IDENTIFICATION

PRODUCT CODE: AC-7962D-MC

PRODUCT NAME: CEKBADO 11/70 CPU #1

DATE CREATED: MAY, 1980

MAINTAINER: DIAGNOSTIC ENGINEERING

CONTENTS

1. ABSTRACT

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

A. BASIC INSTRUCTION TESTS

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

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

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

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

B. ADVANCED INSTRUCTION AND MISCELLANEOUS LOGIC TESTS

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

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

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

C. IMPORTANT NOTE

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

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

2. REQUIREMENTS

2.1 EQUIPMENT

PDP 11/70 CPU WITH OPERATORS CONSOLE
LA30 OR EQUIVALENT TERMINAL
NOTE: THIS DIAGNOSTIC SUPPORTS THE PDP-11/74, AN EXPERIMENTAL, IN-HOUSE PROCESSOR.

2.2 STORAGE

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

2.3 PRELIMINARY PROGRAMS

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

CEKBB REQUIRES THAT CEKBA RUN

3. LOADING PROCEDURE

3.1 METHOD

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

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE 5.1

4.2 STARTING ADDRESS

200

4.3 PROGRAM AND OPERATOR ACTION

A. CEKBA

1. LOAD PROGRAM INTO MEMORY (SEE SECTION 3)
2. LOAD ADDRESS 200
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. CEKBB

5.2.1 SPURIOUS ERROR HANDLER

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

5.2.2 SCOPE

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

5.2.3 ERROR

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

5.2.4 TRAP CATCHER

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

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

5.2.5 TRAP

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

5.2.5.1 TYPE ($TYPE)

ROUTINE TO TYPE AN ASCII STRING ON THE TTY.

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

5.2.5.2 TYPEOC ($TYPOC)

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

5.2.5.3 TYPEDS ($TYPDS)

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

5.2.6 POWER DOWN AND UP

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

5.2.7 MONITOR RESTORE (QUIT)

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

5.2.8 CHECK TEST SEQUENCE (SEQENC)

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

5.2.9 PERIPHERAL DETERMINATOR AND INTERRUPT ENABLE ROUTINES

5.2.9.1 PERIPHERAL DETERMINATOR

THIS ROUTINE IS EXECUTED, IN LINE, BETWEEN TESTS 32 AND 33 OF CEKBB. IT CHECKS THE SYSTEM TO DETERMINE IF ONE OF FOUR PERIPHERALS IS AVAILABLE (RS04, RP04, TM, OR RK05) FOR BUS REQUEST LEVEL 5 TESTING AND IF A LINE CLOCK IS AVAILABLE FOR LEVEL 6 TESTING.
IF A DEVICE IS FOUND, THE ADDRESS OF "INT5SU" (INTERRUPT 5 SUBROUTINE) AND SKW11L/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 AC) 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.

CEKBAD - DOCUMENTATION CHANGES ONLY.

DOCUMENT
****
11/70 CPU #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

;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
327 ;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'

;+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
419 ;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/CLASS STUCK LOW
                         34                 H               GRAB OBD(1) STUCK H OR NGT RACL E71
```

	34	H	GRAD DRMX00 STUCK H OR GRAB E50 BAD	SEQ 0019
	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 U/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	RACE JMP+JSR+SWAB NOT GOING HIGH	

```
75              10          IRCB E63 BAD-R5 CONTAINS 'T67+2'                    SEQ 0020

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    E5?(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 TES: 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     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

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
```

```
                                                    M  2
                P=(A3+B3)*(A2+B2)*(A1+B1)*(A0+B0)
                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*O/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*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

1976 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

2037 TEST 34 THREE MICROSTATES (DAC*DM1*TST.B*DRO(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 OBD(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*DRO(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*DRO(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*DRO(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
```

```
2414    THE LOGICAL SEQUENCE HERE WOULD BE TO TEST THE BIT & CMP INSTRUCTIONS
        BUT NO ADDITIONAL LOGIC IS TESTED BY THEM.

2420    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

2461    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 C0 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

2499    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.

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

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*DRO(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*O/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*DRO(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]*DRO(1) INSTRUCTION FOLLOWED BY A DAC*DM6*[TST.B+BIT.B+CMP.B]*
        DRO(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*SRO(1)
        FOLLOWED BY A BIN*SM4*DM0*DF7*SRO(0)
        FOLLOWED BY A BIN*SM4*DM0*DF7*SRO(1) INSTRUCTION BUT NO ADDITIONAL LOGIC IS TESTED.

2781    TEST 61 FIVE  MICROSTATES (BIN*SM6*DM0*-DF7*SRO(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) SEQ 0032

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 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

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 ..1*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-7,251,162,231,233,311,157

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*0/CLASS)                    SEQ 0036

                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*SR0(0)*DR0(0)*
        P/CLASS FOLLOWED BY A BIN*SM12*DM4*SR0(0)*DR0(1)*[TST.B+BIT.B+CMP.B]
        FOLLOWED BY A BIN*SM12*DM4*SR0(1)*DR0(0)*P/CLASS FOLLOWED BY A
        BIN*SM12*DM4*SR0(1)*DR0(1)*[TST.B+BIT.B+CMP.B] INSTRUCTION, BUT NO
        ADDITIONAL LOGIC IS TESTED.

3423    TEST 100        SEVEN MICROSTATES (BIN*SM12*DM6*0/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*SR0(0)*DR0(1)*[TST.B+BIT.B+CMP.B] AND BIN*SM5*DM0*DF7*SR0(1)
        BUT NOT ADDITIONAL LOGIC IS TESTED.

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

3462    TEST 101        EIGHT MICROSTATES (BIN*SM7*DM0*-DF7*SR0(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 SROM7 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*SR0(0)*DR0(0)*[TST.B+
        BIT.B+CMP.B] BUT NO ADDITIONAL LOGIC IS TESTED.

3498    TEST 102        EIGHT MICROSTATES (BIN*SM12*DM3*SR0(1)*DR0(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
```

```
3652    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 BE
                ;REPLACED WITH SPACES.
                ;CALL:
                ;       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  CEKBAD0 11/70 CPU #1
                                ;*COPYRIGHT (C) 1975,1980
                                ;*DIGITAL EQUIPMENT CORP.
                                ;*MAYNARD, MASS. 01754
                                ;*
                                ;*
                                ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                                ;*PACKAGE (MAINDEC-11-DZQAC-A5).
                                ;*
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)
                        BIT15=  100000
                        BIT14=  40000
                        BIT13=  20000
                        BIT12=  10000
                        BIT11=  4000
                        BIT10=  2000
                        BIT09=  1000
                        BIT08=  400
                        BIT07=  200
                        BIT06=  100
                        BIT05=  40
                        BIT04=  20
                        BIT03=  10
                        BIT02=  4
                        BIT01=  2
                        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
```

```
  250          172224                  SDPDR2= 172224
  251          172226                  SDPDR3= 172226
  252          172230                  SDPDR4= 172230
  253          172232                  SDPDR5= 172232
  254          172234                  SDPDR6= 172234
  255          172236                  SDPDR7= 172236
  256
  257                                  ;*SUPERVISOR ''I'' PAGE ADDRESS REGISTERS
  258
  259          172240                  SIPAR0= 172240
  260          172242                  SIPAR1= 172242
  261          172244                  SIPAR2= 172244
  262          172246                  SIPAR3= 172246
  263          172250                  SIPAR4= 172250
  264          172252                  SIPAR5= 172252
  265          172254                  SIPAR6= 172254
  266          172256                  SIPAR7= 172256
  267
  268                                  ;*SUPERVISOR ''D'' PAGE ADDRESS REGISTERS
  269
  270          172260                  SDPAR0= 172260
  271          172262                  SDPAR1= 172262
  272          172264                  SDPAR2= 172264
  273          172266                  SDPAR3= 172266
  274          172270                  SDPAR4= 172270
  275          172272                  SDPAR5= 172272
  276          172274                  SDPAR6= 172274
  277          172276                  SDPAR7= 172276
  278
  279                                  ;*KERNEL ''I'' PAGE DESCRIPTOR REGISTERS
  280
  281          172300                  KIPDR0= 172300
  282          172302                  KIPDR1= 172302
  283          172304                  KIPDR2= 172304
  284          172306                  KIPDR3= 172306
  285          172310                  KIPDR4= 172310
  286          172312                  KIPDR5= 172312
  287          172314                  KIPDR6= 172314
  288          172316                  KIPDR7= 172316
  289
  290                                  ;*KERNEL ''D'' PAGE DESCRIPTOR REGISTERS
  291
  292          172320                  KDPDR0= 172320
  293          172322                  KDPDR1= 172322
  294          172324                  KDPDR2= 172324
  295          172326                  KDPDR3= 172326
  296          172330                  KDPDR4= 172330
  297          172332                  KDPDR5= 172332
  298          172334                  KDPDR6= 172334
  299          172336                  KDPDR7= 172336
  300
  301                                  ;*KERNEL ''I'' PAGE ADDRESS REGISTERS
  302
  303          172340                  KIPAR0= 172340
  304          172342                  KIPAR1= 172342
  305          172344                  KIPAR2= 172344
```

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

                        ;*KERNEL 'D' PAGE ADDRESS REGISTERS

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

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

```
                        MAPH16 = 170272
                        MAPL17 = 170274
                        MAPH17 = 170276
                        MAPL20 = 170300
                        MAPH20 = 170302
                        MAPL21 = 170304
                        MAPH21 = 170306
                        MAPL22 = 170310
                        MAPH22 = 170312
                        MAPL23 = 170314
                        MAPH23 = 170316
                        MAPL24 = 170320
                        MAPH24 = 170320
                        MAPL25 = 170324
                        MAPH25 = 170326
                        MAPL26 = 170330
                        MAPH26 = 170332
                        MAPL27 = 170334
                        MAPH27 = 170336
                        MAPL30 = 170340
                        MAPH30 = 170342
                        MAPL31 = 170344
                        MAPH31 = 170346
                        MAPL32 = 170350
                        MAPH32 = 170352
                        MAPL33 = 170354
                        MAPH33 = 170356
                        MAPL34 = 170360
                        MAPH34 = 170362
                        MAPL35 = 170364
                        MAPH35 = 170366
                        MAPL36 = 170370
                        MAPH36 = 170372
                        MAPL37 = 170374
                        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         ;;JUMP TO STARTING ADDRESS OF PROGRAM
                                ;;*****************************************************************

                                .SBTTL          ACT11 HOOKS

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

000204                                  $SVPC=.                 ;;SAVE LOCATION COUNTER
000046                                  .=46                    ;;SET LOCATION COUNTER
000046  011014                          .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  ERROR POINTER TABLE

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

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

001202                  $ERRTB:
                        ;;*****************************************************************
                        ;*      FOLLOWING IS AN INDEX OF THE POSSIBLE FAILURES THAT
                        ;*      CAUSE A TRAP TO LOCATIONS 4, 10, 14, OR 24 OR THAT
                        ;*      CAUSE THE PROCESSOR TO HANG (H). NGT=NOT GETTING THRU
                        ;*
                        ;*   TEST NUMBER        EFFECT                  CAUSE
                        ;*
                        ;*      25              10              RACH NEG.B*DM0 NOT GOING HIGH
                        ;*      25              24              RACH A2 RAB00 NOT GOING LOW
                        ;*      25              H               RACH E57(6) NOT GOING LOW
                        ;*                                      OR NGT RACL RADR07
                        ;*      26              H               RACL RADR00 NOT GOING LOW
                        ;*      26              4               EITHER IRCC SM357 STUCK H
                        ;*                                      OR RACL E70 BAD
                        ;*      26              14              IRCC C0 RAB03 NOT GOING H OR
                        ;*                                      RACL RADR03 INPUT STUCK LOW
                        ;*      26              4               SRC CONST ADDED 1 OR 3
                        ;*
                        ;*      27              H               RACK RADR01 NOT GOING LOW
                        ;*
                        ;*      31              10              RACK A0 RAB01 NOT GOING LOW OR
                        ;*                                      NGT RACL RADR01
                        ;*      31              4               RACK BRCAB05 NOT GOING LOW OR
                        ;*                                      NGT PACL RADR05
                        ;*                                      1$ 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 STUKK 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
```

```
559                             ;*      36              10      IR DECODE ROM WORD BAD
560                             ;*
561                             ;*      37              10      RACE E45 BAD
562                             ;*
563                             ;*      40              10      RACE A0 RAB00 DOES NOT GO LOW
564                             ;*      40              10      IRCB(JMP+JSR) IS STUCK LOW
565                             ;*      40              H       IRCB FJ CLASS IS STUCK HIGH
566                             ;*
567                             ;*      41              10      RACE E45 IS BAD
568                             ;*
569                             ;*      42              10      RACE E33 IS BAD
570                             ;*
571                             ;*      43              10      RACH U/CLASS NOT GOING HIGH
572                             ;*                              5$-2 ON TOP OF STACK
573                             ;*      43              10      EITHER RACE E10 BAD OR
574                             ;*                              RACE BIN NOT GOING H
575                             ;*                              5$ ON TOP OF STACK
576                             ;*
577                             ;*      44              10      RACJ AFIR 14(1) NGT RACE E42
578                             ;*      44              10      IRCB E38(6) STUCK HIGH
579                             ;*
580                             ;*      45              H       GRAB OBD(0) STUCK H OR NGT RACK E51
581                             ;*
582                             ;*      46              10      RACE E33 BAD
583                             ;*
584                             ;*      54              10      RACF E3 DOES NOT GO HIGH
585                             ;*
586                             ;*      56              10      PART PCLASS FIELD BAD IN IR DECODE ROM
587                             ;*
588                             ;*      60              10      PART PCLASS FIELD BAD IN IR DECODE ROM
589                             ;*
590                             ;*      62              10      IRCC C0 RAB03 STUCK H OR NGT RACL RADR03
591                             ;*                              OR FORK C MUX INPUT B0 STUCK H
592                             ;*
593                             ;*      65              10      B FORK MUX SELECT STUCK LOW
594                             ;*      65              H       IRCB B0 RAB04 NGT RADR05
595                             ;*      65              H       B FORK MUX INPUT B0 STUCK H OR
596                             ;*                              IRCC B0 RAB00 STUCK H
597                             ;*      65              4       B FORK MUX INPUT B3 OR
598                             ;*                              IRCB B0 RAB00 STUCK L
599                             ;*      65              H       IRCB E46(10) STUCK L-MICRO ADR 170
600                             ;*
601                             ;*      67              10      RACE E35(1) BAD
602                             ;*
603                             ;*      70              10      B FORK MUX STROBE STUCK L (CHIP FAILURE)
604                             ;*
605                             ;*      75              10      RACE JMP+JSR+SWAB NOT GOING HIGH
606                             ;*      75              10      IRCB E63 BAD-R5 CONTAINS 'T67+2'
607                             ;*
608                             ;*      105             4       RACF (HALT:OP CODE 7) DOES NOT GO HIGH
609                             ;*      105             4       RACE E7 BAD-ODD ADR BIT SET IN ERROR REG
610                             ;*
611                             ;*********************************************************************
612                             ;*      THE FOLLOWING FIVE CONDITION CODE AND BRANCH TESTS ARE A
613                             ;*      FUNCTION OF RACK F TRUE 1. SECTION 1 OF EACH TEST IS
614                             ;*      DEPENDENT ON TRUE 1 NOT GOING HIGH, WHILE SECTION 2 IS
```

```
                                ;*      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$:
001260  000000                  HALT                            ;RACF E48 FAILED
                                                                ;FOR LOOPING CHANGE TO 'BR TST3'' (772)
                        ;SECTION 2
001262  102001          2$:     BVC     3$
001264  102401                  BVS     TST4                    ;;GO TO NEXT TEST
001266                  3$:
001266  000000                  HALT                            ;EITHER RACF E48 OR E47(1) FAILED
                                                                ;FOR LOOPING CHANGE TO '8R 2$'' (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$:
001310  000000                  HALT                            ;EITHER RACF E59 OR E47(11) FAILED
                                                                ;FOR LOOPING CHANGE TO '8R 2$'' (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
```

```
 727  001314  000270                    SEN                     ;CC=1000
 728  001316  101402                    BLOS    1$
 729  001320  102401                    BVS     1$
 730  001322  101001                    BHI     2$              ;GO TO NEXT SECTION
 731  001324                    1$:
 732  001324  000000                    HALT                    ;RACF E59 FAILED
 733                                                            ;FOR LOOPING CHANGE TO 'BR TST5'' (772)
 734                            ;SECTION 2
 735  001326  100001            2$:     BPL     3$
 736  001330  100401                    BMI     TST6            ;;GO TO NEXT TEST
 737  001332                    3$:
 738  001332  000000                    HALT                    ;EITHER RACF E59 OR E47(13) FAILED
 739                                                            ;FOR LOOPING CHANGE TO 'BR 2$'' (775)
 740                            ;;*****************************************************************
 741                            ;*      THE FOLLOWING SEVEN TESTS ARE A FUNCTION OF RACF TRUE 2.
 742                            ;*SECTION 1 OF EACH TEST IS DEPENDENT ON TRUE 2 NOT GOING HIGH
 743                            ;*WHILE SECTION 2 IS DEPENDENT ON TRUE 2 GOING HIGH.
 744                            ;*
 745                            ;;*****************************************************************
 746                            ;*TEST 6        BRANCHES THRU FET.13
 747                            ;*
 748                            ;*      THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
 749                            ;*         SECTION 1
 750                            ;*              BLE     E58(2,1)H,L     E59(13,1,2)L,H,H        E48(1,13,2)H,H,L
 751                            ;*              BLT     E59(13,1,2)L,H,H        E48(1,13,2)H,H,L        E58(10,9)L,H
 752                            ;*              BEQ     E58(2,1)H,L     E58(10,9)H,L
 753                            ;;*****************************************************************
 754  001334  000257            TST6:   CCC                     ;CC=0000
 755  001336  003403                    BLE     1$
 756  001340  002402                    BLT     1$
 757  001342  001401                    BEQ     1$
 758  001344  003001                    BGT     TST7            ;;GO TO NEXT TEST
 759  001346                    1$:
 760  001346  000000                    HALT                    ;RACF TRUE 2 WENT HIGH
 761                                                            ;FOR LOOPING CHANGE TO 'BR TST6'' (772)
 762                            ;;*****************************************************************
 763                            ;*TEST 7        BRANCH THRU FET.13 AND FET.12
 764                            ;*
 765                            ;*      THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
 766                            ;*         SECTION 1
 767                            ;*              BEQ     E48(1,13,2)     L,H,H
 768                            ;*         SECTION 2
 769                            ;*              BGT     E48(1,13,2)     H,H,H
 770                            ;;*****************************************************************
 771  001350  000257            TST7:   CCC                     ;CC=0000
 772                            ;SECTION 1
 773  001352  000262                    SEV                     ;CC=0010
 774  001354  001401                    BEQ     1$
 775  001356  001001                    BNE     2$              ;GO TO SECTION 2
 776  001360                    1$:
 777  001360  000000                    HALT                    ;RACF E48 FAILED
 778                                                            ;FOR LOOPING CHANGE TO 'BR TST7'' (773)
 779                            ;SECTION 2
 780  001362  003001            2$:     BGT     3$
 781  001364  003401                    BLE     TST10           ;;GO TO NEXT TEST
 782  001366                    3$:
```

```
 783  001366  000000                    HALT                    ;EITHER RACF TRUE 2 DID NOT GO HIGH
 784                                                              ;OR IT DID NOT GET THRU RACH A2 RAB01
 785                                                              ;FOR LOOPING CHANGE TO 'BR 2$'' (775)
 786                                ;;*************************************************************
 787                                ;*TEST 10       BRANCH THRU FET.13 AND FET.12
 788                                ;*
 789                                ;*       THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
 790                                ;*          SECTION 1
 791                                ;*               BLT     E58(2,1)        L,H
 792                                ;*          SECTION 2
 793                                ;*               BNE     E58(2,1)        H,H
 794                                ;;*************************************************************
 795  001370  000257                TST10:  CCC                     ;CC=0000
 796                                ;SECTION 1
 797  001372  000264                        SEZ                     ;CC=0100
 798  001374  002401                        BLT     1$
 799  001376  002001                        BGE     2$              ;GO TO SECTION 2
 800  001400                        1$:
 801  001400  000000                        HALT                    ;RACF E58 FAILED
 802                                                              ;FOR LOOPING CHANGE TO 'BR TST10'' (773)
 803                                ;SECTION 2
 804  001402  001001                2$:     BNE     3$
 805  001404  001401                        BEQ     TST11           ;;GO TO NEXT TEST
 806  001406                        3$:
 807  001406  000000                        HALT                    ;EITHER RACF E58 OR E46(12) FAILED
 808                                                              ;FOR LOOPING CHANGE TO 'BR 2$'' (775)
 809                                ;;*************************************************************
 810                                ;*TEST 11       BRANCH THRU FET.13 AND FET.12
 811                                ;*
 812                                ;*       THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
 813                                ;*          SECTION 1
 814                                ;*               BEQ     E59(13,1,2)     H,H,L
 815                                ;*          SECTION 2
 816                                ;*               BGE     E59(13,1,2)     H,H,H
 817                                ;;*************************************************************
 818  001410  000257                TST11:  CCC                     ;CC=0000
 819                                ;SECTION 1
 820  001412  000270                        SEN                     ;CC=1000
 821  001414  001401                        BEQ     1$
 822  001416  001001                        BNE     2$              ;GO TO NEXT SECTION
 823  001420                        1$:
 824  001420  000000                        HALT                    ;RACF E59 FAILED
 825                                                              ;FOR LOOPING CHANGE TO 'BR TST11'' (773)
 826                                ;SECTION 2
 827  001422  002001                2$:     BGE     3$
 828  001424  002401                        BLT     TST12           ;;GO TO NEXT TEST
 829  001426                        3$:
 830  001426  000000                        HALT                    ;EITHER RACF E59 OR E46(13) FAILED
 831                                                              ;FOR LOOPING CHANGE TO 'BR 2$'' (775)
 832                                ;;*************************************************************
 833                                ;*TEST 12       BRANCHES THRU FET.13 AND FET.12
 834                                ;*
 835                                ;*       THIS TEST PUTS THE FOLLOWING PATTERNS ON THE GATES OF TRUE 2:
 836                                ;*          SECTION 1
 837                                ;*               BEQ     E58(2,1)H,L     E58(10,9)H,L
 838                                ;*               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
```

```
  895  001532  005500                    ADC     R0              ;R0=000001, CC'S=0000
  896  001534  100403                    BMI     ADCR0
  897  001536  001402                    BEQ     ADCR0
  898  001540  102401                    BVS     ADCR0
  899  001542  103001                    BCC     .+4
  900  001544                  ADCR0:
  901  001544  000000                    HALT                    ;ERROR, INCORRECT CC'S AFTER ADC
  902                                                            ;FOR LOOPING CHANGE TO 'BR COMR0+2'' (770)
  903
  904  001546  005000                    CLR     R0
  905  001550  000261                    SEC
  906  001552  005500                    ADC     R0
  907  001554  000257                    CCC
  908  001556  000270                    SEN                     ;R0=000001, CC'S=1000
  909  001560  006000                    ROR     R0              ;R0=000000, CC'S=0111
  910  001562  100403                    BMI     RORR0
  911  001564  001002                    BNE     RORR0
  912  001566  102001                    BVC     RORR0
  913  001570  103401                    BCS     .+4
  914  001572                  RORR0:
  915  001572  000000                    HALT                    ;ERROR, INCORRECT CC'S AFTER ROR
  916                                                            ;FOR LOOPING CHANGE TO 'BR ADCR0+2'' (765)
  917
  918  001574  005000                    CLR     R0
  919  001576  000277                    SCC
  920  001600  000250                    CLN                     ;R0=000000, CC'S=0111
  921  001602  005300                    DEC     R0              ;R0=177777, CC'S=1001
  922  001604  100003                    BPL     DECR0
  923  001606  001402                    BEQ     DECR0
  924  001610  102401                    BVS     DECR0
  925  001612  103401                    BCS     .+4
  926  001614                  DECR0:
  927  001614  000000                    HALT                    ;ERROR, INCORRECT CC'S AFTER DEC
  928                                                            ;FOR LOOPING CHANGE TO 'BR RORR0+2'' (767)
  929
  930  001616  005000                    CLR     R0
  931  001620  000277                    SCC                     ;R0=000000, CC'S=1111
  932  001622  005200                    INC     R0              ;R0=000001, CC'S=0000
  933  001624  100403                    BMI     INCR0
  934  001626  001402                    BEQ     INCR0
  935  001630  102401                    BVS     INCR0
  936  001632  103401                    BCS     .+4
  937  001634                  INCR0:
  938  001634  000000                    HALT                    ;ERROR, INCORRECT CC'S AFTER INC
  939                                                            ;FOR LOOPING CHANGE TO 'BR DECR0+2'' (770)
  940
  941  001636  005000                    CLR     R0
  942  001640  000277                    SCC
  943  001642  000244                    CLZ                     ;R0=000000, CC'S=1011
  944  001644  006300                    ASL     R0              ;R0=000000, CC'S=0100
  945  001646  100403                    BMI     ASLR0
  946  001650  001002                    BNE     ASLR0
  947  001652  102401                    BVS     ASLR0
  948  001654  103001                    BCC     .+4
  949  001656                  ASLR0:
  950  001656  000000                    HALT                    ;ERROR, INCORRECT CC'S AFTER ASL
```

```
                                                        ;FOR LOOPING 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     ASRR0
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
001774  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
```

```
 1007  002004  005300                    DEC     R0
 1008  002006  000257                    CCC
 1009  002010  000262                    SEV                     ;R0=177777, CC'S=0010
 1010  002012  006700                    SXT     R0              ;R0=000000, CC'S=0100
 1011  002014  100403                    BMI     SXTR0
 1012  002016  001002                    BNE     SXTR0
 1013  002020  102401                    BVS     SXTR0
 1014  002022  103001                    BCC     .+4
 1015  002024                    SXTR0:
 1016  002024  000000                    HALT                    ;ERROR, INCORRECT CC'S AFTER SXT
 1017                                                            ;FOR LOOPING CHANGE TO 'BR SWABR0+2'' (766)
 1018  002026  005000                    CLR     R0
 1019  002030  000270                    SEN                     ;R0=000000, CC'S=1XXX
 1020  002032  006700                    SXT     R0              ;R0=177777, CC'S=1XXXX
 1021  002034  005200                    INC     R0
 1022  002036  001401                    BEQ     .+4
 1023  002040                    SXT2:
 1024  002040  000000                    HALT                    ;SIGN EXTEND FAILED WITH N SET
 1025                                                            ;FOR LOOPING CHANGE TO 'BR SXTR0+2'' (772)
 1026
 1027  002042  005000                    CLR     R0
 1028  002044  005300                    DEC     R0
 1029  002046  000277                    SCC
 1030  002050  000244                    CLZ                     ;R0=177777, CC'S=1011
 1031  002052  074000                    XOR     R0,R0           ;R0=000000, CC'S=0101
 1032  002054  100403                    BMI     XORR0
 1033  002056  001002                    BNE     XORR0
 1034  002060  102401                    BVS     XORR0
 1035  002062  103401                    BCS     .+4
 1036  002064                    XORR0:
 1037  002064  000000                    HALT                    ;ERROR, INCORRECT CC'S AFTER XOR
 1038                                                            ;FOR LOOPING CHANGE TO 'BR SXT2+2'' (766)
 1039
 1040  002066  005000                    CLR     R0
 1041  002070  000261                    SEC
 1042  002072  006000                    ROR     R0
 1043  002074  000300                    SWAB    R0
 1044  002076  000277                    SCC
 1045  002100  000250                    CLN                     ;R0=000200, CC'S=0111
 1046  002102  110000                    MOVB    R0,R0           ;R0=177600, CC'S=1001
 1047  002104  100010                    BPL     MOVBR0
 1048  002106  001407                    BEQ     MOVBR0
 1049  002110  102406                    BVS     MOVBR0
 1050  002112  103005                    BCC     MOVBR0
 1051  002114  000250                    CLN
 1052  002116  000264                    SEZ                     ;R0=177600, CC'S=0100
 1053  002120  005700                    TST     R0              ;           CC'S=1000
 1054  002122  100002                    BPL     MOVR0
 1055  002124  001002                    BNE     .+6
 1056  002126                    MOVBR0:
 1057  002126  000000                    HALT                    ;ERROR, INCORRECT CC'S AFTER MOVB
 1058                                                            ;FOR LOOPING CHANGE TO 'BR XORR0+2'' (757)
 1059  002130                    MOVR0:
 1060  002130  000000                    HALT                    ;ERROR, MOVB DID NOT SIGN EXTEND
 1061                                                            ;FOR LOOPING CHANGE TO 'BR XORR0+2'' (756)
 1062
```

```
1063  002132  005000                    CLR     R0
1064  002134  000277                    SCC
1065  002136  000244                    CLZ                     ;R0=000000, CC'S=1011
1066  002140  030000                    BIT     R0,R0           ;R0=000000, CC'S=0101
1067  002142  100403                    BMI     BITR0
1068  002144  001002                    BNE     BITR0
1069  002146  102401                    BVS     BITR0
1070  002150  103401                    BCS     .+4
1071  002152                    BITR0:
1072  002152  000000                    HALT                    ;ERROR, INCORRECT CC'S AFTER BIT
1073                                                            ;FOR LOOPING CHANGE TO 'BR MOVR0+2' (767)
1074
1075  002154  005000                    CLR     R0
1076  002156  005200                    INC     R0
1077  002160  000277                    SCC
1078  002162  000244                    CLZ                     ;R0=000001, CC'S=1011
1079  002164  040000                    BIC     R0,R0           ;R0=000000, CC'S=0101
1080  002166  100403                    BMI     BICR0
1081  002170  001002                    BNE     BICR0
1082  002172  102401                    BVS     BICR0
1083  002174  103401                    BCS     .+4
1084  002176                    BICR0:
1085  002176  000000                    HALT                    ;ERROR, INCORRECT CC'S AFTER BIC
1086                                                            ;FOR LOOPING CHANGE TO 'BR BITR0+2' (766)
1087
1088  002200  005000                    CLR     R0
1089  002202  005200                    INC     R0
1090  002204  000277                    SCC                     ;R0=000001, CC'S=1111
1091  002206  050000                    BIS     R0,R0           ;R0=000001, CC'S=0001
1092  002210  100403                    BMI     BISR0
1093  002212  001402                    BEQ     BISR0
1094  002214  102401                    BVS     BISR0
1095  002216  103401                    BCS     .+4
1096  002220                    BISR0:
1097  002220  000000                    HALT                    ;ERROR, INCORRECT CC'S AFTER BIS
1098                                                            ;FOR LOOPING CHANGE TO 'BR BICR0+2' (767)
1099
1100  002222  005000                    CLR     R0
1101  002224  000261                    SEC
1102  002226  006000                    ROR     R0
1103  002230  006000                    ROR     R0
1104  002232  000277                    SCC
1105  002234  000252                    +CLN!CLV                ;R0=040000, CC'S=0101
1106  002236  060000                    ADD     R0,R0           ;R0=100000, CC'S=1010
1107  002240  100003                    BPL     ADDR0
1108  002242  001402                    BEQ     ADDR0
1109  002244  102001                    BVC     ADDR0
1110  002246  103001                    BCC     .+4
1111  002250                    ADDR0:
1112  002250  000000                    HALT                    ;ERROR, INCORRECT CC'S AFTER ADD
1113                                                            ;FOR LOOPING CHANGE TO 'BR BISR0+2' (764)
1114
1115  002252  005000                    CLR     R0
1116  002254  005200                    INC     R0
1117  002256  000277                    SCC
1118  002260  000244                    CLZ                     ;R0=000001, CC'S=1011
```

