NOT GOING LOW. EITHER RACE E33 OR
                E6(9&8) IS BAD.

                IF THE DST CONST FAILS TO SUBTRACT 2, THE ERROR AT EITHER 1$
                OR 1$-2 WILL REPORT THE FAILURE.

                IF BEN01 FAILS (CAUSED BY IRCD DM357 STUCK HIGH)
                EXECUTION WILL GO TO D10.00 WHICH WILL EXECUTE A MODE 5
                INSTEAD OF MODE 4.

                IF THE DESTINATION IS NOT LOADED PROPERLY,
                THE ERROR AT 3$ WILL REPORT THE FAILURE.
```

ROM FLOW-4,122,157

2037 TEST 34 THREE MICROSTATES (DAC*DM1*TST.B*DR0(0))

IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
THIS WILL ONLY HAPPEN IF RA SAME?

2055

IF BEN15 FAILS EXECUTION WILL GO TO STATE D12.60.

IF B FORK FAILS (AFTER STATE D12.10) CAUSED BY IRCB
K/CLASS STUCK LOW EXECUTION WILL GO TO STATE RSD.00 CAUSING A TRAP TO 10.
IF IRCB B1 RAB00 IS STUCK HIGH EXECUTION WILL GO FROM
D12.10 TO JSR.40. THIS WILL CAUSE THE PROCESSOR TO HANG.
IF EITHER GRAB 0BD(1) IS STUCK HIGH OR NOT GETTING THRU TO RACL E71,
EXECUTION WILL GO TO STATE D12.30.
IF GRAD DRMX00 IS STUCK HIGH OR GRAB E50 IS BAD, EXECUTION
WILL GO TO D10.60 WHICH WILL HANG THE PROCESSOR.

ROM FLOW-1,175,33

2093 TEST 35 THREE MICROSTATES (DAC*DM1*BIT.B*DR0(0))

THIS TEST IS THE SAME AS THE PREVIOUS TEST EXCEPT A BIT INSTRUCTION IS USED.
IF FORK A FAILS RACE BIN*SM0 H FAILED.

IF FORK B FAILS THE INSTRUCTION DECODE ROM WORD IS BAD.

IF THE RESULTANT DATA IS BAD STATE TST.10 FAILED.

ROM FLOW-1,175,33

2116 TEST 36 THREE MICROSTATES (DAC*DM1*CMP.B*DR0(0))

THIS TEST IS THE SAME AS THE PREVIOUS TWO TESTS
EXCEPT A CMP INSTRUCTION IS USED.

ROM FLOW-1,175,33

2135 TEST 37 THREE MICROSTATES (DAC*DM2*TST.B*DR0(0))

IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
THIS WILL ONLY HAPPEN IF RACE E45 IS BAD (AFIR04(1)*R/CLASS).

BEN15 & FORK B HAVE ALREADY BEEN TESTED
IF THE AUTO INC FAILS IT WILL BE DUE TO A BAD
FIELD IN ROM STATE D12.10.

ROM FLOW-2,175,33

```
2165    THE LOGICAL SEQUENCE WOULD NEXT TEST THE BIT.B AND CMP.B INSTRUCTIONS BUT
        THESE WILL NOT BE TESTED WITH INDIVIDUAL TESTS SINCE THEY DO NOT USE
        ANY HARDWARE THAT HAS NOT ALREADY BEEN TESTED.

2172    TEST 40 THREE MICROSTATES (JMP*DM1)

                IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
                THIS WILL ONLY HAPPEN IF RACE A0 RAB00 DOES NOT GO LOW
                (AFIR03(1)*[JMP+JSR+SWAB]).


2178            A FORK B FAILURE WOULD BE ONE OF THE FOLLOWING:
                IF IRCB (JMP+JSR) IS STUCK LOW EXECUTION WILL GO TO RSD.00 CAUSING
                A TRAP TO 10.
                IF IRCB IR(14:9) 04 IS STUCK LOW EXECUTION WILL
                GO TO JSR.00 WHICH WILL EXECUTE A JSR INSTEAD OF A JMP.
                IF IRCB B FORK MUX FAILS EXECUTION WILL GO TO
                FOP.00.
                IF IRCB FJ CLASS IS STUCK HIGH EXECUTION WILL GO TO
                STATE D12.00 AND THE JMP WON'T JUMP.

                IF THE INSTRUCTION FAILS TO JUMP, STATE JMP.00 IS REPORTED AS BAD.

                ROM FLOW-1,135,35

2212    TEST 41 THREE MICROSTATES (JMP*DM2)

                THIS TEST IS THE SAME AS THE PREVIOUS TEST EXCEPT THE DM-2.
                IF FORK A FAILS RAC E45 IS BAD (AFIR04(1)*[JMP+JSR+SWAB]).

                ROM FLOW-2,135,35

2229    TEST 42 THREE MICROSTATES (JMP*DM4)

                IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.

2232            THIS WILL ONLY HAPPEN IF RACE E33(AFIR05(1)*[JMP+JSR+SWAB]) IS BAD.

                ALL OTHER LOGIC HAS BEEN TESTED.

                ROM FLOW-4,122,35

2246    TEST 43 THREE MICROSTATES (SOB)

                IF RACH U/CLASS IS NOT GOING HIGH EXECUTION WILL GO TO RSD.00
                CAUSING A TRAP TO LOCATION 10 WITH THE ADDRESS OF 5$-2
                ON THE TOP OF THE STACK.
                IF EITHER RACE E10 IS BAD OR RACE BIN DOES NOT GO HIGH
                EXECUTION WILL GO TO D67.01. EXECUTION WILL EVENTUALL GO
                TO RSD.00 AFTER STATE D10.60 AND THE STACKED PC WILL BE AT 5$.
                IF RACF E8 FAILS EXECUTION WILL GO TO ASC.10 WHICH WILL
                PERFORM AN ASHC*DM0 OPERATION.

                IF THE SOB BRANCHES WHEN IT IS NOT SUPPOSE TO EITHER
```

GRAE SR EQ ONE IS STUCK HIGH OR RACK E63 IS BAD
OR STATE SOB.10 DOES NOT RESTORE THE OLD PC.

IF THE SOB DOES NOT BRANCH WHEN IT IS SUPPOSE TO EITHER
STATE SOB.00 IS BAD OR GRAE SR EQ ONE STUCK LOW OR
RACK E63(C1) IS BAD.

IF THE REGISTER DOES NOT DECREMENT STATE SOB.20 IS BAD.

ROM FLOW-57,242/262.262

```
        ************************************************************************
2311
        ************************************************************************
```

2313 TEST 44 FOUR MICROSTATES (DAC*DM12*P/CLASS*DR0(0)

IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
THIS WILL ONLY HAPPEN IF RACJ AFIR 14(1) DOES NOT
GET THRU RAC E42.

THE FOLLOWING COULD BE FORK B FAILURES:
IF IRCB B0 RAB00 IS STUCK HIGH OR NOT GETTING THRU
TO RACL RADR00 EXECUTION WILL GO TO EXC.90 WHICH WOULD
PUT THE RESULT INTO A REGISTER RATHER THAN MEMORY.
IF IRCB E38(6) IS STUCK HIGH EXECUTION WILL GO TO
RSD.00 CAUSING A TRAP TO 10.
IF THE BIC DOES NOT HAPPEN THEN EXC.00 IS BAD.

ROM FLOW-2,175,31,132

2358 TEST 45 FOUR MICROSTATES (DAC*DM12*TST.B*DR0(1))

AFTER STATE D12.10 IF EITHER GRAB OBD(1) DOES NOT GO HIGH
OR DOES NOT GET THRU RACL E71 EXECUTION WILL GO
TO TST.10. IF EITHER GRAB OBD(0) IS STUCK HIGH OR NOT GETTING
THRU RACK E41, EXECUTION WILL GO TO D10.60. THIS WILL CAUSE
THE PROCESSOR TO HANG UP IN THE PAUSE STATE AT MICRO
ADDRESS 177. IF THE TEST FAILS THEN STATE D12.30 FAILED.

ROM FLOW-1,175,137,33

2382 TEST 46 FOUR MICROSTATES (DAC*DM4*TST.B*DR0(0))

IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
THIS WILL ONLY OCCUR IF RACE E33(AFIR05(1)*R/CLASS) IS BAD.

AFTER D10.30 IF IRCB FJ/CLASS DOES NOT GET TO RACL E71
OR IF RACL E71 IS BAD EXECUTION WILL GO TO SVC.50.

IF THE INSTRUCTION DOESN'T WORK THEN D10.60 IS BAD.

ROM FLOW-4,122,177,33

THE LOGICAL SEQUENCE HERE WOULD BE TO TEST THE BIT & CMP INSTRUCTIONS
BUT NO ADDITIONAL LOGIC IS TESTED BY THEM.

TEST 47 FOUR MICROSTATES (DAC*DM6*0/CLASS)

FORK A SHOULD NOT FAIL ON THIS TEST.

BEN01 SHOULD NOT FAIL.

BEN15*FEN2 SHOULD NOT FAIL.

IF D67.00 FAILS TO INCREMENT THE PC AN RTI INSTRUCTION
WILL BE EXECUTED.
IF D67.00 FAILS TO CLOCK THE BR THE INSTRUCTION
WILL BE ADDED AS THE INDEX WORD.
IF D67.10 FAILS TO ADD THE INDEX NUMBER THE SOURCE
WILL BE PUT IN THE WRONG LOCATION.

ROM FLOW-6,251,122,157

TEST 50 FOUR MICROSTATES (BIN*SM12*DM0*-DF7*SR0(1))

IF FORK A FAILS EXECTUTION WILL GO TO STATE D12.00.
THIS WILL HAPPEN IF RACE BF1=7 DOES NOT GO HIGH.
STATE D12.00 WOULD BITB R5 & THE CONTENTS OF LOCATION 200
WHICH IS 137.

THE FOLLOWING COULD BE BEN14*FORK C FAILURES:
IF IRCC CO RAB00 DOES NOT GO HIGH EXECUTION WILL GO TO
D00.90 WHICH WILL TEST THE LOW BYTE INSTEAD OF THE HIGH BYTE.

IF STATE D00.80 FAILS TO SWAP THE BYTES IT WILL LOOK LIKE
A FORK C FAILURE.

ROM FLOW-21,27,204,205

TEST 51 FOUR MICROSTATES (BIN*SM12*DM0*DF7*SR0(0))

IF FORK A FAILS EXECUTION WILL EITHER GO TO STATE D12.00 OR EXC.80.

STATE D12.00 WOULD ADD R5 TO THE CONTENTS OF THE PC.
STATE EXC.80 WOULD NOT CHANGE THE PC.

IF FORK C FAILS EXECUTION WILL EITHER GO TO JSR.10 OR ASC.80.
JSR.10 WILL CAUSE R5 TO BE STACKED, THE PC TO BE PUT
IN R5, AND THE PC REPLACED BY R5 WHICH WILL CAUSE AND RTI SINCE
THE CONTENTS OF THE LOCATION POINTED TO BY R5 IS 000002.
ASC.80 WILL CAUSE THE ADD TO LOOK LIKE IT FAILED.

IF STATE D07.10 FAILS TO LOAD THE SHFTR THE PC WILL
DOUBLE AND THE PROGRAM WILL BLOW UP.

IF THE SHFTR FAILS TO BE PUT IN THE SR THE PC
WILL NOT CHANGE.

ROM FLOW-21,27,203,30

2567 TEST 52 FOUR MICROSTATES (BIN*SM12*DM0*DF7*SR0(1))

IF FORK A FAILS EXECUTION WILL GO TO D12.00.

FORK C SHOULD NOT FAIL SINCE THE LOGIC HAS ALREADY BEEN TESTED.

IF STATE D07.00 FAILS TO SWAP THE BYTES THE CC'S WILL
BE BAD.
IF D07.00 FAILS TO LOAD THE SHIFTER, THE THIRD CMPB WILL FAIL.
IF D07.00 FAILS TO LOAD THE SR THE SECOND CMPB WILL FAIL.

ROM FLOW-21,27,202,30

2604 TEST 53 FOUR MICROSTATES (BIN*SM4*DM0*-DF7*SR0(0))

IF FORK A FAILS EXECUTION WILL
GO TO D45.00 WHICH WILL EXECUTE A SM0*DM4 INSTRUCTION EXCEPT
THE DESTINATION REGISTER WILL NOT DECREMENT.

2610 FORK C WILL NOT FAIL SINCE IT HAS ALREADY BEEN TESTED.

IF THE SRC FAILS TO AUTO DECREMENT STATE S45.00 IS BAD.

ROM FLOW-24,23,27,205

2639 TEST 54 FOUR MICROSTATES (RTS)

IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
THIS WILL ONLY HAPPEN IF RACF E3 DOES NOT GO HIGH.

IF THE PC OR R5 FAILS THE TEST WILL HALT.

ROM FLOW-40,223,224,342

2661 TEST 55 FOUR MICROSTATES (JMP*DM6)

IF FORK A FAILS EXECUTION WILL GO TO STATE D12.01.

2664 THIS WOULD CAUSE A JMP*DM1 TO EXECUTE.

NEITHER BEN01 NOR BEN15*FEN2 SHOULD FAIL SINCE THEY HAVE ALREADY BEEN
TESTED.

ROM FLOW-6,251,122,35

```
                ;*****************************************************************
        2681
                ;*****************************************************************

                2683    TEST 56 FIVE  MICROSTATES (DAC*DM12*P/CLASS*DR0(1))

                                FORK A SHOULDN'T FAIL.

                                BEN15 SHOULDN'T FAIL SINCE THIS LOGIC HAS BEEN TESTED.
                                NEITHER SHOULD BEN05*FEN2.

                                IF FORK B FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
                                THIS FAILURE WOULD BE CAUSED BY A BAD FIELD (PART PCLASS)
                                IN THE IR DECODE ROM.

                                ROM FLOW-1,175,137,31,132

                2708    TEST 57 FIVE  MICROSTATES (DAC*DM3*0/CLASS)

                                FORK A SHOULD NOT FAIL SINCE THE LOGIC HAS ALREADY BEEN TESTED.

                                IF THE DR DOES NOT AUTO INC THEN STATE D30.10 IS BAD.

                                IF THE CONDITION CODES ARE BAD THEN EITHER STATE D10.50
                                DID NOT LOAD THE BR OR THE DOUBLE DEFERED DIDN'T WORK.

                                ROM FLOW-3,221,233,311,157

                2741    TEST 60 FIVE  MICROSTATES (DAC*DM4*P/CLASS*DR0(0))

                                FORK A SHOULD NOT FAIL.

                                IF FORK B FAILS, AFTER D10.60, EXECUTION WILL GO TO
                                RSD.00 CAUSING A TRAP TO 10. THIS WILL ONLY HAPPEN IF
                                THE IR DECODE ROM HAS A BAD FIELD (PART PCLASS).

                                ROM FLOW-4,122,177,31,132

        2765    THE LOGICAL SEQUENCE AT THIS POINT WOULD BE TO EXECUTE A DAC*DM4*[TST.B+
                BIT.B+CMP.B]*DR0(1) INSTRUCTION FOLLOWED BY A DAC*DM6*[TST.B+BIT.B+CMP.B]*
                DR0(0) INSTRUCTION TEST BUT NO ADDITIONAL LOGIC IS TESTED.

        2774    THE LOGICAL SEQUENCE AT THIS POINT WOULD BE TO EXECUTE A BIN*SM4*DM0*-DF7*SR0(1)
                FOLLOWED BY A BIN*SM4*DM0*DF7*SR0(0)
                FOLLOWED BY A BIN*SM4*DM0*DF7*SR0(1) INSTRUCTION BUT NO ADDITIONAL LOGIC IS TESTED.

        2781    TEST 61 FIVE  MICROSTATES (BIN*SM6*DM0*-DF7*SR0(0))

                                IF FORK A FAILS EXECUTION WILL GO TO STATE D67.00
                                WHICH WOULD EXECUTE A SM0*DM6 INSTRUCTION.

                                BEN14*FEN4 SHOULD NOT FAIL SINCE IT HAS ALREADY BEEN TESTED

                                ROM FLOW-26,54,141,142,205
```

```
2807    TEST 62 FIVE  MICROSTATES (BIN*SM12*DM12*0/CLASS)

                IF FORK A FAILS EXECUTION WILL GO TO D12.01.THIS WOULD CAUSE R5
                TO BE WRITTEN INTO $TMP2.

                IF FORK C FAILS EXECUTION WOULD GO TO ONE OF THE
                FOLLOWING STATES: D45.80,D30.80,FOP.00,WAT.00, D00.90, AND ASH.20.
                STATE D45.80 WOULD EXECUTE A SM2*DM4 INSTEAD OF SM2*DM1.
                STATE D30.80 WOULD EXECUTE A SM1*DM3.
                STATE FOP.00 WOULD CAUSE A TRAP TO LOCATION 10.
                THIS WILL ONLY HAPPEN IF EITHER IRCC C0 RAB03 IS STUCK
                OR IT IS NOT GETTING THRU RACL RADR03 OR
                IRCC FORK C MUX INPUT B0 IS HIGH.
                THE LA30 PRINTER BUFFER WILL BE SET UP TO GENERATE AN INTERRUPT IN
                CASE THE TEST FAILS TO THE WAT.00 STATE.
                STATE D00.90 WILL EXECUTE A DM0 INSTEAD OF A DM1.
                STATE ASH.20 WILL CLEAR THE C BIT.

                ROM FLOW-22,27,111,155,312

2900    THE LOGICAL FLOW AT THIS POINT WOULD TEST A BIN*SM12*DM12*SR0(0)*DR0(0)
        *[TST.B+BIT.B+CMP.B] INSTRUCTION BUT NO ADDITIONAL LOGIC WOULD BE TESTED.

2906    TEST 63 FIVE  MICROSTATES (BIN*SM12*DM12*SR0(1)*DR0(0)*CMPB)

                THE ONLY THING THAT SHOULD FAIL WOULD BE STATE D12.90(110).

                ROM FLOW-21,27,110,175,33

2925    TEST 64 FIVE  MICROSTATES (BIN*SM12*DM4*0/CLASS)

                WHICH WILL EXECUTE A SM1*DM2 TYPE INSTRUCTION.

                IF THE INSTRUCTION FAILS EITHER D45.80 OR D40.20 FAILED.

                ROM FLOW-21,27,115,121,157

        ************************************************************************
2950
        ************************************************************************

2952    TEST 65 SIX   MICROSTATES (DAC*DM12*ASRB*DR0(1))

                NEITHER FORK A NOR BEN15 NOR BEN05*FEN2 SHOULD FAIL
                SINCE THEY HAVE ALREADY BEEN TESTED.

                IF FORK B FAILS AFTER D12.30 EXECUTION WILL GO TO
                ONE OF THE FOLLOWING: RSD.00,D45.00,EXC.00,S45.00,
                CCP.00,MUL.00,SVC.10,MFP.00 OR DEP.00.
                RSD.00 WILL CAUSE A TRAP TO LOCATION 10.
                THIS WILL HAPPEN IF THE B FORK MUX SELECT IS STUCK LOW.
                IF STATE D45.00 IS ENTERED THE PROCESSOR WILL HANG UP
                IN A LOOP BETWEEN STATES D45.00 AND D10.30.
                IF STATE S45.00 IS ENTERED EXECUTION WILL GO TO STATE
```

D12.80 AFTER S13.10 WHICH WILL HANG UP THE PROCESSOR.
STATE CCP.00 WOULD SET OR CLEAR THE CONDITION CODES
ACCORDING TO IR(4:0).
STATE MUL.00 WOULD CAUSE THE DESTINATION OPERAND TO
BE MULTIPLIED BY REGISTER 2 AND THE RESULT WOULD BE
STORED IN REGISTER 2 AND 3.
IF SVC.10 IS ENTERED A TRAP TO 4 WILL OCCUR BECAUSE
THE DESTINATION REGISTER IS ODD. THIS WILL ONLY HAPPEN
IF EITHER B FORK MUX INPUT B3 OR IRCB B0 RAB00 IS STUCK LOW.
IF MFP.00 IS ENTERED AN MFPI INSTRUCTION WILL BE EXECUTED
THIS WILL PUSH THE ADDRESS OF 1$ ONTO THE STACK.
DEP.00 WILL CAUSE THE PROCESSOR TO HANG IN MICRO ADDRESS
170 WITH THE RUN LIGHT ON.

ROM FLOW-1,175,137,64,123,132

3028 TEST 66 SIX MICROSTATES (DAC*DM12*RORB*DR0(1))

THIS TEST IS THE SAME AS THE LAST ONE EXCEPT A RORB
IS USED INSTEAD OF AN ASRB.

FORK B WILL ONLY FAIL IF IRCB E36(13) IS STUCK HIGH
WHICH WILL CAUSE EXECUTION TO GO TO EXC.00.

ROM FLOW-2,175,137,64,123,132

3053 THE LOGICAL FLOW AT THIS POINT WOULD TEST A DAC*DM3*[TST.B+BIT.B+CMP.B]*DR0(0)
FOLLOWED BY A DAC*DM4*P/CLASS*DR0(0) INSTRUCTION BUT NO ADDITIONAL LOGIC
IS TESTED

3061 TEST 67 SIX MICROSTATES (DAC*DM6*XOR*DR0(0))

IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
THIS SHOULD ONLY HAPPEN IF RACE E35(1) IS BAD.

IF THE INSTRUCTION DOESN'T WORK IT WILL HALT IN THIS TEST.

ROM FLOW-6,251,122,177,31,132

3086 THE LOGICAL SEQUENCE WOULD TEST A DAC*DM6*[TST.B+BIT.B+CMP.B]*DR0(1) BUT ALL
THE LOGIC HAS BEEN TESTED.

3093 TEST 70 SIX MICROSTATES (NEG.B*DM12*DR0(0))

NEITHER FORK A NOR BGN15 SHOULD FAIL.

3096

IF FORK B FAILS EXECUTION WILL GO TO RSD.00 CAUSING
A TRAP TO LOCATION 10 OR EXC.00.
RSD.00 SHOULD ONLY OCCUR IF THE B FORK MUX
STROBE IS BEING HELD LOW(CHIP FAILURE).

ROM FLOW-1,175,67,271,163,132

3135 THE LOGICAL SEQUENCE WOULD NEXT EXECUTE A JMP*DM3 BUT NO
ADDITIONAL LOGIC IS TESTED.

3141 TEST 71 SIX MICROSTATES (BIN*SM3*DM0*-DF7*SR0(0))

FORK A SHOULD NOT FAIL.

IF BEN14*FEN4 FAILS EXECUTION WILL GO TO D00.90. THIS
WILL CAUSE A SM2 TO BE EXECUTED. THIS SHOULD ONLY
HAPPEN IF IRCC SM357 IS STUCK LOW OR NOT GETTING THRU RACL E70.

ROM FLOW-22,27,317,143,146,205

3170 THE LOGICAL SEQUENCE WOULD NEXT TEST A BIN*SM6*DM0*DF7*SR0(1), THEN A BIN*SM12*
DM12*SR0(0)*DR0(1)*[TST.B+BIT.B+CMP.B] THEN A BIN*SM12*DM12*SR0(0)*
DR0(0)*P/CLASS THEN A BIN*SM12*DM12*SR0(1)*DR0(1)*[TST.B+BIT.B+CMP.B]
THEN A BIN*SM12*DM12*SR0(1)*DR0(0)*P/CLASS AND THEN A BIN*SM12*DM4*
SR0(0)*DR0(0)*[TST.B+BIT.B+CMP.B] INSTRUCTION, BUT NO ADDITIONAL LOGIC
IS TESTED.

3180 TEST 72 SIX MICROSTATES (BIN*SM12*DM4*SR0(1)*DR0(0)*CMPB)

A FAILURE WILL ONLY OCCUR IF STATE D45.90 FAILS.

IF THE DST REG. DOES NOT DECREMENT STATE D45.90 IS BAD.
IF THE CONDITION CODES ARE BAD THEN STATE D40.30
PROBABLY DID NOT SWAP THE BYTES OF THE SRC OPERAND.

ROM FLOW-1,27,114,131,177,33

3212 THE LOGICAL FLOW WOULD NEXT TEST A BIN*SM4*DM12*SR0(0)*DR0(0)*O/CLASS
THEN A BIN*SM4*DM12*SR0(0)*DR0(0)*[TST.B+BIT.B+CMP.B] THEN A BIN*SM4*
DM12*SR0(1)*DR0(0)*[TST.B+BIT.B+CMP.B] THEN A BIN*SM4*DM4*SR0(0)*DR0(0)*
O/CLASS INSTRUCTION, BUT NO ADDITIONAL LOGIC IS TESTED.

3220 TEST 73 SIX MICROSTATES (DAC*DM5*DR0(0)*O/CLASS)

FORK A SHOULD NOT FAIL.

IF IRCD DM357 IS STUCK LOW OR NOT GETTING THRU
TO RACK E51 A DM4 WILL BE EXECUTED.
IF STATE D10.00 OR D10.10 FAIL TO FETCH THE DEFERED ADDRESS
THE SOURCE WILL BE STORED IN THE DESTINATION.

ROM FLOW-5,162,231,233,311,157

3252 THE LOGICAL SEQUENCE WOULD NEXT TEST A DAC*DM3*DR0(0)*P/CLASS THEN A
DAC*DM3*DR0(1)*[TST.B+BIT.B+CMP.B] THEN A DAC*DM4*DR0(1)*[ASRB+
RORB] THEN A DAC*DM5*DR0(0)*[TST.B+BIT.B+CMP.B] THEN A DAC*DM6*
DR0(1)*P/CLASS INSTRUCTION, BUT NO ADDITIONAL LOGIC IS TESTED.

```
3259
```

```
3261    TEST 74 SEVEN MICROSTATES (DAC*DM7*0/CLASS)

                FORK A SHOULD NOT FAIL.

                IF IRCD DM357 DOES NOT GO HIGH A DM6 WILL BE EXECUTED.

                ALL OTHER LOGIC HAS BEEN TESTED.

                ROM FLOW-7,251,162,231,233,31',157

3292    THE LOGICAL SEQUENCE WOULD NEXT TEST A NEG.B*DM12*DR0(1) THEN A NEG.B*DM4*DR0(0)
        THEN A JMP*DM5 INSTRUCTION, BUT NO ADDITIONAL LOGIC IS TESTED.

3298    TEST 75 SEVEN MICROSTATES (JSR*DM12)

                IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO LOCATION 10.
                THIS WILL ONLY OCCUR IF RACE JMP+JSR+SWAB DOES NOT GO HIGH.

                IF EITHER IRCB IR(14:9)04 DOES NOT GO LOW OR E63 IS BAD
                EXECUTION WILL GO FROM D12.10 TO EXC.00.
                IF IRCB E63 IS BAD (PIN 10 OR 485 FLOATING) EXECUTION WILL
                GO TO RSD.00 CAUSING A TRAP TO LOCATION 10. THIS FAILURE
                WOULD INCREMENT THE DST REG. BEFORE THE TRAP.

                IF THE INSTRUCTION FAILS THEN ONE OF THE JSR STATES FAILED.

                ROM FLOW-2,135,34,201,274,275,32

3342    THE LOGICAL SEQUENCE WOULD NEXT EXECUTE A BIN*SM3*DM0*-DF7*SR0(1) THEN
        A BIN*SM3*DM0*DF7*SR0(0) THEN A BIN*SM3*DM0*DF7*SR0(1) INSTRUCTION,
        BUT NO ADDITIONAL LOGIC IS TESTED.

3349    TEST 76 SEVEN MICROSTATES (BIN*SM5*DM0*-DF7*SR0(0))

                FORK A SHOULD NOT FAIL.

                IF BEN14*FEN4 FAILS EXECUTION WILL GO TO D00.90
                CAUSING A SM4 INSTRUCTION TO BE EXECUTED. THIS WILL ONLY
                OCCUR IF EITHER IRCC SRCM5 DOES NOT GO LOW OR IF IRCC E28 IS BAD.

                ROM FLOW-24,23,27,317,143,146,205

3378    THE LOGICAL SEQUENCE WOULD NEXT EXECUTE A BIN*SM12*DM12*SR0(0)*DR0(1)*
        P/CLASS THEN A BIN*SM12*DM12*SR0(1)*DR0(1)P/CLASS INSTRUCTION, BUT
        NO ADDITIONAL LOGIC IS TESTED.
```

```
3385    TEST 77 SEVEN MICROSTATES (BIN*+SM12*DM3*O/CLASS)

                IF IRCC C FORK MUX INPUT B2 IS NOT GOING LOW OR IRCC E40
                IS BAD A DM2 WILL BE EXECUTED.

                THE ONLY OTHER  POSSIBLE FAILURE IS STATE D30.80.

                ROM FLOW-21,27,113,221,233,311,157

3414    THE LOGICAL SEQUENCE WOULD NEXT TEST A BIN*SM12*DM4*SRO(0)*DRO(0)*
        P/CLASS FOLLOWED BY A BIN*SM12*DM4*SRO(0)*DRO(1)*[TST.B+BIT.B+CMP.B]
        FOLLOWED BY A BIN*SM12*DM4*SRO(1)*DRO(0)*P/CLASS FOLLOWED BY A
        BIN*SM12*DM4*SRO(1)*DRO(1)*[TST.B+BIT.B+CMP.B] INSTRUCTION, BUT NO
        ADDITIONAL LOGIC IS TESTED.

3423    TEST 100        SEVEN MICROSTATES (BIN*SM12*DM6*O/CLASS)

                IF FORK C FAILS EXECUTION WILL GO TO D45.90 AND A
                DM4 WILL BE EXECUTED. THIS WILL ONLY HAPPEN IF IRCC E39 PIN 5
                IS NOT GOING LOW. THIS WILL CAUSE AN RTI SINCE THE LOCATION
                FOLLOWING THE INSTRUCTION CONTAINS 000002.

                THE ONLY OTHER FAILURE WOULD BE CAUSED BY STATE D67.80 BEING BAD.

                ROM FLOW-21,27,117,6,251,122,157

3454    THE LOGICAL SEQUENCE WOULD NEXT TEST THE INSTRUCTIONS BETWEEN
        BIN*SM4*DM12*SRO(0)*DRO(1)*[TST.B+BIT.B+CMP.B] AND BIN*SM5*DM0*DF7*SRO(1)
        BUT NOT ADDITIONAL LOGIC IS TESTED.

        ********************************************************************
3460
        ********************************************************************

3462    TEST 101        EIGHT MICROSTATES (BIN*SM7*DM0*-DF7*SRO(0))

                FORK A SHOULD NOT FAIL.

                IF FEN4*BEN14 FAILS EXECUTION WILL GO TO D00.90 CAUSING
                A SM6 TO BE EXECUTED. THIS WILL ONLY HAPPEN IF EITHER
                IRCC SRCM7 DOES NOT GO LOW OR IF IRCC E28(1) IS BAD.

                ROM FLOW-26,54,141,142,317,143,146,205

3492    THE LOGICAL SEQUENCE WOULD NEXT TEST A BIN*SM12*DM3*SRO(0)*DRO(0)*[TST.B+
        BIT.B+CMP.B] BUT NO ADDITIONAL LOGIC IS TESTED.

3498    TEST 102        EIGHT MICROSTATES (BIN*SM12*DM3*SRO(1)*DRO(0)*CMPB)

                THE ONLY POSSIBLE FAILURE WOULD BE IN STATE D30.90
                SINCE ALL THE OTHER LOGIC HAS BEEN TESTED.

                ROM FLOW-21,27,112,221,233,311,177,33
```

```
                3520    THE LOGICAL SEQUENCE WOULD NEXT TEST INSTRUCTIONS BIN*SM12*DM4*SR0(0)*DR0(1)*
                        P/CLASS THRU BIN*SM12*DM6*SR0(0)*DR0(0)*[TST.B+BIT.B+CMP.B] BUT NO
                        ADDITIONAL LOGIC IS TESTED.

                3527    TEST 103        EIGHT MICROSTATES (BIN*SM12*DM6*SR0(1)*DR0(0)*CMPB)

                3528
                                THE ONLY POSSIBLE FAILURE IN THIS TEST IS STATE D67.90.

                                ROM FLOW-21,27,116,6,251,122,177,33

                ************************************************************************
        3543
                ************************************************************************

                3545    TEST 104        NINE  MICROSTATES (BIN*SM12*DM5*SR0(0)*DR0(0)*CMP.B)

                                THE ONLY POSSIBLE FAILURE IN THIS TEST IS STATE D50.20.

                                ROM FLOW-21,27,115,161,231,233,311,177,33

                3564    TEST 105        EIGHT MICROSTATES (BIN*SM12*DM5*SR0(1)*DR0(0)*CMPB)

                                THE ONLY POSSIBLE FAILURE IN THIS TEST IS STATE D50.30.

                3580    TEST 106        WRITE/READ PSW


                3582            THIS TEST VERIFIES THAT THE PSW CAN BE READ THRU THE DATA MUX.
                                IF THE TEST FAILS ONE OF MANY THINGS COULD BE BAD WHICH CANNOT BE
                                DETERMINED IN THIS DIAGNOSTIC.
                                THIS TEST REQUIRES THAT SCCE PS ADRS GETS TO TMC,
                                THAT THE TMC DMUX SELECT LINES GET TO PDR, THAT THE PSW
                                BITS GET TO THE DMUX, THAT SCCE INTERNAL ADDRESS GETS TO TMC,
                                AND SCCA VA00 GETS TO UBC.

        3615    TEST 107        RTI

                                IF FORK A FAILS EXECUTION WILL GO TO ONE OF THREE STATES.
                                RSD.00 WILL CAUSE A TRAP TO LOCATION 4. THIS WOULD HAPPEN
                                IF RACF (HALT:OP CD 7) DOES NOT GO HIGH.
                                STATE D12.01 WOULD CAUSE AN ODD ADDRESS TRAP SINCE R0
                                WILL CONTAIN A 1. THIS WILL HAPPEN IF RACE E7 IS BAD.
                                HLT.00 WILL CAUSE THE PROCESSOR TO HALT ON THE INSTRUCTION
                                UNDER TEST AND WILL  OCCUR IF RACF E17 IS BAD.
                                IF THE INSTRUCTION DOESN'T WORK THEN ONE OF THE RTI MACHINE
                                STATES IS BAD.

                                ROM FLOW-12,156,212,213,214,215,172
```

```
                THE RTT WILL NOT BE TESTED HERE SINCE THE ONLY POSSIBLE FORK A
                FAILURE WOULD CAUSE AN RTI TO BE EXECUTED. THE T BIT FUNCTIONS OF
                THE RTI & RTT ARE TESTED IN PART 2.

3659            TEST 110        EMT AND TRAP

                        FORK A SHOULD NOT FAIL.
                        THE INSTRUCTIONS ARE EXECUTED AND THE STACK IS CHECKED TO
                        VERIFY THAT EVERYTHING WORKED OK.

                        ROM FLOW-0,345,354,SVC.00-SVC.90

3741            TEST 111        IOT

                        FORK A SHOULD NOT FAIL.

3744
                        IF THE TRAP VECTOR LOGIC FAILS THE TRAP VECTOR COULD COME
                        OUT TO BE 0, 4, OR 24.

                        THE ONLY OTHER POSSIBLE FAILURE WOULD BE STATE TRP.01.
                        IF THIS STATE FAILS TO LOAD THE DR THE TRAP VECTOR WILL
                        BE WHATEVER IS IN R4.  IF IT FAILS TO LOAD THE BR THE OLD
                        PS WILL FAIL TO BE STACKED.

                        ROM FLOW-14,354,(SVC.00-SVC.90) 355,65,357,360,367,37,25,41,222,300

        ;*****************************************************************
3815    END OF PASS ROUTINE
        ;*****************************************************************

3817    INCREMENT THE PASS NUMBER ($PASS)
        INDICATE END-OF-PROGRAM AFTER 144 PASSES THRU THE PROGRAM
        TYPE 'END PASS''
        IF THERES A MONITOR GO TO IT
        IF THERE ISN'T JUMP TO TST1
        IF IT IS DESIRED TO HAVE A BELL INDICATE THE 'END OF PASS'' LOCATION
        $ENDMG CAN BE CHANGED TO 7.
```

```
                ;*****************************************************************
3851            ;TYPE ROUTINE
                ;*****************************************************************

                3853    ;ROUTINE TO TYPE ASCIZ MESSAGE. MESSAGE MUST TERMINATE WITH A 0 BYTE.
                        ;THE ROUTINE WILL INSERT A NUMBER OF NULL CHARACTERS AFTER A LINE FEED.
                        ;NOTE1:  $NULL CONTAINS THE CHARACTER TO BE USED AS THE FILLER CHARACTER.
                        ;NOTE2:  $FILLS CONTAINS THE NUMBER OF FILLER CHARACTERS REQUIRED.
                        ;NOTE3:  $FILLC CONTAINS THE CHARACTER TO FILL AFTER.

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

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

                ;*****************************************************************
3924            ;BINARY TO OCTAL (ASCII) AND TYPE
                ;*****************************************************************

                3926    ;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
```

```
                ;*****************************************************************
4002            ;CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
                ;*****************************************************************

4004            ;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 BF
                ;REPLACED WITH SPACES.
                ;ALL:
                        MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
                        TYPDS                   ;;GO TO THE ROUTINE

                ;*****************************************************************
4070            ;TRAP DECODER
                ;*****************************************************************

4072            ;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.

