KD11-K

FLOAT PT UN INST EXE

MD-11-DQFPC-B

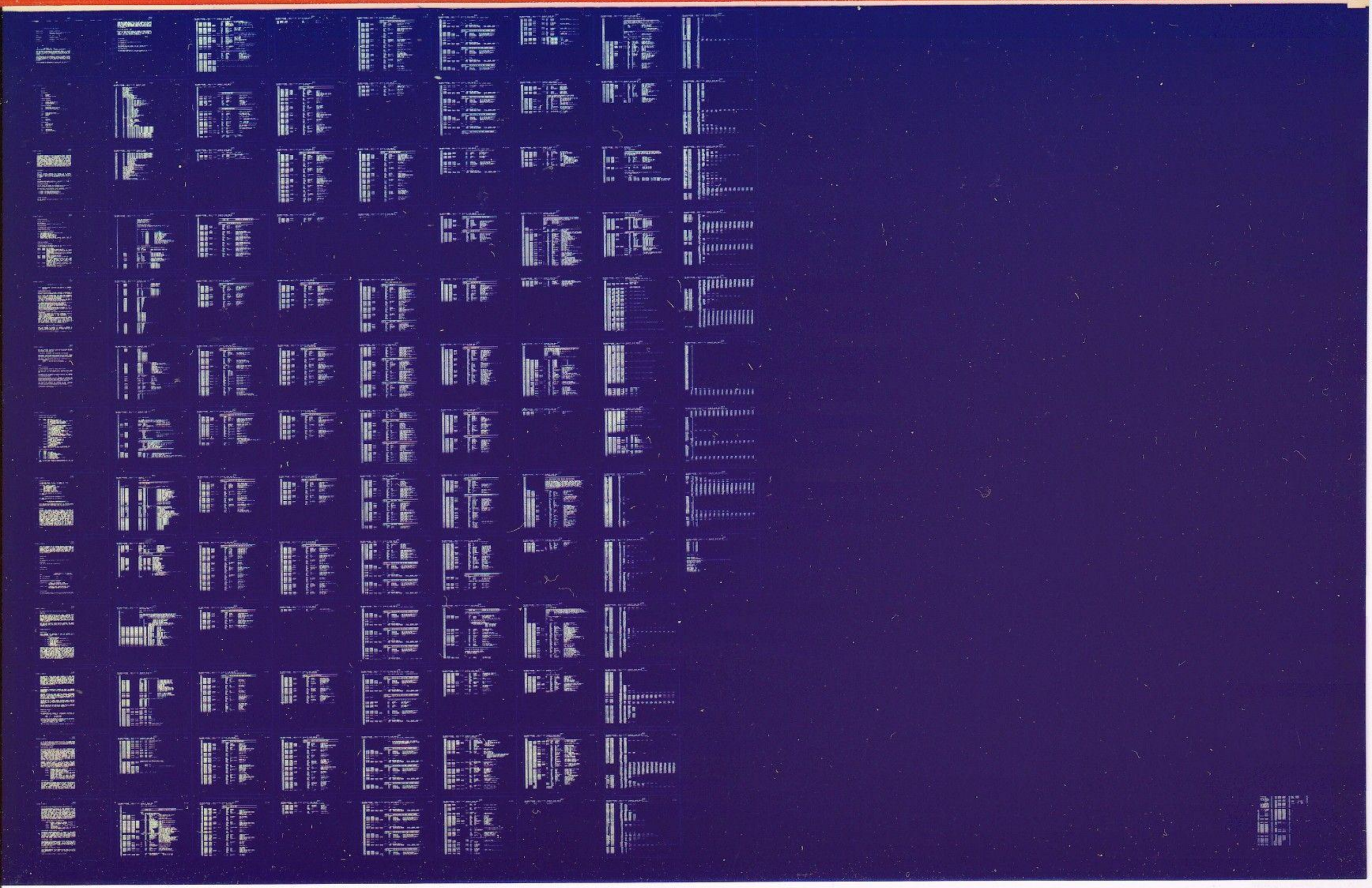
EP-DQFPC-B-DL-A

FICHE 1 OF 1

JUN 1977

digital

MADE IN USA



I D E N T I F I C A T I O N

PRODUCT CODE: MAINDEC-11-D0FPC-B-D

PRODUCT NAME: PDP-11/6X - FP11-E FLOATING POINT UNIT INSTRUCTION EXERCISER

DATE : MAY, 1977

MAINTAINER: DIAGNOSTIC GROUP

AUTHOR: BOB BRAIN / STANLEY HARACKIEWICZ

REVISED BY: DON NORTH

CONTENTS
---

1. ABSTRACT

THIS PROGRAM CONTAINS THE PDP-11/6X FLOATING POINT PROCESSOR INSTRUCTION EXERCISER. THIS EXERCISER PROGRAM TESTS THE COMPLETE FLOATING POINT INSTRUCTION SET IN AN EXERCISER ENVIRONMENT, USING VARIOUS COMBINATIONS OF INSTRUCTIONS IN COMMON SOFTWARE CONFIGURATIONS AS THE BASIS FOR THE TESTS. EVERY CONCEIVABLE FLOATING POINT ERROR CONDITION IS DEVELOPED, AND THE RESPONSE CHECKED FOR CORRECTNESS. IN ADDITION, INTERACTION BETWEEN FLOATING POINT INSTRUCTION EXECUTION, INTERRUPTS, AND BASE MACHINE INTERRUPTS (USING THE DL11-W LINE AND/OR KW11-P PROGRAMMABLE CLOCKS) IS ALSO TESTED TO INSURE CORRECT PROCESSING BY BOTH THE BASE MACHINE AND FLOATING POINT MICROCODE. BOTH "HOT" (FP11-E OPTION) AND "WARM" (PDP-11/6X MICROCODE) FLOATING POINT UNITS CAN BE SELECTED FOR TESTING.

2. REQUIREMENTS

2.1 EQUIPMENT

PDP-11/6X STANDARD COMPUTER WITH MINIMUM 16K OF MEMORY. OPTIONAL FP11-E FLOATING POINT UNIT, IF SELECTED. THE DL11-W LINE CLOCK IS USED TO PROVIDE I/O INTERRUPTS DURING EXECUTION. IF PRESENT, THE KW11-P PROGRAMMABLE CLOCK WILL ALSO BE UTILIZED.

2.2 STORAGE

THE PROGRAM USES MEMORY 0-20772(8). THE UPPER 2.0K WORDS ARE RESERVED FOR THE XXDP MONITOR, IF EMPLOYED.

2.3 PRELIMINARY PROGRAMS

THE CPU, CACHE, AND MEMORY TEST PROGRAMS MUST BE RUN FIRST TO VERIFY THE CORRECT OPERATION OF THE BASE MACHINE.

THE PDP-11/6X - FP11-E FLOATING POINT PROCESSOR INSTRUCTION SET TESTS SHOULD THEN BE RUN IN THE FOLLOWING ORDER:

(1) DQFPA FPU BASIC INSTRUCTION TESTS
(2) DQFPB FPU ADVANCED INSTRUCTION TESTS
(3) DQFPC FPU INSTRUCTION EXERCISER
(4) DQFPD FPU ADD/SUB/MUL/DIV RANDOM EXERCISER

3. LOADING PROCEDURE

USE THE STANDARD PROCEDURE FOR ABSOLUTE TAPES, OR LOAD VIA XXDP MEDIA.

4. STARTING PROCEDURE

4.1 CONTROL SWITCH SETTINGS

SEE SECTION 5.1
SWITCH REGISTER (000000) IS WORST CASE TEST.

4.2 STARTING ADDRESS

THE PROGRAM MUST ALWAYS BE STARTED AT LOCATION 200(8).

4.3 PROGRAM/OPERATOR ACTION

LOADING VIA ABSOLUTE PAPERTAPE:

(1) LOAD PROGRAM INTO MEMORY USING ABS LOADER.
(2) LOAD ADDRESS 200 (8).
(3) SET SWITCHES (SEE SECTION 5.1)
SR=(000000) IS WORST CASE TEST.
(4) PRESS CONTROL/START TO BEGIN.
(5) PROGRAM TYPES IDENTIFICATION HEADER (VERIFY THAT THE CORRECT PROGRAM HAS BEEN LOADED!), AND EXECUTION BEGINS.

5. OPERATING PROCEDURE

5.1 OPERATIONAL SWITCH SETTINGS

THE DEFINITION OF THE SPECIFIC BITS IN THE SWITCH REGISTER (EITHER HARDWARE OR SOFTWARE) ARE AS FOLLOWS:

SW15=1	100000	HALT ON ERROR
SW14=1	040000	LOOP ON CURRENTLY EXECUTING TEST
SW13=1	020000	INHIBIT ERROR TYPEOUTS (WHICH IS AN "ERROR MESSAGE" RESULTING FROM AN ERROR DETECTED IN THE HARDWARE)
SW12=1	010000	INHIBIT STATUS TYPEOUTS (WHICH IS A NON-ERROR RELATED INFORMATIVE MESSAGE, SUCH AS "END PASS #XXX")
SW11=1	004000	INHIBIT ITERATIONS PER TEST
SW10	002000	SET=BELL ON ERROR/CLEAR=BELL ON PASS END
SW09=1	001000	LOOP ON ERROR
SW08=1	000400	LOOP ON TEST NUMBER IN "SLPTST" IF SET, THEN THE TEST SPECIFIED BY THE TEST NUMBER CONTAINED IN THE MEMORY WORD "SLPTST" (SEE PROGRAM LISTING) WILL SPECIFY THE DESIRED TEST ON WHICH TO LOOP.
SW01	000002	CLEAR=TEST HOT-FP/WARM-FP ALTERNATELY EACH PASS (IE, PASS#1 HFP, PASS#1 WFP, PASS#2 HFP, PASS#2 WFP, ETC) SET=TEST ONLY UNIT SPECIFIED IN SW00
SW00	000001	SET=SELECT WARM FP, IF SW01=1

CLEAR=SELECT HOT FP, IF SW01=1

NOTE FOR SW01, SW00 - IF NO HOT FP (FP11-E) IS PRESENT, THEN WARM FP (PDP-11/6X MICROCODE) IS AUTOMATICALLY SELECTED.

5.2 PROGRAM/OPERATOR ACTION

ONCE EXECUTION HAS BEGUN, MINIMAL OPERATOR INTERVENTION IS REQUIRED, UNLESS THE PROGRAM DETECTS AN ERROR IN THE HARDWARE.

IF ALL IS WELL, THE PROGRAM TYPES ITS NAME UPON BEGINNING; AND AT THE START OF EACH PASS, THE CURRENT PASS NUMBER (IN OCTAL) IS ECHOED. NOTE THAT SETTING SW<12>=1 WILL INHIBIT THE TYPEOUT OF THE BEGIN AND END PASS MESSAGES.
IF SW<10>=0, THE CONSOLE BELL WILL BE RUNG AT THE END OF EACH PASS. NOTE THAT ONLY SW<10> AFFECTS THE BELL RINGING AT END OF PASS - SW<12> HAS NO EFFECT ON THIS FUNCTION.

IF AN ERROR OCCURS DURING EXECUTION, MANY VARIATIONS IN ACTION ARE POSSIBLE DEPENDING UPON THE SWITCH SETTINGS.
SW<15>=1 WILL CAUSE THE CPU TO HALT AFTER AN ERROR.
SW<13>=1 WILL ALSO INHIBIT ANY ERROR MESSAGE TYPEOUT THAT WOULD OCCUR AT THIS TIME.
SW<10>=1 WILL CAUSE THE CONSOLE BELL TO BE RUNG ONLY WHEN AN ERROR IS DETECTED (AND NOT AT THE END OF A PASS).
SW<9>=1 CAUSES THE PROGRAM TO LOOP ON THE MOST RECENT ERROR, AS LONG AS IT CONTINUES TO OCCUR.

THERE ARE ALSO SEVERAL OTHER GENERAL USE FUNCTIONS DEFINED BY THE SWITCHES:
SW<11>=1 WILL INHIBIT THE ITERATIONS (=2000(10)) PERFORMED OF EACH TEST ON PASSES 2,3,4, ... THRU THE PROGRAM.
SW<14>=1 CAUSES THE PROGRAM TO LOOP INDEFINATELY ON THE CURRENTLY EXECUTING TEST.
SW<8>=1 CAUSES THE PROGRAM TO CONTINUE EXECUTION AS NORMAL, EXCEPT WHEN THE CONTENTS OF MEMORY WORD "SLPTST" MATCHES THE NUMBER OF THE TEST CURRENTLY EXECUTING. AT THIS POINT, THE TEST IS LOOPED ON INDEFINATELY, UNTIL EITHER SW<8>=0 OR "SLPTST" IS CHANGED. NOTE THAT IF "SLPTST" DOES NOT MATCH THE TEST NUMBER OF ANY TEST, THE CONTENTS OF "SLPTST" ARE EFFECTIVELY IGNORED, AND EXECUTION PROCEEDS NORMALLY.

5.3 HOT (FP11-E) / WARM (PDP-11/6X) SELECTION

WHEN THE PROGRAM IS STARTED (AT 200(8)), A MESSAGE IS OPTIONALLY PRINTED INDICATING THE PRESCENCE/ABSCENCE OF AN FP11-E HOT FLOATING POINT UNIT OPTION (BASED UPON WHETHER "WHAMI" BIT<04> IS 1/0 RESPECTIVELY).

IF NO FP11-E HOT FP OPTION IS PRESENT, THE MESSAGE IS TYPED,

AND ANY ATTEMPTS TO SELECT IT FOR TESTING VIA SW01 AND SW00 ARE IGNORED. ONLY WARM FP (PDP-11/6X MICROCODE) FLOATING POINT CAN BE TESTED/SELECTED.

IF THE FP11-E IS PRESENT, TEST SELECTION IS AS FOLLOWS:

WHEN SW01=0, THE HOT AND WARM FLOATING POINT UNITS ARE TESTED ALTERNATELY EACH PASS - IN THE ORDER (1) HOT, THEN (2) WARM. NOTE THAT EACH "PASS" NOW CONSISTS OF TWO SEPARATE SUB-PASSES.

WHEN SW01=1, THEN DEDICATED SELECTION OF A PARTICULAR UNIT IS SPECIFIED IN SW00:

SW00=0 --> TEST HFP FP11-E OPTION ONLY
SW00=1 --> TEST HFP PDP-11/6X MICROCODE ONLY

6. ERRORS

6.1 FORMAT OF MESSAGES

6.1.1 ALL ERROR MESSAGES CONSIST OF THREE LINES OF DATA:

THE FIRST LINE IS A BRIEF MESSAGE WHICH EXPLAINS WHAT ERROR WAS DETECTED (EG, THE RESULT OF THE "ABSF" INSTRUCTION WAS BAD).

THE PREFIX "HOT:" OR "WARM:" IS ALSO ATTACHED TO THE MESSAGE TO INDICATE THE SOURCE OF THE ERROR; THE FP11-E UNIT OR THE PDP-11/6X RESPECTIVELY.

THE SECOND LINE CONSISTS OF DATA HEADERS TO IDENTIFY THE VALUES TYPED OUT ON LINE THREE. THESE HEADERS WILL EITHER BE OF THE FORM "EXPECTED" AND "RECEIVED" DATA, OR WILL BE A MNEMONIC NAME OF A WORD LOCATION IN MEMORY OR REGISTERS.

THE THIRD LINE DISPLAYS THE CONTENTS OF THE LOCATIONS SPECIFIED BY LINE TWO AS SIX DIGIT OCTAL NUMBERS. NOTE THAT ALL DATA DISPLAYED IN ANY MESSAGES ARE OCTAL NUMBERS.

AS EXPLAINED IN SECTION 5.2, SETTING SW<13>=1 WILL SUPPRESS THE TYPING OF THESE MESSAGES.

6.1.2 FLOATING POINT UNIT DATA FORMATS:

FLOATING POINT STATUS WORD (FPS):

BIT##	OCTAL	FUNCTION
15	100000	FER - FLOATING ERROR FLAG SET WHEN EITHER FIUV, FIU, FIV, FIC ENABLED AND APPROPRIATE EXCEPTION OCCURRED.
14	040000	FID - FLOATING DISABLE INTERRUPTS NO FP INTERRUPTS TO VECTOR 244(8) IF SET.
13, 12		NOT USED
11	004000	FIUV - FLOATING UNDEFINED VARIABLE INTERRUPT IF SET, (-0) MEMORY DATA IS ERROR
10	002000	FIU - FLOATING INTR UNDERFLOW IF SET AND UNDERFLOW, SET FER, STORE ANSWER, EXPONENT WRONG BY +400(8) IF CLEAR AND UNDERFLOW, ANSWER <-- ZERO
9	001000	FIV - FLOATING OVERFLOW INTERRUPT IF SET AND OVERFLOW, SET FER, STORE ANSWER, EXPONENT WRONG BY -400(8) IF CLEAR AND OVERFLOW, ANSWER <-- ZERO
8	000400	FIC - FLOATING INTEGER CONVERSION INTERRUPT IF SET AND "STCFI" ERROR, ANSWER <-- ZERO, SET ERROR IF CLEAR AND "STCFI" ERROR, ANSWER <-- ZERO
7	000200	FD - FLOATING MODE 1=DOUBLE, 64 BIT OPERANDS (4W) 0=SINGLE, 32 BIT OPERANDS (2W)
6	000100	FL - INTEGER MODE 1=LONG, 32 BIT INTEGERS (2W) 0=SHORT, 16 BIT INTEGERS (1W)
5	000040	FT - ROUND/TRUNCATE MODE 1=TRUNCATE RESULTS 0=ROUND RESULTS
4	000020	FMM - PUT FP11-E ONLY IN MAINTENANCE MODE
3:0	000017	FN-FZ-FV-FC - FLOATING CONDITION CODES

FLOATING EXCEPTION CODES (FEC):

OCTAL	ENABLE	
00	(NONE)	(NOT USED)
02	(NONE)	FP OPCODE ERROR
04	(NONE)	FP DIVIDE-BY-ZERO ERROR
06	W/FIC	FP INTEGER CONVERSION ERROR
10	W/FIV	FP OVERFLOW ERROR
12	W/FIU	FP UNDERFLOW ERROR
14	W/FIUV	FP UNDEFINED-VARIABLE/(-0) ERROR
16	W/FMM	FP MAINTENANCE TRAP

NOTE - IN "FEC" CODE TYPEOUTS IN ERROR MESSAGES ONLY THE LOW ORDER BYTE IS USED - IGNORE THE PROGRAM FLAG BIT IN THE UPPER BYTE.

FLOATING POINT DATA:

IN FLOAT MODE (FD=0), IS 2-16. BIT WORDS, 32. BITS
IN DOUBLE MODE (FD=1), IS 4-16. BIT WORDS, 64. BITS

FIRST WORD: (BOTH F, D MODES)
B15=SIGN OF NUMBER (1/-, 0/+)
B14:07=EXPONENT, 8.BITS, FROM -128./+127.
B06:00=FRACTION, 7.BITS
SECOND WORD: (BOTH F, D MODES)
B15:00=FRACTION, 16.BITS
THIRD, FOURTH WORDS: (ONLY D MODE)
B15:00, B15:00=FRACTION, 32. BITS

IN F MODE, THE COMPOSITE 24. BIT FRACTION
IS FORMED BY:

.1#[WORD1-BIT<06:00>]#[WORD2-BIT<15:00>]

IN D MODE, THE COMPOSITE 56. BIT FRACTION
IS FORMED BY:

.1#[WORD1-BIT<06:00>]#[WORD2-BIT<15:00>]
#[WORD3-BIT<15:00>]#[WORD4-BIT<15:00>]

FOR A MORE DETAILED OPERATION/EXPLANATION OF FLOATING POINT DATA FORMATS AND OPERATIONS, SEE THE PDP-11/6X PROCESSOR HANDBOOK SECTION ON THE FLOATING POINT INSTRUCTION SET.

6.2 RECOVERY

RECOVERY FROM ERRORS HAS BEEN ATTEMPTED TO BE MADE AS AUTOMATIC AND EFFORTLESS AS POSSIBLE. HOWEVER, IN MANY CASES, DUE TO THE NATURE OF THE ERROR, THE PROGRAM MAY NOT EVEN BE ABLE TO BE RUN (EG, IF THE FLOATING POINT MODULE IS IN A HUNG STATE, AND CAN NEVER ENTER THE READY STATE TO ACCEPT A NEW FPP INSTRUCTION). AT THIS POINT, SOLVING THE PROBLEM IS A DIRECT FUNCTION OF THE OPERATORS INGENUITY. THIS TEST SERIES HAS BEEN DESIGNED TO TEST THE FLOATING POINT PROCESSOR SO THAT THESE TYPES OF FAILURES TO RUN WILL BE MINIMAL. THE TESTS HAVE BEEN PLACED IN A SPECIFICALLY STRUCTURED SEQUENCE IN THE PROGRAM TO IMPLEMENT THIS STRATEGY: TESTING THE MOST BASIC ELEMENTS FIRST, PROCEEDING UPWARD IN COMPLEXITY AFTER ESTABLISHING THEIR CORRECT OPERATION. THIS IS WHY IT IS EXTREMELY IMPORTANT THAT THE FLOATING POINT TEST PROGRAMS BE (1) RUN IN THE PRESCRIBED ORDER, AND (2) ONLY BE STARTED AT THEIR BEGINNING ADDRESS (USUALLY 200(8)). THE PROGRAM WILL DISPLAY, AT AN ERROR, THE MOST PERTINENT INFORMATION RELATING TO THE ERROR, AND A BRIEF EXPLANATION OF THE FAILING FUNCTION.

6.3 CAUSES

THESE TEST PROGRAMS ARE NOT HARDWARE ORIENTED, AND AS SUCH IT IS NOT POSSIBLE TO CALL OUT PARTICULAR HARDWARE AREAS AND MODULES RELATING TO A GIVEN FUNCTIONAL FAILURE. HARDWARE DIAGNOSIS FOR A PARTICULAR MACHINE MUST BE DONE USING THE APPROPRIATE ENGINEERING ROM FLOWS AND PRINTS, ALONG WITH THE KNOWN FUNCTIONAL ERRORS (AS DETECTED BY THE PROGRAMS). THIS IS THE INTENT UNDER WHICH THESE INSTRUCTION TESTS WERE DESIGNED AND CODED.

7. RESTRICTIONS

7.1 STARTING

THE PROGRAM MUST BE STARTED AT LOCATION 200(8) ALWAYS.

7.2 OPERATIONAL

THERE ARE NO OPERATIONAL RESTRICTIONS.

8. MISCELLANEOUS

8.1 EXECUTION TIME

MODEL	AVERAGE EXECUTION TIME PER PASS SHORTEST PASS	LONGEST PASS
PDP-11/6X MICROCODE	0:01	1:15
PDP-11/6X W/FP11-E	0:01	0:30

TIMES SPECIFIED AS (MINUTES):(SECONDS)

SHORTEST PASS ::= PASS=1, NO ITERATIONS, USING:
SWR=(004003) FOR PDP-11/6X MICROCODE
SWR=(004002) FOR PDP-11/6X W/FP11-E

LONGEST PASS ::= PASS>=2, 2000. ITERATIONS/TEST, USING:
SWR=(000003) FOR PDP-11/6X MICROCODE
SWR=(000002) FOR PDP-11/6X W/FP11-E

8.2 STACK POINTER

THE STACK POINTER IS SET TO 1100(8) AT THE START OF EACH PASS. IF ALL IS OPERATING CORRECTLY, IT SHOULD ALSO BE THIS VALUE AT

THE START OF EACH TEST, AND AT THE END OF A PASS.

8.3 POWER FAIL

THE TESTS MAY BE POWER FAILED AT ANY TIME. SPURIOUS ERROR MESSAGES MAY OCCUR IF THE FAILURE OCCURRED WHILE THE F.P.U. WAS EXECUTING A FUNCTION, AS NONE OF ITS REGISTERS (FPS, FEC, FEA, ACCUMULATORS) ARE SAVED IN THE EVENT OF A POWER FAILURE. HOWEVER, THESE MESSAGES SHOULD ONLY OCCUR ONCE (IF AT ALL) IMMEDIATELY AFTER POWER IS RESTORED. WHEN POWER IS RESTORED, "POWER" IS TYPED ON THE CONSOLE AND EXECUTION CONTINUES WHERE IT WAS INTERRUPTED.

NOTE THAT THE "VOLATILE" SWITCH REGISTER CONTENTS ARE SAVED AND RESTORED FROM THE STACK IN A POWER FAIL SEQUENCE; THEREFORE THE SWITCH REGISTER SETTINGS SHOULD NOT BE LOST OVER A POWER FAIL.

9. PROGRAM DESCRIPTION

9.1 ORGANIZATION

THESE PROGRAMS ARE ORGANIZED AS MUCH AS POSSIBLE IN A STRAIGHTFORWARD, LINEAR MANNER. THE MAIN BODY OF CODE IS STRUCTURED AS FOLLOWS:

(1) INITIALIZATION ROUTINE
    - SETS UP VECTORS, TYPES HEADER, ETC.
(2) MAIN BODY OF TESTS
    - INLINE TEST CODE, INLINE TEST CALLS
(3) END OF PASS ROUTINE
    - END OF PASS PROCESSING
(4) TEST SUBROUTINES
    - SUBROUTINES CONTAINING COMMON TEST CODE
(5) OVERHEAD ROUTINES
    - SERVICE SUBROUTINES (TYPEOUT,ETC.)

WHEREVER FEASIBLE, COMMON SECIONS OF CODE FOR WIDELY USED FUNCTIONS ARE CONDENSED INTO SUBROUTINES TO CONSERVE MEMORY. THIS INCLUDES NOT ONLY STANDARD SERVICE ROUTINES (SUCH AS SCOPE, ERROR, AND ASCII TYPEOUT), BUT ALSO TESTING ROUTINES WHICH PERFORM VERY SIMILAR FUNCTIONS. THUS IN MANY CASES (THE "ADDF" INSTRUCTION TESTING, FOR EXAMPLE) A SINGLE BODY OF CODE (A SUBROUTINE) IS USED TO PERFORM ALL THE FUNCTIONAL TESTS, WITH A VARIABLE PARAMETER LIST PASSED AT EACH CALL CONTAINING THE DATA OPERANDS AND EXPECTED RESULT FOR EACH INDIVIDUAL TEST. THIS CONSTRUCTION FACILITATES THE ADDITION/DELETION OF TESTS (SHOULD THAT EVER BE NECESSARY), AND ALSO GREATLY CONSERVES MEMORY SPACE REQUIREMENTS WHEN A LARGE NUMBER OF CALLS TO A GIVEN BODY OF CODE ARE REQUIRED.

THE INDIVIDUAL TESTS WITHIN EACH PROGRAM HAVE ALSO BEEN SEQUENCED IN A PARTICULAR ORDER TO FACILITATE THE DETECTION AND RESOLUTION OF ERRORS AS QUICKLY AS POSSIBLE. EACH OF THE TESTS BEGINS AS SIMPLY AS POSSIBLE, FIRST TESTING THE MOST BASIC ELEMENTS, MORE COMPLEX ELEMENTS ARE TESTED AFTERWARDS, EMPLOYING A PHILOSOPHY THAT THE SIMPLER THE TEST, THE BETTER THE RESOLUTION. ALL FUNCTIONS ARE EVENTUALLY TESTED, BUT HOPEFULLY MOST ERRORS WILL BE CAUGHT AND CORRECTED EARLY. A MUCH MORE DETAILED ANALYSIS OF THE SEQUENCE OF TESTS PERFORMED IS PRESENTED IN SECTION 9.2.

9.2 TEST DESCRIPTION

THIS DIAGNOSTIC CONSISTS OF A NUMBER OF TESTS TO EXERCISE COMBINATIONS OF FLOATING POINT INSTRUCTIONS, USING VARIOUS ADDRESS MODES, IN A MORE OR LESS APPLICATION-PROGRAM-TYPE ENVIRONMENT.

THE FIRST SECTION OF TESTS CONTAINS COMMONLY ENCOUNTERED SEQUENCES OF FLOATING POINT CODE, VARIOUS OPERANDS, ADDRESS MODES, AND FLOATING POINT ENVIRONMENTS (I.E., F/D MODES) ARE EMPLOYED. WHEN PRESENT, THE LINE AND/OR PROGRAMMABLE CLOCKS WILL BE EMPLOYED TO GENERATE I/O INTERRUPTS TO CHECK THE FP INSTRUCTION RESTART FEATURE.

THE NEXT SECTION OF CODE CHECKS FLOATING POINT EXCEPTION CONDITIONS. ALL EXCEPTION CONDITIONS ARE GENERATED, AND THE ENABLED/DISABLED AND TRAP/NO-TRAP MODES USED TO VERIFY CORRECT OPERATION.

THE LAST SECTION OF CODE CONSISTS OF A MORE COMPLEX EXERCISER SECTION THAN THAT IN THE FIRST SECTION, TO CHECK A MORE COMPLICATED SERIES OF FLOATING POINT OPERATIONS.

9.3 SUBROUTINE ABSTRACTS

9.3.1 TRAPCATCHER

THE TRAPCATCHER IS A SERIES OF INSTRUCTIONS OCCUPYING THE INTERRUPT VECTOR AREA OF MEMORY. IT CONSISTS OF THE SEQUENCE:

```
        .WORD   .+2     ;PC AFTER TRAP
        .WORD   0       ;PS AFTER TRAP
```

PLACED AT EACH VECTOR ADDRESS IN LOCATIONS 4-776(8) OF MEMORY. THE FIRST WORD OF EACH PAIR ("PC AFTER TRAP") POINTS TO THE SECOND WORD, WHICH SERVES A DUAL PURPOSE AS
(1) THE NEW LOADED PS (ALL ZEROS), AND (2) THE NEXT INSTRUCTION TO EXECUTE (0=HALT).

WHEN THE PROGRAM IS EXECUTING, ANY REQUIRED VECTORS ARE SET UP

IN THE VECTOR AREA WITH APPROPRIATE VALUES; THE OTHERS BEING LEFT IN THE "TRAPCATCHER" STATE. THUS, IF AN UNEXPECTED TRAP EVER OCCURS IN THE MACHINE, IT WILL BE CAUGHT, AND THE MACHINE SUBSEQUENTLY HALTED, DISPLAYING THE VECTOR ADDRESS + PLUS FOUR + IN THE ADDRESS LIGHTS.

9.3.2 SCOPE ROUTINE - $SCOPE

THE SCOPE ROUTINE IS ENTERED FROM THE FIRST INSTRUCTION OF EACH TEST IN THE PROGRAM. (NOTE THAT BY DEFINITION, A "TEST" WILL BE DESIGNATED AS THE SECTION OF CODE BETWEEN TWO "SCOPE" STATEMENTS.) THIS ROUTINE PROVIDES THE OVERHEAD CODE NECESSARY TO IMPLEMENT SEVERAL OF THE SWITCH REGISTER CONTROL OPTIONS. UPON ENTRANCE TO A TEST, THE SCOPE STATEMENT AT THE BEGINNING SETS UP CERTAIN LOCATIONS (SEE BELOW) TO SPECIFY THE CURRENT TEST NUMBER AND LOOPING ADDRESS (FOR ITERATIONS). CONTROL IS THEN PASSED TO THE ACTUAL TEST CODE, PERFORMING THE DESIRED TEST. UPON EXIT, THE SCOPE STATEMENT OF THE NEXT TEST IS ENTERED, WHICH DETERMINES WHETHER TO (1) LOOP BACK TO THE PREVIOUS TEST (EG, FOR ITERATIONS) OR (2) INITIALIZE FOR THE NEXT TEST (AS DESCRIBED EARLIER, ABOVE).

ENTRANCE TO THE SCOPE ROUTINE IS VIA AN "IOT" TRAP CALL THROUGH LOCATION 20(8). (FROM THE SCOPE=IOT EQUATE). DEPENDING UPON THE SWITCH SETTINGS (SEE 5.2), CODE IS PRESENT TO: LOAD THE FP11 MICRO BREAK REGISTER, LOOP ON THE CURRENTLY EXECUTING TEST, LOOP ON A SPECIFIC TEST, PERFORM ITERATIONS OF EACH TEST, AND SET UP ADDRESSES FOR POSSIBLE LOOPING ON ERRORS. IMPORTANT VALUES USED IN THIS ROUTINE ARE:

- $MXCNT - MAXIMUM NUMBER OF ITERATIONS PER TEST (GENERALLY WILL BE 2000(10))
- $TSTNM - A COUNTER INDICATING THE NUMBER (1-377(8)) OF THE TEST CURRENTLY BEING EXECUTED
- $LPADR - CONTAINS THE ADDRESS TO WHICH THE SCOPE ROUTINE 10370 WILL LOOP, IF THE CURRENT TEST IS BEING LOOPED UPON
- $LPERR - CONTAINS THE ADDRESS TO WHICH THE ERROR ROUTINE (SEE 9.3.3) WILL LOOP, IF AN ERROR OCCURS AND THE LOOPING ON AN ERROR OPTION IS SPECIFIED IN THE SWITCHES. SET UP BY SCOPE, GENERALLY WILL BE THE SAME AS $LPADR, ABOVE.

9.3.3 ERROR ROUTINE - $ERROR

THE ERROR ROUTINE IS ENTERED WHEN THE TEST CODE HAS DETERMINED THAT AN ERROR HAS OCCURRED AS PART OF A TEST. THROUGH USE OF THIS ROUTINE, THE TEST HAS A MEANS OF SIGNALING AN ERROR TO THE 10550 OPERATOR/MONITOR; AND IMPLEMENTING THE CONTROL FUNCTIONS FOR HALTING ON ERROR, BELL ON ERROR, AND LOOPING ON ERROR. IN ADDITION, THE ERROR ROUTINE HAS THE PROVISION TO TYPE OUT ON THE OPERATOR'S CONSOLE A MESSAGE BRIEFLY EXPLAINING THE ERROR, AND SOME OF THE MOST PERTINENT

DATA VALUES TO HELP DIAGNOSE THE CAUSE (SEE SECTION 6.2).

THE CALLING MECHANISM IS SIMILAR TO THAT EMPLOYED FOR THE SCOPE ROUTINE (VIA A TRAP), EXCEPT IN THIS INSTANCE, THE "EMT" INSTRUCTION IS USED, TRAPPING THROUGH LOCATION 30(8). (NOTE THE EQUATE ERROR N=EMT N). THE LOWER BYTE OF THE EMT INSTRUCTION IS CAPABLE OF TRANSMITTING A NUMBER FROM 0-377(8), WHICH WILL BE TERMED THE "ERROR ITEM NUMBER." THIS NUMBER DETERMINES WHICH ERROR MESSAGE, AND ASSOCIATED DATA VALUES WILL BE TYPED OUT WHEN A PARTICULAR ERROR IS SIGNALED. IF THIS NUMBER IS ZERO, JUST THE PC OF THE CALLING "ERROR" INSTRUCTION WILL BE TYPED, OTHERWISE, THE NUMBER IS USED AS AN INDEX THROUGH THE ERROR TABLE ($ERRTB) TO FIND THE APPROPRIATE VALUES TO TYPE (SEE PROGRAM LISTING FOR FURTHER DETAILS).