```
 1119  002262  160000                    SUB     R0,R0           ;R0=000000, CC'S=0100
 1120  002264  100403                    BMI     SUBR0
 1121  002266  001002                    BNE     SUBR0
 1122  002270  102401                    BVS     SUBR0
 1123  002272  103001                    BCC     .+4
 1124  002274                    SUBR0:
 1125  002274  000000                    HALT                    ;ERROR, INCORRECT CC'S AFTER SUB
 1126                                                            ;FOR LOOPING CHANGE TO 'BR ADDR0+2'' (766)
 1127
 1128  002276  005000                    CLR     R0
 1129  002300  000277                    SCC
 1130  002302  000244                    CLZ                     ;R0=000000, CC'S=1011
 1131  002304  020000                    CMP     R0,R0           ;R0=000000, CC'S=0100
 1132  002306  100403                    BMI     CMPR0
 1133  002310  001002                    BNE     CMPR0
 1134  002312  102401                    BVS     CMPR0
 1135  002314  103001                    BCC     .+4
 1136  002316                    CMPR0:
 1137  002316  000000                    HALT                    ;ERROR, INCORRECT CC'S AFTER CMP
 1138                                                            ;FOR LOOPING CHANGE TO 'BR SUBR0+2'' (767)
 1139                              ;
 1140                              ;
 1141                              ;;*****************************************************************
 1142                              ;*TEST 14       REGISTER SELECTION TEST
 1143                              ;*
 1144                              ;*      THIS TEST ENSURES THAT THE 6 ADDRESS LINES INTO
 1145                              ;*      THE GENERAL PURPOSE REGISTERS (GPR) ARE NOT STUCK.
 1146                              ;*      THE LABELS OF THE ADDRESS LINES ARE:
 1147                              ;*              GSAX    GENERAL SOURCE ADDRESS LINE
 1148                              ;*              GDAX    GENERAL DESTINATION ADDRESS LINE
 1149                              ;*      WHERE X STANDS FOR LINE 0,1, OR 2.
 1150                              ;*      THE CLASSES OF ERRORS DESCRIBED IN THIS TEST
 1151                              ;*      ARE DEFINED AS FOLLOWS:
 1152                              ;*              CLASS A=GDAX OK
 1153                              ;*                      GSAX STUCK
 1154                              ;*              CLASS B=GSAX OK
 1155                              ;*                      GDAX STUCK
 1156                              ;*              CLASS C=GSAX STUCK
 1157                              ;*                      GDAX STUCK
 1158                              ;;*****************************************************************
 1159  002320  005000              TST14:  CLR     R0
 1160  002322  005201                      INC     R1
 1161  002324  005700                      TST     R0              ;DID INC AFFECT R0?
 1162  002326  001406                      BEQ     OVER            ;BRANCH ON NOT CLASS B OR C
 1163  002330  005000              ROUT0:  CLR     R0
 1164  002332  005201                      INC     R1
 1165  002334  010002                      MOV     R0,R2           ;DID S0 REMAIN 0 ON INC R1?
 1166  002336  001001                      BNE     2$              ;BR IF YES-NOT CLASS B
 1167  002340  000000                      HALT                    ;ERROR, CLASS B FAILURE ON GRA0
 1168                                                            ;FOR LOOPING CHANGE TO 'BR ROUT0'' (773)
 1169  002342                      2$:
 1170  002342  000000                      HALT                    ;ERROR, CLASS C FAILURE ON GRA0
 1171                                                            ;FOR LOOPING CHANGE TO 'BR ROUT0'' (772)
 1172  002344  005201              OVER:   INC     R1              ;INCREMENT S1 AND D1
 1173  002346  010001                      MOV     R0,R1           ;MOVE S0 TO S1 AND D1
 1174  002350  001401                      BEQ     GRA1T           ;BRANCH-GRA0 OK
```

```
1175  002352  000000                    HALT                    ;ERROR, CLASS A FAILURE ON GRA0
1176                                                            ;FOR LOOPING CHANGE TO 'BR TST14'' (762)
1177                                    ;
1178  002354  005000            GRA1T:  CLR     R0
1179  002356  005202                    INC     R2
1180  002360  005700                    TST     R0              ;DID INC AFFECT R2?
1181  002362  001406                    BEQ     OVER2           ;BRANCH ON NOT CLASS B OR C
1182  002364  005000            ROUT1:  CLR     R0
1183  002366  005202                    INC     R2
1184  002370  010001                    MOV     R0,R1           ;DID S0 REMAIN 0 ON INC R2?
1185  002372  001001                    BNE     4$              ;BR IF YES-NOT CLASS B
1186  002374  000000                    HALT                    ;ERROR, CLASS B FAILURE ON GRA1
1187                                                            ;FOR LOOPING CHANGE TO 'BR ROUT1'' (773)
1188  002376                    4$:
1189  002376  000000                    HALT                    ;ERROR, CLASS C FAILURE ON GRA1
1190                                                            ;FOR LOOPING CHANGE TO 'BR ROUT1'' (772)
1191  002400  005202            OVER2:  INC     R2              ;INCREMENT S2 AND D2
1192  002402  010002                    MOV     R0,R2           ;MOVE S0 TO S2 AND D2
1193  002404  001401                    BEQ     GRA2T           ;BRANCH-GRA1 OK
1194  002406  000000                    HALT                    ;ERROR, CLASS A FAILURE ON GRA1
1195                                                            ;FOR LOOPING CHANGE TO 'BR GRA1T'' (762)
1196                                    ;
1197  002410  005000            GRA2T:  CLR     R0
1198  002412  005204                    INC     R4
1199  002414  005700                    TST     R0              ;DID INC AFFECT R4?
1200  002416  001406                    BEQ     OVER3           ;YES, CLASS A
1201  002420  005000            ROUT2:  CLR     R0
1202  002422  005204                    INC     R4
1203  002424  010001                    MOV     R0,R1           ;DID S0 REMAIN 0 AFTER INC R4?
1204  002426  001001                    BNE     6$              ;BR IF YES-NOT CLASS B
1205  002430  000000                    HALT                    ;ERROR, CLASS B FAILURE ON GRA2
1206                                                            ;FOR LOOPING CHANGE TO 'BR ROUT2'' (773)
1207  002432                    6$:
1208  002432  000000                    HALT                    ;ERROR, CLASS C FAILURE ON GRA2
1209                                                            ;FOR LOOPING CHANGE TO 'BR ROUT2'' (772)
1210  002434  005204            OVER3:  INC     R4              ;INCREMENT S4 AND D4
1211  002436  010004                    MOV     R0,R4           ;MOVE S0 TO S4 AND D4
1212  002440  001401                    BEQ     8$              ;;BRANCH I. OK
1213  002442  000000                    HALT                    ;ERROR, CLASS A FAILURE ON GRA2
1214                                                            ;FOR LOOPING CHANGE TO 'BR GRA2T'' (762)
1215  002444  005001            8$:     CLR     R1
1216  002446  005003                    CLR     R3
1217  002450  005005                    CLR     R5
1218  002452  005201                    INC     R1              ;CHANGE R1
1219  002454  005703                    TST     R3              ;DID R3 CHANGE?
1220  002456  001401                    BEQ     1$              ;BRANCH IF NO
1221  002460  000000                    HALT                    ;ADDRESS LINE 0 AND 1 TIED TOGETHER
1222                                                            ;FOR LOOPING CHANGE TO 'BR 8$'' (771)
1223  002462  010303            1$:     MOV     R3,R3           ;CHECK SRC ADDRESS LINES
1224  002464  001401                    BEQ     2$              ;BRANCH IF OK
1225  002466  000000                    HALT                    ;ADDRESS LINE 0 AND 1 TIED TOGETHER(SRC)
1226                                                            ;FOR LOOPING CHANGE TO 'BR 8$'' (766)
1227  002470  005705            2$:     TST     R5              ;DID R5 CHANGE?
1228  002472  001401                    BEQ     3$              ;BRANCH IF NO
1229  002474  000000                    HALT                    ;ADDRESS LINES 0 & 2 TIED TOGETHER(DST)
1230                                                            ;FOR LOOPING CHANGE TO 'BR 8$'' (763)
```

```
1231  002476  010505          3$:     MOV     R5,R5
1232  002500  001401                  BEQ     4$
1233  002502  000000                  HALT                    ;ADDRESS LINES 0 & 2 TIED TOGETHER(SRC)
1234                                                          ;FOR LOOPING CHANGE TO 'BR 8$'' (760)
1235  002504  005002          4$:     CLR     R2
1236  002506  005006                  CLR     R6
1237  002510  005202                  INC     R2
1238  002512  005706                  TST     R6              ;DID R3 CHANGE?
1239  002514  001401                  BEQ     5$              ;BRANCH IF NO
1240  002516  000000                  HALT                    ;ADDRESS LINES 1 & 2 TIED TOGETHER(DST)
1241                                                          ;FOR LOOPING CHANGE TO 'BR 4$'' (772)
1242  002520  010606          5$:     MOV     R6,R6           ;GET R6 SRC
1243  002522  001401                  BEQ     TST15           ;;BRANCH IF SRC OK
1244  002524  000000                  HALT                    ;ADDRESS LINES 1 & 2 TIED TOGETHER(SRC)
1245                                                          ;FOR LOOPING CHANGE TO 'BR 4$'' (767)
1246                              ;;*****************************************************************
1247                              ;*TEST 15       GPR1 STUCK BIT TEST
1248                              ;*
1249                              ;*      LOADS GPR1 WITH ZEROS AND ONES AND COMPARES R1 SOURCE AND
1250                              ;*      DESTINATIONS WITH R0. IF THE COMPARISON FAILS A BIT IS STUCK.
1251                              ;;*****************************************************************
1252  002526  005000          TST15:  CLR     R0
1253  002530  005001                  CLR     R1
1254  002532  020001                  CMP     R0,R1           ;DID R1 DST CLR?
1255  002534  001401                  BEQ     1$              ;BRANCH IF YES
1256  002536  000000                  HALT                    ;ERROR, R1 SOURCE STUCK HIGH
1257                                                          ;FOR LOOPING CHANGE TO 'BR TST15'' (773)
1258  002540  020100          1$:     CMP     R1,R0           ;DID R1 SRC CLR?
1259  002542  001401                  BEQ     2$              ;BRANCH IF YES
1260  002544  000000                  HALT                    ;ERROR, R1 SOURCE STUCK HIGH
1261                                                          ;FOR LOOPING CHANGE TO 'BR TST15'' (770)
1262  002546  005100          2$:     COM     R0
1263  002550  005101                  COM     R1
1264  002552  020001                  CMP     R0,R1           ;DID R1 DST SET TO ALL ONES?
1265  002554  001401                  BEQ     3$              ;BRANCH IF YES
1266  002556  000000                  HALT                    ;ERROR, R1 DST STUCK LOW
1267                                                          ;FOR LOOPING CHANGE TO 'BR TST15'' (763)
1268  002560  020100          3$:     CMP     R1,R0           ;DID R1 SRC SET TO ALL ONES?
1269  002562  001401                  BEQ     TST16           ;;BRANCH IF YES
1270  002564  000000                  HALT                    ;ERROR, R1 SRC STUCK LOW
1271                                                          ;FOR LOOPING CHANGE TO 'BR TST15'' (760)
1272                              ;;*****************************************************************
1273                              ;*TEST 16       GPR2 STUCK BIT TEST
1274                              ;*
1275                              ;*      LOADS GPR2 WITH ZEROS AND ONES AND COMPARES R2 SOURCE AND
1276                              ;*      DESTINATION WITH R0.
1277                              ;;*****************************************************************
1278  002566  005000          TST16:  CLR     R0
1279  002570  005002                  CLR     R2
1280  002572  020200                  CMP     R2,R0           ;DID R2 SRC CLEAR?
1281  002574  001401                  BEQ     1$              ;BRANCH IF YES
1282  002576  000000                  HALT                    ;ERROR, R2 SRC STUCK HIGH
1283                                                          ;FOR LOOPING CHANGE TO 'BR TST16'' (773)
1284  002600  020002          1$:     CMP     R0,R2           ;DID R2 DST CLEAR?
1285  002602  001401                  BEQ     2$              ;BRANCH IF YES
1286  002604  000000                  HALT                    ;ERROR, R2 DST STUCK HIGH
```

```
1287                                                    ;FOR LOOPING CHANGE TO 'BR TST16'' (770)
1288  002606  005100          2$:     COM     R0
1289  002610  005102                  COM     R2
1290  002612  020002                  CMP     R0,R2       ;DID R2 DST SET?
1291  002614  001401                  BEQ     3$          ;BRANCH IF YES
1292  002616  000000                  HALT                ;ERROR, R2 DST STUCK LOW
1293                                                    ;FOR LOOPING CHANGE TO 'BR TST16'' (763)
1294  002620  020200          3$:     CMP     R2,R0       ;DID R2 SRC SET?
1295  002622  001401                  BEQ     TST17       ;;BRANCH IF YES
1296  002624  000000                  HALT                ;ERROR, R2 SRC STUCK LOW
1297                                                    ;FOR LOOPING CHANGE TO 'BR TST16'' (760)
1298                          ;;******************************************************************
1299                          ;*TEST 17        GPR3 STUCK BIT TEST
1300                          ;*
1301                          ;*      LOADS GPR3 WITH ZEROS AND ONES AND COMPARES
1302                          ;*      R3 SOURCE AND DESTINATION WITH R0.
1303                          ;;******************************************************************
1304  002626  005000          TST17:  CLR     R0
1305  002630  005003                  CLR     R3
1306  002632  020300                  CMP     R3,R0       ;DID R3 SRC CLEAR?
1307  002634  001401                  BEQ     1$          ;BRANCH IF YES
1308  002636  000000                  HALT                ;ERROR, R3 SRC STUCK HIGH
1309                                                    ;FOR LOOPING CHANGE TO 'BR TST17' (773)
1310  002640  020003          1$:     CMP     R0,R3       ;DID R3 DST CLEAR?
1311  002642  001401                  BEQ     2$          ;BRANCH IF YES
1312  002644  000000                  HALT                ;ERROR, R3 DST STUCK HIGH
1313                                                    ;FOR LOOPING CHANGE TO 'BR TST17' (770)
1314  002646  005100          2$:     COM     R0
1315  002650  005103                  COM     R3
1316  002652  020300                  CMP     R3,R0       ;DID R3 SRC SET TO ALL ONES?
1317  002654  001401                  BEQ     3$          ;BRANCH IF YES
1318  002656  000000                  HALT                ;ERROR,R3 SRC STUCK LOW
1319                                                    ;FOR LOOPING CHANGE TO 'BR TST17' (763)
1320  002660  020003          3$:     CMP     R0,R3       ;DID R3 DST SET TO ALL ONES?
1321  002662  001401                  BEQ     TST20       ;;BRANCH IF YES
1322  002664  000000                  HALT                ;ERROR, R3 DST STUCK LOW
1323                                                    ;FOR LOOPING CHANGE TO 'BR TST17' (760)
1324                          ;;******************************************************************
1325                          ;*TEST 20        GPR4 STUCK BIT TEST
1326                          ;*
1327                          ;*      LOADS GPR4 WITH ZEROS AND ONES AND COMPARES
1328                          ;*      R4 SOURCE AND DESTINATION WITH R0.
1329                          ;;******************************************************************
1330  002666  005000          TST20:  CLR     R0
1331  002670  005004                  CLR     R4
1332  002672  020400                  CMP     R4,R0       ;DID R4 SRC CLEAR?
1333  002674  001401                  BEQ     1$          ;BRANCH IF YES
1334  002676  000000                  HALT                ;ERROR, R4 SRC STUCK HIGH
1335                                                    ;FOR LOOPING CHANGE TO 'BR TST20' (773)
1336  002700  020004          1$:     CMP     R0,R4       ;DID R4 DST CLEAR?
1337  002702  001401                  BEQ     2$          ;BRANCH IF YES
1338  002704  000000                  HALT                ;ERROR, R4 DST STUCK HIGH
1339                                                    ;FOR LOOPING CHANGE TO 'BR TST20' (770)
1340  002706  005100          2$:     COM     R0
1341  002710  005104                  COM     R4
1342  002712  020004                  CMP     R0,R4       ;DID R4 DST SET?
```

```
1343  002714  001401                    BEQ     3$              ;BRANCH IF YES
1344  002716  000000                    HALT                    ;ERROR, R4 DST STUCK LOW
1345                                                            ;FOR LOOPING CHANGE TO 'BR TST20'' (763)
1346  002720  020400            3$:     CMP     R4,R0           ;DID R4 SRC SET?
1347  002722  001401                    BEQ     TST21           ;;BRANCH IF YES
1348  002724  000000                    HALT                    ;ERROR, R4 SRC STUCK LOW
1349                                                            ;FOR LOOPING CHANGE TO 'BR TST20'' (760)
1350                              ;;*****************************************************************
1351                              ;*TEST 21       GPR5 STUCK BIT TEST
1352                              ;*
1353                              ;*      LOADS R5 WITH ZEROS AND ONES AND COMPARES
1354                              ;*      R5 SOURCE AND DESTINATION WITH R0.
1355                              ;;*****************************************************************
1356  002726  005000            TST21:  CLR     R0
1357  002730  005005                    CLR     R5
1358  002732  020500                    CMP     R5,R0           ;DID R5 SRC CLEAR?
1359  002734  001401                    BEQ     1$              ;BRANCH IF YES
1360  002736  000000                    HALT                    ;ERROR, R5 SRC STUCK HIGH
1361                                                            ;FOR LOOPING CHANGE TO 'BR TST21'' (773)
1362  002740  020005            1$:     CMP     R0,R5           ;DID R5 DST CLEAR?
1363  002742  001401                    BEQ     2$              ;BRANCH IF YES
1364  002744  000000                    HALT                    ;ERROR, R5 DST STUCK HIGH
1365                                                            ;FOR LOOPING CHANGE TO 'BR TST21'' (770)
1366  002746  005100            2$:     COM     R0
1367  002750  005105                    COM     R5
1368  002752  020005                    CMP     R0,R5           ;DID R5 DST SET TO ALL ONES?
1369  002754  001401                    BEQ     3$              ;BRANCH IF YES
1370  002756  000000                    HALT                    ;ERROR, R5 DST STUCK LOW
1371                                                            ;FOR LOOPING CHANGE TO 'BR TST21'' (763)
1372  002760  020500            3$:     CMP     R5,R0           ;DID R5 SRC SET TO ALL ONES?
1373  002762  001401                    BEQ     TST22           ;;BRANCH IF YES
1374  002764  000000                    HALT                    ;ERROR, R5 SRC STUCK LOW
1375                                                            ;FOR LOOPING CHANGE TO 'BR TST21'' (760)
1376                              ;;*****************************************************************
1377                              ;*TEST 22       GPR6 STUCK BIT TEST
1378                              ;*
1379                              ;*      LOADS R6 WITH ZEROS AND ONES AND COMPARES
1380                              ;*      R6 SOURCE AND DESTINATION WITH R0.
1381                              ;;*****************************************************************
1382  002766  005000            TST22:  CLR     R0
1383  002770  005006                    CLR     R6
1384  002772  020006                    CMP     R0,R6           ;DID R6 DST CLEAR?
1385  002774  001401                    BEQ     1$              ;BRANCH IF YES
1386  002776  000000                    HALT                    ;ERROR, R6 DST STUCK HIGH
1387                                                            ;FOR LOOPING CHANGE TO 'BR TST22'' (773)
1388  003000  020600            1$:     CMP     R6,R0           ;DID R6 SRC CLEAR?
1389  003002  001401                    BEQ     2$              ;BRANCH IF YES
1390  003004  000000                    HALT                    ;ERROR, R6 SRC STUCK HIGH
1391                                                            ;FOR LOOPING CHANGE TO 'BR TST22'' (770)
1392  003006  005100            2$:     COM     R0
1393  003010  005106                    COM     R6
1394  003012  020006                    CMP     R0,R6           ;DID R6 DST SET?
1395  003014  001401                    BEQ     3$              ;BRANCH IF YES
1396  003016  000000                    HALT                    ;ERROR, R6 DST STUCK LOW
1397                                                            ;FOR LOOPING CHANGE TO 'BR TST22'' (763)
1398  003020  020600            3$:     CMP     R6,R0           ;DID R6 SRC SET?
```

```
1399  003022  001401                    BEQ     TST23           ;;BRANCH IF YES
1400  003024  000000                    HALT                    ;ERROR,R6 SRC STUCK LOW
1401                                                            ;FOR LOOPING CHANGE TO 'BR TST22'' (760)
1402                              ;;*****************************************************************
1403                              ;*TEST 23       GPR  SHORTED BIT TEST
1404                              ;*
1405                              ;*      TEST IF GPR'S 1 THRU 6 HAVE TWO BITS TIED TOGETHER
1406                              ;*
1407                              ;*      NOTE:   R0 IS CONSIDERED 'HARDCORE''
1408                              ;;*****************************************************************
1409  003026  005000              TST23:  CLR     R0              ;THIS SECTION OF CODE
1410  003030  005200                      INC     R0              ;    *
1411  003032  006100                      ROL     R0              ;    *
1412  003034  006100                      ROL     R0              ;    *
1413  003036  005200                      INC     R0              ;    *
1414  003040  006100                      ROL     R0              ;    *
1415  003042  006100                      ROL     R0              ;    *
1416  003044  005200                      INC     R0              ;    *
1417  003046  006100                      ROL     R0              ;    *
1418  003050  006100                      ROL     R0              ;    *
1419  003052  005200                      INC     R0              ;    *
1420  003054  006100                      ROL     R0              ;    *
1421  003056  006100                      ROL     R0              ;    *
1422  003060  005200                      INC     R0              ;    *
1423  003062  006100                      ROL     R0              ;    *
1424  003064  006100                      ROL     R0              ;    *
1425  003066  005200                      INC     R0              ;    *
1426  003070  006100                      ROL     R0              ;    *
1427  003072  006100                      ROL     R0              ;    *
1428  003074  005200                      INC     R0              ;    *
1429  003076  006100                      ROL     R0              ;    *
1430  003100  006100                      ROL     R0              ;    *
1431  003102  005200                      INC     R0              ;PUTS 52525 IN R0
1432  003104  010001              2$:     MOV     R0,R1           ;PUT R0 SRC IN R1
1433  003106  020001                      CMP     R0,R1           ;IS R1 DST OK?
1434  003110  001401                      BEQ     3$              ;BRANCH IF YES
1435  003112  000000                      HALT                    ;ERROR, R1DST HAS SHORTED BITS
1436                                                              ;FOR LOOPING CHANGE TO 'BR TST15'' (605)
1437  003114  020100              3$:     CMP     R1,R0           ;IS R1 SRC OK?
1438  003116  001401                      BEQ     4$              ;BRANCH IF YES
1439  003120  000000                      HALT                    ;ERROR, R1 SRC HAS SHORTEa BITS
1440                                                              ;FOR LOOPING CHANGE TO 'BR TST15'' (602)
1441  003122  010002              4$:     MOV     R0,R2
1442  003124  020002                      CMP     R0,R2           ;IS R2 DST OK?
1443  003126  001401                      BEQ     5$              ;BRANCH IF YES
1444  003130  000000                      HALT                    ;ERROR, R2 DST HAS SHORTED BITS
1445                                                              ;FOR LOOPING CHANGE TO 'BR TST15'' (576)
1446  003132  020200              5$:     CMP     R2,R0           ;IS R2 SRC OK?
1447  003134  001401                      BEQ     6$              ;BRANCH IF YES
1448  003136  000000                      HALT                    ;ERROR, R2 SRC HAS SHORTED BITS
1449                                                              ;FOR LOOPING CHANGE TO 'BR TST15'' (573)
1450  003140  010003              6$:     MOV     R0,R3           ;
1451  003142  020003                      CMP     R0,R3           ;IS R3 DST OK?
1452  003144  001401                      BEQ     7$              ;BRANCH IF YES
1453  003146  000000                      HALT                    ;ERROR, R3 DST HAS SHORTED BITS
1454                                                              ;FOR LOOPING CHANGE TO 'BR TST15'' (567)
```

```
1455  003150  020300                  7$:   CMP   R3,R0          ;IS R3 SRC OK?
1456  003152  001401                        BEQ   8$            ;BRANCH IF YES
1457  003154  000000                        HALT                ;ERROR R3 SRC HAS SHORTED BITS
1458                                                            ;FOR LOOPING CHANGE TO 'BR TST15'' (564)
1459  003156  010004                  8$:   MOV   R0,R4         ;
1460  003160  020004                        CMP   R0,R4         ;IS R4 DST OK?
1461  003162  001401                        BEQ   9$
1462  003164  000000                        HALT                ;ERROR, R4 DST HAS SHORTED BITS
1463                                                            ;FOR LOOPING CHANGE TO 'BR TST15'' (560)
1464  003166  020400                  9$:   CMP   R4,R0         ;IS R4 SRC OK?
1465  003170  001401                        BEQ   10$           ;BRANCH IF YES
1466  003172  000000                        HALT                ;ERROR, R4 SRC HAS SHORTED BITS
1467                                                            ;FOR LOOPING CHANGE TO 'BR TST15'' (555)
1468  003174  010005                  10$:  MOV   R0,R5
1469  003176  020005                        CMP   R0,R5         ;IS R5 DST OK?
1470  003200  001401                        BEQ   11$           ;BRANCH IF YES
1471  003202  000000                        HALT                ;ERROR, R5 DST HAS SHORTED BITS
1472                                                            ;FOR LOOPING CHANGE TO 'BR TST15'' (551)
1473  003204  020500                  11$:  CMP   R5,R0         ;IS R5 SRC OK?
1474  003206  001401                        BEQ   12$           ;BRANCH IF YES
1475  003210  000000                        HALT                ;ERROR, R5 SRC HAS SHORTED BITS
1476                                                            ;FOR LOOPING CHANGE TO 'BR TST15'' (546)
1477  003212  010006                  12$:  MOV   R0,R6         ;
1478  003214  020006                        CMP   R0,R6         ;IS R6 DST OK?
1479  003216  001401                        BEQ   13$           ;BRANCH IF YES
1480  003220  000000                        HALT                ;ERROR, R6 DST HAS SHORTED BITS
1481                                                            ;FOR LOOPING CHANGE TO 'BR TST15'' (542)
1482  003222  020600                  13$:  CMP   R6,R0         ;IS R6 SRC OK?
1483  003224  001401                        BEQ   14$           ;BRANCH IF YES
1484  003226  000000                        HALT                ;ERROR, R6 SRC HAS SHORTED BITS
1485                                                            ;FOR LOOPING CHANGE TO 'BR TST15'' (537)
1486  003230  005006                  14$:  CLR   SP            ;THIS CODE PUTS
1487  003232  005206                        INC   SP            ;1100 IN THE SP
1488  003234  006106                        ROL   SP
1489  003236  006106                        ROL   SP
1490  003240  006106                        ROL   SP
1491  003242  005206                        INC   SP
1492  003244  006106                        ROL   SP
1493  003246  006106                        ROL   SP
1494  003250  006106                        ROL   SP
1495  003252  006106                        ROL   SP
1496  003254  006106                        ROL   SP
1497  003256  006106                        ROL   SP
1498                                  ;;=====================================================================
1499  003260  005000                        CLR   R0            ;THIS CODE
1500  003262  005200                        INC   R0            ;INITIALIZES
1501  003264  006100                        ROL   R0            ;THE
1502  003266  006100                        ROL   R0            ;TEST
1503  003270  006100                        ROL   R0            ;NUMBER
1504  003272  005200                        INC   R0
1505  003274  006100                        ROL   R0            ;STORAGE
1506  003276  005200                        INC   R0            ;REGISTER
1507  003300  012737  000044  177770        MOV   #44,@#177770  ;SETUP MICROPROCESSOR BREAK REG
1508                                        .SBTTL
1509                                  ;;=====================================================================
1510                                  ;*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
                                        ;;*************************************************************************
003306  005200                  TST24:  INC     R0              ;INCREMENT TEST NUMBER
003310  005005                          CLR     R5              ;ENSURE R5 CLEAR
003312  005205                          INC     R5              ;SET R5 EQUAL
003314  006105                          ROL     R5              ;TO 52
003316  006105                          ROL     R5              ;WHICH
003320  005205                          INC     R5              ;WILL CAUSE
003322  006105                          ROL     R5              ;A JUMP
003324  006105                          ROL     R5              ;TO TESTCC
003326  005205                          INC     R5
003330  006105                          ROL     R5              ;TEST
003332                          SYNC24:
003332  000264                          SEZ                     ;ENSURE Z SET
003334                          IUT24:
003334  060507                          ADD     R5,PC           ;ADD SHOULD SKIP TO TAG TESTCC
                                ;FAILURE- TRY E/CLASS*BRQ*DM0
003336  012737  003366  000014          MOV     #FORKA,@#14     ;SETUP VECTOR FOR RETURN
003344  005005                          CLR     R5              ;ENSURE R5 CLEAR
                                        ;;*************************************************************************
003346  012746  000020                  MOV     #BIT4,-(SP)     ;THIS CODE
003352  012746  003360                  MOV     #1$,-(SP)       ;SETS THE
003356  000006                          RTT                     ;T BIT
                                        ;;*************************************************************************
003360  005205                  1$:     INC     R5              ;TRAP HERE IF FORK A OK
003362  005205                          INC     R5              ;WILL EXECUTE IF FORK A FAILED
003364  000240                          NOP                     ;ALLOW T BIT TRAP IF FORK FAILED
003366  012737  000016  000014  FORKA:  MOV     #16,@#14        ;RESTORE T BIT VECTOR
003374  062706  000004                  ADD     #4,SP           ;RESTORE SP
003400  006005                          ROR     R5              ;IS R5 1 OR 2?
003402  103401                          BCS     1$              ;BRANCH ON 1 (FORK A OK)
003404  000000                          HALT                    ;FORK A FAILURE INTO ROM STATE EXC.90
                                                                ;FOR LOOPING CHANGE TO 'BR TST24+2' (741)
```

```
1550  003406                          1$:
1551  003406  000000                        HALT                    ;EITHER PCB DID NOT LOAD OR RACH DF7 STUCK HIGH
1552                                                                ;FOR LOOPING CHANGE TO 'BR TST24+2' (740)
1553
1554  003410                          TESTCC:
1555  003410  001001                        BNE     TST25           ;;GO TO NEXT TEST IF CCLD$ OK
1556  003412  000000                        HALT                    ;STATE EXC.90 BAD
1557                                                                ;FOR LOOPING CHANGE TO 'BR TST24+2' (736)
1558                                        .SBTTL
1559                                ;;**************************************************************
1560                                ;*TEST 25       TWO   MICROSTATES (NEG*DM0)
1561                                ;*
1562                                ;*      THIS TEST EXECUTES A NEGATE INSTRUCTION WITH DM0.
1563                                ;*
1564                                ;*      IF FORK A FAILS EXECUTION WOULD GO TO EITHER RSD.00, ZAP.00,
1565                                ;*      OR FOP.00.
1566                                ;*      FOP.00 WILL CAUSE THE PROCESSOR TO HANG.
1567                                ;*      RSD.00 WOULD CAUSE A TRAP TO LOCATION 10. THIS WOULD ONLY HAPPEN
1568                                ;*      IF RACH NEG.B*DM0 DID NOT GO HIGH.
1569                                ;*      ZAP.00 WOULD CAUSE A TRAP TO LOCATION 24. THIS WOULD ONLY HAPPEN
1570                                ;*      IF RACK A2 RAB00 DID NOT GO LOW.
1571                                ;*
1572                                ;*      ROM FLOW-301,210
1573                                ;;**************************************************************
1574  003414  005200                TST25:  INC     R0              ;INCREMENT TEST NUMBER
1575  003416  005004                        CLR     R4              ;INITIALIZE CC ERROR RECORD
1576  003420  005005                        CLR     R5              ;SET UP R5
1577  003422  005205                        INC     R5              ;FOR TEST
1578  003424  000257                        CCC                     ;
1579  003426                        SYNC25:
1580  003426  000266                        +SEZ!SEV                ;CC'S=0110
1581  003430                        IUT25:
1582  003430  005405                        NEG     R5              ;EXECUTE NEGATE CC'S=1001
1583  003432  000401                        BR      1$              ;GET OVER ERROR CALL
1584  003434  000000                        HALT                    ;RACH E57(6) DOES NOT GO LOW
1585                                                                ;FOR LOOPING CHANGE TO 'BR TST25' (767)
1586  003436  100003                1$:     BPL     NEGR5
1587  003440  001402                        BEQ     NEGR5
1588  003442  102401                        BVS     NEGR5
1589  003444  103401                        BCS     .+4
1590  003446  005204                NEGR5:  INC     R4              ;ERROR, INCORRECT CC'S AFTER NEG.
1591  003450  005001                        CLR     R1              ;SET R1 TO
1592  003452  005301                        DEC     R1              ;-1 WITHOUT NEGATING
1593  003454  020501                        CMP     R5,R1           ;DID R5 GET -1?
1594  003456  001414                        BEQ     R5OK            ;BRANCH IF R5 OK
1595  003460  005704                        TST     R4              ;IS THERE A CC PROBLEM?
1596  003462  001405                        BEQ     3$              ;BRANCH IF NO
1597  003464  022705  177776                CMP     #177776,R5      ;DID NEG ONE'S COMPLEMENT?
1598  003470  001001                        BNE     2$              ;BRANCH IF NO
1599  003472  000000                        HALT                    ;CC'S BAD AND NEG.90 DID NOT ADD 1
1600                                                                ;FOR LOOPING CHANGE TO 'BR TST25+2' (751)
1601  003474                        2$:
1602  003474  000000                        HALT                    ;CC'S BAD AND R5 BAD
1603                                                                ;FOR LOOPING CHANGE TO 'BR TST25+2' (750)
1604  003476  022705  177776        3$:     CMP     #177776,R5      ;DID NEGATE DO A ONE'S COMPLIMENT?
1605  003502  001001                        BNE     4$              ;BRANCH IF NO
```