                ;*****************************************************************
4085            ;TRAP TABLE
                ;*****************************************************************

4087            ;THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
                ;BY THE "TRAP" INSTRUCTION.
```

```
                                .TITLE  PDP 11/70-74MP CPU DIAGNOSTIC PART 1
                                ;*COPYRIGHT (C) 1975,1979
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*PROGRAM BY DONALD W. MONROE
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-A5).
                                ;*
160000                          $SWR=160000     ;;HALT ON ERROR, LOOP ON TEST, INHIBIT ERROR TYPOUT
```

```
                                .SBTTL  BASIC DEFINITIONS

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

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

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

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

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

```
                        SW14=   40000
                        SW13=   20000
                        SW12=   10000
                        SW11=   4000
                        SW10=   2000
                        SW09=   1000
                        SW08=   400
                        SW07=   200
                        SW06    100
                        SW05=   40
                        SW04=   20
                        SW03    10
                        SW02    4
                        SW01=   2
                        SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0

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

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

```
                        RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
                        TBITVEC=14              ;;''T'' BIT
                        TRTVEC= 14              ;;TRACE TRAP
                        BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
                        IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
                        PWRVEC= 24              ;;POWER FAIL
                        EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
                        TRAPVEC=34              ;;''TRAP'' TRAP
                        TKVEC=  60              ;;TTY KEYBOARD VECTOR
                        TPVEC=  64              ;;TTY PRINTER VECTOR
                        CACHVEC=114             ;;CACHE ERROR INTERRUPT VECTOR
                        PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR
                        MMVEC=  250             ;;MEMORY MANAGEMENT VECTOR

                        .SBTTL  CACHE   REGISTER DEFINITIONS


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


                        .SBTTL  CPU REGISTER DEFINITIONS


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



                        .SBTTL  MEMORY MANAGEMENT DEFINITIONS


                        ;*MEMORY MANAGEMENT STATUS REGISTER ADDRESSES

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

                        ;*USER ''I'' PAGE DESCRIPTOR REGISTERS

                        UIPDR0  177600
```

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

                                ;*USER 'D' PAGE DESCRIPTOR REGISTORS

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

                                ;*USER 'I' PAGE ADDRESS REGISTERS

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

                                ;*USER 'D' PAGE ADDRESS REGISTERS

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

                                ;*SUPERVISOR 'I' PAGE DESCRIPTOR REGISTERS

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

                                ;*SUPERVISOR 'D' PAGE DESCRIPTOR REGISTERS

                172220          SDPDR0= 172220
                172222          SDPDR1= 172222
```

```
                                ;*SUPERVISOR ''I'' PAGE ADDRESS REGISTERS

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

                                ;*SUPERVISOR ''D'' PAGE ADDRESS REGISTERS

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

                                ;*KERNEL ''I'' PAGE DESCRIPTOR REGISTERS

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

                                ;*KERNEL ''D'' PAGE DESCRIPTOR REGISTERS

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

                                ;*KERNEL ''I'' PAGE ADDRESS REGISTERS

172340                          KIPAR0= 172340
172342                          KIPAR1= 172342
172344                          KIPAR2= 172344
```

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

                                ;*KERNEL 'D'' PAGE ADDRESS REGISTERS

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



                                .SBTTL  UNIBUS MAP REGISTER DEFINITIONS

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

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

```
                170272          MAPH16 = 170272
                170274          MAPL17 = 170274
                170276          MAPH17 = 170276
                170300          MAPL20 = 170300
                170302          MAPH20 = 170302
                170304          MAPL21 = 170304
                170306          MAPH21 = 170306
                170310          MAPL22 = 170310
                170312          MAPH22 = 170312
                170314          MAPL23 = 170314
                170316          MAPH23 = 170316
                170320          MAPL24 = 170320
                170320          MAPH24 = 170320
                170324          MAPL25 = 170324
                170326          MAPH25 = 170326
                170330          MAPL26 = 170330
                170332          MAPH26 = 170332
                170334          MAPL27 = 170334
                170336          MAPH27 = 170336
                170340          MAPL30 = 170340
                170342          MAPH30 = 170342
                170344          MAPL31 = 170344
                170346          MAPH31 = 170346
                170350          MAPL32 = 170350
                170352          MAPH32 = 170352
                170354          MAPL33 = 170354
                170356          MAPH33 = 170356
                170360          MAPL34 = 170360
                170362          MAPH34 = 170362
                170364          MAPL35 = 170364
                170366          MAPH35 = 170366
                170370          MAPL36 = 170370
                170372          MAPH36 = 170372
                170374          MAPL37 = 170374
                170376          MAPH37 = 170376
                                .EQUIV  MAPL00,MAPL0
                                .EQUIV  MAPH00,MAPH0
                                .EQUIV  MAPL01,MAPL1
                                .EQUIV  MAPH01,MAPH1
                                .EQUIV  MAPL02,MAPL2
                                .EQUIV  MAPH02,MAPH2
                                .EQUIV  MAPL03,MAPL3
                                .EQUIV  MAPH03,MAPH3
                                .EQUIV  MAPL04,MAPL4
                                .EQUIV  MAPH04,MAPH4
                                .EQUIV  MAPL05,MAPL5
                                .EQUIV  MAPH05,MAPH5
                                .EQUIV  MAPL06,MAPL6
                                .EQUIV  MAPH06,MAPH6
                                .EQUIV  MAPL07,MAPL7
                                .EQUIV  MAPH07,MAPH7
```

```
                        .SBTTL  TRAP CATCHER

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

                        .SBTTL  STARTING ADDRESS(ES)
000200                          . 200

000200  000137  001202            JMP     @#START         ;; JMP TO STARTING ADDRESS OF PROGRAM
                        ;;*****************************************************************

                        .SBTTL          ACT11 HOOKS

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

000204                  $SVPC=.                 ;;SAVE LOCATION COUNTER
000046                  .-46                    ;;SET LOCATION COUNTER
000046  011034          .WORD   $ENDAD          ;;SET LOC.46 TO ADDRESS $ENDAD
000052                  . 52                    ;;SET LOCATION COUNTER
000052  000000          .WORD   0               ;;SET LOC.52 TO ZERO
000204                  .-$SVPC                 ;; RESTORE LOCATION COUNTER
```

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

                                .SBTTL  COMMON TAGS

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

          001100                        .=1100

001100                          $CMTAG:                 ;;START OF COMMON TAGS
001100  000000                  $PASS:  .WORD   0       ;;CONTAINS PASS COUNT
001102     000                  $TSTNM: .BYTE   0       ;;CONTAINS THE TEST NUMBER
001103     000                  $ERFLG: .BYTE   0       ;;CONTAINS ERROR FLAG
001104  000000                  $ICNT:  .WORD   0       ;;CONTAINS SUBTEST ITERATION COUNT
001106  000000                  $LPADR: .WORD   0       ;;CONTAINS SCOPE LOOP
001110  000000                  $LPERR: .WORD   0       ;;CONTAINS SCOPE RETURN FOR ERRORS
001112  000000                  $ERTTL: .WORD   0       ;;CONTAINS TOTAL ERRORS DETECTED
001114     000                  $ITEMB: .BYTE   0       ;;CONTAINS ITEM CONTROL BYTE
001115     001                  $ERMAX: .BYTE   1       ;;CONTAINS MAX. ERRORS PER TEST
001116  000000                  $ERRPC: .WORD   0       ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001120  000000                  $GDADR: .WORD   0       ;;CONTAINS  OF 'GOOD' DATA
001122  000000                  $BDADR: .WORD   0       ;;CONTAINS  OF 'BAD' DATA
001124  000000                  $GDDAT: .WORD   0       ;;CONTAINS 'GOOD' DATA
001126  000000                  $BDDAT: .WORD   0       ;;CONTAINS 'BAD' DATA
001130  000000  000000  000000          .WORD   0,0,0   ;;RESERVED--NOT TO BE USED
001136  177560                  $TKS:   177560          ;;TTY KBD STATUS
001140  177562                  $TKB:   177562          ;;TTY KBD BUFFER
001142  177564                  $TPS:   177564          ;;TTY PRINTER STATUS REG.
001144  177566                  $TPB:   177566          ;;TTY PRINTER BUFFER REG.
001146     000                  $NULL:  .BYTE   0       ;;CONTAINS NULL CHARACTER FOR FILLS
001147     002                  $FILLS: .BYTE   2       ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001150     012                  $FILLC: .BYTE   12      ;;INSERT FILL CHARS. AFTER A 'LINE FEED''
001151     000                  $TPFLG: .BYTE   0       ;;'TERMINAL AVAILABLE'' FLAG (BIT<07>=0=YES)
001152  000000                  $REGAD: .WORD   0       ;;CONTAINS THE  FROM
                                                        ;;WHICH  ($REG0) WAS OBTAINED
001154  000000                  $REG0:  .WORD   0       ;;CONTAINS (($REGAD)+0)
001156  000000                  $REG1:  .WORD   0       ;;CONTAINS (($REGAD)+2)
001160  000000                  $REG2:  .WORD   0       ;;CONTAINS (($REGAD)+4)
001162  000000                  $TMP0:  .WORD   0       ;;USER DEFINED
001164  000000                  $TMP1:  .WORD   0       ;;USER DEFINED
001166  000000                  $TMP2:  .WORD   0       ;;USER DEFINED
001170  000000                  $TMP3.  .WORD   0       ;;USER DEFINED
001172     077                  $QUES:  .ASCII  /?/     ;;QUESTION MARK
001173     015                  $CRLF:  .ASCII  <15>    ;;CARRIAGE RETURN
001174  000012                  $LF:    .ASCIZ  <12>    ;;LINE FEED
001176                          $ERPSW:
001176  000000                  .WORD
001200  000000                  0
```

```
                        ;;************ ************************************************************
                        .SBTTL  EPROR POINTER TABLE

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

                        ;*      EM              ;;POINTS TO THE ERROR MESSAGE
                        ;*      D4              ;;POINTS TO THE DATA HEADER
                        ;*      DT              ;;POINTS TO THE DATA
                        ;*      DF              ;;POINTS TO THE DATA FORMAT

  001202                $ERRTB:
                        ;;*****************************************************************
                        ;*      FOLLOWING IS AN INDEX OF THE POSSIBLE FAILURES THAT
                        ;*      CAUSE A TRAP TO LOCATIONS 4, 10, 14, OR 24 OR THAT
                        ;*      CAUSE THE PROCESSOR TO HANG (H). NGT-NOT GETTING THRU
                        ;*
                        ;*   TEST NUMBER        EFFECT                          CAUSE
                        ;*
                        ;*      25                  10                  RACH NEG.B*DMO NOT GOING HIGH
                        ;*      25                  24                  RACH A2 RAB00 NOT GOING LOW
                        ;*      25                  H                   RACH E57(6) NOT GOING LOW
                        ;*                                              OR NGT RACL RADR07
                        ;*      26                  H                   RACL RADR00 NOT GOING LOW
                        ;*      26                  4                   EITHER IRCC SM357 STUCK H
                        ;*                                              OR RACL E70 BAD
                        ;*      26                  14                  IRCC C0 RAB03 NOT GOING H OR
                        ;*                                              RACL RADR03 INPUT STUCK LOW
                        ;*      26                  4                   SRC CONST ADDED 1 OR 3
                        ;*
                        ;*      27                  H                   RACK RADR01 NOT GOING LOW
                        ;*
                        ;*      31                  10                  RACK A0 RAB01 NOT GOING LOW OR
                        ;*                                              NGT RACL RADR01
                        ;*      31                  4                   RACK BRCAB05 NOT GOING LOW OR
                        ;*                                              NGT PACL RADR05
                        ;*                                              1$ ON TOP OF STACK
                        ;*      31                  4                   DST CONST ADDED 1 OR 3
                        ;*      32                  10                  RACE A0 RAB00 DOES NOT GO LOW
                        ;*
                        ;*      33                  10                  RACE A0 RAB02 DOES NOT GO LOW
                        ;*
                        ;*      34                  10                  RACE E44 IS BAD
                        ;*      34                  10                  IRCB K/CLASS STUXK LOW
                        ;*      34                  H                   GRAB OBD(1) STUCK H OR NGT RACL E71
                        ;*      34                  H                   GRAD DRMX00 STUCK H OR GRAB E50 BAD
                        ;*
                        ;*      35                  10                  RACE BIN*SM0 H DID NOT GO HIGH
                        ;*      35                  10                  IR DECODE ROM WORD BAD
                        ;*
                        ;*      36                  10                  RACE BIN*SM0 FAILED
```

```
560                     ;*      36              10              IR DECODE ROM WORD BAD
561                     ;*
562                     ;*      37              10              RACE E45 BAD
563                     ;*
564                     ;*      40              10              RACE A0 RAB00 DOES NOT GO LOW
565                     ;*      40              10              IRCB(JMP+JSR) IS STUCK LOW
566                     ;*      40              H               IRCB FJ CLASS IS STUCK HIGH
567                     ;*
568                     ;*      41              10              RACE E45 IS BAD
569                     ;*
570                     ;*      42              10              RACE E33 IS BAD
571                     ;*
572                     ;*      43              10              RACH U/CLASS NOT GOING HIGH
573                     ;*                                      5$-2 ON TOP OF STACK
574                     ;*      43              10              EITHER RACE E10 BAD OR
575                     ;*                                      RACE BIN NOT GOING H
576                     ;*                                      5$ ON TOP OF STACK
577                     ;*
578                     ;*      44              10              RACJ AFIR 14(1) NGT RACE E42
579                     ;*      44              10              IRCB E38(6) STUCK HIGH
580                     ;*
581                     ;*      45              H               GRAB OBD(0) STUCK H OR NGT RACK E51
582                     ;*
583                     ;*      46              10              RACE E33 BAD
584                     ;*
585                     ;*      54              10              RACF E3 DOES NOT GO HIGH
586                     ;*
587                     ;*      56              10              PART PCLASS FIELD BAD IN IR DECODE ROM
588                     ;*
589                     ;*      60              10              PART PCLASS FIELD BAD IN IR DECODE ROM
590                     ;*
591                     ;*      62              10              IRCC C0 RAB03 STUCK H OR NGT RACL RADR03
592                     ;*                                      OR FORK C MUX INPUT B0 STUCK H
593                     ;*
594                     ;*      65              10              B FORK MUX SELECT STUCK LOW
595                     ;*      65              H               IRCB B0 RAB04 NGT RADR05
596                     ;*      65              H               B FORK MUX INPUT B0 STUCK H OR
597                     ;*                                      IRCC B0 RAB00 STUCK H
598                     ;*      65              4               B FORK MUX INPUT B3 OR
599                     ;*                                      IRCB B0 RAB00 STUCK L
600                     ;*      65              H               IRCB E46(10) STUCK L-MICRO ADR 170
601                     ;*
602                     ;*      67              10              RACE E35(1) BAD
603                     ;*
604                     ;*      70              10              B FORK MUX STROBE STUCK L (CHIP FAILURE)
605                     ;*
606                     ;*      75              10              RACE JMP+JSR+SWAB NOT GOING HIGH
607                     ;*      75              10              IRCB E63 BAD-R5 CONTAINS 'T67+2'
608                     ;*
609                     ;*      105             4               RACF (HALT:OP CODE 7) DOES NOT GO HIGH
610                     ;*      105             4               RACE E7 BAD-ODD ADR BIT SET IN ERROR REG
611                     ;*
612                     ;**************************************************************************
613                     ;*      THE FOLLOWING FIVE CONDITION CODE AND BRANCH TESTS ARE A
614                     ;*      FUNCTION OF RACK F TRUE 1. SECTION 1 OF EACH TEST IS
615                     ;*      DEPENDENT ON TRUE 1 NOT GOING HIGH, WHILE SECTION 2 I
```

```
                                ;*      DEPENDENT ON TRUE ONE GOING HIGH.
                                ;*
001202  012737  000014  177746  START:  MOV     #14,@#CONTRL    ;FORCE MISSES IN CACHE
                                ;;*****************************************************************
                                ;*TEST 1        CCC*BRANCH THRU FET.13
                                ;*
                                ;*      THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 1:
                                ;*         SECTION 1
                                ;*              BCS     E58(13,12)      L,H
                                ;*              BMI     E59(10,11,9)    H,L,H
                                ;*              BVS     E48(5,3,4)      H,L,H
                                ;*              BLOS    E59(4,3,5)      H,L,H
                                ;;*****************************************************************
001210  000257                TST1:   CCC                     ;CC=0000
                                ;SECTION 1
001212  103404                        BCS     1$
001214  100403                        BMI     1$
001216  102402                        BVS     1$
001220  101401                        BLOS    1$
001222  103001                        BCC     TST2            ;;GO TO NEXT TEST
001224                          1$:
001224  000000                        HALT                    ;RACF TRUE 1 WENT HIGH OR
                                                                ;RACH A2 RAB02 IS NOT GOING HIGH
                                                                ;OR NOT GETTING THRU RACL RADR02
                                                                ;FOR LOOPING CHANGE TO 'BR TST1'' (771)
                                ;;*****************************************************************
                                ;*TEST 2        SEC*BRANCH THRU FET.13 AND FET.11
                                ;*
                                ;*      THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 1:
                                ;*         SECTION 1
                                ;*              BMI     E58(13,12)      H,L
                                ;*              BVS     E58(13,12)      H,L
                                ;*         SECTION 2
                                ;*              BCC     E58(13,12)      H,H
                                ;;*****************************************************************
001226  000261                TST2:   SEC                     ;CC 0001
                                ;SECTION 1
001230  100402                        BMI     1$
001232  102401                        BVS     1$
001234  100001                        BPL     2$              ;GO TO SECTION 2
001236                          1$:
001236  000000                        HALT                    ;RACF E58 FAILED
                                                                ;FOR LOOPING CHANGE TO 'BR TST2'' (773)
                                ;SECTION 2
001240  103001                2$:     BCC     3$
001242  103401                        BCS     TST3            ;;GO TO NEXT TEST
001244                          3$:
001244  000000                        HALT                    ;EITHER RACF TRUE 1 DID NOT GO HIGH
                                                                ;OR IT DID NOT GET THRU RACH A2 RAB00
                                                                ;FOR LOOPING CHANGE TO 'BR 2$'' (775)
                                ;;*****************************************************************
                                ;*TEST 3        SEV*BRANCH THRU FET.13 AND FET.11
                                ;*
                                ;*      THE FOLLOWING IS A LIST OF PATTERNS PUT ON THE GATES OF TRUE 1:
                                ;*         SECTION 1
                                ;*              BCS     E48(5,3,4)      L,H,H
```

```
                                ;*              BMI     E48(5,3,4)      H,H,L
                                ;*          SECTION 2
                                ;*              BVC     E48(5,3,4)      H,H,H
                                ;;*****************************************************************
001246  000257                  TST3:   CCC                             ;CC=0000
                                ;SECTION 1
001250  000262                          SEV                             ;CC=0010
001252  103402                          BCS     1$
001254  100401                          BMI     1$
001256  100001                          BPL     2$
001260                          1$:
001260  000000                          HALT                            ;RACF E48 FAILED
                                                                        ;FOR LOOPING CHANGE TO 'BR TST3'' (???)
                                ;SECTION 2
001262  102001                  2$:     BVC     3$
001264  102401                          BVS     TST4                    ;;GO TO NEXT TEST
001266                          3$:
001266  000000                          HALT                            ;EITHER RACF E48 OR E47(1) FAILED
                                                                        ;FOR LOOPING CHANGE TO 'BR 2$' (775)
                                ;;*****************************************************************
                                ;*TEST 4         SEZ*BRANCH THRU FET.13 AND FET.11
                                ;*
                                ;*      THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 1:
                                ;*          SECTION 1
                                ;*              BCS     E59(4,3,5)      H,H,L
                                ;*              BMI     E59(4,3,5)      L,H,H
                                ;*          SECTION 2
                                ;*              BHI     E59(4,3,5)      H,H,H
                                ;;*****************************************************************
001270  000257                  TST4:   CCC                             ;CC=0000
                                ;SECTION 1
001272  000264                          SEZ                             ;CC=0100
001274  103402                          BCS     1$
001276  100401                          BMI     1$
001300  102001                          BVC     2$                      ;GO TO SECTION 2
001302                          1$:
001302  000000                          HALT                            ;RACF E59 FAILED
                                                                        ;FOR LOOPING CHANGE TO 'BR TST4'' (772)
                                ;SECTION 2
001304  101001                  2$:     BHI     3$
001306  101401                          BLOS    TST5                    ;;GO TO NEXT TEST
001310                          3$:
001310  000000                          HALT                            ;EITHER RACF E59 OR E47(11) FAILED
                                                                        ;FOR LOOPING CHANGE TO 'BR 2$'' (775)
                                ;;*****************************************************************
                                ;*TEST 5         SEN*BRANCH THRU FET.13 AND FET.11
                                ;*
                                ;*      THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 1:
                                ;*          SECTION 1
                                ;*              BLOS    E59(10,11,9)    H,H,L
                                ;*              BVS     E59(10,11,9)    L,H,H
                                ;*          SECTION 2
                                ;*              BPL     E59(10,11,9)    H,H,H
                                ;;*****************************************************************
001312  000257                  TST5:   CCC                             ;CC=0000
                                ;SECTION 1
```

```
                                                SEN                     ;CC=1000
                                                BLOS    '$
                                                BVS     '$
                                                BHI     2$              ;GO TO NEXT SECTION
                                        1$:
                                                HALT                    ;RACF E59 FAILED
                                                                        ;FOR LOOPING CHANGE TO 'BR TST5'' (772)
                                        ;SECTION 2
                                        2$:     BPL     3$
                                                BMI     TST6            ;;GO TO NEXT TEST
                                        3$:
                                                HALT                    ;EITHER RACF E59 OR E47(13) FAILED
                                                                        ;FOR LOOPING CHANGE TO 'BR 2$'' (775
                                        ;;*****************************************************************
                                        ;*      THE FOLLOWING SEVEN TESTS ARE A FUNCTION OF RACF TRUE 2.
                                        ;*SECTION 1 OF EACH TEST IS DEPENDENT ON TRUE 2 NOT GOING HIGH
                                        ;*WHILE SECTION 2 IS DEPENDENT ON TRUE 2 GOING HIGH.
                                        ;*
                                        ;;*****************************************************************
                                        ;*TEST 6        BRANCHES THRU FET.13
                                        ;*
                                        ;*      THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
                                        ;*         SECTION 1
                                        ;*              BLE     E58(2,1)H,L     E59(13,1,2)L,H,H        E48(1,13,2)H,H,L
                                        ;*              BLT     E59(13,1,2)L,H,H        E48(1,13,2)H,H,L        E58(10,9)L,H
                                        ;*              BEQ     E58(2,1)H,L     E58(10,9)H,L
                                        ;;*****************************************************************
001334  000257                  TST6:   CCC                     ;CC=0000
001336  003403                          BLE     1$
001340  002402                          BLT     1$
001342  001401                          BEQ     1$
001344  003001                          BGT     TST7            ;;GO TO NEXT TEST
001346                          1$:
001346  000000                          HALT                    ;RACF TRUE 2 WENT HIGH
                                                                ;FOR LOOPING CHANGE TO 'BR TST6'' (772)
                                        ;;*****************************************************************
                                        ;*TEST 7        BRANCH THRU FET.13 AND FET.12
                                        ;*
                                        ;*      THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
                                        ;*         SECTION 1
                                        ;*              BEQ     E48(1,13,2)     L,H,H
                                        ;*         SECTION 2
                                        ;*              BGT     E48(1,13,2)     H,H,H
                                        ;;*****************************************************************
001350  000257                  TST7:   CCC                     ;CC=0000
                                        ;SECTION 1
001352  000262                          SEV                     ;CC=0010
001354  001401                          BEQ     1$
001356  001001                          BNE     2$              ;GO TO SECTION 2
001360                          1$:
001360  000000                          HALT                    ;RACF E48 FAILED
                                                                ;FOR LOOPING CHANGE TO 'BR TST7'' (773)
                                        ;SECTION 2
001362  003001                  2$:     BGT     3$
001364  003401                          BLE     TST10           ;;GO TO NEXT TEST
001366                          3$:
```

```
001366  000000                  HALT                    ;EITHER RACF TRUE 2 DID NOT GO HIGH
                                                        ;OR IT DID NOT GET THRU RACH A2 RABC'
                                                        ;FOR LOOPING CHANGE TO 'BR 2$'' (775)
                        ;;**************************************************************
                        ;*TEST 10       BRANCH THRU FET.13 AND FET.12
                        ;*
                        ;*      THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
                        ;*         SECTION 1
                        ;*              BLT     E58(2,1)        L,H
                        ;*         SECTION 2
                        ;*              BNE     E58(2,1)        H,H
                        ;;**************************************************************
001370  000257          TST10:  CCC                     ;CC-0000
                        ;SECTION 1
001372  000264                  SEZ                     ;CC=0100
001374  002401                  BLT     1$
001376  002001                  BGE     2$              ;GO TO SECTION 2
001400                  1$:
001400  000000                  HALT                    ;RACF E58 FAILED
                                                        ;FOR LOOPING CHANGE TO 'BR TST10'' (773)
                        ;SECTION 2
001402  001001          2$:     BNE     3$
001404  001401                  BEQ     TST11           ;;GO TO NEXT TEST
001406                  3$:
001406  000000                  HALT                    ;EITHER RACF E58 OR F46(12) FAILED
                                                        ;FOR LOOPING CHANGE TO 'BR 2$'' (775)
                        ;;**************************************************************
                        ;*TEST 11       BRANCH THRU FET.13 AND FET.12
                        ;*
                        ;*      THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
                        ;*         SECTION 1
                        ;*              BEQ     E59(13,1,2)     H,H,L
                        ;*         SECTION 2
                        ;*              BGE     E59(13,1,2)     H,H,H
                        ;;**************************************************************
001410  000257          TST11:  CCC                     ;CC=0000
                        ;SECTION 1
001412  000270                  SEN                     ;CC=1000
001414  001401                  BEQ     1$
001416  001001                  BNE     2$              ;GO TO NEXT SECTION
001420                  1$:
001420  000000                  HALT                    ;RACF E59 FAILED
                                                        ;FOR LOOPING CHANGE TO 'BR TST11'' (773)
                        ;SECTION 2
001422  002001          2$:     BGE     3$
001424  002401                  BLT     TST12           ;;GO TO NEXT TEST
001426                  3$:
001426  000000                  HALT                    ;EITHER RACF E59 OR E46(13) FAILED
                                                        ;FOR LOOPING CHANGE TO 'BR 2$'' (775)
                        ;;**************************************************************
                        ;*TEST 12       BRANCHES THRU FET.13 AND FET.12
                        ;*
                        ;*      THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
                        ;*         SECTION 1
                        ;*              BEQ     E58(2,1)H,L     E58(10,9)H,L
                        ;*              BLT     E59(13,1,2)H,L,H        E48(1,13,2)H,L,H        E58(10,9)L,H
```

```
                                ;;*****************************************************************
001430  000262                  TST12:  SEV                     ;CC=1010
                                ;SECTION 1
001432  001402                          BEQ     1$
001434  002401                          BLT     1$
001436  001001                          BNE     TST13           ;;GO TO NEXT TEST
001440                          1$:
001440  000000                          HALT                    ;RACF TRUE 2 WENT HIGH
                                                                ;FOR LOOPING CHANGE TO 'BR TST12'' (773)
                                ;;*****************************************************************
                                ;*TEST 13       UNIARY AND BINARY (SM0)
                                ;*
                                ;*      THE FOLLOWING TEST TESTS ALL THE E/CLASS
                                ;*      INSTRUCTIONS WITH A DESTINATION MODE OF 0
                                ;*      AND DESTINATION FIELD OF NOT 7. THIS CLASS CONSISTS
                                ;*      OF ALL THE UNIARY (EXCEPT NEG) INSTRUCTIONS AND
                                ;*      ALL THE BINARY INSTRUCTIONS WITH A SOURCE MODE
                                ;*      OF 0.
                                ;;*****************************************************************
001442  000277                  TST13:  SCC
001444  000244                          CLZ                     ;CC'S=1011
001446  005000                          CLR     R0              ;R0=000000, CC'S=0100
001450  103403                          BCS     CLRR0
001452  102402                          BVS     CLRR0
001454  100401                          BMI     CLRR0
001456  001401                          BEQ     .+4
001460                          CLRR0:
001460  000000                          HALT                    ;ERROR, INCORRECT CC'S AFTER CLR
                                                                ;FOR LOOPING CHANGE TO 'BR TST13'' (770)

001462  005000                          CLR     R0
001464  000277                          SCC                     ;CC'S=1111
001466  000244                          CLZ                     ;R0=000000, CC'S=1011
001470  005700                          TST     R0              ;R0=000000, CC'S=0100
001472  103403                          BCS     TSTR0
001474  102402                          BVS     TSTR0
001476  100401                          BMI     TSTR0
001500  001401                          BEQ     .+4
001502                          TSTR0:
001502  000000                          HALT                    ;ERROR, INCORRECT CC'S AFTER TST
                                                                ;FOR LOOPING CHANGE TO 'BR CLRR0+2'' (767)

001504  005000                          CLR     R0
001506  000257                          CCC                     ;CC'S=0000
001510  000266                          +SEZ!SEV                ;R0=000000, CC'S=0110
001512  005100                          COM     R0              ;R0=177777, CC'S=1001
001514  100003                          BPL     COMR0
001516  001402                          BEQ     COMR0
001520  102401                          BVS     COMR0
001522  103401                          BCS     .+4
001524                          COMR0:
001524  000000                          HALT                    ;ERROR, INCORRECT CC'S AFTER COM
                                                                ;FOR LOOPING CHANGE TO 'BR TSTR0+2'' (767)

001526  005000                          CLR     R0
001530  000277                          SCC                     ;R0=000000, CC'S=1111
```

```
  896  001532  005500                ADC     R0              ;R0=000001, CC'S=0000
  897  001534  100403                BMI     ADCR0
  898  001536  001402                BEQ     ADCR0
  899  001540  102401                BVS     ADCR0
  900  001542  103001                BCC     .+4
  901  001544                ADCR0:
  902  001544  000000                HALT                    ;ERROR, INCORRECT CC'S AFTER ADC
  903                                                        ;FOR LOOPING CHANGE TO 'BR COMR0+2'' (770)
  904
  905  001546  005000                CLR     R0
  906  001550  000261                SEC
  907  001552  005500                ADC     R0
  908  001554  000257                CCC
  909  001556  000270                SEN                     ;R0=000001, CC'S=1000
  910  001560  006000                ROR     R0              ;R0=000000, CC'S=0111
  911  001562  100403                BMI     RORR0
  912  001564  001002                BNE     RORR0
  913  001566  102001                BVC     RORR0
  914  001570  103401                BCS     .+4
  915  001572                RORR0:
  916  001572  000000                HALT                    ;ERROR, INCORRECT CC'S AFTER ROR
  917                                                        ;FOR LOOPING CHANGE TO 'BR ADCR0+2'' (765)
  918
  919  001574  005000                CLR     R0
  920  001576  000277                SCC
  921  001600  000250                CLN                     ;R0=000000, CC'S=0111
  922  001602  005300                DEC     R0              ;R0=177777, CC'S=1001
  923  001604  100003                BPL     DECR0
  924  001606  001402                BEQ     DECR0
  925  001610  102401                BVS     DECR0
  926  001612  103401                BCS     .+4
  927  001614                DECR0:
  928  001614  000000                HALT                    ;ERROR, INCORRECT CC'S AFTER DEC
  929                                                        ;FOR LOOPING CHANGE TO 'BR RORR0+2'' (767)
  930
  931  001616  005000                CLR     R0
  932  001620  000277                SCC                     ;R0=000000, CC'S=1111
  933  001622  005200                INC     R0              ;R0-000001, CC'S=0000
  934  001624  100403                BMI     INCR0
  935  001626  001402                BEQ     INCR0
  936  001630  102401                BVS     INCR0
  937  001632  103401                BCS     .+4
  938  001634                INCR0:
  939  001634  000000                HALT                    ;ERROR, INCORRECT CC'S AFTER INC
  940                                                        ;FOR LOOPING CHANGE TO 'BR DECR0+2'' (770)
  941
  942  001636  005000                CLR     R0
  943  001640  000277                SCC
  944  001642  000244                CLZ                     ;R0=000000, CC'S=1011
  945  001644  006300                ASL     R0              ;R0=000000, CC'S=0100
  946  001646  100403                BMI     ASLR0
  947  001650  001002                BNE     ASLR0
  948  001652  102401                BVS     ASLR0
  949  001654  103001                BCC     .+4
  950  001656                ASLR0:
  951  001656  000000                HALT                    ;ERROR, INCORRECT CC'S AFTER ASL
```

```
                                                ;FOR _OOPING CHANGE TO 'BR INCR0+2'' (767)

001660  005000          CLR     R0
001662  000261          SEC
001664  006000          ROR     R0              ;R0=100000, CC'S=1000
001666  006100          ROL     R0              ;R0-000000, CC'S=0111
001670  100403          BMI     ROLR0
001672  001002          BNE     ROLR0
001674  102001          BVC     ROLR0
001676  103401          BCS     .+4
001700                  ROLR0:
001700  000000          HALT                    ;ERROR, INCORRECT CC'S AFTER ROL
                                                ;FOR LOOPING CHANGE TO 'BR ASLR0+2'' (767)

001702  005000          CLR     R0
001704  000261          SEC
001706  006000          ROR     R0
001710  005200          INC     R0
001712  000277          SCC
001714  000251          +CLN!CLC                ;R0=100001, CC'S=0110
001716  006200          ASR     R0              ;R0=140000, CC'S=1001
001720  100003          BPL     ASRRC
001722  001402          BEQ     ASRR0
001724  102401          BVS     ASRR0
001726  103401          BCS     .+4
001730                  ASRR0:
001730  000000          HALT                    ;ERROR, INCORRECT CC'S AFTER ASR
                                                ;FOR LOOPING CHANGE TO 'BR RORR0+2'' (721)

001732  005000          CLR     R0
001734  000277          SCC
001736  000250          CLN                     ;R0=000000, CC'S-0111
001740  005600          SBC     R0              ;R0-177777, CC'S=1001
001742  100003          BPL     SBCR0
001744  001402          BEQ     SBCR0
001746  102401          BVS     SBCR0
001750  103401          BCS     .+4
001752                  SBCR0:
001752  000000          HALT                    ;ERROR, INCORRECT CC'S AFTER SBC
                                                ;FOR LOOPING CHANGE TO 'BR ASRR0+2'' (767)

001754  005000          CLR     R0
001756  000261          SEC
001760  006000          ROR     R0
001762  000277          SCC
001764  000250          CLN                     ;R0=100000, CC'S-0111
001766  000300          SWAB    R0              ;R0 000200, CC'S=1000
001770  100003          BPL     SWABR0
001772  001402          BEQ     SWABR0
G01774  102401          BVS     SWABR0
001776  103001          BCC     .+4
002000                  SWABR0:
002000  000000          HALT                    ;ERROR, INCORRECT CC'S AFTER SWAB
                                                ;FOR LOOPING CHANGE TO 'BR SBCR0+2'' (765)


002002  005000          CLR     R0
```

```
1008  002004  005300                  DEC     R0
1009  002006  000257                  CCC
1010  002010  000262                  SEV                     ;R0=177777, CC'S=0010
1011  002012  006700                  SXT     R0              ;R0=000000, CC'S=0100
1012  002014  100403                  BMI     SXTR0
1013  002016  001002                  BNE     SXTR0
1014  002020  102401                  BVS     SXTR0
1015  002022  103001                  BCC     .+4
1016  002024                  SXTR0:
1017  002024  000000                  HALT                    ;ERROR, INCORRECT CC'S AFTER SXT
1018                                                          ;FOR LOOPING CHANGE TO 'BR SWABR0+2'' (766)
1019  002026  005000                  CLR     R0
1020  002030  000270                  SEN                     ;R0=000000, CC'S=1XXX
1021  002032  006700                  SXT     R0              ;R0=177777, CC'S=1XXXX
1022  002034  005200                  INC     R0
1023  002036  001401                  BEQ     .+4
1024  002040                  SXT2:
1025  002040  000000                  HALT                    ;SIGN EXTEND FAILED WITH N SET
1026                                                          ;FOR LOOPING CHANGE TO 'BR SXTR0+2'' (772)
1027
1028  002042  005000                  CLR     R0
1029  002044  005300                  DEC     R0
1030  002046  000277                  SCC
1031  002050  000244                  CLZ                     ;R0-177777, CC'S=1011
1032  002052  074000                  XOR     R0,R0           ;R0 000000, CC'S-0101
1033  002054  100403                  BMI     XORR0
1034  002056  001002                  BNE     XORR0
1035  002060  102401                  BVS     XORR0
1036  002062  103401                  BCS     .+4
1037  002064                  XORR0:
1038  002064  000000                  HALT                    ;ERROR, INCORRECT CC'S AFTER XOR
1039                                                          ;FOR LOOPING CHANGE TO 'BR SXT2+2'' (766)
1040
1041  002066  005000                  CLR     R0
1042  002070  000261                  SEC
1043  002072  006000                  ROR     R0
1044  002074  000300                  SWAB    R0
1045  002076  000277                  SCC
1046  002100  000250                  CLN                     ;R0=000200, CC'S=0111
1047  002102  110000                  MOVB    R0,R0           ;R0=177600, CC'S=1001
1048  002104  100010                  BPL     MOVBR0
1049  002106  001407                  BEQ     MOVBR0
1050  002110  102406                  BVS     MOVBR0
1051  002112  103005                  BCC     MOVBR0
1052  002114  000250                  CLN
1053  002116  000264                  SEZ                     ;R0=177600, CC'S=0100
1054  002120  005700                  TST     R0              ;           CC'S=1000
1055  002122  100002                  BPL     MOVR0
1056  002124  001002                  BNE     .+6
1057  002126                  MOVBR0:
1058  002126  000000                  HALT                    ;ERROR, INCORRECT CC'S AFTER MOVB
1059                                                          ;FOR LOOPING CHANGE TO 'BR XORR0+2'' (757)
1060  002130                  MOVR0:
1061  002130  000000                  HALT                    ;ERROR, MOVB DID NOT SIGN EXTEND
1062                                                          ;FOR LOOPING CHANGE TO 'BR XORR0+2'' (756)
1063
```

```
1064  002132  005000                  CLR     R0
1065  002134  000277                  SCC
1066  002136  000244                  CLZ                     ;R0=000000, CC'S=1011
1067  002140  030000                  BIT     R0,R0           ;R0=000000, CC'S=0101
1068  002142  100403                  BMI     BITR0
1069  002144  001002                  BNE     BITR0
1070  002146  102401                  BVS     BITR0
1071  002150  103401                  BCS     .+4
1072  002152                  BITR0:
1073  002152  000000                  HALT                    ;ERROR, INCORRECT CC'S AFTER BIT
1074                                                          ;FOR LOOPING CHANGE TO 'BR MOVR0+2'' (767)
1075
1076  002154  005000                  CLR     R0
1077  002156  005200                  INC     R0
1078  002160  000277                  SCC
1079  002162  000244                  CLZ                     ;R0=000001, CC'S=1011
1080  002164  040000                  BIC     R0,R0           ;R0=000000, CC'S=0101
1081  002166  100403                  BMI     BICR0
1082  002170  001002                  BNE     BICR0
1083  002172  102401                  BVS     BICR0
1084  002174  103401                  BCS     .+4
1085  002176                  BICR0:
1086  002176  000000                  HALT                    ;ERROR, INCORRECT CC'S AFTER BIC
1087                                                          ;FOR LOOPING CHANGE TO 'BR BITR0+2'' (766)
1088
1089  002200  005000                  CLR     R0
1090  002202  005200                  INC     R0
1091  002204  000277                  SCC                     ;R0=000001, CC'S=1111
1092  002206  050000                  BIS     R0,R0           ;R0=000001, CC'S=0001
1093  002210  100403                  BMI     BISR0
1094  002212  001402                  BEQ     BISR0
1095  002214  102401                  BVS     BISR0
1096  002216  103401                  BCS     .+4
1097  002220                  BISR0:
1098  002220  000000                  HALT                    ;ERROR, INCORRECT CC'S AFTER BIS
1099                                                          ;FOR LOOPING CHANGE TO 'BR BICR0+2'' (767)
1100
1101  002222  005000                  CLR     R0
1102  002224  000261                  SEC
1103  002226  006000                  ROR     R0
1104  002230  006000                  ROR     R0
1105  002232  000277                  SCC
1106  002234  000252                  +CLN!CLV                ;R0=040000, CC'S=0101
1107  002236  060000                  ADD     R0,R0           ;R0=100000, CC'S=1010
1108  002240  100003                  BPL     ADDR0
1109  002242  001402                  BEQ     ADDR0
1110  002244  102001                  BVC     ADDR0
1111  002246  103001                  BCC     .+4
1112  002250                  ADDR0:
1113  002250  000000                  HALT                    ;ERROR, INCORRECT CC'S AFTER ADD
1114                                                          ;FOR LOOPING CHANGE TO 'BR BISR0+2'' (764)
1115
1116  002252  005000                  CLR     R0
1117  002254  005200                  INC     R0
1118  002256  000277                  SCC
1119  002260  000244                  CLZ                     ;R0=000001, CC'S=1011
```

```
1120  002262  160000                SUB     R0,R0           ;R0-000000, CC'S=0100
1121  002264  100403                BMI     SUBR0
1122  002266  001002                BNE     SUBR0
1123  002270  102401                BVS     SUBR0
1124  002272  103001                BCC     .+4
1125  002274                SUBR0:
1126  002274  000000                HALT                    ;ERROR, INCORRECT CC'S AFTER SUB
1127                                                          ;FOR LOOPING CHANGE TO 'BR ADDR0+2'' (766)
1128
1129  002276  005000                CLR     R0
1130  002300  000277                SCC
1131  002302  000244                CLZ                     ;R0 000000, CC'S=1011
1132  002304  020000                CMP     R0,R0           ;R0 000000, CC'S-0100
1133  002306  100403                BMI     CMPR0
1134  002310  001002                BNE     CMPR0
1135  002312  102401                BVS     CMPR0
1136  002314  103001                BCC     .+4
1137  002316                CMPR0:
1138  002316  000000                HALT                    ;ERROR, INCORRECT CC'S AFTER CMP
1139                                                          ;FOR LOOPING CHANGE TO 'BR SUBR0+2'' (767)
1140                                ;
1141                                ;
1142                                ;;*****************************************************************
1143                                ;*TEST 14       REGISTER SELECTION TEST
1144                                ;*
1145                                ;*      THIS TEST ENSURES THAT THE 6 ADDRESS LINES INTO
1146                                ;*      THE GENERAL PURPOSE REGISTERS (GPR) ARE NOT STUCK.
1147                                ;*      THE LABELS OF THE ADDRESS LINES ARE:
1148                                ;*              GSAX    GENERAL SOURCE ADDRESS LINE
1149                                ;*              GDAX    GENERAL DESTINATION ADDRESS LINE
1150                                ;*      WHERE X STANDS FOR LINE 0,1, OR 2.
1151                                ;*      THE CLASSES OF ERRORS DESCRIBED IN THIS TEST
1152                                ;*      ARE DEFINED AS FOLLOWS:
1153                                ;*              CLASS A=GDAX OK
1154                                ;*                      GSAX STUCK
1155                                ;*              CLASS B=GSAX OK
1156                                ;*                      GDAX STUCK
1157                                ;*              CLASS C-GSAX STUCK
1158                                ;*                      GDAX STUCK
1159                                ;;*****************************************************************
1160  002320  005000        TST14:  CLR     R0
1161  002322  005201                INC     R1
1162  002324  005700                TST     R0              ;DID INC AFFECT R0?
1163  002326  001406                BEQ     OVER            ;BRANCH ON NOT CLASS B OR C
1164  002330  005000        ROUT0:  CLR     R0
1165  002332  005201                INC     R1
1166  002334  010002                MOV     R0,R2           ;DID S0 REMAIN 0 ON INC R1?
1167  002336  001001                BNE     2$              ;BR IF YES-NOT CLASS B
1168  002340  000000                HALT                    ;ERROR, CLASS B FAILURE ON GRA0
1169                                                          ;FOR LOOPING CHANGE TO 'BR ROUT0'' (773)
1170  002342                2$:
1171  002342  000000                HALT                    ;ERROR, CLASS C FAILURE ON GRA0
1172                                                          ;FOR LOOPING CHANGE TO 'BR ROUT0'' (772)
1173  002344  005201        OVER:   INC     R1              ;INCREMENT S1 AND D1
1174  002346  010001                MOV     R0,R1           ;MOVE S0 TO S1 AND D1
1175  002350  001401                BEQ     GRA1T           ;BRANCH-GRA0 OK
```

```
1176  002352  000000                HALT                    ;ERROR, CLASS A FAILURE ON GRA0
1177                                                        ;FOR LOOPING CHANGE TO 'BR TST14'' (762)
1178                                ;
1179  002354  005000  GRA1T:  CLR     R0
1180  002356  005202          INC     R2
1181  002360  005700          TST     R0                  ;DID INC AFFECT R2?
1182  002362  001406          BEQ     OVER2               ;BRANCH ON NOT CLASS B OR C
1183  002364  005000  ROUT1:  CLR     R0
1184  002366  005202          INC     R2
1185  002370  010001          MOV     R0,R1               ;DID S0 REMAIN 0 ON INC R2?
1186  002372  001001          BNE     4$                  ;BR IF YES-NOT CLASS B
1187  002374  000000          HALT                        ;ERROR, CLASS B FAILURE ON GRA1
1188                                                        ;FOR LOOPING CHANGE TO 'BR ROUT1'' (773)
1189  002376                  4$:
1190  002376  000000          HALT                        ;ERROR, CLASS C FAILURE ON GRA1
1191                                                        ;FOR LOOPING CHANGE TO 'BR ROUT1'' (772)
1192  002400  005202  OVER2:  INC     R2                  ;INCREMENT S2 AND D2
1193  002402  010002          MOV     R0,R2               ;MOVE S0 TO S2 AND D2
1194  002404  001401          BEQ     GRA2T               ;BRANCH-GRA1 OK
1195  002406  000000          HALT                        ;ERROR, CLASS A FAILURE ON GRA1
1196                                                        ;FOR LOOPING CHANGE TO 'BR GRA1T'' (762)
1197                                ;
1198  002410  005000  GRA2T:  CLR     R0
1199  002412  005204          INC     R4
1200  002414  005700          TST     R0                  ;DID INC AFFECT R4?
1201  002416  001406          BEQ     OVER3               ;YES, CLASS A
1202  002420  005000  ROUT2:  CLR     R0
1203  002422  005204          INC     R4
1204  002424  010001          MOV     R0,R1               ;DID S0 REMAIN 0 AFTER INC R4?
1205  002426  001001          BNE     6$                  ;BR IF YES-NOT CLASS B
1206  002430  000000          HALT                        ;ERROR, CLASS B FAILURE ON GRA2
1207                                                        ;FOR LOOPING CHANGE TO 'BR ROUT2'' (773)
1208  002432                  6$:
1209  002432  000000          HALT                        ;ERROR, CLASS C FAILURE ON GRA2
1210                                                        ;FOR LOOPING CHANGE TO 'BR ROUT2'' (772)
1211  002434  005204  OVER3:  INC     R4                  ;INCREMENT S4 AND D4
1212  002436  010004          MOV     R0,R4               ;MOVE S0 TO S4 AND D4
1213  002440  001401          BEQ     8$                  ;;BRANCH IF OK
1214  002442  000000          HALT                        ;ERROR, CLASS A FAILURE ON GRA2
1215                                                        ;FOR LOOPING CHANGE TO 'BR GRA2T'' (762)
1216  002444  005001  8$:     CLR     R1
1217  002446  005003          CLR     R3
1218  002450  005005          CLR     R5
1219  002452  005201          INC     R1                  ;CHANGE R1
1220  002454  005703          TST     R3                  ;DID R3 CHANGE?
1221  002456  001401          BEQ     1$                  ;BRANCH IF NO
1222  002460  000000          HALT                        ;ADDRESS LINE 0 AND 1 TIED TOGETHER
1223                                                        ;FOR LOOPING CHANGE TO 'BR 8$'' (771)
1224  002462  010303  1$:      MOV     R3,R3               ;CHECK SRC ADDRESS LINES
1225  002464  001401          BEQ     2$                  ;BRANCH IF OK
1226  002466  000000          HALT                        ;ADDRESS LINE 0 AND 1 TIED TOGETHER(SRC)
1227                                                        ;FOR LOOPING CHANGE TO 'BR 8$'' (766)
1228  002470  005705  2$:      TST     R5                  ;DID R5 CHANGE?
1229  002472  001401          BEQ     3$                  ;BRANCH IF NO
1230  002474  000000          HALT                        ;ADDRESS LINES 0 & 2 TIED TOGETHER(DST)
1231                                                        ;FOR LOOPING CHANGE TO 'BR 8$'' (763)
```