IMPORTANT VALUES USED IN THIS ROUTINE ARE:

- $EREG0 THRU EREG7 - CONTENTS OF GENERAL REGISTERS R0 THRU R7 JUST BEFORE ERROR CALL
- $ERTTL - COMULATIVE NUMBER OF ERRORS ENCOUNTERED TO DATE
- $ERRPC - CONTAINS THE PC OF THE "ERROR" INSTRUCTION JUST EXECUTED
- $LPERR - CONTAINS THE ADDRESS WHICH WILL BE LOOPED UPON FOR THE ERROR LOOPING FACILITY

9.3.4 ERROR MESSAGE TYPEOUT ROUTINE - $TYPERR

THIS ROUTINE ($TYPERR ENTRY POINT) IS CALLED BY THE ERROR PROCESSING ROUTINE DESCRIBED IN 9.3.3 ABOVE: ITS PURPOSE IS TO IMPLEMENT THE ERROR MESSAGE/DATA VALUE ERROR TYPEOUT FACILITY. THE SUBROUTINE WILL, GIVEN THE INDEXING BYTE FROM THE ERROR CALL INSTRUCTION, PICK UP THE CORRECT ERROR MESSAGE VECTOR FROM $ERRTB (ERROR TABLE), AND TYPE OUT THE ERROR MESSAGE, DATA HEADER, AND DATA VALUES ON THE CONSOLE.

9.3.5 TYPE ROUTINE - $TYPE

THIS ROUTINE IS THE STANDARD SYSTEM TYPEOUT ROUTINE FOR ASCII SINGLE-CHARACTER-PER-BYTE STRINGS. IT IS CALLED THROUGH A TRAP INSTRUCTION WITH THE NEXT WORD CONTAINING THE ADDRESS OF THE FIRST CHARACTER IN THE STRING. TYPING TERMINATES WHEN AN ALL-ZERO BYTE IS FOUND. HORIZONTAL TAB STOPS ARE ALSO AUTOMATICALLY PLACED.

9.3.6 OCTAL NUMBER TYPE ROUTINE - $TYPOC

THIS ROUTINE CONVERTS THE TOP NUMBER ON THE STACK TO A 6-DIGIT OCTAL REPRESENTATION, AND TYPES IT ON THE CONSOLE USING THE TYPE ROUTINE $TYPE. SEE LISTING FOR OPTIONS AND FURTHER DETAILS.

9.3.7 POWER UP AND DOWN ROUTINES - $PWRUP AND $PWRDN

THESE TWO ROUTINES ARE ENTERED FOR THE POWER UP AND DOWN CONDITIONS, RESPECTIVELY. THE POWER DOWN ROUTINE (SPWRDN) SAVES THE GENERAL REGISTERS AND STACK POINTER. THE POWER UP ROUTINE (SPWRUP) CORRESPONDINGLY RESTORES THE REGISTERS, STACK POINTER, AND TYPES THE MESSAGE "POWER" WHEN POWER IS RESTORED. THE VOLATILE INTERNAL SWITCH REGISTER IS ALSO SAVED/RESTORED BY THIS ROUTINE.

9.3.8 END OF PASS ROUTINE - SECP

THE END OF PASS ROUTINE COUNTS THE NUMBER OF PASSES PERFORMED, DINGS THE BELL/TYPES A MESSAGE (IF ENABLED), SETS/CLEARS THE T-BIT (IF ENABLED), AND ALSO INTERFACES TO THE MONITOR, IF PRESENT. IT ALSO OPTIONALLY LOOPS FOR A NUMBER OF SUBPASSES BEFORE SIGNALLING AN END OF PASS CONDITION.

10. ACT/APT/XXDP

10.1 ACT COMPATIBILITY

THIS PROGRAM WILL RUN UNDER THE ACT SYSTEM.

10.2 APT COMPATIBILITY

THIS PROGRAM WILL RUN UNDER THE APT SYSTEM MONITOR. ALL NECESSARY SOFTWARE COMMUNICATION HOOKS ARE PRESENT.