```
1606  003504  000000                        HALT                    ;CC'S OK BUT NEG.90 DID NOT ADD 1
1607                                                                  ;FOR LOOPING CHANGE TO 'BR TST25+2'' (744)
1608  003506                          4$:
1609  003506  000000                        HALT                    ;CC'S OK BUT R5 BAD
1610                                                                  ;FOR LOOPIN  CHANGE TO 'BR TST25+2'' (743)
1611  003510  005704                  R5OK:   TST     R4              ;CC PROBLEM?
1612  003512  001401                        BEQ     TST26           ;;GO TO NEXT TEST IF NO
1613  003514  000000                        HALT                    ;NEGATE OK BUT INCORRECT CC'S
1614                                                                  ;FOR LOOPING CHANGE TO 'BR TST25+2'' (740)
1615                                        .SBTTL
1616                                  ;;*****************************************************************
1617                                  ;*TEST 26        THREE MICROSTATES (BIN*SM1*DM0*-DF7*SR0(0)
1618                                  ;*
1619                                  ;*      IF FORK A FAILS EXECUTION WILL GO TO EITHER EXEC.80 OR D12.00.
1620                                  ;*      EXC.80 WILL HANG THE PROCESSOR IN THE PAUSE STATE AT MICRO ADDRESS 343.
1621                                  ;*      THIS WILL ONLY HAPPEN IF RACL RADR00 IS NOT GOING LOW DUE
1622                                  ;*      TO RACF A1 RAB00 (AFIR59(1)*[-BIN+SM01]*U/CLASS).
1623                                  ;*      D12.00 WOULD MOV THE PC TO LOCATION 0.
1624                                  ;*
1625                                  ;*      IF FORK C FAILS EXECUTION WOULD GO FROM S13.10 TO D00.80 OR
1626                                  ;*      D45.01 OR S13.20 OR D12.00 OR JSR.10 OR ASC.80 OR RTI.50 OR ASH.20 OR FOP.50.
1627                                  ;*      D00.80 WOULD SWAP THE BYTES OF THE SOURCE OPERAND BEFORE
1628                                  ;*      PUTTING THEM IN R5.
1629                                  ;*      D45.01 WOULD MOVE THE PC TO LOCATION 0.
1630                                  ;*      S13.20 WILL EXECUTE A SM3 (AND NO AUTO INC) INSTR.  THIS WILL CAUSE
1631                                  ;*      AN ODD ADDRESS TRAP SINCE LOCATION POSERR CONTAINS AN ODD WORD.
1632                                  ;*      JSR.10 WOULD PUSH THE ADDR OF POSERR ONTO THE STACK.
1633                                  ;*      ASC.80 WILL HALT AT 8$.
1634                                  ;*      RTI.50 WILL CAUSE 1004XX TO BE PLACED IN THE PS WORD
1635                                  ;*      AND THE PROCESSOR WILL TRAP TO LOCATION 14.
1636                                  ;*      FOP.50 WILL ?????.
1637                                  ;*      ASH.20 WOULD CAUSE A BAD CC.
1638                                  ;*      IF THE SRC CONST FAILS IN STATE S13.00 AND ADDS 1 OR 3, AN ODD
1639                                  ;*      ADDRESS TRAP WILL OCCUR.
1640                                  ;*      IF THE SRC CONSTANT ADDS 2, THE ERROR AT 6$ WILL REPORT THE FAILURE.
1641                                  ;*
1642                                  ;*      ROM FLOW-21,27,205
1643                                  ;*****************************************************************
1644  003516  005200                  TST26:  INC     R0              ;INCREMENT TEST NUMBER
1645  003520  005005                        CLR     R5              ;SETUP R5
1646  003522                          SYNC26:
1647  003522  000244                        CLZ                     ;ENSURE Z CLEAR TO CATCH A FAILURE TO ASC.80
1648  003524                          IUT26:
1649  003524  011705                        MOV     (PC), R5        ;MOVE 1004XX TO R5(XX MUST BE
1650                                                                  ;ODD TO CATCH A FAILURE TO S13.20)
1651  003526  100443                  POSERR: BMI     7$              ;BRANCH IF CC OK(ENSURE OFFSET IS ODD)
1652  003530  100441                        BMI     6$              ;BRANCH IF SRC CONST ADDED 2
1653                                  ;FAILURE
1654  003532  013767  177776  175436        MOV     @#PSW,$ERPSW    ;SAVE ERROR PSW
1655  003540  022737  003526  000000  1$:   CMP     #POSERR,@#0     ;DID FORK A GO TO D12.00 OR FORK C TO D45.01?
1656  003546  001006                        BNE     3$              ;BRANCH IF NO
1657                                  ;EITHER FORK A OR C FAILED-FIND OUT WHICH ONE
1658  003550  005005                        CLR     R5              ;SETUP R5
1659  003552  012501                        MOV     (R5)+,R1        ;TEST TO SEE IF FORK A OR FORK C FAILED
1660  003554  005705                        TST     R5              ;DID R5 INCREMENT
1661  003556  001401                        BEQ     2$              ;BRANCH IF NO
```

```
1662  003560  000000                        HALT                    ;FORK C FAILED,EITHER IRCC DM0 NOT GETTING
1663                                                                  ;THRU TO RACL RADR07 OR IRCC DM0 IS STUCK HIGH
1664                                                                  ;FOR LOOPING CHANGE TO 'BR TST26+2' (757)
1665  003562                        2$:
1666  003562  000000                        HALT                    ;FORK A FAILED, RACH A1 RAB04 NOT GOING LOW
1667                                                                  ;DUE TO EITHER RACE:BF1=7 OR
1668                                                                  ;BF1=0 OR SM0 STUCK LOW OR RACH E11 BAD
1669                                                                  ;FOR LOOPING CHANGE TO 'BR TST26+2' (756)
1670  003564  000305                3$:     SWAB    R5              ;SETUP R5 TO TEST IF INSTR. WENT THRU D00.80
1671  003566  020537  003526              CMP     R5,@#POSERR     ;DID INSTR GO THRU D00.3?
1672  003572  001001                      BNE     4$              ;BRANCH IF NO
1673  003574  000000                      HALT                    ;IRCC RAB00 NOT GETTING TO RACL RADR00
1674                                                                ;FOR LOOPING CHANGE TO 'BR TST26+2' (751)
1675  003576  026627  000010  003526  4$:     CMP     10(SP),#POSERR  ;DID FORK C GO TO JSR.10?
1676  003604  001001                      BNE     5$              ;BRANCH IF NO
1677  003606  000000                      HALT                    ;INPUT TO IRCC E40 PIN 5 STUCK HIGH
1678                                                                ;FOR LOOPING CHANGE TO 'BR TST26+2' (744)
1679  003610  032767  000017  175360  5$:     BIT     #17,$ERPSW      ;DID ALL CC'S CLEAR?
1680  003616  001404                      BEQ     9$              ;BRANCH IF YES
1681  003620  032767  000020  175350          BIT     #BIT4,$ERPSW    ;DID Z BIT SET?
1682  003626  001401                      BEQ     8$              ;BRANCH IF NO
1683  003630                        9$:
1684  003630  000000                      HALT                    ;BAD CONDITION CODE
1685                                                                ;FOR LOOPING CHANGE TO 'BR TST26+2' (733)
1686  003632                        8$:
1687  003632  000000                      HALT                    ;EITHER INPUT TO C MUX STUCK LOW OR MUX BAD OR
1688                                                                ;;C0 RAB01 STUCK LOW OR RACL RADR01 INPUT STUCK LOW
1689                                                                ;FOR LOOPING CHANGE TO 'BR TST26+2' (732)
1690  003634                        6$:
1691  003634  000000                      HALT                    ;SRC CONST EQUAL TO 2 SHOULD BE 0
1692                                                                ;FOR LOOPING CHANGE TO 'BR TST26+2' (731)
1693  003636  000241                7$:     CLC                     ;ENSURE C CLEAR
1694  003640  006105                      ROL     R5              ;CHECK IF R5 WAS LOADED
1695  003642  103401                      BCS     TST27           ;;GO TO NEXT TEST IF C SET
1696  003644  000000                      HALT                    ;ERROR, R5 DID NOT LOAD
1697                                                                ;FOR LOOPING CHANGE TO 'BR TST26+2' (725)
1698                                ;;*****************************************************************
1699                                ;*TEST 27       THREE MICROSTATES (BIN*SM2*DM0*-DF7*SR0(0)
1700                                ;*
1701                                ;*      THIS TEST IS THE SAME AS THE PREVIOUS TEST EXCEPT FOR THE
1702                                ;*      SM. A WORD AND BYTE INSTRUCTION IS EXECUTED TO VERIFY THAT STATE
1703                                ;*      S13.01 ADDS THE CORRECT SOURCE CONSTANT.
1704                                ;*
1705                                ;*      IF FORK A FAILS EXECUTION WILL GO TO EXC.80.
1706                                ;*      EXC.80 WILL HANG THE PROCESSOR IN THE PAUSE STATE AT MICRO ADDRESS 343.
1707                                ;*      THIS WILL ONLY HAPPEN IF RACL RADR01
1708                                ;*      IS NOT GOING LOW DUE TO RACF A1 RAB01 (AFIR10(1)*U/CLASS).
1709                                ;*
1710                                ;*      ROM FLOW-22,27,205
1711                                ;;*****************************************************************
1712  003646  005200                TST27:  INC     R0              ;INCREMENT TEST NUMBER
1713  003650  005005                        CLR     R5              ;SETUP R5
1714  003652  000240                SYNC27: NOP
1715  003654                        IUT27:
1716  003654  012705                        MOV     (PC)+,R5        ;MOVE 000401 TO R5
1717  003656  000401                ER25:   BR      3$
```

```
1718  003660  000412                    BR      4$
1719                              ;FAILURE-SOURCE CONSTANT FAILED. TRY SM3
1720  003662  012705  011766      3$:   MOV     #SUBTAB,R5      ;GET ADDRESS OF LOCATION THAT CONTAINS ADDR.
1721  003666  010501                    MOV     R5,R1           ;SAVE R5 IN R1
1722  003670  013502                    MOV     @(R5)+,R2       ;EXECUTE AN SM3 INSTRUCTION
1723  003672  005201                    INC     R1              ;ADJUST R1 TO
1724  003674  005201                    INC     R1              ;LOOK LIKE R5
1725  003676  020501                    CMP     R5,R1           ;DID R5 AUTO INCREMENT?
1726  003700  001401                    BEQ     2$              ;BRANCH IF YES
1727  003702  000000                    HALT                    ;SOURCE CONST FAILURE ON IRC
1728                                                            ;FOR LOOPING CHANGE TO 'BR TST27+2' (762)
1729  003704                      2$:
1730  003704  000000                    HALT                    ;SRC CONST FAILURE EITHER ON IRC OR DAP
1731                                                            ;FOR LOOPING CHANGE TO 'BR TST27+2' (761)
1732  003706  010705              4$:   MOV     PC,R5           ;SETUP R5 TO HOLD ADDRESS
1733  003710  010501                    MOV     R5,R1           ;SAVE R5 IN R1
1734  003712  112502                    MOVB    (R5)+,R2        ;TEST TO SEE IF SCR CONST=1 ON BYTE
1735  003714  005201                    INC     R1              ;SETUP R1 TO LOOK LIKE R5
1736  003716  020501                    CMP     R5,R1           ;DID R5 AUTOINCREMENT BY 1?
1737  003720  001410                    BEQ     TST30           ;;BRANCH IF YES
1738                              ;FAILURE-SOURCE CONSTANT FAILED ON BYTE. TRY SM4
1739  003722  010705                    MOV     PC,R5           ;PUT ADDRESS IN R5
1740  003724  010501                    MOV     R5,R1           ;SAVE R5 IN R1
1741  003726  114502                    MOVB    -(R5),R2        ;EXECUTE AN SM4 INSTRUCTION
1742  003730  005301                    DEC     R1              ;ADJUST R1 TO LOOK LIKE R5
1743  003732  020501                    CMP     R5,R1           ;DID R5 AUTO DECREMENT?
1744  003734  001001                    BNE     5$              ;BRANCH IF NO
1745  003736  000000                    HALT                    ;IRCC SRCM2 STUCK HIGH INTO IRC E8
1746                                                            ;FOR LOOPING CHANGE TO 'BR TST27+2' (744)
1747  003740                      5$:
1748  003740  000000                    HALT                    ;BYTE SRC CONST FAILURE EITHER ON IRC OR DAP
1749                                                            ;FOR LOOPING CHANGE TO 'BR TST27+2' (743)
1750                              ;;*****************************************************************
1751                              ;*TEST 30       ALU CARRY FUNCTIONAL TEST
1752                              ;*
1753                              ;*      THIS TEST DOES A COMPLETE CHECK OF THE ALU CARRY FUNCTIONS
1754                              ;*      THE FIRST SECTION ENSURES THAT ALL THE INPUT AND OUTPUT LINES ARE
1755                              ;*      OK AND THE REST OF THE TEST ENSURES THAT THE CARRY LOGIC IS OK.
1756                              ;*      FOLLOWING ARE THE LOGIC EQUATIONS FOR A 74S181 AND 74S182 THAT
1757                              ;*      DICTATED THE PATTERNS USED IN EACH SECTION:
1758                              ;*         74S181
1759                              ;*              G=A3*B3+A2*B2*(A3+B3)+A1*B1*(A2+B2)*(A3+B3)+A0*B0*(A1+B1)*(A2+B2)*(A3+B3
1760                              ;*              P=(A3+B3)*(A2+B2)*(A1+B1)*(A0+B0)
1761                              ;*              COUT=G+P*CIN
1762                              ;*         74S182
1763                              ;*              CX=G0+P0*CIN
1764                              ;*              CY=G1+P1*G0+P1*P0*CIN
1765                              ;*              CZ=G2+P2*G1+P2*P1*G0+P2*P1*P0*CIN
1766                              ;;*****************************************************************
1767  003742  005200              TST30:  INC     R0
1768                              ;SECTION 1-INPUT/OUTPUT BIT TEST
1769  003744  012701  125252            MOV     #125252,R1      ;PUT DATA PATTERN IN R1
1770  003750  062701  052525            ADD     #52525,R1       ;ADD COMPLIMENT PATTERN
1771  003754  005101                    COM     R1              ;MAKE RESULT 0
1772  003756  001401                    BEQ     2$              ;BRANCH IF IT IS ZERO
1773  003760  000000                    HALT                    ;BIT FAILED DURING ADD
```

```
1774                                                    ;FOR LOOPING CHANGE TO 'BR TST30+2'' (771)
1775  003762  012701  052525    2$:     MOV     #52525,R1       ;PUT PATTERN IN R1
1776  003766  062701  125252            ADD     #125252,R1      ;ADD COMPLIMENT PATTERN
1777  003772  005101                     COM     R1              ;MAKE IT ZERO
1778  003774  001401                     BEQ     3$              ;BRANCH IF IT WENT TO ZERO
1779  003776  000000                     HALT                    ;BIT FAILED DURING ADD
1780                                                    ;FOR LOOPING CHANGE TO 'BR 2$'' (771)
1781                                    ;;**************************************************************
1782                                    ;SECTION 2-G=A3*B3
1783  004000  012702  167357    3$:     MOV     #167357,R2      ;PUT COMPLIMENT OF EXPECTED PATTERN IN R2
1784  004004  012701  104210            MOV     #104210,R1      ;PUT PATTERN IN R1
1785  004010  060101                     ADD     R1,R1           ;ADD IT TO ITSELF
1786  004012  103401                     BCS     4$              ;BRANCH IF DAPH COUT15 OK
1787  004014  000000                     HALT                    ;DAPH COUT15 DID NOT GO LOW
1788                                                    ;FOR LOOPING CHANGE TO 'BR 3$'' (771)
1789  004016  050201    4$:     BIS     R2,R1           ;MAKE R1 -1
1790  004020  005101                     COM     R1              ;MAKE IT ZERO
1791  004022  001401                     BEQ     5$              ;BRANCH IF IT IS
1792  004024  000000                     HALT                    ;A G LINE FAILED
1793                                                    ;FOR LOOPING CHANGE TO 'BR 3$'' (765)
1794                                    ;;**************************************************************
1795                                    ;SECTION 3-G=A2*B2*(A3+B3)
1796  004026  012701  146314    5$:     MOV     #146314,R1      ;PUT PATTERN IN R1
1797  004032  062701  042104            ADD     #42104,R1       ;ADD PATTERN
1798  004036  050201                     BIS     R2,R1           ;MAKE R1 -1
1799  004040  005101                     COM     R1              ;MAKE IT ZERO
1800  004042  001401                     BEQ     6$              ;BRANCH IF IT IS ZERO
1801  004044  000000                     HALT                    ;A G LINE FAILED
1802                                                    ;FOR LOOPING CHANGE TO 'BR 5$'' (770)
1803  004046  012701  042104    6$:     MOV     #42104,R1       ;REVERSE INPUTS
1804  004052  062701  146314            ADD     #146314,R1      ;TO ALU
1805  004056  050201                     BIS     R2,R1           ;MAKE R1 -1
1806  004060  005101                     COM     R1              ;MAKE IT ZERO
1807  004062  001401                     BEQ     7$              ;BRANCH IF IT IS
1808  004064  000000                     HALT                    ;A G LINE FAILED
1809                                                    ;FOR LOOPING CHANGE TO 'BR 6$'' (770)
1810                                    ;;**************************************************************
1811                                    ;SECTION 4-G=A1*B1*(A2+B2)*(A3+B3)
1812  004066  012701  167356    7$:     MOV     #167356,R1      ;PUT PATTERN IN R1
1813  004072  062701  021042            ADD     #21042,R1       ;ADD PATTERN
1814  004076  050201                     BIS     R2,R1           ;MAKE R1 -1
1815  004100  005101                     COM     R1              ;MAKE IT ZERO
1816  004102  001401                     BEQ     8$              ;BRANCH IF IT IS
1817  004104  000000                     HALT                    ;A G LINE FAILED
1818                                                    ;FOR LOOPING CHANGE TO 'BR 7$'' (770)
1819  004106  012701  021042    8$:     MOV     #21042,R1       ;REVERSE INPUTS
1820  004112  062701  167356            ADD     #167356,R1      ;TO THE ALU
1821  004116  050201                     BIS     R2,R1           ;MAKE R1 -1
1822  004120  005101                     COM     R1              ;MAKE IT ZERO
1823  004122  001401                     BEQ     9$              ;BRANCH IF IT IS
1824  004124  000000                     HALT                    ;A G LINE FAILED
1825                                                    ;FOR LOOPING CHANGE TO 'BR 8$'' (770)
1826                                    ;;**************************************************************
1827                                    ;SECTION 5-G=A0*B0*(A1+B1)*(A2+B2)*(A3+B3)
1828  004126  012701  177777    9$:     MOV     #-1,R1          ;PUT PATTERN IN R1
1829  004132  062701  010421            ADD     #10421,R1       ;ADD PATTERN TO R1
```

```
1830  004136  050201              BIS     R2,R1           ;MAKE R1 -1
1831  004140  005101              COM     R1              ;MAKE IT ZERO
1832  004142  001401              BEQ     10$             ;BRANCH IF IT IS
1833  004144  000000              HALT                    ;A G LINE FAILED
1834                                                      ;FOR LOOPING CHANGE TO 'BR 9$'' (770)
1835  004146  012701  010421  10$:  MOV     #10421,R1       ;REVERSE INPUTS
1836  004152  062701  177777      ADD     #-1,R1          ;TO THE ALU
1837  004156  050201              BIS     R2,R1           ;MAKE R1 -1
1838  004160  005101              COM     R1              ;MAKE IT ZERO
1839  004162  001401              BEQ     11$             ;BRANCH IF IT IS
1840  004164  000000              HALT                    ;A G LINE FAILED
1841                                                      ;FOR LOOPING CHANGE TO 'BR 10$'' (770)
1842                              ;;*********************************************************************
1843                              ;SECTION 6-P OUTPUTS AND CX=P0*CIN, CY=P1*P0*CIN, CZ=P2*P1*P0*CIN
1844  004166  012701  177777  11$:  MOV     #-1,R1          ;PUT PATTERN IN R1
1845  004172  000261              SEC                     ;SET C
1846  004174  005501              ADC     R1              ;CAUSES CARRY TO GO ALL THE WAY
1847  004176  103401              BCS     12$             ;BRANCH IF CARRY CAME OUT
1848  004200  000000              HALT                    ;DAPH COUT15 DID NOT GO LOW
1849                                                      ;FOR LOOPING CHANGE TO 'BR 11$'' (772)
1850  004202  001401  12$:  BEQ     13$             ;BRANCH IF R1 WENT TO ZERO
1851  004204  000000              HALT                    ;EITHER A P LINE OR THE 74S182 FAILED
1852                                                      ;FOR LOOPING CHANGE TO 'BR 11$'' (770)
1853                              ;;*********************************************************************
1854                              ;SECTION 7-CY=P1*G0
1855  004206  012701  000370  13$:  MOV     #370,R1         ;PUT PATTERN IN R1
1856  004212  062701  000010      ADD     #10,R1          ;ADD DATA TO R1
1857  004216  052701  177377      BIS     #177377,R1      ;MAKE R1 -1
1858  004222  005101              COM     R1              ;MAKE IT ZERO
1859  004224  001401              BEQ     14$             ;BRANCH IF IT WORKED
1860  004226  000000              HALT                    ;DAPF E44 FAILED
1861                                                      ;FOR LOOPING CHANGE TO 'BR 13$'' (767)
1862                              ;;*********************************************************************
1863                              ;SECTION 8-CZ=P2*G1
1864  004230  012701  007600  14$:  MOV     #7600,R1        ;PUT DATA IN R1
1865  004234  062701  000200      ADD     #200,R1         ;ADD DATA TO R1
1866  004240  052701  167777      BIS     #167777,R1      ;MAKE R1 -1
1867  004244  005101              COM     R1              ;MAKE IT ZERO
1868  004246  001401              BEQ     15$             ;BRANCH IF IT WORKED
1869  004250  000000              HALT                    ;DAPF E44 FAILED
1870                                                      ;FOR LOOPING CHANGE TO 'BR 14$'' (767)
1871                              ;;*********************************************************************
1872                              ;SECTION 9-CZ=P2*P1*G0
1873  004252  012701  007770  15$:  MOV     #7770,R1        ;PUT DATA IN R1
1874  004256  062701  000010      ADD     #10,R1          ;ADD DATA TO R1
1875  004262  052701  167777      BIS     #167777,R1      ;MAKE R1 -1
1876  004266  005101              COM     R1              ;MAKE IT ZERO
1877  004270  001401              BEQ     TST31           ;;BRANCH IF IT WORKED
1878  004272  000000              HALT                    ;DAPF E44 FAILED
1879                                                      ;FOR LOOPING CHANGE TO 'BR 15$'' (767)
1880                              ;;*********************************************************************
1881                              ;*TEST 31        THREE MICROSTATES (DAC*DM2*0/CLASS)
1882                              ;*
1883                              ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00. THIS WILL ONLY HAPPEN
1884                              ;*      IF RACE A0 RAB01 DOES NOT GO LOW OR DOES NOT GET THRU
1885                              ;*      TO RACL RADR01.  THIS WILL CAUSE A TRAP TO LOCATION 10.
```

```
1886                                    ;*
1887                                    ;*      IF BEN15 FAILS EXECUTION WILL GO TO D45.80.
1888                                    ;*      THIS WILL CAUSE A TRAP TO 4 WITH THE ADDRESS OF 1$ ON THE STACK.
1889                                    ;*
1890                                    ;*      IF THE DESTINATION CONSTANT FAILS(ADDS 1 OR 3) IN STATE D12.60
1891                                    ;*      AN ODD ADDRESS TRAP WILL OCCUR.
1892                                    ;*      IF THE DST CONST ADDS 0, THE ERROR AT 3$ WILL REPORT THE FAILURE.
1893                                    ;*      IF THE DESTINATION IS NOT LOADED WITH THE SOURCE, THE
1894                                    ;*      ERROR AFTER 5$ WILL REPORT THE FAILURE.
1895                                    ;*
1896                                    ;*      ROM FLOW-2,155,312
1897                                    ;;*****************************************************************
1898  004274  005200                   TST31:  INC     R0              ;INCREMENT TEST NUMBER
1899  004276  012705                           MOV     (PC)+,R5        ;PUT 'BR 4$'' IN R5
1900  004300  000401                           .WORD   000401          ;CONTAINS BINARY OF 'BR 4$'
1901  004302  000240                   SYNC31: NOP
1902  004304                           IUT:
1903  004304  010527                   1$:     MOV     R5,(PC)+        ;EXECUTE INSTRUCTION UNDER TEST
1904  004306  000401                           BR      4$              ;WILL EXECUTE THIS IF INSTR FAILS TO AUTO INC
1905  004310  000415                           BR      5$              ;AUTO INC OK, GO CHECK IF LOAD OK
1906                                   ;FAILURE-DESTINATION CONSTANT FAILED. TRY DM4.
1907  004312  012705  001164           4$:     MOV     #$TMP1,R5       ;PUT ADDRESS IN R5
1908  004316  010125                           MOV     R1,(R5)+        ;EXECUTE INSTRUCTION UNDER TEST
1909  004320  022705  001164                   CMP     #$TMP1,R5       ;DID R5 STAY THE SAME?
1910  004324  001006                           BNE     2$              ;BRANCH IF NO
1911  004326  010145                           MOV     R1,-(R5)        ;SEE IF AUTO DEC. WORKS
1912  004330  022705  001162                   CMP     #$TMP1-2,R5     ;DID R5 AUTO DEC?
1913  004334  001001                           BNE     3$              ;BRANCH IF NO
1914  004336  000000                           HALT                    ;AUTO DEC WORKS SO ROM STATE D12.60 PROBABLY BAD
1915                                                                   ;FOR LOOPING CHANGE TO 'BR TST31+2' (757)
1916  004340                           3$:
1917  004340  000000                           HALT                    ;IRCD DSTCON=2 EITHER STUCK LOW OR
1918                                                                   ;NOT GETTING THRU KOMUX
1919                                                                   ;FOR LOOPING CHANGE TO 'BR TST31+2' (756)
1920  004342                           2$:
1921  004342  000000                           HALT                    ;BEN15 OK & DST CONST OK BUT INSTR DOSEN'T WORK
1922                                                                   ;FOR LOOPING CHANGE TO 'BR TST31+2' (755)
1923  004344  012705  004304           5$:     MOV     #1$,R5          ;GET ADDR OF INSTR UNDER TEST
1924  004350  011505                           MOV     (R5),R5         ;GET INSTR UNDER TEST
1925  004352  022705  000401                   CMP     #401,R5         ;DID AUTO DEC OCCUR?
1926  004356  001005                           BNE     8$              ;BRANCH IF NO
1927  004360  012705  010027                   MOV     #10027,R5       ;GET OP CODE OF INSTR UNDER TEST
1928  004364  010567  177714                   MOV     R5,1$           ;RESTORE INSTR UNDER TEST
1929  004370  000000                           HALT                    ;EITHER RACK BRCABO5 NOT GOING LOW OR
1930                                                                   ;IT IS NOT GETTING THRU RACL RADR05
1931                                                                   ;FOR LOOPING CHANGE TO 'BR TST31+2' (742)
1932  004372  010527                   8$:     MOV     R5,(PC)+        ;TEST TO SEE IF (PC)+ WAS LOADED
1933  004374  100000                   7$:     .WORD   100000          ;STORAGE LOCATION FOR PREVIOUS INSTR.
1934  004376  012701  004374                   MOV     #7$,R1          ;PUT ADDRESS OF 7$ IN R1
1935  004402  011102                           MOV     (R1),R2         ;GET CONTENTS OF 7$
1936  004404  100001                           BPL     6$              ;BRANCH IF LOAD OK
1937  004406  000000                           HALT                    ;ERROR, (PC)+ DID NOT LOAD CORRECTLY
1938                                                                   ;FOR LOOPING CHANGE TO 'BR TST31+2' (733)
1939  004410  012702  100000           6$:     MOV     #BIT15,R2       ;SET SIGN BIT IN R2
1940  004414  010221                           MOV     R2,(R1)+        ;RESTORE 7$
1941                                   ;;*****************************************************************
```

```
 1942                                   ;*TEST 32       THREE MICROSTATES (DAC*DM1*0/CLASS)
 1943                                   ;*
 1944                                   ;*      THIS TEST IS THE SAME AS THE PREVIOUS TEST EXCEPT FOR THE DM.
 1945                                   ;*      IF FORK A FAILS, EXECUTION WILL GO TO RSD.00.
 1946                                   ;*      THIS WILL CAUSE A TRAP TO LOCATION 10 WITH AN ODD ADDRESS ERROR.
 1947                                   ;*      THIS WILL ONLY HAPPEN IF RACE A0 RAB00 DOES NOT GO LOW OR
 1948                                   ;*      DOES NOT GET TO RACL.
 1949                                   ;*      EITHER E44 OR E6 IS BAD.
 1950                                   ;*
 1951                                   ;*      IF THE INSTRUCTION FAILS TO MOVE R5 TO THE PC INDIRECT,
 1952                                   ;*      THE ERROR AFTER 1$ WILL REPORT THE FAILURE.
 1953                                   ;*
 1954                                   ;*      ROM FLOW-1,155,312
 1955                                   ;;*****************************************************************
 1956  004416  005200                  TST32:  INC     R0              ;INCREMENT TEST NUMBER
 1957  004420  012705                          MOV     (PC)+,R5        ;PUT BR IN R5
 1958  004422  000401                          .WORD   000401          ;BINARY WORD FOR BR .+2
 1959  004424  000240                  SYNC32: NOP
 1960  004426                          IUT32:
 1961  004426  010517                          MOV     R5,(PC)         ;EXECUTE INSTRUCTION UNDER TST
 1962  004430  000240                  1$:     .WORD   240             ;SHOULD REPLACE THIS WITH BR 2$
 1963  004432  000000                          HALT                    ;LOCATION POINTED TO BY PC DID NOT LOAD
 1964                                                                  ;FOR LOOPING CHANGE TO 'BR TST32+2' (772)
 1965  004434  012705  000240          2$:     MOV     #240,R5         ;THIS CODE
 1966  004440  012701  004430                  MOV     #1$,R1          ;RESTORES THE NOP
 1967  004444  010511                          MOV     R5,(R1)         ;AT LOCATION 1$
 1968  004446  000264                          SEZ                     ;ENSURE Z SET
 1969  004450  010517                          MOV     R5,(PC)         ;EXECUTE INSTRUCTION UNDER TEST
 1970  004452  000000                          .WORD   0
 1971  004454  001001                          BNE     TST33           ;;CC OK, GO TO NEXT TEST
 1972  004456  000000                          HALT                    ;STATE D12.20 BAD
 1973                                                                  ;FOR LOOPING CHANGE TO 'BR TST32+2' (760)
 1974                                   ;;*****************************************************************
 1975                                   ;*TEST 33       THREE MICROSTATES (DAC*DM4*0/CLASS)
 1976                                   ;*
 1977                                   ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00.
 1978                                   ;*      THIS WILL CAUSE A TRAP TO LOCATION 10. THIS WILL ONLY HAPPEN IF
 1979                                   ;*      RACE A0 RAB02 IS NOT GOING LOW. EITHER RACE E33 OR
 1980                                   ;*      E6(9&8) IS BAD.
 1981                                   ;*
 1982                                   ;*      IF THE DST CONST FAILS TO SUBTRACT 2, THE ERROR AT EITHER 1$
 1983                                   ;*      OR 1$-2 WILL REPORT THE FAILURE.
 1984                                   ;*
 1985                                   ;*      IF BEN01 FAILS (CAUSED BY IRCD DM357 STUCK HIGH)
 1986                                   ;*      EXECUTION WILL GO TO D10.00 WHICH WILL EXECUTE A MODE 5
 1987                                   ;*      INSTEAD OF MODE 4.
 1988                                   ;*
 1989                                   ;*      IF THE DESTINATION IS NOT LOADED PROPERLY,
 1990                                   ;*      THE ERROR AT 3$ WILL REPORT THE FAILURE.
 1991                                   ;*
 1992                                   ;*      ROM FLOW-4,122,157
 1993                                   ;;*****************************************************************
 1994  004460  005200                  TST33:  INC     R0              ;INCREMENT TEST NUMBER
 1995  004462  012702  000001                  MOV     #1,R2           ;SETUP R2
 1996  004466  012705  001166                  MOV     #$TMP2,R5       ;PUT ADDRESS OF $TMP2 IN R5
 1997  004472  010501                          MOV     R5,R1           ;SAVE R5
```