```
1232  002476  010505          3$:     MOV     R5,R5
1233  002500  001401                  BEQ     4$
1234  002502  000000                  HALT                    ;ADDRESS LINES 0 & 2 TIED TOGETHER(SRC)
1235                                                          ;FOR LOOPING CHANGE TO 'BR 8$'' (760)
1236  002504  005002          4$:     CLR     R2
1237  002506  005006                  CLR     R6
1238  002510  005202                  INC     R2
1239  002512  005706                  TST     R6              ;DID R3 CHANGE?
1240  002514  001401                  BEQ     5$              ;BRANCH IF NO
1241  002516  000000                  HALT                    ;ADDRESS LINES 1 & 2 TIED TOGETHER(DST)
1242                                                          ;FOR LOOPING CHANGE TO 'BR 4$'' (772)
1243  002520  010606          5$:     MOV     R6,R6           ;GET R6 SRC
1244  002522  001401                  BEQ     TST15           ;;BRANCH IF SRC OK
1245  002524  000000                  HALT                    ;ADDRESS LINES 1 & 2 TIED TOGETHER(SRC)
1246                                                          ;FOR LOOPING CHANGE TO 'BR 4$' (767)
1247                                  ;;**************************************************************
1248                                  ;*TEST 15       GPR1 STUCK BIT TEST
1249                                  ;*
1250                                  ;*      LOADS GPR1 WITH ZEROS AND ONES AND COMPARES R1 SOURCE AND
1251                                  ;*      DESTINATIONS WITH R0. IF THE COMPARISON FAILS A BIT IS STUCK.
1252                                  ;;**************************************************************
1253  002526  005000          TST15:  CLR     R0
1254  002530  005001                  CLR     R1
1255  002532  020001                  CMP     R0,R1           ;DID R1 DST CLR?
1256  002534  001401                  BEQ     1$              ;BRANCH IF YES
1257  002536  000000                  HALT                    ;ERROR, R1 SOURCE STUCK HIGH
1258                                                          ;FOR LOOPING CHANGE TO 'BR TST15'' (773)
1259  002540  020100          1$:     CMP     R1,R0           ;DID R1 SRC CLR?
1260  002542  001401                  BEQ     2$              ;BRANCH IF YES
1261  002544  000000                  HALT                    ;ERROR, R1 SOURCE STUCK HIGH
1262                                                          ;FOR LOOPING CHANGE TO 'BR TST15'' (770)
1263  002546  005100          2$:     COM     R0
1264  002550  005101                  COM     R1
1265  002552  020001                  CMP     R0,R1           ;DID R1 DST SET TO ALL ONES?
1266  002554  001401                  BEQ     3$              ;BRANCH IF YES
1267  002556  000000                  HALT                    ;ERROR, R1 DST STUCK LOW
1268                                                          ;FOR LOOPING CHANGE TO 'BR TST15'' (763)
1269  002560  020100          3$:     CMP     R1,R0           ;DID R1 SRC SET TO ALL ONES?
1270  002562  001401                  BEQ     TST16           ;;BRANCH IF YES
1271  002564  000000                  HALT                    ;ERROR, R1 SRC STUCK LOW
1272                                                          ;FOR LOOPING CHANGE TO 'BR TST15'' (760)
1273                                  ;;**************************************************************
1274                                  ;*TEST 16       GPR2 STUCK BIT TEST
1275                                  ;*
1276                                  ;*      LOADS GPR2 WITH ZEROS AND ONES AND COMPARES R2 SOURCE AND
1277                                  ;*      DESTINATION WITH R0.
1278                                  ;;**************************************************************
1279  002566  005000          TST16:  CLR     R0
1280  002570  005002                  CLR     R2
1281  002572  020200                  CMP     R2,R0           ;DID R2 SRC CLEAR?
1282  002574  001401                  BEQ     1$              ;BRANCH IF YES
1283  002576  000000                  HALT                    ;ERROR, R2 SRC STUCK HIGH
1284                                                          ;FOR LOOPING CHANGE TO 'BR TST16'' (773)
1285  002600  020002          1$:     CMP     R0,R2           ;DID R2 DST CLEAR?
1286  002602  001401                  BEQ     2$              ;BRANCH IF YES
1287  002604  000000                  HALT                    ;ERROR, R2 DST STUCK HIGH
```

```
1288
1289  002606  005100                    2$:     COM     R0
1290  002610  005102                            COM     R2
1291  002612  020002                            CMP     R0,R2           ;DID R2 DST SET?
1292  002614  001401                            BEQ     3$              ;BRANCH IF YES
1293  002616  000000                            HALT                    ;ERROR, R2 DST STUCK LOW
1294                                                                    ;FOR LOOPING CHANGE TO 'BR TST16'' (763)
1295  002620  020200                    3$:     CMP     R2,R0           ;DID R2 SRC SET?
1296  002622  001401                            BEQ     TST17           ;;BRANCH IF YES
1297  002624  000000                            HALT                    ;ERROR, R2 SRC STUCK LOW
1298                                                                    ;FOR LOOPING CHANGE TO 'BR TST16'' (760)
1299                                    ;;**************************************************************
1300                                    ;*TEST 17        GPR3 STUCK BIT TEST
1301                                    ;*
1302                                    ;*      LOADS GPR3 WITH ZEROS AND ONES AND COMPARES
1303                                    ;*      R3 SOURCE AND DESTINATION WITH R0.
1304                                    ;;**************************************************************
1305  002626  005000                    TST17:  CLR     R0
1306  002630  005003                            CLR     R3
1307  002632  020300                            CMP     R3,R0           ;DID R3 SRC CLEAR?
1308  002634  001401                            BEQ     1$              ;BRANCH IF YES
1309  002636  000000                            HALT                    ;ERROR, R3 SRC STUCK HIGH
1310                                                                    ;FOR LOOPING CHANGE TO 'BR TST17'' (773)
1311  002640  020003                    1$:     CMP     R0,R3           ;DID R3 DST CLEAR?
1312  002642  001401                            BEQ     2$              ;BRANCH IF YES
1313  002644  000000                            HALT                    ;ERROR, R3 DST STUCK HIGH
1314                                                                    ;FOR LOOPING CHANGE TO 'BR TST17'' (770)
1315  002646  005100                    2$:     COM     R0
1316  002650  005103                            COM     R3
1317  002652  020300                            CMP     R3,R0           ;DID R3 SRC SET TO ALL ONES?
1318  002654  001401                            BEQ     3$              ;BRANCH IF YES
1319  002656  000000                            HALT                    ;ERROR,R3 SRC STUCK LOW
1320                                                                    ;FOR LOOPING CHANGE TO 'BR TST17'' (763)
1321  002660  020003                    3$:     CMP     R0,R3           ;DID R3 DST SET TO ALL ONES?
1322  002662  001401                            BEQ     TST20           ;;BRANCH IF YES
1323  002664  000000                            HALT                    ;ERROR, R3 DST STUCK LOW
1324                                                                    ;FOR LOOPING CHANGE TO 'BR TST17'' (760)
1325                                    ;;**************************************************************
1326                                    ;*TEST 20        GPR4 STUCK BIT TEST
1327                                    ;*
1328                                    ;*      LOADS GPR4 WITH ZEROS AND ONES AND COMPARES
1329                                    ;*      R4 SOURCE AND DESTINATION WITH R0.
1330                                    ;;**************************************************************
1331  002666  005000                    TST20:  CLR     R0
1332  002670  005004                            CLR     R4
1333  002672  020400                            CMP     R4,R0           ;DID R4 SRC CLEAR?
1334  002674  001401                            BEQ     1$              ;BRANCH IF YES
1335  002676  000000                            HALT                    ;ERROR, R4 SRC STUCK HIGH
1336                                                                    ;FOR LOOPING CHANGE TO 'BR TST20'' (773)
1337  002700  020004                    1$:     CMP     R0,R4           ;DID R4 DST CLEAR?
1338  002702  001401                            BEQ     2$              ;BRANCH IF YES
1339  002704  000000                            HALT                    ;ERROR, R4 DST STUCK HIGH
1340                                                                    ;FOR LOOPING CHANGE TO 'BR TST20'' (770)
1341  002706  005100                    2$:     COM     R0
1342  002710  005104                            COM     R4
1343  002712  020004                            CMP     R0,R4           ;DID R4 DST SET?
```

```
1344  002714  001401                    BEQ     3$              ;BRANCH IF YES
1345  002716  000000                    HALT                    ;ERROR, R4 DST STUCK LOW
1346                                                            ;FOR LOOPING CHANGE TO 'BR TST20'' (763)
1347  002720  020400            3$:     CMP     R4,R0           ;DID R4 SRC SET?
1348  002722  001401                    BEQ     TST2'           ;;BRANCH IF YES
1349  002724  000000                    HALT                    ;ERROR, R4 SRC STUCK LOW
1350                                                            ;FOR LOOPING CHANGE TO 'BR TST20'' (760)
1351                              ;;*****************************************************************
1352                              ;*TEST 21        GPR5 STUCK BIT TEST
1353                              ;*
1354                              ;*      LOADS R5 WITH ZEROS AND ONES AND COMPARES
1355                              ;*      R5 SOURCE AND DESTINATION WITH R0.
1356                              ;;*****************************************************************
1357  002726  005000            TST21:  CLR     R0
1358  002730  005005                    CLR     R5
1359  002732  020500                    CMP     R5,R0           ;DID R5 SRC CLEAR?
1360  002734  001401                    BEQ     1$              ;BRANCH IF YES
1361  002736  000000                    HALT                    ;ERROR, R5 SRC STUCK HIGH
1362                                                            ;FOR LOOPING CHANGE TO 'BR TST21'' (773)
1363  002740  020005            1$:     CMP     R0,R5           ;DID R5 DST CLEAR?
1364  002742  001401                    BEQ     2$              ;BRANCH IF YES
1365  002744  000000                    HALT                    ;ERROR, R5 DST STUCK HIGH
1366                                                            ;FOR LOOPING CHANGE TO 'BR TST21'' (770)
1367  002746  005100            2$:     COM     R0
1368  002750  005105                    COM     R5
1369  002752  020005                    CMP     R0,R5           ;DID R5 DST SET TO ALL ONES?
1370  002754  001401                    BEQ     3$              ;BRANCH IF YES
1371  002756  000000                    HALT                    ;ERROR, R5 DST STUCK LOW
1372                                                            ;FOR LOOPING CHANGE TO 'BR TST21'' (763)
1373  002760  020500            3$:     CMP     R5,R0           ;DID R5 SRC SET TO ALL ONES?
1374  002762  001401                    BEQ     TST22           ;;BRANCH IF YES
1375  002764  000000                    HALT                    ;ERROR, R5 SRC STUCK LOW
1376                                                            ;FOR LOOPING CHANGE TO 'BR TST21'' (760)
1377                              ;;*****************************************************************
1378                              ;*TEST 22        GPR6 STUCK BIT TEST
1379                              ;*
1380                              ;*      LOADS R6 WITH ZEROS AND ONES AND COMPARES
1381                              ;*      R6 SOURCE AND DESTINATION WITH R0.
1382                              ;;*****************************************************************
1383  002766  005000            TST22:  CLR     R0
1384  002770  005006                    CLR     R6
1385  002772  020006                    CMP     R0,R6           ;DID R6 DST CLEAR?
1386  002774  001401                    BEQ     1$              ;BRANCH IF YES
1387  002776  000000                    HALT                    ;ERROR, R6 DST STUCK HIGH
1388                                                            ;FOR LOOPING CHANGE TO 'BR TST22'' (773)
1389  003000  020600            1$:     CMP     R6,R0           ;DID R6 SRC CLEAR?
1390  003002  001401                    BEQ     2$              ;BRANCH IF YES
1391  003004  000000                    HALT                    ;ERROR, R6 SRC STUCK HIGH
1392                                                            ;FOR LOOPING CHANGE TO 'BR TST22'' (770)
1393  003006  005100            2$:     COM     R0
1394  003010  005106                    COM     R6
1395  003012  020006                    CMP     R0,R6           ;DID R6 DST SET?
1396  003014  001401                    BEQ     3$              ;BRANCH IF YES
1397  003016  000000                    HALT                    ;ERROR, R6 DST STUCK LOW
1398                                                            ;FOR LOOPING CHANGE TO 'BR TST22'' (763)
1399  003020  020600            3$:     CMP     R6,R0           ;DID R6 SRC SET?
```

```
1400  003022  001401                    BEQ     TST23           ;;BRANCH IF YES
1401  003024  000000                    HALT                    ;ERROR,R6 SRC STUCK LOW
1402                                                            ;FOR LOOPING CHANGE TO 'BR TST22'' (760)
1403                              ;;*****************************************************************
1404                              ;*TEST 23       GPR  SHORTED BIT TEST
1405                              ;*
1406                              ;*      TEST IF GPR'S 1 THRU 6 HAVE TWO BITS TIED TOGETHER
1407                              ;*
1408                              ;*      NOTE:   R0 IS CONSIDERED 'HARDCORE''
1409                              ;;*****************************************************************
1410  003026  005000              TST23:  CLR     R0              ;THIS SECTION OF CODE
1411  003030  005200                      INC     R0              ;   *
1412  003032  006100                      ROL     R0              ;   *
1413  003034  006100                      ROL     R0              ;   *
1414  003036  005200                      INC     R0              ;   *
1415  003040  006100                      ROL     R0              ;   *
1416  003042  006100                      ROL     R0              ;   *
1417  003044  005200                      INC     R0              ;   *
1418  003046  006100                      ROL     R0              ;   *
1419  003050  006100                      ROL     R0              ;   *
1420  003052  005200                      INC     R0              ;   *
1421  003054  006100                      ROL     R0              ;   *
1422  003056  006100                      ROL     R0              ;   *
1423  003060  005200                      INC     R0              ;   *
1424  003062  006100                      ROL     R0              ;   *
1425  003064  006100                      ROL     R0              ;   *
1426  003066  005200                      INC     R0              ;   *
1427  003070  006100                      ROL     R0              ;   *
1428  003072  006100                      ROL     R0              ;   *
1429  003074  005200                      INC     R0              ;   *
1430  003076  006100                      ROL     R0              ;   *
1431  003100  006100                      ROL     R0              ;   *
1432  003102  005200                      INC     R0              ;PUTS 52525 IN R0
1433  003104  010001              2$:     MOV     R0,R1           ;PUT R0 SRC IN R1
1434  003106  020001                      CMP     R0,R1           ;IS R1 DST OK?
1435  003110  001401                      BEQ     3$              ;BRANCH IF YES
1436  003112  000000                      HALT                    ;ERROR, R1DST HAS SHORTED BITS
1437                                                            ;FOR LOOPING CHANGE TO 'BR TST15'' (605)
1438  003114  020100              3$:     CMP     R1,R0           ;IS R1 SRC OK?
1439  003116  001401                      BEQ     4$              ;BRANCH IF YES
1440  003120  000000                      HALT                    ;ERROR, R1 SRC HAS SHORTED BITS
1441                                                            ;FOR LOOPING CHANGE TO 'BR TST15'' (602)
1442  003122  010002              4$:     MOV     R0,R2
1443  003124  020002                      CMP     R0,R2           ;IS R2 DST OK?
1444  003126  001401                      BEQ     5$              ;BRANCH IF YES
1445  003130  000000                      HALT                    ;ERROR, R2 DST HAS SHORTED BITS
1446                                                            ;FOR LOOPING CHANGE TO 'BR TST15'' (576)
1447  003132  020200              5$:     CMP     R2,R0           ;IS R2 SRC OK?
1448  003134  001401                      BEQ     6$              ;BRANCH IF YES
1449  003136  000000                      HALT                    ;ERROR, R2 SRC HAS SHORTED BITS
1450                                                            ;FOR LOOPING CHANGE TO 'BR TST15'' (573)
1451  003140  010003              6$:     MOV     R0,R3           ;
1452  003142  020003                      CMP     R0,R3           ;IS R3 DST OK?
1453  003144  001401                      BEQ     7$              ;BRANCH IF YES
1454  003146  000000                      HALT                    ;ERROR, R3 DST HAS SHORTED BITS
1455                                                            ;FOR LOOPING CHANGE TO 'BR TST15'' (567)
```

```
1456  003150  020300                  7$:     CMP     R3,R0           ;IS R3 SRC OK?
1457  003152  001401                          BEQ     8$              ;BRANCH IF YES
1458  003154  000000                          HALT                    ;ERROR R3 SRC HAS SHORTED BITS
1459                                                                  ;FOR LOOPING CHANGE TO 'BR TST15'' (564)
1460  003156  010004                  8$:     MOV     R0,R4           ;
1461  003160  020004                          CMP     R0,R4           ;IS R4 DST OK?
1462  003162  001401                          BEQ     9$
1463  003164  000000                          HALT                    ;ERROR, R4 DST HAS SHORTED BITS
1464                                                                  ;FOR LOOPING CHANGE TO 'BR TST15'' (560)
1465  003166  020400                  9$:     CMP     R4,R0           ;IS R4 SRC OK?
1466  003170  001401                          BEQ     10$             ;BRANCH IF YES
1467  003172  000000                          HALT                    ;ERROR, R4 SRC HAS SHORTED BITS
1468                                                                  ;FOR LOOPING CHANGE TO 'BR TST15'' (555)
1469  003174  010005                  10$:    MOV     R0,R5
1470  003176  020005                          CMP     R0,R5           ;IS R5 DST OK?
1471  003200  001401                          BEQ     11$             ;BRANCH IF YES
1472  003202  000000                          HALT                    ;ERROR, R5 DST HAS SHORTED BITS
1473                                                                  ;FOR LOOPING CHANGE TO 'BR TST15'' (551)
1474  003204  020500                  11$:    CMP     R5,R0           ;IS R5 SRC OK?
1475  003206  001401                          BEQ     12$             ;BRANCH IF YES
1476  003210  000000                          HALT                    ;ERROR, R5 SRC HAS SHORTED BITS
1477                                                                  ;FOR LOOPING CHANGE TO 'BR TST15'' (546)
1478  003212  010006                  12$:    MOV     R0,R6           ;
1479  003214  020006                          CMP     R0,R6           ;IS R6 DST OK?
1480  003216  001401                          BEQ     13$             ;BRANCH IF YES
1481  003220  000000                          HALT                    ;ERROR, R6 DST HAS SHORTED BITS
1482                                                                  ;FOR LOOPING CHANGE TO 'BR TST15'' (542)
1483  003222  020600                  13$:    CMP     R6,R0           ;IS R6 SRC OK?
1484  003224  001401                          BEQ     14$             ;BRANCH IF YES
1485  003226  000000                          HALT                    ;ERROR, R6 SRC HAS SHORTED BITS
1486                                                                  ;FOR LOOPING CHANGE TO 'BR TST15'' (537)
1487  003230  005006                  14$:    CLR     SP              ;THIS CODE PUTS
1488  003232  005206                          INC     SP              ;1100 IN THE SP
1489  003234  006106                          ROL     SP
1490  003236  006106                          ROL     SP
1491  003240  006106                          ROL     SP
1492  003242  005206                          INC     SP
1493  003244  006106                          ROL     SP
1494  003246  006106                          ROL     SP
1495  003250  006106                          ROL     SP
1496  003252  006106                          ROL     SP
1497  003254  006106                          ROL     SP
1498  003256  006106                          ROL     SP
1499                                  ;;=================================================================
1500  003260  005000                          CLR     R0              ;THIS CODE
1501  003262  005200                          INC     R0              ;INITIALIZES
1502  003264  006100                          ROL     R0              ;THE
1503  003266  006100                          ROL     R0              ;TEST
1504  003270  006100                          ROL     R0              ;NUMBER
1505  003272  005200                          INC     R0
1506  003274  006100                          ROL     R0              ;STORAGE
1507  003276  005200                          INC     R0              ;REGISTER
1508  003300  012737  000044  177770          MOV     #44,@#177770    ;SETUP MICROPROCESSOR BREAK REG
1509                                          .SBTTL
1510                                  ;;=================================================================
1511                                  ;*TEST 24       ONE   MICROSTATE  (E/CLASS*DM0*DF7)
```

```
1512                                    ;*
1513                                    ;*      THIS TEST EXECUTES AN ADD INSTRUCTION WITH SMO,DMO, AND DF7.
1514                                    ;*      IF THE TEST FAILS THE SAME FLOW (EXC. 90) IS
1515                                    ;*      TRIED WITH A PENDING BRQ (T BIT TRAP) INSTEAD OF A DF7. IF THIS TEST FAILS
1516                                    ;*      A FORK A FAILURE IS REPORTED. IF IT PASSES, A TEST 1 FAILURE IS REPORTED.
1517                                    ;*
1518                                    ;*      ROM FLOW-30
1519                                    ;;**************************************************************************
1520  003306  005200                    TST24:  INC     R0              ;INCREMENT TEST NUMBER
1521  003310  005005                            CLR     R5              ;ENSURE R5 CLEAR
1522  003312  005205                            INC     R5              ;SET R5 EQUAL
1523  003314  006105                            ROL     R5              ;TO 52
1524  003316  006105                            ROL     R5              ;WHICH
1525  003320  005205                            INC     R5              ;WILL CAUSE
1526  003322  006105                            ROL     R5              ;A JUMP
1527  003324  006105                            ROL     R5              ;TO TESTCC
1528  003326  005205                            INC     R5
1529  003330  006105                            ROL     R5              ;TEST
1530  003332                            SYNC24:
1531  003332  000264                            SEZ                     ;ENSURE Z SET
1532  003334                            IUT24:
1533  003334  060507                            ADD     R5,PC           ;ADD SHOULD SKIP TO TAG TESTCC
1534                                    ;FAILURE- TRY E/CLASS*BRQ*DMO
1535  003336  012737  003366  000014            MOV     #FORKA,@#14     ;SETUP VECTOR FOR RETURN
1536  003344  005005                            CLR     R5              ;ENSURE R5 CLEAR
1537                                    ;;**************************************************************************
1538  003346  012746  000020                    MOV     #BIT4,-(SP)     ;THIS CODE
1539  003352  012746  003360                    MOV     #1$,-(SP)       ;SETS THE
1540  003356  000006                            RTT                     ;T BIT
1541                                    ;;**************************************************************************
1542  003360  005205                    1$:     INC     R5              ;TRAP HERE IF FORK A OK
1543  003362  005205                            INC     R5              ;WILL EXECUTE IF FORK A FAILED
1544  003364  000240                            NOP                     ;ALLOW T BIT TRAP IF FORK FAILED
1545  003366  012737  000016  000014    FORKA:  MOV     #16,@#14        ;RESTORE T BIT VECTOR
1546  003374  062706  000004                    ADD     #4,SP           ;RESTORE SP
1547  003400  006005                            ROR     R5              ;IS R5 1 OR 2?
1548  003402  103401                            BCS     1$              ;BRANCH ON 1 (FORK A OK)
1549  003404  000000                            HALT                    ;FORK A FAILURE INTO ROM STATE EXC.90
1550                                            ;FOR LOOPING CHANGE TO 'BR TST24+2'' (741
```

```
1551  003406
1552  003406  000000                1$:
1553                                      HALT                    ;EITHER PCB DID NOT LOAD OR RACH DF7 STUCK HIGH
1554                                                              ;FOR LOOPING CHANGE TO 'BR TST24+2'' (740)
1555  003410
1556  003410  001001                TESTCC:
1557  003412  000000                      BNE     TST25           ;;GO TO NEXT TEST IF CCLD$ OK
1558                                      HALT                    ;STATE EXC.90 BAD
1559                                                              ;FOR LOOPING CHANGE TO 'BR TST24+2'' (736)
1560                                      .SBTTL
1561                               ;;*****************************************************************
1562                               ;*TEST 25       TWO   MICROSTATES (NEG*DM0)
1563                               ;*
1564                               ;*      THIS TEST EXECUTES A NEGATE INSTRUCTION WITH DM0.
1565                               ;*
1566                               ;*      IF FORK A FAILS EXECUTION WOULD GO TO EITHER RSD.00, ZAP.00,
1567                               ;*      OR FOP.00.
1568                               ;*      FOP.00 WILL CAUSE THE PROCESSOR TO HANG.
1569                               ;*      RSD.00 WOULD CAUSE A TRAP TO LOCATION 10. THIS WOULD ONLY HAPPEN
1570                               ;*      IF RACH NEG.B*DM0 DID NOT GO HIGH.
1571                               ;*      ZAP.00 WOULD CAUSE A TRAP TO LOCATION 24. THIS WOULD ONLY HAPPEN
1572                               ;*      IF RACH A2 RAB00 DID NOT GO LOW.
1573                               ;*
1574                               ;*      ROM FLOW-301,210
1575  003414  005200               ;;*****************************************************************
1576  003416  005004               TST25:  INC     RC              ;INCREMENT TEST NUMBER
1577  003420  005005                       CLR     R4              ;INITIALIZE CC ERROR RECORD
1578  003422  005205                       CLR     R5              ;SET UP R5
1579  003424  000257                       INC     R5              ;FOR TEST
1580  003426                               CCC                     ;
1581  003426  000266               SYNC25:
1582  003430                               *SEZ!SEV                ;CC'S=0110
1583  003430  005405               IUT25:
1584  003432  000401                       NEG     R5              ;EXECUTE NEGATE CC'S 1001
1585  003434  000000                       BR      1$              ;GET OVER ERROR CALL
1586                                       HALT                    ;RACH E57(6) DOES NOT GO LOW
1587  003436  100003                                               ;FOR LOOPING CHANGE TO 'BR TST25'' 767
1588  003440  001402               1$:     BPL     NEGR5
1589  003442  102401                       BEQ     NEGR5
1590  003444  103401                       BVS     NEGR5
1591  003446  005204                       BCS     .+4
1592  003450  005001               NEGR5:  INC     R4              ;ERROR, INCORRECT CC'S AFTER NEG.
1593  003452  005301                       CLR     R1              ;SET R1 TO
1594  003454  020501                       DEC     R1              ;-1 WITHOUT NEGATING
1595  003456  001414                       CMP     R5,R1           ;DID R5 GET -1?
1596  003460  005704                       BEQ     R5OK            ;BRANCH IF R5 OK
1597  003462  001405                       TST     R4              ;IS THERE A CC PROBLEM?
1598  003464  022705  177776               BEQ     3$              ;BRANCH IF NO
1599  003470  001001                       CMP     #177776,R5      ;DID NEG ONE'S COMPLEMENT?
1600  003472  000000                       BNE     2$              ;BRANCH IF NO
1601                                       HALT                    ;CC'S BAD AND NEG.90 DID NOT ADD 1
1602  003474                                                       ;FOR LOOPING CHANGE TO 'BR TST25+2'' (751)
1603  003474  000000               2$:
1604                                       HALT                    ;CC'S BAD AND R5 BAD
1605  003476  022705  177776                                       ;FOR LOOPING CHANGE TO 'BR TST25+2'' (750)
1606  003502  001001               3$:     CMP     #177776,R5      ;DID NEGATE DO A ONE'S COMPLIMENT?
                                       BNE     4$              ;BRANCH IF NO
```

```
1607  003504  000000                          HALT                    ;CC'S OK BUT NEG.90 DID NOT ADD 1
1608                                                                  ;FOR LOOPING CHANGE TO 'BR TST25+2'' (744)
1609  003506                                4$:
1610  003506  000000                          HALT                    ;CC'S OK BUT R5 BAD
1611                                                                  ;FOR LOOPING CHANGE TO 'BR TST25+2'' (743)
1612  003510  005704                    R5OK: TST     R4              ;CC PROBLEM?
1613  003512  001401                          BEQ     TST26           ;;GO TO NEXT TEST IF NO
1614  003514  000000                          HALT                    ;NEGATE OK BUT INCORRECT CC'S
1615                                                                  ;FOR LOOPING CHANGE TO 'BR TST25+2'' (740)
1616                                          .SBTTL
1617                                  ;;**********************************************************************
1618                                  ;*TEST 26       THREE MICROSTATES (BIN*SM1*DM0*-DF7*SR0(0)
1619                                  ;*
1620                                  ;*      IF FORK A FAILS EXECUTION WILL GO TO EITHER EXEC.80 OR D12.00.
1621                                  ;*      EXC.80 WILL HANG THE PROCESSOR IN THE PAUSE STATE AT MICRO ADDRESS 343.
1622                                  ;*      THIS WILL ONLY HAPPEN IF RACL RADR00 IS NOT GOING LOW DUE
1623                                  ;*      TO RACF A1 RAB00 (AFIR59(1)*[-BIN+SM01]*U/CLASS).
1624                                  ;*      D12.00 WOULD MOV THE PC TO LOCATION 0.
1625                                  ;*
1626                                  ;*      IF FORK C FAILS EXECUTION WOULD GO FROM S13.10 TO D00.80 OR
1627                                  ;*      D45.01 OR S13.20 OR D12.00 OR JSR.10 OR ASC.80 OR RTI.50 OR ASH.20 OR FOP.50.
1628                                  ;*      D00.80 WOULD SWAP THE BYTES OF THE SOURCE OPERAND BEFORE
1629                                  ;*      PUTTING THEM IN R5.
1630                                  ;*      D45.01 WOULD MOVE THE PC TO LOCATION 0.
1631                                  ;*      S13.20 WILL EXECUTE A SM3 (AND NO AUTO INC) INSTR.  THIS WILL CAUSE
1632                                  ;*      AN ODD ADDRESS TRAP SINCE LOCATION POSERR CONTAINS AN ODD WORD.
1633                                  ;*      JSR.10 WOULD PUSH THE ADDR OF POSERR ONTO THE STACK.
1634                                  ;*      ASC.80 WILL HALT AT 8$.
1635                                  ;*      RTI.50 WILL CAUSE 1004XX TO BE PLACED IN THE PS WORD
1636                                  ;*      AND THE PROCESSOR WILL TRAP TO LOCATION 14.
1637                                  ;*      FOP.50 WILL ?????.
1638                                  ;*      ASH.20 WOULD CAUSE A BAD CC.
1639                                  ;*      IF THE SRC CONST FAILS IN STATE S13.00 AND ADDS 1 OR 3, AN ODD
1640                                  ;*      ADDRESS TRAP WILL OCCUR.
1641                                  ;*      IF THE SRC CONSTANT ADDS 2, THE ERROR AT 6$ WILL REPORT THE FAILURE.
1642                                  ;*
1643                                  ;*      ROM FLOW-21,27,205
1644                                  ;**********************************************************************
1645  003516  005200                  TST26:  INC     R0              ;INCREMENT TEST NUMBER
1646  003520  005005                          CLR     R5              ;SETUP R5
1647  003522                          SYNC26:
1648  003522  000244                          CLZ                     ;ENSURE Z CLEAR TO CATCH A FAILURE TO ASC.80
1649  003524                          IUT26:
1650  003524  011705                          MOV     (PC), R5        ;MOVE 1004XX TO R5(XX MUST BE
1651                                                                  ;ODD TO CATCH A FAILURE TO S13.20)
1652  003526  100443                  POSERR: BMI     7$              ;BRANCH IF CC OK(ENSURE OFFSET IS ODD)
1653  003530  100441                          BMI     6$              ;BRANCH IF SRC CONST ADDED 2
1654                                  ;FAILURE
1655  003532  013767  177776  175436        MOV     @#PSW,$ERPSW    ;SAVE ERROR PSW
1656  003540  022737  003526  000000  1$:     CMP     #POSERR,@#0     ;DID FORK A GO TO D12.00 OR FORK C TO D45,01?
1657  003546  001006                          BNE     3$              ;BRANCH IF NO
1658                                  ;EITHER FORK A OR C FAILED-FIND OUT WHICH ONE
1659  003550  005005                          CLR     R5              ;SETUP R5
1660  003552  012501                          MOV     (R5)+,R1        ;TEST TO SEE IF FORK A OR FORK C FAILED
1661  003554  005705                          TST     R5              ;DID R5 INCREMENT
1662  003556  001401                          BEQ     2$              ;BRANCH IF NO
```

```
1663  003560  000000                        HALT                    ;FORK C FAILED,EITHER IRCC DM0 NOT GETTING
1664                                                                  ;THRU TO RACL RADR07 OR IRCC DM0 IS STUCK HIGH
1665                                                                  ;FOR LOOPING CHANGE TO 'BR TST26+2'' (757)
1666  003562                        2$:
1667  003562  000000                        HALT                    ;FORK A FAILED, RACH A1 RAB04 NOT GOING LOW
1668                                                                  ;DUE TO EITHER RACE:BF1=7 OR
1669                                                                  ;BF1=0 OR SM0 STUCK LOW OR RACH E11 BAD
1670                                                                  ;FOR LOOPING CHANGE TO 'BR TST26+2'' (756)
1671  003564  000305                3$:     SWAB    R5              ;SETUP R5 TO TEST IF INSTR. WENT THRU D00.80
1672  003566  020537  003526              CMP     R5,@#POSERR     ;DID INSTR GO THRU D00.3?
1673  003572  001001                      BNE     4$              ;BRANCH IF NO
1674  003574  000000                      HALT                    ;IRCC RAB00 NOT GETTING TO RACL RADR00
1675                                                                ;FOR LOOPING CHANGE TO 'BR TST26+2'' (751)
1676  003576  026627  000010  003526  4$:     CMP     10(SP),#POSERR  ;DID FORK C GO TO JSR.10?
1677  003604  001001                      BNE     5$              ;BRANCH IF NO
1678  003606  000000                      HALT                    ;INPUT TO IRCC E40 PIN 5 STUCK HIGH
1679                                                                ;FOR LOOPING CHANGE TO 'BR TST26+2'' (744)
1680  003610  032767  000017  175560  5$:     BIT     #17,$ERPSW      ;DID ALL CC'S CLEAR?
1681  003616  001404                      BEQ     9$              ;BRANCH IF YES
1682  003620  032767  000020  175350        BIT     #PIT4,$ERPSW    ;DID Z BIT SET?
1683  003626  001401                      BEQ     8$              ;BRANCH IF NO
1684  003630                        9$:
1685  003630  000000                       HALT                   ;BAD CONDITION CODE
1686                                                                ;FOR LOOPING CHANGE TO 'BR TST26+2'' (733)
1687  003632                        8$:
1688  003632  000000                        HALT                    ;EITHER INPUT TO C MUX STUCK LOW OR MUX BAD OR
1689                                                                  ;;C0 RAB01 STUCK LOW OR RACL RADR01 INPUT STUCK LOW
1690                                                                  ;FOR LOOPING CHANGE TO 'BR TST26+2'' (732)
1691  003634                        6$:
1692  003634  000000                       HALT                    ;SRC CONST EQUAL TO 2 SHOULD BE 0
1693                                                                ;FOR LOOPING CHANGE TO 'BR TST26+2'' (731)
1694  003636  000241                7$:     CLC                     ;ENSURE C CLEAR
1695  003640  006105                       ROL     R5              ;CHECK IF R5 WAS LOADED
1696  003642  103401                       BCS     TST27           ;;GO TO NEXT TEST IF C SET
1697  003644  000000                       HALT                    ;ERROR, R5 DID NOT LOAD
1698                                                                ;FOR LOOPING CHANGE TO 'BR TST26+2'' (725)
1699                                ;;*****************************************************************
1700                                ;*TEST 27       THREE MICROSTATES (BIN*SM2*DM0*-DF7*SR0(0)
1701                                ;*
1702                                ;*      THIS TEST IS THE SAME AS THE PREVIOUS TEST EXCEPT FOR THE
1703                                ;*      SM. A WORD AND BYTE INSTRUCTION IS EXECUTED TO VERIFY THAT STATE
1704                                ;*      S13.01 ADDS THE CORRECT SOURCE CONSTANT.
1705                                ;*
1706                                ;*      IF FORK A FAILS EXECUTION WILL GO TO EXC.80.
1707                                ;*      EXC.80 WILL HANG THE PROCESSOR IN THE PAUSE STATE AT MICRO ADDRESS 343.
1708                                ;*      THIS WILL ONLY HAPPEN IF RACL RADR01
1709                                ;*      IS NOT GOING LOW DUE TO RACF A1 RAB01 (AFIR10(1)*U/CLASS).
1710                                ;*
1711                                ;*      ROM FLOW-22,27,205
1712                                ;*****************************************************************
1713  003646  005200                TST27:  INC     R0              ;INCREMENT TEST NUMBER
1714  003650  005005                        CLR     R5              ;SETUP R5
1715  003652  000240                SYNC27: NOP
1716  003654                        IUT27:
1717  003654  012705                        MOV     (PC)+,R5        ;MOVE 000401 TO R5
1718  003656  000401                ER25:   BR      3$
```

```
1719  003660  000412                        BR      4$
1720                                  ;FAILURE-SOURCE CONSTANT FAILED. TRY SM3
1721  003662  012705  012006          3$:   MOV     #SUBTAB,R5      ;GET ADDRESS OF LOCATION THAT CONTAINS ADDR.
1722  003666  010501                        MOV     R5,R1           ;SAVE R5 IN R1
1723  003670  013502                        MOV     @(R5)+,R2       ;EXECUTE AN SM3 INSTRUCTION
1724  003672  005201                        INC     R1              ;ADJUST R1 TO
1725  003674  005201                        INC     R1              ;LOOK LIKE R5
1726  003676  020501                        CMP     R5,R1           ;DID R5 AUTO INCREMENT?
1727  003700  001401                        BEQ     2$              ;BRANCH IF YES
1728  003702  000000                        HALT                    ;SOURCE CONST FAILURE ON IRC
1729                                                                ;FOR LOOPING CHANGE TO 'BR TST27+2'' (762)
1730  003704                          2$:
1731  003704  000000                        HALT                    ;SRC CONST FAILURE EITHER ON IRC OR DAP
1732                                                                ;FOR LOOPING CHANGE TO 'BR TST27+2'' (761)
1733  003706  010705                  4$:   MOV     PC,R5           ;SETUP R5 TO HOLD ADDRESS
1734  003710  010501                        MOV     R5,R1           ;SAVE R5 IN R1
1735  003712  112502                        MOVB    (R5)+,R2        ;TEST TO SEE IF SCR CONST 1 ON BYTE
1736  003714  005201                        INC     R1              ;SETUP R1 TO LOOK LIKE R5
1737  003716  020501                        CMP     R5,R1           ;DID R5 AUTOINCREMENT BY 1?
1738  003720  001410                        BEQ     TST30           ;;BRANCH IF YES
1739                                  ;FAILURE-SOURCE CONSTANT FAILED ON BYTE. TRY SM4
1740  003722  010705                        MOV     PC,R5           ;PUT ADDRESS IN R5
1741  003724  010501                        MOV     R5,R1           ;SAVE R5 IN R1
1742  003726  114502                        MOVB    -(R5),R2        ;EXECUTE AN SM4 INSTRUCTION
1743  003730  005301                        DEC     R1              ;ADJUST R1 TO LOOK LIKE R5
1744  003732  020501                        CMP     R5,R1           ;DID R5 AUTO DECREMENT?
1745  003734  001001                        BNE     5$              ;BRANCH IF NO
1746  003736  000000                        HALT                    ;IRCC SRCM2 STUCK HIGH INTO IRC E8
1747                                                                ;FOR LOOPING CHANGE TO 'BR TST27+2'' (744)
1748  003740                          5$:
1749  003740  000000                        HALT                    ;BYTE SRC CONST FAILURE EITHER ON IRC OR DAP
1750                                                                ;FOR LOOPING CHANGE TO 'BR TST27+2'' (743)
1751                                  ;;*****************************************************************
1752                                  ;*TEST 30       ALU CARRY FUNCTIONAL TEST
1753                                  ;*
1754                                  ;*      THIS TEST DOES A COMPLETE CHECK OF THE ALU CARRY FUNCTIONS
1755                                  ;*      THE FIRST SECTION ENSURES THAT ALL THE INPUT AND OUTPUT LINES ARE
1756                                  ;*      OK AND THE REST OF THE TEST ENSURES THAT THE CARRY LOGIC IS OK.
1757                                  ;*      FOLLOWING ARE THE LOGIC EQUATIONS FOR A 74S181 AND 74S182 THAT
1758                                  ;*      DICTATED THE PATTERNS USED IN EACH SECTION:
1759                                  ;*         74S181
1760                                  ;*              G=A3*B3+A2*B2*(A3+B3)+A1*B1*(A2+B2)*(A3+B3)+A0*B0*(A1+B1)*(A2+B2)*(A3+B3
1761                                  ;*              P=(A3+B3)*(A2+B2)*(A1+B1)*(A0+B0)
1762                                  ;*              COUT=G+P*CIN
1763                                  ;*         74S182
1764                                  ;*              CX=G0+P0*CIN
1765                                  ;*              CY=G1+P1*G0+P1*P0*CIN
1766                                  ;*              CZ=G2+P2*G1+P2*P1*G0+P2*P1*P0*CIN
1767                                  ;;*****************************************************************
1768  003742  005200                  TST30:  INC     R0
1769                                  ;SECTION 1-INPUT/OUTPUT BIT TEST
1770  003744  012701  125252                  MOV     #125252,R1      ;PUT DATA PATTERN IN R1
1771  003750  062701  052525                  ADD     #52525,R1       ;ADD COMPLIMENT PATTERN
1772  003754  005101                          COM     R1              ;MAKE RESULT 0
1773  003756  001401                          BEQ     2$              ;BRANCH IF IT IS ZERO
1774  003760  000000                          HALT                    ;BIT FAILED DURING ADD
```