10.3 XXDP COMPATIBILITY

FOR XXDP MEDIA COMPATIBILITY, THE TOP 2K WORDS OF THE 16K WORD MINIMUM MEMORY AREA ARE NOT DISTURBED DURING EXECUTION.

```
                        .TITLE  FPU INSTR EXERCISER
                        ;*COPYRIGHT (C) 1976
                        ;*DIGITAL EQUIPMENT CORP.
                        ;*MAYNARD, MASS. 01754
                        ;*
                        ;*PROGRAM BY DONALD NORTH
                        ;*
                        ;*THIS PROGRAM WAS ASSEMBLED USING THE PDP-11 MAINDEC SYSMAC
                        ;*PACKAGE (MAINDEC-11-DZQAC-C3), JAN 19, 1977.
                        ;*

                        .SBTTL  OPERATIONAL SWITCH SETTINGS
                        ;*
                        ;*      SWITCH  OCTAL           USE
                        ;*      ------  ------          ------------------
                        ;*        15    100000          HALT ON ERROR
                        ;*        14    040000          LOOP ON CURRENTLY EXECUTING TEST
                        ;*        13    020000          INHIBIT ERROR TYPEOUTS
                        ;*        12    010000          INHIBIT STATUS TYPEOUTS
                        ;*        11    004000          INHIBIT ITERATIONS
                        ;*        10    000000          0=BELL ON PASS END
                        ;*              002000          1=BELL ON ERROR
                        ;*         9    001000          LOOP ON ERROR
                        ;*         8    000400          LOOP ON TEST NUMBER IN "SLPTST"
                        ;*         1    000000          0=TEST HFP/WFP ALTERNATELY EACH PASS
                        ;*              000002          1=TEST ONLY UNIT SPECIFIED IN SW<00>
                        ;*         0    000002          0=SELECT HFP, IF SW<01>=1
                        ;*              000003          1=SELECT WFP, IF SW<01>=1

                        .SBTTL  BASIC DEFINITIONS

                        ;*INITIAL ADDRESS OF THE STACK POINTER *** 1100 ***
001100                  STACK=  1100
                        .EQUIV  EMT,ERROR       ;;BASIC DEFINITION OF ERROR CALL
                        .EQUIV  IOT,SCOPE       ;;BASIC DEFINITION OF SCOPE CALL

                        ;*MISCELLANEOUS DEFINITIONS
000011                  HT=     11              ;;CODE FOR HORIZONTAL TAB
000012                  LF=     12              ;;CODE FOR LINE FEED
000015                  CR=     15              ;;CODE FOR CARRIAGE RETURN
000200                  CRLF=   200             ;;CODE FOR CARRIAGE RETURN-LINE FEED
177776                  PS=     177776          ;;PROCESSOR STATUS WORD
                        .EQUIV  PS,PSW
177774                  STKLMT= 177774          ;;STACK LIMIT REGISTER
177772                  PIRQ=   177772          ;;PROGRAM INTERRUPT REQUEST REGISTER
177570                  DSWR=   177570          ;;HARDWARE SWITCH REGISTER
177570                  DDISP=  177570          ;;HARDWARE DISPLAY REGISTER

                        ;*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
```

```
                        ;
                        ;GENERAL REGISTER DEFINITIONS
000006                  R6=     %6              ;;GENERAL REGISTER
000007                  R7=     %7              ;;GENERAL REGISTER
000006                  SP=     %6              ;;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
040000                  SW14=   40000
020000                  SW13=   20000
010000                  SW12=   10000
004000                  SW11=   4000
002000                  SW10=   2000
001000                  SW09=   1000
000400                  SW08=   400
000200                  SW07=   200
000100                  SW06=   100
000040                  SW05=   40
000020                  SW04=   20
000010                  SW03=   10
000004                  SW02=   4
000002                  SW01=   2
000001                  SW00=   1
                        .EQUIV  SW09,SW9
                        .EQUIV  SW08,SW8
                        .EQUIV  SW07,SW7
                        .EQUIV  SW06,SW6
                        .EQUIV  SW05,SW5
                        .EQUIV  SW04,SW4
                        .EQUIV  SW03,SW3
                        .EQUIV  SW02,SW2
                        .EQUIV  SW01,SW1
                        .EQUIV  SW00,SW0
                        ;
                        ;*DATA BIT DEFINITIONS (BIT00 TO BIT15)
100000                  BIT15=  100000
040000                  BIT14=  40000
020000                  BIT13=  20000
010000                  BIT12=  10000
004000                  BIT11=  4000
002000                  BIT10=  2000
001000                  BIT09=  1000
000400                  BIT08=  400
000200                  BIT07=  200
000100                  BIT06=  100
000040                  BIT05=  40
000020                  BIT04=  20
```

```
                        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
000010                  RESVEC= 10              ;;RESERVED AND ILLEGAL INSTRUCTIONS
000014                  TBITVEC=14              ;;"T" BIT
000014                  TRTVEC= 14              ;;TRACE TRAP
000014                  BPTVEC= 14              ;;BREAKPOINT TRAP (BPT)
000020                  IOTVEC= 20              ;;INPUT/OUTPUT TRAP (IOT) **SCOPE**
000024                  PWRVEC= 24              ;;POWER FAIL
000030                  EMTVEC= 30              ;;EMULATOR TRAP (EMT) **ERROR**
000034                  TRAPVEC=34              ;;"TRAP" TRAP
000060                  TKVEC=  60              ;;TTY KEYBOARD VECTOR
000064                  TPVEC=  64              ;;TTY PRINTER VECTOR
000240                  PIRQVEC=240             ;;PROGRAM INTERRUPT REQUEST VECTOR

                        ;*MED INSTR EQUATES
076600                  MED=    076600          ;OPCODE

000022                  RWHAMI= 022             ;READ WHAMI

000144                  RFLAG=  144             ;READ FLAGS
000344                  WFLAG=  344             ;WRITE FLAGS

                        ;*LINE CLOCK VECTORS, ADDRESSES
177546                  LKS=    177546          ;STATUS                 KW11-L LINE CLOCK
000100                  LKV=    100             ;INTERRUPT VECTOR

172540                  PLKS=   172540          ;STATUS                 KW11-P PROG LINE CLOCK
172542                  PLKR=   172542          ;COUNT SET REGISTER
172544                  PLKD=   172544          ;READ COUNT REGISTER
000104                  PLKV=   104             ;INTERRUPT VECTOR

                        ;*FLOATING POINT INTERRUPT VECTOR
000244                  FPPVEC= 244

                        ;*FLOATING POINT REGISTER DEFINITIONS
000000                  AC0=    %0
000001                  AC1=    %1
000002                  AC2=    %2
000003                  AC3=    %3
000004                  AC4=    %4
000005                  AC5=    %5
```

```
                                        .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
        000174                          .=174
000174  000000                 DISPREG: .WORD  0                ;;SOFTWARE DISPLAY REGISTER
000176  000000                 SWREG:   .WORD  0                ;;SOFTWARE SWITCH REGISTER
                                        .SBTTL  STARTING ADDRESS(ES)
000200  000137  002400                  JMP     @#START ;;JUMP TO STARTING ADDRESS OF PROGRAM

                                        .SBTTL  ACT11 HOOKS

                                        ;;**************************************************************
                                        ;HOOKS REQUIRED BY ACT11
        000204                          $SVPC=.                  ;SAVE PC
        000046                          .=46
000046  014264                          $ENDAD                   ;;1)SET LOC.46 TO ADDRESS OF $ENDAD IN .$EOP
        000052                          .=52
000052  000000                          .WORD   0                ;;2)SET LOC.52 TO ZERO
        000204                          .=$SVPC                  ;; RESTORE PC
        001000                          .=1000
                                        .SBTTL  APT PARAMETER BLOCK

                                        ;;**************************************************************
                                        ;SET LOCATIONS 24 AND 44 AS REQUIRED FOR APT
                                        ;;**************************************************************
        001000                                  .$X=.   ;;SAVE CURRENT LOCATION
        000024                          .=24    ;;SET POWER FAIL TO POINT TO START OF PROGRAM
000024  000200                          200     ;;FOR APT START UP
        000044                          .=44    ;;POINT TO APT INDIRECT ADDRESS PNTR.
000044  001000                          $APTHDR ;;POINT TO APT HEADER BLOCK
        001000                          .=.$X   ;;RESET LOCATION COUNTER
                                        ;;**************************************************************
                                        ;SETUP APT PARAMETER BLOCK AS DEFINED IN THE APT-PDP11 DIAGNOSTIC
                                        ;INTERFACE SPEC.

001000                          $APTHD:
001000  000000                 $HIBTS: .WORD   0        ;;TWO HIGH BITS OF 18 BIT MAILBOX ADDR.
001002  001244                 $MBADR: .WORD   $MAIL    ;;ADDRESS OF APT MAILBOX (BITS 0-15)
001004  000012                 $TSTM:  .WORD   10.      ;;RUN TIM OF LONGEST TEST
001006  000002                 $PASTM: .WORD   2.       ;;RUN TIME IN SECS. OF 1ST PASS ON 1 UNIT (QUICK VERIFY)
001010  000000                 $UNITM: .WORD   0        ;;ADDITIONAL RUN TIME (SECS) OF A PASS FOR EACH ADDITIONAL UNIT
001012  000014                         .WORD   $ETEND-$MAIL/2 ;;LENGTH MAILBOX-ETABLE(WORDS)
```

```
                                .SBTTL  COMMON TAGS

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

                        001100          .=1100
001100                  SCMTAG:                         ;;START OF COMMON TAGS
                                ;;-----START OF CLEAR COMMON TAGS-----
001100  000000                  .WORD   0
001102  000000          STSTNM: .WORD   0               ;;CONTAINS THE TEST NUMBER
001104  000000          SERFLG: .WORD   0               ;;CONTAINS ERROR FLAG
001106  000000          SICNT:  .WORD   0               ;;CONTAINS SUBTEST ITERATION COUNT
001110  000000          SLPADR: .WORD   0               ;;CONTAINS SCOPE LOOP ADDRESS
001112  000000          SLPERR: .WORD   0               ;;CONTAINS SCOPE RETURN FOR ERRORS
001114  000000          SERTTL: .WORD   0               ;;CONTAINS TOTAL ERRORS DETECTED
001116  000000          SITEMB: .WORD   0               ;;CONTAINS ITEM CONTROL BYTE
001120  000001          SERMAX: .WORD   1               ;;CONTAINS MAX. ERRORS PER TEST
001122  000000          SERRPC: .WORD   0               ;;CONTAINS PC OF LAST ERROR INSTRUCTION
001124  000000          SGDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'GOOD' DATA
001126  000000          SBDADR: .WORD   0               ;;CONTAINS ADDRESS OF 'BAD' DATA
001130  000000          SGDDAT: .WORD   0               ;;CONTAINS 'GOOD' DATA
001132  000000          SBDDAT: .WORD   0               ;;CONTAINS 'BAD' DATA
001134  000000                  .WORD   0               ;;RESERVED--NOT TO BE USED
001136  000000                  .WORD   0
001140  000             SAUTOB: .BYTE   0               ;;AUTOMATIC MODE INDICATOR
001141  000             SINTAG: .BYTE   0               ;;INTERRUPT MODE INDICATOR
001142  000000                  .WORD   0
                                ;;-----END OF CLEAR COMMON TAGS-----
001144  177570          SWR:    .WORD   DSWR            ;;ADDRESS OF SWITCH REGISTER
001146  177570          DISPLA: .WORD   DDISP           ;;ADDRESS OF DISPLAY REGISTER
001150  000000          SLPTST: .WORD   0               ;;CONTAINS TEST NUMBER TO LOOP UPON
001152  177560          STKS:   177560                  ;;TTY KBD STATUS
001154  177562          STKB:   177562                  ;;TTY KBD BUFFER
001156  177564          STPS:   177564                  ;;TTY PRINTER STATUS REG. ADDRESS
001160  177566          STPB:   177566                  ;;TTY PRINTER BUFFER REG. ADDRESS
001162  000             SNULL:  .BYTE   0               ;;CONTAINS NULL CHARACTER FOR FILLS
001163  002             SFILLS: .BYTE   2               ;;CONTAINS # OF FILLER CHARACTERS REQUIRED
001164  012             SFILLC: .BYTE   12              ;;[illegible] FILL CHARS. AFTER A "LINE FEED"
001165  000             STPFLG: .BYTE   0               ;;"[illegible]" FLAG (BIT<07>=0=YES)
001166  000000          SREGAD: .WORD   0               ;;CONTAINS THE ADDRESS FROM
                                                        ;;WHICH ($REGO) WAS OBTAINED
001170  000000          SREG0:  .WORD   0               ;;CONTAINS (($REGAD)+0)
001172  000000          SREG1:  .WORD   0               ;;CONTAINS (($REGAD)+2)
001174  000000          SREG2:  .WORD   0               ;;CONTAINS (($REGAD)+4)
001176  000000          SREG3:  .WORD   0               ;;CONTAINS (($REGAD)+6)
001200  000000          SREG4:  .WORD   0               ;;CONTAINS (($REGAD)+10)
001202  000000          SREG5:  .WORD   0               ;;CONTAINS (($REGAD)+12)
001204  000000          SREG6:  .WORD   0               ;;CONTAINS (($REGAD)+14)
001206  000000          SREG7:  .WORD   0               ;;CONTAINS (($REGAD)+16)
001210  000000          STMP0:  .WORD   0               ;;USER DEFINED
001212  000000          STMP1:  .WORD   0               ;;USER DEFINED
001214  000000          STMP2:  .WORD   0               ;;USER DEFINED
001216  000000          STMP3:  .WORD   0               ;;USER DEFINED
001220  000000          STMP4:  .WORD   0               ;;USER DEFINED
001222  000000          STMP5:  .WORD   0               ;;USER DEFINED
```

```
                                ;TIMES: .WORD   0               ;;USER DEFINED
                                ;TMP7:  .WORD   0               ;;USER DEFINED
                                $TIMES: 0                       ;;MAX. NUMBER OF ITERATIONS
                                $ESCAPE:0                       ;;ESCAPE ON ERROR ADDRESS
                                $BELL:  .ASCIZ  <207><377><377> ;;CODE FOR BELL
                                $QUES:  .ASCII  /?/             ;;QUESTION MARK
                                $CRLF:  .ASCII  <15>            ;;CARRIAGE RETURN
                                $LF:    .ASCIZ  <12>            ;;LINE FEED
                                ;;*****************************************************************
                                .SBTTL  APT MAILBOX-ETABLE

                                ;;*****************************************************************
                                .EVEN
                                $MAIL:                          ;;APT MAILBOX
                                $MSGTY: .WORD   AMSGTY          ;;MESSAGE TYPE CODE
                                $FATAL: .WORD   AFATAL          ;;FATAL ERROR NUMBER
                                $TESTN: .WORD   ATESTN          ;;TEST NUMBER
                                $PASS:  .WORD   APASS           ;;PASS COUNT
                                $DEVCT: .WORD   ADEVCT          ;;DEVICE COUNT
                                $UNIT:  .WORD   AUNIT           ;;I/O UNIT NUMBER
                                $MSGAD: .WORD   AMSGAD          ;;MESSAGE ADDRESS
                                $MSGLG: .WORD   AMSGLG          ;;MESSAGE LENGTH
                                $ETABLE:                        ;;APT ENVIRONMENT TABLE
                                $ENV:   .BYTE   AENV            ;;ENVIRONMENT BYTE
                                $ENVM:  .BYTE   AENVM           ;;ENVIRONMENT MODE BITS
                                $SWREG: .WORD   ASWREG          ;;APT SWITCH REGISTER
                                $USWR:  .WORD   AUSWR           ;;USER SWITCHES
                                $CPUOP: .WORD   ACPUOP          ;;CPU TYPE,OPTIONS
                                ;*                              BITS 15-11=CPU TYPE
                                ;*                                      11/04=01,11/05=02,11/20=03,11/40=04,11/45=05
                                ;*                                      11/70=06,PDQ=07,Q=10
                                ;*                              BIT 10=REAL TIME CLOCK
                                ;*                              BIT  9=FLOATING POINT PROCESSOR
                                ;*                              BIT  8=MEMORY MANAGEMENT
                                $ETEND:
                                .MEXIT
```

```
                                .SBTTL  ERROR POINTER TABLE

001274                          SERRTB:

                                ;*THIS TABLE CONTAINS THE INFORMATION FOR EACH ERROR THAT CAN OCCUR.
                                ;*THE INFORMATION IS OBTAINED BY USING THE INDEX NUMBER FOUND IN
                                ;*LOCATION SITEMB. THIS NUMBER INDICATES WHICH ITEM IN THE TABLE IS PERTINENT.
                                ;*NOTE1:        IF SITEMB IS 0 THE ONLY PERTINENT DATA IS (SERRPC).
                                ;*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
                                                ;*NOTE:  ERROR VECTOR TABLE (SERRTB) HAS BEEN MODIFIED,
                                                ;*       ELIMINATING UNUSED VALUE FOR DATA FORMAT POINTER.
                                                ;*       ERROR TYPING ROUTINE HAS ALSO BEEN MODIFIED
                                                ;*       ACCORDINGLY.
001274  017431  020374  020726  EMV001: .WORD   EMA,DHA,DTA      ;FPS BAD
001302  017455  020401  020660  EMV002: .WORD   EMB,DHB,DTB      ;FEC BAD
001310  017501  020406  020740  EMV003: .WORD   EMC,DHC,DTC      ;FEA BAD
001316  017525  020413  020664  EMV004: .WORD   EMD,DHD,DTD      ;PS CC-S BAD
001324  017557  020425  020666  EMV005: .WORD   EME,DHE,DTE      ;BAD ADDR IN R0
001332  017601  020432  020672  EMV006: .WORD   EMF,DHF,DTF      ;BAD SP
001340  020001  000000  000000  EMV007: .WORD   EMP,0,0          ;DIDNT TRAP, SHOULD HAVE
001346  017623  020466  020712  EMV010: .WORD   EMH,DHI,DTI      ;RESULT BAD - D
001354  017623  020452  020704  EMV011: .WORD   EMH,DHH,DTH      ;RESULT BAD - F, L
001362  017623  020444  020700  EMV012: .WORD   EMH,DHG,DTG      ;RESULT BAD - I
001370  017623  020425  020666  EMV013: .WORD   EMH,DHE,DTE      ;RESULT BAD - R0
                                ;*****  VECTORS FOR TRAP/FEC TESTER ROUTINE  *****
001376  020001  000000  000000  EMV014: .WORD   EMP,0,0          ;NO TRAP, SHOULD HAVE    TRAP/FEC TSTR
001404  020067  000000  000000  EMV015: .WORD   EMQ,0,0          ;TRAPPED, SHOULDN'T HAVE
001412  020154  020576  020762  EMV016: .WORD   EMR,DHL,DTAG     ;SP OK
001420  020223  020576  020770  EMV017: .WORD   EMS,DHL,DTAH     ;STORED PC
001426  020266  020576  020776  EMV020: .WORD   EMT,DHL,DTAI     ;STORED PS
001434  020331  020576  021004  EMV021: .WORD   EMU,DHL,DTAJ     ;LOADED PS
001442  017675  020536  020732  EMV022: .WORD   EMI,DHK,DTK      ;FEC/FEA BAD
001450  017431  020516  020724  EMV023: .WORD   EMA,DHJ,DTJ      ;FPS BAD
                                ;*****  VECTOR FOR UNEXPECTED FPP TRAP  *****
001456  017725  020612  020744  EMV024: .WORD   EMJ,DHM,DTL      ;UNEXPECTED TRAP
```

```
                                        .SBTTL  PROGRAM DEFINED COMMON TAGS
                                        ;*VARIABLES
001464  000000                  FPS:      .WORD   0               ;FPS STORED HERE AFTER STFPS
001466  000000                  FEC:      .WORD   0               ;FEC STORED HERE AFTER STST
001470  000000                  FEA:      .WORD   0               ;FEA STORED HERE AFTER STST
001472  000000                  FPPOPC:   .WORD   0               ;OLD PC SAVED HERE AFTER TRAP
001474  000000                  FPPOPS:   .WORD   0               ;OLD PS SAVED HERE AFTER TRAP
001476  000000                  FPPOSP:   .WORD   0               ;SP AFTER TRAP
001500  000000                  EXPFEA:   .WORD   0               ;EXPECTED FEA
001502  000000                  EXPRTS:   .WORD   0               ;EXPECTED RETURN ADDR
001504  000000                  EXPSP:    .WORD   0               ;EXPECTED SP VALUE AFTER TRAP
001506  000000                  OPCRCV:   .WORD   0               ;PC PUSHED ON STACK AFTER TRAP
001510  000000                  OPSRCV:   .WORD   0               ;PS PUSHED ON STACK AFTER TRAP
001512  000000                  NPSLOD:   .WORD   0               ;PS THAT WAS LOADED AFTER TRAP
001514  000000                  OSPRCV:   .WORD   0               ;SP AFTER FPP TRAP

001516  000000                  CLKPRS:   .WORD   000000          ;BIT07=1 -> KW11-L PRESENT
                                                                  ;BIT15=1 -> KW11-P PRESENT
001520  000000                  KW11LC:   .WORD   0               ;KW11-L COUNT OF # OF INTERRUPTS
001522  000000                  KW11PC:   .WORD   0               ;KW11-P COUNT OF # OF INTERRUPTS
001524  000000                  KW11PR:   .WORD   0               ;KW11-P COUNT SET

                                        ;*REGISTER CONTENTS, AT ERROR, STORED HERE
001526  000000                  EREG0:    .WORD   0
001530  000000                  EREG1:    .WORD   0
001532  000000                  EREG2:    .WORD   0
001534  000000                  EREG3:    .WORD   0
001536  000000                  EREG4:    .WORD   0
001540  000000                  EREG5:    .WORD   0
001542  000000                  EREG6:    .WORD   0
001544  000000                  EREG7:    .WORD   0

                                        ;*CONSTANTS
001546  040200  000000  000000  FLTONE:  .WORD   040200,0,0,0
001554  000000
001556  177777                  D1010:    .WORD   -1
001560  000000  177777  000000  D0101:    .WORD   0,-1,0
001566  177777  000000  000000  D1001:    .WORD   -1,0,0
001574                          DSMALL:
001574  177777  000001  000000  WEIRD:    .WORD   -1,1,0,-1
001602  177777
001604  001560                  AD0101:   .WORD   D0101
001606  001566                  AD1001:   .WORD   D1001
001610  001726                  AD1000:   .WORD   D1000
001612  077777  000000  177777  DBIG:     .WORD   77777,0,-1,0
001620  000000
001622  100000  000000  000000  DMZERO:   .WORD   100000,0,0,0
001630  000000
001632  001170                  ASREG0:   .WORD   $REG0
001634  040252  125252  125252  DALTA:    .WORD   40252,125252,125252,125252
001642  125252
001644  040325  052525  052525  DALTB:    .WORD   40325,52525,52525,52525
001652  052525
```

```
001654 040325 052525 052525 052525 DALTC:  .WORD   40325,52525,52525,52526
001662 052526
001664 040000 000000 000000 000000 D40:    .WORD   40000,0,0,0
001672 000000
001674 037400 000000 000000 000000 D37:    .WORD   37400,0,0,0
001702 000000
001704 040600 000000 000000 000000 D46:    .WORD   40600,0,0,0
001712 000000
001714 020000 000000 000000 000000 D20:    .WORD   20000,0,0,0
001722 000000
001724 000000                      D0100:  .WORD   0
001726 177777 000000 000000 000000 D1000:  .WORD   -1,0,0,0
001734 000000
001736 000100 000000 000000 000000 D0100X: .WORD   100,0,0
001744 000000 177777 177777 177777 D0111:  .WORD   0,-1,-1,-1
001752 177777
001754 000000 054321               D5T01:  .WORD   0,54321
001760 001754                      AD5T01: .WORD   D5T01
001762 043661 121000 000000 000000 F5T01:  .WORD   43661,121000,0,0
001770 000000


                                   ;*MESSAGES FOR BEGIN PROGRAM/START OF PASS
001772 005015 005012 042115 042120 BGNMES: .ASCII  <CR><LF><LF><LF>"MD-11-DQFPC-"
002000 030455 026461 050504
002006 050106 026503
002012 027102                      .ASCII  "B."
002014 027056                      .ASCII  "..."
002016 042120 026520 030464        .ASCIZ  "PDP-11/6X F.P.U. INSTRUCTION EXERCISER"<CR><LF>
002024 033057 020130 027106
002032 027120 027125 044440
002040 051516 051124 041525
002046 044524 047117 042440
002054 042530 041522 051511
002062 051105 005015 000000
002067 015 050012 051501 NWPAS1: .ASCIZ  <CR><LF>"PASS #"
002074 020123 000043
```

```
                                .SBTTL  START OF PASS ROUTINE


                                ;;**************************************************************
                                        .ENABL  AMA             ;ASSEMBLE ALL RELATIVE REFERENCES AS ABSOLUTE
                                ;;**************************************************************

002400                          START:
                                .SBTTL  INITIALIZE THE COMMON TAGS
                                ;;CLEAR THE COMMON TAGS ($CMTAG) AREA
002400  012706  001100                  MOV     #$CMTAG,R6      ;;FIRST LOCATION TO BE CLEARED
002404  005026                  CLR     (R6)+           ;;CLEAR MEMORY LOCATION
002406  022706  001144                  CMP     #SWR,R6 ;;DONE?
002412  001374                  BNE     .-6             ;;LOOP BACK IF NO
002414  012706  001100                  MOV     #STACK,SP       ;;SETUP THE STACK POINTER
                                ;;INITIALIZE A FEW VECTORS
002420  012737  015244  000020          MOV     #$SCOPE,@#IOTVEC ;;IOT VECTOR FOR SCOPE ROUTINE
002426  012737  000340  000022          MOV     #340,@#IOTVEC+2 ;;LEVEL 7
002434  012737  015522  000030          MOV     #ERROR,@#EMTVEC ;;EMT VECTOR FOR ERROR ROUTINE
002442  012737  000340  000032          MOV     #340,@#EMTVEC+2 ;;LEVEL 7
002450  012737  017156  000034          MOV     #TRAP,@#TRAPVEC ;;TRAP VECTOR FOR TRAP CALLS
002456  012737  000340  000036          MOV     #340,@#TRAPVEC+2;LEVEL 7
002464  012737  017224  000024          MOV     #PWRDN,@#PWRVEC ;;POWER FAILURE VECTOR
002472  012737  000340  000026          MOV     #340,@#PWRVEC+2 ;;LEVEL 7
002500  013737  014234  014226          MOV     $ENDCT,$EOPCT   ;;SETUP END-OF-PROGRAM COUNTER
002506  005037  001230                  CLR     $ITIMES         ;;INITIALIZE NUMBER OF ITERATIONS
002512  005037  001232                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE ON ERROR ADDRESS
002516  012737  000001  001120          MOV     #1,$ERMAX       ;;ALLOW ONE ERROR PER TEST
002524  012737  002524  001110          MOV     #.,$LPADR       ;;INITIALIZE THE LOOP ADDRESS FOR SCOPE
002532  012737  002532  001112          MOV     #.,$LPERR       ;;SETUP THE ERROR LOOP ADDRESS
                                ;;SIZE FOR A HARDWARE SWITCH REGISTER. IF NOT FOUND OR IT IS
                                ;;EQUAL TO A "-1", SETUP FOR A SOFTWARE SWITCH REGISTER.
002540  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE ERROR VECTOR
002544  012737  002600  000004          MOV     #64$,@#ERRVEC   ;;SET UP ERROR VECTOR
002552  012737  177570  001144          MOV     #0SWR,SWR       ;;SETUP FOR A HARDWARE SWICH REGISTER
002560  012737  177570  001146          MOV     #0DISP,DISPLAY  ;;AND A HARDWARE DISPLAY REGISTER
002566  022777  177777  176350          CMP     #-1,@SWR        ;;TRY TO REFERENCE HARDWARE SWR
002574  001012                  BNE     66$             ;;BRANCH IF NO TIMEOUT TRAP OCCURRED
                                                        ;;AND  THE HARDWARE SWR IS NOT = -1
002576  000403                  BR      65$             ;;BRANCH IF NO TIMEOUT
002600  012716  002606          64$:    MOV     #65$,(SP)       ;;SET UP FOR TRAP RETURN
002604  000002                  RTI
002606  012737  000176  001144  65$:    MOV     #SWREG,SWR      ;;POINT TO SOFTWARE SWR
002614  012737  000174  001146          MOV     #DISPREG,DISPLAY
002622  012637  000004          66$:    MOV     (SP)+,@#ERRVEC  ;;RESTORE ERROR VECTOR

002626  005037  001252                  CLR     $PASS           ;;CLEAR PASS COUNT
002632  132737  000200  001265          BITB    #APTSIZE,$ENVM  ;;TEST USER SIZE UNDER APT
002640  001403                  BEQ     67$             ;;YES,USE NON-APT SWITCH
002642  012737  001266  001144          MOV     #$SWREG,SWR     ;;NO,USE APT SWITCH REGISTER
002650                          67$:

                                ;SET UP FPP UNEXPECTED TRAP CATCHER - - - - - - - - -
002650  012737  015160  000244          MOV     #FPPILT,@#FPPVEC        ;NEW PC AT FPP TRAP
002656  012737  000300  000246          MOV     #PR6,@#FPPVEC+2         ;NEW PS AT FPP TRAP
```

```
                                                ;FIND, INITIALIZE ANY LINE CLOCKS PRESENT - - - - - - - - - -
002764  005037  001516                          CLR     CLKPRS          ;CLEAR CLOCK PRESENT FLAG
002770  013746  000004                          MOV     @#ERRVEC,-(SP)          ;SAVE OLD TIMEOUT VECTOR
002774  013746  000006                          MOV     @#ERRVEC+2,-(SP)        ;
003000  012737  002750  000004                  MOV     #NKW11L,@#ERRVEC        ;SET  NEW TIMEOUT VECTOR
003006  012737  000340  000006                  MOV     #PR7,@#ERRVEC+2 ;
003014  005037  177546                          CLR     @#LKS           ;KW11-L THERE ? (CLEAR STATUS IF YES)
003020  052737  000200  001516                  BIS     #BIT7,CLKPRS    ;YES, IT WAS
003026  005037  001520                          CLR     KW11LC          ;CLEAR COUNT
003032  012737  015022  000100                  MOV     #IKW11L,@#LKV   ;KW11-L SERVICE PC, PS
003040  012737  000300  000102                  MOV     #PR6,@#LKV+2    ;
003046  000402                                  BR      TKW11P          ;
002750  062706  000004          NKW11L: ADD     #4,SP           ;NO KW11-L (TIMEOUT); POP JUNK OFF STK
                                                ;
002754  012737  003032  000004  TKW11P: MOV     #NKW11P,@#ERRVEC        ;SET NEW TIMEOUT VECTOR (SAME PS)
                                                ;
002762  005037  172540                          CLR     @#PLKS          ;KW11-P THERE ? (CLEAR STATUS IF YES)
002766  052737  100000  001516                  BIS     #BIT15,CLKPRS   ;YES, IT WAS
002774  005037  001522                          CLR     KW11PC          ;CLEAR COUNT
003000  012737  001777  001524                  MOV     #1777,KW11PR    ;INITIAL COUNT REGISTER
003006  012737  015050  000104                  MOV     #IKW11P,@#PLKV  ;KW11-P SERVICE PC, PS
003014  012737  000300  000106                  MOV     #PR6,@#PLKV+2   ;
003022  013737  001524  172542                  MOV     KW11PR,@#PLKR   ;SET COUNT REGISTER
003030  000402                                  BR      CLKDON          ;
003032  062706  000004          NKW11P: ADD     #4,SP           ;NO KW11-P (TIMEOUT); POP JUNK OFF STK
                                                ;
003036  012637  000006          CLKDON: MOV     (SP)+,@#ERRVEC+2        ;RESTORE OLD TIMEOUT VECTOR
003042  012637  000004                  MOV     (SP)+,@#ERRVEC          ;
                                                ;
003046  104401  001772                  TYPE    ,BGNMES         ;IL MESSAGE AT START
                                                ;
                                                ;////////////////////////////////////////////////////////////////////////
                                                ;       MESSAGE ON WHETHER OR NOT HFP UNIT IS PRESENT
                                                ;
003052  076600  000022                  MFD     ,R0#WHAMI               ;WHAMI INTO R0
003056  032700  000020                  BIT     #BIT04,R0               ;IS THERE A HFP UNIT ?
003062  001403                          BEQ     66$                     ;NO, BR
003064  104401  003100                  TYPE    ,64$                    ;INDICATE FP11-E PRESENT
003070  000453                          BR      NEWPAS                  ;GO FOR SUBPASS INIT
003072  104401  003140          66$:    TYPE    ,65$                    ;INDICATE NO FP11-E
003076  000450                          BR      NEWPAS                  ;GO FOR SUBPASS INIT
                                                ;
003100  005015  020052  050106  64$:    .ASCIZ  <15><12>"* FP11-E HFP UNIT PRESENT *"<15><12>
003106  030461  042455  044040
003114  050106  052440  044516
003122  020124  051120  051505
003130  047105  020124  006452
003136  000012
003140  005015  020052  047516  65$:    .ASCIZ  <15><12>"* NO FP11-E HFP UNIT - ALL TESTS WFP ONLY *"<15><12>
003146  043040  030520  026461
003154  020105  043110  020120
003162  047125  052111  026440
003170  040440  046114  052040
003176  051505  051524  053440
003204  050106  047440  046116
```

```
                                                .EVEN
                                        ;////////////////////////////////////////////////////////////////////////
                                        ;;**********************************************************************
                                        ;NEW PASS ENTERS HERE
                                        ;;**********************************************************************
003220  012706  001100          NEWPAS: MOV     #STACK,SP               ;RESET STACK PTR
003224  032777  010000  175712          BIT     #BIT12,@SWR             ;INHIBIT STATUS TYPEOUTS ?
003232  001011                          BNE     SUBPAS                  ;BR IF YES
003234  104401  002067                  TYPE    ,NWPAS1                 ;"PASS #"
003240  013746  001252                  MOV     $PASS,-(SP)             ;PASS COUNT INTO ...
003244  005216                          INC     (SP)                    ;  1-N RANGE
003246  104403                          TYPOS                           ;TYPE OCTAL
003250     006     000                  .BYTE   6,0                     ;  6 DIGITS, NO LEADING ZEROS
003252  104401  001241                  TYPE    ,$CRLF                  ;END THE LINE

                                        ;;**********************************************************************
                                        ;NEW SUBPASS ENTERS HERE
                                        ;;**********************************************************************
003256  012706  001100          SUBPAS: MOV     #STACK,SP               ;RESET SP FOR INSURANCE
003262  076600  000022                  MED     ,RWHAMI                 ;GET WHAMI INTO R0
003266  032700  000020                  BIT     #BIT04,R0               ;1=HFP PRESENT, 0=NO
003272  001430                          BEQ     20$                     ;IF NO HFP, TEST WARM ONLY
003274  076600  000144                  MED     ,RFLAG                  ;GET FLAGS INTO R0
003300  032777  000002  175636          BIT     #SW01,@SWR              ;SW01:  1=HFP OR HFP TEST ONLY
003306  001413                          BEQ     1$                      ;       0=ALTERNATE HFP/WFP PER PASS
003310  032777  000001  175626          BIT     #SW00,@SWR              ;SW00:  1=WFP ONLY
003316  001403                          BEQ     2$                      ;       0=HFP ONLY
003320  042700  010000                  BIC     #BIT12,R0               ;CLEAR HFP ENABLE FLAG<5> FOR WFP
003324  000402                          BR      3$                      ;
003326  052700  010000          2$:     BIS     #BIT12,R0               ;SET HFP ENABLE FLAG<5> FOR HFP
003332  076600  000344          3$:     MED     ,WFLAG                  ;REWRITE FLAGS
003336  032700  010000          1$:     BIT     #BIT12,R0               ;TEST WHO'S ENABLED: HOT, WARM
003342  001404                          BEQ     20$                     ;SET APPROPRIATE HEADER:
003344  012737  017414  015766  19$:     MOV     #ASCHOT,HOTWRM          ;"HOT: "
003352  000403                          BR      21$                     ;
003354  012737  017422  015766  20$:     MOV     #ASCWRM,HOTWRM          ;"WARM: "
003362                          21$:

                                        ;*START KW11-L LINE CLOCK IF PRESENT
003362  105737  001516                  TSTB    CLKPRS                  ;KW11-L PRESENT ?
003366  100003                          BPL     NKWL1                   ;NO
```

```
611  003370  052737  000100  177546          BIS     #BIT6,@#LKS     ;YES, START IT
612
613                                          ;*START KW11-P PROGRAMMABLE LINE CLOCK, IF PRESENT
614  003376  005737  001516          NKWL1:  TST     CLKPRS          ;KW11-P PRESENT ?
615  003402  100003                  BPL     NKWP1           ;NO
616  003404  052737  000101  172540          BIS     #BIT6+BIT0,@#PLKS       ;YES, START IT
617
618  003412  005037  177776          NKWP1:  CLR     @#PSW           ;SETUP FOR INTERRUPT ENABLE, PRO
619
620
```

```
                                ;;*****************************************************************
                                        .DSABL  AMA             ;ASSEMBLE ALL REFERENCES AS THEY ARE PRINTED
                                ;;*****************************************************************

                                ;;*****************************************************************
                                ;*TEST 1         TEST OF WRITABILITY OF FPS
                                ;;*****************************************************************
003416  000004                  TST1:   SCOPE
003420  170127  147757                  LDFPS   #147757         ;TEST FPS WITH ALL ONES
003424  170267  176034                  STFPS   FPS             ;GET STATUS
003430  022767  147757  176026          CMP     #147757,FPS     ;CHECK IT
003436  001401                          BEQ     .+4
003440  104001                          ERROR   1               ;FPS NOT 147777

003442  170127  040000                  LDFPS   #40000          ;TEST FPS WITH 40000
003446  170267  176012                  STFPS   FPS             ;STORE FLOATING POINT STATUS
003452  022767  040000  176004          CMP     #40000,FPS      ;CHECK FLOATING POINT STATUS
003460  001401                          BEQ     .+4             ;BRANCH IF OK
003462  104001                          ERROR   1               ;FPS NOT EQUAL TO 40000

003464  170011                          SETD                    ;TEST OF DOUBLE BIT ON A 1
003466  170267  175772                  STFPS   FPS             ;STORE FLOATING POINT STATUS
003472  022767  040200  175764          CMP     #40200,FPS      ;CHECK FLOATING POINT STATUS
003500  001401                          BEQ     .+4             ;BRANCH IF OK
003502  104001                          ERROR   1               ;FPS NOT EQUAL TO 40200

003504  170001                          SETF                    ;TEST OF DOUBLE BIT ON A 0
003506  170267  175752                  STFPS   FPS             ;STORE FLOATING POINT STATUS
003512  022767  040000  175744          CMP     #40000,FPS      ;CHECK FLOATING POINT STATUS
003520  001401                          BEQ     .+4             ;BRANCH IF OK
003522  104001                          ERROR   1               ;FPS NOT EQUAL TO 40000

003524  170012                          SETL                    ;TEST OF LONG BIT ON A 1
003526  170267  175732                  STFPS   FPS             ;STORE FLOATING POINT STATUS
003532  022767  040100  175724          CMP     #40100,FPS      ;CHECK FLOATING POINT STATUS
003540  001401                          BEQ     .+4             ;BRANCH IF OK
003542  104001                          ERROR   1               ;FPS NOT EQUAL TO 40100

003544  170002                          SETI                    ;TEST OF LONG BIT ON A 0
003546  170267  175712                  STFPS   FPS             ;STORE FLOATING POINT STATUS
003552  022767  040000  175704          CMP     #40000,FPS      ;CHECK FLOATING POINT STATUS
003560  001401                          BEQ     .+4             ;BRANCH IF OK
003562  104001                          ERROR   1               ;FPS NOT EQUAL TO 40000
```

```
                                        ;;*****************************************************************
                                        ;*TEST 2         TEST OF CFCC
                                        ;;*****************************************************************
003564  000004                  TST2:   SCOPE
003566  170127  040017                  LDFPS   #40017          ;LOAD ALL STATUS BITS TO 1'S
003572  170000                          CFCC                    ;GET THEM INTO PS
003574  013700  177776                  MOV     @#PS,R0         ;GET THEM FOR TYPING
003600  042700  177760                  BIC     #177760,R0      ;CLEAR JUNK
003604  022700  000017                  CMP     #17,R0          ;ALL SET?
003610  001401                          BEQ     .+4
003612  104004                          ERROR   4               ;PS NOT 17

003614  170127  040012                  LDFPS   #40012          ;LOAD FPS WITH 12
003620  170000                          CFCC                    ;GET INTO PS
003622  013700  177776                  MOV     @#PS,R0         ;GET FOR TYPING
003626  042700  177760                  BIC     #177760,R0      ;CLEAR JUNK
003632  022700  000012                  CMP     #12,R0          ;SAME AS LD?
003636  001401                          BEQ     .+4
003640  104004                          ERROR   4               ;PS NOT 12

003642  170127  040005                  LDFPS   #40005          ;LOAD FPS WITH 5
003646  170000                          CFCC                    ;GET BITS
003650  013700  177776                  MOV     @#PS,R0         ;GET FOR TYPING
003654  042700  177760                  BIC     #177760,R0      ;CLEAR JUNK
003660  022700  000005                  CMP     #5,R0           ;SAME?
003664  001401                          BEQ     .+4
003666  104004                          ERROR   4               ;PS NOT 5
```

```
                                        ;;*****************************************************************
                                        ;*TEST 3        TEST OF LDD, STD, CMPD OF -1,0,-1,0
                                        ;;*****************************************************************
003670  000004                  ;ST3:   SCOPE
003672  170127  047600                  LDFPS   #47600
003676  012700  001556                  MOV     #D1010,R0       ;GET ADDRESS OF DATA WORD
003702  172420                          LDD     (R0)+,AC0       ;LOAD INTO AC0
003704  022700  001566                  CMP     #D1010+10,R0    ;IS THE NEW ADDRESS RIGHT?
003710  001401                          BEQ     .+4
003712  104005                          ERROR   5               ;R0 NOT D1010+10

003714  010601                          MOV     SP,R1
003716  162701  000010                  SUB     #10,R1
003722  174046                          STD     AC0,-(SP)       ;GET THE DATA BACK
003724  010600                          MOV     SP,R0           ;SAVE THE SP FOR TYPING
003726  020106                          CMP     R1,SP           ;SP DECREMENTED PROPERLY?
003730  001401                          BEQ     .+4
003732  104006                          ERROR   6               ;SP NOT SP-10

003734  170267  175524                  STFPS   FPS             ;STORE FLOATING POINT STATUS
003740  022767  047610  175516          CMP     #47610,FPS      ;CHECK FLOATING POINT STATUS
003746  001401                          BEQ     .+4             ;BRANCH IF OK
003750  104001                          ERROR   1               ;FPS NOT EQUAL TO 47610

003752  010602                          MOV     SP,R2           ;MOVE ANSWER TO DISPLAY
003754  012703  001170                  MOV     #$REG0,R3       ;
003760  012223                          MOV     (R2)+,(R3)+     ;
003762  012223                          MOV     (R2)+,(R3)+     ;
003764  012223                          MOV     (R2)+,(R3)+     ;
003766  012223                          MOV     (R2)+,(R3)+     ;
003770  021667  175562                  CMP     (SP),D1010      ;CHECK FIRST PIECE OF DATA
003774  001401                          BEQ     .+4
003776  104010                          ERROR   10              ;DATA IN (SP) NOT D1010
004000  026667  000002  175552          CMP     2(SP),D1010+2   ;CHECK SECOND
004006  001401                          BEQ     .+4
004010  104010                          ERROR   10              ;DATA IN 2(SP) NOT D1010+2
004012  026667  000004  175542          CMP     4(SP),D1010+4   ;CHECK THIRD
004020  001401                          BEQ     .+4
004022  104010                          ERROR   10              ;DATA IN 4(SP) NOT D1010+4
004024  026667  000006  175532          CMP     6(SP),D1010+6   ;CHECK FOURTH
004032  001401                          BEQ     .+4
004034  104010                          ERROR   10              ;DATA IN 6(SP) NOT D1010+6

004036  062701  000010                  ADD     #10,R1
004042  173426                          CMPD    (SP)+,AC0       ;RECHECK DATA AND SP
004044  010600                          MOV     SP,R0           ;SAVE SP FOR TYPING
004046  020106                          CMP     R1,SP           ;CHECK ADDRESS IN SP
004050  001401                          BEQ     .+4
004052  104006                          ERROR   6               ;SP NOT RESTORED
004054  170267  175404                  STFPS   FPS             ;GET STATUS
004060  170000                          CFCC                    ;NOW GET THE FP CONDITION CODES
004062  001401                          BEQ     .+4             ;IF IT HALTS HERE IT MUST BE THE
004064  104010                          ERROR   10              ;CMPD BECAUSE CFCC IS ALREADY CONFIRMED
```

```
                                        ;;*****************************************************************
                                        ;*TEST 4        TEST OF LDD, STD, CMPD OF 0,-1,0,-1
                                        ;;*****************************************************************
                                        TST4:   SCOPE
004066  000004                                  LDFPS   #47600
004070  170127  047600                          LDD     @#00101,AC0     ;LOAD 0,-1,0,-1 INTO AC0 *PIC*
004074  172437  001560                          STFPS   FPS             ;STORE FLOATING POINT STATUS
004100  170267  175360                          CMP     #47604,FPS      ;CHECK FLOATING POINT STATUS
004104  022767  047604  175352                  BEQ     .+4             ;BRANCH IF OK
004112  001401                                  ERROR   1               ;FPS NOT EQUAL TO 47604
004114  104001
                                                MOV     #$REG0,R0       ;ADDRESS TO BE STORED INTO
004116  012700  001170                          STD     AC0,(R0)        ;STORE IT INTO $REG0-3 *PIC*
004122  174010                                  CMP     D0101,$REG0     ;FIRST WORD OK?
004124  026767  175430  175036                  BEQ     .+4
004132  001401                                  ERROR   10              ;$REG0 NOT D0101
004134  104010                                  CMP     D0101+2,$REG1   ;SECOND
004136  026767  175420  175026                  BEQ     .+4
004144  001401                                  ERROR   10              ;$REG1 NOT D0101+2
004146  104010                                  CMP     D0101+4,$REG2   ;THIRD
004150  026767  175410  175016                  BEQ     .+4
004156  001401                                  ERROR   10              ;$REG2 NOT D0101+4
004160  104010                                  CMP     D0101+6,$REG3   ;FOURTH
004162  026767  175400  175006                  BEQ     .+4
004170  001401                                  ERROR   10              ;$REG3 NOT D0101+6
004172  104010
                                                MOV     #A00101-2,R4    ;ADDRESS-2 OF DATA
004174  012704  001602                          CMPD    2(4),AC0        ;CHECK DATA IN AC0 *PIC*
004200  173474  000002                                                  ;NOTE:  CMPD SEES A ZERO EXP IN AC0, AND THUS
                                                                        ;  AFTER ITS EXECUTION AC0 <- (0,0,0,0),
                                                                        ;  A TRUE FLTPT ZERO
004204  170267  175254                          STFPS   FPS             ;GET STATUS
004210  170000                                  CFCC                    ;GET CONDITION CODES
004212  001401                                  BEQ     .+4
004214  104010                                  ERROR   10              ;CMPD FAILED
```

```
                                ;;*****************************************************************
                                ;*TEST 5         TEST OF LDF, STF, CMPF OF -1,0
                                ;;*****************************************************************
004216  000004                  TST5:   SCOPE
                                                                ;NOTE:  AC0 = (0,0,0,0) FROM PREV TEST
004220  170127  047400                  LDFPS   #47400          ;SET FLOATING MODE
004224  172467  175326                  LDF     D1010,AC0       ;LOAD -1.0 INTO AC0
004230  012700  001170                  MOV     #$REG0,R0       ;POINTED TO ANSWER AREA *PIC*
004234  174020                          STF     AC0,(R0)+       ;STORE RESULT
004236  022700  001174                  CMP     #$REG2,R0       ;INCREMENTED PROPERLY
004242  001401                          BEQ     .+4
004244  104005                          ERROR   5               ;R0 NOT $REG0+4
                                
004246  026767  175304  174714          CMP     D1010,$REG0     ;CHECK FIRST WORD
004254  001401                          BEQ     .+4
004256  104011                          ERROR   11              ;$REG0 NOT D1010
004260  026767  175274  174704          CMP     D1010+2,$REG1   ;SECOND
004266  001401                          BEQ     .+4
004270  104011                          ERROR   11              ;$REG1 NOT D1010+2
                                
004272  170011                          SETD                    ;GO TO DOUBLE MODE
004274  012700  001200                  MOV     #$REG4,R0       ;ADDRESS OF DATA+10
004300  174040                          STD     AC0,-(R0)       ;GET DATA
004302  022700  001170                  CMP     #$REG0,R0       ;CHECK FOR PROPER DECREMENTATION
004306  001401                          BEQ     .+4
004310  104005                          ERROR   5               ;R0 NOT $REG0
                                
004312  012700  001610                  MOV     #AD1000,R0      ;LOAD ADDRESS OF ADDRESS OF DATA
004316  173430                          CMPD    @(R0)+,AC0      ;CHECK THE DATA
004320  022700  001612                  CMP     #AD1000+2,R0    ;R0 GETS INCREMENTED BY 2
004324  001401                          BEQ     .+4
004326  104005                          ERROR   5               ;R0 NOT AD1001+2
                                
004330  170267  175130                  STFPS   FPS             ;GET STATUS
004334  170000                          CFCC                    ;COPY CONDITION CODES
004336  001401                          BEQ     .+4             ;EITHER CMPD FAILED OR THE
004340  104011                          ERROR   11              ;LDF MODIFIED RIGHT HALF
```

```
                                ;;*****************************************************************
                                ;*TEST 6         TEST OF LDF, STF, CMPF WITH <=> IN ALL AC'S
                                ;;*****************************************************************
                                ;
004342  000004                  TST6:   SCOPE
004344  170127  047400                  LDFPS   #47400          ;LOAD FLOATING MODE
004350  172427  140640                  LDF     #-5,AC0         ;LOAD AC0 WITH -5
004354  172527  140200                  LDF     #-1,AC1         ;LOAD AC1 WITH -1
004360  172627  000000                  LDF     #0,AC2          ;LOAD AC2 WITH 0
004364  172727  040640                  LDF     #5,AC3          ;LOAD AC3 WITH 5
004370  174004                          STF     AC0,AC4         ;LOAD AC4 WITH -5
004372  174305                          STF     AC3,AC5         ;LOAD AC5 WITH 5
                                ;
004374  174067  174570                  STF     AC0,$REG0       ;GET AC0