```
1998                                    ;;*****************************************************************
1999                                    ;THESE THREE INSTRUCTIONS ARE USED TO CATCH IRCD DM357 STUCK HIGH
2000  004474  012703  001164                  MOV     #$TMP1,R3       ;PUT ADDR OF $TMP1 IN R3
2001  004500  010513                        MOV     R5,(R3)         ;PUT ADDR. OF $TMP2 IN $TMP1
2002                                    ;;*****************************************************************
2003  004502  000240                    SYNC33: NOP
2004  004504                            IUT33:
2005  004504  010245                          MOV     R2,-(R5)        ;EXECUTE INSTRUCTION UNDER TEST
2006  004506  020503                          CMP     R5,R3           ;DID R5 AUTO DECREMENT?
2007  004510  001411                          BEQ     4$              ;BRANCH IF YES.
2008  004512  012705  001166                  MOV     #$TMP1+2,R5     ;PUT ADDR. IN R5
2009  004516  012701  001164                  MOV     #$TMP1,R1       ;PUT ADDR IN R1
2010  004522  011145                          MOV     (R1),-(R5)      ;EXECUTE INSTRUCTION
2011  004524  020105                          CMP     R1,R5           ;DID R5 AUTO DECREMENT?
2012  004526  001401                          BEQ     1$              ;BRANCH IF YES
2013  004530  000000                          HALT                    ;IRCC DSTM4 STUCK HIGH
2014                                                                  ;FOR LOOPING CHANGE TO 'BR TST33+2' (754)
2015  004532                            1$:
2016  004532  000000                          HALT                    ;IRCC DSTM4 OK BUT D45.00 DOES NOT DECREMENT
2017                                                                  ;FOR LOOPING CHANGE TO 'BR TST33+2' (753)
2018  004534  011505                    4$:   MOV     (R5),R5         ;GET CONTENTS OF $TMP1
2019  004536  020205                          CMP     R2,R5           ;DID DEST. GET LOADED?
2020  004540  001005                          BNE     2$              ;BRANCH IF NO
2021  004542  005205                          INC     R5              ;ADJUST R5 TO EVEN ADDRESS
2022  004544  000264                          SEZ                     ;ENSURE Z SET
2023  004546  010245                          MOV     R2,-(R5)        ;EXECUTE INSTRUCTION UNDER TEST
2024  004550  001020                          BNE     TST34           ;;CC'S OK
2025  004552  000000                          HALT                    ;STATE D10.40 BAD
2026                                                                  ;FOR LOOPING CHANGE TO 'BR TST33+2' (743)
2027  004554  011101                    2$:   MOV     (R1),R1         ;GET CONTENTS OF $TMP2
2028  004556  020201                          CMP     R2,R1           ;DID A MODE 5 TAKE PLACE?
2029  004560  001001                          BNE     3$              ;BRANCH IF NO
2030  004562  000000                          HALT                    ;EITHER IRCD DM357 STUCK HIGH OR RACK E49(C1) BAD
2031                                                                  ;FOR LOOPING CHANGE TO 'BR TST33+2' (737)
2032  004564                            3$:
2033  004564  000000                          HALT                    ;DST($TMP1) DID NOT GET LOADED FROM SRC(R2)
2034                                                                  ;FOR LOOPING CHANGE TO 'BR TST33+2' (736)
2035                                    ;;*****************************************************************
2036                                    ;*TEST 34        THREE MICROSTATES (DAC*DM1*TST.B*DR0(0))
2037                                    ;*
2038                                    ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
2039                                    ;*      THIS WILL ONLY HAPPEN IF RA SAME?
2040  004566  001372                          BNE     2$              ;BRANCH IF NO
2041  004570  010145                          MOV     R1,-(R5)        ;SEE IF AUTO DEC. WORKS
2042  004572  022705  001162                  CMP     #$TMP1-2,R5     ;DID R5 AUTO DEC?
2043  004576  001372                          BNE     3$              ;BRANCH IF NO
2044  004600  000000                          HALT                    ;AUTO DEC WORKS SO ROM STATE D12.60 PROBABLY BAD
2045                                                                  ;FOR LOOPING CHANGE TO 'BR TST33+2' (730)
2046  004602                            3$:
2047  004602  000000                          HALT                    ;IRCD DSTCON=2 EITHER STUCK LOW OR
2048                                                                  ;NOT GETTING THRU KOMUX
2049                                                                  ;FOR LOOPING CHANGE TO 'BR TST33+2' (727)
2050  004604                            2$:
2051  004604  000000                          HALT                    ;BEN15 OK & DST CONST OK BUT INSTR DOSEN'T WORK
2052                                                                  ;FOR LOOPING CHANGE TO 'BR TST33+2' (726)
2053  004606  012705  004532            5$:   MOV     #1$,R5          ;GET ADDR OF INSTR UNDER TCE E44 IS BAD (AFIR53(1)*R/CLA
```

```
                                        ;*
                                        ;*      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 OBD(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
                                        ;;*****************************************************************
004612  005200                  TST34:  INC     R0              ;INCREMENT TEST NUMBER
004614  012702  001164                  MOV     #$TMP1,R2       ;PUT ADDRESS OF $TMP1 IN R2
004620  012705  177400                  MOV     #177400,R5      ;PUT 177400 IN R5
004624  010512                          MOV     R5,(R2)         ;PUT -1 IN $TMP1
004626  000240                  SYNC34: NOP
004630                          IUT34:
004630  005712                          TST     (R2)            ;EXECUTE INSTRUCTION UNDER TEST
004632                          1$:
004632  100416                          BMI     TST35           ;;BRANCH IF INSTRUCTION SET CC'S
                                ;FAILURE-TRY ROM FLOW 1,175,31,132
004634  012737  177777  001164          MOV     #-1,@#$TMP1     ;PUT -1 IN $TMP1
004642  012705  001164                  MOV     #$TMP1,R5       ;PUT ADDR OF $TMP1 IN R5
004646  005215                          INC     (R5)            ;INCREMENT $TMP1
004650  001401                          BEQ     2$              ;BRANCH IF INC WORKED
004652  000000                          HALT                    ;EITHER D12.10 FAILED OR BEN15 FAILED
                                                                ;FOR LOOPING CHANGE TO 'BR TST34+2' (760)
004654  022737  000000  001164  2$:     CMP     #0,@#$TMP1      ;DID $TMP1 GO TO ZERO?
004662  001401                          BEQ     3$              ;BRANCH IF YES
004664  000000                          HALT                    ;CANNOT DIAGNOSE ERROR
                                                                ;FOR LOOPING CHANGE TO 'BR TST34+2' (753)
004666                          3$:
004666  000000                          HALT                    ;TST.10 FAILED
                                                                ;FOR LOOPING CHANGE TO 'BR TST34+2' (752)
                                ;;*****************************************************************
                                ;*TEST 35        THREE MICROSTATES (DAC*DM1*BIT.B*DR0(0))
                                ;*
                                ;*      THIS TEST IS THE SAME AS THE PREVIOUS TEST EXCEPT A BIT INSTRUCTION IS USED.
                                ;*      IF FORK A FAILS RACE BIN*SM0 H FAILED.
                                ;*
                                ;*      IF FORK B FAILS THE INSTRUCTION DECODE ROM WORD IS BAD.
                                ;*
                                ;*      IF THE RESULTANT DATA IS BAD STATE TST.10 FAILED.
                                ;*
                                ;*      ROM FLOW-1,175,33
                                ;;*****************************************************************
004670  005200                  TST35:  INC     R0              ;INCREMENT TEST NUMBER
004672  005005                          CLR     R5              ;CLEAR R5
004674  012701  001164                  MOV     #$TMP1,R1       ;PUT ADDR OF $TMP1 IN R1
004700  010511                          MOV     R5,(R1)         ;CLEAR $TMP1.
004702  012705  100000                  MOV     #BIT15,R5       ;PUT 100000 IN R5
004706  000240                  SYNC35: NOP
004710                          IUT35:
```

```
2110  004710  030511                        BIT     R5,(R1)         ;EXECUTE INSTRUCTION UNDER TEST
2111  004712  001401                        BEQ     TST36           ;;BRANCH IF CC OK
2112  004714  000000                        HALT                    ;STATE TST.10 FAILED
2113                                                                ;FOR LOOPING CHANGE TO 'BR TST35+2' (766)
2114                                ;;*****************************************************************
2115                                ;*TEST 36       THREE MICROSTATES (DAC*DM1*CMP.B*DR0(0))
2116                                ;*
2117                                ;*      THIS TEST IS THE SAME AS THE PREVIOUS TWO TESTS
2118                                ;*      EXCEPT A CMP INSTRUCTION IS USED.
2119                                ;*
2120                                ;*      ROM FLOW-1,175,33
2121                                ;;*****************************************************************
2122  004716  005200                TST36:  INC     R0              ;INCREMENT TEST NUMBER
2123  004720  005005                        CLR     R5              ;CLEAR R5
2124  004722  012701  001164                MOV     #$TMP1,R1       ;PUT ADDR OF $TMP1 IN R1
2125  004726  010511                        MOV     R5,(R1)         ;CLEAR $TMP1
2126  004730                        SYNC36:
2127  004730  000244                        CLZ                     ;ENSURE Z CLEAR
2128  004732                        IUT36:
2129  004732  020511                        CMP     R5,(R1)         ;EXECUTE INSTRUCTION UNDER TEST
2130  004734  001401                        BEQ     TST37           ;;BRANCH IF CC OK
2131  004736  000000                        HALT                    ;STATE TST.10 FAILED
2132                                                                ;FOR LOOPING CHANGE TO 'BR TST36+2' (770)
2133                                ;;*****************************************************************
2134                                ;*TEST 37       THREE MICROSTATES (DAC*DM2*TST.B*DR0(0))
2135                                ;*
2136                                ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
2137                                ;*      THIS WILL ONLY HAPPEN IF RACE E45 IS BAD (AFIR04(1)*R/CLASS).
2138                                ;*
2139                                ;*      BEN15 & FORK B HAVE ALREADY BEEN TESTED
2140                                ;*      IF THE AUTO INC FAILS IT WILL BE DUE TO A BAD
2141                                ;*      FIELD IN ROM STATE D12.10.
2142                                ;*
2143                                ;*      ROM FLOW-2,175,33
2144                                ;;*****************************************************************
2145  004740  005200                TST37:  INC     R0              ;INCREMENT TEST NUMBER
2146  004742  005001                        CLR     R1              ;CLEAR R1
2147  004744  012705  001164                MOV     #$TMP1,R5       ;PUT ADDR. OF $TMP1 IN R5
2148  004750  010115                        MOV     R1,(R5)         ;CLEAR $TMP1
2149  004752  000240                SYNC37: NOP
2150  004754                        IUT37:
2151  004754  005725                        TST     (R5)+           ;EXECUTE INSTR. UNDER TEST
2152  004756  001006                        BNE     1$              ;BRANCH IF BAD CC
2153  004760  022705  001166                CMP     #$TMP1+2,R5     ;DID R5 AUTO INC?
2154  004764  001407                        BEQ     TST4C           ;;BRANCH IF TEST OK
2155  004766  010567  174162                MOV     R5,$REG0        ;SAVE REG 0 FOR TYPEOUT
2156  004772  000000                        HALT                    ;NO AUTO INC STATE D12.10 BAD
2157                                                                ;FOR LOOPING CHANGE TO 'BR TST37+2' (763)
2158  004774  013767  177776  174174  1$:   MOV     @#PSW,$ERPSW    ;SAVE PSW FOR TYPEOUT
2159  005002  000000                        HALT                    ;BAD CC
2160                                                                ;FOR LOOPING CHANGE TO 'BR TST37+2' (757)
2161
2162
2163                                ;;*****************************************************************
2164                                ;*THE LOGICAL SEQUENCE WOULD NEXT TEST THE BIT.B AND CMP.B INSTRUCTIONS BUT
2165                                ;*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 RACE 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$:
005030  000000                          HALT                    ;IRCB FJ/CLASS NOT GETTING THRU B FORK MUX
                                                                ;FOR LOOPING CHANGE TO 'BR TST40+2' (766)
005032  005701                  JMP1ADR:TST     R1              ;IS R1 STILL ZERO?
005034  001403                          BEQ     TST41           ;;BRANCH IF YES
005036  062706  000002                  ADD     #2,SP           ;JSR OCCURRED RE-ADJUST SP
005042  000000                          HALT                    ;RACB IR(14:9)04 STUCK LOW
                                                                ;FOR LOOPING CHANGE TO 'BR TST40+2' (761)
                                ;;*****************************************************************
                                ;*TEST 41        THREE MICROSTATES (JMP*DM2)
                                ;*
                                ;*      THIS TEST IS THE SAME AS THE PREVIOUS TEST EXCEPT THE DM=2.
                                ;*      IF FORK A FAILS RAC E45 IS BAD (AFIR04(1)*[JMP+JSR+SWAB]).
                                ;*
                                ;*      ROM FLOW-2,135,35
                                ;;*****************************************************************
005044  005200                  TST41:  INC     R0              ;INCREMENT TEST NUMBER
005046  012705  005056                  MOV     #JMP2ADR,R5     ;PUT ADDRESS OF 1$ IN R5
005052  000240                  SYNC41: NOP
005054                          IUT41:
```

```
2222  005054  000125                  JMP     (R5)+           ;EXECUTE INSTRUCTION UNDER TEST
2223  005056  022705  005060  JMP2ADR:CMP     #JMP2ADR+2,R5   ;DID R5 AUTO INC?
2224  005062  001401                  BEQ     TST42           ;;BRANCH IF YES
2225  005064  000000                  HALT                    ;NO AUTO INC
2226                                                          ;FOR LOOPING CHANGE TO 'BR TST41+2' (770)
2227                                ;;*****************************************************************
2228                                ;*TEST 42        THREE MICROSTATES (JMP*DM4)
2229                                ;*
2230                                ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
2231                                ;*      THIS WILL ONLY HAPPEN IF RACE E33(AFIR05(1)*[JMP+JSR+SWAB]) IS BAD.
2232                                ;*
2233                                ;*      ALL OTHER LOGIC HAS BEEN TESTED.
2234                                ;*
2235                                ;*      ROM FLOW-4,122,35
2236                                ;;*****************************************************************
2237  005066  005200                TST42:  INC     R0              ;INCREMENT TEST NUMBER
2238  005070  012701  005102                MOV     #NEXTT,R1       ;PUT ADDRESS OF 1$+2 IN R1
2239  005074  000240                SYNC42: NOP
2240  005076                        IUT42:
2241  005076  000141                        JMP     -(R1)           ;EXECUTE INSTRUCTION UNDER TEST
2242  005100  000240                        NOP
2243  005102  000240                NEXTT:  NOP
2244                                ;;*****************************************************************
2245                                ;*TEST 43        THREE MICROSTATES (SOB)
2246                                ;*
2247                                ;*      IF RACH U/CLASS IS NOT GOING HIGH EXECUTION WILL GO TO RSD.00
2248                                ;*      CAUSING A TRAP TO LOCATION 10 WITH THE ADDRESS OF 5$-2
2249                                ;*      ON THE TOP OF THE STACK.
2250                                ;*      IF EITHER RACE E10 IS BAD OR RACE BIN DOES NOT GO HIGH
2251                                ;*      EXECUTION WILL GO TO D67.01. EXECUTION WILL EVENTUALL GO
2252                                ;*      TO RSD.00 AFTER STATE D10.60 AND THE STACKED PC WILL BE AT 5$.
2253                                ;*      IF RACF E8 FAILS EXECUTION WILL GO TO ASC.10 WHICH WILL
2254                                ;*      PERFORM AN ASHC*DM0 OPERATION.
2255                                ;*
2256                                ;*      IF THE SOB BRANCHES WHEN IT IS NOT SUPPOSE TO EITHER
2257                                ;*      GRAE SR EQ ONE IS  STUCK HIGH OR RACK E63 IS BAD
2258                                ;*      OR STATE SOB.10 DOES NOT RESTORE THE OLD PC.
2259                                ;*
2260                                ;*      IF THE SOB DOES NOT BRANCH WHEN IT IS SUPPOSE TO EITHER
2261                                ;*      STATE SOB.00 IS BAD OR GRAE SR EQ ONE STUCK LOW OR
2262                                ;*      RACK E63(C1) IS BAD.
2263                                ;*
2264                                ;*      IF THE REGISTER DOES NOT DECREMENT STATE SOB.20 IS BAD.
2265                                ;*
2266                                ;*      ROM FLOW-57,242/262,262
2267                                ;;*****************************************************************
2268  005104  005200                TST43:  INC     R0              ;INCREMENT TEST NUMBER
2269  005106  012706  001100                MOV     #STACK,SP       ;INITIALIZE THE SP
2270  005112  012705  007777                MOV     #7777,R5        ;SETUP R5
2271  005116  000401                        BR      SKIP            ;SKIP NEXT INSTRUCTION
2272  005120  000410                GOOD:   BR      BAD+2           ;LOCATION FOR SOB TO BRANCH TO
2273  005122  005004                SKIP:   CLR     R4              ;SETUP R4
2274  005124  000240                SYNC43: NOP
2275  005126                        IUT43:
2276  005126  077404                        SOB     R4,GOOD         ;EXECUTE SOB WITH BRANCH
2277  005130  005705                        TST     R5              ;DID R5 CLEAR?
```

```
2278  005132  001001                5$:     BNE     3$              ;BRANCH IF NO
2279  005134  000000                        HALT                    ;RACF E8 IS BAD(BUF AFIR11(1)*U/CLASS)
2280                                                                ;FOR LOOPING CHANGE TO 'BR TST43+2'' (764)
2281  005136                        3$:
2282  005136  000000                        HALT                    ;EITHER GRAE SR EQ ONE IS STUCK LOW
2283                                                                ;OR RACK E63(C1) IS BAD OR SOB.10 IS BAD
2284                                                                ;FOR LOOPING CHANGE TO 'BR TST43+2'' (763)
2285  005140                        BAD:
2286  005140  000000                        HALT                    ;SOB FAILED TO BRANCH. SOB.00 BAD
2287                                                                ;EITHER GRAE SR EQ ONE STUCK HIGH OR
2288                                                                ;RACK E63(C1) BAD OR SOB.00 BAD
2289                                                                ;FOR LOOPING CHANGE TO 'BR TST43+2'' (762)
2290  005142  005005                        CLR     R5              ;SET UP R5 FOR
2291  005144  005205                        INC     R5              ;NO BRANCH CONDITION
2292  005146  000240                SYNC43: NOP
2293  005150                        IUT43:
2294  005150  077505                        SOB     R5,BAD          ;EXECUTE SOB WITHOUT BRANCH
2295  005152  005705                        TST     R5              ;DID R5 DECREMENT?
2296  005154  001401                        BEQ     1$              ;BRANCH IF YES
2297  005156  000000                        HALT                    ;R5 DID NOT DECREMENT. SOB.20 BAD
2298                                                                ;FOR LOOPING CHANGE TO 'BR BAD+2+2'' (772)
2299  005160  005005                1$:     CLR     R5
2300  005162  005001                        CLR     R1
2301  005164  005201                2$:     INC     R1
2302  005166  077502                        SOB     R5,2$           ;CHECK ALL PATTERNS OF R5 FOR SOB
2303  005170  005705                        TST     R5              ;DID R5 GET ALL THE WAY BACK TO 0?
2304  005172  001002                        BNE     3$              ;BRANCH IF NO
2305  005174  005701                        TST     R1              ;DID R1 ROLL OVER?
2306  005176  001401                        BEQ     TST44           ;;BRANCH IF YES
2307  005200                        3$:
2308  005200  000000                        HALT                    ;THE SIGNAL 'SR EQ ONE'' FAILED
2309                                                                ;FOR LOOPING CHANGE TO 'BR 1$'' (767)
2310                                        .SBTTL
2311                                ;;**************************************************************
2312                                ;*TEST 44        FOUR  MICROSTATES (DAC*DM12*P/CLASS*DRO(0)
2313                                ;*
2314                                ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
2315                                ;*      THIS WILL ONLY HAPPEN IF RACJ AFIR 14(1) DOES NOT
2316                                ;*      GET THRU RAC E42.
2317                                ;*
2318                                ;*      THE FOLLOWING COULD BE FORK B FAILURES:
2319                                ;*      IF IRCB B0 RAB00 IS STUCK HIGH OR NOT GETTING THRU
2320                                ;*      TO RACL RADR00 EXECUTION WILL GO TO EXC.90 WHICH WOULD
2321                                ;*      PUT THE RESULT INTO A REGISTER RATHER THAN MEMORY.
2322                                ;*      IF IRCB E38(6) IS STUCK HIGH EXECUTION WILL GO TO
2323                                ;*      RSD.00 CAUSING A TRAP TO 10.
2324                                ;*      IF THE BIC DOES NOT HAPPEN THEN EXC.00 IS BAD.
2325                                ;*
2326                                ;*      ROM FLOW-2,175,31,132
2327                                ;;**************************************************************
2328  005202  005200                TST44:  INC     R0              ;INCREMENT TEST NUMBER
2329  005204  012701  001164                MOV     #STMP1,R1       ;PUT ADDRESS OF STMP1 IN R1
2330  005210  005005                        CLR     R5              ;PUT A 1
2331  005212  005205                        INC     R5              ;IN R5
2332  005214  010511                        MOV     R5,(R1)         ;PUT A 1 IN STMP1
2333  005216  000240                SYNC44: NOP
```

```
2334  005220                    IUT44:
2335  005220  040521                    BIC     R5,(R1)+        ;EXECUTE INSTR. UNDER TEST
2336  005222  022701  001166            CMP     #$TMP1+2,R1     ;DID R1 AUTO INC?
2337  005226  001404                    BEQ     1$              ;BRANCH IF YES
2338  005230  005701                    TST     R1              ;DID INSTR GO THRU EXC.90?
2339  005232  001001                    BNE     2$              ;BRANCH IF NO
2340  005234  000000                    HALT                    ;EITHER IRCB B0 RAB00 IS STUCK HIGH
2341                                                            ;OR IT IS NOT GETTING THRU TO RACL RADR50
2342                                                            ;FOR LOOPING CHANGE TO 'BR TST44+2' (763)
2343  005236                    2$:
2344  005236  000000                    HALT                    ;INSTRUCTION FAILED. CAN'T DETERMINE CAUSE
2345                                                            ;FOR LOOPING CHANGE TO 'BR TST44+2' (752)
2346  005240  012701  001164    1$:     MOV     #$TMP1,R1       ;PUT ADDR OF $TMP1 IN R1
2347  005244  005711                    TST     (R1)            ;DID BIC WORK?
2348  005246  001401                    BEQ     3$              ;BRANCH IF YES
2349  005250  000000                    HALT                    ;STATE EXC.00 FAILED
2350                                                            ;FOR LOOPING CHANGE TO 'BR TST44+2' (755)
2351  005252  010511            3$:     MOV     R5,(R1)         ;PUT 1 IN $TMP2 & CLEAR Z
2352  005254  040521                    BIC     R5,(R1)+        ;EXECUTE INSTRUCTION UNDER TEST
2353  005256  001401                    BEQ     TST45           ;;CC'S OK
2354  005260  000000                    HALT                    ;STATE EXC.00 BAD
2355                                                            ;FOR LOOPING CHANGE TO 'BR TST44+2' (751)
2356                              ;;*****************************************************************
2357                              ;*TEST 45       FOUR  MICROSTATES (DAC*DM12*TST.B*DR0(1))
2358                              ;*
2359                              ;*      AFTER STATE D12.10 IF EITHER GRAB OBD(1) DOES NOT GO HIGH
2360                              ;*      OR DOES NOT GET THRU RACL E71 EXECUTION WILL GO
2361                              ;*      TO TST.10. IF EITHER GRAB OBD(0) IS STUCK HIGH OR NOT GETTING
2362                              ;*      THRU RACK E41, EXECUTION WILL GO TO D10.60. THIS WILL CAUSE
2363                              ;*      THE PROCESSOR TO HANG UP IN THE PAUSE STATE AT MICRO
2364                              ;*      ADDRESS 177.  IF THE TEST FAILS THEN STATE D12.30 FAILED.
2365                              ;*
2366                              ;*      ROM FLOW-1,175,137,33
2367                              ;*****************************************************************
2368  005262  005200            TST45:  INC     R0              ;INCREMENT TEST NUMBER
2369  005264  012705  001164            MOV     #$TMP1,R5       ;PUT ADDRESS OD $TMP1 IN R5
2370  005270  012701  100000            MOV     #BIT15,R1       ;PUT NEGATIVE UPPER BYTE IN R1
2371  005274  010115                    MOV     R1,(R5)         ;MOVE R1 TO $TMP1
2372  005276  005205                    INC     R5              ;PUT ODD ADDR IN R5
2373  005300  000240            SYNC45: NOP
2374  005302                    IUT45:
2375  005302  105715                    TSTB    (R5)            ;EXECUTE INSTR UNDER TEST
2376  005304  100401                    BMI     TST46           ;;TEST OK, GO TO NEXT TEST
2377  005306  000000                    HALT                    ;EITHER STATE D12.30 FAILED OR
2378                                                            ;BEN05*FEN2 FAILED(SEE ABOVE)
2379                                                            ;FOR LOOPING CHANGE TO 'BR TST45+2' (766)
2380                              ;;*****************************************************************
2381                              ;*TEST 46       FOUR  MICROSTATES (DAC*DM4*TST.B*DR0(0))
2382                              ;*
2383                              ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
2384                              ;*      THIS WILL ONLY OCCUR IF RACE E33(AFIR05(1)*R/CLASS) IS BAD.
2385                              ;*
2386                              ;*      AFTER D10.30 IF IRCB FJ/CLASS DOES NOT GET TO RACL E71
2387                              ;*      OR IF RACL E71 IS BAD EXECUTION WILL GO TO SVC.50.
2388                              ;*
2389                              ;*      IF THE INSTRUCTION DOESN'T WORK THEN D10.60 IS BAD.
```

```
2390                                    ;*
2391                                    ;*      ROM FLOW-4,122,177,33
2392                                    ;;*****************************************************************
2393  005310  005200                    TST46:  INC     R0              ;INCREMENT TEST NUMBER
2394  005312  012701  100000                    MOV     #BIT15,R1       ;SET SIGN BIT
2395  005316  012705  001164                    MOV     #$TMP1,R5       ;PUT ADDR OF $TMP1 IN R5
2396  005322  010125                            MOV     R1,(R5)+        ;SET SIGN BIT IN $TMP1
2397  005324  005015                            CLR     (R5)            ;ENSURE $TEMP2 CLEAR
2398  005326  000240                    SYNC46: NOP
2399  005330                            IUT46:
2400  005330  005745                            TST     -(R5)           ;EXECUTE INSTRUCTION UNDER TEST
2401  005332  100407                            BMI     TST47           ;;TEST OK, GO TO NEXT TEST
2402  005334  022706  001160                    CMP     #$TMP1-4,SP     ;DID EXECUTION GO TO SVC.50?
2403  005340  001003                            BNE     1$              ;BRANCH IF NO
2404  005342  012706  001100                    MOV     #STACK,SP       ;RESTORE THE SP
2405  005346  000000                            HALT                    ;BEN15*FEN2 FAILED(SEE ABOVE)
2406                                                                    ;FOR LOOPING CHANGE TO 'BR TST46+2' (761)
2407  005350                            1$:
2408  005350  000000                            HALT                    ;STATE D10.60 FAILED
2409                                                                    ;FOR LOOPING CHANGE TO 'BR TST46+2' (760)
2410
2411
2412                                    ;;*****************************************************************
2413                                    ;*THE LOGICAL SEQUENCE HERE WOULD BE TO TEST THE BIT & CMP INSTRUCTIONS
2414                                    ;*BUT NO ADDITIONAL LOGIC IS TESTED BY THEM.
2415                                    ;;*****************************************************************
2416
2417
2418                                    ;;*****************************************************************
2419                                    ;*TEST 47        FOUR  MICROSTATES (DAC*DM6*0/CLASS)
2420                                    ;*
2421                                    ;*      FORK A SHOULD NOT FAIL ON THIS TEST.
2422                                    ;*
2423                                    ;*      BEN01 SHOULD NOT FAIL.
2424                                    ;*
2425                                    ;*      BEN15*FEN2 SHOULD NOT FAIL.
2426                                    ;*
2427                                    ;*      IF D67.00 FAILS TO INCREMENT THE PC AN RTI INSTRUCTION
2428                                    ;*      WILL BE EXECUTED.
2429                                    ;*      IF D67.00 FAILS TO CLOCK THE BR THE INSTRUCTION
2430                                    ;*      WILL BE ADDED AS THE INDEX WORD.
2431                                    ;*      IF D67.10 FAILS TO ADD THE INDEX NUMBER THE SOURCE
2432                                    ;*      WILL BE PUT IN THE WRONG LOCATION.
2433                                    ;*
2434                                    ;*      ROM FLOW-6,251,122,157
2435                                    ;;*****************************************************************
2436  005352  005200                    TST47:  INC     R0              ;INCREMENT TEST NUMBER
2437  005354  005306                            DEC     SP              ;THIS
2438  005356  005306                            DEC     SP              ;GROUP
2439  005360  012705  000340                    MOV     #340,R5         ;OF INSTRUCTIONS
2440  005364  010516                            MOV     R5,(SP)         ;SETS UP THE
2441  005366  005306                            DEC     SP              ;STACK
2442  005370  005306                            DEC     SP              ;TO HANDLE
2443  005372  012705  005432                    MOV     #BAD1,R5        ;AN ERONEOUS
2444  005376  010516                            MOV     R5,(SP)         ;RTI INSTRUCTION
2445  005400  012705  001164                    MOV     #$TMP1,R5       ;PUT ADDR OF $TMP1 IN R5
```

```
2446  005404  005015                      CLR     (R5)            ;CLEAR $TMP1
2447  005406  012701  100000              MOV     #BIT15,R1       ;SET SIGN BIT IN R1
2448  005412  000240                SYNC47: NOP
2449  005414                         IUT47:
2450  005414  010165  000002              MOV     R1,2(R5)        ;EXECUTE INSTRUCTION UNDER TEST
2451  005420  012706  001100              MOV     #STACK,SP       ;RESTORE THE SP
2452  005424  005715                      TST     (R5)            ;DID INDEX WORD GET ADDED?
2453  005426  100002                      BPL     TST50           ;;BRANCH IF YES
2454  005430  000000                      HALT                    ;D67.10 FAILED TO ADD INDEX
2455                                                               ;FOR LOOPING CHANGE TO 'BR TST47+2' (751)
2456  005432                         BAD1:
2457  005432  000000                      HALT                    ;D67.00 FAILED TO INC PC
2458                                                               ;FOR LOOPING CHANGE TO 'BR TST47+2' (750)
2459                                 ;;*********************************************************************
2460                                 ;*TEST 50       FOUR  MICROSTATES (BIN*SM12*DM0*-DF7*SR0(1))
2461                                 ;*
2462                                 ;*      IF FORK A FAILS EXECTUTION WILL  GO TO STATE D12.00.
2463                                 ;*      THIS WILL HAPPEN IF RACE BF1=7 DOES NOT GO HIGH.
2464                                 ;*      STATE D12.00 WOULD BITB R5 & THE CONTENTS OF LOCATION 200
2465                                 ;*      WHICH IS 137.
2466                                 ;*
2467                                 ;*      THE FOLLOWING COULD BE BEN14*FORK C FAILURES:
2468                                 ;*      IF IRCC C0 RAB00 DOES NOT GO HIGH EXECUTION WILL GO TO
2469                                 ;*      D00.90 WHICH WILL TEST THE LOW BYTE INSTEAD OF THE HIGH BYTE.
2470                                 ;*
2471                                 ;*      IF STATE D00.80 FAILS TO SWAP THE BYTES IT WILL LOOK LIKE
2472                                 ;*      A FORK C FAILURE.
2473                                 ;*
2474                                 ;*      ROM FLOW-21,27,204,205
2475                                 ;;*********************************************************************
2476  005434  005200                TST50:  INC     R0              ;INCREMENT TEST NUMBER
2477  005436  012705  001164              MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
2478  005442  012701  100000              MOV     #BIT15,R1       ;PUT NEG HIGH & POS LOW BYTE IN R1
2479  005446  010115                      MOV     R1,(R5)         ;PUT IN $TMP1
2480  005450  005205                      INC     R5              ;PUT ADDR OF $TMP1 H BYTE IN R5
2481  005452  012701  000200              MOV     #200,R1         ;PUT POS HIGH & NEG LOW BYTE IN R1
2482  005456  000240                SYNC50: NOP
2483  005460                         IUT50:
2484  005460  131501                      BITB    (R5),R1         ;EXECUTE INSTRUCTION UNDER TEST
2485  005462  100405                      BMI     TST51           ;;TEST OK, GO TO NEXT TEST
2486  005464  010215                      MOV     R2,(R5)         ;PUT 200 IN $TMP1
2487  005466  131501                      BITB    (R5),R1         ;EXECUTE FAILED INSTRUCTION
2488  005470  100401                      BMI     1$              ;BRANCH IF FORK C FAILED
2489  005472  000000                      HALT                    ;RACE BF1=7 NOT GOING HIGH
2490                                                               ;EITHER RACJ AFIR14(1) DOES NOT
2491                                                               ;GET TO RACE E40 OR E40 BAD
2492                                                               ;FOR LOOPING CHANGE TO 'BR TST50+2' (761)
2493  005474                         1$:
2494  005474  000000                      HALT                    ;EITHER IRCC C0 RAB00 DOES NOT GO HIGH
2495                                                               ;OR STATE D00.80 FAILED TO SWAP THE BYTES
2496                                                               ;FOR LOOPING CHANGE TO 'BR TST50+2' (760)
2497                                 ;;*********************************************************************
2498                                 ;*TEST 51       FOUR  MICROSTATES (BIN*SM12*DM0*DF7*SR0(0))
2499                                 ;*
2500                                 ;*      IF FORK A FAILS EXECUTION WILL EITHER GO TO STATE D12.00 OR EXC.80.
2501                                 ;*      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 CO 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)
005610                          $RTI:
005610  000000                          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
                                ;;**************************************************************
005612  005200                  TST52:  INC     R0              ;INCREMENT TEST NUMBER
005614  005005                          CLR     R5              ;PUT ADDRESS 0 IN R5
005616  012701  005634                  MOV     #POINT,R1       ;PUT ADDR OF POINT IN R1
005622  000301                          SWAB    R1              ;EXCHANGE BYTES
005624  010115                          MOV     R1,(R5)         ;PUT DATA IN LOCATION 0
005626  005205                          INC     R5              ;CHANGE R5 TO HIGH BYTE ADDRESS
005630  000240                  SYNC52: NOP
005632                          IUT52:
005632  121507                          CMPB    (R5),PC         ;EXECUTE INSTRUCTION UNDER TEST
005634  001406                  POINT:  BEQ     4$              ;BRANCH IF OK
005636  016705  000002                  MOV     2$,R5           ;PUT CONTENTS OF 2$ IN R5
005642  121507                          CMPB    (R5),PC         ;EXECUTE FAILED INSTRUCTION
005644  001401                  2$:     BEQ     3$              ;BRANCH IF FORK A FAILED
005646  000000                          HALT                    ;STATE D07.00 FAILED TO SWAB OR LOAD SR
                                                                ;FOR LOOPING CHANGE TO 'BR TST52+2'' (762)
005650                          3$:
005650  000000                          HALT                    ;RACE BF1=0 NOT GOING HIGH
                                                                ;FOR LOOPING CHANGE TO 'BR TST52+2'' (761)
005652  121507                  4$:     CMPB    (R5),PC         ;DID SHFTR GET LOADED
005654  001001                          BNE     TST53           ;;BRANCH IF YES
005656  000000                          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 ALRE   ' BEEN TESTED.
                                ;*
                                ;*      IF THE SRC FAILS TO AUTO DECREMENT STATE S45.00 IS BAD.
                                ;*
                                ;*      ROM FLOW-24,23,27,205
```