```
1775
1776  003762  012701  052525          2$:     MOV     #52525,R1       ;FOR LOOPING CHANGE TO 'BR TST30+2'' (771)
1777  003766  062701  125252                  ADD     #125252,R1      ;PUT PATTERN IN R1
1778  003772  005101                          COM     R1              ;ADD COMPLIMENT PATTERN
1779  003774  001401                          BEQ     3$              ;MAKE IT ZERO
1780  003776  000000                          HALT                    ;BRANCH IF IT WENT TO ZERO
1781                                                                  ;BIT FAILED DURING ADD
1782                                                                  ;FOR LOOPING CHANGE TO 'BR 2$'' (771)
1783                                  ;;*****************************************************************
1784  004000  012702  167357          ;SECTION 2-G=A3*B3
1785  004004  012701  104210          3$:     MOV     #167357,R2      ;PUT COMPLIMENT OF EXPECTED PATTERN IN R2
1786  004010  060101                          MOV     #104210,R1      ;PUT PATTERN IN R1
1787  004012  103401                          ADD     R1,R1           ;ADD IT TO ITSELF
1788  004014  000000                          BCS     4$              ;BRANCH IF DAPH COUT15 OK
1789                                          HALT                    ;DAPH COUT15 DID NOT GO LOW
1790  004016  050201                                                  ;FOR LOOPING CHANGE TO 'BR 3$'' (771)
1791  004020  005101                  4$:     BIS     R2,R1           ;MAKE R1 -1
1792  004022  001401                          COM     R1              ;MAKE IT ZERO
1793  004024  000000                          BEQ     5$              ;BRANCH IF IT IS
1794                                          HALT                    ;A G LINE FAILED
1795                                                                  ;FOR LOOPING CHANGE TO 'BR 3$'' (765)
1796                                  ;;*****************************************************************
1797  004026  012701  046314          ;SECTION 3-G=A2*B2*(A3+B3)
1798  004032  062701  042104          5$:     MOV     #146314,R1      ;PUT PATTERN IN R1
1799  004036  050201                          ADD     #42104,R1       ;ADD PATTERN
1800  004040  005101                          BIS     R2,R1           ;MAKE R1 -1
1801  004042  001401                          COM     R1              ;MAKE IT ZERO
1802  004044  000000                          BEQ     6$              ;BRANCH IF IT IS ZERO
1803                                          HALT                    ;A G LINE FAILED
1804  004046  012701  042104                                          ;FOR LOOPING CHANGE TO 'BR 5$'' (770)
1805  004052  062701  146314          6$:     MOV     #42104,R1       ;REVERSE INPUTS
1806  004056  050201                          ADD     #146314,R1      ;TO ALU
1807  004060  005101                          BIS     R2,R1           ;MAKE R1 -1
1808  004062  001401                          COM     R1              ;MAKE IT ZERO
1809  004064  000000                          BEQ     7$              ;BRANCH IF IT IS
1810                                          HALT                    ;A G LINE FAILED
1811                                                                  ;FOR LOOPING CHANGE TO 'BR 6$'' (770)
1812                                  ;;*****************************************************************
1813  004066  012701  167356          ;SECTION 4-G=A1*B1*(A2+B2)*(A3+B3)
1814  004072  062701  021042          7$:     MOV     #167356,R1      ;PUT PATTERN IN R1
1815  004076  050201                          ADD     #21042,R1       ;ADD PATTERN
1816  004100  005101                          BIS     R2,R1           ;MAKE R1 -1
1817  004102  001401                          COM     R1              ;MAKE IT ZERO
1818  004104  000000                          BEQ     8$              ;BRANCH IF IT IS
1819                                          HALT                    ;A G LINE FAILED
1820  004106  012701  021042                                          ;FOR LOOPING CHANGE TO 'BR 7$'' (770)
1821  004112  062701  167356          8$:     MOV     #21042,R1       ;REVERSE INPUTS
1822  004116  050201                          ADD     #167356,R1      ;TO THE ALU
1823  004120  005101                          BIS     R2,R1           ;MAKE R1 -1
1824  004122  001401                          COM     R1              ;MAKE IT ZERO
1825  004124  000000                          BEQ     9$              ;BRANCH IF IT IS
1826                                          HALT                    ;A G LINE FAILED
1827                                                                  ;FOR LOOPING CHANGE TO 'BR 8$'' (770)
1828                                  ;;*****************************************************************
1829  004126  012701  177777          ;SECTION 5-G=A0*B0*(A1+B1)*(A2+B2)*(A3+B3)
1830  004132  062701  010421          9$:     MOV     #-1,R1          ;PUT PATTERN IN R1
                                              ADD     #10421,R1       ;ADD PATTERN TO R1
```

```
1831  004136  050201                  BIS     R2,R1           ;MAKE R1 -1
1832  004140  005101                  COM     R1              ;MAKE IT ZERO
1833  004142  001401                  BEQ     10$             ;BRANCH IF IT IS
1834  004144  000000                  HALT                    ;A G LINE FAILED
1835                                                          ;FOR LOOPING CHANGE TO 'BR 9$'' (770)
1836  004146  012701  010421  10$:    MOV     #10421,R1       ;REVERSE INPUTS
1837  004152  062701  177777          ADD     #-1,R1          ;TO THE ALU
1838  004156  050201                  BIS     R2,R1           ;MAKE R1 -1
1839  004160  005101                  COM     R1              ;MAKE IT ZERO
1840  004162  001401                  BEQ     11$             ;BRANCH IF IT IS
1841  004164  000000                  HALT                    ;A G LINE FAILED
1842                                                          ;FOR LOOPING CHANGE TO 'BR 10$'' (770)
1843                                ;;*****************************************************************
1844                                ;SECTION 6-P OUTPUTS AND CX-P0*CIN, CY=P1*P0*CIN, CZ=P2*P1*P0*CIN
1845  004166  012701  177777  11$:    MOV     #-1,R1          ;PUT PATTERN IN R1
1846  004172  000261                  SEC                     ;SET C
1847  004174  005501                  ADC     R1              ;CAUSES CARRY TO GO ALL THE WAY
1848  004176  103401                  BCS     12$             ;BRANCH IF CARRY CAME OUT
1849  004200  000000                  HALT                    ;DAPH COUT15 DID NOT GO LOW
1850                                                          ;FOR LOOPING CHANGE TO 'BR 11$'' (772)
1851  004202  001401  12$:    BEQ     13$             ;BRANCH IF R1 WENT TO ZERO
1852  004204  000000                  HALT                    ;EITHER A P LINE OR THE 74S182 FAILED
1853                                                          ;FOR LOOPING CHANGE TO 'BR 11$'' (770)
1854                                ;;*****************************************************************
1855                                ;SECTION 7-CY=P1*G0
1856  004206  012701  000370  13$:    MOV     #370,R1         ;PUT PATTERN IN R1
1857  004212  062701  000010          ADD     #10,R1          ;ADD DATA TO R1
1858  004216  052701  177377          BIS     #177377,R1      ;MAKE R1 -1
1859  004222  005101                  COM     R1              ;MAKE IT ZERO
1860  004224  001401                  BEQ     14$             ;BRANCH IF IT WORKED
1861  004226  000000                  HALT                    ;DAPF E44 FAILED
1862                                                          ;FOR LOOPING CHANGE TO 'BR 13$'' (767)
1863                                ;;*****************************************************************
1864                                ;SECTION 8-CZ-P2*G1
1865  004230  012701  007600  14$:    MOV     #7600,R1        ;PUT DATA IN R1
1866  004234  062701  000200          ADD     #200,R1         ;ADD DATA TO R1
1867  004240  052701  167777          BIS     #167777,R1      ;MAKE R1 -1
1868  004244  005101                  COM     R1              ;MAKE IT ZERO
1869  004246  001401                  BEQ     15$             ;BRANCH IF IT WORKED
1870  004250  000000                  HALT                    ;DAPF E44 FAILED
1871                                                          ;FOR LOOPING CHANGE TO 'BR 14$'' (767)
1872                                ;;*****************************************************************
1873                                ;SECTION 9-CZ P2*P1*G0
1874  004252  012701  007770  15$:    MOV     #7770,R1        ;PUT DATA IN R1
1875  004256  062701  000010          ADD     #10,R1          ;ADD DATA TO R1
1876  004262  052701  167777          BIS     #167777,R1      ;MAKE R1 -1
1877  004266  005101                  COM     R1              ;MAKE IT ZERO
1878  004270  001401                  BEQ     TST31           ;;BRANCH IF IT WORKED
1879  004272  000000                  HALT                    ;DAPF E44 FAILED
1880                                                          ;FOR LOOPING CHANGE TO 'BR 15$'' (767)
1881                                ;;*****************************************************************
1882                                ;*TEST 31        THREE MICROSTATES (DAC*DM2*0/CLASS)
1883                                ;*
1884                                ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00. THIS WILL ONLY HAPPEN
1885                                ;*      IF RACE A0 RAB01 DOES NOT GO LOW OR DOES NOT GET THRU
1886                                ;*      TO RACL RADR01.  THIS WILL CAUSE A TRAP TO LOCATION 10.
```

```
                                ;*
                                ;*      IF BEN15 FAILS EXECUTION WILL GO TO D45.80.
                                ;*      THIS WILL CAUSE A TRAP TO 4 WITH THE ADDRESS OF 1$ ON THE STACK.
                                ;*
                                ;*      IF THE DESTINATION CONSTANT FAILS(ADDS 1 OR 3) IN STATE D12.60
                                ;*      AN ODD ADDRESS TRAP WILL OCCUR.
                                ;*      IF THE DST CONST ADDS 0, THE ERROR AT 3$ WILL REPORT THE FAILURE.
                                ;*      IF THE DESTINATION IS NOT LOADED WITH THE SOURCE, THE
                                ;*      ERROR AFTER 5$ WILL REPORT THE FAILURE.
                                ;*
                                ;*      ROM FLOW-2,155,312
                                ;;*****************************************************************
004274  005200          TST31:  INC     R0              ;INCREMENT TEST NUMBER
004276  012705                  MOV     (PC)+,R5        ;PUT 'BR 4$'' IN R5
004300  000401                  .WORD   000401          ;CONTAINS BINARY OF 'BR 4$''
004302  000240          SYNC31: NOP
004304                  IUT:
004304  010527          1$:     MOV     R5,(PC)+        ;EXECUTE INSTRUCTION UNDER TEST
004306  000401                  BR      4$              ;WILL EXECUTE THIS IF INSTR FAILS TO AUTO INC
004310  000415                  BR      5$              ;AUTO INC OK, GO CHECK IF LOAD OK
                        ;FAILURE-DESTINATION CONSTANT FAILED. TRY DM4.
004312  012705  001164    4$:     MOV     #$TMP1,R5       ;PUT ADDRESS IN R5
004316  010125                  MOV     R1,(R5)+        ;EXECUTE INSTRUCTION UNDER TEST
004320  022705  001164            CMP     #$TMP1,R5       ;DID R5 STAY THE SAME?
004324  001006                  BNE     2$              ;BRANCH IF NO
004326  010145                  MOV     R1,-(R5)        ;SEE IF AUTO DEC. WORKS
004330  022705  001162            CMP     #$TMP1-2,R5     ;DID R5 AUTO DEC?
004334  001001                  BNE     3$              ;BRANCH IF NO
004336  000000                  HALT                    ;AUTO DEC WORKS SO ROM STATE D12.60 PROBABLY BAD
                                                        ;FOR LOOPING CHANGE TO 'BR TST31+2'' (757)
004340                  3$:
004340  000000                  HALT                    ;IRCD DSTCON=2 EITHER STUCK LOW OR
                                                        ;NOT GETTING THRU KOMUX
                                                        ;FOR LOOPING CHANGE TO 'BR TST31+2'' (756)
004342                  2$:
004342  000000                  HALT                    ;BEN15 OK & DST CONST OK BUT INSTR DOSEN'T WORK
                                                        ;FOR LOOPING CHANGE TO 'BR TST31+2'' (755)
004344  012705  004304  5$:     MOV     #1$,R5          ;GET ADDR OF INSTR UNDER TEST
004350  011505                  MOV     (R5),R5         ;GET INSTR UNDER TEST
004352  022705  000401            CMP     #401,R5         ;DID AUTO DEC OCCUR?
004356  001005                  BNE     8$              ;BRANCH IF NO
004360  012705  010027            MOV     #10027,R5       ;GET OP CODE OF INSTR UNDER TEST
004364  010567  177714            MOV     R5,1$           ;RESTORE INSTR UNDER TEST
004370  000000                  HALT                    ;EITHER RACK BRCAB05 NOT GOING LOW OR
                                                        ;IT IS NOT GETTING THRU RACL RADR05
                                                        ;FOR LOOPING CHANGE TO 'BR TST31+2'' (742)
004372  010527          8$:     MOV     R5,(PC)+        ;TEST TO SEE IF (PC)+ WAS LOADED
004374  100000          7$:     .WORD   100000          ;STORAGE LOCATION FOR PREVIOUS INSTR.
004376  012701  004374            MOV     #7$,R1          ;PUT ADDRESS OF 7$ IN R1
004402  011102                  MOV     (R1),R2         ;GET CONTENTS OF 7$
004404  100001                  BPL     6$              ;BRANCH IF LOAD OK
004406  000000                  HALT                    ;ERROR, (PC)+ DID NOT LOAD CORRECTLY
                                                        ;FOR LOOPING CHANGE TO 'BR TST31+2'' (733)
004410  012702  100000  6$:     MOV     #BIT15,R2       ;SET SIGN BIT IN R2
004414  010221                  MOV     R2,(R1)+        ;RESTORE 7$
                                ;;*****************************************************************
```

```
                                ;*TEST 32       THREE MICROSTATES (DAC*DM1*O/CLASS)
                                ;*
                                ;*      THIS TEST IS THE SAME AS THE PREVIOUS TEST EXCEPT FOR THE DM.
                                ;*      IF FORK A FAILS, EXECUTION WILL GO TO RSD.00.
                                ;*      THIS WILL CAUSE A TRAP TO LOCATION 10 WITH AN ODD ADDRESS ERROR.
                                ;*      THIS WILL ONLY HAPPEN IF RACE A0 RAB00 DOES NOT GO LOW OR
                                ;*      DOES NOT GET TO RACL.
                                ;*      EITHER E44 OR E6 IS BAD.
                                ;*
                                ;*      IF THE INSTRUCTION FAILS TO MOVE R5 TO THE PC INDIRECT,
                                ;*      THE ERROR AFTER 1$ WILL REPORT THE FAILURE.
                                ;*
                                ;*      ROM FLOW-1,155,312
                                ;;*****************************************************************
004416  005200                  TST32:  INC     R0              ;INCREMENT TEST NUMBER
004420  012705                          MOV     (PC)+,R5        ;PUT BR IN R5
004422  000401                          .WORD   000401          ;BINARY WORD FOR BR .+2
004424  000240                  SYNC32: NOP
004426                          IUT32:
004426  010517                          MOV     R5,(PC)         ;EXECUTE INSTRUCTION UNDER TST
004430  000240                  1$:     .WORD   240             ;SHOULD REPLACE THIS WITH BR 2$
004432  000000                          HALT                    ;LOCATION POINTED TO BY PC DID NOT LOAD
                                                                ;FOR LOOPING CHANGE TO 'BR TST32+2'' (772)
004434  012705  000240          2$:     MOV     #240,R5         ;THIS CODE
004440  012701  004430                  MOV     #1$,R1          ;RESTORES THE NOP
004444  010511                          MOV     R5,(R1)         ;AT LOCATION 1$
004446  000264                          SEZ                     ;ENSURE Z SET
004450  010517                          MOV     R5,(PC)         ;EXECUTE INSTRUCTION UNDER TEST
004452  000000                          .WORD   0
004454  001001                          BNE     TST33           ;;CC OK, GO TO NEXT TEST
004456  000000                          HALT                    ;STATE D12.20 BAD
                                                                ;FOR LOOPING CHANGE TO 'BR TST32+2'' (760)
                                ;;*****************************************************************
                                ;*TEST 33       THREE MICROSTATES (DAC*DM4*O/CLASS)
                                ;*
                                ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
                                ;*      THIS WILL CAUSE A TRAP TO LOCATION 10. THIS WILL ONLY HAPPEN IF
                                ;*      RACE A0 RAB02 IS NOT GOING LOW. EITHER RACE E33 OR
                                ;*      E6(9&8) IS BAD.
                                ;*
                                ;*      IF THE DST CONST FAILS TO SUBTRACT 2, THE ERROR AT EITHER 1$
                                ;*      OR 1$-2 WILL REPORT THE FAILURE.
                                ;*
                                ;*      IF BEN01 FAILS (CAUSED BY IRCD DM357 STUCK HIGH)
                                ;*      EXECUTION WILL GO TO D10.00 WHICH WILL EXECUTE A MODE 5
                                ;*      INSTEAD OF MODE 4.
                                ;*
                                ;*      IF THE DESTINATION IS NOT LOADED PROPERLY,
                                ;*      THE ERROR AT 3$ WILL REPORT THE FAILURE.
                                ;*
                                ;*      ROM FLOW-4,122,157
                                ;;*****************************************************************
004460  005200                  TST33:  INC     R0              ;INCREMENT TEST NUMBER
004462  012702  000001                  MOV     #1,R2           ;SETUP R2
004466  012705  001166                  MOV     #$TMP2,R5       ;PUT ADDRESS OF $TMP2 IN R5
004472  010501                          MOV     R5,R1           ;SAVE R5
```

```
                                ;;*****************************************************************
                                ;THESE THREE INSTRUCTIONS ARE USED TO CATCH IRCD DM357 STUCK HIGH
                                        MOV     #$TMP1,R3       ;PUT ADDR OF $TMP1 IN R3
                                        MOV     R5,(R3)         ;PUT ADDR. OF $TMP2 IN $TMP'
                                ;;*****************************************************************
                                SYNC33: NOP
                                IUT33:
                                        MOV     R2,-(R5)        ;EXECUTE INSTRUCTION UNDER TEST
                                        CMP     R5,R3           ;DID R5 AUTO DECREMENT?
                                        BEQ     4$              ;BRANCH IF YES.
                                        MOV     #$TMP1+2,R5     ;PUT ADDR. IN R5
                                        MOV     #$TMP',R1       ;PUT ADDR IN R1
                                        MOV     (R1),-(R5)      ;EXECUTE INSTRUCTION
                                        CMP     R1,R5           ;DID R5 AUTO DECREMENT?
                                        BEQ     1$              ;BRANCH IF YES
                                        HALT'                   ;IRCC DSTM4 STUCK HIGH
                                                                ;FOR LOOPING CHANGE TO 'BR TST33+2'' (754)
                                '$:
                                        HA '                    ;IRCC DSTM4 OK BUT D45.00 DOES NOT DECREMEN'
                                                                ;FOR LOOPING CHANGE TO 'BR TST33+2'' (753)
                                4$:     MOV     (R5),R5         ;GET CONTENTS OF $TMP1
                                        CMP     R2,R5           ;DID DEST. GET LOADED?
                                        BNE     2$              ;BRANCH IF NO
                                        INC     R5              ;ADJUST R5 TO EVEN ADDRESS
                                        SEZ                     ;ENSURE Z SET
                                        MOV     R2,-(R5)        ;EXECUTE INSTRUCTION UNDER TEST
                                        BNE     TST34           ;;CC'S OK
                                        HALT                    ;STATE D10.40 BAD
                                                                ;FOR LOOPING CHANGE TO 'BR TST33+2'' (743)
                                2$:     MOV     (R1),R1         ;GET CONTENTS OF $TMP2
                                        CMP     R2,R1           ;DID A MODE 5 TAKE PLACE?
                                        BNE     3$              ;BRANCH IF NO
                                        HALT                    ;EITHER IRCD DM357 STUCK HIGH OR RACK E49(C1) BAD
                                                                ;FOR LOOPING CHANGE TO 'BR TST33+2'' (737)
                                3$:
                                        HALT                    ;DST($TMP1) DID NOT GET LOADED FROM SR((R2)
                                                                ;FOR LOOPING CHANGE TO 'BR TST33+2'' (736)
                                ;;*****************************************************************
                                ;*TEST 34        THREE MICROSTATES (DAC*DM1*TST.B*DR0(0))
                                ;*
                                ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
                                ;*      THIS WILL ONLY HAPPEN IF RA SAME?
                                        BNE     2$              ;BRANCH IF NO
                                        MOV     R1,-(R5)        ;SEE IF AUTO DEC. WORKS
                                        CMP     #$TMP1-2,R5     ;DID R5 AUTO DEC?
                                        BNE     3$              ;BRANCH IF NO
                                        HALT                    ;AUTO DEC WORKS SO ROM STATE D12.60 PROBABL Y BAD
                                                                ;FOR LOOPING CHANGE TO 'BR TST33+2'' (730)
                                3$:
                                        HALT                    ;IRCD DSTCON=2 EITHER STUCK LOW OR
                                                                ;NOT GETTING THRU KOMUX
                                                                ;FOR LOOPING CHANGE TO 'BR TST33+2'' (727)
                                2$:
                                        HALT'                   ;BEN15 OK & DST CONST OK BUT INSTR DOSEN'T WORK
                                                                ;FOR LOOPING CHANGE TO 'BR TST33+2'' (726)
                                5$:     MOV     #'$,R5          ;GET ADDR OF INSTR UNDER TCE E44 IS BAD (AFIR53(1)*R/CLA
```

```
2055                                    ;*
2056                                    ;*      IF BEN15 FAILS EXECUTION WILL GO TO STATE D12.60.
2057                                    ;*
2058                                    ;*      IF B FORK FAILS (AFTER STATE D12.10) CAUSED BY IRCB
2059                                    ;*      K/CLASS STUCK LOW EXECUTION WILL GO TO STATE RSD.00 CAUSING A TRAP TO 10.
2060                                    ;*      IF IRCB B1 RAB00 IS STUCK HIGH EXECUTION WILL GO FROM
2061                                    ;*      D12.10 TO JSR.40. THIS WILL CAUSE THE PROCESSOR TO HANG.
2062                                    ;*      IF EITHER GRAB 08D(1) IS STUCK HIGH OR NOT GETTING THRU TO RACL E71,
2063                                    ;*      EXECUTION WILL GO TO STATE D12.30.
2064                                    ;*      IF GRAD DRMX00 IS STUCK HIGH OR GRAB E50 IS BAD, EXECUTION
2065                                    ;*      WILL GO TO D10.60 WHICH WILL  HANG THE PROCESSOR.
2066                                    ;*
2067                                    ;*      ROM FLOW-1,175,33
2068                                    ;;*****************************************************************
2069  004612  005200                    TST34:  INC     R0              ;INCREMENT TEST NUMBER
2070  004614  012702  001164                    MOV     #$TMP1,R2       ;PUT ADDRESS OF $TMP1 IN R2
2071  004620  012705  177400                    MOV     #177400,R5      ;PUT 177400 IN R5
2072  004624  010512                            MOV     R5,(R2)         ;PUT -1 IN $TMP1
2073  004626  000240                    SYNC34: NOP
2074  004630                            IUT34:
2075  004630  005712                            TST     (R2)            ;EXECUTE INSTRUCTION UNDER TEST
2076  004632                            1$:
2077  004632  100416                            BMI     TST35           ;;BRANCH IF INSTRUCTION SET CC'S
2078                                    ;FAILURE-TRY ROM FLOW 1,175,31,132
2079  004634  012737  177777  001164            MOV     #-1,@#$TMP1     ;PUT -1 IN $TMP1
2080  004642  012705  001164                    MOV     #$TMP1,R5       ;PUT ADDR OF $TMP1 IN R5
2081  004646  005215                            INC     (R5)            ;INCREMENT $TMP1
2082  004650  001401                            BEQ     2$              ;BRANCH IF INC WORKED
2083  004652  000000                            HALT                    ;EITHER D12.10 FAILED OR BEN15 FAILED
2084                                                                    ;FOR LOOPING CHANGE TO 'BR TST34+2'' (760)
2085  004654  022737  000000  001164    2$:     CMP     #0,@#$TMP1      ;DID $TMP1 GO TO ZERO?
2086  004662  001401                            BEQ     3$              ;BRANCH IF YES
2087  004664  000000                            HALT                    ;CANNOT DIAGNOSE ERROR
2088                                                                    ;FOR LOOPING CHANGE TO 'BR TST34+2'' (753)
2089  004666                            3$:
2090  004666  000000                            HALT                    ;TST.10 FAILED
2091                                                                    ;FOR LOOPING CHANGE TO 'BR TST34+2'' (752)
2092                                    ;;*****************************************************************
2093                                    ;*TEST 35       THREE MICROSTATES (DAC*DM1*BIT.B*DR0(0))
2094                                    ;*
2095                                    ;*      THIS TEST IS THE SAME AS THE PREVIOUS TEST EXCEPT A BIT INSTRUCTION IS USED.
2096                                    ;*      IF FORK A FAILS RACE BIN*SM0 H FAILED.
2097                                    ;*
2098                                    ;*      IF FORK B FAILS THE INSTRUCTION DECODE ROM WORD IS BAD.
2099                                    ;*
2100                                    ;*      IF THE RESULTANT DATA IS BAD STATE TST.10 FAILED.
2101                                    ;*
2102                                    ;*      ROM FLOW-1,175,33
2103                                    ;;*****************************************************************
2104  004670  005200                    TST35:  INC     R0              ;INCREMENT TEST NUMBER
2105  004672  005005                            CLR     R5              ;CLEAR R5
2106  004674  012701  001164                    MOV     #$TMP1,R1       ;PUT ADDR OF $TMP1 IN R1
2107  004700  010511                            MOV     R5,(R1)         ;CLEAR $TMP1.
2108  004702  012705  100000                    MOV     #BIT15,R5       ;PUT 100000 IN R5
2109  004706  000240                    SYNC35: NOP
2110  004710                            IUT35:
```

```
2111  004710  030511                        BIT     R5,(R1)         ;EXECUTE INSTRUCTION UNDER TEST
2112  004712  001401                        BEQ     TST36           ;;BRANCH IF CC OK
2113  004714  000000                        HALT                    ;STATE TST.10 FAILED
2114                                                                ;FOR LOOPING CHANGE TO 'BR TST35+2'' (766)
2115                                ;;**********************************************************************
2116                                ;*TEST 36       THREE MICROSTATES (DAC*DM1*CMP.B*DR0(0))
2117                                ;*
2118                                ;*      THIS TEST IS THE SAME AS THE PREVIOUS TWO TESTS
2119                                ;*      EXCEPT A CMP INSTRUCTION IS USED.
2120                                ;*
2121                                ;*      ROM FLOW-1,175,33
2122                                ;**********************************************************************
2123  004716  005200                TST36:  INC     R0              ;INCREMENT TEST NUMBER
2124  004720  005005                        CLR     R5              ;CLEAR R5
2125  004722  012701  001164                MOV     #$TMP1,R1       ;PUT ADDR OF $TMP1 IN R1
2126  004726  010511                        MOV     R5,(R1)         ;CLEAR $TMP1
2127  004730                        SYNC36:
2128  004730  000244                        CLZ                     ;ENSURE Z CLEAR
2129  004732                        IUT36:
2130  004732  020511                        CMP     R5,(R1)         ;EXECUTE INSTRUCTION UNDER TEST
2131  004734  001401                        BEQ     TST37           ;;BRANCH IF CC OK
2132  004736  000000                        HALT                    ;STATE TST.10 FAILED
2133                                                                ;FOR LOOPING CHANGE TO 'BR TST36+2'' (770)
2134                                ;;**********************************************************************
2135                                ;*TEST 37       THREE MICROSTATES (DAC*DM2*TST.B*DR0(0))
2136                                ;*
2137                                ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
2138                                ;*      THIS WILL ONLY HAPPEN IF RACE E45 IS BAD (AFIR04(1)*R/CLASS).
2139                                ;*
2140                                ;*      BEN15 & FORK B HAVE ALREADY BEEN TESTED
2141                                ;*      IF THE AUTO INC FAILS IT WILL BE DUE TO A BAD
2142                                ;*      FIELD IN ROM STATE D12.10.
2143                                ;*
2144                                ;*      ROM FLOW-2,175,33
2145                                ;**********************************************************************
2146  004740  005200                TST37:  INC     R0              ;INCREMENT TEST NUMBER
2147  004742  005001                        CLR     R1              ;CLEAR R1
2148  004744  012705  001164                MOV     #$TMP1,R5       ;PUT ADDR. OF $TMP1 IN R5
2149  004750  010115                        MOV     R1,(R5)         ;CLEAR $TMP1
2150  004752  000240                SYNC37: NOP
2151  004754                        IUT37:
2152  004754  005725                        TST     (R5)+           ;EXECUTE INSTR. UNDER TEST
2153  004756  001006                        BNE     1$              ;BRANCH IF BAD CC
2154  004760  022705  001166                CMP     #$TMP1+2,R5     ;DID R5 AUTO INC?
2155  004764  001407                        BEQ     TST40           ;;BRANCH IF TEST OK
2156  004766  010567  174162                MOV     R5,$REG0        ;SAVE REG 0 FOR TYPEOUT
2157  004772  000000                        HALT                    ;NO AUTO INC STATE D12.10 BAD
2158                                                                ;FOR LOOPING CHANGE TO 'BR TST37+2'' (763)
2159  004774  013767  177776  174174  1$:   MOV     @#PSW,$ERPSW    ;SAVE PSW FOR TYPEOUT
2160  005002  000000                        HALT                    ;BAD CC
2161                                                                ;FOR LOOPING CHANGE TO 'BR TST37+2'' (757)
2162
2163
2164                                ;;**********************************************************************
2165                                ;*THE LOGICAL SEQUENCE WOULD NEXT TEST THE BIT.B AND CMP.B INSTRUCTIONS BUT
2166                                ;*THESE WILL NOT BE TESTED WITH INDIVIDUAL TESTS SINCE THEY DO NOT USE
```

```
                                ;*ANY HARDWARE THAT HAS NOT ALREADY BEEN TESTED.
                                ;;*****************************************************************


                                ;;*****************************************************************
                                ;*TEST 40        THREE MICROSTATES (JMP*DM1)
                                ;*
                                ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
                                ;*      THIS WILL ONLY HAPPEN IF RALE A0 RAB00 DOES NOT GO LOW
                                ;*      (AFIR03(1)*[JMP+JSR+SWAB]).
                                ;*
                                ;*      A FORK B FAILURE WOULD BE ONE OF THE FOLLOWING:
                                ;*      IF IRCB (JMP+JSR) IS STUCK LOW EXECUTION WILL GO TO RSD.00 CAUSING
                                ;*      A TRAP TO 10.
                                ;*      IF IRCB IR(14:9) 04 IS STUCK LOW EXECUTION WILL
                                ;*      GO TO JSR.00 WHICH WILL EXECUTE A JSR INSTEAD OF A JMP.
                                ;*      IF IRCB B FORK MUX FAILS EXECUTION WILL GO TO
                                ;*      FOP.00.
                                ;*      IF IRCB FJ CLASS IS STUCK HIGH EXECUTION WILL GO TO
                                ;*      STATE D12.00 AND THE JMP WON'T JUMP.
                                ;*
                                ;*      IF THE INSTRUCTION FAILS TO JUMP, STATE JMP.00 IS REPORTED AS BAD.
                                ;*
                                ;*      ROM FLOW-1,135,35
                                ;;*****************************************************************
005004  005200          TST40:  INC     R0              ;INCREMENT TEST NUMBER
005006  005001                  CLR     R1              ;ENSURE R1 CLEAR
005010  012705  005032          MOV     #JMP1ADR,R5     ;PUT JMP ADDR. IN R5
005014  010502                  MOV     R5,R2           ;PUT JUMP ADDR IN R2
005016  000240          SYNC40: NOP
005020                  IUT40:
005020  000115                  JMP     (R5)            ;EXECUTE INSTRUCTION UNDER TEST
005022  020205                  CMP     R2,R5           ;DID NEGATE OCCUR?
005024  001001                  BNE     2$              ;BRANCH IF YES
005026  000000                  HALT                    ;STATE JMP.00 DID NOT LOAD PCB OR BEN15 FAILED
                                                        ;FOR LOOPING CHANGE TO 'BR TST40+2'' (767)
005030                  2$:
005030  000000                  HALT                    ;IRCB FJ/CLASS NOT GETTING THRU B FORK MUX
                                                        ;FOR LOOPING CHANGE TO 'BR TST40+2'' (766)
005032  005701          JMP1ADR:TST     R1              ;IS R1 STILL ZERO?
005034  001403                  BEQ     TST41           ;;BRANCH IF YES
005036  062706  000002          ADD     #2,SP           ;JSR OCCURRED RE-ADJUST SP
005042  000000                  HALT                    ;RACB IR(14:9)04 STUCK LOW
                                                        ;FOR LOOPING CHANGE TO 'BR TST40+2'' (76
                                ;;*****************************************************************
                                ;*TEST 41        THREE MICROSTATES (JMP*DM2)
                                ;*
                                ;*      THIS TEST IS THE SAME AS THE PREVIOUS TEST EXCEPT THE DM 2.
                                ;*      IF FORK A FAILS RAC E45 IS BAD (AFIR04(1)*[JMP+JSR+SWAB] .
                                ;*
                                ;*      ROM FLOW-2,135,35
                                ;;*****************************************************************
005044  005200          TST41:  INC     R0              ;INCREMENT TEST NUMBER
005046  012705  005056          MOV     #JMP2ADR,R5     ;PUT ADDRESS OF 1$ IN R5
005052  000240          SYNC41: NOP
005054                  IUT41:
```

```
                                        JMP     (R5)+           ;EXECUTE INSTRUCTION UNDER TEST
005060                  JMP2ADR:CMP     #JMP2ADR+2,R5   ;DID R5 AUTO INC?
                                        BEQ     TST42           ;;BRANCH IF YES
                                        HALT                    ;NO AUTO INC
                                                                ;FOR LOOPING CHANGE TO 'BR TST41+2'' (770)
                        ;;**************************************************************
                        ;*TEST 42       THREE MICROSTATES (JMP*DM4)
                        ;*
                        ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
                        ;*      THIS WILL ONLY HAPPEN IF RACE E33(AFIR05(1)*[JMP*JSR*SWAB]) IS BAD.
                        ;*
                        ;*      ALL OTHER LOGIC HAS BEEN TESTED.
                        ;*
                        ;*      ROM FLOW-4,122,35
                        ;;**************************************************************
005200                  TST42:  INC     R0              ;INCREMENT TEST NUMBER
005102                          MOV     #NEXTT,R1       ;PUT ADDRESS OF 1$+2 IN R1
000240                  SYNC42: NOP
                        IUT42:
000141                          JMP     -(R1)           ;EXECUTE INSTRUCTION UNDER TEST
000240                          NOP
000240                  NEXTT:  NOP
                        ;;**************************************************************
                        ;*TEST 43       THREE MICROSTATES (SOB)
                        ;*
                        ;*      IF RACH U/CLASS IS NOT GOING HIGH EXECUTION WILL GO TO RSD.00
                        ;*      CAUSING A TRAP TO LOCATION 10 WITH THE ADDRESS OF 5$-2
                        ;*      ON THE TOP OF THE STACK.
                        ;*      IF EITHER RACE E10 IS BAD OR RACE BIN DOES NOT GO HIGH
                        ;*      EXECUTION WILL GO TO D67.01. EXECUTION WILL EVENTUALL GO
                        ;*      TO RSD.00 AFTER STATE D10.60 AND THE STACKED PC WILL BE AT 5$.
                        ;*      IF RACE E8 FAILS EXECUTION WILL GO TO ASC.10 WHICH WILL
                        ;*      PERFORM AN ASHC*DM0 OPERATION.
                        ;*
                        ;*      IF THE SOB BRANCHES WHEN IT IS NOT SUPPOSE TO EITHER
                        ;*      GRAE SR EQ ONE IS  STUCK HIGH OR RACK E63 IS BAD
                        ;*      OR STATE SOB.10 DOES NOT RESTORE THE OLD PC.
                        ;*
                        ;*      IF THE SOB DOES NOT BRANCH WHEN IT IS SUPPOSE TO EITHER
                        ;*      STATE SOB.00 IS BAD OR GRAE SR EQ ONE STUCK LOW OR
                        ;*      RACK E63(C1) IS BAD.
                        ;*
                        ;*      IF THE REGISTER DOES NOT DECREMENT STATE SOB.20 IS BAD.
                        ;*
                        ;*      ROM FLOW-57,242/262.262
                        ;;**************************************************************
005200                  TST43:  INC     R0              ;INCREMENT TEST NUMBER
001100                          MOV     #STACK,SP       ;INITIALIZE THE SP
007777                          MOV     #7777,R5        ;SETUP R5
000401                          BR      SKIP            ;SKIP NEXT INSTRUCTION
000410                  GOOD:   BR      BAD+2           ;LOCATION FOR SOB TO BRANCH TO
005004                  SKIP:   CLR     R4              ;SETUP R4
000240                  SYNC43: NOP
                        IUT43:
077404                          SOB     R4,GOOD         ;EXECUTE SOB WITH BRANCH
005705                          TST     R5              ;DID R5 CLEAR?
```

```
2279  005132  001001          5$:     BNE     3$              ;BRANCH IF NO
2280  005134  000000                  HALT                    ;RACF E8 IS BAD(BUF AFIR11(1)*C/CLASS)
2281                                                          ;FOR LOOPING CHANGE TO 'BR TST43+2'' (764)
2282  005136                  3$:
2283  005136  000000                  HALT                    ;EITHER GRAE SR EQ ONE IS STUCK LOW
2284                                                          ;OR RACK E63(C1) IS BAD OR SOB.10 IS BAD
2285                                                          ;FOR LOOPING CHANGE TO 'BR TST43+2'' (763)
2286  005140                  BAD:
2287  005140  000000                  HALT                    ;SOB FAILED TO BRANCH. SOB.00 BAD
2288                                                          ;EITHER GRAE SR EQ ONE STUCK HIGH OR
2289                                                          ;RACK E63(C1) BAD OR SOB.00 BAD
2290                                                          ;FOR LOOPING CHANGE TO 'BR TST43+2'' (762)
2291  005142  005005                  CLR     R5              ;SET UP R5 FOR
2292  005144  005205                  INC     R5              ;NO BRANCH CONDITION
2293  005146  000240          SYNC43: NOP
2294  005150                  IUT43:
2295  005150  077505                  SOB     R5,BAD          ;EXECUTE SOB WITHOUT BRANCH
2296  005152  005705                  TST     R5              ;DID R5 DECREMENT?
2297  005154  001401                  BEQ     1$              ;BRANCH IF YES
2298  005156  000000                  HALT                    ;R5 DID NOT DECREMENT. SOB.20 BAD
2299                                                          ;FOR LOOPING CHANGE TO 'BR BAD+2+2'' (772)
2300  005160  005005          1$:     CLR     R5
2301  005162  005001                  CLR     R1
2302  005164  005201          2$:     INC     R1
2303  005166  077502                  SOB     R5,2$           ;CHECK ALL PATTERNS OF R5 FOR SOB
2304  005170  005705                  TST     R5              ;DID R5 GET ALL THE WAY BACK TO 0?
2305  005172  001002                  BNE     3$              ;BRANCH IF NO
2306  005174  005701                  TST     R1              ;DID R1 ROLL OVER?
2307  005176  001401                  BEQ     TST44           ;;BRANCH IF YES
2308  005200                  3$:
2309  005200  000000                  HALT                    ;THE SIGNAL 'SR EQ ONE'' FAILED
2310                                                          ;FOR LOOPING CHANGE TO 'BR 1$'' (767)
2311                                  .SBTTL
2312                          ;;*****************************************************************
2313                          ;*TEST 44       FOUR  MICROSTATES (DAC*DM12*P/CLASS*DR0(0)
2314                          ;*
2315                          ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
2316                          ;*      THIS WILL ONLY HAPPEN IF RACJ AFIR 14(1) DOES NOT
2317                          ;*      GET THRU RAC E42.
2318                          ;*
2319                          ;*      THE FOLLOWING COULD BE FORK B FAILURES:
2320                          ;*      IF IRCB B0 RAB00 IS STUCK HIGH OR NOT GETTING THRU
2321                          ;*      TO RACL RADR00 EXECUTION WILL GO TO EXC.90 WHICH WOULD
2322                          ;*      PUT THE RESULT INTO A REGISTER RATHER THAN MEMORY.
2323                          ;*      IF IRCB E38(6) IS STUCK HIGH EXECUTION WILL GO TO
2324                          ;*      RSD.00 CAUSING A TRAP TO 10.
2325                          ;*      IF THE BIC DOES NOT HAPPEN THEN EXC.00 IS BAD.
2326                          ;*
2327                          ;*      ROM FLOW-2,175,31,132
2328                          ;;*****************************************************************
2329  005202  005200          TST44:  INC     R0              ;INCREMENT TEST NUMBER
2330  005204  012701  001164          MOV     #$TMP1,R1       ;PUT ADDRESS OF $TMP1 IN R1
2331  005210  005005                  CLR     R5              ;PUT A 1
2332  005212  005205                  INC     R5              ;IN R5
2333  005214  010511                  MOV     R5,(R1)         ;PUT A 1 IN $TMP1
2334  005216  000240          SYNC44: NOP
```

```
2335  005220                          IUT44:
2336  005220  040521                          BIC     R5,(R1)+        ;EXECUTE INSTR. UNDER TEST
2337  005222  022701  001166                  CMP     #$TMP1+2,R1     ;DID R1 AUTO INC?
2338  005226  001404                          BEQ     1$              ;BRANCH IF YES
2339  005230  005701                          TST     R1              ;DID INSTR GO THRU EXC.90?
2340  005232  001001                          BNE     2$              ;BRANCH IF NO
2341  005234  000000                          HALT                    ;EITHER IRCB B0 RAB00 IS STUCK HIGH
2342                                                                  ;OR IT IS NOT GETTING THRU TO RACL RADR50
2343                                                                  ;FOR LOOPING CHANGE TO 'BR TST44+2'' (763)
2344  005236                          2$:
2345  005236  000000                          HALT                    ;INSTRUCTION FAILED. CAN'T DETERMINE CAUSE
2346                                                                  ;FOR LOOPING CHANGE TO 'BR TST44+2'' (762)
2347  005240  012701  001164          1$:     MOV     #$TMP1,R1       ;PUT ADDR OF $TMP1 IN R1
2348  005244  005711                          TST     (R1)            ;DID BIC WORK?
2349  005246  001401                          BEQ     3$              ;BRANCH IF YES
2350  005250  000000                          HALT                    ;STATE EXC.00 FAILED
2351                                                                  ;FOR LOOPING CHANGE TO 'BR TST44+2'' (755)
2352  005252  010511                  3$:     MOV     R5,(R1)         ;PUT 1 IN $TMP2 & CLEAR Z
2353  005254  040521                          BIC     R5,(R1)+        ;EXECUTE INSTRUCTION UNDER TEST
2354  005256  001401                          BEQ     TST45           ;;CC'S OK
2355  005260  000000                          HALT                    ;STATE EXC.00 BAD
2356                                                                  ;FOR LOOPING CHANGE TO 'BR TST44+2'' (751)
2357                                  ;;**************************************************************
2358                                  ;*TEST 45       FOUR  MICROSTATES (DAC*DM12*TST.B*DR0(1))
2359                                  ;*
2360                                  ;*      AFTER STATE D12.10 IF EITHER GRAB OBD(1) DOES NOT GO HIGH
2361                                  ;*      OR DOES NOT GET THRU RACL E71 EXECUTION WILL GO
2362                                  ;*      TO TST.10. IF EITHER GRAB OBD(0) IS STUCK HIGH OR NOT GETTING
2363                                  ;*      THRU RACK E41, EXECUTION WILL GO TO D10.60. THIS WILL CAUSE
2364                                  ;*      THE PROCESSOR TO HANG UP IN THE PAUSE STATE AT MICRO
2365                                  ;*      ADDRESS 177.  IF THE TEST FAILS THEN STATE D12.30 FAILED.
2366                                  ;*
2367                                  ;*      ROM FLOW-1,175,137,33
2368                                  ;**************************************************************
2369  005262  005200                  TST45:  INC     R0              ;INCREMENT TEST NUMBER
2370  005264  012705  001164                  MOV     #$TMP1,R5       ;PUT ADDRESS OD $TMP1 IN R5
2371  005270  012701  100000                  MOV     #BIT15,R1       ;PUT NEGATIVE UPPER BYTE IN R1
2372  005274  010115                          MOV     R1,(R5)         ;MOVE R1 TO $TMP1
2373  005276  005205                          INC     R5              ;PUT ODD ADDR IN R5
2374  005300  000240                  SYNC45: NOP
2375  005302                          IUT45:
2376  005302  105715                          TSTB    (R5)            ;EXECUTE INSTR UNDER TEST
2377  005304  100401                          BMI     TST46           ;;TEST OK, GO TO NEXT TEST
2378  005306  000000                          HALT                    ;EITHER STATE D12.30 FAILED OR
2379                                                                  ;BEN05*FEN2 FAILED(SEE ABOVE)
2380                                                                  ;FOR LOOPING CHANGE TO 'BR TST45+2'' (766)
2381                                  ;;**************************************************************
2382                                  ;*TEST 46       FOUR  MICROSTATES (DAC*DM4*TST.B*DR0(0))
2383                                  ;*
2384                                  ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 0.
2385                                  ;*      THIS WILL ONLY OCCUR IF RACE E33(AFIR05(1)*R/CLASS) IS BAD.
2386                                  ;*
2387                                  ;*      AFTER D10.30 IF IRCB FJ/CLASS DOES NOT GET TO RACL E71
2388                                  ;*      OR IF RACL E71 IS BAD EXECUTION WILL GO TO SVC.50.
2389                                  ;*
2390                                  ;*      IF THE INSTRUCTION DOESN'T WORK THEN D10.60 IS BAD.
```