004400  173427  140640                  CMPF    #-5,AC0         ;CHECK IT
004404  170267  175054                  STFPS   FPS             ;GET STATUS
004410  170000                          CFCC                    ;GET CC
004412  001401                          BEQ     .+4
004414  104011                          ERROR   11              ;AC0 NOT -5
                                ;
004416  174137  001170                  STF     AC1,@#$REG0     ;GET AC1
004422  173567  174542                  CMPF    $REG0,AC1       ;CHECK IT
004426  170267  175032                  STFPS   FPS             ;GET STATUS
004432  170000                          CFCC                    ;GET CC
004434  001401                          BEQ     .+4
004436  104011                          ERROR   11              ;AC1 NOT -1
004440  012704  001170                  MOV     #$REG0,R4       ;POINTER TO ANSWER AREA
004444  174214                          STF     AC2,(4)         ;PUT DATA INTO $REG0
004446  173667  174516                  CMPF    $REG0,AC2       ;CHECK DATA
004452  170267  175006                  STFPS   FPS             ;GET STATUS
004456  170000                          CFCC                    ;GET CC
004460  001401                          BEQ     .+4
004462  104011                          ERROR   11              ;AC2 NOT 0
                                ;
004464  174377  175142                  STF     AC3,@#$REG0     ;PUT DATA INTO $REG0
004470  173767  174474                  CMPF    $REG0,AC3       ;CHECK DATA
004474  170267  174764                  STFPS   FPS             ;GET STATUS
004500  170000                          CFCC                    ;GET CC
004502  001401                          BEQ     .+4
004504  104011                          ERROR   11              ;AC3 NOT 5
                                ;
004506  173404                          CMPF    AC4,AC0         ;CHECK AC4 FOR -5
004510  170267  174750                  STFPS   FPS             ;GET STATUS
004514  170000                          CFCC                    ;GET CC
004516  001401                          BEQ     .+4
004520  104000                          ERROR   0               ;AC0 NOT AC4
                                ;
004522  173705                          CMPF    AC5,AC3         ;CHECK AC5 FOR 5
004524  170267  174734                  STFPS   FPS             ;GET STATUS
004530  170000                          CFCC                    ;GET CC
004532  001401                          BEQ     .+4
004534  104000                          ERROR   0               ;AC3 NOT AC5
```

```
                                ;;*****************************************************************
                                ;*TEST 7        TEST OF CMPF WITH DATA IN AC0-AC5
                                ;;*****************************************************************
004536  000004                  TST7:   SCOPE
004540  170127  041000                  LDFPS   #41000          ;LOAD STATUS WITH 0
004544  173405                  CMPF    AC5,AC0         ;CMP 5 TO -5
004546  170267  174712                  STFPS   FPS             ;STORE FLOATING POINT STATUS
004552  022767  041000  174704          CMP     #41000,FPS      ;CHECK FLOATING POINT STATUS
004560  001401                  BEQ     .+4             ;BRANCH IF OK
004562  104001                  ERROR   1               ;FPS NOT EQUAL TO 41000

004564  173704                  CMPF    AC4,AC3         ;CMP -5 TO 5
004566  170267  174672                  STFPS   FPS             ;STORE FLOATING POINT STATUS
004572  022767  041010  174664          CMP     #41010,FPS      ;CHECK FLOATING POINT STATUS
004600  001401                  BEQ     .+4             ;BRANCH IF OK
004602  104001                  ERROR   1               ;FPS NOT EQUAL TO 41010

004604  173767  175002                  CMPF    DBIG,AC3        ;MAKE IT OVERFLOW
004610  170267  174650                  STFPS   FPS             ;GET STATUS
004614  022767  041000  174642          CMP     #41000,FPS      ;CHECK FLOATING POINT STATUS
004622  001401                  BEQ     .+4
004624  104001                  ERROR   1               ;FPS NOT 141002
```

```
                                ;;*****************************************************************
                                ;*TEST 10       TEST OF TSTF AND TSTD USING OLD AC0-AC5
                                ;;*****************************************************************
                                ;
004626  000004                  TST10:  SCOPE
004630  170127  040000                  LDFPS   #40000
004634  170501                          TSTF    AC1             ;TEST AC1 = -1
004636  170267  174622                  STFPS   FPS             ;GET STATUS
004642  022767  040010  174614          CMP     #40010,FPS      ;CHECK STATUS
004650  001401                          BEQ     .+4
004652  104001                          ERROR   1               ;N BIT NOT SET

004654  170505                          TSTF    AC5             ;TEST AC5 = 5
004656  170267  174602                  STFPS   FPS             ;GET STATUS
004662  022767  040000  174574          CMP     #40000,FPS      ;CHECK STATUS
004670  001401                          BEQ     .+4
004672  104001                          ERROR   1               ;NOT 0

004674  170502                          TSTF    AC2             ;TEST AC2 = 0
004676  170267  174562                  STFPS   FPS             ;GET STATUS
004702  022767  040004  174554          CMP     #40004,FPS      ;CHECK STATUS
004710  001401                          BEQ     .+4
004712  104001                          ERROR   1               ;Z BIT NOT SET

004714  170527  177777                  TSTF    #-1             ;TEST FOR THE N BIT IN LINE
004720  000401                          BR      .+4             ;SHOULD GO HERE
004722  104001                          ERROR   1               ;INCREMENTED BY 4 NOT 2
004724  170267  174534                  STFPS   FPS             ;GET STATUS
004730  022767  040010  174526          CMP     #40010,FPS      ;CHECK THE N BIT
004736  001401                          BEQ     .+4
004740  104001                          ERROR   1               ;N BIT NOT SET

004742  170011                          SETD                    ;SET DOUBLE MODE
004744  170527  000000                  TSTD    #0              ;TEST FOR Z BIT IN LINE
004750  000403                          BR      .+10            ;SHOULD GO HERE
004752  104001                          ERROR   1               ;NOT HERE
004754  104001                          ERROR   1               ;OR HERE
004756  104001                          ERROR   1               ;OR HERE
004760  170267  174500                  STFPS   FPS             ;GET STATUS
004764  022767  040204  174472          CMP     #40204,FPS      ;CHECK STATUS
004772  001401                          BEQ     .+4
004774  104001                          ERROR   1               ;Z BIT NOT SET
```

```
                                ;;*****************************************************************
                                ;*TEST 11       TEST OF CLRX INSTRUCTIONS
                                ;;*****************************************************************
005776  000004                  TST11:  SCOPE
005000  170127  040200                  LDFPS   #40200          ;DOUBLE MODE
005004  172467  174546                  LDD     D1010,AC0       ;LOAD A -1
005010  174067  174154                  STD     AC0,$REG0       ;PUT INTO $REG0
005014  170001                          SETF                    ;SET FLOATING
005016  170467  174146                  CLRF    $REG0           ;CLEAR IT OUT
005022  170267  174436                  STFPS   FPS             ;STORE FLOATING POINT STATUS
00 26   022767  040004  174430          CMP     #40004,FPS      ;CHECK FLOATING POINT STATUS
005034  001401                          BEQ     .+4             ;BRANCH IF OK
005036  104001                          ERROR   1               ;FPS NOT EQUAL TO 40004
                                
005040  170567  174124                  TSTF    $REG0           ;TEST FOR ZERO
005044  170000                          CFCC                    ;GET CC
005046  001401                          BEQ     .+4
005050  104011                          ERROR   11              ;AC0 NOT ZERO
                                
005052  026767  174504  174114          CMP     D1010+4,$REG2   ;CHECK THIRD WORD
005060  001401                          BEQ     .+4
005062  104010                          ERROR   10              ;$REG2 NOT -1
00 64   026767  174474  174104          CMP     D1010+6,$REG3   ;CHECK FOURTH
00 72   001401                          BEQ     .+4
005074  104010                          ERROR   10              ;$REG3 NOT 0
                                
005076  170011                          SETD
005100  170100                          CLRD    AC0             ;CLEAR THE REST OF AC0
005102  170267  174356                  STFPS   FPS             ;STORE FLOATING POINT STATUS
005106  022767  040204  174350          CMP     #40204,FPS      ;CHECK FLOATING POINT STATUS
005114  001401                          BEQ     .+4             ;BRANCH IF OK
005116  104001                          ERROR   1               ;FPS NOT EQUAL TO 40204
005120  174067  174044                  STD     AC0,$REG0       ;STORE RESULT
005124  170500                          TSTD    AC0             ;DID IT CLEAR
005126  170000                          CFCC                    ;GET STATUS
005130  001401                          BEQ     .+4
005132  104010                          ERROR   10              ;DID NOT CLEAR
                                
005134  172467  174462                  LDD     DMZERO,AC0      ;LOAD A MINUS ZERO
0  140  170400                          CLRD    AC0             ;CLEAR IT OUT
005142  170267  174316                  STFPS   FPS             ;STORE FLOATING POINT STATUS
005146  022767  040204  174310          CMP     #40204,FPS      ;CHECK FLOATING POINT STATUS
005154  001401                          BEQ     .+4             ;BRANCH IF OK
005156  104001                          ERROR   1               ;FPS NOT EQUAL TO 40204
005160  174067  174004                  STD     AC0,$REG0       ;STORE RESULT
005164  170500                          TSTD    AC0             ;CHECK IT
005166  170000                          CFCC                    ;GET CC
005170  001401                          BEQ     .+4
005172  104010                          ERROR   10              ;DID NOT CLEAR
```

```
                        ;;*****************************************************************
                        ;*TEST 12        TEST OF NEGX
                        ;;*****************************************************************
005174 000004           TST12:  SCOPE
005176 170127 040000            LDFPS   #40000
005202 172427 140640            LDF     #-5,AC0         ;LOAD AC0 WITH -5
005206 170700                   NEGF    AC0             ;MAKE IT 5
005210 170267 174250            STFPS   FPS             ;STORE FLOATING POINT STATUS
005214 022767 040000 174242     CMP     #40000,FPS      ;CHECK FLOATING POINT STATUS
005222 001401                   BEQ     .+4             ;BRANCH IF OK
005224 104001                   ERROR   1               ;FPS NOT EQUAL TO 40000

005226 174067 173736            STF     AC0,$REG0       ;GET THE RESULT
005232 173427 040640            CMPF    #5,AC0          ;CHECK THE RESULT
005236 170000                   CFCC                    ;GET CC
005240 001401                   BEQ     .+4
005242 104011                   ERROR   11              ;RESULT NOT 5

005244 170767 173720            NEGF    $REG0           ;MAKE IT -5
005250 170267 174210            STFPS   FPS             ;STORE FLOATING POINT STATUS
005254 022767 040010 174202     CMP     #40010,FPS      ;CHECK FLOATING POINT STATUS
005262 001401                   BEQ     .+4             ;BRANCH IF OK
005264 104001                   ERROR   1               ;FPS NOT EQUAL TO 40010

005266 022767 140640 173674     CMP     #140640,$REG0   ;CHECK THE RESULT
005274 001401                   BEQ     .+4
005276 104011                   ERROR   11              ;RESULT NOT -5

005300 005767 173666            TST     $REG1           ;REST 0?
005304 001401                   BEQ     .+4             ;SKIP IF OK
005306 104011                   ERROR   11

005310 170127 047400            LDFPS   #47400          ;TURN ON INTERUPTS
005314 170400                   CLRF    AC0             ;CLEAR AC0
005316 170700                   NEGF    AC0             ;NEGATE IT
005320 170267 174140            STFPS   FPS             ;STORE FLOATING POINT STATUS
005324 022767 047404 174132     CMP     #47404,FPS      ;CHECK FLOATING POINT STATUS
005332 001401                   BEQ     .+4             ;BRANCH IF OK
005334 104001                   ERROR   1               ;FPS NOT EQUAL TO 47404

005336 174067 173626            STF     AC0,$REG0       ;GET RESULT
005342 170500                   TSTF    AC0             ;CHECK IT
005344 170000                   CFCC                    ;GET CC
005346 001401                   BEQ     .+4
005350 104011                   ERROR   11              ;RESULT NOT 0

005352 170011                   SETD                    ;SET DOUBLE MODE
005354 170400                   CLRD    AC0             ;CLEAR AC0
005356 170700                   NEGD    AC0             ;NEGATE AC0
005360 170267 174100            STFPS   FPS             ;STORE FLOATING POINT STATUS
005364 022767 047604 174072     CMP     #47604,FPS      ;CHECK FLOATING POINT STATUS
005372 001401                   BEQ     .+4             ;BRANCH IF OK
005374 104001                   ERROR   1               ;FPS NOT EQUAL TO 47604

005376 174067 173566            STF     AC0,$REG0       ;GET RESULT
005402 170500                   TSTF    AC0             ;TEST RESULT
```

```
1037  005404  170000                  CFCC                    ;GET CC
1038  005406  001401                  BEQ     .+4
1039  005410  104011                  ERROR   11              ;RESULT NOT 0
1040
```

```
                                ;;*****************************************************************
                                ;*TEST 13        TEST OF ABSX
                                ;;*****************************************************************
005412  000004                  TST13:  SCOPE
005414  170127  040000                  LDFPS   #40000
005420  172427  140200                  LDF     #-1,AC0         ;LOAD A -1
005424  174067  173540                  STF     AC0,$REG0       ;GET IT
005430  170667  173534                  ABSF    $REG0           ;MAKE IT A 1
005434  170267  174024                  STFPS   FPS             ;STORE FLOATING POINT STATUS
005440  022767  040000  174016          CMP     #40000,FPS      ;CHECK FLOATING POINT STATUS
005446  001401                          BEQ     .+4             ;BRANCH IF OK
005450  104001                          ERROR   1               ;FPS NOT EQUAL TO 40000
                                
005452  022767  040200  173510          CMP     #40200,$REG0    ;CHECK FOR A 1
005460  001401                          BEQ     .+4
005462  104011                          ERROR   11              ;RESULT NOT 1
                                
005464  005767  173502                  TST     $REG1           ;REST 0?
005470  001401                          BEQ     .+4             ;SKIP IF OK
005472  104011                          ERROR   11
                                
005474  170600                          ABSF    AC0             ;ABS IT AGAIN
005476  170267  173762                  STFPS   FPS             ;GET STATUS
005502  174067  173462                  STF     AC0,$REG0       ;STORE ANSWER
005506  173427  040200                  CMPF    #1,AC0          ;CHECK FOR A 1
005512  170000                          CFCC                    ;GET CC
005514  001401                          BEQ     .+4
005516  104011                          ERROR   11              ;RESULT NOT 1
                                
005520  172467  174076                  LDF     DMZERO,AC0      ;LOAD A MINUS ZERO
005524  170600                          ABSF    AC0             ;ABS IT
005526  170267  173732                  STFPS   FPS             ;STORE FLOATING POINT STATUS
005532  022767  040004  173724          CMP     #40004,FPS      ;CHECK FLOATING POINT STATUS
005540  001401                          BEQ     .+4             ;BRANCH IF OK
005542  104001                          ERROR   1               ;FPS NOT EQUAL TO 40004
                                
005544  174067  173420                  STF     AC0,$REG0       ;GET RESULT
005550  170500                          TSTF    AC0             ;TEST FOR 0
005552  170000                          CFCC                    ;GET CC
005554  001401                          BEQ     .+4
005556  104011                          ERROR   11              ;RESULT NOT 0
                                
005560  170011                          SETD                    ;SET DOUBLE MODE
005562  172467  173770                  LDD     D1010,AC0       ;LOAD -1,0,-1,0
005566  170600                          ABSD    AC0             ;ABS IT
005570  170267  173670                  STFPS   FPS             ;STORE FLOATING POINT STATUS
005574  022767  040200  173662          CMP     #40200,FPS      ;CHECK FLOATING POINT STATUS
005602  001401                          BEQ     .+4             ;BRANCH IF OK
005604  104001                          ERROR   1               ;FPS NOT EQUAL TO 40200
005606  174067  173356                  STD     AC0,$REG0       ;GET RESULT
005612  173467  173774                  CMPD    DBIG,AC0        ;CHECK THE RESULT
005616  170000                          CFCC                    ;GET CC
005620  001401                          BEQ     .+4
005622  104010                          ERROR   10              ;RESULT NOT 77777,0,-1,0
```

```
                                ;;*****************************************************************
                                ;*TEST 14        TEST OF LDEXP & STEXP
                                ;;*****************************************************************
005624  000004                  TST14:  SCOPE
005626  170127  040200                  LDFPS   #40200          ;SET DOUBLE MODE
005632  172467  173720                  LDD     D1010,AC0       ;LOAD A -1,0,-1,0
005636  176427  177600                  LDEXP   #-200,AC0       ;CLEAR THE EXPONENT
005642  170267  173616                  STFPS   FPS             ;STORE FLOATING POINT STATUS
005646  022767  040204  173610          CMP     #40204,FPS      ;CHECK FLOATING POINT STATUS
005654  001401                          BEQ     .+4             ;BRANCH IF OK
005656  104001                          ERROR   1               ;FPS NOT EQUAL TO 40204

005660  174067  173304                  STD     AC0,$REG0       ;GET THE RESULT
005664  170500                          TSTD    AC0             ;IS IT 0
005666  170000                          CFCC
005670  001401                          BEQ     .+4
005672  104010                          ERROR   10              ;AC0 NOT 0

005674  175067  173270                  STEXP   AC0,$REG0       ;GET THE RESULT
005700  013700  177776                  MOV     @#PS,R0         ;GET PS BITS
005704  042700  177760                  BIC     #177760,R0      ;CLEAR JUNK
005710  022700  000010                  CMP     #10,R0          ;IS IT OK?
005714  001401                          BEQ     .+4             ;SKIP IF OK
005716  104004                          ERROR   4               ;PS IS WRONG

005720  022767  177600  173242          CMP     #-200,$REG0     ;CHECK IT
005726  001401                          BEQ     .+4
005730  104012                          ERROR   12              ;EXPONENT NOT 0

005732  170001                          SETF                    ;SET FLOATING MODE
005734  172467  173620                  LDF     D0101,AC0       ;LOAD A 0,-1
005740  176427  000200                  LDEXP   #200,AC0        ;SET EXPONENT TO -1
005744  170267  173514                  STFPS   FPS             ;STORE FLOATING POINT STATUS
005750  022767  040006  173506          CMP     #40006,FPS      ;CHECK FLOATING POINT STATUS
005756  001401                          BEQ     .+4             ;BRANCH IF OK
005760  104001                          ERROR   1               ;FPS NOT EQUAL TO 40006

005762  174067  173202                  STF     AC0,$REG0       ;SAVE RESULT
005766  005767  173176                  TST     $REG0           ;CHECK FIRST WORD
005772  001401                          BEQ     .+4
005774  104011                          ERROR   11              ;$REG0 NOT 0

005776  005767  173170                  TST     $REG1           ;CHECK SECOND WORD
006002  001401                          BEQ     .+4
006004  104011                          ERROR   11              ;$REG1 NOT ZERO

006006  175067  173156                  STEXP   AC0,$REG0       ;GET THE EXPONENT BACK
006012  022767  177600  173150          CMP     #-200,$REG0     ;CHECK IT
006020  001401                          BEQ     .+4
006022  104012                          ERROR   12              ;EXPONENT NOT -200

006024  176527  000052                  LDEXP   #52,AC1         ;LOAD ALT 1'S
006030  175100                          STEXP   AC1,R0          ;GET THEM BACK
006032  022700  000052                  CMP     #52,R0          ;OK?
006036  001401                          BEQ     .+4
006040  104013                          ERROR   13              ;EXP NOT 252
```

```
1152
1153  006042  176627  000025          LDEXP   #25,AC2         ;LOAD OTHER ALT 1'S
1154  006046  175200                  STEXP   AC2,R0          ;GET IT BACK
1155  006050  022700  000025          CMP     #25,R0          ;CHECK IT
1156  006054  001401                  BEQ     .+4
1157  006056  104013                  ERROR   13              ;EXP NOT 125
1158
```

```
1159                                    ;;*****************************************************************
1160                                    ;*TEST 15       TEST OF ADDF & SUBF
1161                                    ;;*****************************************************************
1162  006060  000004                   TST15:  SCOPE
1163  006062  170127  047400                   LDFPS   #47400          ;LOAD FLOATING MODE
1164  006066  172427  040600                   LDF     #4,AC0          ;LOAD A 4
1165  006072  172027  040400                   ADDF    #2,AC0          ;ADD A 2
1166  006076  170267  173362                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1167  006102  022767  047400  173354           CMP     #47400,FPS      ;CHECK FLOATING POINT STATUS
1168  006110  001401                           BEQ     .+4             ;BRANCH IF OK
1169  006112  104001                           ERROR   1               ;FPS NOT EQUAL TO 47400
1170
1171  006114  174067  173050                   STF     AC0,$REG0       ;STORE RESULT
1172  006120  173427  040700                   CMPF    #6,AC0          ;CHECK RESULT
1173  006124  170000                           CFCC                    ;GET CC
1174  006126  001401                           BEQ     .+4
1175  006130  104011                           ERROR   11              ;RESULT NOT 6
1176
1177  006132  173027  041020                   SUBF    #9,AC0          ;SUBTRACT 9 FROM 6
1178  006136  170267  173322                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1179  006142  022767  047410  173314           CMP     #47410,FPS      ;CHECK FLOATING POINT STATUS
1180  006150  001401                           BEQ     .+4             ;BRANCH IF OK
1181  006152  104001                           ERROR   1               ;FPS NOT EQUAL TO 47410
1182
1183  006154  174067  173010                   STF     AC0,$REG0       ;STORE RESULT IN $REG0
1184  006160  173427  140500                   CMPF    #-3,AC0         ;CHECK RESULT
1185  006164  170000                           CFCC                    ;GET CC
1186  006166  001401                           BEQ     .+4
1187  006170  104011                           ERROR   11              ;6 - 9 NOT -3?
1188
1189
```

```
1190                                    ;;*****************************************************************
1191                                    ;*TEST 16       TEST OF ADDD AND SUBD
1192                                    ;;*****************************************************************
1193  006172  000004               TST16:  SCOPE
1194  006174  170127  047600               LDFPS   #47600          ;SET DOUBLE MODE
1195  006200  172527  141400               LDD     #-32.,AC1       ;LOAD A -32.
1196  006204  172127  040640               ADDD    #5,AC1          ;ADD 5 TO -32.
1197  006210  170267  173250               STFPS   FPS             ;STORE FLOATING POINT STATUS
1198  006214  022767  047610  173242       CMP     #47610,FPS      ;CHECK FLOATING POINT STATUS
1199  006222  001401                       BEQ     .+4             ;BRANCH IF OK
1200  006224  104001                       ERROR   1               ;FPS NOT EQUAL TO 47610
1201
1202  006226  174177  173400               STD     AC1,@#$REGO     ;GET RESULT
1203  006232  173527  141330               CMPD    #-27.,AC1       ;CHECK RESULT
1204  006236  170000                       CFCC                    ;GET CC
1205  006240  001401                       BEQ     .+4
1206  006242  104010                       ERROR   10              ;-32.+5 NOT -27
1207
1208  006244  172667  173364               LDD     DALTA,AC2       ;LOAD ALT 1'S
1209  006250  172267  173370               ADDD    DALTB,AC2       ;ADD OTHER 1'S
1210  006254  170267  173204               STFPS   FPS             ;STORE FLOATING POINT STATUS
1211  006260  022767  047600  173176       CMP     #47600,FPS      ;CHECK FLOATING POINT STATUS
1212  006266  001401                       BEQ     .+4             ;BRANCH IF OK
1213  006270  104001                       ERROR   1               ;FPS NOT EQUAL TO 47600
1214
1215  006272  174277  173334               STD     AC2,@#$REGO     ;GET RESULT
1216  006276  173627  040500               CMPD    #3,AC2          ;CHECK RESULT
1217  006302  170000                       CFCC                    ;GET CC
1218  006304  001401                       BEQ     .+4
1219  006306  104010                       ERROR   10              ;$REGO NOT DALTAN
1220
1221  006310  173267  173320               SUBD    DALTA,AC2       ;SUBTRACT IT BACK
1222  006314  170267  173144               STFPS   FPS             ;STORE FLOATING POINT STATUS
1223  006320  022767  047600  173136       CMP     #47600,FPS      ;CHECK FLOATING POINT STATUS
1224  006326  001401                       BEQ     .+4             ;BRANCH IF OK
1225  006330  104001                       ERROR   1               ;FPS NOT EQUAL TO 47600
1226
1227  006332  174267  172632               STD     AC2,$REGO       ;GET THE RESULT
1228  006336  173667  173312               CMPD    DALTC,AC2       ;CHECK IT
1229  006342  170000                       CFCC                    ;GET CC
1230  006344  001401                       BEQ     .+4
1231  006346  104010                       ERROR   10              ;$REGO NOT DALTA
```

```
1232                                    ;;*****************************************************************
1233                                    ;*TEST 17       TEST OF MULF AND DIVF
1234                                    ;;*****************************************************************
1235  006350  000004                   TST17:  SCOPE
1236  006352  170127  047400                   LDFPS   #47400          ;LOAD FLOATING MODE
1237  006356  172727  040740                   LDF     #7,AC3          ;LOAD A7
1238  006362  171327  140500                   MULF    #-3,AC3         ;X -3
1239  006366  170267  173072                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1240  006372  022767  047410  173064           CMP     #47410,FPS      ;CHECK FLOATING POINT STATUS
1241  006400  001401                           BEQ     .+4             ;BRANCH IF OK
1242  006402  104001                           ERROR   1               ;FPS NOT EQUAL TO 47410
1243
1244  006404  174367  172560                   STF     AC3,$REGO       ;GET RESULT
1245  006410  173727  141250                   CMPF    #-21.,AC3       ;CHECK RESULT
1246  006414  170000                           CFCC                    ;GET CC
1247  006416  001401                           BEQ     .+4
1248  006420  104011                           ERROR   11              ;X -3 NOT -21.
1249
1250  006422  174727  140740                   DIVF    #-7,AC3         ;DIVIDE BY -3
1251  006426  170267  173032                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1252  006432  022767  047400  173024           CMP     #47400,FPS      ;CHECK FLOATING POINT STATUS
1253  006440  001401                           BEQ     .+4             ;BRANCH IF OK
1254  006442  104001                           ERROR   1               ;FPS NOT EQUAL TO 47400
1255
1256  006444  174367  172520                   STF     AC3,$REGO       ;GET RESULT
1257  006450  173727  040500                   CMPF    #3,AC3          ;CHECK RESULT
1258  006454  170000                           CFCC                    ;GET CC
1259  006456  001401                           BEQ     .+4
1260  006460  104011                           ERROR   11              ;-21. / -7 NOT 3
```

```
1261
1262                                ;;*****************************************************************
1263                                ;*TEST 20        TEST OF MULD AND DIVD
1264                                ;;*****************************************************************
1265  006462  000004               TST20:  SCOPE
1266  006464  170127  047600               LDFPS   #47600          ;LOAD DOUBLE MODE
1267  006470  172427  140640               LDD     #-5,AC0         ;LOAD A -5
1268  006474  171027  140500               MULD    #-3,AC0         ;MUL BY -3
1269  006500  170267  172760               STFPS   FPS             ;STORE FLOATING POINT STATUS
1270  006504  022767  047600  172752       CMP     #47600,FPS      ;CHECK FLOATING POINT STATUS
1271  006512  001401                       BEQ     .+4             ;BRANCH IF OK
1272  006514  104001                       ERROR   1               ;FPS NOT EQUAL TO 47600
1273
1274  006516  174067  172446               STD     AC0,$REG0       ;GET RESULT
1275  006522  173427  041160               CMPD    #15.,AC0        ;CHECK RESULT
1276  006526  170000                       CFCC                    ;GET CC
1277  006530  001401                       BEQ     .+4
1278  006532  104010                       ERROR   10              ;-5 X -3 NOT +15.
1279
1280  006534  174427  140400               DIVD    #-2,AC0         ;15. / -2
1281  006540  170267  172720               STFPS   FPS             ;STORE FLOATING POINT STATUS
1282  006544  022767  047610  172712       CMP     #47610,FPS      ;CHECK FLOATING POINT STATUS
1283  006552  001401                       BEQ     .+4             ;BRANCH IF OK
1284  006554  104001                       ERROR   1               ;FPS NOT EQUAL TO 47610
1285
1286  006556  174067  172406               STD     AC0,$REG0       ;STORE RESULT
1287  006562  173427  140760               CMPD    #-7.5,AC0       ;CHECK RESULT
1288  006566  170000                       CFCC                    ;GET CC
1289  006570  001401                       BEQ     .+4
1290  006572  104010                       ERROR   10              ;15. / -2 NOT -7.5
```

```
1290                                    ;;*****************************************************************
1291                                    ;*TEST 21       TEST OF LDCFD,LDCDF
1292                                    ;;*****************************************************************
1293  006574  000004                   TST21:  SCOPE
1294  006576  170127  047600                   LDFPS   #47600          ;SET DOUBLE MODE
1295  006602  172477  172776                   LDD     #D00101,AC0     ;LOAD A 0,-1,0,-1
1296  006606  012700  001556                   MOV     #D1010,R0       ;GET ADDRESS OF DATA
1297  006612  177420                           LDCFD   (R0)+,AC0       ;LOAD A -1,0,0,0
1298  006614  022700  001562                   CMP     #D1010+4,R0     ;INC BY 4?
1299  006620  001401                           BEQ     .+4
1300  006622  104005                           ERROR   5               ;R0 NOT #D1010+4
1301
1302  006624  170267  172634                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1303  006630  022767  047610  172626           CMP     #47610,FPS      ;CHECK FLOATING POINT STATUS
1304  006636  001401                           BEQ     .+4             ;BRANCH IF OK
1305  006640  104001                           ERROR   1               ;FPS NOT EQUAL TO 47610
1306
1307  006642  174067  172322                   STD     AC0,$REG0       ;GET ANSWER
1308  006646  173467  173054                   CMPD    D1000,AC0       ;CHECK RESULT
1309  006652  170000                           CFCC                    ;GET CC
1310  006654  001401                           BEQ     .+4
1311  006656  104010                           ERROR   10              ;AC0 NOT -1,0,0,0
1312
1313  006660  172567  172674                   LDD     D0101,AC1       ;LOAD A 0,-1,0,-1
1314  006664  170001                           SETF                    ;SET FLOATING MODE
1315  006666  012700  001566                   MOV     #D1010+10,R0    ;GET ADDRESS OF DATA +10
1316  006672  177540                           LDCDF   -(R0),AC1       ;LOAD A -1,0
1317  006674  022700  001556                   CMP     #D1010,R0       ;ADDRESS DECREMENT BY 10?
1318  006700  001401                           BEQ     .+4
1319  006702  104005                           ERROR   5               ;R0 NOT #D1010
1320
1321  006704  170267  172554                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1322  006710  022767  047410  172546           CMP     #47410,FPS      ;CHECK FLOATING POINT STATUS
1323  006716  001401                           BEQ     .+4             ;BRANCH IF OK
1324  006720  104001                           ERROR   1               ;FPS NOT EQUAL TO 47410
1325
1326  006722  170011                           SETD
1327  006724  174167  172240                   STD     AC1,$REG0       ;GET RESULT
1328  006730  173567  172640                   CMPD    WEIRD,AC1       ;CHECK RESULT
1329  006734  170000                           CFCC
1330  006736  001401                           BEQ     .+4
1331  006740  104010                           ERROR   10              ;RESULT NOT -1,1,0,-1
1332
1333  006742  170127  047440                   LDFPS   #47440          ;SET DOUBLE AND TRUNCATE MODES
1334  006746  177567  172604                   LDCDF   D1010,AC1       ;LOAD IT
1335  006752  174167  172212                   STD     AC1,$REG0       ;GET RESULT
1336  006756  173567  172604                   CMPD    D1001,AC1       ;CHECK RESULT
1337  006762  170267  172476                   STFPS   FPS             ;GET STATUS
1338  006766  170000                           CFCC                    ;GET CC
1339  006770  001401                           BEQ     .+4
1340  006772  104010                           ERROR   10              ;AC1 NOT -1,0,0,1
1341
1342  006774  170127  047400                   LDFPS   #47400          ;SET ROUND AND FLOATING MODES
1343  007000  177567  172552                   LDCDF   D1010,AC1       ;LOAD A -1.0
1344  007004  174167  172160                   STD     AC1,$REG0       ;GET THE RESULT
1345  007010  170267  172450                   STFPS   FPS             ;GET STATUS
```

```
1346  007014  022767  000001  172150          CMP     #1,$REG1        ;CHECK WORD 2 FOR A 1
1347  007022  001401                          BEQ     .+4
1348  007024  104010                          ERROR   10              ;LDCDF DID NOT ROUND PROPERLY
1349
```

```
1350                                    ;;*****************************************************************
1351                                    ;*TEST 22        TEST OF STCFD,STCDF
1352                                    ;;*****************************************************************
1353  007026  000004                    TST22:  SCOPE
1354  007030  170127  040200                    LDFPS   #40200          ;SET DOUBLE MODE
1355  007034  172667  172516                    LDD     D1010,AC2       ;LOAD A -1,0,-1,0
1356  007040  170001                            SETF                    ;SET FLOATING MODE
1357  007042  176202                            STCDF   AC2,AC2         ;CLEAR RIGHT HALF
1358  007044  170011                            SETD                    ;SET DOUBLE MODE
1359  007046  174267  172116                    STD     AC2,$REG0       ;GET RESULT
1360  007052  173667  172650                    CMPD    D1000,AC2       ;IS IT -1,0,0,0
1361  007056  170000                            CFCC                    ;GET CC
1362  007060  001401                            BEQ     .+4
1363  007062  104010                            ERROR   10              ;AC1 NOT -1,0,0,0
1364
1365  007064  172767  172654                    LDD     D0111,AC3       ;LOAD A 0,-1,0,-1
1366  007070  176303                            STCFD   AC3,AC3         ;CLEAR OUT RIGHT HALF!?
1367  007072  174367  172072                    STD     AC3,$REG0       ;GET RESULT
1368  007076  173767  172634                    CMPD    D0100X,AC3      ;CHECK RESULT
1369  007102  170000                            CFCC                    ;GET CC
1370  007104  001401                            BEQ     .+4
1371  007106  104010                            ERROR   10              ;$REG2 NOT 100,0,0 (ROUND)
1372
1373  007110  172467  172630                    LDD     D0111,AC0       ;LOAD 0,-1,0,-1
1374  007114  170127  000040                    LDFPS   #40             ;FLOATING AND TRUNCATE MODES
1375  007120  176000                            STCFD   AC0,AC0         ;CLEAR RIGHT HALF
1376  007122  170011                            SETD                    ;SET DOUBLE
1377  007124  174067  172040                    STD     AC0,$REG0       ;GET RESULT
1378  007130  173467  172570                    CMPD    D0100,AC0       ;CHECK IT
1379  007134  170000                            CFCC                    ;GET CC
1380  007136  001401                            BEQ     .+4
1381  007140  104010                            ERROR   10              ;AC0 NOT 0,-1,0,0 (TRUNCATE)
1382
```

```
1383                                    ;;*****************************************************************
1384                                    ;*TEST 23       TEST OF LDCIF,LDCID,STCFI,STCDI
1385                                    ;;*****************************************************************
1386  007142  000004                    TST23:  SCOPE
1387  007144  170127  047400                    LDFPS   #47400          ;FLOATING MODE
1388  007150  177027  000005                    LDCIF   #5,AC0          ;STORE A 5
1389  007154  170267  172304                    STFPS   FPS             ;STORE FLOATING POINT STATUS
1390  007160  022767  047400  172276            CMP     #47400,FPS      ;CHECK FLOATING POINT STATUS
1391  007166  001401                            BEQ     .+4             ;BRANCH IF OK
1392  007170  104001                            ERROR   1               ;FPS NOT EQUAL TO 47400
1393
1394  007172  174067  171772                    STF     AC0,$REG0       ;GET THE RESULT
1395  007176  173427  040640                    CMPF    #5,AC0          ;CHECK IT
1396  007202  170000                            CFCC                    ;GET CC
1397  007204  001401                            BEQ     .+4
1398  007206  104011                            ERROR   11              ;AC0 NOT 5.0
1399
1400  007210  175400                            STCFI   AC0,R0          ;CONVERT IT BACK
1401  007212  022700  000005                    CMP     #5,R0           ;CHECK RESULT
1402  007216  001401                            BEQ     .+4
1403  007220  104013                            ERROR   13              ;R0 NOT 5
1404
1405  007222  170011                            SETD                    ;SET DOUBLE MODE
1406  007224  172567  172330                    LDD     D0101,AC1       ;LOAD JUNK
1407  007230  012700  001556                    MOV     #D1010,R0       ;LOAD ADDRESS OF DATA
1408  007234  177120                            LDCID   (R0)+,AC1       ;CONVERT TO -1.0
1409  007236  022700  001560                    CMP     #D1010+2,R0     ;CHECK ADDRESS
1410  007242  001401                            BEQ     .+4
1411  007244  104005                            ERROR   5               ;R0 NOT #D1010+2
1412
1413  007246  170267  172212                    STFPS   FPS             ;STORE FLOATING POINT STATUS
1414  007252  022767  047610  172204            CMP     #47610,FPS      ;CHECK FLOATING POINT STATUS
1415  007260  001401                            BEQ     .+4             ;BRANCH IF OK
1416  007262  104001                            ERROR   1               ;FPS NOT EQUAL TO 47610
1417
1418  007264  174167  171700                    STD     AC1,$REG0       ;GET RESULT
1419  007270  173527  140200                    CMPD    #-1,AC1         ;CHECK RESULT
1420  007274  170000                            CFCC                    ;GET CC
1421  007276  001401                            BEQ     .+4
1422  007300  104010                            ERROR   10              ;AC1 NOT -1.0
1423
1424  007302  175500                            STCDI   AC1,R0          ;CONVERT IT BACK
1425  007304  022700  177777                    CMP     #-1,R0          ;CHECK RESULT
1426  007310  001401                            BEQ     .+4
1427  007312  104013                            ERROR   13              ;R0 NOT -1
1428
1429  007314  170001                            SETF                    ;SET FLOATING MODE
1430  007316  177227  054321                    LDCIF   #54321,AC2      ;LOAD 54321
1431  007322  174267  171642                    STF     AC2,$REG0       ;GET RESULT
1432  007326  170267  172132                    STFPS   FPS             ;GET STATUS
1433  007332  173667  172424                    CMPF    F5T01,AC2       ;CHECK IT
1434  007336  170000                            CFCC                    ;CHECK CC
1435  007340  001401                            BEQ     .+4
1436  007342  104011                            ERROR   11              ;AC2 NOT 54321.
1437
1438  007344  052737  000017  177776            BIS     #17,@#PS        ;SET PS
```

```
1439  007352  175667  171612          STCFI   AC2,$REGO       ;CONVERT IT BACK
1440  007356  013700  177776          MOV     @#PS,R0         ;GET PS
1441  007362  042700  177760          BIC     #177760,R0      ;CLEAR JUNK
1442  007366  001401                  BEQ     .+4             ;SKIP IF OK
1443  007370  104004                  ERROR   4               ;PS NOT 0
1444  007372  170267  172066          STFPS   FPS             ;GET STATUS
1445
1446  007376  022767  054321  171564  CMP     #54321,$REGO    ;CHECK RESULT
1447  007404  001401                  BEQ     .+4
1448  007406  104012                  ERROR   12              ;$REGO NOT 54321
1449
```

```
1450                                    ;;*****************************************************************
1451                                    ;*TEST 24        TEST OF LDCLF,LDCLD,STCFL,STCDL
1452                                    ;;*****************************************************************
1453  007410  000004                    TST24:  SCOPE
1454  007412  170127  047500                   LDFPS   #47500          ;FLOATING AND LONG MODES
1455  007416  177327  177773                   LDCLF   #-5,AC3         ;LOAD A -5
1456  007422  000401                           BR      .+4
1457  007424  104000                           ERROR   0               ;LDCLF INCREMENTED BY 4 NOT 2
1458
1459  007426  170267  172032                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1460  007432  022767  047510  172024           CMP     #47510,FPS      ;CHECK FLOATING POINT STATUS
1461  007440  001401                           BEQ     .+4             ;BRANCH IF OK
1462  007442  104001                           ERROR   1               ;FPS NOT EQUAL TO 47510
1463
1464  007444  174367  171520                   STF     AC3,$REG0       ;GET THE RESULT
1465  007450  173727                           CMPF    (7)+,AC3        ;CHECK IT
1466  007452  144640                           144640
1467  007454  170000                           CFCC                    ;GET CC
1468  007456  001401                           BEQ     .+4
1469  007460  104011                           ERROR   11              ;AC3 NOT -5
1470
1471  007462  012700  001632                   MOV     #A$REG0,R0      ;SET UP ADDRESS OF ADDRESS
1472  007466  175730                           STCFL   AC3,@(R0)+      ;STORE IN $REG0
1473  007470  022700  001634                   CMP     #A$REG0+2,R0    ;CHECK ADDRESS
1474  007474  001401                           BEQ     .+4
1475  007476  104005                           ERROR   5               ;ADDRESS IN FPU NOT A$REG0+2
1476
1477  007500  170267  171760                   STFPS   FPS             ;GET STATUS
1478  007504  022767  177773  171456           CMP     #-5,$REG0       ;CHECK LEFT HALF
1479  007512  001401                           BEQ     .+4
1480  007514  104011                           ERROR   11              ;LEFT NOT -5
1481
1482  007516  005767  171450                   TST     $REG1           ;CHECK RIGHT HALF OF RESULT
1483  007522  001401                           BEQ     .+4
1484  007524  104011                           ERROR   11              ;$REG1 NOT 0
1485
1486  007526  170011                           SETD                    ;SET DOUBLE MODE
1487  007530  177067  172220                   LDCLD   D5T01,AC0       ;LOAD WEIRD NUMBER
1488  007534  170267  171724                   STFPS   FPS             ;STORE FLOATING POINT STATUS
1489  007540  022767  047700  171716           CMP     #47700,FPS      ;CHECK FLOATING POINT STATUS
1490  007546  001401                           BEQ     .+4             ;BRANCH IF OK
1491  007550  104001                           ERROR   1               ;FPS NOT EQUAL TO 47700
1492
1493  007552  174067  171412                   STD     AC0,$REG0       ;GET RESULT
1494  007556  173467  172200                   CMPD    F5T01,AC0       ;CHECK IT
1495  007562  170000                           CFCC                    ;GET CC
1496  007564  001401                           BEQ     .+4
1497  007566  104010                           ERROR   10              ;AC0 NOT F5T01
1498
1499  007570  012700  001174                   MOV     #$REG2,R0
1500  007574  175440                           STCDL   AC0,-(R0)       ;GET IT BACK
1501  007576  022700  001170                   CMP     #$REG0,R0       ;CHECK ADDRESS
1502  007602  001401                           BEQ     .+4
1503  007604  104005                           ERROR   5               ;R0 NOT #$REG0
1504
1505  007606  170267  171652                   STFPS   FPS             ;STORE FLOATING POINT STATUS
```

```
1506  007612  022767  047700  171644          CMP     #47700,FPS      ;CHECK FLOATING POINT STATUS
1507  007620  001401                          BEQ     .+4             ;BRANCH IF OK
1508  007622  104001                          ERROR   1               ;FPS NOT EQUAL TO 47700
1509
1510  007624  170267  171634                  STFPS   FPS             ;GET STATUS
1511  007630  026767  172120  171332          CMP     DST01,$REG0     ;CHECK LEFT
1512  007636  001401                          BEQ     .+4
1513  007640  104011                          ERROR   11              ;$REG0 NOT
1514
1515  007642  026767  172110  171322          CMP     DST01+2,$REG1   ;CHECK RIGHT
1516  007650  001401                          BEQ     .+4
1517  007652  104011                          ERROR   11              ;$REG1 NOT
1518
```

```
1519                                    ;;*****************************************************************
1520                                    ;*TEST 25        TEST OF MODF AND MODD
1521                                    ;;*****************************************************************
1522  007654  0000                      TST25:  SCOPE
1523  007656  1701:  047400                    LDFPS   #47400          ;SET FLOATING MODE
1524  007662  1724:  171760                    LDF     DALTB+2,AC0     ;LOAD 52525,52525 INTO AC0
1525  007666  171427  :200                    MODF    #1,AC0          ;MODF BY 2 | #MGT#
1526  007672  170267  S                       STFPS   FPS             ;STORE FLOATING POINT STATUS
1527  007676  022767  :  :    171560         CMP     #47404,FPS      ;CHECK FLOATING POINT STATUS
1528  007704  001401                         BEQ     .+4             ;BRANCH IF OK
1529  007706  104001                         ERROR   1               ;FPS NOT EQUAL TO 47404
1530
1531  007710  174067  171254                 STF     AC0,$REG0       ;GET RESULT
1532  007714  170500                         TSTF    AC0             ;CHECK FRACTION
1533  007716  170000                         CFCC
1534  007720  001401                         BEQ     .+4
1535  007722  104011                         ERROR   11              ;FRACTION NOT 0
1536
1537  007724  174167  171240                 STF     AC1,$REG0       ;GET IT
1538  007730  173567  171712                 CMPF    DALTB+2,AC1     ;CHECK INTEGER
1539  007734  170000                         CFCC
1540  007736  001401                         BEQ     .+4
1541  007740  104011                         ERROR   11              ;INTEGER IS NOT 52525,52525
1542
1543  007742  172627  040200                 LDF     #1,AC2          ;LOAD A 1
1544  007746  171667  171742                 MODF    D20,AC2         ;MOD BY 20000,0
1545  007752  170267  171506                 STFPS   FPS             ;STORE FLOATING POINT STATUS
1546  007756  022767  047400  171500         CMP     #47400,FPS      ;CHECK FLOATING POINT STATUS
1547  007764  001401                         BEQ     .+4             ;BRANCH IF OK
1548  007766  104001                         ERROR   1               ;FPS NOT EQUAL TO 47400
1549
1550  007770  174267  171174                 S  -    AC2,$REG0       ;GET IT
1551  007774  173667  171714                 C  .F   D20,AC2         ;CHECK FRACT
1552  010000  170000                         C FCC
1553  010002  001401                         BEQ     .+4
1554  010004  104011                         ERROR   11              ;RESULT NOT 20000,0
1555
1556  010006  174367  171156                 STF     AC3,$REG0       ;GET INT
1557  010012  170503                         TSTF    AC3             ;CHECK FOR 0
1558  010014  170000                         CFCC
1559  010016  001401                         BEQ     .+4
1560  010020  104011                         ERROR   11              ;RESULT NOT 0
1561
1562  010022  170011                         SETD                    ;SET DOUBLE MODE
1563  010024  172467  171654                 LDD     D46,AC0         ;LOAD A 40600,0,0,0
1564  010030  171467  171640                 MODD    D37,AC0         ;MOD BY 37400,0,0,0
1565  010034  174067  171130                 STD     AC0,$REG0       ;STORE ANSWER
1566  010040  173467  171620                 CMPD    D40,AC0         ;CHECK FOR 40000,0,0,0
1567  010044  170000                         CFCC
1568  010046  001401                         BEQ     .+4
1569  010050  104010                         ERROR   10              ;RESULT NOT 40000,0,0,0
1570
1571  010052  174367  171112                 STD     AC3,$REG0       ;GET THE RESULT
1572  010056  170503                         TSTD    AC3             ;CHECK FOR 0
1573  010060  170000                         CFCC
1574  010062  001401                         BEQ     .+4
```