```
2614                                    ;;*****************************************************************
2615  005660  005200                    TST53:  INC     R0              ;INCREMENT TEST NUMBER
2616  005662  012705  001164                    MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
2617  005666  012701  100000                    MOV     #BIT15,R1       ;SET SIGN BIT IN R1
2618  005672  010125                            MOV     R1,(R5)+        ;SET SIGN BIT $TMP1 & STEP R5 TO $TMP2
2619  005674  005015                            CLR     (R5)            ;CLEAR $TMP2
2620  005676  012701  000002                    MOV     #2,R1           ;PUT ADDRESS OF LOC 2 IN R1
2621  005702  005011                            CLR     (R1)            ;CLEAR LOCATION ZERO
2622  005704  000240                    SYNC53: NOP
2623  005706                            IUT53:
2624  005706  054501                            BIS     -(R5),R1        ;EXECUTE INSTRUCTION UNDER TEST
2625  005710  100411                            BMI     TST54           ;;TEST OK, GO TO NEXT TEST
2626  005712  020537  000002                    CMP     R5,@#2          ;DID FORK A FAIL?
2627  005716  001001                            BNE     1$              ;BRANCH IF NO
2628  005720  000000                            HALT                    ;EITHER RACE BF1=7 OR SM0 DID NOT GO HIGH
2629                                            ;FOR LOOPING CHANGE TO 'BR TST53+2' (760)
2630  005722  020527  001166            1$:     CMP     R5,#$TMP2       ;DID R5 FAIL TO DECREMENT?
2631  005726  001001                            BNE     2$              ;BRANCH IF NO
2632  005730  000000                            HALT                    ;STATE S45.00 DID NOT DECREMENT R5
2633                                            ;FOR LOOPING CHANGE TO 'BR TST53+2' (754)
2634  005732                            2$:
2635  005732  000000                            HALT                    ;INSTRUCTION FAILED
2636                                            ;FOR LOOPING CHANGE TO 'BR TST53+2' (753)
2637                                    ;;*****************************************************************
2638                                    ;*TEST 54        FOUR  MICROSTATES (RTS)
2639                                    ;*
2640                                    ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
2641                                    ;*      THIS WILL ONLY HAPPEN IF RACF E3 DOES NOT GO HIGH.
2642                                    ;*
2643                                    ;*      IF THE PC OR R5 FAILS THE TEST WILL HALT.
2644                                    ;*
2645                                    ;*      ROM FLOW-40,223,224,342
2646                                    ;;*****************************************************************
2647  005734  005200                    TST54:  INC     R0              ;INCREMENT TEST NUMBER
2648  005736  012706  001100                    MOV     #STACK,SP       ;INITIALIZE THE STACK
```

```
 2649  005742  012705  100000            MOV     #BIT15,R5       ;SET SIGN BIT IN R5
 2650  005746  010546                    MOV     R5,-(SP)        ;PUT R5 ON THE STACK
 2651  005750  012705  005760            MOV     #1$,R5          ;PUT ADDRESS TO RETURN TO IN R5
 2652  005754  000205                    RTS     R5              ;EXECUTE INSTRUCTION UNDER TEST
 2653  005756  000000                    HALT                    ;STATE RTS.00 FAILED TO PUT R5 IN THE PC
 2654                                                            ;FOR LOOPING CHANGE TO 'BR TST54+2'' (767)
 2655  005760  005705            1$:     TST     R5              ;DID R5 GET THE TOP OF THE STACK?
 2656  005762  100401                    BMI     TST55           ;;BRANCH IF YES
 2657  005764  000000                    HALT                    ;THE RTS FAILED TO PUT THE STACK IN R5
 2658                                                            ;FOR LOOPING CHANGE TO 'BR TST54+2'' (764)
 2659                              ;;*****************************************************************
 2660                              ;*TEST 55        FOUR  MICROSTATES (JMP*DM6)
 2661                              ;*
 2662                              ;*      IF FORK A FAILS EXECUTION WILL GO TO STATE D12.01.
 2663                              ;*      THIS WOULD CAUSE A JMP*DM1 TO EXECUTE.
 2664                              ;*
 2665                              ;*      NEITHER BEN01 NOR BEN15*FEN2 SHOULD FAIL SINCE THEY HAVE ALREADY BEEN
 2666                              ;*      TESTED.
 2667                              ;*
 2668                              ;*      ROM FLOW-6,251,122,35
 2669                              ;;*****************************************************************
 2670  005766  005200            TST55:  INC     R0              ;INCREMENT TEST NUMBER
 2671  005770  012705  006002            MOV     #JMPTO,R5       ;PUT ADDRESS-2 TO JUMP TO IN R5
 2672  005774  000240            SYNC55: NOP
 2673  005776                    IUT55:
 2674  005776  000165  000002            JMP     2(R5)           ;EXECUTE INSTRUCTION UNDER TEST
 2675  006002                    JMPTO:
 2676  006002  000000                    HALT                    ;JMP*DM6 DID NOT JUMP OR THE OFFSET
 2677                                                            ;DID NOT GET ADDED OR RACE E33 FAILED
 2678                                                            ;AND A JMP*DM1 WAS EXECUTED
 2679                                                            ;FOR LOOPING CHANGE TO 'BR TST55+2'' (772)
 2680                                      .SBTTL
 2681                              ;;*****************************************************************
 2682                              ;*TEST 56        FIVE  MICROSTATES (DAC*DM12*P/CLASS*DR0(1))
 2683                              ;*
 2684                              ;*      FORK A SHOULDN'T FAIL.
 2685                              ;*
 2686                              ;*      BEN15 SHOULDN'T FAIL SINCE THIS LOGIC HAS BEEN TESTED.
 2687                              ;*      NEITHER SHOULD BEN05*FEN2.
 2688                              ;*
 2689                              ;*      IF FORK B FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
 2690                              ;*      THIS FAILURE WOULD BE CAUSED BY A BAD FIELD (PART PCLASS)
 2691                              ;*      IN THE IR DECODE ROM.
 2692                              ;*
 2693                              ;*      ROM FLOW-1,175,137,31,132
 2694                              ;;*****************************************************************
 2695  006004  005200            TST56:  INC     R0              ;INCREMENT TEST NUMBER
 2696  006006  012705  001164            MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
 2697  006012  012701  040000            MOV     #BIT14,R1       ;SET BIT 14 IN R1
 2698  006016  010115                    MOV     R1,(R5)         ;SET BIT 14 IN $TMP1
 2699  006020  005205                    INC     R5              ;SET R5 TO HIGH BYTE OF $TMP1
 2700  006022  000240            SYNC56: NOP
 2701  006024                    IUT56:
 2702  006024  106115                    ROLB    (R5)            ;  'ECUTE INSTRUCTION UNDER TEST
 2703  006026  100401                    BMI     TST57           ,,.EST OK, GO TO NEXT TEST
 2704  006030  000000                    HALT                    ;ROLB*DM1*DR5(1)) FAILED (BAD CC)
```

```
                                                        ;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'' (767)
```

```
                                ;;*****************************************************************
                                ;*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.
                                ;;*****************************************************************



                                ;;*****************************************************************
                                ;*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.
                                ;;*****************************************************************


                                ;;*****************************************************************
                                ;*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
                                ;;*****************************************************************
006126  005200                  TST61:  INC     R0              ;INCREMENT TEST NUMBER
006130  012705  001164                  MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
006134  010565  000002                  MOV     R5,2(R5)        ;PUT ADDRESS OF $TMP1 IN $TMP2
006140  010501                          MOV     R5,R1           ;PUT ADDRESS OF $TMP1 IN R1
006142  000240                  SYNC61: NOP
006144                          OUT61:
006144  046501  000002                  BIC     2(R5),R1        ;EXECUTE INSTRUCTION UNDER TEST
006150  005701                          TST     R1              ;DID R1 GO TO ZERO?
006152  001405                          BEQ     TST62           ;;BRANCH IF YES
006154  005767  173006                  TST     $TMP2           ;DID $TMP2 GO TO ZERO?
006160  001001                          BNE     1$              ;BRANCH IF NO
006162  000000                          HALT                    ;RACE BF1=0 DID NOT GO HIGH ON BIC
                                                                ;FOR LOOPING CHANGE TO 'BR TST61+2' (762)
006164                          1$:
006164  000000                          HALT                    ;INSTRUCTION FAILED
                                                                ;FOR LOOPING CHANGE TO 'BR TST61+2' (761)
                                ;;*****************************************************************
                                ;*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 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 IMPUT B1 IS STUCK LOW
```

```
2873
2874  006370  011104              5$:     MOV     (R1),R4         ;FOR LOOPING CHANGE TO 'BR TST62+2'' (700)
2875  006372  020504                      CMP     R5,R4           ;GET CONTENTS OF $TMP2
2876  006374  001001                      BNE     6$              ;DID D12.01 GET EXECUTED?
2877  006376  000000                      HALT                    ;BRANCH IF NO
2878                                                              ;RACE E29 IS BAD
2879  006400  022706  001074        6$:     CMP     #1074,SP        ;FOR LOOPING CHANGE TO 'BR TST62+2'' (674)
2880  006404  001001                      BNE     2$              ;DID INSTRUCTION CAUSE WAIT TO OCCUR?
2881  006406  000000                      HALT                    ;BRANCH IF NO
2882                                                              ;EITHER IRCC DSTM0 H IS STUCK LOW OR
2883                                                              ;IT IS NOT GETTING THRU RACL RADR05
2884  006410  005004                      CLR     R4              ;FOR LOOPING CHANGE TO 'BR TST62+2'' (670)
2885  006412  005714                      TST     (R4)            ;PUT ADDR. OF LOCATION ZERO IN R4
2886  006414  100001                      BPL     IT              ;DID A SM1*DM3 GET EXECUTED?
2887  006416  000000                      HALT                    ;BRANCH IF NO
2888                                                              ;IRCC C FORK MUX INPUT B2 IS STUCK LOW
2889  006420  105737                      TSTB    @(PC)+          ;FOR LOOPING CHANGE TO 'BR TST62+2'' (664)
2890  006422  177564                      .WORD   177564          ;IS PRINTER DONE?
2891  006424  100375                      BPL     IT
2892  006426  032702  000001            BIT     #BIT0,R2        ;BRANCH IF NO
2893  006432  001001                      BNE     TST63           ;DID INSTRUCTION LEAVE C BIT SET?
2894  006434  000000                      HALT                    ;;BRANCH IF YES
2895                                                              ;IRCC C FORK MUX SELECT NOT GOING HIGH(ON CHIP)
2896                                                              ;FOR LOOPING CHANGE TO 'BR TST62+2'' (655)
2897
2898                              ;;*****************************************************************
2899                              ;*THE LOGICAL FLOW AT THIS POINT WOULD TEST A BIN*SM12*DM12*SR0(0)*DR0(0)
2900                              ;**[TST.B+BIT.B+CMP.B] INSTRUCTION BUT NO ADDITIONAL LOGIC WOULD BE TESTED.
2901                              ;;*****************************************************************
2902
2903
2904                              ;;*****************************************************************
2905                              ;*TEST 63       FIVE  MICROSTATES (BIN*SM12*DM12*SR0(1)*DR0(0)*CMPB)
2906                              ;*
2907                              ;*      THE ONLY THING THAT SHOULD FAIL WOULD BE STATE D12.90(110).
2908                              ;*
2909                              ;*      ROM FLOW-21,27,110,175,33
2910                              ;;*****************************************************************
2911  006436  005200              TST63:  INC     R0              ;INCREMENT TEST NUMBER
2912  006440  012705  001164              MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
2913  006444  112715  000377              MOVB    #377,(R5)       ;SET LOW BYTE OF $TMP1 TO ALL ONE'S
2914  006450  012701  001166              MOV     #$TMP2,R1       ;PUT ADDRESS OF $TMP2 IN R1
2915  006454  012711  177400              MOV     #177400,(R1)    ;SET HIGH BYTE OF $TMP2 TO ALL ONES
2916  006460  005201                      INC     R1              ;ADJUST R1 TO $TEMP2 HIGH BYTE
2917  006462  000240              SYNC63: NOP
2918  006464                      IUT63:
2919  006464  121115                      CMPB    (R1),(R5)       ;EXECUTE INSTRUCTION UNDER TEST
2920  006466  001401                      BEQ     TST64           ;;TEST OK, GO TO NEXT TEST
2921  006470  000000                      HALT                    ;STATE D12.90 FAILED
2922                                                              ;FOR LOOPING CHANGE TO 'BR TST63+2'' (763)
2923                              ;;*****************************************************************
2924                              ;*TEST 64       FIVE  MICROSTATES (BIN*SM12*DM4*0/CLASS)
2925                              ;*
2926                              ;;      IF FORK C FAILS EXECUTION WILL GO TO D12.80
2927                              ;*      WHICH WILL EXECUTE A SM1*DM2 TYPE INSTRUCTION.
2928                              ;*
```

```
                        ;*      IF THE INSTRUCTION FAILS EITHER D45.80 OR D40.20 FAILED.
                        ;*
                        ;*      ROM FLOW-21,27,115,121,157
                        ;;*****************************************************************
006472  005200          TST64:  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' (761)
                                .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
```

```
2985  006552  005205                  T65   SIX   INC     R5              ;SET R5 TO HIGH BYTE OF $TMP1
2986  006554  005002                        SIX   CLR     R2              ;ENSURE R2 CLEAR
2987  006556  012703  000001              SIX   MOV     #1,R3           ;ENSURE R3 NOT CLEAR
2988  006562  012701  177776              SIX   MOV     #PSW,R1         ;PUT ADDRESS OF PSW IN R1
2989  006566  000264                      SIX   SEZ                     ;ENSURE Z BIT SET
2990  006570                        SYNC65:
2991  006570                        IUT65:
2992  006570  106215                      SIX   ASRB    (R5)            ;EXECUTE INSTRUCTION UNDER TEST
2993  006572  011102                AFTER:SIX   MOV     (R1),R2         ;SAVE PSW
2994  006574  010567  172354              SIX   MOV     R5,$REG0        ;SAVE R5
2995  006600  005703                      SIX   TST     R3              ;DID MULTIPLY OCCUR & CLEAR R3?
2996  006602  001001                      SIX   BNE     3$              ;BRANCH IF NO
2997  006604  000000                      SIX   HALT                    ;B FORK MUX INPUT B1 STUCK HIGH
2998                                                                    ;FOR LOOPING CHANGE TO 'BR TST65+2' (753)
2999  006606  012701  006572        3$:   SIX   MOV     #AFTER,R1       ;PUT ADDRESS OF 1$ IN R1
3000  006612  000301                      SIX   SWAB    R1              ;REVERSE BYTES
3001  006614  106001                      SIX   RORB    R1              ;MAKE IT LOOK LIKE EXC.00 WAS ENTERED
3002  006616  012703  001164              SIX   MOV     #$TMP1,R3       ;PUT ADDRESS OF $TMP1 IN R3
3003  006622  020113                      SIX   CMP     R1,(R3)         ;DID EXC.00 GET ENTERED?
3004  006624  001001                      SIX   BNE     4$              ;BRANCH IF NO
3005  006626  000000                      SIX   HALT                    ;IRCB 08D(ASRB OR RORB) STUCK HIGH
3006                                                                    ;FOR LOOPING CHANGE TO 'BR TST65+2' (742)
3007  006630  022716  006572        4$:   SIX   CMP     #AFTER,(SP)     ;DID MFP.00 EXECUTE?
3008  006634  001001                      SIX   BNE     5$              ;BRANCH IF NO
3009  006636  000000                      SIX   HALT                    ;B FORK MUX INPUT B2 STUCK LOW
3010                                                                    ;FOR LOOPING CHANGE TO 'BR TST65+2' (736)
3011  006640  032702  000004        5$:   SIX   BIT     #4,R2           ;DID CCP.00 EXECUTE?
3012  006644  001401                      SIX   BEQ     6$              ;BRANCH IF NO
3013  006646  000000                      SIX   HALT                    ;IRCC B0 RAB04 NOT GETTING THRU RACL RADR54
3014                                                                    ;FOR LOOPING CHANGE TO 'BR TST65+2' (732)
3015  006650  012701  006572        6$:   SIX   MOV     #AFTER,R1       ;GET ADDRESS OF AFTER
3016  006654  010105                      SIX   MOV     R1,R5           ;SAVE R1
3017  006656  006001                      SIX   ROR     R1              ;RIGHT SHIFT R1 WITHOUT USING ASR
3018  006660  042701  000377              SIX   BIC     #377,R1         ;CLEAR LOWER BYTE OF R1
3019  006664  042705  177400              SIX   BIC     #177400,R5      ;CLEAR UPPER BYTE OF R5
3020  006670  050501                      SIX   BIS     R5,R1           ;MAKE LOWER BYTE OF R1OW
3021                                                                    ;BYTE OF DST. OPERAND
3022  006672  020167  172266              SIX   CMP     R1,$TMP1        ;DID DESTINATION GET ASR'D PROPERLY?
3023  006676  001401                      SIX   BEQ     TST66           ;;BRANCH IF YES
3024  006700  000000                      SIX   HALT                    ;EITHER SHR.00 OR SHR.10 FAILED
3025                                                                    ;FOR LOOPING CHANGE TO 'BR TST65+2' (715)
3026                                  ;;*****************************************************************
3027                                  ;*TEST 66       SIX   MICROSTATES (DAC*DM12*RORB*DR0(1))
3028                                  ;*
3029                                  ;*      THIS TEST IS THE SAME AS THE LAST ONE EXCEPT A RORB
3030                                  ;*      IS USED INSTEAD OF AN ASRB.
3031                                  ;*
3032                                  ;*      FORK B WILL ONLY FAIL IF IRCB E36(13) IS STUCK HIGH
3033                                  ;*      WHICH WILL CAUSE EXECUTION TO GO TO EXC.00.
3034                                  ;*
3035                                  ;*      ROM FLOW-2,175,137,64,123,132
3036                                  ;;*****************************************************************
3037  006702  005200                  TST66:SIX   INC     R0              ;INCREMENT TEST NUMBER
3038  006704  012705  001164              SIX   MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 HIGH BYTE IN R5
3039  006710  112725  000100              SIX   MOVB    #100,(R5)+      ;SET BIT 6 IN LOW BYTE OF $TMP1
3040  006714  112715  000200              SIX   MOVB    #200,(R5)       ;SET SIGN BIT IN HIGH BYTE OF $TMP1
```

```
3041  006720  000240                SYNC66: NOP
3042  006722                        IUT66:
3043  006722  106025                        RORB    (R5)+           ;EXECUTE INSTRUCTION UNDER TEST
3044  006724  022745  040100                CMP     #40100,-(R5)    ;DID INSTRUCTION WORK?
3045  006730  001401                        BEQ     TST67           ;;BRANCH IF YES
3046  006732  000000                        HALT                    ;IRCB E36(13) NOT GOING LOW ON RORB
3047                                                                ;FOR LOOPING CHANGE TO 'BR TST66+2' (764)
3048
3049
3050
3051                                ;;**************************************************************
3052                                ;*THE LOGICAL FLOW AT THIS POINT WOULD TEST A DAC*DM3*[TST.B+BIT.B+CMP.B]*DR0(0)
3053                                ;*FOLLOWED BY A DAC*DM4*P/CLASS*DR0(0) INSTRUCTION BUT NO ADDITIONAL LOGIC
3054                                ;*IS TESTED
3055                                ;;**************************************************************
3056
3057
3058
3059                                ;;**************************************************************
3060                                ;*TEST 67        SIX   MICROSTATES (DAC*DM6*XOR*DR0(0))
3061                                ;*
3062                                ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO 10.
3063                                ;*      THIS SHOULD ONLY HAPPEN IF RACE E35(1) IS BAD.
3064                                ;*
3065                                ;*      IF THE INSTRUCTION DOESN'T WORK IT WILL HALT IN THIS TEST.
3066                                ;*
3067                                ;*      ROM FLOW-6,251,122,177,31,132
3068                                ;;**************************************************************
3069  006734  005200                TST67:  INC     R0              ;INCREMENT TEST NUMBER
3070  006736  005004                        CLR     R4              ;SETUP R4
3071  006740  005104                        COM     R4              ;SET ALL BITS IN R4
3072  006742  010437  001164                MOV     R4,@#STMP1      ;SET ALL BITS IN $TMP1
3073  006746  000240                SYNC67: NOP
3074  006750                        IUT67:
3075  006750  074467  172210                XOR     R4,$TMP1        ;EXECUTE INSTRUCTION UNDER TEST
3076  006754  005767  172204                TST     $TMP1           ;DID $TMP1 CLEAR?
3077  006760  001401                        BEQ     TST70           ;;BRANCH IF YES
3078  006762  000000                        HALT                    ;INSTRUCTION FAILED
3079                                                                ;FOR LOOPING CHANGE TO 'BR TST67+2' (765)
3080
3081
3082
3083
3084                                ;;**************************************************************
3085                                ;*THE LOGICAL SEQUENCE WOULD TEST A DAC*DM6*[TST.B+BIT.B+CMP.B]*DR0(1) BUT ALL
3086                                ;*THE LOGIC HAS BEEN TESTED.
3087                                ;;**************************************************************
3088
3089
3090
3091                                ;;**************************************************************
3092                                ;*TEST 70        SIX   MICROSTATES (NEG.B*DM12*DR0(0))
3093                                ;*
3094                                ;*      NEITHER FORK A NOR BGN15 SHOULD FAIL.
3095                                ;*
3096                                ;*      IF FORK B FAILS EXECUTION WILL GO TO RSD.00 CAUSING
```

```
3097                                    ;*      A TRAP TO LOCATION 10 OR EXC.00.
3098                                    ;*      RSD.00 SHOULD ONLY OCCUR IF THE B FORK MUX
3099                                    ;*      STROBE IS BEING HELD LOW(CHIP FAILURE).
3100                                    ;*
3101                                    ;*      ROM FLOW-1,175,67,271,163,132
3102                                    ;;*****************************************************************
3103  006764  005200                    TST70:  INC     R0              ;INCREMENT TEST NUMBER
3104  006766  005067  172172                    CLR     $TMP1           ;ENSURE $TMP1 CLEAR
3105  006772  005267  172166                    INC     $TMP1           ;PUT 1 IN $TMP1
3106  006776  012705  001164                    MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
3107  007002  000240                    SYNC70: NOP
3108  007004                             IUT70:
3109  007004  005415                            NEG     (R5)            ;EXECUTE INSTRUCTION UNDER TEST
3110  007006  022715  177777                    CMP     #177777,(R5)    ;DID $TMP1 NEGATE?
3111  007012  001411                            BEQ     3$              ;BRANCH IF YES
3112  007014  022715  177776                    CMP     #177776,(R5)    ;DID $TMP1 COMPLEMENT?
3113  007020  001001                            BNE     1$              ;BRANCH IF NO
3114  007022  000000                            HALT                    ;EITHER STATE NEG.10 FAILED
3115                                                                    ;OR FORK B FAILED. IRCB NEG.B H
3116                                                                    ;IS STUCK HIGH.
3117                                                                    ;FOR LOOPING CHANGE TO 'BR TST70+2' (761)
3118  007024  022715  000001           1$:     CMP     #1,(R5)         ;DID $TMP1 STAY THE SAME?
3119  007030  001001                            BNE     2$              ;BRANCH IF NO
3120  007032  000000                            HALT                    ;$TMP1 DID NOT GET LOADED
3121                                                                    ;FOR LOOPING CHANGE TO 'BR TST70+2' (755)
3122  007034                             2$:
3123  007034  000000                            HALT                    ;INSTRUCTION FAILED
3124                                                                    ;FOR LOOPING CHANGE TO 'BR TST70+2' (754)
3125  007036  000264                    3$:     SEZ                     ;ENSURE Z SET
3126  007040  005415                            NEG     (R5)            ;EXECUTE INSTRUCTION UNDER TEST
3127  007042  001001                            BNE     TST71           ;;CC'S OK
3128  007044  000000                            HALT                    ;STATE NEG.10 BAD
3129                                                                    ;FOR LOOPING CHANGE TO 'BR TST70+2' (750)
3130
3131
3132
3133                                    ;;*****************************************************************
3134                                    ;*THE LOGICAL SEQUENCE WOULD NEXT EXECUTE A JMP*DM3 BUT NO
3135                                    ;*ADDITIONAL LOGIC IS TESTED.
3136                                    ;;*****************************************************************
3137
3138
3139                                    ;;*****************************************************************
3140                                    ;*TEST 71        SIX   MICROSTATES (BIN*SM3*DM0*-DF7*SR0(0))
3141                                    ;*
3142                                    ;*      FORK A SHOULD NOT FAIL.
3143                                    ;*
3144                                    ;*      IF BEN14*FEN4 FAILS EXECUTION WILL GO TO D00.90. THIS
3145                                    ;*      WILL CAUSE A SM2 TO BE EXECUTED. THIS SHOULD ONLY
3146                                    ;*      HAPPEN IF IRCC SM357 IS STUCK LOW OR NOT GETTING THRU RACL E70.
3147                                    ;*
3148                                    ;*      ROM FLOW-22,27,317,143,146,205
3149                                    ;;*****************************************************************
3150  007046  005200                    TST71:  INC     R0              ;INCREMENT TEST NUMBER
3151  007050  012705  001164                    MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
3152  007054  012715  007064                    MOV     #POINT2,(R5)    ;PUT ADDRESS OF POINT2 IN $TMP1
```

```
 3153  007060  000240                  SYNC71: NOP
 3154  007062                          IUT71:
 3155  007062  013501                          MOV     @(R5)+,R1       ;EXECUTE INSTRUCTION UNDER TEST
 3156  007064  026701  177774          POINT2: CMP     POINT2,R1       ;DID R1 GET CORRECT DATA?
 3157  007070  001405                          BEQ     TST72           ;;BRANCH IF YES
 3158  007072  022701  007064                  CMP     #POINT2,R1      ;DID A SM2 GET EXECUTED?
 3159  007076  001001                          BNE     2$              ;BRANCH IF NO
 3160  007100  000000                          HALT                    ;EITHER IRCC SM357 STUCK LOW
 3161                                                                  ;OR NOT GETTING THRU RACL E70
 3162                                                                  ;FOR LOOPING CHANGE TO 'BR TST71+2' (763)
 3163  007102                          2$:
 3164  007102  000000                          HALT                    ;EITHER S13.20 OR S13.30 OR S13.40 FAILED
 3165                                                                  ;FOR LOOPING CHANGE TO 'BR TST71+2' (762)
 3166
 3167
 3168                                  ;;*********************************************************************
 3169                                  ;*THE LOGICAL SEQUENCE WOULD NEXT TEST A BIN*SM6*DM0*DF7*SR0(1), THEN A BIN*SM12*
 3170                                  ;*DM12*SR0(0)*DR0(1)*[TST.B+BIT.B+CMP.B] THEN A BIN*SM12*DM12*SR0(0)*
 3171                                  ;*DR0(0)*P/CLASS THEN A BIN*SM12*DM12*SR0(1)*DR0(1)*[TST.B+BIT.B+CMP.B]
 3172                                  ;*THEN A BIN*SM12*DM12*SR0(1)*DR0(0)*P/CLASS AND THEN A BIN*SM12*DM%*
 3173                                  ;*SR0(0)*DR0(0)*[TST.B+BIT.B+CMP.B] INSTRUCTION, BUT NO ADDITIONAL LOGIC
 3174                                  ;*IS TESTED.
 3175                                  ;;*********************************************************************
 3176
 3177
 3178                                  ;;*********************************************************************
 3179                                  ;*TEST 72        SIX   MICROSTATES (BIN*SM12*DM%*SR0(1)*DR0(0)*CMPB)
 3180                                  ;*
 3181                                  ;*      A FAILURE WILL ONLY OCCUR IF STATE D45.90 FAILS.
 3182                                  ;*
 3183                                  ;*      IF THE DST REG. DOES NOT DECREMENT STATE D45.90 IS BAD.
 3184                                  ;*      IF THE CONDITION CODES ARE BAD THEN STATE D40.30
 3185                                  ;*      PROBABLY DID NOT SWAP THE BYTES OF THE SRC OPERAND.
 3186                                  ;*
 3187                                  ;*      ROM FLOW-1,27,114,131,177,33
 3188                                  ;;*********************************************************************
 3189  007104  005200                  TST72:  INC     R0              ;INCREMENT TEST NUMBER
 3190  007106  012705  100000                  MOV     #BIT15,R5       ;SET SIGN BIT IN R5
 3191  007112  010567  172046                  MOV     R5,$TMP1        ;SET SIGN BIT IN $TMP1
 3192  007116  012701  001166                  MOV     #$TMP2,R1       ;PUT ADDRESS OF $TMP2 IN R1
 3193  007122  010105                          MOV     R1,R5           ;PUT ADDRESS OF $TMP2 IN R5
 3194  007124  112721  000200                  MOVB    #BIT7,(R1)+     ;SET LOW BYTE SIGN IN $TMP2 & STEP R1
 3195  007130  105011                          CLRB    (R1)            ;ENSURE $TMP2 HIGH BYTE CLEAR
 3196  007132  005305                          DEC     R5              ;ADJUST R5 TO POINT AT $TMP1 HIGH BYTE
 3197  007134  000240                  SYNC72: NOP
 3198  007136                          IUT72:
 3199  007136  121541                          CMPB    (R5),-(R1)      ;EXECUTE INSTRUCTION UNDER TEST
 3200  007140  001405                          BEQ     TST73           ;;TEST OK, GO TO NEXT TEST
 3201  007142  020127  001166                  CMP     R1,#$TMP2       ;DID R1 DECREMENT?
 3202  007146  001001                          BNE     1$              ;BRANCH IF YES
 3203  007150  000000                          HALT                    ;STATE D45.90 DID NOT DECREMENT
 3204                                                                  ;FOR LOOPING CHANGE TO 'BR TST72+2' (756)
 3205  007152                          1$:
 3206  007152  000000                          HALT                    ;INSTRUCTION FAILED
 3207                                                                  ;FOR LOOPING CHANGE TO 'BR TST72+2' (755)
 3208
```

```
                                ;;*****************************************************************
                                ;*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.
                                ;;*****************************************************************


                                ;;*****************************************************************
                                ;*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
                                ;;*****************************************************************
007154  005200                  TST73:  INC     R0              ;INCREMENT TEST NUMBER
007156  012705  100000                  MOV     #BIT15,R5       ;SET SIGN BIT IN R5
007162  012701  001166                  MOV     #$TMP2,R1       ;PUT ADDRESS OF $TMP2 IN R1
007166  010167  171772                  MOV     R1,$TMP1        ;PUT ADDRESS OF $TMP2 IN $TMP1
007172  005011                        CLR     (R1)            ;ENSURE $TMP2 CLEAR
007174  000240                  SYNC73: NOP
007176                          IUT73:
007176  010551                        MOV     R5,@-(R1)       ;EXECUTE INSTRUCTION UNDER TEST
007200  005767  171762                TST     $TMP2           ;DID SIGN BIT GET SET IN $TMP2?
007204  100405                        BMI     TST74           ;;BRANCJ IF YES
007206  020567  171752                CMP     R5,$TMP1        ;DID MODE 4 GET EXECUTED?
007212  001001                        BNE     1$
007214  000000                        HALT                    ;EITHER IRCD DM357 IS NOT GOING LOW
                                                                ;OR NOT GETTING THRU TO RACK E51
                                                                ;FOR LOOPING CHANGE TO 'BR TST73+2' (760)
007216                          1$:
007216  000000                        HALT                    ;INSTRUCTION FAILED
                                                                ;FOR LOOPING CHANGE TO 'BR TST73+2' (757)


                                ;;*****************************************************************
                                ;*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.
                                ;;*****************************************************************


                                        .SBTTL
                                ;;*****************************************************************
                                ;*TEST 74       SEVEN MICROSTATES (DAC*DM7*O/CLASS)
                                ;*
                                ;*      FORK A SHOULD NOT FAIL.
                                ;*
                                ;*      IF IRCD DM357 DOES NOT GO HIGH A DM6 WILL BE EXECUTED.
```