```
2391                                    ;*
2392                                    ;*      ROM FLOW-4,122,177,33
2393                                    ;;*****************************************************************
2394  005310  005200                    TST46:  INC     R0              ;INCREMENT TEST NUMBER
2395  005312  012701  100000C           MOV     #BIT15,R1       ;SET SIGN BIT
2396  005316  012705  001164            MOV     #$TMP1,R5       ;PUT ADDR OF $TMP1 IN R5
2397  005322  010125                    MOV     R1,(R5)+        ;SET SIGN BIT IN $TMP1
2398  005324  005015                    CLR     (R5)            ;ENSURE $TEMP2 CLEAR
2399  005326  000240                    SYNC46: NOP
2400  005330                            IUT46:
2401  005330  005745                    TST     -(R5)           ;EXECUTE INSTRUCTION UNDER TEST
2402  005332  100407                    BMI     TST47           ;;TEST OK, GO TO NEXT TEST
2403  005334  022706  001160            CMP     #$TMP1-4,SP     ;DID EXECUTION GO TO SVC.50?
2404  005340  001003                    BNE     1$              ;BRANCH IF NO
2405  005342  012706  001100            MOV     #STACK,SP       ;RESTORE THE SP
2406  005346  000000                    HALT                    ;BEN15*FEN2 FAILED(SEE ABOVE)
2407                                                            ;FOR LOOPING CHANGE TO 'BR TST46+2'' (761)
2408  005350                            1$:
2409  005350  000000                    HALT                    ;STATE D10.60 FAILED
2410                                                            ;FOR LOOPING CHANGE TO 'BR TST46+2'' (760)
2411
2412
2413                                    ;;*****************************************************************
2414                                    ;*THE LOGICAL SEQUENCE HERE WOULD BE TO TEST THE BIT & CMP INSTRUCTIONS
2415                                    ;*BUT NO ADDITIONAL LOGIC IS TESTED BY THEM.
2416                                    ;;*****************************************************************
2417
2418
2419                                    ;;*****************************************************************
2420                                    ;*TEST 47        FOUR  MICROSTATES (DAC*DM6*O/CLASS)
2421                                    ;*
2422                                    ;*      FORK A SHOULD NOT FAIL ON THIS TEST.
2423                                    ;*
2424                                    ;*      BEN01 SHOULD NOT FAIL.
2425                                    ;*
2426                                    ;*      BEN15*FEN2 SHOULD NOT FAIL.
2427                                    ;*
2428                                    ;*      IF D67.00 FAILS TO INCREMENT THE PC AN RTI INSTRUCTION
2429                                    ;*      WILL BE EXECUTED.
2430                                    ;*      IF D67.00 FAILS TO CLOCK THE BR THE INSTRUCTION
2431                                    ;*      WILL BE ADDED AS THE INDEX WORD.
2432                                    ;*      IF D67.10 FAILS TO ADD THE INDEX NUMBER THE SOURCE
2433                                    ;*      WILL BE PUT IN THE WRONG LOCATION.
2434                                    ;*
2435                                    ;*      ROM FLOW-6,251,122,157
2436                                    ;;*****************************************************************
2437  005352  005200                    TST47:  INC     R0              ;INCREMENT TEST NUMBER
2438  005354  005306                    DEC     SP              ;THIS
2439  005356  005306                    DEC     SP              ;GROUP
2440  005360  012705  000340            MOV     #340,R5         ;OF INSTRUCTIONS
2441  005364  010516                    MOV     R5,(SP)         ;SETS UP THE
2442  005366  005306                    DEC     SP              ;STACK
2443  005370  005306                    DEC     SP              ;TO HANDLE
2444  005372  012705  005432            MOV     #BAD1,R5        ;AN ERONEOUS
2445  005376  010516                    MOV     R5,(SP)         ;RTI INSTRUCTION
2446  005400  012705  001164            MOV     #$TMP1,R5       ;PUT ADDR OF $TMP1 IN R5
```

```
2447  005404  005015                  CLR     (R5)            ;CLEAR $TMP1
2448  005406  012701  100000          MOV     #BIT15,R1       ;SET SIGN BIT IN R1
2449  005412  000240           SYNC47: NOP
2450  005414                    IUT47:
2451  005414  010165  000002          MOV     R1,2(R5)        ;EXECUTE INSTRUCTION UNDER TEST
2452  005420  012706  001100          MOV     #STACK,SP       ;RESTORE THE SP
2453  005424  005715                  TST     (R5)            ;DID INDEX WORD GET ADDED?
2454  005426  100002                  BPL     TST50           ;;BRANCH IF YES
2455  005430  000000                  HALT                    ;D67.10 FAILED TO ADD INDEX
2456                                                          ;FOR LOOPING CHANGE TO 'BR TST47+2'' (751)
2457  005432                    BAD1:
2458  005432  000000                  HALT                    ;D67.00 FAILED TO INC PC
2459                                                          ;FOR LOOPING CHANGE TO 'BR TST47+2'' (750)
2460                              ;;*****************************************************************
2461                              ;*TEST 50       FOUR  MICROSTATES (BIN*SM12*DM0*-DF7*SR0(1))
2462                              ;*
2463                              ;*      IF FORK A FAILS EXECTUTION WILL  GO TO STATE D12.00.
2464                              ;*      THIS WILL HAPPEN IF RACE BF1=7 DOES NOT GO HIGH.
2465                              ;*      STATE D12.00 WOULD BITB R5 & THE CONTENTS OF LOCATION 200
2466                              ;*      WHICH IS 137.
2467                              ;*
2468                              ;*      THE FOLLOWING COULD BE BEN14*FORK C FAILURES:
2469                              ;*      IF IRCC C0 RAB00 DOES NOT GO HIGH EXECUTION WILL GO TO
2470                              ;*      D00.90 WHICH WILL TEST THE LOW BYTE INSTEAD OF THE HIGH BYTE.
2471                              ;*
2472                              ;*      IF STATE D00.80 FAILS TO SWAP THE BYTES IT WILL LOOK LIKE
2473                              ;*      A FORK C FAILURE.
2474                              ;*
2475                              ;*      ROM FLOW-21,27,204,205
2476                              ;;*****************************************************************
2477  005434  005200            TST50:  INC     R0              ;INCREMENT TEST NUMBER
2478  005436  012705  001164          MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
2479  005442  012701  100000          MOV     #BIT15,R1       ;PUT NEG HIGH & POS LOW BYTE IN R1
2480  005446  010115                  MOV     R1,(R5)         ;PUT IN $TMP1
2481  005450  005205                  INC     R5              ;PUT ADDR OF $TMP1 H BYTE IN R5
2482  005452  012701  000200          MOV     #200,R1         ;PUT POS HIGH & NEG LOW BYTE IN R1
2483  005456  000240           SYNC50: NOP
2484  005460                    IUT50:
2485  005460  131501                  BITB    (R5),R1         ;EXECUTE INSTRUCTION UNDER TEST
2486  005462  100405                  BMI     TST51           ;;TEST OK, GO TO NEXT TES;
2487  005464  010215                  MOV     R2,(R5)         ;PUT 200 IN $TMP1
2488  005466  131501                  BITB    (R5),R1         ;EXECUTE FAILED INSTRUCTION
2489  005470  100401                  BMI     1$              ;BRANCH IF FORK C FAILED
2490  005472  000000                  HALT                    ;RACE BF1=7 NOT GOING HIGH
2491                                                          ;EITHER RACJ AFIR14(1) DOES NOT
2492                                                          ;GET TO RACE E40 OR E40 BAD
2493                                                          ;FOR LOOPING CHANGE TO 'BR TST50+2'' (761)
2494  005474                    1$:
2495  005474  000000                  HALT                    ;EITHER IRCC C0 RAB00 DOES NOT GO HIGH
2496                                                          ;OR STATE D00.80 FAILED TO SWAP THE BYTES
2497                                                          ;FOR LOOPING CHANGE TO 'BR TST50+2'' (760)
2498                              ;;*****************************************************************
2499                              ;*TEST 51       FOUR  MICROSTATES (BIN*SM12*DM0*DF7*SR0(0))
2500                              ;*
2501                              ;*      IF FORK A FAILS EXECUTION WILL EITHER GO TO STATE D12.00 OR EXC.80.
2502                              ;*      STATE D12.00 WOULD ADD R5 TO THE CONTENTS OF THE PC.
```

```
                                        ;*      STATE EXC.80 WOULD NOT CHANGE THE PC.
                                        ;*
                                        ;*      IF FORK C FAILS EXECUTION WILL EITHER GO TO JSR.10 OR ASC.80.
                                        ;*      JSR.10 WILL CAUSE R5 TO BE STACKED, THE PC TO BE PUT
                                        ;*      IN R5, AND THE PC REPLACED BY R5 WHICH WILL CAUSE AND RTI SINCE
                                        ;*      THE CONTENTS OF THE LOCATION POINTED TO BY R5 IS 000002.
                                        ;*      ASC.80 WILL CAUSE THE ADD TO LOOK LIKE IT FAILED.
                                        ;*
                                        ;*      IF STATE D07.10 FAILS TO LOAD THE SHFTR THE PC WILL
                                        ;*      DOUBLE AND THE PROGRAM WILL BLOW UP.
                                        ;*
                                        ;*      IF THE SHFTR FAILS TO BE PUT IN THE SR THE PC
                                        ;*      WILL NOT CHANGE.
                                        ;*
                                        ;*      ROM FLOW-21,27,203,30
                                        ;;*****************************************************************
005476  005200          TST51:  INC     R0              ;INCREMENT TEST NUMBER
005500  012706  001100          MOV     #STACK,SP       ;INITIALIZE SP
005504  012701  000340          MOV     #340,R1         ;PUT PRIORITY LEVEL 7 IN R5
005510  010146                  MOV     R1,-(SP)        ;PUT ON STACK
005512  012701  005610          MOV     #$RTI,R1        ;PUT RETURN ADDR IN R1
005516  010146                  MOV     R1,-(SP)        ;PUT ON STACK FOR FORK C FAILURE
005520  005306                  DEC     SP              ;ADJUST THE
005522  005306                  DEC     SP              ;SP
005524  005005                  CLR     R5              ;PUT ADDRESS 0 IN R5
005526  012701  000002          MOV     #2,R1           ;PUT OFFSET IN R1
005532  010115                  MOV     R1,(R5)         ;PUT OFFSET IN LOCATION 0
005534  000240          SYNC51: NOP
005536                  IUT51:
005536  061507                  ADD     (R5),PC         ;EXECUTE INSTRUCTION UNDER TEST
005540  000401                  BR      6$              ;EITHER FORK A OR FORK C OR D07.10 FAILED
005542  000423                  BR      TST52           ;;TEST OK GO TO NEXT TEST
005544  012705  000004  6$:     MOV     #4,R5           ;SET BIT 2 IN R5
005550  061507                  ADD     (R5),PC         ;EXECUTE FAILED INSTR.
005552  000261          3$:     SEC                     ;WILL CHANGE TO SEC!SEZ IF THE INSTR GOES
                                                        ;THRU D12.00.  STATE EXC.8 OR ASC.8
                                                        ;WOULD NOT SET Z WHILE A FAILURE OF
                                                        ;STATE D07.10 WILL CAUSE THE ERROR
                                                        ;1$-2 TO REPORT.
005554  000401                  BR      1$              ;SKIP NEXT INSTRUCTION
005556  000000                  HALT                    ;STATE D07.10 DID NOT LOAD SR
                                                        ;FOR LOOPING CHANGE TO 'BR TST51+2'' (750)
005560  001004          1$:     BNE     2$              ;BRANCH IF Z DID NOT SET
005562  012767  000261  177762  MOV     #261,3$         ;RESTORE ORIGINAL VALUE OF LOCATION 3$
005570  000000                  HALT                    ;RACE BF1=7 DID NOT GO HIGH
                                                        ;FOR LOOPING CHANGE TO 'BR TST51+2'' (743)
005572  012705  100000  2$:     MOV     #BIT15,R5       ;SET SIGN BIT IN R5
005576  000250                  CLN                     ;ENSURE N CLEAR
005600  061507                  ADD     (R5),PC         ;EXECUTE FAILED INSTR
005602  100401                  BMI     4$              ;BRANCH IF ADD OCCURED IN STATE EXC.80
005604  000000                  HALT                    ;EITHER IRCC C0 RAB02 IS BEING HELD LOW
                                                        ; OR IT IS NOT GETTING THRU
                                                        ;TO RACL RADR02
                                                        ;FOR LOOPING CHANGE TO 'BR TST51+2'' (735)
005606                  4$:
005606  000000                  HALT                    ;RACE BIN IS NOT GOING LOW
```

```
                                        ;FOR LOOPING CHANGE TO 'BR TST51+2'' (734)
$RTI:
        HALT                            ;EITHER IRCC C0 RAB01 IS STUCK HIGH
                                        ;OR IRC E40 IS BAD OR IRC E40(14)
                                        ;IS STUCK HIGH
                                        ;OR C0 RAB01 IS NOT GETTING THRU TO RACL RADR01
                                        ;FOR LOOPING CHANGE TO 'BR TST51+2'' (733)
;;*****************************************************************
;*TEST 52       FOUR MICROSTATES (BIN*SM12*DM0*DF7*SR0(1))
;*
;*      IF FORK A FAILS EXECUTION WILL GO TO D12.00.
;*
;*      FORK C SHOULD NOT FAIL SINCE THE LOGIC HAS ALREADY BEEN TESTED.
;*
;*      IF STATE D07.00 FAILS TO SWAP THE BYTES THE CC'S WILL
;*      BE BAD.
;*      IF D07.00 FAILS TO LOAD THE SHIFTER, THE THIRD CMPB WILL FAIL.
;*      IF D07.00 FAILS TO LOAD THE SR THE SECOND CMPB WILL FAIL.
;*
;*      ROM FLOW-21,27,202,30
;;*****************************************************************
TST52:  INC     R0              ;INCREMENT TEST NUMBER
        CLR     R5              ;PUT ADDRESS 0 IN R5
        MOV     #POINT,R1       ;PUT ADDR OF POINT IN R1
        SWAB    R1              ;EXCHANGE BYTES
        MOV     R1,(R5)         ;PUT DATA IN LOCATION 0
        INC     R5              ;CHANGE R5 TO HIGH BYTE ADDRESS
SYNC52: NOP
IUT52:
        CMPB    (R5),PC         ;EXECUTE INSTRUCTION UNDER TEST
POINT:  BEQ     4$              ;BRANCH IF OK
        MOV     2$,R5           ;PUT CONTENTS OF 2$ IN R5
        CMPB    (R5),PC         ;EXECUTE FAILED INSTRUCTION
2$:     BEQ     3$              ;BRANCH IF FORK A FAILED
        HALT                    ;STATE D07.00 FAILED TO SWAB OR LOAD SR
                                ;FOR LOOPING CHANGE TO 'BR TST52+2'' (762)
3$:
        HALT                    ;RACE BF1=0 NOT GOING HIGH
                                ;FOR LOOPING CHANGE TO 'BR TST52+2'' (761)
4$:     CMPB    (R5),PC         ;DID SHFTR GET LOADED
        BNE     TST53           ;;BRANCH IF YES
        HALT                    ;D07.00 DID NOT LOAD SHFTR
                                ;FOR LOOPING CHANGE TO 'BR TST52+2'' (756)

;;*****************************************************************
;*TEST 53       FOUR MICROSTATES (BIN*SM4*DM0*-DF7*SR0(0))
;*
;*      IF FORK A FAILS EXECUTION WILL
;*      GO TO D45.00 WHICH WILL EXECUTE A SM0*DM4 INSTRUCTION EXCEPT
;*      THE DESTINATION REGISTER WILL NOT DECREMENT.
;*
;*      FORK C WILL NOT FAIL SINCE IT HAS ALREADY BEEN TESTED.
;*
;*      IF THE SRC FAILS TO AUTO DECREMENT STATE S45.00 IS BAD.
;*
;*      ROM FLOW-24,23,27,205
```

```
2615
2616  005660  005200                  ;;*****************************************************************
2617  005662  012705  001164  TST53:  INC     R0              ;INCREMENT TEST NUMBER
2618  005666  012701  100000          MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
2619  005672  010125                  MOV     #BIT15,R1       ;SET SIGN BIT IN R1
2620  005674  005015                  MOV     R1,(R5)+        ;SET SIGN BIT $TMP1 & STEP R5 TO $TMP2
2621  005676  012701  000002          CLR     (R5)            ;CLEAR $TMP2
2622  005702  005011                  MOV     #2,R1           ;PUT ADDRESS OF LOC 2 IN R1
2623  005704  000240          SYNC53: CLR     (R1)            ;CLEAR LOCATION ZERO
2624  005706                  IUT53:  NOP
2625  005706  054501                  BIS     -(R5),R1        ;EXECUTE INSTRUCTION UNDER TEST
2626  005710  100411                  BMI     TST54           ;;TEST OK, GO TO NEXT TEST
2627  005712  020537  000002          CMP     R5,@#2          ;DID FORK A FAIL?
2628  005716  001001                  BNE     1$              ;BRANCH IF NO
2629  005720  000000                  HALT                    ;EITHER RACE BF1=7 OR SM0 DID NOT GO HIGH
2630                                                          ;FOR LOOPING CHANGE TO 'BR TST53+2'' (760)
2631  005722  020527  001166  1$:     CMP     R5,#$TMP2       ;DID R5 FAIL TO DECREMENT?
2632  005726  001001                  BNE     2$              ;BRANCH IF NO
2633  005730  000000                  HALT                    ;STATE S45.00 DID NOT DECREMENT R5
2634                                                          ;FOR LOOPING CHANGE TO 'BR TST53+2'' (754)
2635  005732                  2$:
2636  005732  000000                  HALT                    ;INSTRUCTION FAILED
2637                                                          ;FOR LOOPING CHANGE TO 'BR TST53+2'' (753)
2638                          ;;*****************************************************************
2639                          ;*TEST 54        FOUR  MICROSTATES (RTS)
2640                          ;*
2641                          ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
2642                          ;*      THIS WILL ONLY HAPPEN IF RACF E3 DOES NOT GO HIGH.
2643                          ;*
2644                          ;*      IF THE PC OR R5 FAILS THF TEST WILL HALT.
2645                          ;*
2646                          ;*      ROM FLOW-40,223,224,342
2647                          ;;*****************************************************************
2648  005734  005200  TST54:  INC     R0              ;INCREMENT TEST NUMBER
2649  005736  012706  001100          MOV     #STACK,SP       ;INITIALIZE THE STACK
```

```
2650  005742  012705  100000          MOV     #BIT15,R5       ;SET SIGN BIT IN R5
2651  005746  010546                  MOV     R5,-(SP)        ;PUT R5 ON THE STACK
2652  005750  012705  005760          MOV     #1$,R5          ;PUT ADDRESS TO RETURN TO IN R5
2653  005754  000205                  RTS     R5              ;EXECUTE INSTRUCTION UNDER TEST
2654  005756  000000                  HALT                    ;STATE RTS.00 FAILED TO PUT R5 IN THE PC
2655                                                          ;FOR LOOPING CHANGE TO 'BR TST54+2'' (767)
2656  005760  005705          1$:     TST     R5              ;DID R5 GET THE TOP OF THE STACK?
2657  005762  100401                  BMI     TST55           ;;BRANCH IF YES
2658  005764  000000                  HALT                    ;THE RTS FAILED TO PUT THE STACK IN R5
2659                                                          ;FOR LOOPING CHANGE TO 'BR TST54+2'' (764)
2660                                ;;**********************************************************************
2661                                ;*TEST 55       FOUR  MICROSTATES (JMP*DM6)
2662                                ;*
2663                                ;*      IF FORK A FAILS EXECUTION WILL GO TO STATE D12.01.
2664                                ;*      THIS WOULD CAUSE A JMP*DM1 TO EXECUTE.
2665                                ;*
2666                                ;*      NEITHER BEN01 NOR BEN15*FEN2 SHOULD FAIL SINCE THEY HAVE ALREADY BEEN
2667                                ;*      TESTED.
2668                                ;*
2669                                ;*      ROM FLOW-6,251,122,35
2670                                ;;**********************************************************************
2671  005766  005200          TST55:  INC     R0              ;INCREMENT TEST NUMBER
2672  005770  012705  006002          MOV     #JMPTO,R5       ;PUT ADDRESS-2 TO JUMP TO IN R5
2673  005774  000240          SYNC55: NOP
2674  005776                  IUT55:
2675  005776  000165  000002          JMP     2(R5)           ;EXECUTE INSTRUCTION UNDER TEST
2676  006002                  JMPTO:
2677  006002  000000                  HALT                    ;JMP*DM6 DID NOT JUMP OR THE OFFSET
2678                                                          ;DID NOT GET ADDED OR RACE E33 FAILED
2679                                                          ;AND A JMP*DM1 WAS EXECUTED
2680                                                          ;FOR LOOPING CHANGE TO 'BR TST55+2'' (772)
2681                                      .SBTTL
2682                                ;;**********************************************************************
2683                                ;*TEST 56       FIVE  MICROSTATES (DAC*DM12*P/CLASS*DR0(1))
2684                                ;*
2685                                ;*      FORK A SHOULDN'T FAIL.
2686                                ;*
2687                                ;*      BEN15 SHOULDN'T FAIL SINCE THIS LOGIC HAS BEEN TESTED.
2688                                ;*      NEITHER SHOULD BEN05*FEN2.
2689                                ;*
2690                                ;*      IF FORK B FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
2691                                ;*      THIS FAILURE WOULD BE CAUSED BY A BAD FIELD (PART PCLASS)
2692                                ;*      IN THE IR DECODE ROM.
2693                                ;*
2694                                ;*      ROM FLOW-1,175,137,31,132
2695                                ;;**********************************************************************
2696  006004  005200          TST56:  INC     R0              ;INCREMENT TEST NUMBER
2697  006006  012705  001164          MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
2698  006012  012701  040000          MOV     #BIT14,R1       ;SET BIT 14 IN R1
2699  006016  010115                  MOV     R1,(R5)         ;SET BIT 14 IN $TMP1
2700  006020  005205                  INC     R5              ;SET R5 TO HIGH BYTE OF $TMP1
2701  006022  000240          SYNC56: NOP
2702  006024                  IUT56:
2703  006024  106115                  ROLB    (R5)            ;EXECUTE INSTRUCTION UNDER TEST
2704  006026  100401                  BMI     TST57           ;;TEST OK, GO TO NEXT TEST
2705  006030  000000                  HALT                    ;ROLB*DM1*DR5(1)) FAILED (BAD C(
```

```
                                        ;FOR LOOPING CHANGE TO 'BR TST56+2'' (766)
                        ;;*****************************************************************
                        ;*TEST 57       FIVE  MICROSTATES (DAC*DM3*0/CLASS)
                        ;*
                        ;*      FORK A SHOULD NOT FAIL SINCE THE LOGIC HAS ALREADY BEEN TESTED.
                        ;*
                        ;*      IF THE DR DOES NOT AUTO INC THEN STATE D30.10 IS BAD.
                        ;*
                        ;*      IF THE CONDITION CODES ARE BAD THEN EITHER STATE D10.50
                        ;*      DID NOT LOAD THE BR OR THE DOUBLE DEFERED DIDN'T WORK.
                        ;*
                        ;*      ROM FLOW-3,221,233,311,157
                        ;;*****************************************************************
006032  005200          TST57:  INC     R0              ;INCREMENT TEST NUMBER
006034  012705  001164          MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
006040  012701  001166          MOV     #$TMP2,R1       ;PUT ADDRESS OF $TMP2 IN R1
006044  010115                  MOV     R1,(R5)         ;PUT ADDRESS OF $TMP2 IN $TMP1
006046  005003                  CLR     R3
006050  010311                  MOV     R3,(R1)         ;CLEAR $TMP2
006052  012703  100000          MOV     #BIT15,R3       ;SET SIGN BIT IN R3
006056  000240          SYNC57: NOP
006060                  IUT57:
006060  010335                  MOV     R3,@(R5)+       ;EXECUTE INSTRUCTION UNDER TEST
006062  100401                  BMI     1$              ;BRANCH IF N BIT SET
006064  000000                  HALT                    ;BAD CONDITION CODES
                                                        ;FOR LOOPING CHANGE TO 'BR TST57+2'' (763)
006066  005711          1$:     TST     (R1)            ;DID MOVE ACTUALLY TAKE PLACE?
006070  100401                  BMI     2$              ;BRANCH IF YES
006072  000000                  HALT                    ;SOURCE DID NOT GET MOVED TO DESTINATION
                                                        ;FOR LOOPING CHANGE TO 'BR TST57+2'' (760)
006074  020501          2$:     CMP     R5,R1           ;DID R5 AUTO INC?
006076  001401                  BEQ     TST60           ;;BRANCH IF YES
006100  000000                  HALT                    ;STATE D30.10 DID NOT INC R5
                                                        ;FOR LOOPING CHANGE TO 'BR TST57+2'' (755)
                        ;;*****************************************************************
                        ;*TEST 60       FIVE  MICROSTATES (DAC*DM4*P/CLASS*DR0(0))
                        ;*
                        ;*      FORK A SHOULD NOT FAIL.
                        ;*
                        ;*      IF FORK B FAILS, AFTER D10.60, EXECUTION WILL GO TO
                        ;*      RSD.00 CAUSING A TRAP TO 10. THIS WILL ONLY HAPPEN IF
                        ;*      THE IR DECODE ROM HAS A BAD FIELD (PART PCLASS).
                        ;*
                        ;*      ROM FLOW-4,122,177,31,132
                        ;;*****************************************************************
006102  005200          TST60:  INC     R0              ;INCREMENT TEST NUMBER
006104  012705  001164          MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
006110  005025                  CLR     (R5)+           ;CLEAR $TMP1 & STEP R5 TO $TMP2
006112  000270                  SEN                     ;SET N
006114  000240          SYNC60: NOP
006116                  IUT60:
006116  006745                  SXT     -(R5)           ;EXECUTE INSTRUCTION UNDER TEST.
006120  005215                  INC     (R5)            ;SHOULD MAKE $TMP1=0
006122  001401                  BEQ     TST61           ;;BRANCH IF TEST OK
006124  000000                  HALT                    ;SXT DID NOT WORK
                                                        ;FOR LOOPING CHANGE TO 'BR TST60+2''
```

```
2762
2763
2764                                    ;;*****************************************************************
2765                                    ;*THE LOGICAL SEQUENCE AT THIS POINT WOULD BE TO EXECUTE A DAC*DM4*[TST.B*
2766                                    ;*BIT.B*CMP.B]*DR0(1) INSTRUCTION FOLLOWED BY A DAC*DM6*[TST.B*BIT.B*CMP.B]*
2767                                    ;*DR0(0) INSTRUCTION TEST BUT NO ADDITIONAL LOGIC IS TESTED.
2768                                    ;;*****************************************************************
2769
2770
2771
2772
2773                                    ;;*****************************************************************
2774                                    ;*THE LOGICAL SEQUENCE AT THIS POINT WOULD BE TO EXECUTE A BIN*SM4*DM0*-DF7*SR0(1)
2775                                    ;*FOLLOWED BY A BIN*SM4*DM0*DF7*SR0(0)
2776                                    ;*FOLLOWED BY A BIN*SM4*DM0*DF7*SR0(1) INSTRUCTION BUT NO ADDITIONAL LOGIC IS TESTED.
2777                                    ;;*****************************************************************
2778
2779
2780                                    ;;*****************************************************************
2781                                    ;*TEST 61       FIVE  MICROSTATES (BIN*SM6*DM0*-DF7*SR0(0))
2782                                    ;*
2783                                    ;*      IF FORK A FAILS EXECUTION WILL GO TO STATE D67.00
2784                                    ;*      WHICH WOULD EXECUTE A SM0*DM6 INSTRUCTION.
2785                                    ;*
2786                                    ;*      BEN14*FEN4 SHOULD NOT FAIL SINCE IT HAS ALREADY BEEN TESTED
2787                                    ;*
2788                                    ;*      ROM FLOW-26,54,141,142,205
2789                                    ;;*****************************************************************
2790  006126  005200                   TST61:  INC     R0              ;INCREMENT TEST NUMBER
2791  006130  012705  001164                   MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
2792  006134  010565  000002                   MOV     R5,2(R5)        ;PUT ADDRESS OF $TMP1 IN $TMP2
2793  006140  010501                           MOV     R5,R1           ;PUT ADDRESS OF $TMP1 IN R1
2794  006142  000240                   SYNC61: NOP
2795  006144                            IUT61:
2796  006144  046501  000002                   BIC     2(R5),R1        ;EXECUTE INSTRUCTION UNDER TEST
2797  006150  005701                           TST     R1              ;DID R1 GO TO ZERO?
2798  006152  001405                           BEQ     TST62           ;;BRANCH IF YES
2799  006154  005767  173006                   TST     $TMP2           ;DID $TMP2 GO TO ZERO?
2800  006160  001001                           BNE     1$              ;BRANCH IF NO
2801  006162  000000                           HALT                    ;RACE BF1=0 DID NOT GO HIGH ON BIC
2802                                                                    ;FOR LOOPING CHANGE TO 'BR TST61+2'' (762)
2803  006164                            1$:
2804  006164  000000                           HALT                    ;INSTRUCTION FAILED
2805                                                                    ;FOR LOOPING CHANGE TO 'BR TST61+2'' (761)
2806                                    ;;*****************************************************************
2807                                    ;*TEST 62       FIVE  MICROSTATES (BIN*SM12*DM12*0/CLASS)
2808                                    ;*
2809                                    ;*      IF FORK A FAILS EXECUTION WILL GO TO D12.01.THIS WOULD CAUSE R5
2810                                    ;*      TO BE WRITTEN INTO $TMP2.
2811                                    ;*
2812                                    ;*      IF FORK C FAILS EXECUTION WOULD GO TO ONE OF THE
2813                                    ;*      FOLLOWING STATES: D45.80,D30.80,FOP.00,WAT.00, D00.90, AND ASH.20.
2814                                    ;*      STATE D45.80 WOULD EXECUTE A SM2*DM4 INSTEAD OF SM2*DM1.
2815                                    ;*      STATE D30.80 WOULD EXECUTE A SM1*DM3.
2816                                    ;*      STATE FOP.00 WOULD CAUSE A TRAP TO LOCATION 10.
2817                                    ;*      THIS WILL ONLY HAPPEN IF EITHER IRCC CO RAB03 IS STUCK
```

```
                                        ;*      OR IT IS NOT GETTING THRU RACL RADR03 OR
                                        ;*      IRCC FORK C MUX INPUT B0 IS HIGH.
                                        ;*      THE LA30 PRINTER BUFFER WILL BE SET UP TO GENERATE AN INTERRUPT IN
                                        ;*      CASE THE TEST FAILS TO THE WAT.00 STATE.
                                        ;*      STATE D00.90 WILL EXECUTE A DM0 INSTEAD OF A DM1.
                                        ;*      STATE ASH.20 WILL CLEAR THE C BIT.
                                        ;*
                                        ;*      ROM FLOW-22,27,111,155,312
                                        ;*************************************************************
006166  005200                  TST62:  INC     R0                      ;INCREMENT TEST NUMBER
006170  016767  172746  000130          MOV     $TPS,3$                 ;SETUP ADDRESSES OF TP
006176  016767  172740  000134          MOV     $TPS,5$                 ;STATUS AND TP BUFFER
006204  016767  172732  000140          MOV     $TPS,POINT1+2           ;INCASE THAY ARE NOT
006212  016767  172724  000202          MOV     $TPS,$$TPS
006220  016767  172720  000074          MOV     $TPB,2$                 ;STANDARD ADDRESSES
006226  016767  172712  000100          MOV     $TPB,4$
006234  012706  001100                  MOV     #STACK,SP               ;INITIALIZE THE SP
006240  012705  001164                  MOV     #$TMP1,R5               ;PUT ADDRESS OF $TMP1 IN R5
006244  012701  001166                  MOV     #$TMP2,R1               ;PUT ADDRESS OF $TMP2 IN R1
006250  012702  100000                  MOV     #BIT15,R2               ;SET SIGN BIT IN R2
006254  010215                          MOV     R2,(R5)                 ;SET SIGN BIT IN $TMP1
006256  005011                          CLR     (R1)                    ;CLEAR $TMP2
006260  005002                          CLR     R2                      ;PUT ADDRESS OF LOCATION 0 IN R2
006262  005012                          CLR     (R2)                    ;CLEAR LOCATION ZERO
006264  012702  000064                  MOV     #64,R2                  ;PUT ADDRESS OF PRINTER VECTOR IN R2
006270  012704  006350                  MOV     #POINT1,R4              ;PUT ADDRESS OF POINT1 IN R4
006274  010412                          MOV     R4,(R2)                 ;PUT ADDRESS OF POINT1 IN PRINTER VECTOR
006276  012702  177776                  MOV     #PSW,R2                 ;PUT ADDRESS OF PSW IN R2
006302  005767  172572                  TST     $PASS                   ;IS THIS PASS 1?
006306  001015                          BNE     SYNC62                  ;BRANCH IF NO
006310  012703  000101                  MOV     #101,R3                 ;PUT ASCII FOR "A" IN R3
006314  012704  000100                  MOV     #BIT06,R4               ;PUT INTERRUPT ENABLE BIT IN R4
006320  010337                          MOV     R3,@(PC)+               ;SEND A TO PRINTER
006322  177566          2$:     .WORD   177566                  ;ADDRESS OF TP BUFFER
006324  105737          1$:     TSTB    @(PC)+                  ;WAIT FOR PRINTER DONE
                                                                ;INCASE DOUBLE BUFFERED
006326  177564          3$:     .WORD   177564                  ;ADDRESS OF TP STATUS
006330  100375                  BPL     1$
006332  010337                  MOV     R3,@(PC)+               ;SEND SECOND A
006334  177566          4$:     .WORD   177566
006336  010437                  MOV     R4,@(PC)+               ;SET THE INTERRUPT FLSG IN TPS
006340  177564          5$:     .WORD   177564
006342                  SYNC62:
006342  000261                  SEC                             ;SET C TO CATCH FAILURE TO ASH.20
006344                  IUT62:
006344  012511                  MOV     (R5)+,(R1)              ;EXECUTE INSTRUCTION UNDER TEST
006346  011202                  MOV     (R2),R2                 ;SAVE PSW IN R2
006350  040437          POINT1: BIC     R4,@(PC)+               ;CLEAR THE INTERR FLAG
006352  177564                  .WORD   177564
006354  005701                  TST     R1                      ;DID A DM0 GET EXECUTED?
006356  100001                  BPL     3$                      ;BRANCH IF NO
006360  000000                  HALT                            ;IRCC DM0 L STUCK LOW
                                                                ;FOR LOOPING CHANGE TO "BR TST62+2" (703)
006362  005711          3$:     TST     (R1)                    ;DID A SM2*DM4 GET EXECUTED?
006364  100401                  BMI     5$                      ;BRANCH IF NO
006366  000000                  HALT                            ;IRCC C FORK MUX INPUT B1 IS STUCK LOW
```

```
                                                ;FOR LOOPING CHANGE TO 'BR TST62+2'' (700)
006370  011104          5$:     MOV     (R1),R4 ;GET CONTENTS OF $TMP2
006372  020504                  CMP     R5,R4   ;DID D12.01 GET EXECUTED?
006374  001001                  BNE     6$      ;BRANCH IF NO
006376  000000                  HALT            ;RACE E29 IS BAD
                                                ;FOR LOOPING CHANGE TO 'BR TST62+2'' (674)
006400  022706  001074    6$:     CMP     #1074,SP ;DID INSTRUCTION CAUSE WAIT TO OCCUR?
006404  001001                  BNE     2$      ;BRANCH IF NO
006406  000000                  HALT            ;EITHER IRCC DSTMO H IS STUCK LOW OR
                                                ;IT IS NOT GETTING THRU RACL RADR05
                                                ;FOR LOOPING CHANGE TO 'BR TST62+2'' (670)
006410  005004          2$:     CLR     R4      ;PUT ADDR. OF LOCATION ZERO IN R4
006412  005714                  TST     (R4)    ;DID A SM1*DM3 GET EXECUTED?
006414  100001                  BPL     IT      ;BRANCH IF NO
006416  000000                  HALT            ;IRCC C FORK MUX INPUT B2 IS STUCK LOW
                                                ;FOR LOOPING CHANGE TO 'BR TST62+2'' (664)
006420  105737          IT:     TSTB    @(PC)+  ;IS PRINTER DONE?
006422  177564          $$TPS:  .WORD   177564
006424  100375                  BPL     IT      ;BRANCH IF NO
006426  032702  000001          BIT     #BIT0,R2 ;DID INSTRUCTION LEAVE C BIT SET?
006432  001001                  BNE     TST63   ;;BRANCH IF YES
006434  000000                  HALT            ;IRCC C FORK MUX SELECT NOT GOING HIGH(ON CHIP)
                                                ;FOR LOOPING CHANGE TO 'BR TST62+2'' (655)


                        ;;**************************************************************
                        ;*THE LOGICAL FLOW AT THIS POINT WOULD TEST A BIN*SM12*DM12*SR0(0)*DR0(0)
                        ;**[TST.B+BIT.B+CMP.B] INSTRUCTION BUT NO ADDITIONAL LOGIC WOULD BE TESTED.
                        ;;**************************************************************


                        ;;**************************************************************
                        ;*TEST 63       FIVE  MICROSTATES (BIN*SM12*DM12*SR0(1)*DR0(0)*CMPB)
                        ;*
                        ;*      THE ONLY THING THAT SHOULD FAIL WOULD BE STATE D12.90(110).
                        ;*
                        ;*      ROM FLOW-21,27,110,175,33
                        ;**************************************************************
006436  005200  TST63:  INC     R0              ;INCREMENT TEST NUMBER
006440  012705  001164          MOV     #$TMP1,R5 ;PUT ADDRESS OF $TMP1 IN R5
006444  112715  000377          MOVB    #377,(R5) ;SET LOW BYTE OF $TMP1 TO ALL ONE'S
006450  012701  001166          MOV     #$TMP2,R1 ;PUT ADDRESS OF $TMP2 IN R1
006454  012711  177400          MOV     #177400,(R1) ;SET HIGH BYTE OF $TMP2 TO ALL ONES
006460  005201                  INC     R1      ;ADJUST R1 TO $TEMP2 HIGH BYTE
006462  000240  SYNC63: NOP
006464  IUT63:
006464  121115          CMPB    (R1),(R5)       ;EXECUTE INSTRUCTION UNDER TEST
006466  001401          BEQ     TST64           ;;TEST OK, GO TO NEXT TEST
006470  000000          HALT                    ;STATE D12.90 FAILED
                                                ;FOR LOOPING CHANGE TO 'BR TST63+2'' (663)
                        ;;**************************************************************
                        ;*TEST 64       FIVE  MICROSTATES (BIN*SM12*DM4*O/CLASS)
                        ;*
                        ;*      IF FORK C FAILS EXECUTION WILL GO TO D12.80
                        ;*      WHICH WILL EXECUTE A SM1*DM2 TYPE INSTRUCTION.
                        ;*
```