E05


```
  1575  010064  104010                        ERROR   10              ;RESULT NOT 0
  1576
```

```
1577                                        .SBTTL  ERROR CONDITIONS (STST)
1578
1579                                ;;**********************************************************************
1580                                ;*TEST 26        LDD OF -0
1581                                ;;**********************************************************************
1582  010066  000004           TST26:  SCOPE
1583  010070  170127  047600           LDFPS   #47600          ;DOUBLE MODE
1584  010074  172467  171564           LDD     D40,AC0         ;LOAD DUMMY DATA
1585  010100  172467  171516     1$:   LDD     DMZERO,AC0      ;LOAD A -0
1586  010104  170267  171354           STFPS   FPS             ;STORE FLOATING POINT STATUS
1587  010110  170367  171352           STST    FEC             ;STORE EXCEPTION CODES
1588  010114  022767  147614  171342     CMP     #147614,FPS     ;CHECK FLOATING POINT STATUS
1589  010122  001401                     BEQ     .+4             ;BRANCH IF OK
1590  010124  104001                     ERROR   1               ;FPS NOT EQUAL TO 147614
1591  010126  022767  000014  171332     CMP     #14,FEC         ;CHECK FLOATING EXCEPTION CODE
1592  010134  001401                     BEQ     .+4             ;BRANCH IF OK
1593  010136  104001                     ERROR   1               ;FEC NOT EQUAL TO 14
1594  010140  022767  010100  171322     CMP     #10100,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
1595  010146  001401                     BEQ     .+4             ;BRANCH IF OK
1596  010150  104001                     ERROR   1               ;FEA NOT EQUAL TO 10100
1597
1598  010152  174067  171012           STD     AC0,$REG0       ;GET RESULT FOR TYPING
1599  010156  173467  171502           CMPD    D40,AC0         ;DID IT CHANGE AC0?
1600  010162  170267  171276           STFPS   FPS             ;GET STATUS
1601  010166  170000                     CFCC                    ;GET CC
1602  010170  001401                     BEQ     .+4             ;SKIP IF OK
1603  010172  104010                     ERROR   10              ;RESULT IS WRONG
1604
1605  010174  170127  040200           LDFPS   #40200          ;DOUBLE MODE
1606  010200  172467  171416           LDD     DMZERO,AC0      ;LOAD A -0
1607  010204  170267  171254           STFPS   FPS             ;STORE FLOATING POINT STATUS
1608  010210  022767  040214  171246     CMP     #40214,FPS      ;CHECK FLOATING POINT STATUS
1609  010216  001401                     BEQ     .+4             ;BRANCH IF OK
1610  010220  104001                     ERROR   1               ;FPS NOT EQUAL TO 40214
1611
1612  010222  174067  170742           STD     AC0,$REG0       ;GET RESULT
1613  010226  173467  171370           CMPD    DMZERO,AC0      ;CMP TO -0
1614  010232  170267  171226           STFPS   FPS             ;GET STATUS
1615  010236  170000                     CFCC                    ;GET CC
1616  010240  001401                     BEQ     .+4             ;SKIP IF OK
1617  010242  104010                     ERROR   10              ;RESULT NOT -0
1618
1619                                ;;**********************************************************************
1620                                ;*TEST 27        MULF ERROR - OVERFLOW
1621                                ;;**********************************************************************
1622  010244  000004           TST27:  SCOPE
1623  010246  170127  001000           LDFPS   #1000           ;FLOATING/OVERFLOW
1624  010252  005001                     CLR     R1              ;CLEAR FLAG WORD
1625  010254  172427  076101           LDF     #1E36,AC0       ;LOAD A LARGE NUMBER INTO AC0
1626  010260  171027  076101     1$:   MULF    #1E36,AC0       ;MULTIPLY BY A LARGE NUMBER
1627  010264  174067  170700           STF     AC0,$REG0       ;GET FOR TYPING
1628  010270  170267  171170           STFPS   FPS             ;GET STATUS
1629  010274  005701                     TST     R1              ;DID IT TRAP?
1630  010276  001001                     BNE     3$              ;YES
1631  010300  104007                     ERROR   7               ;DID NOT TRAP ON OVERFLOW
1632  010302                     3$:
```

```
                        STFPS   FPS             ;STORE FLOATING POINT STATUS
                        STST    FEC             ;STORE EXCEPTION CODES
                        CMP     #101002,FPS     ;CHECK FLOATING POINT STATUS
                        BEQ     .+4             ;BRANCH IF OK
                        ERROR   1               ;FPS NOT EQUAL TO 101002
                        CMP     #10,FEC         ;CHECK FLOATING EXCEPTION CODE
                        BEQ     .+4             ;BRANCH IF OK
                        ERROR   1               ;FEC NOT EQUAL TO 10
                        CMP     #10260,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
                        BEQ     .+4             ;BRANCH IF OK
                        ERROR   1               ;FEA NOT EQUAL TO 10260

                ;;*****************************************************************
                ;*TEST 30       DIVF ERROR - UNDERFLOW
                ;;*****************************************************************
                TST30:  SCOPE
                        LDFPS   #2000           ;FLOATING/UNDERFLOW
                        CLR     R1              ;CLEAR FLAG WORD
                        LDF     #1E-36,AC0      ;LOAD A SMALL NUMBER
                1$:     DIVF    #1E36,AC0       ;DIVIDE BY A LARGE NUMBER
                        STF     AC0,$REG0       ;GET FOR TYPING
                        STFPS   FPS             ;GET STATUS
                        TST     R1              ;DID IT TRAP?
                        BNE     3$              ;SKIP IF SET
                        ERROR   7               ;DID NOT TRAP ON UNDERFLOW
                3$:
                        STFPS   FPS             ;STORE FLOATING POINT STATUS
                        STST    FEC             ;STORE EXCEPTION CODES
                        CMP     #102000,FPS     ;CHECK FLOATING POINT STATUS
                        BEQ     .+4             ;BRANCH IF OK
                        ERROR   1               ;FPS NOT EQUAL TO 102000
                        CMP     #12,FEC         ;CHECK FLOATING EXCEPTION CODE
                        BEQ     .+4             ;BRANCH IF OK
                        ERROR   1               ;FEC NOT EQUAL TO 12
                        CMP     #10364,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
                        BEQ     .+4             ;BRANCH IF OK
                        ERROR   1               ;FEA NOT EQUAL TO 10364

                ;;*****************************************************************
                ;*TEST 31       STCFI ERROR - CONVERSION(6)
                ;;*****************************************************************
                TST31:  SCOPE
                        LDFPS   #400            ;FLOATING/INTEGER/CONVERSION
                        CLR     R1              ;CLEAR FLAG WORD
                        LDF     #1E36,AC1       ;LOAD LARGE NUMBER
                1$:     STCFI   AC1,$REG0       ;TRY TO STUFF INTO 16 BITS
                        STFPS   FPS             ;GET STATUS
                        TST     R1              ;TRAP FLAG SET?
                        BNE     3$              ;SKIP IF SET
                        ERROR   7               ;DID NOT TRAP ON CONVERT
                3$:
                        STFPS   FPS             ;STORE FLOATING POINT STATUS
                        STST    FEC             ;STORE EXCEPTION CODES
                        CMP     #100405,FPS     ;CHECK FLOATING POINT STATUS
                        BEQ     .+4             ;BRANCH IF OK
                        ERROR   1               ;FPS NOT EQUAL TO 100405
```

```
1689  010530  022767  000006  170730          CMP     #6,FEC          ;CHECK FLOATING EXCEPTION CODE
1690  010536  001401                          BEQ     .+4             ;BRANCH IF OK
1691  010540  104001                          ERROR   1               ;FEC NOT EQUAL TO 6
1692  010542  022767  010470  170720          CMP     #10470,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
1693  010550  001401                          BEQ     .+4             ;BRANCH IF OK
1694  010552  104001                          ERROR   1               ;FEA NOT EQUAL TO 10470
1695
1696                                  ;;*****************************************************************
1697                                  ;*TEST 32        DIVF BY 0 ERROR
1698                                  ;;*****************************************************************
1699  010554  000004                  TST32:  SCOPE
1700  010556  170127  000000                  LDFPS   #0              ;FLOATING
1701  010562  005001                          CLR     R1              ;CLEAR FLAG
1702  010564  174527  000000          1$:     DIVF    #0,AC1          ;DIVIDE BY 0
1703  010570  170267  170670                  STFPS   FPS             ;GET STATUS
1704  010574  005701                          TST     R1              ;CHECK FLAG
1705  010576  001001                          BNE     3$              ;SKIP IF SET
1706  010600  104007                          ERROR   7               ;DIVIDE BY 0 DID NOT TRAP
1707  010602                          3$:
1708  010602  170267  170656                  STFPS   FPS             ;STORE FLOATING POINT STATUS
1709  010606  170367  170654                  STST    FEC             ;STORE EXCEPTION CODES
1710  010612  022767  100000  170644          CMP     #100000,FPS     ;CHECK FLOATING POINT STATUS
1711  010620  001401                          BEQ     .+4             ;BRANCH IF OK
1712  010622  104001                          ERROR   1               ;FPS NOT EQUAL TO 100000
1713  010624  022767  000004  170634          CMP     #4,FEC          ;CHECK FLOATING EXCEPTION CODE
1714  010632  001401                          BEQ     .+4             ;BRANCH IF OK
1715  010634  104001                          ERROR   1               ;FEC NOT EQUAL TO 4
1716  010636  022767  010564  170624          CMP     #10564,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
1717  010644  001401                          BEQ     .+4             ;BRANCH IF OK
1718  010646  104001                          ERROR   1               ;FEA NOT EQUAL TO 10564
1719
1720                                  ;;*****************************************************************
1721                                  ;*TEST 33        LDF -0 ERROR
1722                                  ;;*****************************************************************
1723  010650  000004                  TST33:  SCOPE
1724  010652  170127  004000                  LDFPS   #4000           ;FLOATING/UNDEFINED VARIABLE
1725  010656  005001                          CLR     R1              ;CLEAR FLAG
1726  010660  172667  170736          1$:     LDF     DMZERO,AC2      ;LOAD AN UNDEFINED VARIABLE
1727  010664  170267  170574                  STFPS   FPS             ;GET STATUS
1728  010670  005701                          TST     R1              ;CHECK FLAG
1729  010672  001001                          BNE     3$              ;SKIP IF SET
1730  010674  104007                          ERROR   7               ;LOAD OF -0 DID NOT TRAP
1731  010676                          3$:
1732  010676  170267  170562                  STFPS   FPS             ;STORE FLOATING POINT STATUS
1733  010702  170367  170560                  STST    FEC             ;STORE EXCEPTION CODES
1734  010706  022767  104014  170550          CMP     #104014,FPS     ;CHECK FLOATING POINT STATUS
1735  010714  001401                          BEQ     .+4             ;BRANCH IF OK
1736  010716  104001                          ERROR   1               ;FPS NOT EQUAL TO 104014
1737  010720  022767  000014  170540          CMP     #14,FEC         ;CHECK FLOATING EXCEPTION CODE
1738  010726  001401                          BEQ     .+4             ;BRANCH IF OK
1739  010730  104001                          ERROR   1               ;FEC NOT EQUAL TO 14
1740  010732  022767  010660  170530          CMP     #10660,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
1741  010740  001401                          BEQ     .+4             ;BRANCH IF OK
1742  010742  104001                          ERROR   1               ;FEA NOT EQUAL TO 10660
1743
1744                                  ;;*****************************************************************
```

```
                                                ;*TEST 34       OPCODE ERROR
                                                ;;*****************************************************************
                                        TST34:  SCOPE
                                                LDFPS   #0              ;FLOATING
                                                CLR     R1              ;CLEAR FLAG
                                        1$:     177707                  ;ILLEGAL OPCOPE
                                                STFPS   FPS             ;GET STATUS
                                                TST     R1              ;CHECK FLAG
                                                BNE     3$              ;SKIP IF SET
                                                ERROR   7               ;NOT AN ILLEGAL OPCODE
                                        3$:
                                                STFPS   FPS             ;STORE FLOATING POINT STATUS
                                                STST    FEC             ;STORE EXCEPTION CODES
                                                CMP     #100000,FPS     ;CHECK FLOATING POINT STATUS
                                                BEQ     .+4             ;BRANCH IF OK
                                                ERROR   1               ;FPS NOT EQUAL TO 100000
                                                CMP     #2,FEC          ;CHECK FLOATING EXCEPTION CODE
                                                BEQ     .+4             ;BRANCH IF OK
                                                ERROR   1               ;FEC NOT EQUAL TO 2
                                                CMP     #10754,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
                                                BEQ     .+4             ;BRANCH IF OK
                                                ERROR   1               ;FEA NOT EQUAL TO 10754

                                                ;;*****************************************************************
                                                ;*TEST 35       ADDF ERROR - OVERFLOW
                                                ;;*****************************************************************
                                        TST35:  SCOPE
                                                LDFPS   #1000           ;FLOATING/OVERFLOW
                                                CLR     R1              ;CLEAR FLAG
                                                LDF     DBIG,AC3        ;LOAD A BIG NUMBER
                                        1$:     ADDF    DBIG,AC3        ;MAKE OVERFLOW
                                                STFPS   FPS             ;GET STATUS
                                                STF     AC3,$REG0       ;GET RESULT
                                                TST     R1              ;FLAG SET?
                                                BNE     3$              ;SKIP IF SET
                                                ERROR   7               ;DID NOT TRAP ON OVERFLOW
                                        3$:
                                                STFPS   FPS             ;STORE FLOATING POINT STATUS
                                                STST    FEC             ;STORE EXCEPTION CODES
                                                CMP     #101006,FPS     ;CHECK FLOATING POINT STATUS
                                                BEQ     .+4             ;BRANCH IF OK
                                                ERROR   1               ;FPS NOT EQUAL TO 101006
                                                CMP     #10,FEC         ;CHECK FLOATING EXCEPTION CODE
                                                BEQ     .+4             ;BRANCH IF OK
                                                ERROR   1               ;FEC NOT EQUAL TO 10
                                                CMP     #11052,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
                                                BEQ     .+4             ;BRANCH IF OK
                                                ERROR   1               ;FEA NOT EQUAL TO 11052

                                                ;;*****************************************************************
                                                ;*TEST 36       SUBF ERROR - UNDERFLOW
                                                ;;*****************************************************************
                                        TST36:  SCOPE
                                                LDFPS   #2000           ;FLOATING/UNDERFLOW
                                                CLR     R1              ;CLEAR FLAG
                                                LDF     #.07E-37,AC0    ;LOAD SMALL NUMBER
```

```
1801  011156  173027  000504          1$:     SUBF    #.09E-37,AC0    ;SUBF SMALL NUMBER
1802  011162  174067  170002                  STF     AC0,$REG0       ;GET RESULT
1803  011166  170267  170272                  STFPS   FPS             ;GET STATUS
1804  011172  005701                          TST     R1              ;FLAG SET?
1805  011174  001001                          BNE     3$              ;SKIP IF SET
1806  011176  104007                          ERROR   7               ;NO TRAP ON UNDERFLOW
1807  011200                          3$:
1808  011200  170267  170260                  STFPS   FPS             ;STORE FLOATING POINT STATUS
1809  011204  170167  170256                  STST    FEC             ;STORE EXCEPTION CODES
1810  011210  022767  102014  170246          CMP     #102014,FPS     ;CHECK FLOATING POINT STATUS
1811  011216  001401                          BEQ     .+4             ;BRANCH IF OK
1812  011220  104001                          ERROR   1               ;FPS NOT EQUAL TO 102014
1813  011222  022767  000012  170236          CMP     #12,FEC         ;CHECK FLOATING EXCEPTION CODE
1814  011230  001401                          BEQ     .+4             ;BRANCH IF OK
1815  011232  104001                          ERROR   1               ;FEC NOT EQUAL TO 12
1816  011234  022767  011156  170226          CMP     #11156,FEA      ;CHECK FLOATING EXCEPTION ADDRESS
1817  011242  001401                          BEQ     .+4             ;BRANCH IF OK
1818  011244  104001                          ERROR   1               ;FEA NOT EQUAL TO 11156
1819
1820
1821                                  ;;*****************************************************************
1822                                  ;*TEST 37       TEST OF FEC-02, WITH NO TRAP, INTERRUPT DISABLED
1823                                  ;;*****************************************************************
1824  011246  000004                  T37:    SCOPE
1825  011250  012704  014640                  MOV     #FEC02,R4       ; PTR TO FPP FEC CODE GENERATOR
1826  011254  012705  011302                  MOV     #FPPN1,R5       ; PTR TO TEST DATA SET
1827  011260  012767  014640  170212          MOV     #FII02,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
1828  011266  012767  014642  170206          MOV     #NXT02,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
1829
1830  011274  004737  014300                  JSR     PC,@#TRPTST     ; GO TEST
1831
1832  011300                          64$:
1833  011300  000406                          BR      T37O            ;;
1834
1835  011302                          FPPN1:  ; TEST DATA SET FPPN-1:
1836  011302  000157  000040  000157          .WORD   000157,000040,000157    ; PSW'S:  BEFORE, LOADED, AFTER
1837  011310  047417  147417  100002          .WORD   047417,147417,100002    ; OLDFPS/NEWFPS/FECCODE
1838
1839
1840                                  ;;*****************************************************************
1841                                  ;*TEST 40       TEST OF FEC-04, WITH NO TRAP, INTERRUPT DISABLED
1842                                  ;;*****************************************************************
1843  011316  000004                  T37O:   SCOPE
1844  011320  012704  014650                  MOV     #FEC04,R4       ; PTR TO FPP FEC CODE GENERATOR
1845  011324  012705  011352                  MOV     #FPPN2,R5       ; PTR TO TEST DATA SET
1846  011330  012767  014654  170142          MOV     #FII04,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
1847  011336  012767  014660  170136          MOV     #NXT04,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
1848
1849  011344  004737  014300                  JSR     PC,@#TRPTST     ; GO TEST
1850
1851  011350                          64$:
1852  011350  000406                          BR      T41             ;;
1853
1854  011352                          FPPN2:  ; TEST DATA SET FPPN-2:
1855  011352  000157  000040  000157          .WORD   000157,000040,000157    ; PSW'S:  BEFORE, LOADED, AFTER
1856  011360  047417  147400  100004          .WORD   047417,147400,100004    ; OLDFPS/NEWFPS/FECCODE
```

```
                                        ;;*****************************************************************
                                        ;*TEST 41        TEST OF FEC-06, WITH NO TRAP, INTERRUPT DISABLED
                                        ;;*****************************************************************
011366  000004                  TST41:  SCOPE
011370  012704  014666                  MOV     #FEC06,R4       ; PTR TO FPP FEC CODE GENERATOR
011374  012705  011422                  MOV     #FPPN3,R5       ; PTR TO TEST DATA SET
011400  012767  014672  170072          MOV     #FII06,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
011406  012767  014676  170066          MOV     #NXT06,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED

011414  004737  014300                  JSR     PC,@#TRPTST     ; GO TEST

011420                          64$:
011420  000406                        BR      TST42           ;;

011422                  FPPN3:  ; TEST DATA SET FPPN-3:
011422  000155  000042  000142          .WORD   000155,000042,000142    ; PSW'S:  BEFORE, LOADED, AFTER
011430  047412  147405  100006          .WORD   047412,147405,100006    ; OLDFPS/NEWFPS/FECCODE

                                        ;;*****************************************************************
                                        ;*TEST 42        TEST OF FEC-10, WITH NO TRAP, INTERRUPT DISABLED
                                        ;;*****************************************************************
011436  000004                  TST42:  SCOPE
011440  012704  014704                  MOV     #FEC10,R4       ; PTR TO FPP FEC CODE GENERATOR
011444  012705  011472                  MOV     #FPPN4,R5       ; PTR TO TEST DATA SET
011450  012767  014714  170022          MOV     #FII10,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
011456  012767  014716  170016          MOV     #NXT10,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED

011464  004737  014300                  JSR     PC,@#TRPTST     ; GO TEST

011470                          64$:
011470  000406                        BR      TST43           ;;

011472                  FPPN4:  ; TEST DATA SET FPPN-4:
011472  000151  000046  000151          .WORD   000151,000046,000151    ; PSW'S:  BEFORE, LOADED, AFTER
011500  047411  147406  100010          .WORD   047411,147406,100010    ; OLDFPS/NEWFPS/FECCODE

                                        ;;*****************************************************************
                                        ;*TEST 43        TEST OF FEC-12, WITH NO TRAP, INTERRUPT DISABLED
                                        ;;*****************************************************************
011506  000004                  TST43:  SCOPE
011510  012704  014724                  MOV     #FEC12,R4       ; PTR TO FPP FEC CODE GENERATOR
011514  012705  011542                  MOV     #FPPN5,R5       ; PTR TO TEST DATA SET
011520  012767  014730  167752          MOV     #FII12,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
011526  012767  014734  167746          MOV     #NXT12,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED

011534  004737  014300                  JSR     PC,@#TRPTST     ; GO TEST

011540                          64$:
011540  000406                        BR      TST44           ;;

011542                  FPPN5:  ; TEST DATA SET FPPN-5:
011542  000157  000040  000157          .WORD   000157,000040,000157    ; PSW'S:  BEFORE, LOADED, AFTER
```

```
                                .WORD   047417,147400,100012    ; OLDFPS/NEWFPS/FECCODE

                        ;;**********************************************************************
                        ;*TEST 44        TEST OF FEC-14, WITH NO TRAP, INTERRUPT DISABLED
                        ;;**********************************************************************
011556  000004          T$T44:  SCOPE
011560  012704  014742          MOV     #FEC14,R4       ; PTR TO FPP FEC CODE GENERATOR
011564  012705  011612          MOV     #FPPN6,R5       ; PTR TO TEST DATA SET
011570  012767  014742  167702  MOV     #FII14,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
011576  012767  014746  167676  MOV     #NXT14,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED

011604  004737  014300          JSR     PC,@#TRPTST     ; GO TEST

011610                  64$:
011610  000406                  BR      TST45           ;;

011612                  FPPN6:  ; TEST DATA SET FPPN-6:
011612  000143  000054  000143          .WORD   000143,000054,000143    ; PSW'S:  BEFORE, LOADED, AFTER
011620  047403  147414  100014          .WORD   047403,147414,100014    ; OLDFPS/NEWFPS/FECCODE

                        ;;**********************************************************************
                        ;*TEST 45        TEST OF FEC-16, WITH NO TRAP, INTERRUPT DISABLED
                        ;;**********************************************************************
011626  000004          T$T45:  SCOPE

                        ;//////////////////////////////////////////////////////////////////////
                                ;THIS TEST IS EXECUTED FOR HFP ONLY, NOT IN WFP MODE

011630  076600  000022          MFD     #WHAMI          ;GET WHAMI INTO R0
011634  032700  000020          BIT     #BIT04,R0       ;HFP IN SYSTEM ?
011640  001421                  BEQ     64$             ;BR IF NONE

011642  076600  000144          MFD     #FLAG           ;GET FLAGS INTO R0
011646  032700  010000          BIT     #BIT12,R0       ;HFP ENABLED ?
011652  001414                  BEQ     64$             ;BR IF NOT

                        ;//////////////////////////////////////////////////////////////////////

011654  012704  014754          MOV     #FEC16,R4       ; PTR TO FPP FEC CODE GENERATOR
011660  012705  011706          MOV     #FPPN7,R5       ; PTR TO TEST DATA SET
011664  012767  014762  167606  MOV     #FII16,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
011672  012767  014764  167602  MOV     #NXT16,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED

011700  004737  014300          JSR     PC,@#TRPTST     ; GO TEST

011704                  64$:
011704  000406                  BR      TST46           ;;

011706                  FPPN7:  ; TEST DATA SET FPPN-7:
011706  000157  000052  000140          .WORD   000157,000052,000140    ; PSW'S:  BEFORE, LOADED, AFTER
011714  047420  147420  100016          .WORD   047420,147420,100016    ; OLDFPS/NEWFPS/FECCODE
```

```
                                        ;////////////////////////////////////////////////////////////////
                                        ;       NOTE:  IN THE FOLLOWING GROUP OF TESTS, THE LOADED PS
                                        ;              MUST CONTAIN PROCESSOR PRIORITY OF (6) OR GREATER
                                        ;              TO LOCK OUT ANY CLOCK INTERRUPTS (AT BR6).  THEY
                                        ;              ARE SETUP HERE WITH PRIORITY (7).
                                        ;////////////////////////////////////////////////////////////////
                                        ;;****************************************************************
                                        ;*TEST 46       TEST OF FEC-02, WITH TRAP, INTERRUPT ENABLED
                                        ;;****************************************************************
011722  000004                  TST46:  SCOPE
011724  012704  014640                  MOV     #FEC02,R4       ; PTR TO FPP FEC CODE GENERATOR
011730  012705  011756                  MOV     #FPPT10,R5      ; PTR TO TEST DATA SET
011734  012767  014640  167536          MOV     #FII02,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
011742  012767  014642  167532          MOV     #NXT02,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED

011750  004737  014300                  JSR     PC,@#TRPTST     ; GO TEST

011754                          64$:
011754  000406                        BR      TST47           ;;

011756                          FPPT10: ; TEST DATA SET FPPT-10:
011756  000353  000104  000353          .WORD   000353,000104,000353    ; PSW'S:  BEFORE, LOADED, AFTER
011764  000017  100017  140002          .WORD   000017,100017,140002    ; OLDFPS/NEWFPS/FECCODE

                                        ;;****************************************************************
                                        ;*TEST 47       TEST OF FEC-04, WITH TRAP, INTERRUPT ENABLED
                                        ;;****************************************************************
011772  000004                  TST47:  SCOPE
011774  012704  014650                  MOV     #FEC04,R4       ; PTR TO FPP FEC CODE GENERATOR
012000  012705  012026                  MOV     #FPPT11,R5      ; PTR TO TEST DATA SET
012004  012767  014654  167466          MOV     #FII04,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
012012  012767  014660  167462          MOV     #NXT04,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED

012020  004737  014300                  JSR     PC,@#TRPTST     ; GO TEST

012024                          64$:
012024  000406                        BR      TST50           ;;

012026                          FPPT11: ; TEST DATA SET FPPT-11:
012026  000350  000107  000350          .WORD   000350,000107,000350    ; PSW'S:  BEFORE, LOADED, AFTER
012034  000017  100000  140004          .WORD   000017,100000,140004    ; OLDFPS/NEWFPS/FECCODE


                                        ;;****************************************************************
                                        ;*TEST 50       TEST OF FEC-06, WITH NO TRAP, INTERRUPT ENABLED
                                        ;;****************************************************************
012042  000004                  TST50:  SCOPE
012044  012704  014666                  MOV     #FEC06,R4       ; PTR TO FPP FEC CODE GENERATOR
012050  012705  012076                  MOV     #FPPN12,R5      ; PTR TO TEST DATA SET
012054  012767  014672  167416          MOV     #FII06,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
012062  012767  014676  167412          MOV     #NXT06,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED

```

```
                                TST0    TEST OF FEC-06, WITH NO TRAP, INTERRUPT ENABLED

2025  012070  004737  014300                JSR     PC,@#TRPTST     ; GO TEST
2026
2027  012074                        64$:
2028  012074  000406                        BR      TST51           ;;
2029
2030  012076                        FPPN12: ; TEST DATA SET FPPN-12:
2031  012076  000353  000104  000345        .WORD   000353,000104,000345    ; PSW'S:  BEFORE, LOADED, AFTER
2032  012104  007012  007005  000000        .WORD   007012,007005,000000    ; OLDFPS/NEWFPS/FECCODE
2033
2034
2035                                ;;*****************************************************************
2036                                ;*TEST 51       TEST OF FEC-06, WITH TRAP, INTERRUPT ENABLED
2037                                ;;*****************************************************************
2038  012112  000004                TST51:  SCOPE
2039  012114  012704  014666                MOV     #FEC06,R4       ; PTR TO FPP FEC CODE GENERATOR
2040  012120  012705  012146                MOV     #FPPT13,R5      ; PTR TO TEST DATA SET
2041  012124  012767  014672  167346        MOV     #FII06,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2042  012132  012767  014676  167342        MOV     #NXT06,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2043
2044  012140  004737  014300                JSR     PC,@#TRPTST     ; GO TEST
2045
2046  012144                        64$:
2047  012144  000406                        BR      TST52           ;;
2048
2049  012146                        FPPT13: ; TEST DATA SET FPPT-13:
2050  012146  000353  000104  000345        .WORD   000353,000104,000345    ; PSW'S:  BEFORE, LOADED, AFTER
2051  012154  000412  100405  140006        .WORD   000412,100405,140006    ; OLDFPS/NEWFPS/FECCODE
2052
2053
2054                                ;;*****************************************************************
2055                                ;*TEST 52       TEST OF FEC-10, WITH NO TRAP, INTERRUPT ENABLED
2056                                ;;*****************************************************************
2057  012162  000004                TST52:  SCOPE
2058  012164  012704  014704                MOV     #FEC10,R4       ; PTR TO FPP FEC CODE GENERATOR
2059  012170  012705  012216                MOV     #FPPN14,R5      ; PTR TO TEST DATA SET
2060  012174  012767  014714  167276        MOV     #FII10,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2061  012202  012767  014716  167272        MOV     #NXT10,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2062
2063  012210  004737  014300                JSR     PC,@#TRPTST     ; GO TEST
2064
2065  012214                        64$:
2066  012214  000406                        BR      TST53           ;;
2067
2068  012216                        FPPN14: ; TEST DATA SET FPPN-14:
2069  012216  000350  000107  000350        .WORD   000350,000107,000350    ; PSW'S:  BEFORE, LOADED, AFTER
2070  012224  006411  006406  000000        .WORD   006411,006406,000000    ; OLDFPS/NEWFPS/FECCODE
2071
2072
2073                                ;;*****************************************************************
2074                                ;*TEST 53       TEST OF FEC-10, WITH TRAP, INTERRUPT ENABLED
2075                                ;;*****************************************************************
2076  012232  000004                TST53:  SCOPE
2077  012234  012704  014704                MOV     #FEC10,R4       ; PTR TO FPP FEC CODE GENERATOR
2078  012240  012705  012266                MOV     #FPPT15,R5      ; PTR TO TEST DATA SET
2079  012244  012767  014714  167226        MOV     #FII10,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2080  012252  012767  014716  167222        MOV     #NXT10,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
```

```
2081
2082  012260  004737  014300                JSR     PC,@#TRPTST     ; GO TEST
2083
2084  012264                        64$:
2085  012264  000406                BR      TST54           ;;
2086
2087  012266                        FPPT15: ; TEST DATA SET FPPT-15:
2088  012266  000350  000107  000350        .WORD   000350,000107,000350    ; PSW'S:  BEFORE, LOADED, AFTER
2089  012274  001011  101006  140010        .WORD   001011,101006,140010    ; OLDFPS/NEWFPS/FECCODE
2090
2091
2092                                ;;*****************************************************************
2093                                ;*TEST 54       TEST OF FEC-12, WITH NO TRAP, INTERRUPT ENABLED
2094                                ;;*****************************************************************
2095  012302  000004                TST54:  SCOPE
2096  012304  012704  014724                MOV     #FEC12,R4       ; PTR TO FPP FEC CODE GENERATOR
2097  012310  012705  012336                MOV     #FPPN16,R5      ; PTR TO TEST DATA SET
2098  012314  012767  014730  167156        MOV     #FII12,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2099  012322  012767  014734  167152        MOV     #NXT12,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2100
2101  012330  004737  014300                JSR     PC,@#TRPTST     ; GO TEST
2102
2103  012334                        64$:
2104  012334  000406                BR      TST55           ;;
2105
2106  012336                        FPPN16: ; TEST DATA SET FPPN-16:
2107  012336  000350  000107  000350        .WORD   000350,000107,000350    ; PSW'S:  BEFORE, LOADED, AFTER
2108  012344  005413  005404  000000        .WORD   005413,005404,000000    ; OLDFPS/NEWFPS/FECCODE
2109
2110
2111                                ;;*****************************************************************
2112                                ;*TEST 55       TEST OF FEC-12, WITH TRAP, INTERRUPT ENABLED
2113                                ;;*****************************************************************
2114  012352  000004                TST55:  SCOPE
2115  012354  012704  014724                MOV     #FEC12,R4       ; PTR TO FPP FEC CODE GENERATOR
2116  012360  012705  012406                MOV     #FPPT17,R5      ; PTR TO TEST DATA SET
2117  012364  012767  014730  167106        MOV     #FII12,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2118  012372  012767  014734  167102        MOV     #NXT12,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2119
2120  012400  004737  014300                JSR     PC,@#TRPTST     ; GO TEST
2121
2122  012404                        64$:
2123  012404  000406                BR      TST56           ;;
2124
2125  012406                        FPPT17: ; TEST DATA SET FPPT-17:
2126  012406  000350  000107  000350        .WORD   000350,000107,000350    ; PSW'S:  BEFORE, LOADED, AFTER
2127  012414  002017  102000  140012        .WORD   002017,102000,140012    ; OLDFPS/NEWFPS/FECCODE
2128
2129
2130                                ;;*****************************************************************
2131                                ;*TEST 56       TEST OF FEC-14, WITH NO TRAP, INTERRUPT ENABLED
2132                                ;;*****************************************************************
2133  012422  000004                TST56:  SCOPE
2134  012424  012704  014742                MOV     #FEC14,R4       ; PTR TO FPP FEC CODE GENERATOR
2135  012430  012705  012456                MOV     #FPPN20,R5      ; PTR TO TEST DATA SET
2136  012434  012767  014742  167036        MOV     #FII14,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
```

```
2137  012442  012767  014746  167032          MOV     #NXT14,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2138
2139  012450  004737  014300                  JSR     PC,@#TRPTST     ; GO TEST
2140
2141  012454                          64$:
2142  012454  000406                          BR      TST57           ;;
2143
2144  012456                          FPPN20: ; TEST DATA SET FPPN-20:
2145  012456  000343  000114  000343          .WORD   000343,000114,000343    ; PSW'S:  BEFORE, LOADED, AFTER
2146  012464  003403  003414  000000          .WORD   003403,003414,000000    ; OLDFPS/NEWFPS/FECCODE
2147
2148
2149                                  ;;*****************************************************************
2150                                  ;*TEST 57       TEST OF FEC-14, WITH TRAP, INTERRUPT ENABLED
2151                                  ;;*****************************************************************
2152  012472  000004                  TST57:  SCOPE
2153  012474  012704  014742                  MOV     #FEC14,R4       ; PTR TO FPP FEC CODE GENERATOR
2154  012500  012705  012526                  MOV     #FPPT21,R5      ; PTR TO TEST DATA SET
2155  012504  012767  014742  166766          MOV     #FII14,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2156  012512  012767  014746  166762          MOV     #NXT14,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2157
2158  012520  004737  014300                  JSR     PC,@#TRPTST     ; GO TEST
2159
2160  012524                          64$:
2161  012524  000406                          BR      TST60           ;;
2162
2163  012526                          FPPT21: ; TEST DATA SET FPPT-21:
2164  012526  000343  000114  000343          .WORD   000343,000114,000343    ; PSW'S:  BEFORE, LOADED, AFTER
2165  012534  004003  104014  140014          .WORD   004003,104014,140014    ; OLDFPS/NEWFPS/FECCODE
2166
2167
2168                                  ;;*****************************************************************
2169                                  ;*TEST 60       TEST OF FEC-16, WITH NO TRAP, INTERRUPT ENABLED
2170                                  ;;*****************************************************************
2171  012542  000004                  TST60:  SCOPE
2172  012544  012704  014754                  MOV     #FEC16,R4       ; PTR TO FPP FEC CODE GENERATOR
2173  012550  012705  012576                  MOV     #FPPN22,R5      ; PTR TO TEST DATA SET
2174  012554  012767  014762  166716          MOV     #FII16,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2175  012562  012767  014764  166712          MOV     #NXT16,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2176
2177  012570  004737  014300                  JSR     PC,@#TRPTST     ; GO TEST
2178
2179  012574                          64$:
2180  012574  000406                          BR      TST61           ;;
2181
2182  012576                          FPPN22: ; TEST DATA SET FPPN-22:
2183  012576  000357  000145  000340          .WORD   000357,000145,000340    ; PSW'S:  BEFORE, LOADED, AFTER
2184  012604  007400  007400  000000          .WORD   007400,007400,000000    ; OLDFPS/NEWFPS/FECCODE
2185
2186
2187                                  ;;*****************************************************************
2188                                  ;*TEST 61       TEST OF FEC-16, WITH TRAP, INTERRUPT ENABLED
2189                                  ;;*****************************************************************
2190  012612  000004                  TST61:  SCOPE
2191
2192                                  ;////////////////////////////////////////////////////////////////
```

```
2193                                                ;THIS TEST IS EXECUTED FOR HFP ONLY, NOT IN WFP MODE
2194
2195  012614  076600  000022                        MFD     ,RWHAMI         ;GET WHAMI INTO R0
2196  012620  032700  000020                        BIT     #BIT04,R0       ;HFP IN SYSTEM ?
2197  012624  001421                                BEQ     64$             ;BR IF NONE
2198
2199  012626  076600  000144                        MFD     ,RFLAG          ;GET FLAGS INTO R0
2200  012632  032700  010000                        BIT     #BIT12,R0       ;HFP ENABLED ?
2201  012636  001414                                BEQ     64$             ;BR IF NOT
2202
2203                                              ;//////////////////////////////////////////////////////////////
2204
2205  012640  012704  014754                        MOV     #FEC16,R4       ; PTR TO FPP FEC CODE GENERATOR
2206  012644  012705  012672                        MOV     #FPPT23,R5      ; PTR TO TEST DATA SET
2207  012650  012767  014762  166622                MOV     #FII16,EXPFEA   ; ADDR(FPP BAD INSTR) EXPECTED
2208  012656  012767  014764  166616                MOV     #NXT16,EXPRTS   ; ADDR(NEXT INSTR) EXPECTED
2209
2210  012664  004737  014300                        JSR     PC,@#TRPTST     ; GO TEST
2211
2212  012670                                    64$:
2213  012670  000406                                BR      TST62           ;;
2214
2215  012672                                    FPPT23: ; TEST DATA SET FPPT-23:
2216  012672  000357  000152  000341                .WORD   000357,000152,000341    ; PSW'S:  BEFORE, LOADED, AFTER
2217  012700  000020  100020  140016                .WORD   000020,100020,140016    ; OLDFPS/NEWFPS/FECCODE
2218
2219
```

```
                                ;;*****************************************************************
                                ;*TEST 62        TEST FOR CONVERSION, ADDO AND SUBO
                                ;;*****************************************************************
                                TST62:  SCOPE
                                        LDFPS   #47600
                                        LDCID   #4,AC3          ;LOAD AC3 WITH A FLOATING 4
                                        LDCID   #2,AC2          ;LOAD AC2 WITH A FLOATING 2
                                        ADDO    AC2,AC3         ;ADD 2+4 = 6 IN AC3
                                        SUBO    AC2,AC3         ;SUB 2 FROM 6
                                        STCDI   AC3,@#$REGO     ;@#$REGO SHOULD =4
                                        CMP     #4,@#$REGO      ;DOES @#$REGO =4
                                        STFPS   FPS             ;GET STATUS
                                        BEQ     .+4             ;YES
                                        ERROR   12              ;@#$REGO SHOULD = 4

                                ;;*****************************************************************
                                ;*TEST 63        LDD AND STD TEST
                                ;;*****************************************************************
                                TST63:  SCOPE
                                        LDCID   #5,AC0          ;LOAD AC0 WITH A FLOATING 5
                                        STD     AC0,AC5         ;NOW PUT IT INTO AC5
                                        LDD     AC5,AC2         ;NOW PUT IT INTO AC2
                                        ADDO    AC0,AC2         ;ADD 5 TO 5
                                        SUBO    AC5,AC2         ;SUB 5 FROM 10
                                        STCDI   AC2,@#$REGO     ;PUT ANS INTO @#$REGO
                                        CMP     #5,@#$REGO      ;WERE THE TWO AC'S EQUAL
                                        STFPS   FPS             ;GET STATUS
                                        BEQ     .+4             ;YES
                                        ERROR   12              ;$REGO SHOULD = 5
```

```
                                ;;*****************************************************************
                                ;*TEST 64        MULD AND DIVD TEST
                                ;;*****************************************************************
013012  000004                  TST64:  SCOPE
013014  012702  001200                  MOV     #$REG4,R2       ;ADDR(DATA)
013020  177227  000052                  LDCID   #52,AC2         ;LOAD AC2 WITH FLOATING DOUBLE 52
013024  174267  166150                  STD     AC2,$REG4       ;PUT IT INTO $REG4
013030  177327  000025                  LDCID   #25,AC3         ;LOAD AC3 WITH FL DB 25
013034  171322                          MULD    (R2)+,AC3       ;MUL 52X25 RESULT IN AC3
013036  000240                          NOP                     ;DELAY SOME
013040  174742                          DIVD    -(R2),AC3       ;DIV 52 INTO RESULT IN AC3
013042  171322                          MULD    (R2)+,AC3       ;MUL 52X25
013044  000240                          NOP                     ;DELAY SOME
013046  173342                          ADDD    -(R2),AC3       ;ADD AC4 TO AC3 TO MAKE 53 TIMES
013050  012704  000025                  MOV     #25,R4          ;SET UP COUNTER
013054  173367  166120          SUBT:   SUBD    $REG4,AC3       ;SUB 52 FROM AC3 25 TIMES
013060  005304                          DEC     R4
013062  001374                          BNE     SUBT            ;SUB 53 TIMES
013064  175737  001170                  STCDI   AC3,@#$REGO     ;ANS SHOULD BE 52
013070  170267  166370                  STFPS   FPS             ;GET STATUS
013074  022737  000052  001170          CMP     #52,@#$REGO     ;IS ANS CORRECT?
013102  001401                          BEQ     .+4             ;YES
013104  104012                          ERROR   12              ;$REGO SHOULD BE 52
```

```
2273                                    ;;*****************************************************************
2274                                    ;*TEST 65        EXERCISER TEST
2275                                    ;;*****************************************************************
2276  013106  000004                    TST65:  SCOPE
2277  013110  177027  025252                    LDCID   #25252,AC0      ;LOAD 25252 INTO AC0
2278  013114  177127  000025                    LDCID   #25,AC1         ;LOAD AC1 WITH 25
2279  013120  171100                            MULD    AC0,AC1         ;MUL 25252X25 ANS IN AC1
2280  013122  174137  001170                    STD     AC1,@#$REG0     ;SAV ANS IN @#$REG0
2281  013126  012702  000024                    MOV     #24,R2          ;SET UP COUNT
2282  013132  174005                            STD     AC0,AC5         ;PUT 25252 INTO AC5
2283  013134  172005                    AAD:    ADDD    AC5,AC0         ;ADD 25252 TO 25252
2284  013136  005302                            DEC     R2              ;DO 25 TIMES
2285  013140  001375                            BNE     AAD             ;DONE?
2286  013142  174037  001200                    STD     AC0,@#$REG4     ;LOAD TO PRINT
2287  013146  170267  166312                    STFPS   FPS             ;GET STATUS
2288  013152  173437  001170                    CMPD    @#$REG0,AC0     ;IS ANS CORRECT?
2289  013156  170000                            CFCC
2290  013160  001401                            BEQ     .+4             ;YES
2291  013162  104010                            ERROR   10              ;EITHER THE ADD OR THE MUL DID NOT WORK
2292  013164  177327  000002                    LDCID   #2,AC3          ;LOAD AC3 WITH A 2
2293  013170  012702  000013                    MOV     #13,R2          ;SET UP COUNTER
2294  013174  174304                            STD     AC3,AC4         ;LOAD AC4 WITH A 2
2295  013176  171304                    MMUL:   MULD    AC4,AC3         ;MUL 2 X 2
2296  013200  005302                            DEC     R2              ;DO 16 TIMES
2297  013202  001375                            BNE     MMUL
2298  013204  175703                            STCDI   AC3,R3          ;PUT ANS IN R3
2299  013206  010337  001170                    MOV     R3,@#$REG0      ;NOW PUT IT INTO @#$REG0
2300  013212  022703  010000                    CMP     #10000,R3       ;ANS SHOULD BE 10000
2301  013216  170267  166242                    STFPS   FPS             ;GET STATUS
2302  013222  001401                            BEQ     .+4             ;CONT. IF ANS IS CORRECT
2303  013224  104012                            ERROR   12              ;ANS SHOULD BE 10000
2304  013226  177227  010000                    LDCID   #10000,AC2      ;LOAD AC2 WITH 10000
2305  013232  177127  000002                    LDCID   #2,AC1          ;LOAD AC1 WITH A 2
2306  013236  012702  000013                    MOV     #13,R2          ;SET UP COUNTER
2307  013242  174601                    DDIV:   DIVD    AC1,AC2         ;DIVD 2 INTO 65536 16 TIMES
2308  013244  005302                            DEC     R2              ;COUNT NO OF TIMES
2309  013246  001375                            BNE     DDIV            ;ARE WE DONE?
2310  013250  175603                            STCDI   AC2,R3          ;STOR ANS INTO R3
2311  013252  010237  001170                    MOV     R2,@#$REG0      ;PUT IT INTO @#$REG0 FOR TYPING
2312  013256  022703  000002                    CMP     #2,R3           ;IS ANS CORRECT
2313  013262  170267  166176                    STFPS   FPS             ;GET STATUS
2314  013266  001401                            BEQ     .+4             ;ANS IS CORRECT
2315  013270  104012                            ERROR   12              ;$REG0 SHOULD EQUAL =2
```

```
2316
2317                                    ;;*****************************************************************
2318                                    ;*TEST 66        MODE ONE TEST
2319  013272  000004                   ;;*****************************************************************
2320  013274  170127  047400           TST66:  SCOPE
2321  013300  177327  000005                   LDFPS   #47400
2322  013304  012702  001170                   LDCIF   #5,AC3          ;AC3=5
2323  013310  177227  000007                   MOV     #$REGO,R2       ;R2=$REGO
2324  013314  174322                           LDCIF   #7,AC2          ;LOAD 7 INTO AC2
2325  013316  173737  001170                   STF     AC3,(R2)+       ;$REGO SHOULD =5=R2
2326  013322  170267  166136                   CMPF    @#$REGO,AC3     ;DOES @#$REGO = 5
2327  013326  170000                           STFPS   FPS             ;GET STATUS
2328  013330  001401                           CFCC
2329  013332  104011                           BEQ     .+4             ;YES BRANCH
2330  013334  171242                           ERROR   11              ;$REGO SHOULD CONTAIN 5
2331  013336  012703  000006                   MULF    -(R2),AC2       ;MUL 5 X 7,AC2 = 35
2332  013342  173212                           MOV     #6,R3           ;SET UP COUNTER
2333  013344  005303                   SUBFM:  SUBF    (R2),AC2        ;SUB 5 FROM 35
2334  013346  001375                           DEC     R3              ;DO 7 TIMES
2335  013350  175637  001170                   BNE     SUBFM
2336  013354  022737  000005  001170           STCFI   AC2,@#$REGO     ;@#$REGO SHOULD =5
2337  013362  170267  166076                   CMP     #5,@#$REGO      ;DOES @#$REGO =5
2338  013366  001401                           STFPS   FPS             ;GET STATUS
2339  013370  104012                           BEQ     .+4             ;BRANCH IF YES
2340                                          ERROR   12              ;ANS SHOULD =5
2341
2342                                    ;;*****************************************************************
2343                                    ;*TEST 67        ADDD EXERCISER
2344  013372  000004                   ;;*****************************************************************
2345  013374  170127  047600           TST67:  SCOPE
2346  013400  177027  000252                   LDFPS   #47600
2347  013404  177227  052525                   LDCID   #252,AC0        ;LOAD AC0 WITH 252
2348  013410  174204                           LDCID   #52525,AC2      ;LOAD AC2 WITH 52525
2349  013412  012702  000251                   STD     AC2,AC4         ;AC4=52525
2350  013416  171002                           MOV     #251,R2         ;SET UP COUNTER
2351  013420  000240                           MULD    AC2,AC0         ;AC0=52525 X 252
2352  013422  172204                           NOP                     ;DELAY SOME
2353  013424  005302                   AADDM:  ADDD    AC4,AC2         ;ADD 52525 TO 252
2354  013426  001375                           DEC     R2              ;DO 252 TIMES
2355  013430  174237  001170                   BNE     AADDM           ;DONE?
2356  013434  170267  166024                   STD     AC2,@#$REGO     ;GET FOR PRINTING
2357  013440  173402                           STFPS   FPS             ;GET STATUS
2358  013442  170000                           CMPD    AC2,AC0         ;DOES AC2 = AC0?
2359  013444  001401                           CFCC
2360  013446  104010                           BEQ     .+4             ;BRANCH IF EQUAL
                                               ERROR   10              ;ANS SHOULD BE.....
```

```
                                ;;*****************************************************************
                                ;*TEST 70        TEST DIVIDE BY 0
                                ;*****************************************************************
013450  000004                  TST70:  SCOPE
013452  170127  047600                  LDFPS   #47600
013456  177027  077777                  LDCID   #77777,AC0
013462  177127  000000                  LDCID   #0,AC1
013466  174401                          DIVD    AC1,AC0         ;DIVIDE 0 INTO 77777
013470  175437  001170                  STCDI   AC0,@#$REG0     ;LOAD @#$REG0 WITH 77777
013474  170267  165764                  STFPS   FPS             ;GET STATUS
013500  170337  001466                  STST    @#FEC           ;GET @#FEC STATUS
013504  022737  077777  001170          CMP     #77777,@#$REG0
013512  001401                          BEQ     .+4
013514  104012                          ERROR   12              ;ANS SHOULD =77777
013516  022737  000004  001466          CMP     #4,@#FEC        ;DID WE TRY TO DIV BY 0?
013524  001401                          BEQ     .+4             ;YES
013526  104002                          ERROR   2               ;FEC SHOULD =4

                                ;;*****************************************************************
                                ;*TEST 71        EXERCISER FOR ADDD,SUBD,MULD AND DIVD
                                ;*****************************************************************
013530  000004                  TST71:  SCOPE
013532  170127  047400                  LDFPS   #47400
013536  170011                          SETD                    ;SET DOUBLE MODE
013540  177027  077777                  LDCID   #77777,AC0      ;LOAD AC0 WITH 77777
013544  177127  002525                  LDCID   #2525,AC1       ;LOAD AC1 WITH 2525
013550  012702  000012                  MOV     #12,R2          ;SET UP COUNTER
013554  174401                  EXLOP:  DIVD    AC1,AC0         ;DIVDE 2525 INTO 77777
013556  171001                          MULD    AC1,AC0         ;MUL 2525 X ANS
013560  172001                          ADDD    AC1,AC0         ;ADD 2525 TO ANS
013562  173001                          SUBD    AC1,AC0         ;SUB 2525 FROM ANS
013564  005302                          DEC     R2              ;DO 12 TIMES
013566  001372                          BNE     EXLOP           ;DONE?
013570  175437  001170                  STCDI   AC0,@#$REG0     ;LOAD ANSWER INTO @#$REG0
013574  170267  165664                  STFPS   FPS             ;GET STATUS
013600  022737  077777  001170          CMP     #77777,@#$REG0  ;IS @#$REG0 CORRECT
013606  001401                          BEQ     .+4             ;BRANCH IF CORRECT
013610  104012                          ERROR   12
013612  172437  001556          EX4:    LDD     @#D1010,AC0     ;GET DATA
013616  012702  001170                  MOV     #$REG0,R2
013622  174022                          STD     AC0,(R2)+
013624  172537  001556                  LDD     @#D1010,AC1
013630  174122                          STD     AC1,(R2)+
013632  172737  001200                  LDD     @#$REG4,AC3
013636  170267  165622                  STFPS   FPS             ;GET STATUS
013642  173737  001170                  CMPD    @#$REG0,AC3
013646  170000                          CFCC
013650  001401                          BEQ     .+4
013652  104010                          ERROR   10              ;$REG0 AND $REG4 SHOULD =1010
013654  172537  001560                  LDD     @#D0101,AC1
013660  174142                          STD     AC1,-(R2)       ;STORED AC1 IN $REG4-7
013662  172012                          ADDD    (R2),AC0
013664  173042                          SUBD    -(R2),AC0
013666  173001                          SUBD    AC1,AC0
013670  175437  001170                  STCDI   AC0,@#$REG0
013674  022737  000000  001170          CMP     #0,@#$REG0
```

```
2417  013702  170267  165556                  STFPS   FPS             ;GET STATUS
2418  013706  001401                          BEQ     .+4
2419  013710  104012                          ERROR   12
2420  013712  170001                  MORE:   SETF                    ;SET FLOATING MODE
2421  013714  177027  000525                  LDCIF   #525,AC0        ;LOAD ACP WITH 525
2422  013720  177127  000252                  LDCIF   #252,AC1        ;LOAD AC1 WITH 252
2423  013724  174104                          STF     AC1,AC4         ;LOAD AC4 WITH 252
2424  013726  172104                          ADDF    AC4,AC1         ;ADD 252 TO 252
2425  013730  172701                          LDF     AC1,AC3         ;PUT ANS IN AC3=524
2426  013732  173003                          SUBF    AC3,AC0         ;SUB 524 FROM 525
2427  013734  175037  001170                  STEXP   AC0,@#$REGO     ;1 IN $REGO
2428  013740  170267  165520                  STFPS   FPS             ;GET STATUS
2429  013744  022737  000001  001170          CMP     #1,@#$REGO      ;CORRECT  ANS SHOULD BE 1
2430  013752  001401                          BEQ     .+4
2431  013754  104012                          ERROR   12              ;ANS SHOULD BE 5
2432  013756  177027  000021                  LDCIF   #21,AC0         ;LOAD AC0 WITH 21
2433  013762  171000                          MULF    AC0,AC0         ;21 TIMES 21 AC0 = 441
2434  013764  174427  040400                  DIVF    #2,AC0          ;DIV BY 2
2435  013770  012701  001552                  MOV     #FLTONE+4,R1    ;ADDR(DATA)+4
2436  013774  171441                          MODF    -(R1),AC0       ;MUL 1 * 441.
2437  013776  170267  165462                  STFPS   FPS             ;GET STATUS
2438  014002  175537  001170                  STCFI   AC1,@#$REGO     ;PUT $REGOIN $REGO
2439  014006  022737  000220  001170          CMP     #220,@#$REGO    ;IS IT EQUAL?
2440  014014  001401                          BEQ     .+4             ;YES
2441  014016  104012                          ERROR   12              ;SHOULD = 220
2442  014020  171427  041040                  MODF    #10.,AC0        ;GET FRACTION
2443  014024  175537  001170                  STCFI   AC1,@#$REGO     ;PUT AC0 INTO $REGO
2444  014030  022737  000005  001170          CMP     #5,@#$REGO      ;$REGO SHOULD = 5
2445  014036  001401                          BEQ     .+4
2446  014040  104012                          ERROR   12              ;$REGO SHOULD = 5
2447
2448
2449
2450                          ;*****************************************************************************
2451                          ;;*************************************************************
2452                          ;*TEST 72        ...CLEAR OUT CLOCKS BEFORE $EOP
2453                          ;;*************************************************************
2454  014042  000004          TST72:  SCOPE
2455  014044  005067  165160          CLR     $TIMES          ;NO ITER. OF THIS TEST
2456  014050  005067  165030          CLR     $ERFLG          ;NO ERRORS EITHER
2457
2458                                  ;*DONE WITH BASIC INSTRUCTION EXERCISER
2459                                  ;*CLEAR ALL LINE CLOCKS ENABLED IN SYSTEM
2460
2461  014054  105767  165436          TSTB    CLKPRS          ;KW11-L PRESENT ?
2462  014060  100002                  BPL     NKWL2           ;NO
2463  014062  005037  177546          CLR     @#LKS           ;YES, CLEAR IT
2464
2465  014066  005767  165424  NKWL2:  TST     CLKPRS          ;KW11-P PRESENT ?
2466  014072  100002                  BPL     NKWP2           ;NO
2467  014074  005037  172540          CLR     @#PLKS          ;YES, CLEAR IT
2468  014100                  NKWP2:
2469
```

```
                        ;;*****************************************************************
                        ;;      .ENABL  AMA             ;ASSEMBEL ALL PC RELATIVE REFERENCES AS ABSOLUTE
                        ;;*****************************************************************
                        ;;*****************************************************************
                        .SBTTL  SUB PASS END CONTROL

014100  000004                  SCOPE           ;CHECK FOR TEST ITERATIONS HERE
014102  005037  001230          CLR     $TIMES          ;DONT ITERATE THIS "TEST"
014106  005037  001104          CLR     $ERFLG          ;NO ERRORS HERE
014112  005037  001102          CLR     $TSTNM          ;ZAP TEST ## WHEN DONE WITH A PASS

                                ;IF TEST ONLY EITHER HFP OR WFP, ENTER "EOP" ROUTINE DIRECTLY

                                ; IF IN ALTERNATE HFP/WFP MODE,
                                ;   COMPLEMENT FLAG<5>, HFP ENABLE BIT,
                                ;   ENTER EOP ROUTINE ONLY IF ABOUT TO TEST HFP NEXT,
                                ;       TESTING SEQUENCE IS:  PASS#1  HFP SUB-PASS
                                ;                             PASS#1  WFP SUB-PASS
                                ;                             PASS#2  HFP SUB-PASS
                                ;                               ...

014116  076600  000022          MFD     ,RWHAMI         ;GET WHAMI INTO R0
014122  032700  000020          BIT     #BIT04,R0       ;1=HFP PRESENT, 0=NONE
014126  001423                  BEQ     $EOP            ;EXIT IF NONE

014130  032777  000002  165006  BIT     #SW01,@SWR      ;1=HFP OR WFP TEST ONLY
014136  001017                  BNE     $EOP            ;0=ALTERNATE HFP AND WFP TESTS

014140  012701  010000          MOV     #BIT12,R1       ;HFP PRESENT, AND IN ALTERNATE MODE;
014144  076600  000144          MFD     ,RFLAG          ;SO READ FLAGS,
014150  030100                  BIT     R1,R0           ;COMPLEMENT FLAG<5>=BIT12=HFP ENABLE FLAG
014152  001402                  BEQ     1$              ;
014154  040100                  BIC     R1,R0           ;CLEAR BIT 12
014156  000401                  BR      2$              ;
014160  050100          1$:     BIS     R1,R0           ;SET BIT 12
014162  076600  000344  2$:     MFD     ,WFLAG          ;REWRITE FLAGS

014166  030100                  BIT     R1,R0           ;HFP OR WFP NEXT ?
014170  001002                  BNE     $EOP            ;IF HFP AGAIN, START NEW PASS
014172  000137  003256          JMP     @#SUBPAS        ;IF WFP, NEXT SUBPASS

                        ;;*****************************************************************

                        .SBTTL  END OF PASS ROUTINE (MODIFIED SYSMAC)

                        ;*INCREMENT THE PASS NUMBER ($PASS)
                        ;*IF SW<10>=0, DING BELL ON PASS END
                        ;*IF THERE'S A MONITOR, GO TO IT
                        ;*  ELSE JUMP TO NEWPAS


014176                  $EOP:                   ;
014176  005037  001104          CLR     $ERFLG          ;ZERO ERROR COUNT
014202  005037  001102          CLR     $TSTNM          ;ZERO TEST NUMBER
```

```
2526  014206  005037  001230            CLR     $TIMES          ;ZERO NUMBER OF ITERATIONS
2527  014212  005237  001252            INC     $PASS           ;INCREMENT PASS COUNT,
2528  014216  042737  100000  001252    BIC     #100000,$PASS   ;  BUT NEVER LET IN GO NEGATIVE
2529  014224  005327                    DEC     (PC)+           ;PASS LOOP ?
2530  014226  000001            $EOPCT: .WORD   1               ;
2531  014230  003021                    BGT     $DOAGN          ;YES
2532  014232  012737                    MOV     (PC)+,@(PC)+    ;RESTORE COUNTER
2533  014234  000001            $ENDCT: .WORD   1               ;
2534  014236  014226                    $EOPCT
2535  014240  032777  002000  164676    BIT     #SW10,@SWR      ;BELL ON PASS END ?
2536  014246  001002                    BNE     $GET42          ;NO
2537  014250  104401  001234            TYPE    ,$BELL          ;YES
2538
2539  014254  013700  000042    $GET42: MOV     @#42,R0         ;GET MONITOR ADDRESS
2540  014260  001405                    BEQ     $DOAGN          ;NO MONITOR
2541  014262  000005                    RESET                   ;CLEAR WORLD
2542
2543  014264  004710            $ENDAD: JSR     PC,(R0)         ;GO TO MONITOR
2544  014266  000240                    NOP                     ;
2545  014270  000240                    NOP                     ;RESERVED FOR ACT11
2546  014272  000240                    NOP                     ;
2547
2548  014274  000137  003220    $DOAGN: JMP     @#NEWPAS        ;RETURN
2549
2550
2551
```

```
                                .SBTTL  TRAP/FEC TESTER SUBR
                        014300  013746  000244  TRPTST: MOV     @#FPPVEC,-(SP)          ; SAVE OLD FPPVEC PC
                        014304  013746  000246          MOV     @#FPPVEC+2,-(SP)        ; AND PS
                        014310  012737  014402  000244  MOV     #FPPTRP,@#FPPVEC        ; NEW VECTOR PC FOR TEST
                        014316  016537  000002  000246  MOV     2(R5),@#FPPVEC+2        ; NEW VECTOR PS FOR TEST
                        014324  012700  000006          MOV     #6,R0           ; LOAD $TMP0-5
                        014330  010501                  MOV     R5,R1           ;   WITH TEST DATA SETS
                        014332  012702  001210          MOV     #$TMP0,R2       ;   FOR DISPLAY LATER
                        014336  012122                  MOV     (R1)+,(R2)+     ;
                        014340  077002                  SOB     R0,.-2          ;
                        014342  012737  014352  001112  MOV     #FLPERR,$LPERR  ; ERROR LOOP TO HERE
                        014350  010602                  MOV     SP,R2           ; SAVE GOOD SP
                        014352  010206          FLPERR: MOV     R2,SP           ; RESET TO GOOD SP
                        014354  170165  000006          LDFPS   6(R5)           ; SET UP INITIAL FPS
                        014360  011537  177776          MOV     (R5),@#PSW      ; SET UP INITIAL PSW
                                                                                ; NOTE:  THE "LOADED PS" IS SETUP WITH PROCESSOR
                                                                                ;       PRIORITY = (7), SO THAT INTERRUPTS FROM
                                                                                ;       ANY CLOCKS ENABLED ARE EFFECTIVELY
                                                                                ;       LOCKED OUT DURING THIS TESTING
                                                                                ;       ROUTINE.
                        014364  000114                  JMP     (R4)            ; GO TEST W/O USING STACK

                                                ;-----  NO TRAP RETURNS HERE  ----------------------------------------
                        014366  032765  040000  000012  FPPNTP: BIT     #BIT14,12(R5)   ; NO TRAP EXPECTED ?
                        014374  001455                  BEQ     FCONT1          ; BR IF YES
                        014376  104014                  ERROR   14              ; DIDNT TRAP, SHOULD HAVE

                        014400  000453                  BR      FCONT1          ; CONTINUE WITH TEST

                                                ;-----  TRAP RETURNS HERE  -------------------------------------------
                        014402  013737  177776  001512  FPPTRP: MOV     @#PSW,MPSLOD    ; SAVE LOADED PSW
                        014410  011637  001506          MOV     (SP),OPCRCV     ; GET PUSHED PC
                        014414  016637  000002  001510  MOV     2(SP),OPSRCV    ; GET PUSHED PS
                        014422  010637  001514          MOV     SP,OSPRCV       ; GET SP AFTER TRAP
                        014426  012716  014434          MOV     #2$,(SP)        ; FUDGE RETURN
                        014432  000002                  RTI                     ; RETURN

                        014434  032765  040000  000012  2$:     BIT     #BIT14,12(R5)   ; TRAP EXPECTED ?
                        014442  001001                  BNE     1$              ; BR IF YES
                        014444  104015                  ERROR   15              ; TRAPPED, SHOULDNT HAVE

                        014446  010237  001504          1$:     MOV     R2,EXPSP        ; GENERATE EXPECTED SP CONTENTS
                        014452  162737  000004  001504  SUB     #4,EXPSP        ;   AFTER TRAP
                        014460  023737  001514  001504  CMP     OSPRCV,EXPSP    ; IS SP WHERE IT SHOULD BE?
                        014466  001401                  BEQ     64$             ;
                        014470  104016                  ERROR   16              ; NOT EQUAL, SIGNAL ERROR
                        014472                  64$:

                        014472  023737  001502  001506  CMP     EXPRTS,OPCRCV   ; CHECK STORED PC IS OK
                        014500  001401                  BEQ     65$             ;
                        014502  104017                  ERROR   17              ; NOT EQUAL, SIGNAL ERROR
                        014504                  65$:
                        014504  026537  000004  001510  CMP     4(R5),OPSRCV    ; CHECK STORED PS IS OK
```

```
2608  014512  001401                          BEQ     66$             ;
2609  014514  104020                          ERROR   20              ; NOT EQUAL, SIGNAL ERROR
2610  014516                          66$:
2611
2612  014516  026537  000002  001512          CMP     2(R5),NPSLOD    ; CHECK LOADED PS IS OK
2613  014524  001401                          BEQ     67$             ;
2614  014526  104021                          ERROR   21              ; NOT EQUAL, SIGNAL ERROR
2615  014530                          67$:
2616
2617                                  ;-----  CONTINUE  -----
2618  014530  170237  001464          FCONT1: STFPS   FPS             ; STORE FPS AFTER
2619  014534  170337  001466                  STST    FEC             ; STORE FEC/FEA AFTER
2620
2621  014540  023765  001464  000010          CMP     FPS,10(R5)      ; CHECK FPS
2622  014546  001401                          BEQ     65$             ; FPS IS OK
2623  014550  104023                          ERROR   23              ; FPS BAD
2624  014552  005765  000012          65$:    TST     12(R5)          ; DOES FEC/FEA APPLY?
2625  014556  100014                          BPL     66$             ; NO - SKIP TEST
2626  014560  013737  001500  001500          MOV     EXPFEA,EXPFEA   ; GET EXPECTED FEA
2627  014566  123765  001466  000012          CMPB    FEC,12(R5)      ; COMPARE FEC-S
2628  014574  001004                          BNE     64$             ; NOT EQUAL
2629  014576  023737  001470  001500          CMP     FEA,EXPFEA      ; COMPARE FEA-S
2630  014604  001401                          BEQ     66$             ; FEC, FEA OK
2631  014606  104022                  64$:    ERROR   22              ; FEC OR FEA ARE BAD
2632  014610                          66$:
2633
2634  014610  010206                          MOV     R2,SP           ; RESTORE GOOD SP AFTER TEST, IN CASE
2635
2636  014612  012637  000246                  MOV     (SP)+,@#FPPVEC+2        ; RESET STANDARD FPP VECTOR
2637  014616  012637  000244                  MOV     (SP)+,@#FPPVEC          ;
2638
2639  014622  005037  177776                  CLR     @#PSW           ; SETUP FOR INTERRUPT ENABLE, PRO
2640  014626  005003                          CLR     R3              ; ZEROES FOR LDUB
2641  014630  170003                          LDUB                    ; HFP UBRK <- R3, ZEROES
2642  014632  170127  040000                  LDFPS   #040000         ; INTR DISABLE, FMM=0
2643  014636  000207                          RTS     PC              ; RETURN TO TEST CALLER
2644
2645                                  ;-----  LOCAL FEC-CODE GENERATING SUBR  ------------------------------
2646
2647  014640                          FEC02:                          ;
2648  014640  170406                  FII02:  170406                  ; ILLEGAL FPP INSTR
2649  014642  174000                  NXT02:  STF     AC0,AC0         ; FORCE HFP/WFP SERVICE, EVEN AT PR7
2650  014644  000137  014366                  JMP     FPPNTP          ;
2651
2652  014650  172437  015002          FEC04:  LDF     F_1001,AC0      ;
2653  014654  174437  014772          FII04:  DIVF    FLTZER,AC0      ; X/0.0 = ?
2654  014660  174000                  NXT04:  STF     AC0,AC0         ; FORCE HFP/WFP SERVICE, EVEN AT PR7
2655  014662  000137  014366                  JMP     FPPNTP          ;
2656
2657  014666  172537  015002          FEC06:  LDF     FLT001,AC1      ;
2658  014672  175537  001170          FII06:  STCFI   AC1,$REG0       ; FLT001 > LGST I
2659  014676  174000                  NXT06:  STF     AC0,AC0         ; FORCE HFP/WFP SERVICE, EVEN AT PR7
2660  014700  000137  014366                  JMP     FPPNTP          ;
2661
2662  014704  172637  015002          FEC10:  LDF     FLT001,AC2      ;
2663  014710  172737  015006                  LDF     FLT002,AC3      ;
```

```
2664  014714  171203                FII10:  MULF    AC3,AC2         ; LG * LG = OVFLW
2665  014716  174000                NXT10:  STF     AC0,AC0         ; FORCE HFP/WFP SERVICE, EVEN AT PR7
2666  014720  000137  014366               JMP     FPPNTP          ;
2667
2668  014724  172637  015012        FEC12:  LDF     FLT003,AC2      ;
2669  014730  171237  015016        FII12:  MULF    FLT004,AC2      ; SM * SM = UNDFLW
2670  014734  174000                NXT12:  STF     AC0,AC0         ; FORCE HFP/WFP SERVICE, EVEN AT PR7
2671  014736  000137  014366               JMP     FPPNTP          ;
2672
2673  014742                        FEC14:                          ;
2674  014742  172737  014776        FII14:  LDF     FLTMZR,AC3      ; -0.0 -> AC3
2675  014746  174000                NXT14:  STF     AC0,AC0         ; FORCE HFP/WFP SERVICE, EVEN AT PR7
2676  014750  000137  014366               JMP     FPPNTP          ;
2677
2678  014754  012703  000342        FEC16:  MOV     #342,R3         ;HFP U-ADDR FOR "LDF/D" INSTR
2679  014760  170003                        LDUB                    ;
2680  014762  172400                FII16:  LDF     AC0,AC0         ; U-BREAK TRAP
2681  014764  174000                NXT16:  STF     AC0,AC0         ; FORCE HFP/WFP SERVICE, EVEN AT PR7
2682  014766  000137  014366               JMP     FPPNTP          ;
2683
2684                                ;-----  CONSTANTS  ----------------------------------------------------
2685  014772  000000  000000        FLTZER: .WORD   000000,000000   ; +0.0
2686  014776  100000  000000        FLTMZR: .WORD   100000,000000   ; -0.0
2687  015002  044000  000000        FLT001: .WORD   044000,000000   ; .1E+20 = 65536
2688  015006  074377  177777        FLT002: .WORD   074377,177777   ; .111 ... 1E+161
2689                                ; NOTE FLT001*FLT002 = .111 ... 1E+200  (OVERFLOW)
2690  015012  030000  000000        FLT003: .WORD   030000,000000   ; .1E-40
2691  015016  010000  000000        FLT004: .WORD   010000,000000   ; .1E-140
2692                                ; NOTE FLT003*FLT004 = .1E-201  (UNDERFLOW)
```

```
                                .SBTTL  LINE CLOCK INTERRUPT SERVICE ROUTINES
                        IKW11L: CLR     @#LKS           ;CLEAR STATUS
                                INC     KW11LC          ;BUMP COUNTER
                                BIC     #BIT15,KW11LC   ;NEVER OVERFLOW
                                BIS     #BIT6,@#LKS     ;RESTART CLOCK
                                RTI                     ;AND RETURN

                        IKW11P: CLR     @#PLKS          ;CLEAR STATUS
                                INC     KW11PC          ;BUMP COUNTER
                                BIC     #BIT15,KW11PC   ;NEVER OVERFLOW

                                ;THE NEXT 13 INSTRUCTIONS GENERATE A 10 BIT RANDOM
                                ;INTEGER FROM 0001-1777 INCLUSIVE TO FEED INTO THE
                                ;KW11-P COUNT SET REGISTER.
                                CLR     -(SP)           ;TEMP STORAGE FOR SHIFTED OUT BIT
                                ASR     KW11PR          ;SFT OLD CONTENTS RITE 1; LOB INTO C
                                ROL     (SP)            ;PUT OLD LOB ON STACK
                                BIT     #BIT2,KW11PR    ;WAS OTHER BIT ON ?
                                BEQ     1$              ;NO
                                COM     (SP)            ;YES, COMPLEMENT OLD LOB
                                BIC     #177776,(SP)    ;ELIMINATE JUNK
                        1$:     SWAB    (SP)            ;FAST SHIFT LOB TO BIT9 POSITION
                                ASL     (SP)            ;
                                BIS     (SP)+,KW11PR    ;PUT NEW HOB INTO COUNTER
                                BIC     #176000,KW11PR  ;NO HIGH ORDER JUNK
                                BNE     2$              ;ZERO IS AN ILLEGAL STATE,
                                MOV     #1777,KW11PR    ;  SO RESET TO START

                        2$:     MOV     KW11PR,@#PLKR   ;SET NEW COUNT
                                BIS     #BIT6+BIT0,@#PLKS       ;RESTART CLOCK
                                RTI                     ;AND RETURN
```

```
2725                                        .SBTTL  FPP TRAP CATCHER
2726
2727  015160  010637  001514          FPPILT: MOV     SP,OSPRCV       ;SP AFTER TRAP
2728  015164  012637  001472                  MOV     (SP)+,FPPOPC    ;POP OLD PC FOR DISPLAY
2729  015170  012637  001474                  MOV     (SP)+,FPPOPS    ;POP OLD PS FOR DISPLAY
2730  015174  170237  001464                  STFPS   FPS             ;GET FPS
2731  015200  170337  001466                  STST    FEC             ;GET FEC/FEA
2732  015204  005737  001464                  TST     FPS             ;TEST ERROR BIT
2733  015210  100007                          BPL     1$              ;OFF - NO ERROR BIT SET, BUT TRAPPED
2734                                                                  ;ON  - IT SHOULD BE ON A TRAP
2735  015212  032737  040000  001464          BIT     #040000,FPS     ;TEST INTERRUPT ENABLE BIT
2736  015220  001003                          BNE     1$              ;ON  - INTR DISABLED, BUT TRAPPED
2737                                                                  ;OFF - ABLE TO INTR, SO IGNORE IT,
2738  015222  012701  177777                  MOV     #-1,R1          ;       BUT FLAG THAT IT OCCURRED
2739  015226  000401                          BR      2$              ;       AND SKIP THE ERROR
2740  015230  104024                  1$:     ERROR   24              ;SIGNAL UNEXPECTED FPP TRAP
2741  015232  013746  001474          2$:     MOV     FPPOPS,-(SP)    ;PUSH PSW
2742  015236  013746  001472                  MOV     FPPOPC,-(SP)    ;PUSH PC
2743  015242  000002                          RTI                     ;CONTINUE,  RECOVER AT LAST TRAP ONLY
2744
```

```
                                .SBTTL  SCOPE HANDLER ROUTINE

                                ;;**********************************************************************
                                ;*THIS ROUTINE CONTROLS THE LOOPING OF SUBTESTS. IT WILL INCREMENT
                                ;*AND LOAD THE TEST NUMBER($TSTNM) INTO THE DISPLAY REG.(DISPLAY<15:0>)
                                ;*THE SWITCH OPTIONS PROVIDED BY THIS ROUTINE ARE:
                                ;*SW14=1         LOOP ON TEST
                                ;*SW11=1         INHIBIT ITERATIONS
                                ;*SW09=1         LOOP ON ERROR
                                ;*SW08=1         LOOP ON TEST IN "$LPTST"
                                ;*CALL
                                ;*      SCOPE           ;;SCOPE=IOT

015244                          $SCOPE:
015244                          64$:
015244  032777  040000  163672  1$:     BIT     #BIT14,@SWR     ;;LOOP ON PRESENT TEST?
015252  001114                          BNE     $OVER           ;;YES IF SW14=1
                                ;*****START OF CODE FOR THE XOR TESTER*****
015254  000416                  $XTSTR: BR      6$              ;;IF RUNNING ON THE "XOR" TESTER CHANGE
                                                                ;;THIS INSTRUCTION TO A "NOP" (NOP=240)
015256  013746  000004                  MOV     @#ERRVEC,-(SP)  ;;SAVE THE CONTENTS OF THE ERROR VECTOR
015262  012737  015302  000004          MOV     #5$,@#ERRVEC    ;;SET FOR TIMEOUT
015270  005737  177060                  TST     @#177060        ;;TIME OUT ON XOR?
015274  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
015300  000463                          BR      $SVLAD          ;;GO TO THE NEXT TEST
015302  022626                  5$:     CMP     (SP)+,(SP)+     ;;CLEAR THE STACK AFTER A TIME OUT
015304  012637  000004                  MOV     (SP)+,@#ERRVEC  ;;RESTORE THE ERROR VECTOR
015310  000423                          BR      7$              ;;LOOP ON THE PRESENT TEST
015312                          6$:;*****END OF CODE FOR THE XOR TESTER*****
015312  032777  000400  163624          BIT     #BIT08,@SWR     ;;LOOP ON SPEC. TEST?
015320  001404                          BEQ     2$              ;;BR IF NO
015322  023737  001150  001102          CMP     $LPTST,$TSTNM   ;;ON THE RIGHT TEST?
015330  001465                          BEQ     $OVER           ;;BR IF YES
015332  005737  001104          2$:     TST     $ERFLG          ;;HAS AN ERROR OCCURRED?
015336  001421                          BEQ     3$              ;;BR IF NO
015340  023737  001120  001104          CMP     $ERMAX,$ERFLG   ;;MAX. ERRORS FOR THIS TEST OCCURRED?
015346  101015                          BHI     3$              ;;BR IF NO
015350  032777  001000  163566          BIT     #BIT09,@SWR     ;;LOOP ON ERROR?
015356  001404                          BEQ     4$              ;;BR IF NO
015360  013737  001112  001110  7$:     MOV     $LPERR,$LPADR   ;;SET LOOP ADDRESS TO LAST SCOPE
015366  000446                          BR      $OVER
015370  005037  001104          4$:     CLR     $ERFLG          ;;ZERO THE ERROR FLAG
015374  005037  001230                  CLR     $TIMES          ;;CLEAR THE NUMBER OF ITERATIONS TO MAKE
015400  000415                          BR      1$              ;;ESCAPE TO THE NEXT TEST
015402  032777  004000  163534  3$:     BIT     #BIT11,@SWR     ;;INHIBIT ITERATIONS?
015410  001011                          BNE     1$              ;;BR IF YES
015412  005737  001252                  TST     $PASS           ;;IF FIRST PASS OF PROGRAM
015416  001406                          BEQ     1$              ;;      INHIBIT ITERATIONS
015420  005237  001106                  INC     $ICNT           ;;INCREMENT ITERATION COUNT
015424  023737  001230  001106          CMP     $TIMES,$ICNT    ;;CHECK THE NUMBER OF ITERATIONS MADE
015432  002024                          BGE     $OVER           ;;BR IF MORE ITERATION REQUIRED
015434  012737  000001  001106  1$:     MOV     #1,$ICNT        ;;REINITIALIZE THE ITERATION COUNTER
015442  013737  015520  001230          MOV     $MXCNT,$TIMES   ;;SET NUMBER OF ITERATIONS TO DO
015450  005237  001102          $SVLAD: INC     $TSTNM          ;;COUNT TEST NUMBERS
015454  013737  001102  001250          MOV     $TSTNM,$TESTN   ;;SET TEST NUMBER IN APT MAILBOX
015462  011637  001110                  MOV     (SP),$LPADR     ;;SAVE SCOPE LOOP ADDRESS
```