```
3265                                    ;*
3266                                    ;*      ALL OTHER LOGIC HAS BEEN TESTED.
3267                                    ;*
3268                                    ;*      ROM FLOW-7,251,162,231,233,311,157
3269                                    ;;*****************************************************************
3270  007220  005200                    TST74:  INC     R0              ;INCREMENT TEST NUMBER
3271  007222  012705  001166                    MOV     #$TMP2,R5       ;PUT ADDRESS OF $TMP2 IN R5
3272  007226  010567  171732                    MOV     R5,$TMP1        ;PUT ADDRESS OF $TMP2 IN $TMP1
3273  007232  012701  001162                    MOV     #$TMP0,R1       ;PUT ADDRESS OF $TMP0 IN R1
3274  007236  012702  100000                    MOV     #BIT15,R2       ;SET SIGN BIT IN R2
3275  007242  005067  171720                    CLR     $TMP2           ;ENSURE $TMP2 CLEAR
3276  007246  000240                    SYNC74: NOP
3277  007250                            IUT74:
3278  007250  010271  000002                    MOV     R2,a2(R1)       ;EXECUTE INSTRUCTION UNDER TEST
3279  007254  005767  171706                    TST     $TMP2           ;DID $TMP2 GET SIGN BIT SET?
3280  007260  100405                            BMI     TST75           ;;BRANCH IF YES
3281  007262  020267  171676                    CMP     R2,$TMP1        ;DID DM6 GET EXECUTED?
3282  007266  001001                            BNE     1$              ;BRANCH IF NO
3283  007270  000000                            HALT                    ;IRCD DM357 DID NOT GO HIGH
3284                                                                    ;FOR LOOPING CHANGE TO 'BR TST74+2' (754)
3285  007272                            1$:
3286  007272  000000                            HALT                    ;INSTRUCTION FAILED
3287                                                                    ;FOR LOOPING CHANGE TO 'BR TST74+2' (753)
3288
3289
3290                                    ;;*****************************************************************
3291                                    ;*THE LOGICAL SEQUENCE WOULD NEXT TEST A NEG.B*DM12*DR0(1) THEN A NEG.B*DM4*DR0(0)
3292                                    ;*THEN A JMP*DM5 INSTRUCTION, BUT NO ADDITIONAL LOGIC IS TESTED.
3293                                    ;;*****************************************************************
3294
3295
3296                                    ;;*****************************************************************
3297                                    ;*TEST 75        SEVEN MICROSTATES (JSR*DM12)
3298                                    ;*
3299                                    ;*      IF FORK A FAILS EXECUTION WILL GO TO RSD.00 CAUSING A TRAP TO LOCATION 10.
3300                                    ;*      THIS WILL ONLY OCCUR IF RACE JMP+JSR+SWAB DOES NOT GO HIGH.
3301                                    ;*
3302                                    ;*      IF EITHER IRCB IR(14:9)04 DOES NOT GO LOW OR E63 IS BAD
3303                                    ;*      EXECUTION WILL GO FROM D12.10 TO EXC.00.
3304                                    ;*      IF IRCB E63 IS BAD (PIN 10 OR 485 FLOATING) EXECUTION WILL
3305                                    ;*      GO TO RSD.00 CAUSING A TRAP TO LOCATION 10. THIS FAILURE
3306                                    ;*      WOULD INCREMENT THE DST REG. BEFORE THE TRAP.
3307                                    ;*
3308                                    ;*      IF THE INSTRUCTION FAILS THEN ONE OF THE JSR STATES FAILED.
3309                                    ;*
3310                                    ;*      ROM FLOW-2,135,34,201,274,275,32
3311                                    ;;*****************************************************************
3312  007274  005200                    TST75:  INC     R0              ;INC TST NUMBER
3313  007276  012706  001076                    MOV     #1076,SP        ;INITIALIZE SP
3314  007302  012705  007322                    MOV     #T67,R5         ;PUT ADDRESS OF T67A IN R5
3315  007306  010701                            MOV     PC,R1           ;PUT RANDOM NUMBER IN R1
3316  007310                            SYNC75:
3317  007310  000277                    T67A:   SCC                     ;ENSURE ALL CC'S SET
3318  007312                            IUT75:
3319  007312  004125                            JSR     R1,(R5)+        ;EXECUTE INSTRUCTION UNDER TEST
3320  007314  100001                    T67B:   BPL     T67C            ;BRANCH IF N CLEARED
```

```
3321  007316  000000                        HALT                    ;PCB DID NOT LOAD
3322                                                                  ;FOR LOOPING CHANGE TO 'BR TST75+2'' (767)
3323  007320                        T67C:
3324  007320  000000                        HALT                    ;FORK B FAILED TO EXC.00(SEE ABOVE)
3325                                                                  ;FOR LOOPING CHANGE TO 'BR TST75+2'' (766)
3326  007322  022716  007310        T67:    CMP     #T67A,(SP)      ;DID R1 GET STACKED?
3327  007326  001401                        BEQ     4$              ;BRANCH IF YES
3328  007330  000000                        HALT                    ;REGISTER DID NOT GET STACKED
3329                                                                  ;FOR LOOPING CHANGE TO 'BR TST75+2'' (762)
3330  007332  022701  007314        4$:     CMP     #T67B,R1                ;DID R1 GET LOADED?
3331  007336  001401                        BEQ     5$              ;BRANCH IF YES
3332  007340  000000                        HALT                    ;REGISTER DID NOT LOAD
3333                                                                  ;FOR LOOPING CHANGE TO 'BR TST75+2'' (756)
3334  007342  022706  001074        5$:     CMP     #1074,SP        ;DID SP GET DECREMENTED?
3335  007346  001401                        BEQ     TST76           ;;BRANCH IF YES
3336  007350  000000                        HALT                    ;SP DID NOT DECREMENT
3337                                                                  ;FOR LOOPING CHANGE TO 'BR TST75+2'' (752)
3338
3339
3340                                ;;*****************************************************************
3341                                ;*THE LOGICAL SEQUENCE WOULD NEXT EXECUTE A BIN*SM3*DM0*-DF7*SR0(1) THEN
3342                                ;*A BIN*SM3*DM0*DF7*SR0(0) THEN A BIN*SM3*DM0*DF7*SR0(1) INSTRUCTION,
3343                                ;*BUT NO ADDITIONAL LOGIC IS TESTED.
3344                                ;;*****************************************************************
3345
3346
3347                                ;;*****************************************************************
3348                                ;*TEST 76        SEVEN MICROSTATES (BIN*SM5*DM0*-DF7*SR0(0))
3349                                ;*
3350                                ;*      FORK A SHOULD NOT FAIL.
3351                                ;*
3352                                ;*      IF BEN14*FEN4 FAILS EXECUTION WILL GO TO D00.90
3353                                ;*      CAUSING A SM4 INSTRUCTION TO BE EXECUTED. THIS WILL ONLY
3354                                ;*      OCCUR IF EITHER IRCC SRCM5 DOES NOT GO LOW OR IF IRCC E28 IS BAD.
3355                                ;*
3356                                ;*      ROM FLOW-24,23,27,317,143,146,205
3357                                ;;*****************************************************************
3358  007352  005200                TST76:  INC     R0              ;INCREMENT TEST NUMBER
3359  007354  012705  001166                MOV     #$TMP2,R5       ;PUT ADDRESS OF $TMP2 IN R5
3360  007360  010567  171600                MOV     R5,$TMP1        ;PUT ADDRESS OF $TMP2 IN $TMP1
3361  007364  012715  100000                MOV     #BIT15,(R5)     ;SET SIGN BIT IN $TMP2
3362  007370  005001                        CLR     R1              ;ENSURE R1 CLEAR
3363  007372  000240                SYNC76: NOP
3364  007374                        IUT76:
3365  007374  015501                        MOV     @-(R5),R1       ;EXECUTE INSTRUCTION UNDER TEST
3366  007376  022701  100000                CMP     #BIT15,R1       ;DID INSTRUCTION WORK?
3367  007402  001405                        BEQ     TST77           ;;BRANCH IF YES
3368  007404  020167  171554                CMP     R1,$TMP1        ;DID MODE 4 EXECUTE?
3369  007410  001001                        BNE     1$              ;BRANCH IF NO
3370  007412  000000                        HALT                    ;EITHER IRCC SRCM5 NOT GOING LOW OR IRCC E28 BAD
3371                                                                  ;FOR LOOPING CHANGE TO 'BR TST76+2'' (760)
3372  007414                        1$:
3373  007414  000000                        HALT                    ;INSTRUCTION FAILED
3374                                                                  ;FOR LOOPING CHANGE TO 'BR TST76+2'' (757)
3375
3376                                ;;*****************************************************************
```

```
                                ;*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.
                                ;;*****************************************************************


                                ;;*****************************************************************
                                ;*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
                                ;;*****************************************************************
007416  005200                TST77:  INC     R0              ;INCREMENT TEST NUMBER
007420  005067  171542                CLR     $TMP2           ;ENSURE $TMP2 CLEAR
007424  012705  001164                MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
007430  012715  001166                MOV     #$TMP2,(R5)     ;PUT ADDRESS OF $TMP2 IN $TMP1
007434  000240              SYNC77: NOP
007436                    IUT77:
007436  011535                      MOV     (R5),@(R5)+     ;EXECUTE INSTRUCTION UNDER TEST
007440  024515                      CMP     -(R5),(R5)      ;DID INSTRUCTION WORK?
007442  001405                      BEQ     TST100          ;;BRANCH IF YES
007444  022745  001166              CMP     #$TMP2,-(R5)    ;DID DM2 GET EXECUTED?
007450  001001                      BNE     1$              ;BRANCH IF NO
007452  000000                      HALT                    ;EITHER C FORK MUX INPUT B2
                                                            ;NOT GOING LOW OR IRCC E40 BAD
                                                            ;FOR LOOPING CHANGE TO 'BR TST77+2' (762)
007454                    1$:
007454  000000                      HALT                    ;INSTRUCTION FAILED
                                                            ;FOR LOOPING CHANGE TO 'BR TST77+2' (761)


                                ;;*****************************************************************
                                ;*THE LOGICAL SEQUENCE WOULD NEXT TEST A BIN*SM12*DM4*SR0(0)*DR0(0)*
                                ;*P/CLASS FOLLOWED BY A BIN*SM12*DM4*SR0(0)*DR0(1)*[TST.B+BIT.B+CMP.B]
                                ;*FOLLOWED BY A BIN*SM12*DM4*SR0(1)*DR0(0)*P/CLASS FOLLOWED BY A
                                ;*BIN*SM12*DM4*SR0(1)*DR0(1)*[TST.B+BIT.B+CMP.B] INSTRUCTION, BUT NO
                                ;*ADDITIONAL LOGIC IS TESTED.
                                ;;*****************************************************************