```
                                ;*      IF THE INSTRUCTION FAILS EITHER D45.80 OR D40.20 FAILED.
                                ;*
                                ;*      ROM FLOW-21,27,115,121,157
                                ;;*****************************************************************
006472  005200                T64:    INC     R0              ;INCREMENT TEST NUMBER
006474  012705  001164                MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
006500  010501                        MOV     R5,R1           ;SAVE ADDRESS OF $TMP1
006502  005025                        CLR     (R5)+           ;CLEAR $TMP1 AND STEP R5 TO $TMP2
006504  012715  100000                MOV     #BIT15,(R5)     ;SET SIGN BIT IN $TMP2
006510  000240                SYNC64: NOP
006512                          IUT64:
006512  011545                        MOV     (R5),-(R5)      ;EXECUTE INSTRUCTION UNDER TEST
006514  022705  001170                CMP     #$TMP2+2,R5     ;DID R5 AUTO INCREMENT?
006520  001001                        BNE     1$              ;BRANCH IF YES
006522  000000                        HALT                    ;IRCC FORK C MUX INPUT B1 STUCK HIGH
                                                                ;FOR LOOPING CHANGE TO 'BR TST64+2'' (764)
006524  005711                1$:     TST     (R1)            ;DID INSTRUCTION WORK?
006526  100401                        BMI     TST65           ;;BRANCH IF YES
006530  000000                        HALT                    ;EITHER STATE D45.80 OR D40.20 FAILED
                                                                ;FOR LOOPING CHANGE TO 'BR TST64+2'' (76')
                                        .SBTTL
                                ;;*****************************************************************
                                ;*TEST 65        SIX   MICROSTATES (DAC*DM12*ASRB*DR0(1))
                                ;*
                                ;*      NEITHER FORK A NOR BEN15 NOR BEN05*FEN2 SHOULD FAIL
                                ;*      SINCE THEY HAVE ALREADY BEEN TESTED.
                                ;*
                                ;*      IF FORK B FAILS AFTER D12.30 EXECUTION WILL GO TO
                                ;*      ONE OF THE FOLLOWING: RSD.00,D45.00,EXC.00,S45.00,
                                ;*      CCP.00,MUL.00,SVC.10,MFP.00 OR DEP.00.
                                ;*      RSD.00 WILL CAUSE A TRAP TO LOCATION 10.
                                ;*      THIS WILL HAPPEN IF THE B FORK MUX SELECT IS STUCK LOW.
                                ;*      IF STATE D45.00 IS ENTERED THE PROCESSOR WILL HANG UP
                                ;*      IN A LOOP BETWEEN STATES D45.00 AND D10.30.
                                ;*      IF STATE S45.00 IS ENTERED EXECUTION WILL GO TO STATE
                                ;*      D12.80 AFTER S13.10 WHICH WILL HANG UP THE PROCESSOR.
                                ;*      STATE CCP.00 WOULD SET OR CLEAR THE CONDITION CODES
                                ;*      ACCORDING TO IR(4:0).
                                ;*      STATE MUL.00 WOULD CAUSE THE DESTINATION OPERAND TO
                                ;*      BE MULTIPLIED BY REGISTER 2 AND THE RESULT WOULD BE
                                ;*      STORED IN REGISTER 2 AND 3.
                                ;*      IF SVC.10 IS ENTERED A TRAP TO 4 WILL OCCUR BECAUSE
                                ;*      THE DESTINATION REGISTER IS ODD. THIS WILL ONLY HAPPEN
                                ;*      IF EITHER B FORK MUX INPUT B3 OR IRCB B0 RAB00 IS STUCK LOW.
                                ;*      IF MFP.00 IS ENTERED AN MFPI INSTRUCTION WILL BE EXECUTED
                                ;*      THIS WILL PUSH THE ADDRESS OF 1$ ONTO THE STACK.
                                ;*      DEP.00 WILL CAUSE THE PROCESSOR TO HANG IN MICRO ADDRESS
                                ;*      170 WITH THE RUN LIGHT ON.
                                ;*
                                ;*      ROM FLOW-1,175,137,64,123,132
                                ;;*****************************************************************
006532  005200                TST65:  INC     R0              ;INCREMENT TEST NUMBER
006534  012706  001074                MOV     #1074,SP        ;INITIALIZE THE STACK
006540  012705  001164                MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
006544  012701  006572                MOV     #AFTER,R1       ;PUT ADDRESS OF AFTER IN R1
006550  010115                        MOV     R1,(R5)         ;STORE IN $TMP1
```

```
2986  006552  005205                  INC     R5              ;SET R5 TO HIGH BYTE OF $TMP1
2987  006554  005002                  CLR     R2              ;ENSURE R2 CLEAR
2988  006556  012703  000001          MOV     #1,R3           ;ENSURE R3 NOT CLEAR
2989  006562  012701  177776          MOV     #PSW,R1         ;PUT ADDRESS OF PSW IN R1
2990  006566  000264                  SEZ                     ;ENSURE Z BIT SET
2991  006570                  SYNC65:
2992  006570                  IUT65:
2993  006570  106215                  ASRB    (R5)            ;EXECUTE INSTRUCTION UNDER TEST
2994  006572  011102          AFTER:  MOV     (R1),R2         ;SAVE PSW
2995  006574  010567  172354          MOV     R5,$REG0        ;SAVE R5
2996  006600  005703                  TST     R3              ;DID MULTIPLY OCCUR & CLEAR R3?
2997  006602  001001                  BNE     3$              ;BRANCH IF NO
2998  006604  000000                  HALT                    ;B FORK MUX INPUT B1 STUCK HIGH
2999                                                          ;FOR LOOPING CHANGE TO 'BR TST65+2'' (753)
3000  006606  012701  006572  3$:     MOV     #AFTER,R1       ;PUT ADDRESS OF 1$ IN R1
3001  006612  000301                  SWAB    R1              ;REVERSE BYTES
3002  006614  106001                  RORB    R1              ;MAKE IT LOOK LIKE EXC.00 WAS ENTERED
3003  006616  012703  001164          MOV     #$TMP1,R3       ;PUT ADDRESS OF $TMP1 IN R3
3004  006622  020113                  CMP     R1,(R3)         ;DID EXC.00 GET ENTERED?
3005  006624  001001                  BNE     4$              ;BRANCH IF NO
3006  006626  000000                  HALT                    ;IRCB 0BD(ASRB OR RORB) STUCK HIGH
3007                                                          ;FOR LOOPING CHANGE TO 'BR TST65+2'' (742)
3008  006630  022716  006572  4$:     CMP     #AFTER,(SP)     ;DID MFP.00 EXECUTE?
3009  006634  001001                  BNE     5$              ;BRANCH IF NO
3010  006636  000000                  HALT                    ;B FORK MUX INPUT B2 STUCK LOW
3011                                                          ;FOR LOOPING CHANGE TO 'BR TST65+2'' (736)
3012  006640  032702  000004  5$:     BIT     #4,R2           ;DID CCP.00 EXECUTE?
3013  006644  001401                  BEQ     6$              ;BRANCH IF NO
3014  006646  000000                  HALT                    ;IRCC B0 RAB04 NOT GETTING THRU RACL RADR54
3015                                                          ;FOR LOOPING CHANGE TO 'BR TST65+2'' (732)
3016  006650  012701  006572  6$:     MOV     #AFTER,R1       ;GET ADDRESS OF AFTER
3017  006654  010105                  MOV     R1,R5           ;SAVE R1
3018  006656  006001                  ROR     R1              ;RIGHT SHIFT R1 WITHOUT USING ASR
3019  006660  042701  000377          BIC     #377,R1         ;CLEAR LOWER BYTE OF R1
3020  006664  042705  177400          BIC     #177400,R5      ;CLEAR UPPER BYTE OF R5
3021  006670  050501                  BIS     R5,R1           ;MAKE LOWER BYTE OF R10W
3022                                                          ;BYTE OF DST. OPERAND
3023  006672  020167  172266          CMP     R1,$TMP1        ;DID DESTINATION GET ASR'D PROPERLY?
3024  006676  001401                  BEQ     TST66           ;;BRANCH IF YES
3025  006700  000000                  HALT                    ;EITHER SHR.00 OR SHR.10 FAILED
3026                                                          ;FOR LOOPING CHANGE TO 'BR TST65+2'' (715)
3027                                  ;;*****************************************************************
3028                                  ;*TEST 66        SIX   MICROSTATES (DAC*DM12*RORB*DR0(1))
3029                                  ;*
3030                                  ;*      THIS TEST IS THE SAME AS THE LAST ONE EXCEPT A RORB
3031                                  ;*      IS USED INSTEAD OF AN ASRB.
3032                                  ;*
3033                                  ;*      FORK B WILL ONLY FAIL IF IRCB E36(13) IS STUCK HIGH
3034                                  ;*      WHICH WILL CAUSE EXECUTION TO GO TO EXC.00.
3035                                  ;*
3036                                  ;*      ROM FLOW-2,175,137,64,123,132
3037                                  ;;*****************************************************************
3038  006702  005200          TST66:  INC     R0              ;INCREMENT TEST NUMBER
3039  006704  012705  001164          MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 HIGH BYTE IN R5
3040  006710  112725  000100          MOVB    #100,(R5)+      ;SET BIT 6 IN LOW BYTE OF $TMP1
3041  006714  112715  000200          MOVB    #200,(R5)       ;SET SIGN BIT IN HIGH BYTE OF $TMP1
```

```
3042  006720  000240                SYNC66: NOP
3043  006722                        IUT66:
3044  006722  106025                        RORB    (R5)+           ;EXECUTE INSTRUCTION UNDER TEST
3045  006724  022745  040100                CMP     #40100,-(R5)    ;DID INSTRUCTION WORK?
3046  006730  001401                        BEQ     TST67           ;;BRANCH IF YES
3047  006732  000000                        HALT                    ;IRCB E36(13) NOT GOING LOW ON RORB
3048                                                                ;FOR LOOPING CHANGE TO 'BR TST66+2'' (764)
3049
3050
3051
3052                                ;;*****************************************************************
3053                                ;*THE LOGICAL FLOW AT THIS POINT WOULD TEST A DAC*DM3*[TST.B*BIT.B*(MP.B]*DR0(0)
3054                                ;*FOLLOWED BY A DAC*DM4*P/CLASS*DR0(0) INSTRUCTION BUT NO ADDITIONAL LOGIC
3055                                ;*IS TESTED
3056                                ;;*****************************************************************
3057
3058
3059
3060                                ;;*****************************************************************
3061                                ;*TEST 67        SIX   MICROSTATES (DAC*DM6*XOR*DR0(0))
3062                                ;*
3063                                ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
3064                                ;*      THIS SHOULD ONLY HAPPEN IF RACE E35(1) IS BAD.
3065                                ;*
3066                                ;*      IF THE INSTRUCTION DOESN'T WORK IT WILL HALT IN THIS TEST.
3067                                ;*
3068                                ;*      ROM FLOW-6,251,122,177,31,132
3069                                ;;*****************************************************************
3070  006734  005200                TST67:  INC     R0              ;INCREMENT TEST NUMBER
3071  006736  005004                        CLR     R4              ;SETUP R4
3072  006740  005104                        COM     R4              ;SET ALL BITS IN R4
3073  006742  010437  001164                MOV     R4,@#$TMP1      ;SET ALL BITS IN $TMP1
3074  006746  000240                SYNC67: NOP
3075  006750                        IUT67:
3076  006750  074467  172210                XOR     R4,$TMP1        ;EXECUTE INSTRUCTION UNDER TEST
3077  006754  005767  172204                TST     $TMP1           ;DID $TMP1 CLEAR?
3078  006760  001401                        BEQ     TST70           ;;BRANCH IF YES
3079  006762  000000                        HALT                    ;INSTRUCTION FAILED
3080                                                                ;FOR LOOPING CHANGE TO 'BR TST67+2'' (765)
3081
3082
3083
3084
3085                                ;;*****************************************************************
3086                                ;*THE LOGICAL SEQUENCE WOULD TEST A DAC*DM6*[TST.B*BIT.B*(MP.B]*DR0(') BUT ALL
3087                                ;*THE LOGIC HAS BEEN TESTED.
3088                                ;;*****************************************************************
3089
3090
3091
3092                                ;;*****************************************************************
3093                                ;*TEST 70        SIX   MICROSTATES (NEG.B*DM12*DR0(0))
3094                                ;*
3095                                ;*      NEITHER FORK A NOR BGN15 SHOULD FAIL.
3096                                ;*
3097                                ;*      IF FORK B FAILS EXECUTION WILL GO TO RSD.00 CAUSING
```

```
3098                                    ;*      A TRAP TO LOCATION 10 OR EXC.00.
3099                                    ;*      RSD.00 SHOULD ONLY OCCUR IF THE B FORK MUX
3100                                    ;*      STROBE IS BEING HELD LOW(CHIP FAILURE).
3101                                    ;*
3102                                    ;*      ROM FLOW-1,175,67,271,163,132
3103                                    ;;*****************************************************************
3104  006764  005200                    TST70:  INC     R0              ;INCREMENT TEST NUMBER
3105  006766  005067  172172                    CLR     $TMP1           ;ENSURE $TMP1 CLEAR
3106  006772  005267  172166                    INC     $TMP1           ;PUT 1 IN $TMP1
3107  006776  012705  001164                    MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
3108  007002  000240                    SYNC70: NOP
3109  007004                            IUT70:
3110  007004  005415                            NEG     (R5)            ;EXECUTE INSTRUCTION UNDER TEST
3111  007006  022715  177777                    CMP     #177777,(R5)    ;DID $TMP1 NEGATE?
3112  007012  001411                            BEQ     3$              ;BRANCH IF YES
3113  007014  022715  177776                    CMP     #177776,(R5)    ;DID $TMP1 COMPLEMENT?
3114  007020  001001                            BNE     1$              ;BRANCH IF NO
3115  007022  000000                            HALT                    ;EITHER STATE NEG.10 FAILED
3116                                                                    ;OR FORK B FAILED. IRCB NEG.B H
3117                                                                    ;IS STUCK HIGH.
3118                                                                    ;FOR LOOPING CHANGE TO 'BR TST70+2'' (761)
3119  007024  022715  000001           1$:     CMP     #1,(R5)         ;DID $TMP1 STAY THE SAME?
3120  007030  001001                            BNE     2$              ;BRANCH IF NO
3121  007032  000000                            HALT                    ;$TMP1 DID NOT GET LOADED
3122                                                                    ;FOR LOOPING CHANGE TO 'BR TST70+2'' (755)
3123  007034                            2$:
3124  007034  000000                            HALT                    ;INSTRUCTION FAILED
3125                                                                    ;FOR LOOPING CHANGE TO 'BR TST70+2'' (754)
3126  007036  000264                    3$:     SEZ                     ;ENSURE Z SET
3127  007040  005415                            NEG     (R5)            ;EXECUTE INSTRUCTION UNDER TEST
3128  007042  001001                            BNE     TST71           ;;CC'S OK
3129  007044  000000                            HALT                    ;STATE NEG.10 BAD
3130                                                                    ;FOR LOOPING CHANGE TO 'BR TST70+2'' (750)
3131
3132
3133
3134                                    ;;*****************************************************************
3135                                    ;*THE LOGICAL SEQUENCE WOULD NEXT EXECUTE A JMP*DM3 BUT NO
3136                                    ;*ADDITIONAL LOGIC IS TESTED.
3137                                    ;;*****************************************************************
3138
3139
3140                                    ;;*****************************************************************
3141                                    ;*TEST 71        SIX   MICROSTATES (BIN*SM3*DM0*-DF7*SR0(0))
3142                                    ;*
3143                                    ;*      FORK A SHOULD NOT FAIL.
3144                                    ;*
3145                                    ;*      IF BEN14*FEN4 FAILS EXECUTION WILL GO TO D00.90. THIS
3146                                    ;*      WILL CAUSE A SM2 TO BE EXECUTED. THIS SHOULD ONLY
3147                                    ;*      HAPPEN IF IRCC SM357 IS STUCK LOW OR NOT GETTING THRU RACL E70.
3148                                    ;*
3149                                    ;*      ROM FLOW-22,27,317,143,146,205
3150                                    ;;*****************************************************************
3151  007046  005200                    TST71:  INC     R0              ;INCREMENT TEST NUMBER
3152  007050  012705  001164                    MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
3153  007054  012715  007064                    MOV     #POINT2,(R5)    ;PUT ADDRESS OF POINT2 IN $TMP1
```

```
3154  007060  000240                  SYNC71: NOP
3155  007062                          IUT71:
3156  007062  013501                          MOV     @(R5)+,R1       ;EXECUTE INSTRUCTION UNDER TEST
3157  007064  026701  177774          POINT2: CMP     POINT2,R1       ;DID R1 GET CORRECT DATA?
3158  007070  001405                          BEQ     TST72           ;;BRANCH IF YES
3159  007072  022701  007064                  CMP     #POINT2,R1      ;DID A SM2 GET EXECUTED?
3160  007076  001001                          BNE     2$              ;BRANCH IF NO
3161  007100  000000                          HALT                    ;EITHER IRCC SM357 STUCK LOW
3162                                                                  ;OR NOT GETTING THRU RACL E70
3163                                                                  ;FOR LOOPING CHANGE TO 'BR TST71+2'' (763)
3164  007102                          2$:
3165  007102  000000                          HALT                    ;EITHER S13.20 OR S13.30 OR S13.40 FAILED
3166                                                                  ;FOR LOOPING CHANGE TO 'BR TST71+2'' (762)
3167
3168
3169                                  ;;*****************************************************************
3170                                  ;*THE LOGICAL SEQUENCE WOULD NEXT TEST A BIN*SM6*DM0*DF7*SR0(1), THEN A BIN*SM12*
3171                                  ;*DM12*SR0(0)*DR0(1)*[TST.B+BIT.B+CMP.B] THEN A BIN*SM12*DM12*SR0(0)*
3172                                  ;*DR0(0)*P/CLASS THEN A BIN*SM12*DM12*SR0(1)*DR0(1)*[TST.B+BIT.B+CMP.B]
3173                                  ;*THEN A BIN*SM12*DM12*SR0(1)*DR0(0)*P/CLASS AND THEN A BIN*SM12*DM4*
3174                                  ;*SR0(0)*DR0(0)*[TST.B+BIT.B+CMP.B] INSTRUCTION, BUT NO ADDITIONAL LOGIC
3175                                  ;*IS TESTED.
3176                                  ;;*****************************************************************
3177
3178
3179                                  ;;*****************************************************************
3180                                  ;*TEST 72       SIX   MICROSTATES (BIN*SM12*DM4*SR0(1)*DR0(0)*CMPB)
3181                                  ;*
3182                                  ;*      A FAILURE WILL ONLY OCCUR IF STATE D45.90 FAILS.
3183                                  ;*
3184                                  ;*      IF THE DST REG. DOES NOT DECREMENT STATE D45.90 IS BAD.
3185                                  ;*      IF THE CONDITION CODES ARE BAD THEN STATE D40.30
3186                                  ;*      PROBABLY DID NOT SWAP THE BYTES OF THE SRC OPERAND.
3187                                  ;*
3188                                  ;*      ROM FLOW-1,27,114,131,177,33
3189                                  ;;*****************************************************************
3190  007104  005200                  TST72:  INC     R0              ;INCREMENT TEST NUMBER
3191  007106  012705  100000                  MOV     #BIT15,R5       ;SET SIGN BIT IN R5
3192  007112  010567  172046                  MOV     R5,$TMP1        ;SET SIGN BIT IN $TMP1
3193  007116  012701  001166                  MOV     #$TMP2,R1       ;PUT ADDRESS OF $TMP2 IN R1
3194  007122  010105                          MOV     R1,R5           ;PUT ADDRESS OF $TMP2 IN R5
3195  007124  112721  000200                  MOVB    #BIT7,(R1)+     ;SET LOW BYTE SIGN IN $TMP2 & STEP R1
3196  007130  105011                          CLRB    (R1)            ;ENSURE $TMP2 HIGH BYTE CLEAR
3197  007132  005305                          DEC     R5              ;ADJUST R5 TO POINT AT $TMP1 HIGH BYTE
3198  007134  000240                  SYNC72: NOP
3199  007136                          IUT72:
3200  007136  121541                          CMPB    (R5),-(R1)      ;EXECUTE INSTRUCTION UNDER TEST
3201  007140  001405                          BEQ     TST73           ;;TEST OK, GO TO NEXT TEST
3202  007142  020127  001166                  CMP     R1,#$TMP2       ;DID R1 DECREMENT?
3203  007146  001001                          BNE     1$              ;BRANCH IF YES
3204  007150  000000                          HALT                    ;STATE D45.90 DID NOT DECREMENT
3205                                                                  ;FOR LOOPING CHANGE TO 'BR TST72+2'' (756)
3206  007152                          1$:
3207  007152  000000                          HALT                    ;INSTRUCTION FAILED
3208                                                                  ;FOR LOOPING CHANGE TO 'BR TST72+2'' (755)
3209
```

```
3210
3211                            ;;*****************************************************************
3212                            ;*THE LOGICAL FLOW WOULD NEXT TEST A BIN*SM4*DM12*SR0(0)*DR0(0)*O/CLASS
3213                            ;*THEN A BIN*SM4*DM12*SR0(0)*DR0(0)*[TST.B+BIT.B+CMP.B] THEN A BIN*SM4*
3214                            ;*DM12*SR0(1)*DR0(0)*[TST.B+BIT.B+CMP.B] THEN A BIN*SM4*DM4*SR0(0)*DR0(0)*
3215                            ;*O/CLASS INSTRUCTION, BUT NO ADDITIONAL LOGIC IS TESTED.
3216                            ;;*****************************************************************
3217
3218
3219                            ;;*****************************************************************
3220                            ;*TEST 73       SIX   MICROSTATES (DAC*DM5*DR0(0)*O/CLASS)
3221                            ;*
3222                            ;*      FORK A SHOULD NOT FAIL.
3223                            ;*
3224                            ;*      IF IRCD DM357 IS STUCK LOW OR NOT GETTING THRU
3225                            ;*      TO RACK E51 A DM4 WILL BE EXECUTED.
3226                            ;*      IF STATE D10.00 OR D10.10 FAIL TO FETCH THE DEFERED ADDRESS
3227                            ;*      THE SOURCE WILL BE STORED IN THE DESTINATION.
3228                            ;*
3229                            ;*      ROM FLOW-5,162,231,233,311,157
3230                            ;;*****************************************************************
3231  007154  005200            TST73:  INC     R0              ;INCREMENT TEST NUMBER
3232  007156  012705  100000            MOV     #BIT15,R5       ;SET SIGN BIT IN R5
3233  007162  012701  001166            MOV     #$TMP2,R1       ;PUT ADDRESS OF $TMP2 IN R1
3234  007166  010167  171772            MOV     R1,$TMP1        ;PUT ADDRESS OF $TMP2 IN $TMP1
3235  007172  005011                    CLR     (R1)            ;ENSURE $TMP2 CLEAR
3236  007174  000240            SYNC73: NOP
3237  007176                    IUT73:
3238  007176  010551                    MOV     R5,@-(R1)       ;EXECUTE INSTRUCTION UNDER TEST
3239  007200  005767  171762            TST     $TMP2           ;DID SIGN BIT GET SET IN $TMP2?
3240  007204  100405                    BMI     TST74           ;;BRANCH IF YES
3241  007206  020567  171752            CMP     R5,$TMP1        ;DID MODE 4 GET EXECUTED?
3242  007212  001001                    BNE     1$
3243  007214  000000                    HALT                    ;EITHER IRCD DM357 IS NOT GOING LOW
3244                                                            ;OR NOT GETTING THRU TO RACK E51
3245                                                            ;FOR LOOPING CHANGE TO 'BR TST73+2'' (760)
3246  007216                    1$:
3247  007216  000000                    HALT                    ;INSTRUCTION FAILED
3248                                                            ;FOR LOOPING CHANGE TO 'BR TST73+2'' (757)
3249
3250
3251                            ;;*****************************************************************
3252                            ;*THE LOGICAL SEQUENCE WOULD NEXT TEST A DAC*DM3*DR0(0)*P/CLASS THEN A
3253                            ;*DAC*DM3*DR0(1)*[TST.B+BIT.B+CMP.B] THEN A DAC*DM4*DR0(1)*[ASRB+
3254                            ;*RORB] THEN A DAC*DM5*DR0(0)*[TST.B+BIT.B+CMP.B] THEN A DAC+DM6*
3255                            ;*DR0(1)*P/CLASS INSTRUCTION, BUT NO ADDITIONAL LOGIC IS TESTED.
3256                            ;;*****************************************************************
3257
3258
3259                                    .SBTTL
3260                            ;;*****************************************************************
3261                            ;*TEST 74       SEVEN MICROSTATES (DAC*DM7*O/CLASS)
3262                            ;*
3263                            ;*      FORK A SHOULD NOT FAIL.
3264                            ;*
3265                            ;*      IF IRCD DM357 DOES NOT GO HIGH A DM6 WILL BE EXECUTED.
```

```
3266
3267                                        ;*
3268                                        ;*      ALL OTHER LOGIC HAS BEEN TESTED.
3269                                        ;*
3270                                        ;*      ROM FLOW-7,251,162,231,233,311,157
3271  007220  005200                        ;;*****************************************************************
3272  007222  012705  001166                TST74:  INC     R0              ;INCREMENT TEST NUMBER
3273  007226  010567  171732                        MOV     #$TMP2,R5       ;PUT ADDRESS OF $TMP2 IN R5
3274  007232  012701  001162                        MOV     R5,$TMP1        ;PUT ADDRESS OF $TMP2 IN $TMP1
3275  007236  012702  100000                        MOV     #$TMP0,R1       ;PUT ADDRESS OF $TMP0 IN R1
3276  007242  005067  171720                        MOV     #BIT15,R2       ;SET SIGN BIT IN R2
3277  007246  000240                        SYNC74: CLR     $TMP2           ;ENSURE $TMP2 CLEAR
3278  007250                                 IUT74:  NOP
3279  007250  010271  000002                        MOV     R2,@2(R1)       ;EXECUTE INSTRUCTION UNDER TEST
3280  007254  005767  171706                        TST     $TMP2           ;DID $TMP2 GET SIGN BIT SET?
3281  007260  100405                                BMI     TST75           ;;BRANCH IF YES
3282  007262  020267  171676                        CMP     R2,$TMP1        ;DID DM6 GET EXECUTED?
3283  007266  001001                                BNE     1$              ;BRANCH IF NO
3284  007270  000000                                HALT                    ;IRCD DM357 DID NOT GO HIGH
3285                                                                         ;FOR LOOPING CHANGE TO 'BR TST74+2'' (754)
3286  007272                                 1$:
3287  007272  000000                                HALT                    ;INSTRUCTION FAILED
3288                                                                         ;FOR LOOPING CHANGE TO 'BR TST74+2'' (753)
3289
3290
3291                                        ;;*****************************************************************
3292                                        ;*THE LOGICAL SEQUENCE WOULD NEXT TEST A NEG.B*DM12*DR0(1) THEN A NEG.B*DM4*DR0(0)
3293                                        ;*THEN A JMP*DM5 INSTRUCTION, BUT NO ADDITIONAL LOGIC IS TESTED.
3294                                        ;;*****************************************************************
3295
3296
3297                                        ;;*****************************************************************
3298                                        ;*TEST 75        SEVEN MICROSTATES (JSR*DM12)
3299                                        ;*
3300                                        ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO LOCATION 10.
3301                                        ;*      THIS WILL ONLY OCCUR IF RACE JMP+JSR+SWAB DOES NOT GO HIGH.
3302                                        ;*
3303                                        ;*      IF EITHER IRCB IR(14:9)04 DOES NOT GO LOW OR E53 IS BAD
3304                                        ;*      EXECUTION WILL GO FROM D12.10 TO EXC.00.
3305                                        ;*      IF IRCB E63 IS BAD (PIN 10 OR 485 FLOATING) EXECUTION WILL
3306                                        ;*      GO TO RSD.00 CAUSING A TRAP TO LOCATION 10. THIS FAILURE
3307                                        ;*      WOULD INCREMENT THE DST REG. BEFORE THE TRAP.
3308                                        ;*
3309                                        ;*      IF THE INSTRUCTION FAILS THEN ONE OF THE JSR STATES FAILED.
3310                                        ;*
3311                                        ;*      ROM FLOW-2,135,34,201,274,275,32
3312                                        ;;*****************************************************************
3313  007274  005200                        TST75:  INC     R0              ;INC TST NUMBER
3314  007276  012706  001076                        MOV     #1076,SP        ;INITIALIZE SP
3315  007302  012705  007322                        MOV     #T67,R5         ;PUT ADDRESS OF T67A IN R5
3316  007306  010701                                MOV     PC,R1           ;PUT RANDOM NUMBER IN R1
3317  007310                                 SYNC75:
3318  007310  000277                        T67A:   SCC                     ;ENSURE ALL CC'S SET
3319  007312                                 IUT75:
3320  007312  004125                                JSR     R1,(R5)+        ;EXECUTE INSTRUCTION UNDER TEST
3321  007314  100001                        T67B:   BPL     T67C            ;BRANCH IF N CLEARED
```

```
3322  007316  000000                  HALT                    ;PCB DID NOT LOAD
3323                                                          ;FOR LOOPING CHANGE TO 'BR TST75+2'' (767)
3324  007320                  T67C:
3325  007320  000000                  HALT                    ;FORK B FAILED TO EXC.00(SEE ABOVE)
3326                                                          ;FOR LOOPING CHANGE TO 'BR TST75+2'' (766)
3327  007322  022716  007310  T67:    CMP     #T67A,(SP)      ;DID R1 GET STACKED?
3328  007326  001401                  BEQ     4$              ;BRANCH IF YES
3329  007330  000000                  HALT                    ;REGISTER DID NOT GET STACKED
3330                                                          ;FOR LOOPING CHANGE TO 'BR TST75+2'' (762)
3331  007332  022701  007314  4$:     CMP     #T67B,R1                ;DID R1 GET LOADED?
3332  007336  001401                  BEQ     5$              ;BRANCH IF YES
3333  007340  000000                  HALT                    ;REGISTER DID NOT LOAD
3334                                                          ;FOR LOOPING CHANGE TO 'BR TST75+2'' (756)
3335  007342  022706  001074  5$:     CMP     #1074,SP        ;DID SP GET DECREMENTED?
3336  007346  001401                  BEQ     TST76           ;;BRANCH IF YES
3337  007350  000000                  HALT                    ;SP DID NOT DECREMENT
3338                                                          ;FOR LOOPING CHANGE TO 'BR TST75+2'' (752)
3339
3340
3341                          ;;*****************************************************************
3342                          ;*THE LOGICAL SEQUENCE WOULD NEXT EXECUTE A BIN*SM3*DM0*-DF7*SR0(1) THEN
3343                          ;*A BIN*SM3*DM0*DF7*SR0(0) THEN A BIN*SM3*DM0*DF7*SR0(1) INSTRUCTION,
3344                          ;*BUT NO ADDITIONAL LOGIC IS TESTED.
3345                          ;;*****************************************************************
3346
3347
3348                          ;;*****************************************************************
3349                          ;*TEST 76       SEVEN MICROSTATES (BIN*SM5*DM0*-DF7*SR0(0))
3350                          ;*
3351                          ;*      FORK A SHOULD NOT FAIL.
3352                          ;*
3353                          ;*      IF BEN14*FEN4 FAILS EXECUTION WILL GO TO D00.90
3354                          ;*      CAUSING A SM4 INSTRUCTION TO BE EXECUTED. THIS WILL ONLY
3355                          ;*      OCCUR IF EITHER IRCC SRCM5 DOES NOT GO LOW OR IF IRCC E28 IS BAD.
3356                          ;*
3357                          ;*      ROM FLOW-24,23,27,317,143,146,205
3358                          ;;*****************************************************************
3359  007352  005200  TST76:  INC     R0                      ;INCREMENT TEST NUMBER
3360  007354  012705  001166          MOV     #$TMP2,R5       ;PUT ADDRESS OF $TMP2 IN R5
3361  007360  010567  171600          MOV     R5,$TMP1        ;PUT ADDRESS OF $TMP2 IN $TMP1
3362  007364  012715  100000          MOV     #BIT15,(R5)     ;SET SIGN BIT IN $TMP2
3363  007370  005001                  CLR     R1              ;ENSURE R1 CLEAR
3364  007372  000240  SYNC76: NOP
3365  007374          IUT76:
3366  007374  015501                  MOV     @-(R5),R1       ;EXECUTE INSTRUCTION UNDER TEST
3367  007376  022701  100000          CMP     #BIT15,R1       ;DID INSTRUCTION WORK?
3368  007402  001405                  BEQ     TST77           ;;BRANCH IF YES
3369  007404  020167  171554          CMP     R1,$TMP1        ;DID MODE 4 EXECUTE?
3370  007410  001001                  BNE     1$              ;BRANCH IF NO
3371  007412  000000                  HALT                    ;EITHER IRCC SRCM5 NOT GOING LOW OR IRCC E28 BAD
3372                                                          ;FOR LOOPING CHANGE TO 'BR TST76+2'' (760)
3373  007414          1$:
3374  007414  000000                  HALT                    ;INSTRUCTION FAILED
3375                                                          ;FOR LOOPING CHANGE TO 'BR TST76+2'' (757)
3376
3377                          ;;*****************************************************************
```

```
3378                                    ;*THE LOGICAL SEQUENCE WOULD NEXT EXECUTE A BIN*SM12*DM12*SR0(0)*DR0(1)*
3379                                    ;*P/CLASS THEN A BIN*SM12*DM12*SR0(1)*DR0(1)P/CLASS INSTRUCTION, BUT
3380                                    ;*NO ADDITIONAL LOGIC IS TESTED.
3381                                    ;;*****************************************************************
3382
3383
3384                                    ;;*****************************************************************
3385                                    ;*TEST 77        SEVEN MICROSTATES (BIN*+SM12*DM3*0/CLASS)
3386                                    ;*
3387                                    ;*      IF IRCC C FORK MUX INPUT B2 IS NOT GOING LOW OR IRCC E40
3388                                    ;*      IS BAD A DM2 WILL BE EXECUTED.
3389                                    ;*
3390                                    ;*      THE ONLY OTHER  POSSIBLE FAILURE IS STATE D30.80.
3391                                    ;*
3392                                    ;*      ROM FLOW-21,27,113,221,233,311,157
3393                                    ;;*****************************************************************
3394  007416  005200                    TST77:  INC     R0              ;INCREMENT TEST NUMBER
3395  007420  005067  171542                    CLR     $TMP2           ;ENSURE $TMP2 CLEAR
3396  007424  012705  001164                    MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
3397  007430  012715  001166                    MOV     #$TMP2,(R5)     ;PUT ADDRESS OF $TMP2 IN $TMP1
3398  007434  000240                    SYNC77: NOP
3399  007436                            IUT77:
3400  007436  011535                            MOV     (R5),@(R5)+     ;EXECUTE INSTRUCTION UNDER TEST
3401  007440  024515                            CMP     -(R5),(R5)      ;DID INSTRUCTION WORK?
3402  007442  001405                            BEQ     TST100          ;,BRANCH IF YES
3403  007444  022745  001166                    CMP     #$TMP2,-(R5)    ;DID DM2 GET EXECUTED?
3404  007450  001001                            BNE     1$              ;BRANCH IF NO
3405  007452  000000                            HALT                    ;EITHER C FORK MUX INPUT B2
3406                                                                    ;NOT GOING LOW OR IRCC E40 BAD
3407                                                                    ;FOR LOOPING CHANGE TO 'BR TST77+2'' (76?)
3408  007454                            1$:
3409  007454  000000                            HALT                    ;INSTRUCTION FAILED
3410                                                                    ;FOR LOOPING CHANGE TO 'BR TST77+2'' (76?)
3411
3412
3413                                    ;;*****************************************************************
3414                                    ;*THE LOGICAL SEQUENCE WOULD NEXT TEST A BIN*SM12*DM4*SR0(0)*DR0(0)*
3415                                    ;*P/CLASS FOLLOWED BY A BIN*SM12*DM4*SR0(0)*DR0(1)*[TST.B BIT.B+CMP.B]
3416                                    ;*FOLLOWED BY A BIN*SM12*DM4*SR0(1)*DR0(0)*P/CLASS FOLLOWED BY A
3417                                    ;*BIN*SM12*DM4*SR0(1)*DR0(1)*[TST.B+BIT.B+CMP.B] INSTRUCTION, BUT NO
3418                                    ;*ADDITIONAL LOGIC IS TESTED.
3419                                    ;;*****************************************************************
3420
3421
3422                                    ;;*****************************************************************
3423                                    ;*TEST 100       SEVEN MICROSTATES (BIN*SM12*DM6*0/CLASS)
3424                                    ;*
3425                                    ;*      IF FORK C FAILS EXECUTION WILL GO TO D45.90 AND A
3426                                    ;*      DM4 WILL BE EXECUTED. THIS WILL ONLY HAPPEN IF IRCC E39 PIN 5
3427                                    ;*      IS NOT GOING LOW. THIS WILL CAUSE AN RTI SINCE THE LOCATION
3428                                    ;*      FOLLOWING THE INSTRUCTION CONTAINS 000002.
3429                                    ;*
3430                                    ;*      THE ONLY OTHER FAILURE WOULD BE CAUSED BY STATE D67.80 BEING BAD.
3431                                    ;*
3432                                    ;*      ROM FLOW-21,27,117,6,251,122,157
3433                                    ;;*****************************************************************
```

```
3434  007456  005200                  TST100: INC     R0              ;INCREMENT TEST NUMBER
3435  007460  012706  001076                  MOV     #1076,SP        ;INITIALIZE THE SP
3436  007464  012746  000340                  MOV     #PR7,-(SP)      ;PUT PRIORITY LEVEL 7 ON STACK
3437  007470  012746  007526                  MOV     #T73,-(SP)      ;PUT ADDRESS OF T73 ON STACK
3438  007474  005067  171466                  CLR     $TMP2           ;ENSURE $TMP2 CLEAR
3439  007500  012705  001164                  MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
3440  007504  012715  100000                  MOV     #BIT15,(R5)     ;SET SIGN BIT IN $TMP1
3441  007510  000240                  SYN100: NOP
3442  007512                          IUT100:
3443  007512  011565  000002                  MOV     (R5),2(R5)      ;EXECUTE INSTRUCTION UNDER TEST
3444  007516  005767  171444                  TST     $TMP2           ;DID INSTRUCTION WORK?
3445  007522  100402                          BMI     TST101          ;;BRANCH IF YES
3446  007524  000000                          HALT                    ;INSTRUCTION FAILED
3447                                                                  ;FOR LOOPING CHANGE TO 'BR TST100+2'' (755)
3448  007526                          T73:
3449  007526  000000                          HALT                    ;IRCC E39(5) IS NOT GOING LOW
3450                                                                  ;FOR LOOPING CHANGE TO 'BR TST100+2'' (754)
3451
3452
3453                                  ;;*****************************************************************
3454                                  ;*THE LOGICAL SEQUENCE WOULD NEXT TEST THE INSTRUCTIONS BETWEEN
3455                                  ;*BIN*SM4*DM12*SR0(0)*DR0(1)*[TST.B+BIT.B+CMP.B] AND BIN*SM5*DM0*DF7*SR(
3456                                  ;*BUT NOT ADDITIONAL LOGIC IS TESTED.
3457                                  ;;*****************************************************************
3458
3459
3460                                          .SBTTL
3461                                  ;;*****************************************************************
3462                                  ;*TEST 101      EIGHT MICROSTATES (BIN*SM7*DM0*-DF7*SR0(0))
3463                                  ;*
3464                                  ;*      FORK A SHOULD NOT FAIL.
3465                                  ;*
3466                                  ;*      IF FEN4*BEN14 FAILS EXECUTION WILL GO TO D00.90 CAUSING
3467                                  ;*      A SM6 TO BE EXECUTED. THIS WILL ONLY HAPPEN IF EITHER
3468                                  ;*      IRCC SRCM7 DOES NOT GO LOW OR IF IRCC E28(1) IS BAD.
3469                                  ;*
3470                                  ;*      ROM FLOW-26,54,141,142,317,143,146,205
3471                                  ;;*****************************************************************
3472  007530  005200                  TST101: INC     R0              ;INCREMENT TEST NUMBER
3473  007532  012767  001164  171426          MOV     #$TMP1,$TMP2    ;PUT ADDRESS OF $TMP1 IN $TMP2
3474  007540  012767  100000  171416          MOV     #BIT15,$TMP1    ;SET SIGN BIT IN $TMP1
3475  007546  012705  001164                  MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
3476  007552  005001                          CLR     R1              ;ENSURE R1 CLEAR
3477  007554  000240                  SYN101: NOP
3478  007556                          IUT101:
3479  007556  017501  000002                  MOV     @2(R5),R1       ;EXECUTE INSTRUCTION UNDER TEST
3480  007562  005701                          TST     R1              ;DID R1 GET SIGN BIT SET?
3481  007564  100405                          BMI     TST102          ;;BRANCH IF YES
3482  007566  022701  001164                  CMP     #$TMP1,R1       ;DID SRCM6 GET EXECUTED?
3483  007572  001001                          BNE     1$              ;BRANCH IF NO
3484  007574  000000                          HALT                    ;EITHER IRCC SRCM7 DOES NOT GO LOW OR IRCC E28 BAD
3485                                                                  ;FOR LOOPING CHANGE TO 'BR TST101+2'' (756)
3486  007576                          1$:
3487  007576  000000                          HALT                    ;INSTRUCTION FAILED
3488                                                                  ;FOR LOOPING CHANGE TO 'BR TST101+2'' (755)
3489
```

```
                                ;;*****************************************************************
                                ;*THE LOGICAL SEQUENCE WOULD NEXT TEST A BIN*SM12*DM3*SR0(0)*DR0(0)*[TST.B*
                                ;*BIT.B*CMP.B] BUT NO ADDITIONAL LOGIC IS TESTED.
                                ;;*****************************************************************


                                ;;*****************************************************************
                                ;*TEST 102      EIGHT MICROSTATES (BIN*SM12*DM3*SR0(1)*DR0'0)*(MPB)
                                ;*
                                ;*      THE ONLY POSSIBLE FAILURE WOULD BE IN STATE D30.90
                                ;*      SINCE ALL THE OTHER LOGIC HAS BEEN TESTED.
                                ;*
                                ;*      ROM FLOW-21,27,112,221,233,311,177,33
                                ;;*****************************************************************
007600  005200                  TST102: INC     R0              ;INCREMENT TEST NUMBER
007602  012767  001166  171354          MOV     #$TMP2,$TMP1    ;PUT ADDRESS OF $TMP2 IN $TMP1
007610  012767  000377  171350          MOV     #377,$TMP2      ;SET LOW BYTE OF $TMP2 TO ALL ONES
007616  012767  177400  171336          MOV     #177400,$TMP0   ;SET HIGH BYTE OF $TMP0 TO ALL ONES
007624  012705  001163                  MOV     #$TMP0+1,R5     ;PUT ADDRESS OF $TMP0 HIGH BYTE IN R5
007630  012701  001164                  MOV     #$TMP1,R1       ;PUT ADDRESS OF $TMP1 IN R1
007634  000240                  SYN102: NOP
007636                          IUT102:
007636  121531                          CMPB    (R5),@(R1)+     ;EXECUTE INSTRUCTION UNDER TEST
007640  001401                          BEQ     TST103          ;;BRANCH IF TEST OK
007642  000000                          HALT                    ;STATE D30.90 FAILED
                                                                ;FOR LOOPING CHANGE TO 'BR TST102+2'' (757)


                                ;;*****************************************************************
                                ;*THE LOGICAL SEQUENCE WOULD NEXT TEST INSTRUCTIONS BIN*SM12*DM4*SR0(0)*DR0(1)*
                                ;*P/CLASS THRU BIN*SM12*DM6*SR0(0)*DR0(0)*[TST.B*BIT.B*CMP.B] BUT NO
                                ;*ADDITIONAL LOGIC IS TESTED.
                                ;;*****************************************************************