F07


```
  2801  015466  011637  001112                  MOV     (SP),$LPERR     ;;SAVE ERROR LOOP ADDRESS
  2802  015472  005037  001232                  CLR     $ESCAPE         ;;CLEAR THE ESCAPE FROM ERROR ADDRESS
  2803  015476  012737  000001  001120          MOV     #1,$ERMAX       ;;ONLY ALLOW ONE(1) ERROR ON NEXT TEST
  2804  015504  013777  001102  163434  $OVER:  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
  2805  015512  013716  001110                  MOV     $LPADR,(SP)     ;;FUDGE RETURN ADDRESS
  2806  015516  000002                          RTI                     ;;FIXES PS
  2807  015520  003720          $MXCNT: 2000.                           ;;MAX. NUMBER OF ITERATIONS
```

```
                                        .SBTTL  ERROR HANDLER ROUTINE

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

015522                                  $ERROR:
015522  010037  001526                          MOV     R0,EREG0        ; DISPLAY R0
015526  010137  001530                          MOV     R1,EREG1        ;         R1
015532  010237  001532                          MOV     R2,EREG2        ;         R2
015536  010337  001534                          MOV     R3,EREG3        ;         R3
015542  010437  001536                          MOV     R4,EREG4        ;         R4
015546  010537  001540                          MOV     R5,EREG5        ;         R5
015552  010637  001542                          MOV     R6,EREG6        ; GET R6(SP) BEFORE TRAP
015556  062737  000004  001542                  ADD     #4,EREG6        ;
015564  011637  001544                          MOV     (SP),EREG7      ; PC -> ERROR CALL INSTR
015570  005237  001104                  7$:     INC     $ERFLG          ;;SET THE ERROR FLAG
015574  001775                                  BEQ     7$              ;;DON'T LET THE FLAG GO TO ZERO
015576  013777  001102  163342                  MOV     $TSTNM,@DISPLAY ;;DISPLAY TEST NUMBER
015604  032777  002000  163332                  BIT     #BIT10,@SWR     ;;BELL ON ERROR?
015612  001402                                  BEQ     1$              ;;NO - SKIP
015614  104401  001234                          TYPE    ,$BELL          ;;RING BELL
015620  005237  001114                  1$:     INC     $ERTTL          ;;COUNT THE NUMBER OF ERRORS
015624  011637  001122                          MOV     (SP),$ERRPC     ;;GET ADDRESS OF ERROR INSTRUCTION
015630  162737  000002  001122                  SUB     #2,$ERRPC
015636  117737  163260  001116                  MOVB    @#ERRPC,$ITEMB  ;;STRIP AND SAVE THE ERROR ITEM CODE
015644  032777  020000  163272                  BIT     #BIT13,@SWR     ;;SKIP TYPEOUT IF SET
015652  001004                                  BNE     20$             ;;SKIP TYPEOUTS
015654  004737  015764                          JSR     PC,$TYPERR      ;;GO TO USER ERROR ROUTINE
015660  104401  001241                          TYPE    ,$CRLF
015664                                  20$:
015664  122737  000001  001264                  CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
015672  001007                                  BNE     2$              ;;NO,SKIP APT ERROR REPORT
015674  113737  001116  015706                  MOVB    $ITEMB,21$      ;;SET ITEM NUMBER AS ERROR NUMBER
015702  004737  016500                          JSR     PC,$ATY4        ;;REPORT FATAL ERROR TO APT
015706     000                  21$:    .BYTE   0
015707     000                          .BYTE   0
015710  000777                  22$:    BR      22$             ;;APT ERROR LOOP
015712  005777  163226          2$:     TST     @SWR            ;;HALT ON ERROR
015716  100001                          BPL     3$              ;;SKIP IF CONTINUE
015720  000000                          HALT                    ;;HALT ON ERROR!
015722  032777  001000  163214  3$:     BIT     #BIT09,@SWR     ;;LOOP ON ERROR SWITCH SET?
015730  001402                          BEQ     4$              ;;BR IF NO
015732  013716  001112                  MOV     $LPERR,(SP)     ;;FUDGE RETURN FOR LOOPING
015736  005737  001232          4$:     TST     $ESCAPE         ;;CHECK FOR AN ESCAPE ADDRESS
015742  001402                          BEQ     5$              ;;BR IF NONE
015744  013716  001232                  MOV     $ESCAPE,(SP)    ;;FUDGE RETURN ADDRESS FOR ESCAPE
015750                          5$:
```

```
 2864  015750  022737  014264  000042          CMP     #SENDAD,@#42    ;;ACT-11 AUTO-ACCEPT?
 2865  015756  001001                          BNE     6$              ;;BRANCH IF NO
 2866  015760  000000                          HALT                    ;;YES
 2867  015762                          6$:
 2868  015762  000002                  64$:    RTI                     ;RETURN
```

```
                                ;;*****************************************************************
                                .SBTTL  ERROR MESSAGE TYPEOUT ROUTINE (MODIFIED SYSMAC)

                                ;*THIS ROUTINE USES THE "ITEM CONTROL BYTE" ($ITEMB) TO DETERMINE WHICH
                                ;*ERROR IS TO BE REPORTED.  IT THEN OBTAINS, FROM THE "ERROR TABLE",
                                ;*($ERRTB) THE ERROR MESSAGE, DATA HEADER, AND DATA VALUES TO PRINT.
                                ;*THIS ROUTINE IS IDENTICAL TO THE SYSMAC ROUTINE $ERRTYP, EXCEPT THIS
                                ;*ROUTINE PUTS A <HT> BETWEEN OCTAL TYPED DATA VALUES, SO THAT EACH
                                ;*VALUE STARTS AT A HORIZONTAL TAB STOP.  ALSO, THE DATA FORMAT
                                ;*POINTER HAS BEEN ELIMINATED FROM THE ERROR VECTOR.  THIS ROUTINE
                                ;*ALSO ALWAYS PRINTS $TESTN AND $ERRPC AS THE FIRST TWO DATA ELEMENTS
                                ;*(WITH APPROPRIATE HEADERS).

015764                          $TYPERR:
015764  104401                          TYPE                    ; TYPE "HOT" OR "WARM"
015766  001241                  HOTWRM: .WORD   $CRLF            ; PTR TO MESSAGE
015770  010046                          MOV     R0,-(SP)        ; SAVE R0
015772  010146                          MOV     R1,-(SP)        ; SAVE R1
015774  005000                          CLR     R0              ; PICKUP ITEM INDEX
015776  153700  001116                  BISB    @#$ITEMB,R0     ;
016002  001004                          BNE     1$              ; IF ITEM NUMBER FROM ERROR 0,
                                                                ;   JUST TYPE PC OF ERROR
016004  013746  001122                  MOV     $ERRPC,-(SP)    ; GET ERROR PC FOR TYPEOUT
016010  104402                          TYPOC                   ; TYPE OCTAL, ALL DIGITS
016012  000452                          BR      7$              ; EXIT
016014  005300                  1$:     DEC     R0              ; ADJUST ERROR # FOR TABLE INDEX
016016  006300                          ASL     R0              ;   OF 6 BYTES/ENTRY
016020  010001                          MOV     R0,R1           ;
016022  006300                          ASL     R0              ;
016024  060100                          ADD     R1,R0           ;
016026  062700  001274                  ADD     #$ERRTB,R0      ; FORM TABLE PTR
016032  012037  016042                  MOV     (R0)+,2$        ; PICKUP "ERROR MESSAGE" PTR
016036  001404                          BEQ     3$              ; SKIP TYPEOUT IF NULL
016040  104401                          TYPE                    ; TYPE "ERROR MESSAGE"
016042  000000                  2$:     .WORD   0               ; "ERROR MESSAGE" PTR HERE
016044  104401  001241                  TYPE    ,$CRLF          ; CR & LF
016050  104401  016160          3$:     TYPE    ,11$            ; "TEST #    ERR PC" HEADER
016054  012037  016064                  MOV     (R0)+,4$        ; PICKUP "DATA HEADER" PTR
016060  001402                          BEQ     5$              ; SKIP TYPEOUT IF NULL
016062  104401                          TYPE                    ; TYPE "DATA HEADER"
016064  000000                  4$:     .WORD   0               ; "DATA HEADER" PTR HERE
016066  104401  001241          5$:     TYPE    ,$CRLF          ; CR & LF
016072  017746  000054                  MOV     @8$,-(SP)       ; ($TESTN)
016076  104402                          TYPOC                   ; OCTAL W/ LEADING ZEROS
016100  104401  016156                  TYPE    ,10$            ; <HT>
016104  017746  000044                  MOV     @9$,-(SP)       ; ($ERRPC)
016110  104402                          TYPOC                   ; OCTAL W/ LEADING ZEROS
016112  104401  016156                  TYPE    ,10$            ; <HT>
016116  011000                          MOV     (R0),R0         ; PICKUP "DATA TABLE" PTR
016120  001407                          BEQ     7$              ; EXIT IF NULL
016122  013046                  6$:     MOV     @(R0)+,-(SP)    ; SAVE ... FOR TYPEOUT
016124  104402                          TYPOC                   ; TYPE OCTAL, ALL DIGITS
016126  005710                          TST     (R0)            ; ANOTHER NUMBER ?
016130  001403                          BEQ     7$              ; NO - EXIT
016132  104401  016156                  TYPE    ,10$            ; TAB BETWEEN ELEMENTS
```

```
2925  016136  000771                        BR      6$              ;LOOP ON DATA TABLE VECTOR
2926  016140  012601                7$:     MOV     (SP)+,R1        ;RESTORE R1
2927  016142  012600                        MOV     (SP)+,R0        ;RESTORE R0
2928  016144  104401  001241                TYPE    ,$CRLF          ;CR & LF
2929  016150  000207                        RTS     PC              ;RETURN
2930  016152  001250                8$:     .WORD   $TESTN          ;
2931  016154  001122                9$:     .WORD   $ERRPC          ;
2932  016156  000011                10$:    .ASCIZ  <11>            ;<HT>
2933  016160  042524  052123  021440  11$:    .ASCIZ  "TEST # ERR PC
2934  016166  042411  051122  050040
2935  016174  004503     000
2936          016200                        .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
                                ;*

016200  105737  001165          $TYPE:  TSTB    $TPFLG          ;;IS THERE A TERMINAL?
016204  100002                          BPL     1$              ;;BR IF YES
016206  000000                          HALT                    ;;HALT HERE IF NO TERMINAL
016210  000430                          BR      3$              ;;LEAVE
016212  010046                  1$:     MOV     R0,-(SP)        ;;SAVE R0
016214  017600  000002                  MOV     @2(SP),R0       ;;GET ADDRESS OF ASCIZ STRING
016220  122737  000001  001264          CMPB    #APTENV,$ENV    ;;RUNNING IN APT MODE
016226  001011                          BNE     62$             ;;NO,GO CHECK FOR APT CONSOLE
016230  132737  000100  001265          BITB    #APTSPOOL,$ENVM ;;SPOOL MESSAGE TO APT
016236  001405                          BEQ     62$             ;;NO,GO CHECK FOR CONSOLE
016240  010037  016250                  MOV     R0,61$          ;;SETUP MESSAGE ADDRESS FOR APT
016244  004737  016470                  JSR     PC,$ATY3        ;;SPOOL MESSAGE TO APT
016250  000000                  61$:    .WORD   0               ;;MESSAGE ADDRESS
016252  132737  000040  001265  62$:    BITB    #APTCSUP,$ENVM  ;;APT CONSOLE SUPPRESSED
016260  001003                          BNE     60$             ;;YES,SKIP TYPE OUT
016262  112046                  2$:     MOVB    (R0)+,-(SP)     ;;PUSH CHARACTER TO BE TYPED ONTO STACK
016264  001005                          BNE     4$              ;;BR IF IT ISN'T THE TERMINATOR
016266  005726                          TST     (SP)+           ;;IF TERMINATOR POP IT OFF THE STACK
016270  012600                  60$:    MOV     (SP)+,R0        ;;RESTORE R0
016272  062716  000002          3$:     ADD     #2,(SP)         ;;ADJUST RETURN PC
016276  000002                          RTI                     ;;RETURN
016300  122716  000011          4$:     CMPB    #HT,(SP)        ;;BRANCH IF <HT>
016304  001430                          BEQ     8$
016306  122716  000200                  CMPB    #CRLF,(SP)      ;;BRANCH IF NOT <CRLF>
016312  001006                          BNE     5$
016314  005726                          TST     (SP)+           ;;POP  <CR><LF> EQUIV
016316  104401                          TYPE                    ;;TYPE A CR AND LF
016320  001241                          $CRLF
016322  105037  016456                  CLRB    $CHARCNT        ;;CLEAR CHARACTER COUNT
016326  000755                          BR      2$              ;;GET NEXT CHARACTER
016330  004737  016412          5$:     JSR     PC,$TYPEC       ;;GO TYPE THIS CHARACTER
016334  123726  001164          6$:     CMPB    $FILLC,(SP)+    ;;IS IT TIME FOR FILLER CHARS.?
016340  001350                          BNE     2$              ;;IF NO GO GET NEXT CHAR.
016342  013746  001162                  MOV     $NULL,-(SP)     ;;GET # OF FILLER CHARS. NEEDED
                                                                ;;AND THE NULL CHAR.
016346  105366  000001          7$:     DECB    1(SP)           ;;DOES A NULL NEED TO BE TYPED?
016352  002770                          BLT     6$              ;;BR IF NO--GO POP THE NULL OFF OF STACK
016354  004737  016412                  JSR     PC,$TYPEC       ;;GO TYPE A NULL
016360  105337  016456                  DECB    $CHARCNT        ;;DO NOT COUNT AS A COUNT
```

```
 2993  016364  000770                        BR      7$              ;;LOOP
 2994
 2995                                        ;HORIZONTAL TAB PROCESSOR
 2996
 2997  016366  112716  000040                8$:     MOVB    #' ,(SP)        ;;REPLACE TAB WITH SPACE
 2998  016372  004737  016412                9$:     JSR     PC,$TYPEC       ;;TYPE A SPACE
 2999  016376  132737  000007  016456                BITB    #7,$CHARCNT     ;;BRANCH IF NOT AT
 3000  016404  001372                                BNE     9$              ;;TAB STOP
 3001  016406  005726                                TST     (SP)+           ;;POP SPACE OFF STACK
 3002  016410  000724                                BR      2$              ;;GET NEXT CHARACTER
 3003  016412  105777  162540          $TYPEC: TSTB    @$TPS           ;;WAIT UNTIL PRINTER IS READY
 3004  016416  100375                                BPL     $TYPEC
 3005  016420  116677  000002  162532                MOVB    2(SP),@$TPB     ;;LOAD CHAR TO BE TYPED INTO DATA REG.
 3006  016426  122766  000015  000002                CMPB    #CR,2(SP)       ;;IS CHARACTER A CARRIAGE RETURN?
 3007  016434  001003                                BNE     1$              ;;BRANCH IF NO
 3008  016436  105037  016456                        CLRB    $CHARCNT        ;;YES--CLEAR CHARACTER COUNT
 3009  016442  000406                                BR      $TYPEX          ;;EXIT
 3010  016444  122766  000012  000002  1$:     CMPB    #LF,2(SP)       ;;IS CHARACTER A LINE FEED?
 3011  016452  001402                                BEQ     $TYPEX          ;;BRANCH IF YES
 3012  016454  105227                                INCB    (PC)+           ;;COUNT THE CHARACTER
 3013  016456  000000          $CHARCNT:.WORD  0               ;;CHARACTER COUNT STORAGE
 3014  016460  000207          $TYPEX: RTS     PC
 3015
```

```
                                .SBTTL  APT COMMUNICATIONS ROUTINE
                                ;;*****************************************************************
016462  112737  000001  016726  $ATY1:  MOVB    #1,$FFLG        ;;TO REPORT FATAL ERROR
016470  112737  000001  016724  $ATY3:  MOVB    #1,$MFLG        ;;TO TYPE A MESSAGE
016476  000403                  BR      $ATYC
016500  112737  000001  016726  $ATY4:  MOVB    #1,$FFLG        ;;TO ONLY REPORT FATAL ERROR
016506                          $ATYC:
016506  010046                  MOV     R0,-(SP)        ;;PUSH R0 ON STACK
016510  010146                  MOV     R1,-(SP)        ;;PUSH R1 ON STACK
016512  105737  016724          TSTB    $MFLG           ;;SHOULD TYPE A MESSAGE?
016516  001450                  BEQ     5$              ;;IF NOT:  BR
016520  122737  000001  001264  CMPB    #APTENV,$ENV    ;;OPERATING UNDER APT?
016526  001031                  BNE     3$              ;;IF NOT:  BR
016530  132737  000100  001265  BITB    #APTSPOOL,$ENVM ;;SHOULD SPOOL MESSAGES?
016536  001425                  BEQ     3$              ;;IF NOT:  BR
016540  017600  000004          MOV     @4(SP),R0       ;;GET MESSAGE ADDR.
016544  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
016552  105737  001244          1$:     TSTB    $MSGTYPE        ;;SEE IF DONE W/ LAST XMISSION?
016556  001375                  BNE     1$              ;;IF NOT:  WAIT
016560  010037  001260          MOV     R0,$MSGAD       ;;PUT ADDR IN MAILBOX
016564  105720                  2$:     TSTB    (R0)+           ;;FIND END OF MESSAGE
016566  001376                  BNE     2$
016570  163700  001260          SUB     $MSGAD,R0       ;;SUB START OF MESSAGE
016574  006200                  ASR     R0              ;;GET MESSAGE LNGTH IN WORDS
016576  010037  001262          MOV     R0,$MSGLGT      ;;PUT LENGTH IN MAILBOX
016602  012737  000004  001244  MOV     #4,$MSGTYPE     ;;TELL APT TO TAKE MSG.
016610  000413                  BR      5$
016612  017637  000004  016636  3$:     MOV     @4(SP),4$       ;;PUT MSG ADDR IN JSR LINKAGE
016620  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDRESS
016626  013746  177776          MOV     177776,-(SP)    ;;PUSH 177776 ON STACK
016632  004737  016200          JSR     PC,$TYPE        ;;CALL TYPE MACRO
016636  000000                  4$:     .WORD   0
016640                          5$:
016640  105737  016726          10$:    TSTB    $FFLG           ;;SHOULD REPORT FATAL ERROR?
016644  001416                  BEQ     12$             ;;IF NOT:  BR
016646  005737  001264          TST     $ENV            ;;RUNNING UNDER APT?
016652  001413                  BEQ     12$             ;;IF NOT:  BR
016654  005737  001244          11$:    TST     $MSGTYPE        ;;FINISHED LAST MESSAGE?
016660  001375                  BNE     11$             ;;IF NOT:  WAIT
016662  017637  000004  001246  MOV     @4(SP),$FATAL   ;;GET ERROR #
016670  062766  000002  000004  ADD     #2,4(SP)                ;;BUMP RETURN ADDR.
016676  005237  001244          INC     $MSGTYPE        ;;TELL APT TO TAKE ERROR
016702  105037  016726          12$:    CLRB    $FFLG           ;;CLEAR FATAL FLAG
016706  105037  016725          CLRB    $LFLG           ;;CLEAR LOG FLAG
016712  105037  016724          CLRB    $MFLG           ;;CLEAR MESSAGE FLAG
016716  012601                  MOV     (SP)+,R1        ;;POP STACK INTO R1
016720  012600                  MOV     (SP)+,R0        ;;POP STACK INTO R0
016722  000207                  RTS     PC              ;;RETURN
016724     000  $MFLG:  .BYTE   0               ;;MESSG. FLAG
016725     000  $LFLG:  .BYTE   0               ;;LOG FLAG
016726     000  $FFLG:  .BYTE   0               ;;FATAL FLAG
016730                  .EVEN
000200          APTSIZE=200
000001          APTENV=001
000100          APTSPOOL=100
```

```
  3072          000040                  APTCSUP=040
```

```
                                .SBTTL  BINARY TO OCTAL (ASCII) AND TYPE

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

016730 017646 000000            $TYPOS: MOV     @(SP),-(SP)     ;;PICKUP THE MODE
016734 116637 000001 017153             MOVB    1(SP),$OFILL    ;;LOAD ZERO FILL SWITCH
016742 112637 017155                    MOVB    (SP)+,$OMODE+1  ;;NUMBER OF DIGITS TO TYPE
016746 062716 000002                    ADD     #2,(SP)         ;;ADJUST RETURN ADDRESS
016752 000406                           BR      $TYPON
016754 112737 000001 017153     $TYPOC: MOVB    #1,$OFILL       ;;SET THE ZERO FILL SWITCH
016762 112737 000006 017155             MOVB    #6,$OMODE+1     ;;SET FOR SIX(6) DIGITS
016770 112737 000005 017152     $TYPON: MOVB    #5,$OCNT        ;;SET THE ITERATION COUNT
016776 010346                           MOV     R3,-(SP)        ;;SAVE R3
017000 010446                           MOV     R4,-(SP)        ;;SAVE R4
017002 010546                           MOV     R5,-(SP)        ;;SAVE R5
017004 113704 017155                    MOVB    $OMODE+1,R4     ;;GET THE NUMBER OF DIGITS TO TYPE
017010 005404                           NEG     R4
017012 062704 000006                    ADD     #6,R4           ;;SUBTRACT IT FOR MAX. ALLOWED
017016 110437 017154                    MOVB    R4,$OMODE       ;;SAVE IT FOR USE
017022 113704 017153                    MOVB    $OFILL,R4       ;;GET THE ZERO FILL SWITCH
017026 016605 000012                    MOV     12(SP),R5       ;;PICKUP THE INPUT NUMBER
017032 005003                           CLR     R3              ;;CLEAR THE OUTPUT WORD
017034 006105                   1$:     ROL     R5              ;;ROTATE MSB INTO "C"
017036 000404                           BR      3$              ;;GO DO MSB
017040 006105                   2$:     ROL     R5              ;;FORM THIS DIGIT
017042 006105                           ROL     R5
017044 006105                           ROL     R5
017046 010503                           MOV     R5,R3
017050 006103                   3$:     ROL     R3              ;;GET LSB OF THIS DIGIT
017052 105337 017154                    DECB    $OMODE          ;;TYPE THIS DIGIT?
017056 100016                           BPL     7$              ;;BR IF NO
017060 042703 177770                    BIC     #177770,R3      ;;GET RID OF JUNK
017064 001002                           BNE     4$              ;;TEST FOR 0
017066 005704                           TST     R4              ;;SUPPRESS THIS 0?
017070 001403                           BEQ     5$              ;;BR IF YES
```

```
 3129  017072  005204                  4$:     INC     R4              ;;DON'T SUPPRESS ANYMORE 0'S
 3130  017074  052703  000060                  BIS     #'0,R3          ;;MAKE THIS DIGIT ASCII
 3131  017100  052703  000040          5$:     BIS     #' ,R3          ;;MAKE ASCII IF NOT ALREADY
 3132  017104  110337  017150                  MOVB    R3,8$           ;;SAVE FOR TYPING
 3133  017110  104401  017150                  TYPE    ,8$             ;;GO TYPE THIS DIGIT
 3134  017114  105337  017152          7$:     DECB    $OCNT           ;;COUNT BY 1
 3135  017120  003347                          BGT     2$              ;;BR IF MORE TO DO
 3136  017122  002402                          BLT     6$              ;;BR IF DONE
 3137  017124  005204                          INC     R4              ;;INSURE LAST DIGIT ISN'T A BLANK
 3138  017126  000744                          BR      2$              ;;GO DO THE LAST DIGIT
 3139  017130  012605                  6$:     MOV     (SP)+,R5        ;;RESTORE R5
 3140  017132  012604                          MOV     (SP)+,R4        ;;RESTORE R4
 3141  017134  012603                          MOV     (SP)+,R3        ;;RESTORE R3
 3142  017136  016666  000002  000004          MOV     2(SP),4(SP)     ;;SET THE STACK FOR RETURNING
 3143  017144  012616                          MOV     (SP)+,(SP)
 3144  017146  000002                          RTI                     ;;RETURN
 3145  017150     000          8$:     .BYTE   0               ;;STORAGE FOR ASCII DIGIT
 3146  017151     000                  .BYTE   0               ;;TERMINATOR FOR TYPE ROUTINE
 3147  017152     000          $OCNT:  .BYTE   0               ;;OCTAL DIGIT COUNTER
 3148  017153     000          $OFILL: .BYTE   0               ;;ZERO FILL SWITCH
 3149  017154  000000          $OMODE: .WORD   0               ;;NUMBER OF DIGITS TO TYPE
```

```
                                .SBTTL  TRAP DECODER

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

017156  010046                  $TRAP:  MOV     R0,-(SP)        ;;SAVE R0
017160  016600  000002                  MOV     2(SP),R0        ;;GET TRAP ADDRESS
017164  005740                          TST     -(R0)           ;;BACKUP BY 2
017166  111000                          MOVB    (R0),R0         ;;GET RIGHT BYTE OF TRAP
017170  006300                          ASL     R0              ;;POSITION FOR INDEXING
017172  016000  017212                  MOV     $TRPAD(R0),R0   ;;INDEX TO TABLE
017176  000200                          RTS     R0              ;;GO TO ROUTINE


                                ;;THIS IS USE TO HANDLE THE "GETPRI" MACRO

017200  011646                  $TRAP2: MOV     (SP),-(SP)      ;;MOVE THE PC DOWN
017202  016666  000004  000002          MOV     4(SP),2(SP)     ;;MOVE THE PSW DOWN
017210  000002                          RTI                     ;;RESTORE THE PSW

                                .SBTTL  TRAP TABLE

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

                                ;       ROUTINE
                                ;       -------
017212  017200                  $TRPAD: .WORD   $TRAP2
017214  016200                          $TYPE   ;;CALL=TYPE     TRAP+1(104401)  TTY TYPEOUT ROUTINE
017216  016754                          $TYPOC  ;;CALL=TYPOC    TRAP+2(104402)  TYPE OCTAL NUMBER (WITH LEADING ZEROS)
017220  016730                          $TYPOS  ;;CALL=TYPOS    TRAP+3(104403)  TYPE OCTAL NUMBER (NO LEADING ZEROS)
017222  016770                          $TYPON  ;;CALL=TYPON    TRAP+4(104404)  TYPE OCTAL NUMBER (AS PER LAST CALL)
```

```
                                        .SBTTL  POWER DOWN AND UP ROUTINES

                                        ;;*****************************************************************
                                        ;POWER DOWN ROUTINE
017224  012737  017376  000024  $PWRDN: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST UP
017232  012737  000340  000026          MOV     #340,@#PWRVEC+2  ;;PRIO:7
017240  010046                          MOV     R0,-(SP)         ;;PUSH R0 ON STACK
017242  010146                          MOV     R1,-(SP)         ;;PUSH R1 ON STACK
017244  010246                          MOV     R2,-(SP)         ;;PUSH R2 ON STACK
017246  010346                          MOV     R3,-(SP)         ;;PUSH R3 ON STACK
017250  010446                          MOV     R4,-(SP)         ;;PUSH R4 ON STACK
017252  010546                          MOV     R5,-(SP)         ;;PUSH R5 ON STACK
017254  017746  161664                  MOV     @SWR,-(SP)       ;;PUSH @SWR ON STACK
017260  010637  017402                  MOV     SP,$SAVR6        ;;SAVE SP
017264  012737  017276  000024          MOV     #$PWRUP,@#PWRVEC ;;SET UP VECTOR
017272  000000                          HALT
017274  000776                          BR      .-2              ;;HANG UP

                                        ;;*****************************************************************
                                        ;POWER UP ROUTINE
017276  012737  017376  000024  $PWRUP: MOV     #$ILLUP,@#PWRVEC ;;SET FOR FAST DOWN
017304  013706  017402                  MOV     $SAVR6,SP        ;;GET SP
017310  005037  017402                  CLR     $SAVR6           ;;WAIT LOOP FOR THE TTY
017314  005237  017402          1$:     INC     $SAVR6           ;;WAIT FOR THE INC
017320  001375                          BNE     1$               ;;OF  WORD
017322  011600                          MOV     (SP),R0          ;GET SAVED SWR OFF STACK
017324  076600  000226                  MED      226             ;RESTORE SWR CONTENTS
017330  012677  161610                  MOV     (SP)+,@SWR       ;;POP STACK INTO @SWR
017334  012605                          MOV     (SP)+,R5         ;;POP STACK INTO R5
017336  012604                          MOV     (SP)+,R4         ;;POP STACK INTO R4
017340  012603                          MOV     (SP)+,R3         ;;POP STACK INTO R3
017342  012602                          MOV     (SP)+,R2         ;;POP STACK INTO R2
017344  012601                          MOV     (SP)+,R1         ;;POP STACK INTO R1
017346  012600                          MOV     (SP)+,R0         ;;POP STACK INTO R0
017350  012737  017224  000024          MOV     #$PWRDN,@#PWRVEC ;;SET UP THE POWER DOWN VECTOR
017356  012737  000340  000026          MOV     #340,@#PWRVEC+2  ;;PRIO:7
017364  104401                          TYPE                     ;REPORT THE POWER FAILURE
017366  017404          $PWRMG: .WORD   $POWER                   ;;POWER FAIL MESSAGE POINTER
017370  012716                  MOV     (PC)+,(SP)               ;;RESTART AT START
017372  002400          $PWRAD: .WORD   START                    ;;RESTART ADDRESS
017374  000002                  RTI
017376  000000          $ILLUP: HALT                             ;;THE POWER UP SEQUENCE WAS STARTED
017400  000776                  BR      .-2                      ;;  BEFORE THE POWER DOWN WAS COMPLETE
017402  000000          $SAVR6: 0                                ;;PUT THE SP HERE
017404  005015  047520  042527  $POWER: .ASCIZ  <15><12>"POWER"
017412  000122
                                        .EVEN
```

```
                                        .SBTTL  ERROR MESSAGES, DATA HEADERS, DATA VECTORS, ETC

                                        ;MESSAGE PREFIXES
017414  047510  035124  000040  ASCHOT: .ASCIZ  "HOT: "
017422  040527  046522  020072  ASCWRM: .ASCIZ  "WARM: "
017430     000

                                        ;ERROR MESSAGES HERE
017431     122  041505  044505  EMA:    .ASCIZ  "RECEIVED FPS IS BAD"
017436  042526  020104  050106
017444  020123  051511  041040
017452  042101     000

017455     122  041505  044505  EMB:    .ASCIZ  "RECEIVED FEC IS BAD"
017462  042526  020104  042506
017470  020103  051511  041040
017476  042101     000

017501     122  041505  044505  EMC:    .ASCIZ  "RECEIVED FEA IS BAD"
017506  042526  020104  042506
017514  020101  051511  041040
017522  042101     000

017525     122  041505  044505  EMD:    .ASCIZ  "RECEIVED PSW CC-S ARE BAD"
017532  042526  020104  051520
017540  020127  041503  051455
017546  040440  042522  041040
017554  042101     000

017557     102  042101  040440  EME:    .ASCIZ  "BAD ADDRESS IN R0"
017564  042104  042522  051523
017572  044440  020116  030122
017600     000

017601     102  042101  051440  EMF:    .ASCIZ  "BAD STACK POINTER"
017606  040524  045503  050040
017614  044517  052116  051105
017622     000

017623     122  051505  046125  EMH:    .ASCIZ  "RESULT OF FPP ARITHMETIC OPERATION IS BAD"
017630  020124  043117  043040
017636  050120  040440  044522
017644  044124  042515  044524
017652  020103  050117  051105
017660  052101  047511  020116
017666  051511  041040  042101
017674     000

017675     122  041505  044505  EMI:    .ASCIZ  "RECEIVED FEC/FEA IS BAD"
017702  042526  020104  042506
017710  027503  042506  020101
017716  051511  041040  042101
017724     000

017725     125  042516  050130  EMJ:    .ASCIZ  "UNEXPECTED FPP TRAP, IGNORED AND CONTINUING"
017732  041505  042524  020104
017740  050106  020120  051124
017746  050101  020054  043511
017754  047516  042522  020104
017762  047101  020104  047503
017770  052116  047111  044525
017776  043516     000

020001     103  052520  042040  EMP:    .ASCIZ  "CPU DID NOT TRAP ON FPP EXCEPTION, BUT IT SHOULD HAVE"
020006  042111  047040  052117
```

```
3290  020014  052040  040522  020120
3291  020022  047117  043040  050120
3292  020030  042440  041530  050105
3293  020036  044524  047117  020054
3294  020044  052502  020124  052111
3295  020052  051440  047510  046125
3296  020060  020104  040510  042526
3297  020066     000
3298  020067     103  052520  052040  EMQ:    .ASCIZ  "CPU TRAPPED ON FPP EXCEPTION, BUT IT SHOULD NOT HAVE"
3299  020074  040522  050120  042105
3300  020102  047440  020116  050106
3301  020110  020120  054105  042503
3302  020116  052120  047511  026116
3303  020124  041040  052125  044440
3304  020132  020124  044123  052517
3305  020140  042114  047040  052117
3306  020146  044040  053101  000105
3307  020154  043101  042524  020122  EMR:    .ASCIZ  "AFTER TRAP, STACK POINTER IS INCORRECT"
3308  020162  051124  050101  020054
3309  020170  052123  041501  020113
3310  020176  047520  047111  042524
3311  020204  020122  051511  044440
3312  020212  041516  051117  042522
3313  020220  052103     000
3314  020223     101  052106  051105  EMS:    .ASCIZ  "AFTER TRAP, STORED PC IS INCORRECT"
3315  020230  052040  040522  026120
3316  020236  051440  047524  042522
3317  020244  020104  041520  044440
3318  020252  020123  047111  047503
3319  020260  051122  041505  000124
3320  020266  043101  042524  020122  EMT:    .ASCIZ  "AFTER TRAP, STORED PS IS INCORRECT"
3321  020274  051124  050101  020054
3322  020302  052123  051117  042105
3323  020310  050040  020123  051511
3324  020316  044440  041516  051117
3325  020324  042522  052103     000
3326  020331     101  052106  051105  EMU:    .ASCIZ  "AFTER TRAP, LOADED PS IS INCORRECT"
3327  020336  052040  040522  026120
3328  020344  046040  040517  042504
3329  020352  020104  051520  044440
3330  020360  020123  047111  047503
3331  020366  051122  041505  000124
3332
3333
3334
3335
3336                                  ;DATA HEADERS HERE
3337  020374  043040  051520     000  DHA:    .ASCIZ  " FPS"
3338  020401     040  042506  000103  DHB:    .ASCIZ  " FEC"
3339  020406  043040  040505     000  DHC:    .ASCIZ  " FEA"
3340  020413     040  050106  004523  DHD:    .ASCIZ  " FPS     R0"
3341  020420  020040  030122     000
3342  020425     040  051040  000060  DHE:    .ASCIZ  "  R0"