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

```
 3433  007456  005200                  TST100: INC     R0              ;INCREMENT TEST NUMBER
 3434  007460  012706  001076                  MOV     #1076,SP        ;INITIALIZE THE SP
 3435  007464  012746  000340                  MOV     #PR7,-(SP)      ;PUT PRIORITY LEVEL 7 ON STACK
 3436  007470  012746  007526                  MOV     #T73,-(SP)      ;PUT ADDRESS OF T73 ON STACK
 3437  007474  005067  171466                  CLR     $TMP2           ;ENSURE $TMP2 CLEAR
 3438  007500  012705  001164                  MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
 3439  007504  012715  100000                  MOV     #BIT15,(R5)     ;SET SIGN BIT IN $TMP1
 3440  007510  000240                  SYN100: NOP
 3441  007512                          IUT100:
 3442  007512  011565  000002                  MOV     (R5),2(R5)      ;EXECUTE INSTRUCTION UNDER TEST
 3443  007516  005767  171444                  TST     $TMP2           ;DID INSTRUCTION WORK?
 3444  007522  100402                          BMI     TST101          ;;BRANCH IF YES
 3445  007524  000000                          HALT                    ;INSTRUCTION FAILED
 3446                                                                  ;FOR LOOPING CHANGE TO 'BR TST100+2' (755)
 3447  007526                          T73:
 3448  007526  000000                          HALT                    ;IRCC E39(5) IS NOT GOING LOW
 3449                                                                  ;FOR LOOPING CHANGE TO 'BR TST100+2' (754)
 3450
 3451
 3452                                  ;;*********************************************************************
 3453                                  ;*THE LOGICAL SEQUENCE WOULD NEXT TEST THE INSTRUCTIONS BETWEEN
 3454                                  ;*BIN*SM4*DM12*SR0(0)*DR0(1)*[TST.B+BIT.B+CMP.B] AND BIN*SM5*DM0*DF7*SR0(1)
 3455                                  ;*BUT NOT ADDITIONAL LOGIC IS TESTED.
 3456                                  ;;*********************************************************************
 3457
 3458
 3459                                          .SBTTL
 3460                                  ;;*********************************************************************
 3461                                  ;*TEST 101       EIGHT MICROSTATES (BIN*SM7*DM0*-DF7*SR0(0))
 3462                                  ;*
 3463                                  ;*      FORK A SHOULD NOT FAIL.
 3464                                  ;*
 3465                                  ;*      IF FEN4*BEN14 FAILS EXECUTION WILL GO TO D00.90 CAUSING
 3466                                  ;*      A SM6 TO BE EXECUTED. THIS WILL ONLY HAPPEN IF EITHER
 3467                                  ;*      IRCC SRCM7 DOES NOT GO LOW OR IRCC E28(1) IS BAD.
 3468                                  ;*
 3469                                  ;*      ROM FLOW-26,54,141,142,317,143,146,205
 3470                                  ;;*********************************************************************
 3471  007530  005200                  TST101: INC     R0              ;INCREMENT TEST NUMBER
 3472  007532  012767  001164  171426          MOV     #$TMP1,$TMP2    ;PUT ADDRESS OF $TMP1 IN $TMP2
 3473  007540  012767  100000  171416          MOV     #BIT15,$TMP1    ;SET SIGN BIT IN $TMP1
 3474  007546  012705  001164                  MOV     #$TMP1,R5       ;PUT ADDRESS OF $TMP1 IN R5
 3475  007552  005001                          CLR     R1              ;ENSURE R1 CLEAR
 3476  007554  000240                  SYN101: NOP
 3477  007556                          IUT101:
 3478  007556  017501  000002                  MOV     @2(R5),R1       ;EXECUTE INSTRUCTION UNDER TEST
 3479  007562  005701                          TST     R1              ;DID R1 GET SIGN BIT SET?
 3480  007564  100405                          BMI     TST102          ;;BRANCH IF YES
 3481  007566  022701  001164                  CMP     #$TMP1,R1       ;DID SRCM6 GET EXECUTED?
 3482  007572  001001                          BNE     1$              ;BRANCH IF NO
 3483  007574  000000                          HALT                    ;EITHER IRCC SRCM7 DOES NOT GO LOW OR IRCC E28 BAD
 3484                                                                  ;FOR LOOPING CHANGE TO 'BR TST101+2' (756)
 3485  007576                          1$:
 3486  007576  000000                          HALT                    ;INSTRUCTION FAILED
 3487                                                                  ;FOR LOOPING CHANGE TO 'BR TST101+2' (755)
 3488
```

```
3489
3490                                    ;;*****************************************************************
3491                                    ;*THE LOGICAL SEQUENCE WOULD NEXT TEST A BIN*SM12*DM3*SR0(0)*DR0(0)*[TST.B+
3492                                    ;*BIT.B+CMP.B] BUT NO ADDITIONAL LOGIC IS TESTED.
3493                                    ;;*****************************************************************
3494
3495
3496                                    ;;*****************************************************************
3497                                    ;*TEST 102      EIGHT MICROSTATES (BIN*SM12*DM3*SR0(1)*DR0(0)*CMPB)
3498                                    ;*
3499                                    ;*      THE ONLY POSSIBLE FAILURE WOULD BE IN STATE D30.90
3500                                    ;*      SINCE ALL THE OTHER LOGIC HAS BEEN TESTED.
3501                                    ;*
3502                                    ;*      ROM FLOW-21,27,112,221,233,311,177,33
3503                                    ;;*****************************************************************
3504  007600  005200                    TST102: INC     R0              ;INCREMENT TEST NUMBER
3505  007602  012767  001166  171354            MOV     #$TMP2,$TMP1    ;PUT ADDRESS OF $TMP2 IN $TMP1
3506  007610  012767  000377  171350            MOV     #377,$TMP2      ;SET LOW BYTE OF $TMP2 TO ALL ONES
3507  007616  012767  177400  171336            MOV     #177400,$TMP0   ;SET HIGH BYTE OF $TMP0 TO ALL ONES
3508  007624  012705  001163                    MOV     #$TMP0+1,R5     ;PUT ADDRESS OF $TMP0 HIGH BYTE IN R5
3509  007630  012701  001164                    MOV     #$TMP1,R1       ;PUT ADDRESS OF $TMP1 IN R1
3510  007634  000240                    SYN102: NOP
3511  007636                            IUT102:
3512  007636  121531                            CMPB    (R5),@(R1)+     ;EXECUTE INSTRUCTION UNDER TEST
3513  007640  001401                            BEQ     TST103          ;;BRANCH IF TEST OK
3514  007642  000000                            HALT                    ;STATE D30.90 FAILED
3515                                                                    ;FOR LOOPING CHANGE TO 'BR TST102+2' (757)
3516
3517
3518                                    ;;*****************************************************************
3519                                    ;*THE LOGICAL SEQUENCE WOULD NEXT TEST INSTRUCTIONS BIN*SM12*DM6*SR0(0)*DR0(1)*
3520                                    ;*P/CLASS THRU BIN*SM12*DM6*SR0(0)*DR0(0)*[TST.B+BIT.B+CMP.B] BUT NO
3521                                    ;*ADDITIONAL LOGIC IS TESTED.
3522                                    ;;*****************************************************************
3523
3524
3525                                    ;;*****************************************************************
3526                                    ;*TEST 103      EIGHT MICROSTATES (BIN*SM12*DM6*SR0(1)*DR0(0)*CMPB)
3527                                    ;*
3528                                    ;*      THE ONLY POSSIBLE FAILURE IN THIS TEST IS STATE D67.90.
3529                                    ;*
3530                                    ;*      ROM FLOW-21,27,116,6,251,122,177,33
3531                                    ;;*****************************************************************
3532  007644  005200                    TST103: INC     R0              ;INCREMENT TEST NUMBER
3533  007646  012767  177400  171310            MOV     #177400,$TMP1   ;SET HIGH BYTE OF $TMP1 TO ALL ONES
3534  007654  012705  001165                    MOV     #$TMP1+1,R5     ;PUT ADDRESS OF $TMP1 HI BYTE IN R5
3535  007660  012767  000377  171300            MOV     #377,$TMP2      ;SET LOW BYTE OF $TMP2 TO ALL ONES
3536  007666  000240                    SYN103: NOP
3537  007670                            IUT103:
3538  007670  121567  171272                    CMPB    (R5),$TMP2      ;EXECUTE INSTRUCTION UNDER TEST
3539  007674  001401                            BEQ     TST104          ;;BRANCH IF TEST OK
3540  007676  000000                            HALT                    ;STATE D67.90 FAILED
3541                                                                    ;FOR LOOPING CHANGE TO 'BR TST103+2' (763)
3542                                            .SBTTL
3543                                    ;;*****************************************************************
3544                                    ;*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
                                        ;;*****************************************************************
007700  005200                  TST104: INC     R0                      ;INCREMENT THE TEST NUMBER
007702  012705  001162                  MOV     #$TMP0,R5       ;PUT ADDRESS OF $TMP0 IN R5
007706  012767  000377  171246          MOV     #377,$TMP0      ;SET LOW BYTE OF $TMP0 TO ALL ONES
007714  012701  001166                  MOV     #$TMP2,R1       ;PUT ADDRESS OF $TMP2 IN R1
007720  012767  000377  171240          MOV     #377,$TMP2      ;SET LOW BYTE OF $TMP2 TO ALL ONES
007726  012767  001166  171230          MOV     #$TMP2,$TMP1    ;PUT ADDRESS OF $TMP2 IN $TMP1
007734  000240                  SYN104: NOP
007736                          IUT104:
007736  021551                          CMP     (R5),@-(R1)     ;EXECUTE INSTRUCTION UNDER TEST
007740  001401                          BEQ     TST105          ;;GO TO NEXT TEST
007742  000000                          HALT                    ;STATE D50.20 IS BAD
                                                                ;FOR LOOPING CHANGE TO 'BR TST104+2' (757)
                                        ;;*****************************************************************
                                        ;*TEST 105      EIGHT MICROSTATES (BIN*SM12*DM5*SR0(1)*DR0(0)*CMPB)
                                        ;*
                                        ;*      THE ONLY POSSIBLE FAILURE IN THIS TEST IS STATE D50.30.
                                        ;;*****************************************************************
007744  005200                  TST105: INC     R0                      ;INCREMENT THE TEST NUMBER
007746  012705  001163                  MOV     #$TMP0+1,R5     ;PUT ADDRESS OF $TMP0 HIGH BYTE IN R5
007752  012767  177400  171202          MOV     #177400,$TMP0   ;SET HIGH BYTE OF $TMP0 TO ALL ONES
007760  012701  001166                  MOV     #$TMP2,R1       ;PUT ADDRESS OF $TMP2 IN R1
007764  012767  000377  171174          MOV     #377,$TMP2      ;SET LOW BYTE OF $TMP2 TO ALL ONES
007772  012767  001166  171164          MOV     #$TMP2,$TMP1    ;PUT ADDRESS OF $TMP2 IN $TMP1
010000  121551                          CMPB    (R5),@-(R1)     ;EXECUTE INSTRUCTION UNDER TEST
010002  001401                          BEQ     TST106          ;;GO TO NEXT TEST
010004  000000                          HALT                    ;STATE D50.30 IS BAD
                                                                ;FOR LOOPING CHANGE TO 'BR TST105+2' (760)

                                        ;;*****************************************************************
                                        ;*TEST 106      WRITE/READ PSW
                                        ;*
                                        ;*      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.
                                        ;;*****************************************************************
010006  005200                  TST106: INC     R0                      ;INCREMENT THE TEST NUMBER
010010  012737  000240  177776          MOV     #PR5,@#PSW      ;SET PRIORITY BITS WITH A DATO
010016  012701  000257                  MOV     #257,R1         ;PUT VALUE OF CC'S IN R1
010022  000277                          SCC                     ;SET ALL THE CC'S WITH A CCOP INSTR
010024  020137  177776                  CMP     R1,@#PSW        ;EXECUTE TEST MODE
010030  001416                          BEQ     TST107          ;;BRANCH IF READ PSW WORKS
010032  012701  177776                  MOV     #PSW,R1         ;GET ADDRESS OF PSW
010036  000277                          SCC                     ;SET ALL THE CONDITION CODES
010040  020137  177776                  CMP     R1,@#PSW        ;DID DMUX SELECT BUS REG?
010044  001001                          BNE     1$              ;BRANCH IF NO
010046  000000                          HALT                    ;EITHER TMCD E28 BAD OR SCCE PS ADDRESS
                                                                ;NOT GETTING TO TMCD *S 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' (764)
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' (754)


                        ;;*****************************************************************
                        ;*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' (731)
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' (725)
010270  000257                3$:     CCC
010272  022737  000240  177776        CMP     #240,@#PSW      ;DID NEW PSW LOAD PROPERLY?
010300  001427                        BEQ     TST111          ;;BRANCH IF YES
010302  000000                        HALT                    ;EITHER PSW DID NOT GET LOADED
                                                              ;PROPERLY OR PSW <7:5> BITS
                                                              ;DO NOT WORK
                                                              ;FOR LOOPING CHANGE TO 'BR TST110+2' (717)
010304                          4$:
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' (716)
010306                          5$:
010306  000000                        HALT                    ;EITHER DAPE TV02 IS NOT GOING HIGH
                                                              ;OR IT IS NOT GETTING TO THE ALU
                                                              ;FOR LOOPING CHANGE TO 'BR TST110+2' (715)
010310                          6$:
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' (714)
010312                          7$:
010312  000000                        HALT                    ;EITHER DAPE TV04 IS STUCK HIGH OR
                                                              ;THE LOW IS NOT GETTING TO THE ALU
```

```
3713
3714  010314                                  8$:                             ;FOR LOOPING CHANGE TO 'BR TST110+2'' (713)
3715  010314  000000                                  HALT                    ;DAPE E24(B0) STUCK HIGH (CHIP FAILURE)
3716                                                                          ;FOR LOOPING CHANGE TO 'BR TST110+2'' (712)
3717  010316                                  9$:
3718  010316  000000                                  HALT                    ;DAPE E25(B0) STUCK HIGH(CHIP FAILURE)
3719                                                                          ;FOR LOOPING CHANGE TO 'BR TST110+2'' (711)
3720  010320                                  10$:
3721  010320  000000                                  HALT                    ;DAPE TV05+07 STUCK HIGH
3722                                                                          ;FOR LOOPING CHANGE TO 'BR TST110+2'' (710)
3723                                          ;NOTE:  THE ONLY WAY THE TRAP INSTRUCTION SHOULD FAIL IS THAT IT GETS
3724                                          ;THE WRONG VECTOR. IF THIS HAPPENS A HALT IN LOW CORE SHOULD OCCUR.
3725  010322  012737  010352  000034                  MOV     #11$,@#TRAPVEC  ;PUT ADDRESS OF 11$ IN TRAP VECTOR
3726  010330  012737  010346  000010                  MOV     #12$,@#RESVEC   ;SETUP RESVEC
3727  010336  012737  010350  000014                  MOV     #13$,@#BPTVEC   ;SETUP LOCATION 14
3728  010344  104777                                  TRAP    377             ;EXECUTE INSTRUCTION UNDER TEST
3729  010346                                  12$:
3730  010346  000000                                  HALT                    ;EITHER IRCD TRAP DOES NOT GO LOW
3731                                                                          ;OR IT IS NOT GETTING THRU DAPE
3732                                                                          ;FOR LOOPING CHANGE TO 'BR TST110+2'' (675)
3733  010350                                  13$:
3734  010350  000000                                  HALT                    ;DAPE E7 IS BAD
3735                                                                          ;FOR LOOPING CHANGE TO 'BR TST110+2'' (674)
3736  010352  012737  000036  000034  11$:    MOV     #36,@#TRAPVEC   ;RESTORE .+2 TO TRAP VECTOR
3737                                          ;;*****************************************************************
3738                                          ;*TEST 111       IOT
3739                                          ;*
3740                                          ;*      FORK A SHOULD NOT FAIL.
3741                                          ;*
3742                                          ;*      IF THE TRAP VECTOR LOGIC FAILS THE TRAP VECTOR COULD COME
3743                                          ;*      OUT TO BE 0, 4, OR 24.
3744                                          ;*
3745                                          ;*      THE ONLY OTHER POSSIBLE FAILURE WOULD BE STATE TRP.01.
3746                                          ;*      IF THIS STATE FAILS TO LOAD THE DR THE TRAP VECTOR WILL
3747                                          ;*      BE WHATEVER IS IN R4.  IF IT FAILS TO LOAD THE BR THE OLD
3748                                          ;*      PS WILL FAIL TO BE STACKED.
3749                                          ;*
3750                                          ;*      ROM FLOW-14,354,(SVC.00-SVC.90) 355,65,357,360,367,37,25,41,222,300
3751                                          ;;*****************************************************************
3752  010360  005200                          TST111: INC     R0              ;INCREMENT TEST NUMBER
3753  010362  012737  010440  000020                  MOV     #1$,@#20        ;SETUP THE IOT VECTOR
3754  010370  012706  001100                          MOV     #STACK,SP       ;SETUP THE SP
3755  010374  012737  010460  000000                  MOV     #2$,@#0         ;SETUP LOCATION ZERO
3756  010402  012737  010462  000014                  MOV     #3$,@#14        ;SETUP LOCATION 14
3757  010410  012704  000014                          MOV     #14,R4          ;SETUP R4
3758  010414  012737  010464  000024                  MOV     #4$,@#24        ;SETUP LOCATION 24
3759  010422  012737  010466  000004                  MOV     #5$,@#ERRVEC    ;SET UP LOCATION 4
3760  010430  012737  000300  177776                  MOV     #PR6,@#PSW      ;SETUP THE PSW
3761  010436  000004                                  IOT                     ;EXECUTE INSTRUCTION UNDER TEST
3762  010440  042766  177437  000002  1$:     BIC     #177437,2(SP)   ;MASK OF THE PRIORITY BITS ON THE OLD PSW
3763  010446  022766  000300  000002                  CMP     #300,2(SP)      ;DID OLD SP GET STACKED?
3764  010454  001405                                  BEQ     SEQENC          ;:BRANCH IF YES
3765  010456  000000                                  HALT                    ;STATE TRP.01 FAILED TO LOAD BR
3766                                                                          ;FOR LOOPING CHANGE TO 'BR TST111+2'' (741)
3767  010460                                  2$:
3768  010460  000000                                  HALT                    ;EITHER DAPE TV04 DOES NOT GO HIGH
```

```
3769                                                    ;OR IT DOES NOT GET TO THE ALU.
3770                                                    ;FOR LOOPING CHANGE TO 'BR TST111+2'' (740)
3771  010462                            3$:
3772  010462  000000                           HALT       ;STATE TRP.01 FILED TO LOAD DR
3773                                                    ;FOR LOOPING CHANGE TO 'BR TST111+2'' (737)
3774  010464                            4$:
3775  010464  000000                           HALT       ;EITHER IRCD IOT DOES NOT GET TO
3776                                                    ;DAPE E7(4) AS A LOW OR E' IS BAD
3777                                                    ;FOR LOOPING CHANGE TO 'BR TST111+2'' (736)
3778  010466                            5$:
3779  010466  000000                           HALT       ;EITHER IRCD IOT DOES NOT GO LOW
3780                                                    ;OR IT DOES NOT GET TO DAPE
3781                                                    ;FOR LOOPING CHANGE TO 'BR TST111+2'' (735)
3782  010470  016737  170404  177570  SEQENC:  MOV      $PASS,@#SWR     ;DISPLAY PASS COUNT IN LIGHTS
3783  010476  005737  000042                   TST      @#42            ;ACT 11 OR XXDP LOAD?
3784  010502  001050                           BNE      1$              ;BRANCH IF YES
3785  010504  012737  011734  000034           MOV      #$TRAP,@#TRAPVEC ;SETUP TRAP VECTOR FOR PRINT
3786  010512  005227  177777                   INC      #-1             ;;FIRST TIME?
3787  010516  001023                           BNE      64$             ;;BRANCH IF NO
3788  010520  022737  011014  000042           CMP      #$ENDAD,@#42    ;;ACT-11?
3789  010526  001417                           BEQ      64$             ;;BRANCH IF YES
3790  010530  104400  010536                   TYPE     ,65$            ;;TYPE ASCIZ STRING
3791  010534  000414                           BR       64$             ;;GET OVER THE ASCIZ
3792                                  ;;65$:   .ASCIZ   <CRLF>?CEKBAD0 11/70 CPU #1?<CRLF><CRLF>
3793  010566                          64$:
3794  010566  005767  170312                   TST      $ICNT           ;CHANGED PASS COUNT YET?
3795  010572  001407                           BEQ      2$              ;BRANCH IF NO
3796  010574  022767  000001  000164           CMP      #1,$EOPCT       ;EOP COUNT AT 1 YET?
3797  010602  001010                           BNE      1$              ;BRANCH IF NO
3798  010604  005067  170274                   CLR      $ICNT           ;CLEAR CHANGED FLAG
3799  010610  000405                           BR       1$
3800  010612  005267  170266          2$:      INC      $ICNT           ;SET CHANGED FLAG
3801  010616  012767  001000  000142           MOV      #1000,$EOPCT    ;CHANGE PASS COUNT
3802  010624  022700  000111          1$:      CMP      #111,R0         ;IS TEST NUMBER OK?
3803  010630  001443                           BEQ      3$              ;BRANCH IF YES
3804  010632  012737  011734  000034           MOV      #$TRAP,@#TRAPVEC ;SETUP TRAP VECTOR
3805  010640  104400  010646                   TYPE     ,67$            ;;TYPE ASCIZ STRING
3806  010644  000414                           BR       66$             ;;GET OVER THE ASCIZ
3807                                  ;;67$:   .ASCIZ   <15><12>/TEST NUMBER(R0) IS  /
3808  010676                          66$:
3809  010676  010067  170252                   MOV      R0,$REG0
3810  010702  016746  170246                   MOV      $REG0,-(SP)     ;;SAVE $REG0 FOR TYPEOUT
3811  010706  104402                           TYPOC                    ;;GO TYPE--OCTAL ASCII(ALL DIGITS)
3812  010710  104400  010716                   TYPE     ,69$            ;;TYPE ASCIZ STRING
3813  010714  000411                           BR       68$             ;;GET OVER THE ASCIZ
3814                                  ;;69$:   .ASCIZ   /  SHOULD BE 111/<15><12>
3815  010740                          68$:
3816  010740  005037  177746          3$:      CLR      @#CONTRL        ;ENABLE CACHE
3817                                  ;;*****************************************************************
3818                                  ;;*****************************************************************
3819
3820                                  .SBTTL  END OF PASS ROUTINE
3821
3822                                  ;*INCREMENT THE PASS NUMBER ($PASS)
3823                                  ;*INDICATE END-OF-PROGRAM AFTER 14 PASSES THRU THE PROGRAM
3824                                  ;*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
                                        ;*SENDMG CAN BE CHANGED TO 7.

010744                                  $EOP:
010744  012737  011734  000034                  MOV     #$TRAP,@#TRAPVEC        ;SETUP TRAP VECTOR
010752  005267  170122                          INC     $PASS           ;;INCREMENT THE PASS NUMBER
010756  042767  100000  170114                  BIC     #100000,$PASS   ;;DON'T ALLOW A NEG. NUMBER
010764  005327                                  DEC     (PC)+           ;;LOOP?
010766  000014                          $EOPCT: .WORD   14
010770  003015                                  BGT     $DOAGN          ;;YES
010772  012737                                  MOV     (PC)+,@(PC)+    ;;RESTORE COUNTER
010774  000014                          $ENDCT: .WORD   14
010776  010766                                  $EOPCT
011000  104400  011030                          TYPE    ,$ENDMG         ;;TYPE 'END PASS'
011004  013700  000042                  $GET42: MOV     @#42,R0         ;;GET MONITOR ADDRESS
011010  001405                                  BEQ     $DOAGN          ;;BRANCH IF NO MONITOR
011012  000005                                  RESET                   ;;CLEAR THE WORLD
011014  004710                          $ENDAD: JSR     PC,(R0)         ;;GO TO MONITOR
011016  000240                                  NOP                     ;;SAVE ROOM
011020  000240                                  NOP                     ;;FOR
011022  000240                                  NOP                     ;;ACT11
011024                                  $DOAGN:
011024  000137  001210                          JMP     @#TST1          ;;RETURN
011030  005015  047105  020104          $ENDMG: .ASCII  <15><12>/END PASS/
011036  040520  051523
011042     377     377     000          $ENULL: .BYTE   -1,-1,0         ;;NULL CHARACTER STRING
        011046                                  .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

011046  105767  170077          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
011052  100002                          BPL     1$              ;;BR IF YES
011054  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
011056  000407                          BR      3$              ;;LEAVE
011060  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
```

```
 3881  011062  017600  000002            MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
 3882  011066  112046            2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
 3883  011070  001005                    BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
 3884  011072  005726                    TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
 3885  011074  012600                    MOV     (SP)+,R0        ;;RESTORE R0
 3886  011076  062716  000002    3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
 3887  011102  000002                    RTI                     ;;RETURN
 3888  011104  122716  000011    4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
 3889  011110  001426                    BEQ     8$
 3890  011112  122716  000200            CMPB    #CRLF,(SP)      ;;BRANCH IF NOT
 3891  011116  001004                    BNE     5$
 3892  011120  005726                    TST     (SP)+           ;;POP  <CR><LF> EQUIV
 3893  011122  104400  001173            TYPE    ,$CRLF
 3894  011126  000757                    BR      2$              ;;GET NEXT CHARACTER
 3895  011130  004767  000056    5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
 3896  011134  126726  170010    6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
 3897  011140  001352                    BNE     2$              ;;IF NO GO GET NEXT CHAR.
 3898  011142  016746  170000            MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
 3899                                                            ;;AND THE NULL CHAR.
 3900  011146  105366  000001    7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
 3901  011152  002770                    BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
 3902  011154  004767  000032            JSR     PC,$TYPEC       ;;GO TYPE A NULL
 3903  011160  105367  000072            DECB    $CHARCNT        ;;DON'T COUNT THE NULL AS A CHARACTER
 3904  011164  000770                    BR      7$              ;;LOOP
 3905
 3906                                    ;;HORIZONTAL TAB PROCESSOR
 3907
 3908  011166  112716  000040    8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
 3909  011172  004767  000014    9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
 3910  011176  132767  000007  000052    BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 3911  011204  001372                    BNE     9$              ;;TAB STOP
 3912  011206  005726                    TST     (SP)+           ;;POP SPACE OFF STACK
 3913  011210  000726                    BR      2$              ;;GET NEXT CHARACTER
 3914  011212  105777  167724    $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 3915  011216  100375                    BPL     $TYPEC
 3916  011220  116677  000002  167716    MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 3917  011226  122766  000015  000002    CMPB    #CR,2(SP)       ;;BRANCH IF
 3918  011234  001003                    BNE     1$              ;;NOT <CR>
 3919  011236  105067  000014            CLRB    $CHARCNT        ;;
 3920  011242  000406                    BR      $TYPEX          ;;EXIT
 3921  011244  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;BRANCH IF
 3922  011252  001402                    BEQ     $TYPEX          ;;<LF>
 3923  011254  105227                    INCB    (PC)+           ;;INC SPACE
 3924  011256  000000            $CHARCNT:.WORD  0               ;;COUNT
 3925  011260  000207            $TYPEX: RTS     PC
 3926
 3927                                    ;;**************************************************************
 3928
 3929                                    .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE
 3930
 3931                                    ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 6-DIGIT
 3932                                    ;*OCTAL (ASCII) NUMBER AND TYPE IT.
 3933                                    ;*$TYPOS--ENTER HERE TO SETUP SUPPRESS ZEROS AND NUMBER OF DIGITS TO TYPE
 3934                                    ;*CALL:
 3935                                    ;*      MOV     NUM,-(SP)       ;;NUMBER TO BE TYPED
 3936                                    ;*      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
011262  017646  000000          $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
011266  116667  000001  000211          MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
011274  112667  000207                  MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
011300  062716  000002                  ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
011304  000406                          BR      $TYPON
011306  112767  000001  000171  $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
011314  112767  000006  000165          MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
011322  112767  000005  000154  $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
011330  010346                          MOV     R3,-(SP)        ;;SAVE R3
011332  010446                          MOV     R4,-(SP)        ;;SAVE R4
011334  010546                          MOV     R5,-(SP)        ;;SAVE R5
011336  116704  000145                  MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
011342  005404                          NEG     R4
011344  062704  000006                  ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
011350  110467  000132                  MOVB    R4,$OMODE       ;;SAVE IT FOR USE
011354  116704  000125                  MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
011360  016605  000012                  MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
011364  005003                          CLR     R3              ;;CLEAR THE OUTPUT WORD
011366  006105                  1$:     ROL     R5              ;;ROTATE MSB INTO 'C'
011370  000404                          BR      3$              ;;GO DO MSB
011372  006105                  2$:     ROL     R5              ;;FORM THIS DIGIT
011374  006105                          ROL     R5
011376  006105                          ROL     R5
011400  010503                          MOV     R5,R3
011402  006103                  3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
011404  105367  000076                  DECB    $OMODE          ;;TYPE THIS DIGIT?
011410  100016                          BPL     7$              ;;BR IF NO
011412  042703  177770                  BIC     #177770,R3      ;;GET RID OF JUNK
011416  001002                          BNE     4$              ;;TEST FOR 0
011420  005704                          TST     R4              ;;SUPPRESS THIS 0?
011422  001403                          BEQ     5$              ;;BR IF YES
011424  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
011426  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
011432  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
011436  110367  000040                  MOVB    R3,8$           ;;SAVE FOR TYPING
011442  104400  011502                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
011446  105367  000032          7$:     DECB    $OCNT           ;;COUNT BY 1
011452  003347                          BGT     2$              ;;BR IF MORE TO DO
011454  002402                          BLT     6$              ;;BR IF DONE
011456  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
```

```
 3993  011460  000744                        BR      2$              ;;GO DO THE LAST DIGIT
 3994  011462  012605                6$:     MOV     (SP)+,R5        ;;RESTORE R5
 3995  011464  012604                        MOV     (SP)+,R4        ;;RESTORE R4
 3996  011466  012603                        MOV     (SP)+,R3        ;;RESTORE R3
 3997  011470  016666  000002  000004       MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 3998  011476  012616                        MOV     (SP)+,(SP)
 3999  011500  000002                        RTI                     ;;RETURN
 4000  011502     000                8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 4001  011503     000                        .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 4002  011504     000                $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 4003  011505     000                $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
 4004  011506  000000                $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
 4005                                ;;**************************************************************************
 4006
 4007                                .SBTTL  CONVERT BINARY TO DECIMAL AND TYPE ROUTINE
 4008
 4009                                ;*THIS ROUTINE IS USED TO CHANGE A 16-BIT BINARY NUMBER TO A 5-DIGIT
 4010                                ;*SIGNED DECIMAL (ASCII) NUMBER AND TYPE IT. DEPENDING ON WHETHER THE
 4011                                ;*NUMBER IS POSITIVE OR NEGATIVE A SPACE OR A MINUS SIGN WILL BE TYPED
 4012                                ;*BEFORE THE FIRST DIGIT OF THE NUMBER. LEADING ZEROS WILL ALWAYS BE
 4013                                ;*REPLACED WITH SPACES.
 4014                                ;*CALL:
 4015                                ;*      MOV     NUM,-(SP)       ;;PUT THE BINARY NUMBER ON THE STACK
 4016                                ;*      TYPDS                   ;;GO TO THE ROUTINE
 4017
 4018  011510                        $TYPDS:
 4019  011510  010046                        MOV     R0,-(SP)        ;;PUSH R0 ON STACK
 4020  011512  010146                        MOV     R1,-(SP)        ;;PUSH R1 ON STACK
 4021  011514  010246                        MOV     R2,-(SP)        ;;PUSH R2 ON STACK
 4022  011516  010346                        MOV     R3,-(SP)        ;;PUSH R3 ON STACK
 4023  011520  010546                        MOV     R5,-(SP)        ;;PUSH R5 ON STACK
 4024  011522  012746  020200                MOV     #20200,-(SP)    ;;SET BLANK SWITCH AND SIGN
 4025  011526  016605  000020                MOV     20(SP),R5       ;;GET THE INPUT NUMBER
 4026  011532  100004                        BPL     1$              ;;BR IF INPUT IS POS.
 4027  011534  005405                        NEG     R5              ;;MAKE THE BINARY NUMBER POS.
 4028  011536  112766  000055  000001       MOVB    #'-,1(SP)       ;;MAKE THE ASCII NUMBER NEG.
 4029  011544  005000                1$:     CLR     R0              ;;ZERO THE CONSTANTS INDEX
 4030  011546  012703  011724                MOV     #$DBLK,R3       ;;SETUP THE OUTPUT POINTER
 4031  011552  112723  000040                MOVB    #' ,(R3)+       ;;SET THE FIRST CHARACTER TO A BLANK
 4032  011556  005002                2$:     CLR     R2              ;;CLEAR THE BCD NUMBER
 4033  011560  016001  011714                MOV     $DTBL(R0),R1    ;;GET THE CONSTANT
 4034  011564  160105                3$:     SUB     R1,R5           ;;FORM THIS BCD DIGIT
 4035  011566  002402                        BLT     4$              ;;BR IF DONE
 4036  011570  005202                        INC     R2              ;;INCREASE THE BCD DIGIT BY 1
 4037  011572  000774                        BR      3$
 4038  011574  060105                4$:     ADD     R1,R5           ;;ADD BACK THE CONSTANT
 4039  011576  005702                        TST     R2              ;;CHECK IF BCD DIGIT=0
 4040  011600  001002                        BNE     5$              ;;FALL THROUGH IF 0
 4041  011602  105716                        TSTB    (SP)            ;;STILL DOING LEADING 0'S?
 4042  011604  100407                        BMI     7$              ;;BR IF YES
 4043  011606  106316                5$:     ASLB    (SP)            ;;MSD?
 4044  011610  103003                        BCC     6$              ;;BR IF NO
 4045  011612  116663  000001  177777       MOVB    1(SP),-1(R3)    ;;YES--SET THE SIGN
 4046  011620  052702  000060        6$:     BIS     #'0,R2          ;;MAKE THE BCD DIGIT ASCII
 4047  011624  052702  000040        7$:     BIS     #' ,R2          ;;MAKE IT A SPACE IF NOT ALREADY A DIGIT
 4048  011630  110223                        MOVB    R2,(R3)+        ;;PUT THIS CHARACTER IN THE OUTPUT BUFFER
```

```
 4049  011632  005720                        TST     (R0)+           ;;JUST INCREMENTING
 4050  011634  020027  000010                CMP     R0,#10          ;;CHECK THE TABLE INDEX
 4051  011640  002746                        BLT     2$              ;;GO DO THE NEXT DIGIT
 4052  011642  003002                        BGT     8$              ;;GO TO EXIT
 4053  011644  010502                        MOV     R5,R2           ;;GET THE LSD
 4054  011646  000764                        BR      6$              ;;GO CHANGE TO ASCII
 4055  011650  105726                8$:     TSTB    (SP)+           ;;WAS THE LSD THE FIRST NON-ZERO?
 4056  011652  100003                        BPL     9$              ;;BR IF NO
 4057  011654  116663  177777  177776        MOVB    -1(SP),-2(R3)   ;;YES--SET THE SIGN FOR TYPING
 4058  011662  105013                9$:     CLRB    (R3)            ;;SET THE TERMINATOR
 4059  011664  012605                        MOV     (SP)+,R5        ;;POP STACK INTO R5
 4060  011666  012603                        MOV     (SP)+,R3        ;;POP STACK INTO R3
 4061  011670  012602                        MOV     (SP)+,R2        ;;POP STACK INTO R2
 4062  011672  012601                        MOV     (SP)+,R1        ;;POP STACK INTO R1
 4063  011674  012600                        MOV     (SP)+,R0        ;;POP STACK INTO R0
 4064  011676  104400  011724                TYPE    ,$DBLK          ;;NOW TYPE THE NUMBER
 4065  011702  016666  000002  000004        MOV     2(SP),4(SP)     ;;ADJUST THE STACK
 4066  011710  012616                        MOV     (SP)+,(SP)
 4067  011712  000002                        RTI                     ;;RETURN TO USER
 4068  011714  023420                $DTBL:  10000.
 4069  011716  001750                        1000.
 4070  011720  000144                        100.
 4071  011722  000012                        10.
 4072  011724  000004                $DBLK:  .BLKW   4
 4073                                  ;;*****************************************************************
 4074
 4075                                  .SBTTL  TRAP DECODER
 4076
 4077                                  ;*THIS ROUTINE WILL PICKUP THE LOWER BYTE OF THE "TRAP" INSTRUCTION
 4078                                  ;*AND USE IT TO INDEX THROUGH THE TRAP TABLE FOR THE STARTING ADDRESS
 4079                                  ;*OF THE DESIRED ROUTINE. THEN USING THE ADDRESS OBTAINED IT WILL
 4080                                  ;*GO TO THAT ROUTINE.
 4081
 4082  011734  010046                $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
 4083  011736  016600  000002                MOV     2(SP),R0        ;;GET TRAP ADDRESS
 4084  011742  005740                        TST     -(R0)           ;;BACKUP BY 2
 4085  011744  111000                        MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
 4086  011746  016000  011754                MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
 4087  011752  000200                        RTS     R0              ;;GO TO ROUTINE
 4088
 4089
 4090                                  .SBTTL  TRAP TABLE
 4091
 4092                                  ;*THIS TABLE CONTAINS THE STARTING ADDRESSES OF THE ROUTINES CALLED
 4093                                  ;*BY THE "TRAP" INSTRUCTION.
 4094
 4095                                  ;       ROUTINE
 4096                                  ;       -------
 4097  011754                        $TRPAD:
 4098  011754  011046                        $TYPE   ;;CALL=TYPE     TRAP+0(104400)  TTY TYPEOUT ROUTINE
 4099  011756  011306                        $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
 4100  011760  011262                        $TYPOS  ;;CALL=TYPOS    TRAP+4(104404)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
 4101  011762  011322                        $TYPON  ;;CALL=TYPON    TRAP+6(104406)  TYPE OCTAL NUMBER (AS PER LAST CALL)
 4102  011764  011510                        $TYPDS  ;;CALL=TYPDS    TRAP+10(104410) TYPE DECIMAL NUMBER (WITH SIGN)
 4103  011766  011766                SUBTAB: .WORD   .
 4104          000001                        .END
```

```
A      = 000107   1529#  1531#  1579#  1581#  1646#  1648#  1714#  1715#  1901#  1959#  1960#  2003#  2004#
                  2072#  2073#  2108#  2109#  2126#  2128#  2149#  2150#  2195#  2196#  2220#  2221#  2239#
                  2240#  2274#  2275#  2292#  2293#  2333#  2334#  2373#  2374#  2398#  2399#  2448#  2449#
                  2482#  2483#  2529#  2530#  2585#  2586#  2622#  2623#  2672#  2673#  2700#  2701#  2725#
                  2726#  2754#  2755#  2793#  2794#  2859#  2861#  2917#  2918#  2938#  2939#  2990#  2991#
                  3041#  3042#  3073#  3074#  3107#  3108#  3153#  3154#  3197#  3198#  3235#  3236#  3276#
                  3277#  3316#  3318#  3363#  3364#  3397#  3398#  3440#  3441#  3476#  3477#  3510#  3511#
                  3536#  3537#  3556#  3557#  3633#  3634#
ADCR0   001544     896    897    898    900#   915
ADDR0   002250    1107   1108   1109   1111#  1125
AFTER   006572    2983   2993#  2999   3007   3015
ASLR0   001656     945    946    947    949#   962
ASRR0   001730     972    973    974    976#   989
BAD     005140    2272   2285#  2294   2297
BAD1    005432    2443   2456#
BICR0   002176    1080   1081   1082   1084#  1097
BISR0   002220    1092   1093   1094   1096#  1112
BITR0   002152    1067   1068   1069   1071#  1085
BIT0   = 000001    134#   2892
BIT00  = 000001    124#    134
BIT01  = 000002    123#    133
BIT02  = 000004    122#    132
BIT03  = 000010    121#    131
BIT04  = 000020    120#    130
BIT05  = 000040    119#    129
BIT06  = 000100    118#    128   2848
BIT07  = 000200    117#    127
BIT08  = 000400    116#    126
BIT09  = 001000    115#    125
BIT1   = 000002    133#
BIT10  = 002000    114#
BIT11  = 004000    113#
BIT12  = 010000    112#
BIT13  = 020000    111#
BIT14  = 040000    110#   2697
BIT15  = 100000    109#   1939   2107   2370   2394   2447   2478   2548   2617   2649   2724   2836   2937
                  3190   3231   3274   3361   3366   3439   3473
BIT2   = 000004    132#
BIT3   = 000010    131#
BIT4   = 000020    130#   1537   1681
BIT5   = 000040    129#
BIT6   = 000100    128#
BIT7   = 000200    127#   3194
BIT8   = 000400    126#
BIT9   = 001000    125#
BPTVEC= 000014     141#   3727*
CACHVE= 000114     148#
CLRR0   001460     861    862    863    865#   878
CMPR0   002316    1132   1133   1134   1136#
COMR0   001524     885    886    887    889#   901
CONTRL= 177746     158#   617*   3816*
CPUERR= 177766     171#
CR     = 000015     46#   3917   3927
CRLF   = 000200     47#   3793   3890   3927
DECR0   001614     922    923    924    926#   938
DISPLA= 177570      41#
```

```
DUM    = 000007          1529#  1579#  1646#  1714#  1901#  1959#  2003#  2072#  2108#  2126#  2149#  2195#  2220#
                         2239#  2274#  2292#  2333#  2373#  2398#  2448#  2482#  2529#  2585#  2622#  2672#  2700#
                         2725#  2754#  2793#  2859#  2917#  2938#  2990#  3041#  3073#  3107#  3153#  3197#  3235#
                         3276#  3316#  3363#  3397#  3440#  3476#  3510#  3536#  3556#  3633#
EMTVEC= 000030             144#  3667*  3668*
ERRVEC= 000004             137#  3759*
ER25    003456            1717#
FORKA   003366            1534   1544#
GNS    = ****** U          424   3792   3807   3814   4098   4099   4100   4101   4102
GOOD    005120            2272#  2276
GRA1T   002354            1174   1178#  1194
GRA2T   002410            1193   1197#  1213
HIADRS= 177742             156#
HITMIS= 177752             160#
HT     = 000011            44#   3888   3927
INCRO   001634             933    934    935    937#   950
IOTVEC= 000020             142#
IT      006420            2886   2889#  2891
IUT     004304            1902#
IUT100  007512            3441#
IUT101  007556            3477#
IUT102  007636            3511#
IUT103  007670            3537#
IUT104  007736            3557#
IUT107  010112            3634#
IUT24   003334            1531#
IUT25   003430            1581#
IUT26   003524            1648#
IUT27   003654            1715#
IUT32   004426            1960#
IUT33   004504            2004#
IUT34   004630            2073#
IUT35   004710            2109#
IUT36   004732            2128#
IUT37   004754            2150#
IUT40   005020            2196#
IUT41   005054            2221#
IUT42   005076            2240#
IUT43   005126            2275#  2293#
IUT44   005220            2334#
IUT45   005302            2374#
IUT46   005330            2399#
IUT47   005414            2449#
IUT50   005460            2483#
IUT51   005536            2530#
IUT52   005632            2586#
IUT53   005706            2623#
IUT55   005776            2673#
IUT56   006024            2701#
IUT57   006060            2726#
IUT60   006116            2755#
IUT61   006144            2794#
IUT62   006344            2861#
IUT63   006464            2918#
IUT64   006512            2939#
IUT65   006570            2991#
```

```
IUT66   006722            3042#
IUT67   006750            3074#
IUT70   007004            3108#
IUT71   007062            3154#
IUT72   007136            3198#
IUT73   007176            3236#
IUT74   007250            3277#
IUT75   007312            3318#
IUT76   007374            3364#
IUT77   007436            3398#
JMPTO   006002            2671    2675#
JMP1AD  005032            2193    2205#
JMP2AD  005056            2219    2223#
KDPAR0= 172360             314#
KDPAR1= 172362             315#
KDPAR2= 172364             316#
KDPAR3= 172366             317#
KDPAR4= 172370             318#
KDPAR5= 172372             319#
KDPAR6= 172374             320#
KDPAR7= 172376             321#
KDPDR0= 172320             292#
KDPDR1= 172322             293#
KDPDR2= 172324             294#
KDPDR3= 172326             295#
KDPDR4= 172330             296#
KDPDR5= 172332             297#
KDPDR6= 172334             298#
KDPDR7= 172336             299#
KERSTK= 001100              31#
KIPAR0= 172340             303#
KIPAR1= 172342             304#
KIPAR2= 172344             305#
KIPAR3= 172346             306#
KIPAR4= 172350             307#
KIPAR5= 172352             308#
KIPAR6= 172354             309#
KIPAR7= 172356             310#
KIPDR0= 172300             281#
KIPDR1= 172302             282#
KIPDR2= 172304             283#
KIPDR3= 172306             284#
KIPDR4= 172310             285#
KIPDR5= 172312             286#
KIPDR6= 172314             287#
KIPDR7= 172316             288#
LF    = 000012              45#   3921    3927
LOADRS= 177740            155#
MAINT = 177750            159#
MAPH0 = 170202            398#
MAPH00= 170202            334#    398
MAPH01= 170206            336#    400
MAPH02= 170212            338#    402
MAPH03= 170216            340#    404
MAPH04= 170222            342#    406
MAPH05= 170226            344#    408
```

```
MAPH06= 170232           346#    410
MAPH07= 170236           348#    412
MAPH1 = 170206           400#
MAPH10= 170242           350#
MAPH11= 170246           352#
MAPH12= 170252           354#
MAPH13= 170256           356#
MAPH14= 170262           358#
MAPH15= 170266           360#
MAPH16= 170272           362#
MAPH17= 170276           364#
MAPH2 = 170212           402#
MAPH20= 170302           366#
MAPH21= 170306           368#
MAPH22= 170312           370#
MAPH23= 170316           372#
MAPH24= 170320           374#
MAPH25= 170326           376#
MAPH26= 170332           378#
MAPH27= 170336           380#
MAPH3 = 170216           404#
MAPH30= 170342           382#
MAPH31= 170346           384#
MAPH32= 170352           386#
MAPH33= 170356           388#
MAPH34= 170362           390#
MAPH35= 170366           392#
MAPH36= 170372           394#
MAPH37= 170376           396#
MAPH4 = 170222           406#
MAPH5 = 170226           408#
MAPH6 = 170232           410#
MAPH7 = 170236           412#
MAPL0 = 170200           397#
MAPL00= 170200           333#    397
MAPL01= 170204           335#    399
MAPL02= 170210           337#    401
MAPL03= 170214           339#    403
MAPL04= 170220           341#    405
MAPL05= 170224           343#    407
MAPL06= 170230           345#    409
MAPL07= 170234           347#    411
MAPL1 = 170204           399#
MAPL10= 170240           349#
MAPL11= 170244           351#
MAPL12= 170250           353#
MAPL13= 170254           355#
MAPL14= 170260           357#
MAPL15= 170264           359#
MAPL16= 170270           361#
MAPL17= 170274           363#
MAPL2 = 170210           401#
MAPL20= 170300           365#
MAPL21= 170304           367#
MAPL22= 170310           369#
MAPL23= 170314           371#
```

```
MAPL24= 170320  373#
MAPL25= 170324  375#
MAPL26= 170330  377#
MAPL27= 170334  379#
MAPL3 = 170214  403#
MAPL30= 170340  381#
MAPL31= 170344  383#
MAPL32= 170350  385#
MAPL33= 170354  387#
MAPL34= 170360  389#
MAPL35= 170364  391#
MAPL36= 170370  393#
MAPL37= 170374  395#
MAPL4 = 170220  405#
MAPL5 = 170224  407#
MAPL6 = 170230  409#
MAPL7 = 170234  411#
MEMERR= 177744  157#
MMR0  = 177572  182#  186
MMR1  = 177574  183#  187
MMR2  = 177576  184#  188
MMR3  = 172516  185#  189
MMVEC = 000250  150#
MOVBR0  002126  1047  1048  1049  1050  1056#
MOVR0   002130  1054  1059#  1072
N     = 000735  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#  1167#  1170#  1175#  1186#  1189#  1194#  1205#  1208#  1213#  1221#  1225#  1229#
                1233#  1240#  1244#  1256#  1260#  1266#  1270#  1282#  1286#  1292#  1296#  1308#  1312#
                1318#  1322#  1334#  1338#  1344#  1348#  1360#  1364#  1370#  1374#  1386#  1390#  139(#
                1400#  1435#  1439#  1444#  1448#  1453#  1457#  1462#  1466#  1471#  1475#  1480#  148(#
                1548#  1551#  1556#  1584#  1599#  1602#  1606#  1609#  1613#  1662#  1666#  1673#  1677#
                1684#  1687#  1691#  1696#  1727#  1730#  1745#  1748#  1773#  1779#  1787#  1792#  1801#
                1808#  1817#  1824#  1833#  1840#  1848#  1851#  1860#  1869#  1878#  1914#  1917#  1921#
                1929#  1937#  1963#  1972#  2013#  2016#  2025#  2030#  2033#  2044#  2047#  2051#  2082#
                2086#  2089#  2112#  2131#  2156#  2159#  2200#  2203#  2208#  2225#  2279#  2282#  2286#
                2297#  2308#  2340#  2344#  2349#  2354#  2377#  2405#  2408#  2454#  2457#  2489#  2494#
                2542#  2546#  2552#  2557#  2560#  2592#  2595#  2599#  2628#  2632#  2635#  2653#  2657#
                2676#  2704#  2729#  2733#  2737#  2759#  2800#  2803#  2868#  2872#  2877#  2881#  2887#
                2894#  2921#  2943#  2947#  2997#  3005#  3009#  3013#  3024#  3046#  3078#  3114#  3120#
                3123#  3128#  3160#  3164#  3203#  3206#  3242#  3246#  3283#  3286#  3321#  3324#  3328#
                3332#  3336#  3370#  3373#  3404#  3408#  3445#  3448#  3483#  3486#  3514#  3540#  3560#
                3575#  3599#  3606#  3611#  3638#  3644#  3678#  3682#  3686#  3693#  3698#  3703#  3707#
                3711#  3715#  3718#  3721#  3730#  3734#  3765#  3768#  3772#  3775#  3779#
NEGR5   003446  1586  1587  1588  1590#
NEXTT   005102  2238  2243#
OVER    002344  1162  1172#
OVER2   002400  1181  1191#
OVER3   002434  1200  1210#
PIRQ  = 177772  39#
PIRQVE= 000240  149#
POINT   005634  2581  2588#
POINT1  006350  2829*  2842  2864#
POINT2  007064  3152  3156#  3158
POSERR  003526  1651#  1655  1671  1675
```

```
PR0     = 000000             71#
PR1     = 000040             72#
PR2     = 000100             73#
PR3     = 000140             74#
PR4     = 000200             75#
PR5     = 000240             76#   3590    3668
PR6     = 000300             77#   3760
PR7     = 000340             78#   3435    3630    3641    3642    3665
PS      = 177776             36#     37
PSW     = 177776             37#   1654    2158    2844    2988   3590*   3593    3595    3597    3604    3640   3665*   3691
                           3760*
PWRVEC= 000024              143#
RESVEC= 000010              138#   3669*   3726*
ROLR0   001700              957     958     959     961#
RORR0   001572              910     911     912     914#    927     977
ROUT0   002330             1163#   1167    1170
ROUT1   002364             1182#   1186    1189
ROUT2   002420             1201#   1205    1208
RSOK    003510             1594    1611#
SBCR0   001752              984     985     986     988#   1003
SDPAR0= 172260              270#
SDPAR1= 172262              271#
SDPAR2= 172264              272#
SDPAR3= 172266              273#
SDPAR4= 172270              274#
SDPAR5= 172272              275#
SDPAR6= 172274              276#
SDPAR7= 172276              277#
SDPDR0= 172220              248#
SDPDR1= 172222              249#
SDPDR2= 172224              250#
SDPDR3= 172226              251#
SDPDR4= 172230              252#
SDPDR5= 172232              253#
SDPDR6= 172234              254#
SDPDR7= 172236              255#
SEQENC  010470             3764    3782#
SIPAR0= 172240              259#
SIPAR1= 172242              260#
SIPAR2= 172244              261#
SIPAR3= 172246              262#
SIPAR4= 172250              263#
SIPAR5= 172252              264#
SIPAR6= 172254              265#
SIPAR7= 172256              266#
SIPDR0= 172200              237#
SIPDR1= 172202              238#
SIPDR2= 172204              239#
SIPDR3= 172206              240#
SIPDR4= 172210              241#
SIPDR5= 172212              242#
SIPDR6= 172214              243#
SIPDR7= 172216              244#
SIZEHI= 177762              168#
SIZELO= 177760              166#
SKIP    005122             2271    2273#
```

```
SR0    = 177572            186#
SR1    = 177574            187#
SR2    = 177576            188#
SR3    = 172516            189#
STACK  = 001100             30#    31     32     33   2269   2404   2451   2519   2648   2833   3629   3666   3754
START    001202            428    617#
STKLMT= 177774              38#
SUBR0    002274           1120   1121   1122   1124#  1137
SUBTAB   011766           1720   4103#
SUPSTK= 000700              32#
SWABR0   002000            998    999   1000   1002#  1016
SWR    = 177570             40#    41   3782*
SW0    = 000001            106#
SW00   = 000001             96#   106
SW01   = 000002             95#   105
SW02   = 000004             94#   104
SW03   = 000010             93#   103
SW04   = 000020             92#   102
SW05   = 000040             91#   101
SW06   = 000100             90#   100
SW07   = 000200             89#    99
SW08   = 000400             88#    98
SW09   = 001000             87#    97
SW1    = 000002            105#
SW10   = 002000             86#
SW11   = 004000             85#
SW12   = 010000             84#
SW13   = 020000             83#
SW14   = 040000             82#
SW15   = 100000             81#
SW2    = 000004            104#
SW3    = 000010            103#
SW4    = 000020            102#
SW5    = 000040            101#
SW6    = 000100            100#
SW7    = 000200             99#
SW8    = 000400             98#
SW9    = 001000             97#
SXTR0    002024           1011   1012   1013   1015#  1024
SXT2     002040           1023#  1037
SYNC24   003332           1529#
SYNC25   003426           1579#
SYNC26   003522           1646#
SYNC27   003652           1714#
SYNC31   004302           1901#
SYNC32   004424           1959#
SYNC33   004502           2003#
SYNC34   004626           2072#
SYNC35   004706           2108#
SYNC36   004730           2126#
SYNC37   004752           2149#
SYNC40   005016           2195#
SYNC41   005052           2220#
SYNC42   005074           2239#
SYNC43   005124           2274#  2292#
SYNC44   005216           2333#
```

```
SYNC45  005300          2373#
SYNC46  005326          2398#
SYNC47  005412          2448#
SYNC50  005456          2482#
SYNC51  005534          2529#
SYNC52  005630          2585#
SYNC53  005704          2622#
SYNC55  005774          2672#
SYNC56  006022          2700#
SYNC57  006056          2725#
SYNC60  006114          2754#
SYNC61  006142          2793#
SYNC62  006342          2846    2859#
SYNC63  006462          2917#
SYNC64  006510          2938#
SYNC65  006570          2990#
SYNC66  006720          3041#
SYNC67  006746          3073#
SYNC70  007002          3107#
SYNC71  007060          3153#
SYNC72  007134          3197#
SYNC73  007174          3235#
SYNC74  007246          3276#
SYNC75  007310          3316#
SYNC76  007372          3363#
SYNC77  007434          3397#
SYN100  007510          3440#
SYN101  007554          3476#
SYN102  007634          3510#
SYN103  007666          3536#
SYN104  007734          3556#
SYN107  010110          3633#
SYSTID= 177764           170#
TBITVE= 000014           139#
TESTCC  003410          1554#
TKVEC = 000060           146#
TPVEC = 000064           147#
TRAPVE= 000034           145#   3725*   3736*   3785*   3804*   3831*
TRTVEC= 000014           140#
TSTR0   001502           873     874     875     877#    890
TST1    001210           628#    636    3849
TST10   001370           781     795#    801
TST100  007456          3401    3433#   3445    3448
TST101  007530          3444    3471#   3483    3486
TST102  007600          3480    3504#   3514
TST103  007644          3513    3532#   3540
TST104  007700          3539    3550#   3560
TST105  007744          3559    3567#   3575
TST106  010006          3574    3589#   3599    3606    3611
TST107  010066          3594    3628#   3638    3644
TST11   001410           805     818#    824
TST110  010140          3643    3664#   3678    3682    3686    3693    3698    3703    3707    3711    3715    3718    3721
                        3730    3734
TST111  010360          3692    3752#   3765    3768    3772    3775    3779
TST12   001430           828     840#    846
TST13   001442           844     858#    866
```