                                ;;*****************************************************************
                                ;*TEST 103      EIGHT MICROSTATES (BIN*SM12*DM6*SR0(1)*DR0(0)*CMPB)
                                ;*
                                ;*      THE ONLY POSSIBLE FAILURE IN THIS TEST IS STATE D67.9C.
                                ;*
                                ;*      ROM FLOW-21,27,116,6,251,122,177,33
                                ;;*****************************************************************
007644  005200                  TST103: INC     R0              ;INCREMENT TEST NUMBER
007646  012767  177400  171310          MOV     #177400,$TMP1   ;SET HIGH BYTE OF $TMP1 TO ALL ONES
007654  012705  001165                  MOV     #$TMP1+1,R5     ;PUT ADDRESS OF $TMP1 HI BYTE IN R5
007660  012767  000377  171300          MOV     #377,$TMP2      ;SET LOW BYTE OF $TMP2 TO ALL ONES
007666  000240                  SYN103: NOP
007670                          IUT103:
007670  121567  171272                  CMPB    (R5),$TMP2      ;EXECUTE INSTRUCTION UNDER TEST
007674  001401                          BEQ     TST104          ;;BRANCH IF TEST OK
007676  000000                          HALT                    ;STATE D67.90 FAILED
                                                                ;FOR LOOPING CHANGE TO 'BR TST103+2'' (767
                                        .SBTTL
                                ;;*****************************************************************
                                ;*TEST 104      NINE  MICROSTATES (BIN*SM12*DM5*SR0(0)*DR0(0)*CMP.B)
```

```
3546                                    ;*
3547                                    ;*      THE ONLY POSSIBLE FAILURE IN THIS TEST IS STATE D50.20.
3548                                    ;*
3549                                    ;*      ROM FLOW-21,27,115,161,231,233,311,177,33
3550                                    ;;*****************************************************************
3551  007700  005200                    TST104: INC     R0                      ;INCREMENT THE TEST NUMBER
3552  007702  012705  001162                    MOV     #$TMP0,R5        PUT ADDRESS OF $TMP0 IN R5
3553  007706  012767  000377  171246            MOV     #377,$TMP0      ;SET LOW BYTE OF $TMP0 TO ALL ONES
3554  007714  012701  001166                    MOV     #$TMP2,R1       ;PUT ADDRESS OF $TMP2 IN R1
3555  007720  012767  000377  171240            MOV     #377,$TMP2      ;SET LOW BYTE OF $TMP2 TO ALL ONES
3556  007726  012767  001166  171230            MOV     #$TMP2,$TMP1    ;PUT ADDRESS OF $TMP2 IN $TMP1
3557  007734  000240                    SYN104: NOP
3558  007736                            IUT104:
3559  007736  021551                            CMP     (R5),@-(R1)     ;EXECUTE INSTRUCTION UNDER TEST
3560  007740  001401                            BEQ     TST105          ;;GO TO NEXT TEST
3561  007742  000000                            HALT                    ;STATE D50.20 IS BAD
3562                                                                    ;FOR LOOPING CHANGE TO 'BR TST104+2'' (752
3563                                    ;;*****************************************************************
3564                                    ;*TEST 105      EIGHT MICROSTATES (BIN*SM12*DM5*SR0(1)*DR0(0)*CMPB)
3565                                    ;*
3566                                    ;*      THE ONLY POSSIBLE FAILURE IN THIS TEST IS STATE D50.30.
3567                                    ;;*****************************************************************
3568  007744  005200                    TST105: INC     R0                      ;INCREMENT THE TEST NUMBER
3569  007746  012705  001163                    MOV     #$TMP0+1,R5     ;PUT ADDRESS OF $TMP0 HIGH BYTE IN R5
3570  007752  012767  177400  171202            MOV     #177400,$TMP0   ;SET HIGH BYTE OF $TMP0 TO ALL ONES
3571  007760  012701  001166                    MOV     #$TMP2,R1       ;PUT ADDRESS OF $TMP2 IN R1
3572  007764  012767  000377  171174            MOV     #377,$TMP2      ;SET LOW BYTE OF $TMP2 TO ALL ONES
3573  007772  012767  001166  171164            MOV     #$TMP2,$TMP1    ;PUT ADDRESS OF $TMP2 IN $TMP1
3574  010000  121551                            CMPB    (R5),@-(R1)     ;EXECUTE INSTRUCTION UNDER TEST
3575  010002  001401                            BEQ     TST106          ;;GO TO NEXT TEST
3576  010004  000000                            HALT                    ;STATE D50.30 IS BAD
3577                                                                    ;FOR LOOPING CHANGE TO 'BR TST105+2'' (760
3578
3579                                    ;;******* ******************************************************
3580                                    ;*TEST 106      WRITE/READ PSW
3581                                    ;*
3582                                    ;*      THIS TEST VERIFIES THAT THE PSW CAN BE READ THRU THE DATA MUX.
3583                                    ;*      IF THE TEST FAILS ONE OF MANY THINGS COULD BE BAD WHICH CANNOT BE
3584                                    ;*      DETERMINED IN THIS DIAGNOSTIC.
3585                                    ;*      THIS TEST REQUIRES THAT SCCE PS ADRS GETS TO TMC,
3586                                    ;*      THAT THE TMC DMUX SELECT LINES GET TO PDR, THAT THE PSW
3587                                    ;*      BITS GET TO THE DMUX, THAT SCCE INTERNAL ADDRESS GETS TO TMC,
3588                                    ;*      AND SCCA VA00 GETS TO UBC.
3589                                    ;;*****************************************************************
3590  010006  005200                    TST106: INC     R0                      ;INCREMENT THE TEST NUMBER
3591  010010  012737  000240  177776            MOV     #PR5,@#PSW      ;SET PRIORITY BITS WITH A DATO
3592  010016  012701  000257                    MOV     #257,R1         ;PUT VALUE OF CC'S IN R1
3593  010022  000277                            SCC                     ;SET ALL THE CC'S WITH A CCOP INSTR
3594  010024  020137  177776                    CMP     R1,@#PSW        ;EXECUTE TEST MODE
3595  010030  001416                            BEQ     TST107          ;;BRANCH IF READ PSW WORKS
3596  010032  012701  177776                    MOV     #PSW,R1         ;GET ADDRESS OF PSW
3597  010036  000277                            SCC                     ;SET ALL THE CONDITION CODES
3598  010040  020137  177776                    CMP     R1,@#PSW        ;DID DMUX SELECT BUS REG?
3599  010044  001001                            BNE     1$              ;BRANCH IF NO
3600  010046  000000                            HALT                    ;EITHER TMCD E28 BAD OR SCCE PS ADDR
3601                                                                    ;NOT GETTING TO TMCD AS A HIGH
```

```
                                                        ;FOR LOOPING CHANGE TO 'BR TST106+2'' (760)
010050  005001          1$:     CLR     R1
010052  000277                  SCC
010054  020137  177776          CMP     R1,@#PSW        ;DOES TMCD LOW BYTE ENABLE GO HIGH?
010060  001001                  BNE     2$              ;BRANCH IF YES
010062  000000                  HALT                    ;EITHER TMCD LO BYTE EN DOES
                                                        ;NOT GO HIGH OR IT DOES NOT GET
                                                        ;THRU TO PDRE
                                                        ;FOR LOOPING CHANGE TO 'BR TST106+2'' (752)
010064          2$:
010064  000000                  HALT                    ;TEST FAILED, SEE TEST DESCRIPTION
                                                        ;FOR LOOPING CHANGE TO 'BR TST106+2''  751
                        ;;**********************************************************
                        ;*TEST 107      RTI
                        ;*
                        ;*      IF FORK A FAILS EXECUTION WILL GO TO ONE OF THREE STATES.
                        ;*      RSD.00 WILL CAUSE A TRAP TO LOCATION 4. THIS WOULD HAPPEN
                        ;*      IF RACF (HALT:OP CD 7) DOES NOT GO HIGH.
                        ;*      STATE D12.01 WOULD CAUSE AN ODD ADDRESS TRAP SINCE R0
                        ;*      WILL CONTAIN A 1. THIS WILL HAPPEN IF RACE E7 IS BAD.
                        ;*      HLT.00 WILL CAUSE THE PROCESSOR TO HALT ON THE INSTRUCTION
                        ;*      UNDER TEST AND WILL  OCCUR IF RACF E17 IS BAD.
                        ;*      IF THE INSTRUCTION DOESN'T WORK THEN ONE OF THE RTI MACHINE
                        ;*      STATES IS BAD.
                        ;*
                        ;*      ROM FLOW-12,156,212,213,214,215,172
                        ;;**********************************************************
010066  005200          TST107: INC     R0              ;INCREMENT TEST NUMBER
010070  012706  001100          MOV     #STACK,SP       ;INITIALIZE THE STACK
010074  012746  000340          MOV     #PR7,-(SP)      ;PUT PRIORITY LEVEL 7 ON STACK
010100  012746  010120          MOV     #T71,-(SP)      ;PUT ADDRESS OF T71 ON STACK
010104  012705  000001          MOV     #1,R5           ;PUT ODD ADDRESS IN R5.
010110  000240          SYN107: NOP
010112          IUT107:
010112  000002                  RTI                     ;EXECUTE INSTRUCTION UNDER TEST
010114  000240                  NOP                     ;IF THE PROCESSOR HALTS HERE
                                                        ;RACF E17 IS BAD(AFIR51(1)*(HALT:OP CD 7
010116  000000                  HALT                    ;PCB DID NOT GET LOADED
                                                        ;FOR LOOPING CHANGE TO 'BR TST107+2'' (76
010120  013705  177776  T71:    MOV     @#PSW,R5        ;GET PSW & PUT IN R5
010124  042705  177437          BIC     #^C<PR7>,R5     ;MASK OUT THE PSW
010130  020527  000340          CMP     R5,#PR7         ;DID PSW GET LOADED?
010134  001401                  BEQ     TST110          ;;BRANCH IF YES
010136  000000                  HALT                    ;EITHER PDRD E70(12) DOES NOT GO
                                                        ;HIGH ON LOAD PS AND KERNEL MODE
                                                        ;OR THE PRIORITY MUX ON PDRD IS BAD
                                                        ;FOR LOOPING CHANGE TO 'BR TST107+2''  75
                        ;;**********************************************************
                        ;*THE RTT WILL NOT BE TESTED HERE SINCE THE ONLY POSSIBLE FORK A
                        ;*FAILURE WOULD CAUSE AN RTI TO BE EXECUTED. THE T BIT FUNCTIONS OF
                        ;*THE RTI & RTT ARE TESTED IN PART 2.
                        ;;**********************************************************
                        ;;**********************************************************
                        ;*TEST 110      EMT AND TRAP
```

```
                                        ;*
                                        ;*      FORK A SHOULD NOT FAIL.
                                        ;*      THE INSTRUCTIONS ARE EXECUTED AND THE STACK IS CHECKED TO
                                        ;*      VERIFY THAT EVERYTHING WORKED OK.
                                        ;*
                                        ;*      ROM FLOW-0,345,354,SVC.00-SVC.90
                                        ;**************************************************************
010140  005200                  TST110: INC     R0              ;INCREMENT TEST NUMBER
010142  012737  000340  177776          MOV     #PR7,@#PSW      ;SET PRIORITY LEVEL AT  7
010150  012706  001100                  MOV     #STACK,SP       ;INITIALIZE THE STACK
010154  012737  010250  000030          MOV     #1$,@#EMTVEC    ;PUT ADDRESS OF 1$ IN EMT VECTOR
010162  012737  000240  000032          MOV     #PR5,@#EMTVEC+2 ;PUT PRIORITY LEVEL 5 IN EMTVEC +2
010170  012737  010304  000010          MOV     #4$,@#RESVEC    ;SETUP RESVEC
010176  012737  010306  000020          MOV     #5$,@#20        ;SETUP LOCATION 20
010204  012737  010310  000034          MOV     #6$,@#34        ;SETUP LOCATION 34
010212  012737  010312  000070          MOV     #7$,@#70        ;SETUP LOCATION 70
010220  012737  010314  000130          MOV     #8$,@#130       ;SETUP LOCATION 130
010226  012737  010316  000430          MOV     #9$,@#430       ;SETUP LOCATION 430
010234  012737  010320  000630          MOV     #10$,@#630      ;SETUP LOCATION 630
010242  000277                          SCC                     ;PUT PSW IN KNOWN CONFIGURATION
010244  104377                          EMT     377             ;EXECUTE INSTRUCTION UNDER TEST
010246  000000                          HALT                    ;NEW PC FAILED TO GET LOADED
                                                                ;FOR LOOPING CHANGE TO 'BR TST110+2'' (735)
010250  022726  010246          1$:     CMP     #1$-2,(SP)+     ;DID CORRECT PC GET STACKED?
010254  001401                          BEQ     2$              ;BRANCH IF YES
010256  000000                          HALT                    ;OLD PC DID NOT STACK CORRECTLY
                                                                ;FOR LOOPING CHANGE TO 'BR TST110+2'' (737)
010260  022716  000357          2$:     CMP     #357,(SP)       ;DID PSW GET STACKED PROPERLY?
010264  001401                          BEQ     3$              ;BRANCH IF YES
010266  000000                          HALT                    ;EITHER PSW DID NOT GET STACKED
                                                                ;PROPERLY OR PSW <7:5> BITS
                                                                ;DO NOT WORK
                                                                ;FOR LOOPING CHANGE TO 'BR TST110+2'' (741)
010270  000257                  3$:     CCC
010272  022737  000240  177776          CMP     #240,@#PSW      ;DID NEW PSW LOAD PROPERLY?
010300  001427                          BEQ     TST111          ;;BRANCH IF YES
010302  000000                          HALT                    ;EITHER PSW DID NOT GET LOADED
                                                                ;PROPERLY OR PSW <7:5> BITS
                                                                ;DO NOT WORK
                                                                ;FOR LOOPING CHANGE TO 'BR TST110+2'' (747)
010304                          4$:
010304  000000                          HALT                    ;EITHER IRCD EMT IS NOT GOING HIGH
                                                                ;OR DAPE TV03 IS NOT GOING HIGH
                                                                ;OR NOT GETTING TO THE ALU
                                                                ;FOR LOOPING CHANGE TO 'BR TST110+2'' (756)
010306                          5$:
010306  000000                          HALT                    ;EITHER DAPE TV02 IS NOT GOING HIGH
                                                                ;OR IT IS NOT GETTING TO THE ALU
                                                                ;FOR LOOPING CHANGE TO 'BR TST110+2'' (765)
010310                          6$:
010310  000000                          HALT                    ;EITHER DAPE TV01 IS STUCK HIGH
                                                                ;OR THE LOW IS NOT GETTING TO THE ALU
                                                                ;FOR LOOPING CHANGE TO 'BR TST110+2'' (774)
010312                          7$:
010312  000000                          HALT                    ;EITHER DAPE TV04 IS STUCK HIGH OR
                                                                ;THE LOW IS NOT GETTING TO THE AL
```

```
3714                                    ;FOR LOOPING CHANGE TO 'BR TST110+2'' (713)
```

```
3715  010314                                  8$:
3716  010314  000000                                HALT                    ;DAPE E24(B0) STUCK HIGH (CHIP FAILURE)
3717
3718                                                                        ;FOR LOOPING CHANGE TO 'BR TST110+2'' (712)
3719  010316                                  9$:
3720  010316  000000                                HALT                    ;DAPE E25(B0) STUCK HIGH(CHIP FAILURE)
3721                                                                        ;FOR LOOPING CHANGE TO 'BR TST110+2'' (711)
3722  010320                                  10$:
3723  010320  000000                                HALT                    ;DAPE TV05*07 STUCK HIGH
3724                                                                        ;FOR LOOPING CHANGE TO 'BR TST110+2'' (710)
3725                                          ;NOTE:  THE ONLY WAY THE TRAP INSTRUCTION SHOULD FAIL IS THAT IT GETS
3726                                          ;THE WRONG VECTOR. IF THIS HAPPENS A HALT IN LOW CORE SHOULD OCCUR.
3727  010322  012737  010352  000034                MOV     #11$,@#TRAPVEC  ;PUT ADDRESS OF 11$ IN TRAP VECTOR
3728  010330  012737  010346  000010                MOV     #12$,@#RESVEC   ;SETUP RESVEC
3729  010336  012737  010350  000014                MOV     #13$,@#BPTVEC   ;SETUP LOCATION 14
3730  010344  104777                               TRAP    377             ;EXECUTE INSTRUCTION UNDER TEST
3731  010346                                  12$:
3732  010346  000000                                HALT                    ;EITHER IRCD TRAP DOES NOT GO LOW
3733                                                                        ;OR IT IS NOT GETTING THRU DAPE
3734                                                                        ;FOR LOOPING CHANGE TO 'BR TST110+2'' (675)
3735  010350                                  13$:
3736  010350  000000                                HALT                    ;DAPE E7 IS BAD
3737                                                                        ;FOR LOOPING CHANGE TO 'BR TST110+2'' (674)
3738  010352  012737  000036  000034        11$:    MOV     #36,@#TRAPVEC   ;RESTORE .+2 TO TRAP VECTOR
3739                                          ;;*****************************************************************
3740                                          ;*TEST 111      IOT
3741                                          ;*
3742                                          ;*      FORK A SHOULD NOT FAIL.
3743                                          ;*
3744                                          ;*      IF THE TRAP VECTOR LOGIC FAILS THE TRAP VECTOR COULD COME
3745                                          ;*      OUT TO BE 0, 4, OR 24.
3746                                          ;*
3747                                          ;*      THE ONLY OTHER POSSIBLE FAILURE WOULD BE STATE TRP.01.
3748                                          ;*      IF THIS STATE FAILS TO LOAD THE DR THE TRAP VECTOR WILL
3749                                          ;*      BE WHATEVER IS IN R4.  IF IT FAILS TO LOAD THE BR THE OLD
3750                                          ;*      PS WILL FAIL TO BE STACKED.
3751                                          ;*
3752                                          ;*      ROM FLOW-14,354,(SVC.00-SVC.90) 355,65,357,360,367,37,25,41,222,300
3753                                          ;;*****************************************************************
3754  010360  005200                          TST111: INC     R0              ;INCREMENT TEST NUMBER
3755  010362  012737  010440  000020                MOV     #1$,@#20        ;SETUP THE IOT VECTOR
3756  010370  012706  001100                       MOV     #STACK,SP       ;SETUP THE SP
3757  010374  012737  010460  000000                MOV     #2$,@#0         ;SETUP LOCATION ZERO
3758  010402  012737  010462  000014                MOV     #3$,@#14        ;SETUP LOCATION 14
3759  010410  012704  000014                       MOV     #14,R4          ;SETUP R4
3760  010414  012737  010464  000024                MOV     #4$,@#24        ;SETUP LOCATION 24
3761  010422  012737  010466  000004                MOV     #5$,@#ERRVEC    ;SET UP LOCATION 4
3762  010430  012737  000300  177776                MOV     #PR6,@#PSW      ;SETUP THE PSW
3763  010436  000004                               IOT                     ;EXECUTE INSTRUCTION UNDER TEST
3764  010440  042766  177437  000002  1$:     BIC     #177437,2(SP)   ;MASK OF THE PRIORITY BITS ON THE OLD PSW
3765  010446  022766  000300  000002          CMP     #300,2(SP)      ;DID OLD SP GET STACKED?
3766  010454  001405                          BEQ     SEQENC          ;;BRANCH IF YES
3767  010456  000000                          HALT                    ;STATE TRP.01 FAILED TO LOAD BR
3768                                                                  ;FOR LOOPING CHANGE TO 'BR TST111+2'' (741)
3769  010460                                  2$:
3770  010460  000000                          HALT                    ;EITHER DAPE TV04 DOES NOT GO HIGH
```

```
3771                                                    ;OR IT DOES NOT GET TO THE ALU.
3772                                                    ;FOR LOOPING CHANGE TO 'BR TST111+2'' (740)
3773  010462                            3$:
3774  010462  000000                          HALT      ;STATE TRP.01 FILED TO LOAD DR
3775                                                    ;FOR LOOPING CHANGE TO 'BR TST111+2'' (737)
3776  010464                            4$:
3777  010464  000000                          HALT      ;EITHER IRCD IOT DOES NOT GET TO
3778                                                    ;DAPE E7(4) AS A LOW OR E' IS BAD
3779                                                    ;FOR LOOPING CHANGE TO 'BR TST111+2'' (736)
3780  010466                            5$:
3781  010466  000000                          HALT      ;EITHER IRCD IOT DOES NOT GO LOW
3782                                                    ;OR IT DOES NOT GET TO DAPE
3783                                                    ;FOR LOOPING CHANGE TO 'BR TST111+2'' (735)
3784  010470  016737  170404  177570  SEQENC: MOV       $PASS,@#SWR     ;DISPLAY PASS COUNT IN LIGHTS
3785  010476  005737  000042                  TST       @#42            ;ACT 11 OR XXDP LOAD?
3786  010502  001060                          BNE       1$              ;BRANCH IF YES
3787  010504  012737  011754  000034          MOV       #$TRAP,@#TRAPVEC ;SETUP TRAP VECTOR FOR PRINT
3788  010512  005227  177777                  INC       #-1             ;;FIRST TIME?
3789  010516  001033                          BNE       64$             ;;BRANCH IF NO
3790  010520  022737  011034  000042          CMP       #$ENDAD,@#42    ;;ACT-11?
3791  010526  001427                          BEQ       64$             ;;BRANCH IF YES
3792  010530  104400  010536                  TYPE      ,65$            ;;TYPE ASCIZ STRING
3793  010534  000424                          BR        64$             ;;GET OVER THE ASCIZ
3794                            ;;65$:  .ASCIZ  <CRLF>/CEKBA-C PDP-11 CPU DIAGNOSTIC PART 1/<CRLF><CRLF>
3795  010606                    64$:
3796  010606  005767  170272                  TST       $ICNT           ;CHANGED PASS COUNT YET?
3797  010612  001407                          BEQ       2$              ;BRANCH IF NO
3798  010614  022767  000001  000164          CMP       #1,$EOPCT       ;EOP COUNT AT 1 YET?
3799  010622  001010                          BNE       1$              ;BRANCH IF NO
3800  010624  005067  170254                  CLR       $ICNT           ;CLEAR CHANGED FLAG
3801  010630  000405                          BR        1$
3802  010632  005267  170246          2$:     INC       $ICNT           ;SET CHANGED FLAG
3803  010636  012767  001000  000142          MOV       #1000,$EOPCT    ;CHANGE PASS COUNT
3804  010644  022700  000111          1$:     CMP       #111,R0         ;IS TEST NUMBER OK?
3805  010650  001443                          BEQ       3$              ;BRANCH IF YES
3806  010652  012737  011754  000034          MOV       #$TRAP,@#TRAPVEC ;SETUP TRAP VECTOR
3807  010660  104400  010666                  TYPE      ,67$            ;;TYPE ASCIZ STRING
3808  010664  000414                          BR        66$             ;;GET OVER THE ASCIZ
3809                            ;;67$:  .ASCIZ  <15><12>/TEST NUMBER(R0) IS  /
3810  010716                    66$:
3811  010716  010067  170232                  MOV       R0,$REG0
3812  010722  016746  170226                  MOV       $REG0,-(SP)     ;;SAVE $REG0 FOR TYPEOUT
3813  010726  104402                          TYPOC                     ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3814  010730  104400  010736                  TYPE      ,69$            ;;TYPE ASCIZ STRING
3815  010734  000411                          BR        68$             ;;GET OVER THE ASCIZ
3816                            ;;69$:  .ASCIZ  /  SHOULD BE 111/<15><12>
3817  010760                    68$:
3818  010760  005037  177746          3$:     CLR       @#CONTRL        ;ENABLE CACHE
3819                            ;;*****************************************************************
3820                            ;;*****************************************************************
3821
3822                            .SBTTL  END OF PASS ROUTINE
3823
3824                            ;*INCREMENT THE PASS NUMBER ($PASS)
3825                            ;*INDICATE END-OF-PROGRAM AFTER 14 PASSES THRU THE PROGRAM
3826                            ;*TYPE 'END PASS''
```

```
                                        ;*IF THERES A MONITOR GO TO IT
                                        ;*IF THERE ISN'T JUMP TO TST1
                                        ;*IF IT IS DESIRED TO HAVE A BELL INDICATE THE 'END OF PASS' LOCATION
                                        ;*$ENDMG CAN BE CHANGED TO 7.

010764                                  $EOP:
010764  012737  011754  000034                  MOV     #$TRAP,@#TRAPVEC        ;SETUP TRAP VECTOR
010772  005267  170102                          INC     $PASS           ;;INCREMENT THE PASS NUMBER
010776  042767  100000  170074                  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
011004  005327                                  DEC     (PC)+           ;;LOOP?
011006  000014                          $EOPCT: .WORD   14
011010  003015                                  BGT     $DOAGN          ;;YES
011012  012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
011014  000014                          $ENDCT: .WORD   14
011016  011006                                  $EOPCT
011020  104400  011050                          TYPE    ,$ENDMG         ;;TYPE 'END PASS'
011024  013700  000042                  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
011030  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
011032  000005                                  RESET                   ;;CLEAR THE WORLD
011034  004710                          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
011036  000240                                  NOP                     ;;SAVE ROOM
011040  000240                                  NOP                     ;;FOR
011042  000240                                  NOP                     ;;ACT11
011044                                  $DOAGN:
011044  000137  001210                          JMP     @#TST1          ;;RETURN
011050  005015  047105  020104          $ENDMG: .ASCII  <15><12>/END PASS/
011056  040520  051523
011062     377     377     000          $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
        011066                                  .EVEN
                                        ;;*****************************************************************

                                        .SBTTL  TYPE ROUTINE

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

011066  105767  170057                  $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
011072  100002                                  BPL     1$              ;;BR IF YES
011074  000000                                  HALT                    ;;HALT HERE IF NO TERMINAL
011076  000407                                  BR      3$              ;;LEAVE
011100  010046                          1$:     MOV     R0,-(SP)        ;;SAVE R0
```

```
3883  011102  017600  000002          MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
3884  011106  112046          2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
3885  011110  001005                  BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
3886  011112  005726                  TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
3887  011114  012600                  MOV     (SP)+,R0        ;;RESTORE R0
3888  011116  062716  000002  3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
3889  011122  000002                  RTI                     ;;RETURN
3890  011124  122716  000011  4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
3891  011130  001426                  BEQ     8$
3892  011132  122716  000200          CMPB    #CRLF,(SP)      ;;BRANCH IF NOT
3893  011136  001004                  BNE     5$
3894  011140  005726                  TST     (SP)+           ;;POP  <CR><LF> EQUIV
3895  011142  104400  001173          TYPE    ,$CRLF
3896  011146  000757                  BR      2$              ;;GET NEXT CHARACTER
3897  011150  004767  000056  5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
3898  011154  126726  167770  6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
3899  011160  001352                  BNE     2$              ;;IF NO GO GET NEXT CHAR.
3900  011162  016746  167760          MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
3901                                                          ;;AND THE NULL CHAR.
3902  011166  105366  000001  7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
3903  011172  002770                  BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
3904  011174  004767  000032          JSR     PC,$TYPEC       ;;GO TYPE A NULL
3905  011200  105367  000072          DECB    $CHARCNT        ;;DON'T COUNT THE NULL AS A CHARACTER
3906  011204  000770                  BR      7$              ;;LOOP
3907
3908                                  ;;HORIZONTAL TAB PROCESSOR
3909
3910  011206  112716  000040  8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
3911  011212  004767  000014  9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
3912  011216  132767  000007  000052  BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
3913  011224  001372                  BNE     9$              ;;TAB STOP
3914  011226  005726                  TST     (SP)+           ;;POP SPACE OFF STACK
3915  011230  000726                  BR      2$              ;;GET NEXT CHARACTER
3916  011232  105777  167704  $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
3917  011236  100375                  BPL     $TYPEC
3918  011240  116677  000002  167676  MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
3919  011246  122766  000015  000002  CMPB    #CR,2(SP)       ;;BRANCH IF
3920  011254  001003                  BNE     1$              ;;NOT <CR>
3921  011256  105067  000014          CLRB    $CHARCNT        ;;
3922  011262  000406                  BR      $TYPEX          ;;EXIT
3923  011264  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;BRANCH IF
3924  011272  001402                  BEQ     $TYPEX          ;;<LF>
3925  011274  105227                  INCB    (PC)+           ;;INC SPACE
3926  011276  000000  $CHARCNT:.WORD  0                       ;;COUNT
3927  011300  000207  $TYPEX: RTS     PC
3928
3929                                  ;;**************************************************************************
3930
3931                                  .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
3932
3933                                  ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
3934                                  ;*OCTAL (ASCII) NUMBER AND TYPE IT.
3935                                  ;*$TYPOS---ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
3936                                  ;*CALL:
3937                                  ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
3938                                  ;*      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

011302  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
011306  116667  000001  000211          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
011314  112667  000207                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
011320  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
011324  000406                          BR      $TYPON
011326  112767  000001  000171  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
011334  112767  000006  000165          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
011342  112767  000005  000154  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
011350  010346                          MOV     R3,-(SP)        ;;SAVE R3
011352  010446                          MOV     R4,-(SP)        ;;SAVE R4
011354  010546                          MOV     R5,-(SP)        ;;SAVE R5
011356  116704  000145                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
011362  005404                          NEG     R4
011364  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
011370  110467  000132                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
011374  116704  000125                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
011400  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
011404  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
011406  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO "C"
011410  000404                          BR      3$              ;;GO DO MSB
011412  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
011414  006105                          ROL     R5
011416  006105                          ROL     R5
011420  010503                          MOV     R5,R3
011422  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
011424  105367  000076                  DECB    $OMODE          ;;TYPE THIS DIGIT?
011430  100016                          BPL     7$              ;;BR IF NO
011432  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
011436  001002                          BNE     4$              ;;TEST FOR 0
011440  005704                          TST     R4              ;;SUPPRESS THIS 0?
011442  001403                          BEQ     5$              ;;BR IF YES
011444  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
011446  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
011452  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
011456  110367  000040                  MOVB    R3,8$           ;;SAVE FOR TYPING
011462  104400  011522                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
011466  105367  000032          7$:     DECB    $OCNT           ;;COUNT BY 1
011472  003347                          BGT     2$              ;;BR IF MORE TO DO
011474  002402                          BLT     6$              ;;BR IF DONE
011476  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
```

```
                                        BR      2$              ;;GO DO THE LAST DIGIT
                                6$:     MOV     (SP)+,R5        ;;RESTORE R5
                                        MOV     (SP)+,R4        ;;RESTORE R4
                                        MOV     (SP)+,R3        ;;RESTORE R3
                                        MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
                                        MOV     (SP)+,(SP)
                                        RTI                     ;;RETURN
                                8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
                                        .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
                                $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
                                $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
                                $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
                                ;;**************************************************************

                                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE

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

                                $TYPDS:
                                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
                                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
                                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
                                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
                                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
                                        MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
                                        MOV     20(SP),R5       ;;GET THE INPUT NUMBER
                                        BPL     1$              ;;BR IF INPUT IS POS.
                                        NEG     R5              ;;MAKE THE BINARY NUMBER POS.
                                        MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
                                1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
                                        MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
                                        MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
                                2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
                                        MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
                                3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
                                        BLT     4$              ;;BR IF DONE
                                        INC     R2              ;;INCREASE THE BCD DIGIT BY 1
                                        BR      3$
                                4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
                                        TST     R2              ;;CHECK IF BCD DIGIT=0
                                        BNE     5$              ;;FALL THROUGH IF 0
                                        TSTB    (SP)            ;;STILL DOING LEADING 0'S?
                                        BMI     7$              ;;BR IF YES
                                5$:     ASLB    (SP)            ;;MSD?
                                        BCC     6$              ;;BR IF NO
                                        MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
                                6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
                                7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
                                        MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
```

```
4051  011652  005720                        TST     (R0)+           ;;JUST INCREMENTING
4052  011654  020027  000010                CMP     R0,#10          ;;CHECK THE TABLE INDEX
4053  011660  002746                        BLT     2$              ;;GO DO THE NEXT DIGIT
4054  011662  003002                        BGT     8$              ;;GO TO EXIT
4055  011664  010502                        MOV     R5,R2           ;;GET THE LSD
4056  011666  000764                        BR      6$              ;;GO CHANGE TO ASCII
4057  011670  105726                R8$:    TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
4058  011672  100003                        BPL     9$              ;;BR IF NO
4059  011674  116663  177777  177776        MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
4060  011702  105013                9$:     CLRB    (R3)            ;;SET THE TERMINATOR
4061  011704  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
4062  011706  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
4063  011710  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
4064  011712  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
4065  011714  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
4066  011716  104400  011744                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
4067  011722  016666  000002  000004        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
4068  011730  012616                        MOV     (SP)+,(SP)
4069  011732  000002                        RTI                     ;;RETURN TO USER
4070  011734  023420                $DTBL:  10000.
4071  011736  001750                        1000.
4072  011740  000144                        100.
4073  011742  000012                        10.
4074  011744  000004                $DBLK:  .BLKW   4
4075                                ;;**************************************************************
4076
4077                                .SBTTL  TRAP DECODER
4078
4079                                ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE ''TRAP'' INSTRUCTION
4080                                ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
4081                                ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
4082                                ;*GO TO THAT ROUTINE.
4083
4084  011754  010046                $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
4085  011756  016600  000002                MOV     2(SP),R0        ;;GET TRAP ADDRESS
4086  011762  005740                        TST     -(R0)           ;;BACKUP BY 2
4087  011764  111000                        MOVB    (R0),R0         ;;GET RIGHT BYTE OF 'RAP
4088  011766  016000  011774                MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
4089  011772  000200                        RTS     R0              ;;GO TO ROUTINE
4090
4091
4092                                .SBTTL  TRAP TABLE
4093
4094                                ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
4095                                ;*BY THE ''TRAP'' INSTRUCTION.
4096
4097                                ;       ROUTINE
4098                                ;       -------
4099  011774                        $TRPAD:
4100  011774  011066                        $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
4101  011776  011326                        $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
4102  012000  011302                        $TYPOS  ;;CALL=TYPOS    TRAP+4(104404)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
4103  012002  011342                        $TYPON  ;;CALL=TYPON    TRAP+6(104406)  TYPE OCTAL NUMBER (AS PER LAST CALL)
4104  012004  011530                        $TYPDS  ;;CALL=TYPDS    TRAP+10(104410) TYPE DECIMAL NUMBER (WITH SIGN)
4105  012006  012006                SUBTAB: .WORD   .
4106          000001                        .END
```

```
A       000107  1530# 1532# 1580# 1582# 1647# 1649# 1715# 1716# 1902# 1960# 1961# 2004# 2005#
                2073# 2074# 2109# 2110# 2127# 2129# 2150# 2151# 2196# 2197# 2221# 2222# 2240#
                2241# 2275# 2276# 2293# 2294# 2334# 2335# 2374# 2375# 2399# 2400# 2449# 2450#
                2483# 2484# 2530# 2531# 2586# 2587# 2623# 2624# 2673# 2674# 2701# 2702# 2726#
                2727# 2755# 2756# 2794# 2795# 2860# 2862# 2918# 2919# 2939# 2940# 2991# 2992#
                3042# 3043# 3074# 3075# 3108# 3109# 3154# 3155# 3198# 3199# 3236# 3237# 3277#
                3278# 3317# 3319# 3364# 3365# 3398# 3399# 3441# 3442# 3477# 3478# 3511# 3512#
                3537# 3538# 3557# 3558# 3634# 3635#
ADCR0   001544   897   898   899   901#  916
ADDR0   002250  1108  1109  1110  1112# 1126
AFTER   006572  2984  2994# 3000  3008  3016
ASLR0   001656   946   947   948   950#  963
ASRR0   001730   973   974   975   977#  990
BAD     005140  2273  2286# 2295  2298
BAD1    005432  2444  2457#
BICR0   002176  1081  1082  1083  1085# 1098
BISR0   002220  1093  1094  1095  1097# 1113
BITR0   002152  1068  1069  1070  1072# 1086
BIT0  = 000001   135# 2893
BIT00 = 000001   125#  135
BIT01 = 000002   124#  134
BIT02 = 000004   123#  133
BIT03 = 000010   122#  132
BIT04 - 000020   121#  131
BIT05 - 000040   120#  130
BIT06 - 000100   119#  129  2849
BIT07 - 000200   118#  128
BIT08 - 000400   117#  127
BIT09 - 001000   116#  126
BIT1  - 000002   134#
BIT10 - 002000   115#
BIT11   004000   114#
BIT12 = 010000   113#
BIT13 - 020000   112#
BIT14 - 040000   111# 2698
BIT15   100000   110# 1940  2108  2371  2395  2448  2479  2549  2618  2650  2725  2837  2938
                3191  3232  3275  3362  3367  3440  3474
BIT2    000004   133#
BIT3  - 000010   132#
BIT4    000020   131# 1538  1682
BIT5    000040   130#
BIT6  = 000100   129#
BIT7  - 000200   128# 3195
BIT8  = 000400   127#
BIT9  - 001000   126#
BPTVEC= 000014   142# 3729*
CACHVE- 000114   149#
CLRR0   001460   862   863   864   866#  879
CMPR0   002316  1133  1134  1135  1137#
COMR0   001524   886   887   888   890#  902
CONTRL- 177746   159#  618* 3818*
CPUERR= 177766   172#
CR    = 000015    47# 3919  3929
CRLF  - 000200    48# 3795  3892  3929
DECR0   001614   923   924   925   927#  939
DISPLA  177570    42#
```

```
DUM    - 000007          1530#  1580#  1647#  1715#  1902#  1960#  2004#  2073#  2109#  2127#  2150#  2196#  2221#
                         2240#  2275#  2293#  2334#  2374#  2399#  2449#  2483#  2530#  2586#  2623#  2673#  2701#
                         2726#  2755#  2794#  2860#  2918#  2939#  2991#  3042#  3074#  3108#  3154#  3198#  3236#
                         3277#  3317#  3364#  3398#  3441#  3477#  3511#  3537#  3557#  3634#
EMTVEC= 000030             145#  3668*  3669*
ERRVEC= 000004             138#  3761*
ER25    003656          1718#
FORKA   003366          1535   1545#
GNS     = ****** U       425   3794   3809   3816   4100   4101   4102   4103   4104
GOOD    005120          2273#  2277
GRA1T   002354          1175   1179#  1195
GRA2T   002410          1194   1198#  1214
HIADRS  177742           157#
HITMIS= 177752           161#
HT      = 000011          45#  3890   3929
INCRO   001634           934    935    936    938#   951
IOTVEC= 000020           143#
IT      006420          2887   2890#  2892
IUT     004304          1903#
IUT100  007512          3442#
IUT101  007556          3478#
IUT102  007636          3512#
IUT103  007670          3538#
IUT104  007736          3558#
IUT107  010112          3635#
IUT24   003334          1532#
IUT25   003430          1582#
IUT26   003524          1649#
IUT27   003654          1716#
IUT32   004426          1961#
IUT33   004504          2005#
IUT34   004630          2074#
IUT35   004710          2110#
IUT36   004732          2129#
IUT37   004754          2151#
IUT40   005020          2197#
IUT41   005054          2222#
IUT42   005076          2241#
IUT43   005126          2276#  2294#
IUT44   005220          2335#
IUT45   005302          2375#
IUT46   005330          2400#
IUT47   005414          2450#
IUT50   005460          2484#
IUT51   005536          2531#
IUT52   005632          2587#
IUT53   005706          2624#
IUT55   005776          2674#
IUT56   006024          2702#
IUT57   006060          2727#
IUT60   006116          2756#
IUT61   006144          2795#
IUT62   006344          2862#
IUT63   006464          2919#
IUT64   006512          2940#
IUT65   006570          2992#
```

```
IUT66   006762          3043#
IUT67   006750          3075#
IUT70   007004          3109#
IUT71   007062          3155#
IUT72   007136          3199#
IUT73   007176          3237#
IUT74   007250          3278#
IUT75   007312          3319#
IUT76   007374          3365#
IUT77   007436          3399#
JMPTO   006002          2672    2676#
JMP1AD  005032          2194    2206#
JMP2AD  005056          2220    2224#
KDPAR0= 172360           315#
KDPAR1= 172362           316#
KDPAR2= 172364           317#
KDPAR3= 172366           318#
KDPAR4= 172370           319#
KDPAR5= 172372           320#
KDPAR6= 172374           321#
KDPAR7- 172376           322#
KDPDR0= 172320           293#
KDPDR1  172322           294#
KDPDR2- 172324           295#
KDPDR3= 172326           296#
KDPDR4- 172330           297#
KDPDR5  172332           298#
KDPDR6= 172334           299#
KDPDR7= 172336           300#
KERSTK- 001100            32#
KIPAR0= 172340           304#
KIPAR1= 172342           305#
KIPAR2- 172344           306#
KIPAR3- 172346           307#
KIPAR4- 172350           308#
KIPAR5= 172352           309#
KIPAR6- 172354           310#
KIPAR7- 172356           311#
KIPDR0- 172300           282#
KIPDR1= 172302           283#
KIPDR2= 172304           284#
KIPDR3- 172306           285#
KIPDR4= 172310           286#
KIPDR5= 172312           287#
KIPDR6= 172314           288#
KIPDR7= 172316           289#
LF      = 000012          46#   3923    3929
LOADRS= 177740           156#
MAINT - 177750           160#
MAPH0 - 170202           399#
MAPH00= 170202           335#    399
MAPH01- 170206           337#    401
MAPH02= 170212           339#    403
MAPH03- 170216           341#    405
MAPH04  170222           343#    407
MAPH05  170226           345#    409
```

```
MAPH06= 170232           347#    411
MAPH07= 170236           349#    413
MAPH1 = 170206           401#
MAPH10= 170242           351#
MAPH11= 170246           353#
MAPH12= 170252           355#
MAPH13= 170256           357#
MAPH14= 170262           359#
MAPH15  170266           361#
MAPH16- 170272           363#
MAPH17= 170276           365#
MAPH2 = 170212           403#
MAPH20= 170302           367#
MAPH21= 170306           369#
MAPH22= 170312           371#
MAPH23= 170316           373#
MAPH24= 170320           375#
MAPH25= 170326           377#
MAPH26= 170332           379#
MAPH27= 170336           381#
MAPH3 = 170216           405#
MAPH30- 170342           383#
MAPH31  170346           385#
MAPH32- 170352           387#
MAPH33- 170356           389#
MAPH34- 170362           391#
MAPH35- 170366           393#
MAPH36= 170372           395#
MAPH37= 170376           397#
MAPH4 - 170222           407#
MAPH5 = 170226           409#
MAPH6 - 170232           411#
MAPH7 - 170236           413#
MAPL0   170200           398#
MAPL00- 170200           334#    398
MAPL01- 170204           336#    400
MAPL02- 170210           338#    402
MAPL03= 170214           340#    404
MAPL04- 170220           342#    406
MAPL05- 170224           344#    408
MAPL06- 170230           346#    410
MAPL07= 170234           348#    412
MAPL1   170204           400#
MAPL10- 170240           350#
MAPL11- 170244           352#
MAPL12  170250           354#
MAPL13- 170254           356#
MAPL14- 170260           358#
MAPL15- 170264           360#
MAPL16- 170270           362#
MAPL17- 170274           364#
MAPL2   170210           402#
MAPL20= 170300           366#
MAPL21= 170304           368#
MAPL22  170310           370#
MAPL23= 170314           372#
```

```
MAPL24= 170320   374#
MAPL25= 170324   376#
MAPL26= 170330   378#
MAPL27= 170334   380#
MAPL3 = 170214   404#
MAPL30= 170340   382#
MAPL31= 170344   384#
MAPL32= 170350   386#
MAPL33= 170354   388#
MAPL34= 170360   390#
MAPL35= 170364   392#
MAPL36= 170370   394#
MAPL37= 170374   396#
MAPL4 = 170220   406#
MAPL5 = 170224   408#
MAPL6 = 170230   410#
MAPL7 = 170234   412#
MEMERR= 177744   158#
MMR0  = 177572   183#  187
MMR1  = 177574   184#  188
MMR2  = 177576   185#  189
MMR3  = 172516   186#  190
MMVEC = 000250   151#
MOVBR0  002126   1048  1049  1050  1051  1057#
MOVR0   002130   1055  1060#  1073
N       000735    637#  657#  663#  683#  689#  708#  714#  733#  739#  761#  778#  784#  802#
                  808#  825#  831#  847#  867#  879#  891#  902#  916#  928#  939#  951#  963#
                  978#  990#  1004#  1017#  1025#  1038#  1058#  1061#  1073#  1086#  1098#  1113#  1126#
                 1138#  1168#  1171#  1176#  1187#  1190#  1195#  1206#  1209#  1214#  1222#  1226#  1230#
                 1234#  1241#  1245#  1257#  1261#  1267#  1271#  1283#  1287#  1293#  1297#  1309#  1313#
                 1319#  1323#  1335#  1339#  1345#  1349#  1361#  1365#  1371#  1375#  1387#  1391#  1397#
                 1401#  1436#  1440#  1445#  1449#  1454#  1458#  1463#  1467#  1472#  1476#  1481#  1485#
                 1549#  1552#  1557#  1585#  1600#  1603#  1607#  1610#  1614#  1663#  1667#  1674#  1678#
                 1685#  1688#  1692#  1697#  1728#  1731#  1746#  1749#  1774#  1780#  1788#  1793#  1802#
                 1809#  1818#  1825#  1834#  1841#  1849#  1852#  1861#  1870#  1879#  1915#  1918#  1922#
                 1930#  1938#  1964#  1973#  2014#  2017#  2026#  2031#  2034#  2045#  2048#  2052#  2083#
                 2087#  2090#  2113#  2132#  2157#  2160#  2201#  2204#  2209#  2226#  2280#  2283#  2287#
                 2298#  2309#  2341#  2345#  2350#  2355#  2378#  2406#  2409#  2455#  2458#  2490#  2495#
                 2543#  2547#  2553#  2558#  2561#  2593#  2596#  2600#  2629#  2633#  2636#  2654#  2658#
                 2677#  2705#  2730#  2734#  2738#  2760#  2801#  2804#  2869#  2873#  2878#  2882#  2888#
                 2895#  2922#  2944#  2948#  2998#  3006#  3010#  3014#  3025#  3047#  3079#  3115#  3121#
                 3124#  3129#  3161#  3165#  3204#  3207#  3243#  3247#  3284#  3287#  3322#  3325#  3329#
                 3333#  3337#  3371#  3374#  3405#  3409#  3446#  3449#  3484#  3487#  3515#  3541#  3561#
                 3576#  3600#  3607#  3612#  3639#  3645#  3679#  3683#  3687#  3694#  3699#  3704#  3708#
                 3712#  3716#  3720#  3723#  3732#  3736#  3767#  3770#  3774#  3777#  3781#
NEGR5   003446   1587  1588  1589  1591#
NEXTT   005102   2239  2244#
OVER    002344   1163  1173#
OVER2   002400   1182  1192#
OVER3   002434   1201  1211#
PIRQ    177772    40#
PIRQVE= 000240   150#
POINT   005634   2582  2589#
POINT1  006350   2830* 2843  2865#
POINT2  007064   3153  3157#  3159
POSERR  003526   1652#  1656  1672  1676
```

```
PR0    = 000000              72#
PR1    - 000040              73#
PR2    - 000100              74#
PR3    - 000140              75#
PR4    = 000200              76#
PR5      000240              77#  3591    3669
PR6    - 000300              78#  3762
PR7      000340              79#  3436    3631    3642    3643    3666
PS       177776              37#    38
PSW    - 177776              38#  1655    2159    2845    2989    3591*   3594    3596    3598    3605    3641    3666*   3692
                           3762*
PWRVEC= 000024              144#
RESVEC= 000010              139#  3670*   3728*
ROLR0    001700             958    959     960     962#
RORR0    001572             911    912     913     915#    928     978
ROUT0    002330            1164#  1168    1171
ROUT1    002364            1183#  1187    1190
ROUT2    002420            1202#  1206    1209
R50K     003510            1595   1612#
SBCR0    001752             985    986     987     989#    1004
SDPAR0= 172260              271#
SDPAR1= 172262              272#
SDPAR2- 172264              273#
SDPAR3- 172266              274#
SDPAR4- 172270              275#
SDPAR5= 172272              276#
SDPAR6= 172274              277#
SDPAR7- 172276              278#
SDPDR0- 172220              249#
SDPDR1= 172222              250#
SDPDR2= 172224              251#
SDPDR3= 172226              252#
SDPDR4- 172230              253#
SDPDR5= 172232              254#
SDPDR6= 172234              255#
SDPDR7= 172236              256#
SEQENC  010470             3766   3784#
SIPAR0= 172240              260#
SIPAR1= 172242              261#
SIPAR2- 172244              262#
SIPAR3= 172246              263#
SIPAR4= 172250              264#
SIPAR5= 172252              265#
SIPAR6= 172254              266#
SIPAR7- 172256              267#
SIPDR0= 172200              238#
SIPDR1= 172202              239#
SIPDR2= 172204              240#
SIPDR3  172206              241#
SIPDR4= 172210              242#
SIPDR5= 172212              243#
SIPDR6= 172214              244#
SIPDR7= 172216              245#
SIZEHI= 177762              169#
SIZELO- 177760              167#
SKIP     005122            2272   2274#
```

```
SR0     = 177572            187#
SR1     = 177574            188#
SR2     = 177576            189#
SR3     = 172516            190#
STACK   = 001100             31#     32      33      34    2270    2405    2452    2520    2649    2834    3630    3667    3756
START     001202            429    618#
STKLMT= 177774               39#
SUBR0     002274           1121   1122    1123   1125#    1138
SUBTAB    012006           1721   4105#
SUPSTK= 000700               33#
SWABR0    002000            999   1000    1001   1003#    1017
SWR     = 177570             41#     42    3784*
SW0     = 000001            107#
SW00    = 000001             97#    107
SW01    = 000002             96#    106
SW02    = 000004             95#    105
SW03    = 000010             94#    104
SW04    = 000020             93#    103
SW05    = 000040             92#    102
SW06    = 000100             91#    101
SW07    = 000200             90#    100
SW08    = 000400             89#     99
SW09    = 001000             88#     98
SW1     - 000002            106#
SW10    = 002000             87#
SW11    - 004000             86#
SW12      010000             85#
SW13    = 020000             84#
SW14    - 040000             83#
SW15    - 100000             82#
SW2     - 000004            105#
SW3     = 000010            104#
SW4     = 000020            103#
SW5     = 000040            102#
SW6     = 000100            101#
SW7     - 000200            100#
SW8       000400             99#
SW9     = 001000             98#
SXTR0     002024           1012   1013    1014   1016#    1025
SXT2      002040           1024#  1038
SYNC24    003332           1530#
SYNC25    003426           1580#
SYNC26    003522           1647#
SYNC27    003652           1715#
SYNC31    004302           1902#
SYNC32    004424           1960#
SYNC33    004502           2004#
SYNC34    004626           2073#
SYNC35    004706           2109#
SYNC36    004730           2127#
SYNC37    004752           2150#
SYNC40    005016           2196#
SYNC41    005052           2221#
SYNC42    005074           2240#
SYNC43    005124           2275#  2293#
SYNC44    005216           2334#
```

```
SYNC45  005300            2374#
SYNC46  005326            2399#
SYNC47  005412            2449#
SYNC50  005456            2483#
SYNC51  005534            2530#
SYNC52  005630            2586#
SYNC53  005704            2623#
SYNC55  005774            2673#
SYNC56  006022            2701#
SYNC57  006056            2726#
SYNC60  006114            2755#
SYNC61  006142            2794#
SYNC62  006342            2847    2860#
SYNC63  006462            2918#
SYNC64  006510            2939#
SYNC65  006570            2991#
SYNC66  006720            3042#
SYNC67  006746            3074#
SYNC70  007002            3108#
SYNC71  007060            3154#
SYNC72  007134            3198#
SYNC73  007174            3236#
SYNC74  007246            3277#
SYNC75  007310            3317#
SYNC76  007372            3364#
SYNC77  007434            3398#
SYN100  007510            3441#
SYN101  007554            3477#
SYN102  007634            3511#
SYN103  007666            3537#
SYN104  007734            3557#
SYN107  010110            3634#
SYSTID= 177764             171#
TBITVE- 000014             140#
TESTCC  003410            1555#
TKVEC = 000060             147#
TPVEC = 000064             148#
TRAPVE= 000034             146#  3727*  3738*  3787*  3806*  3833*
TRTVEC- 000014             141#
TSTR0   001502             874    875    876    878#   891
TST1    001210             629#   637    3851
TST10   001370             782    796#   802
TST100  007456            3402   3434#  3446   3449
TST101  007530            3445   3472#  3484   3487
TST102  007600            3481   3505#  3515
TST103  007644            3514   3533#  3541
TST104  007700            3540   3551#  3561
TST105  007744            3560   3568#  3576
TST106  010006            3575   3590#  3600   3607   3612
TST107  010066            3595   3629#  3639   3645
TST11   001410             806    819#   825
TST110  010140            3644   3665#  3679   3683   3687   3694   3699   3704   3708   3712   3716   3720   3723
                          3732   3736
TST111  010360            3693   3754#  3767   3770   3774   3777   3781
TST12   001430             829    841#   847
TST13   001442             845    859#   867
```

```
TST14   002320          1160#  1176
TST15   002526          1244   1253#  1257   1261   1267   1271   1436   1440   1445   1449   1454   1458   1463
                        1467   1472   1476   1481   1485
TST16   002566          1270   1279#  1283   1287   1293   1297
TST17   002626          1296   1305#  1309   1313   1319   1323
TST2    001226           635    651#   657
TST20   002666          1322   1331#  1335   1339   1345   1349
TST21   002726          1348   1357#  1361   1365   1371   1375
TST22   002766          1374   1383#  1387   1391   1397   1401
TST23   003026          1400   1410#
TST24   003306          1520#  1549   1552   1557
TST25   003414          1556   1575#  1585   1600   1603   1607   1610   1614
TST26   003516          1613   1645#  1663   1667   1674   1678   1685   1688   1692   1697
TST27   003646          1696   1713#  1728   1731   1746   1749
TST3    001246           661    676#   683
TST30   003742          1738   1768#  1774
TST31   004274          1878   1899#  1915   1918   1922   1930   1938
TST32   004416          1957#  1964   1973
TST33   004460          1972   1995#  2014   2017   2026   2031   2034   2045   2048   2052
TST34   004612          2025   2069#  2083   2087   2090
TST35   004670          2077   2104#  2113
TST36   004716          2112   2123#  2132
TST37   004740          2131   2146#  2157   2160
TST4    001270           687    701#   708
TST40   005004          2155   2192#  2201   2204   2209
TST41   005044          2207   2219#  2226
TST42   005066          2225   2238#
TST43   005104          2269#  2280   2283   2287
TST44   005202          2307   2329#  2341   2345   2350   2355
TST45   005262          2354   2369#  2378
TST46   005310          2377   2394#  2406   2409
TST47   005352          2402   2437#  2455   2458
TST5    001312           712    726#   733
TST50   005434          2454   2477#  2490   2495
TST51   005476          2486   2519#  2543   2547   2553   2558   2561
TST52   005612          2534   2580#  2593   2596   2600
TST53   005660          2599   2616#  2629   2633   2636
TST54   005734          2626   2648#  2654   2658
TST55   005766          2657   2671#  2677
TST56   006004          2696#  2705
TST57   006032          2704   2719#  2730   2734   2738
TST6    001334           737    755#   761
TST60   006102          2737   2751#  2760
TST61   006126          2759   2790#  2801   2804
TST62   006166          2798   2827#  2869   2873   2878   2882   2888   2895
TST63   006436          2894   2912#  2922
TST64   006472          2921   2934#  2944   2948
TST65   006532          2947   2981#  2998   3006   3010   3014   3025
TST66   006702          3024   3038#  3047
TST67   006734          3046   3070#  3079
TST7    001350           759    772#   778
TST70   006764          3078   3104#  3115   3121   3124   3129
TST71   007046          3128   3151#  3161   3165
TST72   007104          3158   3190#  3204   3207
TST73   007154          3201   3231#  3243   3247
TST74   007220          3240   3271#  3284   3287
```

```
TST75   007274            3281    3313#   3322    3325    3329    3333    3337
TST76   007352            3336    3359#   3371    3374
TST77   007416            3368    3394#   3405    3409
TYPDS = 104410            4104#
TYPE  = 104400            3792    3807    3814    3842    3895    3990    4066    4100#
TYPOC = 104402            3813    4101#
TYPON = 104406            4103#
TYPOS = 104404            4102#
T67     007322            3315    3327#
T67A    007310            3318#   3327
T67B    007314            3321#   3331
T67C    007320            3321    3324#
T71     010120            3632    3641#
T73     007526            3437    3448#
UDPAR0= 177660             227#
UDPAR1= 177662             228#
UDPAR2= 177664             229#
UDPAR3= 177666             230#
UDPAR4= 177670             231#
UDPAR5= 177672             232#
UDPAR6= 177674             233#
UDPAR7= 177676             234#
UDPDR0= 177620             205#
UDPDR1= 177622             206#
UDPDR2= 177624             207#
UDPDR3= 177626             208#
UDPDR4= 177630             209#
UDPDR5= 177632             210#
UDPDR6= 177634             211#
UDPDR7= 177636             212#
UIPAR0= 177640             216#
UIPAR1= 177642             217#
UIPAR2= 177644             218#
UIPAR3= 177646             219#
UIPAR4= 177650             220#
UIPAR5= 177652             221#
UIPAR6= 177654             222#
UIPAR7= 177656             223#
UIPDR0= 177600             194#
UIPDR1= 177602             195#
UIPDR2= 177604             196#
UIPDR3= 177606             197#
UIPDR4= 177610             198#
UIPDR5= 177612             199#
UIPDR6= 177614             200#
UIPDR7= 177616             201#
USESTK= 000600              34#
XORR0   002064            1033    1034    1035    1037#   1058    1061
Y     = 000767             640#    658#    665#    684#    690#    709#    715#    734#    740#    762#    779#    786#    803#
                           809#    826#    832#    848#    868#   1169#   1172#   1177#   1188#   1191#   1196#   1207#   1210#
                          1215#   1223#   1227#   1231#   1235#   1242#   1246#   1258#   1262#   1268#   1272#   1284#   1288#
                          1294#   1298#   1310#   1314#   1320#   1324#   1336#   1340#   1346#   1350#   1362#   1366#   1372#
                          1376#   1388#   1392#   1398#   1402#   1437#   1441#   1446#   1450#   1455#   1459#   1464#   1468#
                          1473#   1477#   1482#   1486#   1586#   1781#   1789#   1794#   1803#   1810#   1819#   1826#   1835#
                          1842#   1850#   1853#   1862#   1871#   1880#   2310#
$BOADR  001122             477#
```

```
$BDDAT  001126            479#
$CHARC  011276           3905*   3912    3921*   3926#
$CMTAG  001100            465#
$CM1   = 000003           491#    492#    493#    494#
$CM2   = 000006           491#    492#    493#    494#
$CM3   = 000003           489#    491
$CM4   = 000004           494#    495#    496#    497#    498#
$CRLF   001173            499#   3895    3929
$DBLK   011744           4032    4066    4074#
$DOAGN  011044           3838    3844    3850#
$DTBL   011734           4035    4070#
$ENDAD  011034            452    3790    3846#
$ENDCT  011014           3840#
$ENDMG  011050           3842    3852#
$ENULL  011062           3854#
$EOP    010764           3832#
$EOPCT  011006           3798    3803*   3837#   3841
$ERFLG  001103            468#
$ERMAX  001115            474#
$ERPSW  001176            501#   1655*   1680    1682    2159*
$ERRPC  001116            475#
$ERRTB  001202            520#
$ERTTL  001112            472#
$FILLC  001150            487#   3898    3929
$FILLS  001147            486#   3929
$GDADR  001120            476#
$GDDAT  001124            478#
$GET42  011024           3843#
$HD    = 000002             26
$ICNT   001104            469#   3796    3800*   3802*
$ITEMB  001114            473#
$LF     001174            500#   3929
$LPADR  001106            470#
$LPERR  001110            471#
$NULL   001146            485#   3900    3929
$NWTST  000001            619#    621     641#    643     666#    668     691#    693     716#    718     746#    748     763#
                           765     787#    789     810#    812     833#    835     849#    851    1142#   1144    1247#   1249
                          1273#   1275    1299#   1301    1325#   1327    1351#   1353    1377#   1379    1403#   1405    1510#
                          1512    1560#   1562    1617#   1619    1699#   1701    1751#   1753    1881#   1883    1942#   1944
                          1975#   1977    2036#   2038    2092#   2094    2115#   2117    2134#   2136    2171#   2173    2211#
                          2213    2228#   2230    2245#   2247    2312#   2314    2357#   2359    2381#   2383    2419#   2421
                          2460#   2462    2498#   2500    2566#   2568    2603#   2605    2638#   2640    2660#   2662    2682#
                          2684    2707#   2709    2740#   2742    2780#   2782    2806#   2808    2905#   2907    2924#   2926
                          2951#   2953    3027#   3029    3060#   3062    3092#   3094    3140#   3142    3179#   3181    3219#
                          3221    3260#   3262    3297#   3299    3348#   3350    3384#   3386    3422#   3424    3461#   3463
                          3497#   3499    3526#   3528    3544#   3546    3563#   3565    3579#   3581    3614#   3616    3656#
                          3658    3739#   3741
$OCNT   011524           3962*   3991*   4004#
$OMODE  011526           3957*   3961*   3966    3969*   3980*   4006#
$PASS   001100            466#   2846    3784    3834*   3835*   3852
$QUES   001172            498#
$RDCHR- ****** U          4105
$RDDEC= ****** U          4105
$RDLIN= ****** U          4105
$RDOCT= ****** U          4105
$REGAD  001152            489#
```

```
$REG0   001154          491# 2156* 2995* 3811* 3812
$REG1   001156          492#
$REG2   001160          493#
$RTI    005610         2523 2560#
$SAVRE= ****** U       4105
$SETUP= 000024          418# 3790 3820# 3834
$STUP = 177777          418# 3820#
$SVPC = 000204          450#  455
$SWR  = 160000           26#  498  630  652  677  702  727  756  773  797  820  842  860
                        1161 1254 1280 1306 1332 1358 1384 1411 1521 1576 1646 1714 1769
                        1900 1958 1996 2070 2105 2124 2147 2193 2220 2239 2270 2330 2370
                        2395 2438 2478 2520 2581 2617 2649 2672 2697 2720 2752 2791 2828
                        2913 2935 2982 3039 3071 3105 3152 3191 3232 3272 3314 3360 3395
                        3435 3473 3506 3534 3552 3569 3591 3630 3666 3755 3827 3834 3845
                        3851 3852
$TKB    001140          482#
$TKS    001136          481#
$TMP0   001162          494# 3274 3508* 3509 3552 3553* 3569 3570*
$TMP1   001164          495# 1908 1910 1913 2001 2009 2010 2043 2070 2079* 2080 2085 2106
                        2125 2148 2154 2330 2337 2347 2370 2396 2403 2446 2478 2617 2697
                        2720 2752 2791 2835 2913 2935 2983 3003 3023 3039 3073* 3076* 3077
                        3105* 3106* 3107 3152 3192* 3234* 3241 3273* 3282 3361* 3369 3396 3439
                        3473 3474* 3475 3482 3506* 3510 3534* 3535 3556* 3573*
$TMP2   001166          496# 1997 2631 2721 2799 2836 2915 2942 3193 3202 3233 3239 3272
                        3276* 3280 3360 3395* 3397 3403 3438* 3444 3473* 3506 3507* 3536* 3539
                        3554 3555* 3556 3571 3572* 3573
$TMP3   001170          497#
$TN     000112            1#   26  619  630#  635  637  641  652#  657  661  666  677#  683
                         687  691  702#  708  712  716  727#  733  737  746  756#  759  761
                         763  773#  778  782  787  797#  802  806  810  820#  825  829  833
                         842#  845  847  849  860#  867 1142 1161# 1176 1244 1247 1254# 1257
                        1261 1267 1270 1271 1273 1280# 1283 1287 1293 1296 1297 1299 1306#
                        1309 1313 1319 1322 1323 1325 1332# 1335 1339 1345 1348 1349 1351
                        1358# 1361 1365 1371 1374 1375 1377 1384# 1387 1391 1397 1400 1401
                        1403 1411# 1510 1521# 1530 1532 1549 1552 1556 1557 1560 1576# 1580
                        1582 1585 1600 1603 1607 1610 1613 1614 1617 1646# 1647 1649 1663
                        1667 1674 1678 1685 1688 1692 1696 1697 1699 1714# 1715 1716 1728
                        1731 1738 1746 1749 1751 1769# 1774 1878 1881 1900# 1902 1915 1918
                        1922 1930 1938 1942 1958# 1960 1961 1964 1972 1973 1975 1996# 2004
                        2005 2014 2017 2025 2026 2031 2034 2036 2045 2048 2052 2070# 2073
                        2074 2077 2083 2087 2090 2092 2105# 2109 2110 2112 2113 2115 2124#
                        2127 2129 2131 2132 2134 2147# 2150 2151 2155 2157 2160 2171 2193#
                        2196 2197 2201 2204 2207 2209 2211 2220# 2221 2222 2225 2226 2228
                        2239# 2240 2241 2245 2270# 2275 2276 2280 2283 2287 2293 2294 2307
                        2312 2330# 2334 2335 2341 2345 2350 2354 2355 2357 2370# 2374 2375
                        2377 2378 2381 2395# 2399 2400 2402 2406 2409 2419 2438# 2449 2450
                        2454 2455 2458 2460 2478# 2483 2484 2486 2490 2495 2498 2520# 2530
                        2531 2534 2543 2547 2553 2558 2561 2566 2581# 2586 2587 2593 2596
                        2599 2600 2603 2617# 2623 2624 2626 2629 2633 2636 2638 2649# 2654
                        2657 2658 2660 2672# 2673 2674 2677 2682 2697# 2701 2702 2704 2705
                        2707 2720# 2726 2727 2730 2734 2737 2738 2740 2752# 2755 2756 2759
                        2760 2780 2791# 2794 2795 2798 2801 2804 2806 2828# 2860 2862 2869
                        2873 2878 2882 2888 2894 2895 2905 2913# 2918 2919 2921 2922 2924
                        2935# 2939 2940 2944 2947 2948 2951 2982# 2991 2992 2998 3006 3010
                        3014 3024 3025 3027 3039# 3042 3043 3046 3047 3060 3071# 3074 3075
                        3078 3079 3092 3105# 3108 3109 3115 3121 3124 3128 3129 3140 3152#
```

```
                        3154    3155    3158    3161    3165    3179    3191#   3198    3199    3201    3204    3207    3219
                        3232#   3236    3237    3240    3243    3247    3260    3272#   3277    3278    3281    3284    3287
                        3297    3314#   3317    3319    3322    3325    3329    3333    3336    3337    3348    3360#   3364
                        3365    3368    3371    3374    3384    3395#   3398    3399    3402    3405    3409    3422    3435#
                        3441    3442    3445    3446    3449    3461    3473#   3477    3478    3481    3484    3487    3497
                        3506#   3511    3512    3514    3515    3526    3534#   3537    3538    3540    3541    3544    3552#
                        3557    3558    3560    3561    3563    3569#   3575    3576    3579    3591#   3595    3600    3607
                        3612    3614    3630#   3634    3635    3639    3644    3645    3656    3666#   3679    3683    3687
                        3693    3694    3699    3704    3708    3712    3716    3720    3723    3732    3736    3739    3755#
                        3767    3770    3774    3777    3781
$TPB    001144           484#   2832    2833    3918*   3929
$TPFLG  001151           488#   3878    3929
$TPS    001142           483#   2828    2829    2830    2831    3916    3929
$TRAP   011754          3787    3806    3833    4084#
$TRP  = 000012          4091#   4101#   4102#   4103#   4104#   4105#
$TRPAD  011774          4088    4099#
$TSTNM  001102           467#
$TYPBN= ****** U        4105
$TYPDS  011530          4020#   4104
$TYPE   011066          3878#   4091    4100
$TYPEC  011232          3897    3904    3911    3916#   3917
$TYPEX  011300          3922    3924    3927#
$TYPOC  011326          3960#   4101
$TYPON  011342          3959    3962#   4103
$TYPOS  011302          3955#   4102
$$GET4- 000000          3845#
$$TN  - 000105           521#    528#    529#    530#    532#    533#    535#    537#    539#    541#    543#    546#    547#
                         549#    551#    552#    553#    554#    556#    557#    559#    560#    562#    564#    565#    566#
                         568#    570#    572#    574#    578#    579#    581#    583#    585#    587#    589#    591#    594#
                         595#    596#    598#    600#    602#    604#    606#    607#    609#    610#
$$TPS   006422          2831*   2891#
$$TRP - 000002          4090#   4101    4102    4103    4104    4105
$OFILL  011525          3956*   3960*   3970    4005#
.       012010           421#    425     427#    450     451#    453#    455#    463#    501     637     657     663     683
                         689     708     714     733     739     761     778     784     802     808     825     831     847
                         865     867     877     879     889     891     900     902     914     916     926     928     937
                         939     949     951     961     963     976     978     988     990    1002    1004    1015    1017
                        1023    1025    1036    1038    1056    1058    1061    1071    1073    1084    1086    1096    1098
                        1111    1113    1124    1126    1136    1138    1168    1171    1176    1187    1190    1195    1206
                        1209    1214    1222    1226    1230    1234    1241    1245    1257    1261    1267    1271    1283
                        1287    1293    1297    1309    1313    1319    1323    1335    1339    1345    1349    1361    1365
                        1371    1375    1387    1391    1397    1401    1436    1440    1445    1449    1454    1458    1463
                        1467    1472    1476    1481    1485    1549    1552    1557    1585    1590    1600    1603    1607
                        1610    1614    1663    1667    1674    1678    1685    1688    1692    1697    1728    1731    1746
                        1749    1774    1780    1788    1793    1802    1809    1818    1825    1834    1841    1849    1852
                        1861    1870    1879    1915    1918    1922    1930    1938    1964    1973    2014    2017    2026
                        2031    2034    2045    2048    2052    2083    2087    2090    2113    2132    2157    2160    2201
                        2204    2209    2226    2280    2283    2287    2298    2309    2341    2345    2350    2355    2378
                        2406    2409    2455    2458    2490    2495    2543    2547    2553    2558    2561    2593    2596
                        2600    2629    2633    2636    2654    2658    2677    2705    2730    2734    2738    2760    2801
                        2804    2869    2873    2878    2882    2888    2895    2922    2944    2948    2998    3006    3010
                        3014    3025    3047    3079    3115    3121    3124    3129    3161    3165    3204    3207    3243
                        3247    3284    3287    3322    3325    3329    3333    3337    3371    3374    3405    3409    3446
                        3449    3484    3487    3515    3541    3561    3576    3600    3607    3612    3639    3645    3679
                        3683    3687    3694    3699    3704    3708    3712    3716    3720    3723    3732    3736    3767
                        3770    3774    3777    3781    3810#   3852    3855#   3929    4074#   4105
```

```
COMA      1663#
COMAA     2378#
COMB      1666#    1667
COMBB     2490#
COMC      1687#    1688
COMCC     3115#
COMD      1930#
COMDD     3776#    3777
COME      1917#    1918     2047#    2048
COMEE     3769#    3770
COMENT    2560#    2561
COMFF     3780#    3781
COMG      2282#    2283
COMGG     3698#    3699
COMH      2282#    2287
COMHH     3703#    3704
COMI      2341#
COMII     3707#    3708
COMJ      2494#    2495
COMJJ     3711#    3712
COMK      2553#
COMKK     3715#    3716
COML      2676#    2677
COMLL     3719#    3720
COMMEN       1#     417#
COMMM     3722#    3723
COMN      3161#
COMNN     3731#    3732
COMO      3243#
COMP      3405#    3687#
COMQ      3694#
COMX      3607#
COMY      3600#
COMZ      3645#
DOC          1#     528      529      530      532      533      535      537      539      541      543      546      547      549      551
           552      553      554      556      557      559      560      562      564      565      566      568      570      572      574
           578      579      581      583      585      587      589      591      594      595      596      598      600      602      604
           606      607      609      610
DOCEXP       1#     528      529      530      532      533      535      537      539      541      543      546      547      549      551
           552      553      554      556      557      559      560      562      564      565      566      568      570      572      574
           578      579      581      583      585      587      589      591      594      595      596      598      600      602      604
           606      607      609      610
ENDCOM       1#     417#
ERROR       35#
ERRORD       1#     636      656      662      682      688      707      713      732      738      760      777      783      801      807
           824      830      846      866      878      890      901      915      927      938      950      962      977      989     1003
          1016     1024     1037     1057     1060     1072     1085     1097     1112     1125     1137     1168     1170     1176     1187
          1189     1195     1206     1208     1214     1222     1226     1230     1234     1241     1245     1257     1261     1267     1271
          1283     1287     1293     1297     1309     1313     1319     1323     1335     1339     1345     1349     1361     1365     1371
          1375     1387     1391     1397     1401     1436     1440     1445     1449     1454     1458     1463     1467     1472     1476
          1481     1485     1549     1551     1557     1585     1600     1602     1607     1609     1614     1663     1666     1674     1678
          1684     1687     1691     1697     1728     1730     1746     1748     1774     1780     1788     1793     1802     1809     1818
          1825     1834     1841     1849     1852     1861     1870     1879     1915              1921     1930     1938     1964     1973
          2014     2016     2026     2031     2033     2045     2047     2051     2083     2087     2089     2113     2132     2157     2160
          2201     2203     2209     2226     2280     2282     2286     2298     2308     2341     2344     2350     2355     2378     2406
          2408     2455     2457     2490     2494     2543     2547     2553     2557     2560     2593     2595     2600     2629     2633
```

```
                2635    2654    2658    2676    2705    2730    2734    2738    2760    2801    2803    2869    2873    2878    2882
                2888    2895    2922    2944    2948    2998    3006    3010    3014    3025    3047    3079    3115    3121    3123
                3129    3161    3164    3204    3206    3243    3246    3284    3286    3322    3324    3329    3333    3337    3371
                3373    3405    3408    3446    3448    3484    3486    3515    3541    3561    3576    3600    3607    3611    3639
                3645    3679    3683    3687    3694    3698    3703    3707    3711    3715    3719    3722    3731    3735    3767
                3769    3773    3776    3780
ESCAPE            1#    417#
HLT               1#     637     657     663     683     689     708     714     733     739     761     778     784     802     808
                825     831     847     867     879     891     902     916     928     939     951     963     978     990    1004
               1017    1025    1038    1058    1067    1073    1086    1098    1113    1126    1138    1168    1171    1176    1187
               1190    1195    1206    1209    1214    1222    1226    1230    1234    1241    1245    1257    1261    1267    1271
               1283    1287    1293    1297    1309    1313    1319    1323    1335    1339    1345    1349    1361    1365    1371
               1375    1387    1391    1597    1407    1436    1440    1445    1449    1454    1458    1463    1467    1472    1476
               1481    1485    1549    1552    1557    1585    1600    1603    1607    1610    1614    1663    1667    1674    1678
               1685    1688    1692    1697    1728    1731    1746    1749    1774    1780    1788    1793    1802    1809    1818
               1825    1834    1841    1849    1852    1861    1870    1879    1915    1918    1922    1930    1938    1964    1973
               2014    2017    2026    2031    2034    2045    2048    2052    2083    2087    2090    2113    2132    2157    2160
               2201    2204    2209    2226    2280    2283    2287    2298    2309    2341    2345    2350    2355    2378    2406
               2409    2455    2458    2490    2495    2543    2547    2553    2558    2561    2593    2596    2600    2629    2633
               2636    2654    2658    2677    2705    2730    2734    2738    2760    2801    2804    2869    2873    2878    2882
               2888    2895    2922    2944    2948    2998    3006    3010    3014    3025    3047    3079    3115    3121    3124
               3129    3161    3165    3204    3207    3243    3247    3284    3287    3322    3325    3329    3333    3337    3371
               3374    3405    3409    3446    3449    3484    3487    3515    3541    3561    3576    3600    3607    3612    3639
               3645    3679    3683    3687    3694    3699    3704    3708    3712    3716    3720    3723    3732    3736    3767
               3770    3774    3777    3781
IDN            2882#
LABEL             1#    1530    1580    1647    1715    1902    1960    2004    2073    2109    2127    2150    2196    2221    2240
                2275    2293    2334    2374    2399    2449    2483    2530    2586    2623    2673    2701    2726    2755    2794
                2860    2918    2939    2991    3042    3074    3108    3154    3198    3236    3277    3317    3364    3398    3441
                3477    3511    3537    3557    3634
LABEL1            1#    1532    1582    1649    1716    1961    2005    2074    2110    2129    2151    2197    2222    2241    2276
                2294    2335    2375    2400    2450    2484    2531    2587    2624    2674    2702    2727    2756    2795    2862
                2919    2940    2992    3043    3075    3109    3155    3199    3237    3278    3319    3365    3399    3442    3478
                3512    3538    3558    3635
MSGD1           662#    663
MSGD10         1325#   1327
MSGD11         1351#   1353
MSGD12         1377#   1379
MSGD13         1403#   1405
MSGD2           783#    784
MSGD5           849#    851
MSGD6          1141#   1144
MSGD7          1247#   1249
MSGD8          1273#   1275
MSGD9          1299#   1301
MSG1            619#    621
MSG10           833#    835
MSG120         3739#   3741
MSG16          1510#   1512
MSG17          1560#   1562
MSG2            641#    643
MSG20          1617#   1619
MSG21          1699#   1701
MSG21A         1751#   1753
MSG22          1881#   1883
MSG23          1942#   1944
```

```
MSG24     1975#    1977
MSG25     2036#    2038
MSG26     2092#    2094
MSG27     2115#    2117
MSG3       666#     668
MSG30     2134#    2136
MSG31     2171#    2173
MSG32     2211#    2213
MSG33     2245#    2247
MSG34     2312#    2314
MSG35     2357#    2359
MSG36     2381#    2383
MSG37     2419#    2421
MSG4       691#     693
MSG40     2228#    2230
MSG41     2460#    2462
MSG42     2498#    2500
MSG43     2566#    2568
MSG44     2603#    2605
MSG45     2638#    2640
MSG46     2682#    2684
MSG47     2707#    2709
MSG5       716#     718
MSG50     2740#    2742
MSG51     2660#    2662
MSG53     2780#    2782
MSG54     2806#    2808
MSG55     2905#    2907
MSG56     2924#    2926
MSG57     2951#    2953
MSG6       746#     748
MSG60     3027#    3029
MSG61     3060#    3062
MSG62     3092#    3094
MSG63     3140#    3142
MSG64     3179#    3181
MSG65     3219#    3221
MSG66     3260#    3262
MSG67     3297#    3299
MSG7       763#     765
MSG70     3348#    3350
MSG71     3614#    3616
MSG72     3384#    3386
MSG73     3422#    3424
MSG74     3461#    3463
MSG75     3497#    3499
MSG76     3526#    3528
MSG77     3656#    3658
MSG77A    3544#    3546
MSG77B    3563#    3565
MSG77C    3579#    3581
MSG8       787#     789
MSG9       810#     812
MULT         1#     417#
NEWMAC     636#     637
NEWTST       1#     417#  619  641  666  691  716  746  763  787  810  833  849  1142  1247
```

```
                1273    1299    1325    1351    1377    1403    1510    1560    1617    1699    1751    1881    1942    1975    2036
                2092    2115    2134    2171    2211    2228    2245    2312    2357    2381    2419    2460    2498    2566    2603
                2638    2660    2682    2707    2740    2780    2806    2905    2924    2951    3027    3060    3092    3140    3179
                3219    3260    3297    3348    3384    3422    3461    3497    3526    3544    3563    3579    3614    3656    3739
POP               1#    417#    4061
PUSH              1#    417#    4020
SAVEAD            1#
SCOPE            36#
SETTRA         4091#    410*    4102    4103    4104
SETUP             1#    417#
SKIP              1#    417#     635     661     687     712     737     759     782     806     829     845    1213    1244    1270
                1296    1322    1348    1374    1400    1555    1613    1696    1738    1878    1972    2025    2076    2112    2131
                2155    2207    2225    2307    2354    2377    2402    2454    2486    2534    2599    2626    2657    2704    2737
                2759    2798    2894    2921    2947    3024    3046    3078    3128    3158    3201    3240    3281    3336    3368
                3402    3445    3481    3514    3540    3560    3575    3595    3644    3693    3766
SLASH             1#    417#
SPACE           417#
STARS             1#    417#     430     456     504     521     612     619     628     641     650     666     675     691     700
                 716     725     741     746     754     763     771     787     795     810     818     833     840     849     858
                1142    1159    1247    1252    1273    1278    1299    1304    1325    1330    1351    1356    1377    1382    1403
                1409    1499    1510    1519    1537    1541    1560    1574    1617    1644    1699    1712    1751    1767    1782
                1795    1811    1827    1843    1854    1863    1872    1881    1898    1942    1956    1975    1994    1999    2003
                2036    2068    2092    2103    2115    2122    2134    2145    2164    2168    2171    2191    2211    2218    2228
                2237    2245    2268    2312    2328    2357    2368    2381    2393    2413    2416    2419    2436    2460    2476
                2498    2518    2566    2579    2603    2615    2638    2647    2660    2670    2682    2695    2707    2718    2740
                2750    2764    2768    2773    2777    2780    2789    2806    2826    2899    2902    2905    2911    2924    2933
                2951    2980    3027    3037    3052    3056    3060    3069    3085    3088    3092    3103    3134    3137    3140
                3150    3169    3176    3179    3189    3211    3216    3219    3230    3251    3256    3260    3270    3291    3294
                3297    3312    3341    3345    3348    3358    3377    3381    3384    3393    3413    3419    3422    3433    3453
                3457    3461    3471    3491    3494    3497    3504    3519    3523    3526    3532    3544    3550    3563    3567
                3579    3589    3614    3628    3651    3655    3656    3664    3739    3753    3819    3820    3856    3929    4007
                4075
TRMTRP         4091#
TYPBIN            1#    417#
TYPDEC            1#    417#
TYPNAM            1#    417#    3788
TYPNUM            1#    417#
TYPOCS            1#    417#
TYPOCT            1#    417#    3812
TYPTXT            1#    417#    3807    3814
$IUT              1#    1532    1582    1649    1716    1961    2005    2074    2110    2129    2151    2197    2222    2241    2276
                2294    2335    2375    2400    2450    2484    2531    2587    2624    2674    2702    2727    2756    2795    2862
                2919    2940    2992    3043    3075    3109    3155    3199    3237    3278    3319    3365    3399    3442    3478
                3512    3538    3558    3635
$SAVEA            1#
$SYNC             1#    1530    1580    1647    1715    1902    1960    2004    2073    2109    2127    2150    2196    2221    2240
                2275    2293    2334    2374    2399    2449    2483    2530    2586    2623    2673    2701    2726    2755    2794
                2860    2918    2939    2991    3042    3074    3108    3154    3198    3236    3277    3317    3364    3398    3441
                3477    3511    3537    3557    3634
$$CMRE          456#     491     492     493
$$CMTM          456#     494     495     496     497
$$ESCA            1#    417#
$$NEWT            1#    417#     619     641     666     691     716     746     763     787     810     833     849    1142    1247
                1273    1299    1325    1351    1377    1403    1510    1560    1617    1699    1751    1881    1942    1975    2036
                2092    2115    2134    2171    2211    2228    2245    2312    2357    2381    2419    2460    2498    2566    2603
                2638    2660    2682    2707    2740    2780    2806    2905    2924    2951    3027    3060    3092    3140    3179
```

```
            3219    3260    3297    3348    3384    3422    3461    3497    3526    3544    3563    3579    3614    3656    3739
$$SET       4091#   4101    4102    4103    4104
$$SKIP         1#    417#    635     661     687     712     737     759     782     806     829     845    1244    1270    1296
            1322    1348    1374    1400    1556    1613    1696    1738    1878    1972    2025    2077    2112    2131    2155
            2207    2225    2307    2354    2377    2402    2454    2486    2534    2599    2626    2657    2704    2737    2759
            2798    2894    2921    2947    3024    3046    3078    3128    3158    3201    3240    3281    3336    3368    3402
            3445    3481    3514    3540    3560    3575    3595    3644    3693
.EQUAT         1#
.HEADE         1#      16
.KT11          1#
.SETUP         1#     418    3820
.SWRHI         1#
.SWRLO         1#
.$ACT1         1#     430
.$CATC         1#     418
.$CMTA         1#     456
.$DB2D         1#
.$DB2O         1#
.$DIV          1#
.$EOP          1#    3820
.$ERRO         1#
.$ERRT         1#
.$MULT         1#
.$POWE         1#
.$RAND         1#
.$RDDE         1#
.$RDOC         1#
.$READ         1#
.$SAVE         1#
.$SB2D         1#
.$SB2O         1#
.$SCOP         1#
.$SIZE         1#
.$SUPR         1#
.$TRAP         1#    3820#   4075
.$TYPB         1#
.$TYPD         1#    3820#   4007
.$TYPE         1#    3820#   3856
.$TYPO         1#    3820#   3929
.1170          1#      27


. ABS.  012010      000

 ERRORS DETECTED:  0

 CEKBAC,CEKBAC.LST/CRF/SOL=CEKBAC.SML,CEKBAC.P11
 RUN-TIME:  72 86 6 SECONDS
 RUN-TIME RATIO: 386/165=2.3
 CORE USED:  35K  (69 PAGES)
```