TST14	002320	1159#	1175											
TST15	002526	1243	1252#	1256	1260	1266	1270	1435	1439	1444	1448	1453	1457	1462
		1466	1471	1475	1480	1484								
TST16	002566	1269	1278#	1282	1286	1292	1296							
TST17	002626	1295	1304#	1308	1312	1318	1322							
TST2	001226	634	650#	656										
TST20	002666	1321	1330#	1334	1338	1344	1348							
TST21	002726	1347	1356#	1360	1364	1370	1374							
TST22	002766	1373	1382#	1386	1390	1396	1400							
TST23	003026	1399	1409#											
TST24	003306	1519#	1548	1551	1556									
TST25	003414	1555	1574#	1584	1599	1602	1606	1609	1613					
TST26	003516	1612	1644#	1662	1666	1673	1677	1684	1687	1691	1696			
TST27	003646	1695	1712#	1727	1730	1745	1748							
TST3	001246	660	675#	682										
TST30	003742	1737	1767#	1773										
TST31	004274	1877	1898#	1914	1917	1921	1929	1937						
TST32	004416	1956#	1963	1972										
TST33	004460	1971	1994#	2013	2016	2025	2030	2033	2044	2047	2051			
TST34	004612	2024	2068#	2082	2086	2097								
TST35	004670	2076	2103#	2112										
TST36	004716	2111	2122#	2131										
TST37	004740	2130	2145#	2156	2159									
TST4	001270	686	700#	707										
TST40	005004	2154	2191#	2200	2203	2208								
TST41	005044	2206	2218#	2225										
TST42	005066	2224	2237#											
TST43	005104	2268#	2279	2282	2286									
TST44	005202	2306	2328#	2340	2344	2349	2354							
TST45	005262	2353	2368#	2377										
TST46	005310	2376	2393#	2405	2408									
TST47	005352	2401	2436#	2454	2457									
TST5	001312	711	725#	732										
TST50	005434	2453	2476#	2489	2494									
TST51	005476	2485	2518#	2542	2546	2552	2557	2560						
TST52	005612	2533	2579#	2592	2595	2599								
TST53	005660	2598	2615#	2628	2632	2635								
TST54	005734	2625	2647#	2653	2657									
TST55	005766	2656	2670#	2676										
TST56	006004	2695#	2704											
TST57	006032	2703	2718#	2729	2733	2737								
TST6	001334	736	754#	760										
TST60	006102	2736	2750#	2759										
TST61	006126	2758	2789#	2800	2803									
TST62	006166	2797	2826#	2868	2872	2877	2881	2887	2894					
TST63	006436	2893	2911#	2921										
TST64	006472	2920	2933#	2943	2947									
TST65	006532	2946	2980#	2997	3005	3009	3013	3024						
TST66	006702	3023	3037#	3046										
TST67	006734	3045	3069#	3078										
TST7	001350	758	771#	777										
TST70	006764	3077	3103#	3114	3120	3123	3128							
TST71	007046	3127	3150#	3160	3164									
TST72	007104	3157	3189#	3203	3206									
TST73	007154	3200	3230#	3242	3246									
TST74	007220	3239	3270#	3283	3286									

```
TST75   007274            3280    3312#   3321    3324    3328    3332    3336
TST76   007352            3335    3358#   3370    3373
TST77   007416            3367    3393#   3404    3408
TYPDS = 104410            4102#
TYPE  = 104400            3790    3805    3812    3840    3893    3988    4064    4098#
TYPOC = 104402            3811    4099#
TYPON = 104406            4101#
TYPOS = 104404            4100#
T67     007322            3314    3326#
T67A    007310            3317#   3326
T67B    007314            3320#   3330
T67C    007320            3320    3323#
T71     010120            3631    3640#
T73     007526            3436    3447#
UDPAR0= 177660             226#
UDPAR1= 177662             227#
UDPAR2= 177664             228#
UDPAR3= 177666             229#
UDPAR4= 177670             230#
UDPAR5= 177672             231#
UDPAR6= 177674             232#
UDPAR7= 177676             233#
UDPDR0= 177620             204#
UDPDR1= 177622             205#
UDPDR2= 177624             206#
UDPDR3= 177626             207#
UDPDR4= 177630             208#
UDPDR5= 177632             209#
UDPDR6= 177634             210#
UDPDR7= 177636             211#
UIPAR0= 177640             215#
UIPAR1= 177642             216#
UIPAR2= 177644             217#
UIPAR3= 177646             218#
UIPAR4= 177650             219#
UIPAR5= 177652             220#
UIPAR6= 177654             221#
UIPAR7= 177656             222#
UIPDR0= 177600             193#
UIPDR1= 177602             194#
UIPDR2= 177604             195#
UIPDR3= 177606             196#
UIPDR4= 177610             197#
UIPDR5= 177612             198#
UIPDR6= 177614             199#
UIPDR7= 177616             200#
USESTK= 000600              33#
XORR0   002064            1032    1033    1034    1036#   1057    1060
Y     = 000767             639#    657#    664#    683#    689#    708#    714#    733#    739#    761#    778#    785#    802#
                           808#    825#    831#    847#    867#   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#   1585#   1780#   1788#   1793#   1802#   1809#   1818#   1825#   1834#
                          1841#   1849#   1852#   1861#   1870#   1879#   2309#
$BDADR  001122             476#
```

```
$BCDAT  001126            478#
$CHARC  011256           3903*  3910   3919*  3924#
$CMTAG  001100            464#
$CM1  = 000003            490#   491#   492#   493#
$CM2  = 000006            490#   491#   492#   493#
$CM3  = 000003            488#   490
$CM4  = 000004            493#   494#   495#   496#   497#
$CRLF   001173            498#  3893   3927
$DBLK   011724           4030   4064   4072#
$DOAGN  011024           3836   3842   3848#
$DTBL   011714           4033   4068#
$ENDAD  011014            451   3788   3844#
$ENDCT  010774           3838#
$ENDMG  011030           3840   3850#
$ENULL  011042           3852#
$EOP    010744           3830#
$EOPCT  010766           3796   3801*  3835#  3839
$ERFLG  001103            467#
$ERMAX  001115            473#
$ERPSW  001176            500#  1654*  1679   1681   2158*
$ERRPC  001116            474#
$ERRTB  001202            519#
$ERTTL  001112            471#
$FILLC  001150            486#  3896   3927
$FILLS  001147            485#  3927
$GDADR  001120            475#
$GDDAT  001124            477#
$GET42  011004           3841#
$HD   = 000002             25
$ICNT   001104            468#  3794   3798*  3800*
$ITEMB  001114            472#
$LF     001174            499#  3927
$LPADR  001106            469#
$LPERR  001110            470#
$NULL   001146            484#  3898   3927
$NWTST= 000001            618#   620   640#   642   665#   667   690#   692   715#   717   745#   747   762#
                          764    786#   788   809#   811   832#   834   848#   850  1141#  1143  1246#  1248
                         1272#  1274  1298#  1300  1324#  1326  1350#  1352  1376#  1378  1402#  1404  1509#
                         1511   1559#  1561  1616#  1618  1698#  1700  1750#  1752  1880#  1882  1941#  1943
                         1974#  1976  2035#  2037  2091#  2093  2114#  2116  2133#  2135  2170#  2172  2210#
                         2212   2227#  2229  2244#  2246  2311#  2313  2356#  2358  2380#  2382  2418#  2420
                         2459#  2461  2497#  2499  2565#  2567  2602#  2604  2637#  2639  2659#  2661  2681#
                         2683   2706#  2708  2739#  2741  2779#  2781  2805#  2807  2904#  2906  2923#  2925
                         2950#  2952  3026#  3028  3059#  3061  3091#  3093  3139#  3141  3178#  3180  3218#
                         3220   3259#  3261  3296#  3298  3347#  3349  3383#  3385  3421#  3423  3460#  3462
                         3496#  3498  3525#  3527  3543#  3545  3562#  3564  3578#  3580  3613#  3615  3655#
                         3657   3737#  3739
$OCNT   011504           3960*  3989*  4002#
$OMODE  011506           3955*  3959*  3964   3967*  3978*  4004#
$PASS   001100            465#  2845   3782   3832*  3833*  3850
$QUES   001172            497#
$RDCHR= ****** U         4103
$RDDEC= ****** U         4103
$RDLIN= ****** U         4103
$RDOCT= ****** U         4103
$REGAD  001152            488#
```

```
$REG0   001154          490#  2155*  2994*  3809*  3810
$REG1   001156          491#
$REG2   001160          492#
$RTI    005610         2522   2559#
$SAVRE= ****** U        4103
$SETUP= 000024          417#  3788   3818#  3832
$STUP = 177777          417#  3818#
$SVPC = 000204          449#   454
$SWR  = 160000           25#   497    629    651    676    701    726    755    772    796    819    841    859
                        1160  1253   1279   1305   1331   1357   1383   1410   1520   1575   1645   1713   1768
                        1899  1957   1995   2069   2104   2123   2146   2192   2219   2238   2269   2329   2369
                        2394  2437   2477   2519   2580   2616   2648   2671   2696   2719   2751   2790   2827
                        2912  2934   2981   3038   3070   3104   3151   3190   3231   3271   3313   3359   3394
                        3434  3472   3505   3533   3551   3568   3590   3629   3665   3753   3825   3832   3843
                        3849  3850
$TKB    001140          481#
$TKS    001136          480#
$TMP0   001162          493#  3273   3507*  3508   3551   3552*  3568   3569*
$TMP1   001164          494#  1907   1909   1912   2000   2008   2009   2042   2069   2078*  2079   2084   2105
                        2124  2147   2153   2329   2336   2346   2369   2395   2402   2445   2477   2616   2696
                        2719  2751   2790   2834   2912   2934   2982   3002   3022   3038   3072*  3075*  3076
                        3104* 3105*  3106   3151   3191*  3233*  3240   3272*  3281   3360*  3368   3395   3438
                        3472  3473*  3474   3481   3505*  3509   3533*  3534   3555*  3572*
$TMP2   001166          495#  1996   2630   2720   2798   2835   2914   2941   3192   3201   3232   3238   3271
                        3275* 3279   3359   3394*  3396   3402   3437*  3443   3472*  3505   3506*  3535*  3538
                        3553  3554*  3555   3570   3571*  3572
$TMP3   001170          496#
$TN   = 000112            1#    25    618    629#   634    636    640    651#   656    660    665    676#   682
                         686   690   701#   707    711    715    726#   732    736    745    755#   758    760
                         762   772#  777    781    786    796#   801    805    809    819#   824    828    832
                         841#  844    846    848    859#   866   1141   1160#  1175   1243   1246   1253#  1256
                        1260  1266   1269   1270   1272   1279#  1282   1286   1292   1295   1296   1298   1305#
                        1308  1312   1318   1321   1322   1324   1331#  1334   1338   1344   1347   1348   1350
                        1357# 1360   1364   1370   1373   1374   1376   1383#  1386   1390   1396   1399   1400
                        1402  1410#  1509   1520#  1529   1531   1548   1551   1555   1556   1559   1575#  1579
                        1581  1584   1599   1602   1606   1609   1612   1613   1616   1645#  1646   1648   1662
                        1666  1673   1677   1684   1687   1691   1695   1696   1698   1713#  1714   1715   1727
                        1730  1737   1745   1748   1750   1768#  1773   1877   1880   1899#  1901   1914   1917
                        1921  1929   1937   1941   1957#  1959   1960   1963   1971   1972   1974   1995#  2003
                        2004  2013   2016   2024   2025   2030   2033   2035   2044   2047   2051   2069#  2072
                        2073  2076   2082   2086   2089   2091   2104#  2108   2109   2111   2112   2114   2123#
                        2126  2128   2130   2131   2133   2146#  2149   2150   2154   2156   2159   2170   2192#
                        2195  2196   2200   2203   2206   2208   2210   2219#  2220   2221   2224   2225   2227
                        2238# 2239   2240   2244   2269#  2274   2275   2279   2282   2286   2292   2293   2306
                        2311  2329#  2333   2334   2340   2344   2349   2353   2354   2356   2369#  2373   2374
                        2376  2377   2380   2394#  2398   2399   2401   2405   2408   2418   2437#  2448   2449
                        2453  2454   2457   2459   2477#  2482   2483   2485   2489   2494   2497   2519#  2529
                        2532  2533   2542   2546   2552   2557   2560   2565   2580#  2585   2586   2592   2595
                        2598  2599   2602   2616#  2622   2623   2625   2628   2632   2635   2637   2648#  2653
                        2656  2657   2659   2671#  2672   2673   2676   2681   2696#  2700   2701   2703   2704
                        2706  2719#  2725   2726   2729   2733   2736   2737   2739   2751#  2754   2755   2758
                        2759  2779   2790#  2793   2794   2797   2800   2803   2805   2827#  2859   2861   2868
                        2872  2877   2881   2887   2893   2894   2904   2912#  2917   2918   2920   2921   2923
                        2934# 2938   2939   2943   2946   2947   2950   2981#  2990   2991   2997   3005   3009
                        3013  3023   3024   3026   3038#  3041   3042   3045   3046   3059   3070#  3073   3074
                        3077  3078   3091   3104#  3107   3108   3114   3120   3123   3127   3128   3139   3151#
```

```
                    3153   3154   3157   3160   3164   3178   3190#  3197   3198   3200   3203   3206   3218
                    3231#  3235   3236   3239   3242   3246   3259   3271#  3276   3277   3280   3283   3286
                    3296   3313#  3316   3318   3321   3324   3328   3332   3335   3336   3347   3359#  3363
                    3364   3367   3370   3373   3383   3394#  3397   3398   3401   3404   3408   3421   3434#
                    3440   3441   3444   3445   3448   3460   3472#  3476   3477   3480   3483   3486   3496
                    3505#  3510   3511   3513   3514   3525   3533#  3536   3537   3539   3540   3543   3551#
                    3556   3557   3559   3560   3562   3568#  3574   3575   3578   3590#  3594   3599   3606
                    3611   3613   3629#  3633   3634   3638   3643   3644   3655   3665#  3678   3682   3686
                    3692   3693   3698   3703   3707   3711   3715   3718   3721   3730   3734   3737   3753#
                    3765   3768   3772   3775   3779
$TPB     001144      483#  2831   2832   3916*  3927
$TPFLG   001151      487#  3876   3927
$TPS     001142      482#  2827   2828   2829   2830   3914   3927
$TRAP    011734     3785   3804   3831   4082#
$TRP   = 000012     4089#  4099#  4100#  4101#  4102#  4103#
$TRPAD   011754     4086   4097#
$TSTNM   001102      466#
$TYPBN=  ******  U  4103
$TYPDS   011510     4018#  4102
$TYPE    011046     3876#  4089   4098
$TYPEC   011212     3895   3902   3909   3914#  3915
$TYPEX   011260     3920   3922   3925#
$TYPOC   011306     3958#  4099
$TYPON   011322     3957   3960#  4101
$TYPOS   011262     3953#  4100
$$GET4=  000000     3843#
$$TN   = 000105      520#   527#   528#   529#   531#   532#   534#   536#   538#   540#   542#   545#   546#
                     548#   550#   551#   552#   553#   555#   556#   558#   559#   561#   563#   564#   565#
                     567#   569#   571#   573#   577#   578#   580#   582#   584#   586#   588#   590#   593#
                     594#   595#   597#   599#   601#   603#   605#   606#   608#   609#
$$TPS    006422     2830*  2890#
$$TRP  = 000002     4088#  4099   4100   4101   4102   4103
$OFILL   011505     3954*  3958*  3968   4003#
.      = 011770      420#   424    426#   449    450#   452#   454#   462#   500    636    656    662    682
                     688    707    713    732    738    760    777    783    801    807    824    830    846
                     864    866    876    878    888    890    899    901    913    915    925    927    936
                     938    948    950    960    962    975    977    987    989   1001   1003   1014   1016
                    1022   1024   1035   1037   1055   1057   1060   1070   1072   1083   1085   1095   1097
                    1110   1112   1123   1125   1135   1137   1167   1170   1175   1186   1189   1194   1205
                    1208   1213   1221   1225   1229   1233   1240   1244   1256   1260   1266   1270   1282
                    1286   1292   1296   1308   1312   1318   1322   1334   1338   1344   1348   1360   1364
                    1370   1374   1386   1390   1396   1400   1435   1439   1444   1448   1453   1457   1462
                    1466   1471   1475   1480   1484   1548   1551   1556   1584   1589   1599   1602   1606
                    1609   1613   1662   1666   1673   1677   1684   1687   1691   1696   1727   1730   1745
                    1748   1773   1779   1787   1792   1801   1808   1817   1824   1833   1840   1848   1851
                    1860   1869   1878   1914   1917   1921   1929   1937   1963   1972   2013   2016   2025
                    2030   2033   2044   2047   2051   2082   2086   2089   2112   2131   2156   2159   2200
                    2203   2208   2225   2279   2282   2286   2297   2308   2340   2344   2349   2354   2377
                    2405   2408   2454   2457   2489   2494   2542   2546   2552   2557   2560   2592   2595
                    2599   2628   2632   2635   2653   2657   2676   2704   2729   2733   2737   2759   2800
                    2803   2868   2872   2877   2881   2887   2894   2921   2943   2947   2997   3005   3009
                    3013   3024   3046   3078   3114   3120   3123   3128   3160   3164   3203   3206   3242
                    3246   3283   3286   3321   3324   3328   3332   3336   3370   3373   3404   3408   3445
                    3448   3483   3486   3514   3540   3560   3575   3599   3606   3611   3638   3644   3678
                    3682   3686   3693   3698   3703   3707   3711   3715   3718   3721   3730   3734   3765
                    3768   3772   3775   3779   3808#  3850   3853#  3927   4072#  4103
```

```
COMA      1662#
COMAA     2377#
COMB      1665#   1666
COMBB     2489#
COMC      1686#   1687
COMCC     3114#
COMD      1929#
COMDD     3774#   3775
COME      1916#   1917    2046#   2047
COMEE     3767#   3768
COMENT    2559#   2560
COMFF     3778#   3779
COMG      2281#   2282
COMGG     3697#   3698
COMH      2281#   2286
COMHH     3702#   3703
COMI      2340#
COMII     3706#   3707
COMJ      2493#   2494
COMJJ     3710#   3711
COMK      2552#
COMKK     3714#   3715
COML      2675#   2676
COMLL     3717#   3718
COMMEN       1#    416#
COMMM     3720#   3721
COMN      3160#
COMNN     3729#   3730
COMO      3242#
COMP      3404#   3686#
COMQ      3693#
COMX      3606#
COMY      3599#
COMZ      3644#
DOC          1#    527     528     529     531     532     534     536     538     540     542     545     546     548     550
           551     552     553     555     556     558     559     561     563     564     565     567     569     571     573
           577     578     580     582     584     586     588     590     593     594     595     597     599     601     603
           605     606     608     609
DOCEXP       1#    527     528     529     531     532     534     536     538     540     542     545     546     548     550
           551     552     553     555     556     558     559     561     563     564     565     567     569     571     573
           577     578     580     582     584     586     588     590     593     594     595     597     599     601     603
           605     606     608     609
ENDCOM       1#    416#
ERROR       34#
ERRORD       1#    635     655     661     681     687     706     712     731     737     759     776     782     800     806
           823     829     845     865     877     889     900     914     926     937     949     961     976     988    1002
          1015    1023    1036    1056    1059    1071    1084    1096    1111    1124    1136    1167    1169    1175    1186
          1188    1194    1205    1207    1213    1221    1225    1229    1233    1240    1244    1256    1260    1266    1270
          1282    1286    1292    1296    1308    1312    1318    1322    1334    1338    1344    1348    1360    1364    1370
          1374    1386    1390    1396    1400    1435    1439    1444    1448    1453    1457    1462    1466    1471    1475
          1480    1484    1548    1550    1556    1584    1599    1601    1606    1608    1613    1662    1665    1673    1677
          1683    1686    1690    1696    1727    1729    1745    1747    1773    1779    1787    1792    1801    1808    1817
          1824    1833    1840    1848    1851    1860    1869    1878    1914    1916    1920    1929    1937    1963    1972
          2013    2015    2025    2030    2032    2044    2046    2050    2082    2086    2088    2112    2131    2156    2159
          2200    2202    2208    2225    2279    2281    2285    2297    2307    2340    2343    2349    2354    2377    2405
          2407    2454    2456    2489    2493    2542    2546    2552    2556    2559    2592    2594    2599    2628    2632
```

	2634	2653	2657	2675	2704	2729	2733	2737	2759	2800	2802	2868	2872	2877	2881
	2887	2894	2921	2943	2947	2997	3005	3009	3013	3024	3046	3078	3114	3120	3122
	3128	3160	3163	3203	3205	3242	3245	3283	3285	3321	3323	3328	3332	3336	3370
	3372	3404	3407	3445	3447	3483	3485	3514	3540	3560	3575	3599	3606	3610	3638
	3644	3678	3682	3686	3693	3697	3702	3706	3710	3714	3717	3720	3729	3733	3765
	3767	3771	3774	3778											
ESCAPE	1#	416#													
HLT	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	1167	1170	1175	1186
	1189	1194	1205	1208	1213	1221	1225	1229	1233	1240	1244	1256	1260	1266	1270
	1282	1286	1292	1296	1308	1312	1318	1322	1334	1338	1344	1348	1360	1364	1370
	1374	1386	1390	1396	1400	1435	1439	1444	1448	1453	1457	1462	1466	1471	1475
	1480	1484	1548	1551	1556	1584	1599	1602	1606	1609	1613	1662	1666	1673	1677
	1684	1687	1691	1696	1727	1730	1745	1748	1773	1779	1787	1792	1801	1808	1817
	1824	1833	1840	1848	1851	1860	1869	1878	1914	1917	1921	1929	1937	1963	1972
	2013	2016	2025	2030	2033	2044	2047	2051	2082	2086	2089	2112	2131	2156	2159
	2200	2203	2208	2225	2279	2282	2286	2297	2308	2340	2344	2349	2354	2377	2405
	2408	2454	2457	2489	2494	2542	2546	2552	2557	2560	2592	2595	2599	2628	2632
	2635	2653	2657	2676	2704	2729	2733	2737	2759	2800	2803	2868	2872	2877	2881
	2887	2894	2921	2943	2947	2997	3005	3009	3013	3024	3046	3078	3114	3120	3123
	3128	3160	3164	3203	3206	3242	3246	3283	3286	3321	3324	3328	3332	3336	3370
	3373	3404	3408	3445	3448	3483	3486	3514	3540	3560	3575	3599	3606	3611	3638
	3644	3678	3682	3686	3693	3698	3703	3707	3711	3715	3718	3721	3730	3734	3765
	3768	3772	3775	3779											
IDN	2881#														
LABEL	1#	1529	1579	1646	1714	1901	1959	2003	2072	2108	2126	2149	2195	2220	2239
	2274	2292	2333	2373	2398	2448	2482	2529	2585	2622	2672	2700	2725	2754	2793
	2859	2917	2938	2990	3041	3073	3107	3153	3197	3235	3276	3316	3363	3397	3440
	3476	3510	3536	3556	3633										
LABEL1	1#	1531	1581	1648	1715	1960	2004	2073	2109	2128	2150	2196	2221	2240	2275
	2293	2334	2374	2399	2449	2483	2530	2586	2623	2673	2701	2726	2755	2794	2831
	2918	2939	2991	3042	3074	3108	3154	3198	3236	3277	3318	3364	3398	3441	3477
	3511	3537	3557	3634											
MSGD1	661#	662													
MSGD10	1324#	1326													
MSGD11	1350#	1352													
MSGD12	1376#	1378													
MSGD13	1402#	1404													
MSGD2	782#	783													
MSGD5	848#	850													
MSGD6	1140#	1143													
MSGD7	1246#	1248													
MSGD8	1272#	1274													
MSGD9	1298#	1300													
MSG1	618#	620													
MSG10	832#	834													
MSG120	3737#	3739													
MSG16	1509#	1511													
MSG17	1559#	1561													
MSG2	640#	642													
MSG20	1616#	1618													
MSG21	1698#	1700													
MSG21A	1750#	1752													
MSG22	1880#	1882													
MSG23	1941#	1943													

```
MSG24    1974#   1976
MSG25    2035#   2037
MSG26    2091#   2093
MSG27    2114#   2116
MSG3      665#    667
MSG30    2133#   2135
MSG31    2170#   2172
MSG32    2210#   2212
MSG33    2244#   2246
MSG34    2311#   2313
MSG35    2356#   2358
MSG36    2380#   2382
MSG37    2418#   2420
MSG4      690#    692
MSG40    2227#   2229
MSG41    2459#   2461
MSG42    2497#   2499
MSG43    2565#   2567
MSG44    2602#   2604
MSG45    2637#   2639
MSG46    2681#   2683
MSG47    2706#   2708
MSG5      715#    717
MSG50    2739#   2741
MSG51    2659#   2661
MSG53    2779#   2781
MSG54    2805#   2807
MSG55    2904#   2906
MSG56    2923#   2925
MSG57    2950#   2952
MSG6      745#    747
MSG60    3026#   3028
MSG61    3059#   3061
MSG62    3091#   3093
MSG63    3139#   3141
MSG64    3178#   3180
MSG65    3218#   3220
MSG66    3259#   3261
MSG67    3296#   3298
MSG7      762#    764
MSG70    3347#   3349
MSG71    3613#   3615
MSG72    3383#   3385
MSG73    3421#   3423
MSG74    3460#   3462
MSG75    3496#   3498
MSG76    3525#   3527
MSG77    3655#   3657
MSG77A   3543#   3545
MSG77B   3562#   3564
MSG77C   3578#   3580
MSG8      786#    788
MSG9      809#    811
MULT        1#    416#
NEWMAC    635#    636
NEWTST      1#    416#  618  640  665  690  715  745  762  786  809  832  848  1141  1246
```

```
        1272  1298  1324  1350  1376  1402  1509  1559  1616  1698  1750  1880  1941  1974  2035
        2091  2114  2133  2170  2210  2227  2244  2311  2356  2380  2418  2459  2497  2565  2602
        2637  2659  2681  2706  2739  2779  2805  2904  2923  2950  3026  3059  3091  3139  3178
        3218  3259  3296  3347  3383  3421  3460  3496  3525  3543  3562  3578  3613  3655  3737
POP       1#  416#  4059
PUSH      1#  416#  4018
SAVEAD    1#
SCOPE    35#
SETTRA 4089#  4099  4100  4101  4102
SETUP     1#  416#
SKIP      1#  416#   634   660   686   711   736   758   781   805   828   844  1212  1243  1269
        1295  1321  1347  1373  1399  1554  1612  1695  1737  1877  1971  2024  2075  2111  2130
        2154  2206  2224  2306  2353  2376  2401  2453  2485  2533  2598  2625  2656  2703  2736
        2758  2797  2893  2920  2946  3023  3045  3077  3127  3157  3200  3239  3280  3335  3367
        3401  3444  3480  3513  3539  3559  3574  3594  3643  3692  3764
SLASH     1#  416#
SPACE   416#
STARS     1#  416#   429   455   503   520   611   618   627   640   649   665   674   690   699
         715   724   740   745   753   762   770   786   794   809   817   832   839   848   857
        1141  1158  1246  1251  1272  1277  1298  1303  1324  1329  1350  1355  1376  1381  1402
        1408  1498  1509  1518  1536  1540  1559  1573  1616  1643  1698  1711  1750  1766  1781
        1794  1810  1826  1842  1853  1862  1871  1880  1897  1941  1955  1974  1993  1998  2002
        2035  2067  2091  2102  2114  2121  2133  2144  2163  2167  2170  2190  2210  2217  2227
        2236  2244  2267  2311  2327  2356  2367  2380  2392  2412  2415  2418  2435  2459  2475
        2497  2517  2565  2578  2602  2614  2637  2646  2659  2669  2681  2694  2706  2717  2739
        2749  2763  2767  2772  2776  2779  2788  2805  2825  2898  2901  2904  2910  2923  2932
        2950  2979  3026  3036  3051  3055  3059  3068  3084  3087  3091  3102  3133  3136  3139
        3149  3168  3175  3178  3188  3210  3215  3218  3229  3250  3255  3259  3269  3290  3293
        3296  3311  3340  3344  3347  3357  3376  3380  3383  3392  3412  3416  3421  3432  3452
        3456  3460  3470  3490  3493  3496  3503  3518  3522  3525  3531  3543  3549  3562  3566
        3578  3588  3613  3627  3650  3654  3655  3663  3737  3751  3817  3818  3854  3927  4005
        4073
TRMTRP  4089#
TYPBIN    1#  416#
TYPDEC    1#  416#
TYPNAM    1#  416#  3786
TYPNUM    1#  416#
TYPOCS    1#  416#
TYPOCT    1#  416#  3810
TYPTXT    1#  416#  3805  3812
$IUT      1#  1531  1581  1648  1715  1960  2004  2073  2109  2128  2150  2196  2221  2240  2275
        2293  2334  2374  2399  2449  2483  2530  2586  2623  2673  2701  2726  2755  2794  2861
        2918  2939  2991  3042  3074  3108  3154  3198  3236  3277  3318  3364  3398  3441  3477
        3511  3537  3557  3634
$SAVEA    1#
$SYNC     1#  1529  1579  1646  1714  1901  1959  2003  2072  2108  2126  2149  2195  2220  2239
        2274  2292  2333  2373  2398  2448  2482  2529  2585  2622  2672  2700  2725  2754  2793
        2859  2917  2938  2990  3041  3073  3107  3153  3197  3235  3276  3316  3363  3397  3440
        3476  3510  3536  3556  3633
$$CMRE  455#   490   491   492
$$CMTM  455#   493   494   495   496
$$ESCA    1#  416#
$$NEWT    1#  416#   618   640   665   690   715   745   762   786   809   832   848  1141  1246
        1272  1298  1324  1350  1376  1402  1509  1559  1616  1698  1750  1880  1941  1974  2035
        2091  2114  2133  2170  2210  2227  2244  2311  2356  2380  2418  2459  2497  2565  2602
        2637  2659  2681  2706  2739  2779  2805  2904  2923  2950  3026  3059  3091  3139  3178
```

```
        3218    3259    3296    3347    3383    3421    3460    3496    3525    3543    3562    3578    3613    3655    3737
$$SET   4089#   4099    4100    4101    4102
$$SKIP     1#    416#     634     660     686     711     736     758     781     805     828     844    1243    1269    1295
        1321    1347    1373    1399    1555    1612    1695    1737    1877    1971    2024    2076    2111    2130    2154
        2206    2224    2306    2353    2376    2401    2453    2485    2533    2598    2625    2656    2703    2736    2758
        2797    2893    2920    2946    3023    3045    3077    3127    3157    3200    3239    3280    3335    3367    3401
        3444    3480    3513    3539    3559    3574    3594    3643    3692
.EQUAT     1#
.HEADE     1#      16
.KT11      1#
.SETUP     1#     417    3818
.SWRHI     1#
.SWRLO     1#
.$ACT1     1#     429
.$CATC     1#     417
.$CMTA     1#     455
.$DB2D     1#
.$DB2O     1#
.$DIV      1#
.$EOP      1#    3818
.$ERRO     1#
.$ERRT     1#
.$MULT     1#
.$PNUE     1#
.$RAND     1#
.$RDDE     1#
.$RDOC     1#
.$READ     1#
.$SAVE     1#
.$SB2D     1#
.$SB2O     1#
.$SCOP     1#
.$SIZE     1#
.$SUPR     1#
.$TRAP     1#    3818#   4073
.$TYPB     1#
.$TYPD     1#    3818#   4005
.$TYPE     1#    3818#   3854
.$TYPO     1#    3818#   3927
.1170      1#      26

. ABS.  011770     000

 ERRORS DETECTED:  0

 CEKBAD.BIN,CEKBAD.LST/CRF/SOL/NL:TOC=CEKBAD.SML,CEKBAD.P11
 RUN-TIME:  71 85 6 SECONDS
 RUN-TIME RATIO: 267/164=1.6
 CORE USED:  35K  (69 PAGES)
```