3343  020432  020040  030522  020011  DHF:    .ASCIZ  "  R1     SP"
3344  020440  051440  000120
3345  020444  051044  043505  000060  DHG:    .ASCIZ  "$REG0"
```

```
                020452  051044  043505  004460  DHH:    .ASCIZ  "$REG0  $REG1"
                020460  051044  043505  000061
                020466  051044  043505  004460  DHI:    .ASCIZ  "$REG0  $REG1   $REG2   $REG3"
                020474  051044  043505  004461
                020502  051044  043505  004462
                020510  051044  043505  000063
                020516  054105  023520  026504  DHJ:    .ASCIZ  "EXP'D-FPS-RCV'D"
                020524  050106  026523  041522
                020532  023526  000104
                020536  054105  023520  026504  DHK:    .ASCIZ  "EXP'D-FEC-RCV'D        EXP'D-FEA-RCV'D"
                020544  042506  026503  041522
                020552  023526  004504  054105
                020560  023520  026504  042506
                020566  026501  041522  023526
                020574  000104
                020576  054105  023520  004504  DHL:    .ASCIZ  "EXP'D  RCV'D"
                020604  041522  023526  000104
                020612  043040  051520  020011  DHM:    .ASCIZ  " FPS    FEC     FEA    OLD PS  OLD PC  SP AFTER"
                020620  042506  004503  043040
                020626  040505  047411  042114
                020634  050040  004523  046117
                020642  020104  041520  051411
                020650  020120  043101  042524
                020656  000122

                                                ;DATA VECTORS HERE
                                                .EVEN
                020660  001466  000000          DTB:    .WORD   FEC,0
                020664  001464                  DTD:    .WORD   FPS
                020666  001526  000000          DTE:    .WORD   EREG0,0
                020672  001530  001542  000000  DTF:    .WORD   EREG1,EREG6,0
                020700  001170  000000          DTG:    .WORD   $REG0,0
                020704  001170  001172  000000  DTH:    .WORD   $REG0,$REG1,0
                020712  001170  001172  001174  DTI:    .WORD   $REG0,$REG1,$REG2,$REG3,0
                020720  001176  000000
                020724  001220                  DTJ:    .WORD   $TMP4
                020726  001464  000000          DTA:    .WORD   FPS,0
                020732  001222  001466  001500  DTK:    .WORD   $TMPS,FEC,EXPFEA
                020740  001470  000000          DTC:    .WORD   FEA,0
                020744  001464  001466  001470  DTL:    .WORD   FPS,FEC,FEA,FPPOPS,FPPOPC,OSPRCV,0
                020752  001474  001472  001514
                020760  000000
                020762  001504  001514  000000  DTAG:   .WORD   EXPSP,OSPRCV,0
                020770  001502  001506  000000  DTAH:   .WORD   EXPRTS,OPCRCV,0
                020776  001214  001510  000000  DTAI:   .WORD   $TMP2,OPSRCV,0
                021004  001212  001512  000000  DTAJ:   .WORD   $TMP1,NPSLOD,0

                                                ;THE END
                000001                          .END
```

```
A0D     013134          2283#   2285
A0DDM   013422          2352#   2354
ABASE = 000000           286
ACDM1 = 000000           286
ACDM2 = 000000           286
ACPU0P= 000000           286     301
ADDM0 = 000000           286
ADDM1 = 000000           286
ADDM10= 000000           286
ADDM11= 000000           286
ADDM12= 000000           286
ADDM13= 000000           286
ADDM14= 000000           286
ADDM15= 000000           286
ADDM2 = 000000           286
ADDM3 = 000000           286
ADDM4 = 000000           286
ADDM5 = 000000           286
ADDM6 = 000000           286
ADDM7 = 000000           286
ADDM8 = 000000           286
ADDM9 = 000000           286
ADEVCT= 000000           286     292
ADEVM = 000000           286
A00101  001604           394#    771     1295
A01000  001610           396#    807      809
A01001  001606           395#
A0ST01  001760           423#
AENV  = 000000           286     297
AENVM = 000000           286     298
AFATAL= 000000           286     289
AMADR1= 000000           286
AMADR2= 000000           286
AMADR3= 000000           286
AMADR4= 000000           286
AMAMS1= 000000           286
AMAMS2= 000000           286
AMAMS3= 000000           286
AMAMS4= 000000           286
AMSGAD= 000000           286     294
AMSGLG= 000000           286     295
AMSGTY= 000000           286     288
AMTYP1= 000000           286
AMTYP2= 000000           286
AMTYP3= 000000           286
AMTYP4= 000000           286
APASS = 000000           286     291
APRIOR= 000000           286
APTCSU= 000040          2967    3072#
APTENV= 000001          2847    2960    3028    3070#
APTSIZ= 000200           490    3069#
APTSP0= 000100          2962    3030    3071#
ASCHOT  017414           603    3237#
ASCHRM  017422           605    3238#
ASWREG= 000000           286     299
ATESTN= 000000           286     290
```

```
AUNIT = 000000            286     293
AUSWR = 000000            286     300
AVECT1= 000000            286
AVECT2= 000000            286
ASREGD  001632            401#    850*   1202*   1215*   1471    1473
BGNMES  001772            429#    529
BIT0  = 000001            126#    616    2723
BIT00 = 000001            116#    126
BIT01 = 000002            115#    125
BIT02 = 000004            114#    124
BIT03 = 000010            113#    123
BIT04 = 000020            112#    122     535     585    1944    2196    2494
BIT05 = 000040            111#    121
BIT06 = 000100            110#    120
BIT07 = 000200            109#    119
BIT08 = 000400            108#    118    2774
BIT09 = 001000            107#    117    2782    2857
BIT1  = 000002            125#
BIT10 = 002000            106#   2835
BIT11 = 004000            105#   2789
BIT12 = 010000            104#    567     595     597     600    1948    2200    2500
BIT13 = 020000            103#   2842
BIT14 = 040000            102#   2578    2592    2760
BIT15 = 100000            101#    517    2697    2703
BIT2  = 000004            124#   2711
BIT3  = 000010            123#
BIT4  = 000020            122#
BIT5  = 000040            121#
BIT6  = 000100            120#    611     616    2698    2723
BIT7  = 000200            119#    507
BIT8  = 000400            118#
BIT9  = 001000            117#
BPTVEC= 000014            133#
CLKDON  003036            523     526#
CLKPRS  001516            369#    500*    507*    517*    609     614    2461    2465
CR    = 000015             41#    429     434     441    3006    3016
CRLF  = 000200             42#   2977    3016
DALTA   001634            402#   1208    1221
DALTB   001644            404#   1209    1524    1538
DALTC   001654            406#   1228
DBIG    001612            397#    885    1091    1774    1775
DOISP = 177570             4E#    248     478
DOIV    013242           2307#   2309
DHA     020374            328    3337#
DHB     020401            329    3338#
DHC     020406            330    3339#
DHD     020413            331    3340#
DHE     020425            332     338    3342#
DHF     020432            333    3343#
DHG     020444            337    3345#
DHH     020452            336    3346#
DHI     020466            335    3348#
DHJ     020516            347    3352#
DHK     020536            346    3355#
DHL     020576            342     343     344     345    3361#
DHM     020612            349    3363#
```

```
DISPLA  001146            248#    478*    486*   2804*   2834*
DISPRE  000174            179#    486
DKZERO  001622            399#    970    1070    1585    1606    1613    1726
DSMALL  001574            391#
DSWR  = 177570             47#    247     477
DTA     020726            328    3383#
DTAG    020762            342    3389#
DTAH    020770            343    3390#
DTAI    020776            344    3391#
DTAJ    021004            345    3392#
DTB     020660            329    3374#
DTC     020740            330    3385#
DTD     020664            331    3375#
DTE     020666            332     338    3376#
DTF     020672            333    3377#
DTG     020700            337    3378#
DTH     020704            336    3379#
DTI     020712            335    3380#
DTJ     020724            347    3382#
DTK     020732            346    3384#
DTL     020744            349    3386#
D0100   001724            416#   1378
D0100X  001736            419#   1368
D0101   001560            389#    394     750     758     761     764     767    1126    1313    1406    2410
D0111   001744            420#   1365    1373
D1000   001726            396     417#   1308    1360
D1001   001566            390#    395    1336
D1010   001556            388#    697     699     722     725     728     731     786     793     796     937     951     954
                         1084    1101    1296    1298    1315    1317    1334    1343    1355    1407    1409    2399    2402
D20     001714            414#   1544    1551
D37     001674            410#   1564
D40     001664            408#   1566    1584    1599
D46     001704            412#   1563
DST01   001754            422#    423    1487    1511    1515
EMA     017431            328     347    3242#
EMB     017455            329    3246#
EMC     017501            330    3250#
EMD     017525            331    3254#
EME     017557            332    3259#
EMF     017601            333    3263#
EMH     017623            335     336     337     338    3267#
EMI     017675            346    3275#
EMJ     017725            349    3280#
EMP     020001            334     340    3288#
EMQ     020067            341    3298#
EMR     020154            342    3307#
EMS     020223            343    3314#
EMT     020266            344    3320#
EMTVEC= 000030            136#    461#    462*
EMU     020331            345    3326#
EMV001  001274            328#
EMV002  001302            329#
EMV003  001310            330#
EMV004  001316            331#
EMV005  001324            332#
EMV006  001332            333#
```

```
EMV007  001340          334#
EMV010  001346          335#
EMV011  001354          336#
EMV012  001362          337#
EMV013  001370          338#
EMV014  001376          340#
EMV015  001404          341#
EMV016  001412          342#
EMV017  001420          343#
EMV020  001426          344#
EMV021  001434          345#
EMV022  001442          346#
EMV023  001450          347#
EMV024  001456          349#
EREG0   001526          376#   2823*   3376
EREG1   001530          377#   2824*   3377
EREG2   001532          378#   2825*
EREG3   001534          379#   2826*
EREG4   001536          380#   2827*
EREG5   001540          381#   2828*
EREG6   001542          382#   2829*   2830*   3377
EREG7   001544          383#   2831*
ERRVEC= 000004          129#    475     476*    487*    501     502     503*    504*    514*    526*    527*    2765    2766*
                       2768*   2771*
EXLOP   013554         2388#   2393
EXPFEA  001500          361#   1827*   1846*   1865*   1884*   1903*   1922*   1955*   1984*   2003*   2022*   2041*   2060*
                       2079*   2098*   2117*   2136*   2155*   2174*   2207*   2626*   2629    3384
EXPRTS  001502          362#   1828*   1847*   1866*   1885*   1904*   1923*   1956*   1985*   2004*   2023*   2042*   2061*
                       2080*   2099*   2118*   2137*   2156*   2175*   2208*   2603    3390
EXPSP   001504          363#   2596*   2597*   2598    3389
EX4     013612         2399#
FCONT1  014530         2579    2582    2618#
FEA     001470          357#   1594    1641    1667    1692    1716    1740    1764    1790    1816    2629    3385    3386
FEC     001466          356#   1587*   1591    1634*   1638    1660*   1664    1685*   1689    1709*   1713    1733*   1737
                       1757*   1761    1783*   1787    1809*   1813    2371*   2375    2619*   2627    2731*   3374    3384
                       3386
FEC02   014640         1825    1982    2647#
FEC04   014650         1844    2001    2652#
FEC06   014666         1863    2020    2039    2657#
FEC10   014704         1882    2058    2077    2662#
FEC12   014724         1901    2096    2115    2668#
FEC14   014742         1920    2134    2153    2673#
FEC16   014754         1953    2172    2205    2678#
FII02   014640         1827    1984    2648#
FII04   014654         1846    2003    2653#
FII06   014672         1865    2022    2041    2658#
FII10   014714         1884    2060    2079    2664#
FII12   014730         1903    2098    2117    2669#
FII14   014742         1922    2136    2155    2674#
FII16   014762         1955    2174    2207    2680#
FLPERR  014352         2563    2566#
FLTMZR  014776         2674    2686#
FLTONE  001546          386#   2435
FLTZER  014772         2653    2685#
FLTO01  015002         2652    2657    2662    2687#
FLTO02  015006         2663    2688#
```

```
FLTDO3  015012            2668    2690#
FLTDO4  015016            2669    2691#
FPPILT  015160             496#   2727#
FPPNTP  014366            2578#   2650    2655    2660    2666    2671    2676    2682
FPPN1   011302            1826    1835#
FPPN12  012076            2021    2030#
FPPN14  012216            2059    2068#
FPPN16  012336            2097    2106#
FPPN2   011352            1845    1854#
FPPN20  012456            2135    2144#
FPPN22  012576            2173    2182#
FPPN3   011422            1864    1873#
FPPN4   011472            1883    1892#
FPPN5   011542            1902    1911#
FPPN6   011612            1921    1930#
FPPN7   011706            1954    1963#
FPPOPC  001472             358#   2728*   2742    3386
FPPOPS  001474             359#   2729*   2741    3386
FPPOSP  001476             360#
FPPTRP  014402            2556    2585#
FPPT10  011756            1983    1992#
FPPT11  012026            2002    2011#
FPPT13  012146            2040    2049#
FPPT15  012266            2078    2087#
FPPT17  012406            2116    2125#
FPPT21  012526            2154    2163#
FPPT23  012672            2206    2215#
FPPVEC= 000244             160#    496*    497*   2554    2555    2556*   2557*   2636*   2637*
FPS     001464             355#    631*    632     637*    638     643*    644     649*    650     655*    656     661*    662
                           711#    712     741#    751*    752     776*    813*    831*    838*    845*    852*    858*    864*
                           874#    875     880#    881     886*    887     897*    898     903*    904     909*    910     917*
                           918     928*    929     941*    942     960*    961     972*    973     988*    989     1000*   1001
                          1016*   1017     1030*   1031    1049*   1050    1063*   1072*   1073    1086*   1087    1103*   1104
                          1128*   1129    1166*   1167    1178*   1179    1197*   1198    1210*   1211    1222*   1223    1239*
                          1240    1251*   1252    1268*   1269    1280*   1281    1302*   1303    1321*   1322    1337*   1345*
                          1389*   1390    1413*   1414    1432*   1444*   1459*   1460    1477*   1488*   1489    1505*   1506
                          1510*   1526*   1527    1545*   1546    1586*   1588    1600*   1607*   1608    1614*   1628*   1633*
                          1635    1654*   1659*   1661    1679*   1684*   1686    1703*   1708*   1710    1727*   1732*   1734
                          1751*   1756*   1758    1776*   1782*   1784    1803*   1808*   1810    2231*   2246*   2269*   2287*
                          2301*   2313*   2326*   2337*   2356*   2370*   2395*   2405*   2417*   2428*   2437*   2618*   2621
                          2730*   2732    2735    3375    3383    3386
FSTO1   001762             424#   1433    1494
GMS     = ****** U         178    3181    3182    3183    3184
HOTWRM  015766             603#    605#   2885#
HT      = 000011            39#   2975    3016
IKW11L  015022             509    2695#
IKW11P  015050             520    2701#
IOTVEC= 000020             134#    459*    460*
KW11LC  001520             371#    508*   2696*   2697*
KW11PC  001522             372#    518*   2702*   2703*
KW11PR  001524             373#    519*    522    2709*   2711    2717*   2718*   2720*   2722
LF      = 000012            40#    429     434     441    3010    3016
LKS     = 177546           151#    506*    611*   2463*   2695*   2698*
LKV     = 000100           152#    509*    510*
MED     = 076600           143#    534     584     588     598    1943    1947    2195    2199    2493    2501    2507    3213
MMUL    013176            2295#   2297
```

```
MORE    013712          2420#
NEWPAS  003220           538     540     565#    2548
NKWL1   003376           610     614#
NKWL2   014066          2462    2465#
NKWP1   003412           615     618#
NKWP2   014100          2466    2468#
NKW11L  002750           503     512#
NKW11P  003032           514     524#
NPSLOO  001512           366#    2585*   2612    3392
NWPAS1  002067           441#     570
NXTO2   014642          1828    1985    2649#
NXTO4   014660          1847    2004    2654#
NXTO6   014676          1866    2023    2042    2659#
NXT10   014716          1885    2061    2080    2665#
NXT12   014734          1904    2099    2118    2670#
NXT14   014746          1923    2137    2156    2675#
NXT16   014764          1956    2175    2208    2681#
OPCRCV  001506           364#    2586*   2603    3390
OPSRCV  001510           365#    2587*   2607    3391
OSPRCV  001514           367#    2588*   2598    2727*   3386    3389
PIRQ    = 177772          46#
PIRQVE= 000240           140#
PLKD    = 172544         156#
PLKR    = 172542         155#     522*   2722*
PLKS    = 172540         154#     516*    616*   2467*   2701*   2723*
PLKV    = 000104         157#     520*    521*
PR0     = 000000          63#
PR1     = 000040          64#
PR2     = 000100          65#
PR3     = 000140          66#
PR4     = 000200          67#
PR5     = 000240          68#
PR6     = 000300          69#     497     510     521
PR7     = 000340          70#     504
PS      = 177776          43#      44     671     679     687     1115    1438*   1440
PSW     = 177776          44#     618*   2568*   2585    2639*
PWRVEC= 000024          135#     465*    466*   3191*   3192*   3201*   3207*   3221*   3222*
RESVEC= 000010          130#
RFLAG   = 000144          147#     588    1947    2199    2501
RWHAMI= 000022          145#     534     584    1943    2195    2493
STACK   = 001100           34#     457     565     582
START   002400          182      450#   3226
STKLMT= 177774           45#
SUBFM   013342          2332#   2334
SUBPAS  003256           568     582#   2511
SUBT    013054          2265#   2267
SWR     001144           247#     455     477*    479     485*    492*    567     590     593     2497    2535    2760    2774
                        2782    2789    2835    2842    2854    2857    3199    3214*
SWREG   000176           180#     485
SW0     = 000001          98#
SW00    = 000001          88#      98     593
SW01    = 000002          87#      97     590    2497
SW02    = 000004          86#      96
SW03    = 000010          85#      95
SW04    = 000020          84#      94
SW05    = 000040          83#      93
```

```
SW06  = 000100            82#     92
SW07  = 000200            81#     91
SW08  = 000400            80#     90
SW09  = 001000            79#     89
SW1   = 000002            97#
SW10  = 002000            78#   2535
SW11  = 004000            77#
SW12  = 010000            76#
SW13  = 020000            75#
SW14  = 040000            74#
SW15  = 100000            73#
SW2   = 000004            96#
SW3   = 000010            95#
SW4   = 000020            94#
SW5   = 000040            93#
SW6   = 000100            92#
SW7   = 000200            91#
SW8   = 000400            90#
SW9   = 001000            89#
TBITVE= 000014           131#
TKVEC = 000060           138#
TKW11P  002754           511     514#
TPVEC = 000064           139#
TRAPVE= 000034           137#    463#    464#
TRPTST  014300          1830    1849    1868    1887    1906    1925    1958    1987    2006    2025    2044    2063    2082
                        2101    2120    2139    2158    2177    2210    2554#
TRTVEC= 000014           132#
TST1    003416           629#
TST10   004626           894#
TST11   004776           935#
TST12   005174           984#
TST13   005412          1044#
TST14   005624          1099#
TST15   006060          1162#
TST16   006172          1193#
TST17   006350          1235#
TST2    003564           668#
TST20   006462          1264#
TST21   006574          1293#
TST22   007026          1353#
TST23   007142          1386#
TST24   007410          1453#
TST25   007654          1522#
TST26   010066          1582#
TST27   010244          1622#
TST3    003670           695#
TST30   010350          1648#
TST31   010454          1674#
TST32   010554          1699#
TST33   010650          1723#
TST34   010744          1747#
TST35   011036          1771#
TST36   011142          1797#
TST37   011246          1824#
TST4    004066           748#
TST40   011316          1833    1843#
```

```
TST41   011366            1852    1862#
TST42   011436            1871    1881#
TST43   011506            1890    1900#
TST44   011556            1909    1919#
TST45   011626            1928    1938#
TST46   011722            1961    1981#
TST47   011772            1990    2000#
TST5    004216             783#
TST50   012042            2009    2019#
TST51   012112            2028    2038#
TST52   012162            2047    2057#
TST53   012232            2066    2076#
TST54   012302            2085    2095#
TST55   012352            2104    2114#
TST56   012422            2123    2133#
TST57   012472            2142    2152#
TST6    004342             820#
TST60   012542            2161    2171#
TST61   012612            2180    2190#
TST62   012706            2213    2223#
TST63   012752            2238#
TST64   013012            2253#
TST65   013106            2276#
TST66   013272            2319#
TST67   013372            2344#
TST7    004536             871#
TST70   013450            2364#
TST71   013530            2382#
TST72   014042            2454#
TYPE  = 104401             529     537     539     570     575    2537    2837    2845    2884    2903    2905    2906    2909
                          2911    2914    2917    2924    2928    2980    3133    3181#   3223
TYPOC = 104402            2893    2913    2916    2921    3182#
TYPON = 104404            3184#
TYPOS = 104403             573    3183#
WEIRD   001574             392#   1328
WFLAG = 000344             148#    598    2507
$APTHD  001000             204     210#
$ASTAT= ****** U          3050    3065
$ATYC   016506            3021    3023#
$ATY1   016462            3019#
$ATY3   016470            2965    3020#
$ATY4   016500            2850    3022#
$AUTOB  001140             243#
$BDAOR  001126             238#
$BDOAT  001132             240#
$BELL   001234             278#   2537    2837    2869
$CHARC  016456            2982#   2992#   2999    3008#   3013#
$CKSWR= ******  U         3187
$CMTAG  001100             225#    452     453     461     467     468     469
$CM1  = 000010             260#    261#    262#    263#    264#    265#    266#    267#    268#
$CM2  = 000020             260#    261#    262#    263#    264#    265#    266#    267#    268#
$CM3  = 000010             258#    260
$CMM  = 000010             268#    269#    270#    271#    272#    273#    274#    275#    276#
$CPUOP  001272             301#
$CRLF   001241             280#    575    2845    2869    2885    2905    2911    2928    2981    3016
$DEVCT  001254             292#
```

```
$DOAGN  014274          2531    2540    2548#
$ENDAD  014264           190    2543#   2864
$ENDCT  014234           467    2533#
$ENV    001264           297#   2847    2960    3028    3052
$ENVM   001265           298#    490    2962    2967    3030
$EOP    014176          2495    2498    2510    2523#
$EOPCT  014226           467*   2530#   2534
$ERFLG  001104           229#   2456*   2480*   2524*   2778    2780    2786*   2808    2832*   2869
$ERMAX  001120           235#    470*   2780    2803*   2808
$ERROR  015522           461    2822#
$ERRPC  001122           236#   2839*   2840*   2841    2869    2892    2931
$ERRTB  001274           312#   2900
$ERTTL  001114           233#   2838*   2869
$ESCAP  001232           277#    469*   2802*   2860    2862    2869
$ETABL  001264           296#
$ETEND  001274           216     308#
$FATAL  001246           289#   3056*
$FFLG   016726          3019*   3022*   3050    3059*   3067#
$FILLC  001164           256#   2985    3016
$FILLS  001163           255#   3016
$GDAOR  001124           237#
$GDOAT  001130           239#
$GET42  014254          2536    2539#
$GTSWR= ****** U        3186
$HD   = 000000            12
$HIBTS  001000           211#
$ICNT   001106           230#   2793*   2794    2796*   2807
$ILLUP  017376          3191    3207    3228#
$INTAG  001141           244#
$ITEMB  001116           234#   2841*   2849    2869    2889
$LF     001242           281#   2869    3016
$LFLG   016725          3060*   3066#
$LPAOR  001110           231#    471*   2784*   2800*   2805    2807
$LPERR  001112           232#    472*   2563*   2784    2801*   2807    2859
$LPTST  001150           249#   2776
$MAIL   001244           212     216     287#    489     2799    2847    2960
$MBAOR  001002           212#
$MFLG   016724          3020*   3026    3061*   3065#
$MSGAD  001260           294#   3036*   3039
$MSGLG  001262           295#   3041*
$MSGTY  001244           288#   3034    3042*   3054    3058*
$MXCNT  015520          2797    2807#
$NULL   001162           254#   2987    3016
$NWTST= 000001           626#    665#    692#    745#    780#    817#    868#    891#    932#    981#   1041#   1096#   1159#
                        1190#   1232#   1261#   1290#   1350#   1383#   1450#   1519#   1579#   1619#   1645#   1671#   1696#
                        1720#   1744#   1768#   1794#   1821#   1840#   1859#   1878#   1897#   1916#   1935#   1978#   1997#
                        2016#   2035#   2054#   2073#   2092#   2111#   2130#   2149#   2168#   2187#   2220#   2235#   2250#
                        2273#   2316#   2341#   2361#   2379#   2451#
$OCNT   017152          3105*   3134*   3147#
$OMODE  017154          3100*   3104*   3109    3112*   3123*   3149#
$OVER   015504          2761    2777    2785    2795    2804#
$PASS   001252           291#    489*    571    2527*   2528*   2791    2808
$PASTM  001006           214#
$POWER  017404          3224    3231#
$PWRAD  017372          3226#
$PWRON  017224           465    3191#   3221
```

```
$PWRMG  017366          3224#
$PWRUP  017276          3201    3207#
$QUES   001240           279#   2869    3016
$RDCHR= ****** U        3187
$RDOEC= ****** U        3187
$RDLIN= ****** U        3187
$RDOCT= ****** U        3187
$REGAD  001166           258#
$REGO   001170           260#    401     717     756     758     787     793     803     829*    836*    837     842     844
                         851     938*    940*    946     964*    976*    993*    999*   1005    1021*   1035*   1047*   1048*
                        1054    1064*   1077*   1090*   1108*   1114*   1121    1133*   1134    1142*   1143    1171*   1183*
                        1227*   1244*   1256*   1273*   1285*   1307*   1327*   1335*   1344*   1359*   1367*   1377*   1394*
                        1418*   1431*   1439*   1446    1464*   1478    1493*   1501    1511    1531*   1537*   1550*   1556*
                        1565*   1571*   1598*   1612*   1627*   1653*   1678*   1777*   1802*   2229*   2230    2244*   2245
                        2268*   2270    2280*   2288    2299*   2311*   2322    2325    2335*   2336    2355*   2369*   2372
                        2394*   2396    2400    2406    2415*   2416    2427*   2429    2438*   2439    2443*   2444    2658*
                        3378    3379    3380
$REG1   001172           261#    761     796     1009    1058    1138    1346    1482    1515    3379    3380
$REG2   001174           262#    764     789     951     1499    3380
$REG3   001176           263#    767     954     3380
$REG4   001200           264#    801     2254    2256*   2265    2286*   2404
$REG5   001202           265#
$REG6   001204           266#
$REG7   001206           267#
$R2A  = ****** U        3187
$SAVRE= ****** U        3187
$SAVR6  017402          3200*   3208    3209*   3210*   3230#
$SCOPE  015244           459    2758#
$SETUP= 000037           444#    458     459     461     463     465     467     468     469     471     2759    2823    2857
                        2864
$STUP = 177777           444#
$SVLAD  015450          2769    2798#
$SVPC = 000204           188#    193
$SWR  = 167400             1#     12     276     277     278     468     469     471     472     630     669     696     749
                         784     821     872     895     936     985    1045    1100    1163    1194    1236    1265    1294
                        1354    1387    1454    1523    1583    1623    1649    1675    1700    1724    1748    1772    1798
                        1825    1844    1863    1882    1901    1920    1939    1982    2001    2020    2039    2058    2077
                        2096    2115    2134    2153    2172    2191    2224    2239    2254    2277    2320    2345    2365
                        2383    2455    2750    2751    2752    2753    2754    2760    2772    2774    2775    2778    2779
                        2780    2787    2788    2789    2801    2804    2807    2814    2815    2816    2817    2818    2835
                        2842    2854    2857    2869    3227
$SWREG  001266           299#    492
$SWRMK= 000000          2754    2776
$TESTN  001250           290#   2799*   2930
$TIMES  001230           276#    468*   2455*   2479*   2526*   2787*   2794    2797*   2807
$TKB    001154           251#
$TKS    001152           250#
$TMP0   001210           268#   2560
$TMP1   001212           269#   3392
$TMP2   001214           270#   3391
$TMP3   001216           271#
$TMP4   001220           272#   3382
$TMP5   001222           273#   3384
$TMP6   001224           274#
$TMP7   001226           275#
$TN   = 000073             1#     12     626     630#    665     669#    692     696#    745     749#    780     784#    817
```

```
CMPFLT      1#
COMMEN    141#
COMM00      1#
COMM01      1#
COMM02      1#
COMM03      1#
COMM04      1#
COMM05      1#
COMM06      1#
COMM07      1#
COMM1       1#
COMM10      1#
COMM11      1#
COMM12      1#
COMM13      1#
COMM14      1#
COMM15      1#
COMM16      1#
COMM17      1#
COMM2       1#
COMM20      1#
COMM21      1#
COMM22      1#
COMM23      1#
COMM24      1#
COMM25      1#
COMM26      1#
COMM27      1#
COMM3       1#
COMM30      1#
COMM31      1#
COMM32      1#
COMM33      1#
COMM34      1#
COMM35      1#
COMM36      1#
COMM37      1#
COMM4       1#
COMM40      1#
COMM41      1#
COMM42      1#
COMM43      1#
COMM44      1#
COMM45      1#
COMM46      1#
COMM47      1#
ENDCOM    141#
ERRCMP      :   2598  2603  2607  2612
ERRLUR      1#  2868
ERROR      35#   634   640   646   652   658   664   675   683   691   701   709   714   724   727
              730   733   740   744   754   760   763   766   769   779   791   795   798   805   811
              816   834   841   848   855   861   867   877   883   889   900   906   912   916   920
              925   926   927   931   944   949   953   956   963   968   975   980   991   997  1003
             1007  1011  1019  1025  1033  1039  1052  1056  1060  1068  1075  1081  1089  1094  1106
             1112  1119  1123  1131  1136  1140  1145  1151  1157  1169  1175  1181  1187  1200  1206
             1213  1219  1225  1231  1242  1248  1254  1260  1271  1277  1283  1289  1300  1305  1311
```

```
SETPRI    141#
SETREG      1#  2823
SETTRA   3173#  3182  3183  3184
SETUP       1#   141#   450
SKIP      141#  1832  1851  1870  1889  1908  1927  1960  1989  2008  2027  2046  2065  2084  2103
         2122  2141  2160  2179  2212
SLASH     141#   531   558  1940  1951  1969  1976  2192  2203
SPACE     141#
STARS     141#   186   197   199   206   220   282   285   446   448   561   563   578   580   622
          624   626   628   665   667   692   694   745   747   780   782   817   819   868   870
          891   893   932   934   981   983  1041  1043  1096  1098  1159  1161  1190  1192  1232
         1234  1261  1263  1290  1292  1350  1352  1383  1385  1450  1452  1519  1521  1579  1581
         1619  1621  1645  1647  1671  1673  1696  1698  1720  1722  1744  1746  1768  1770  1794
         1796  1821  1823  1840  1842  1859  1861  1878  1880  1897  1899  1916  1918  1935  1937
         1978  1980  1997  1999  2016  2018  2035  2037  2054  2056  2073  2075  2092  2094  2111
         2113  2130  2132  2149  2151  2168  2170  2187  2189  2220  2222  2235  2237  2250  2252
         2273  2275    23    2318  2341  2343  2361  2363  2379  2381  2451  2453  2471  2473  2475
         2513  2747    28.   2869  2939  3018  3075  3152  3189  3205
STATUS      1#   637    64.    649   655   661   711   751   874   880   941   960   972   988  1000
         1016  1030  1049  1072  1086  1103  1128  1166  1178  1197  1210  1222  1239  1251  1268
         1280  1302  1321  1389  1413  1459  1488  1505  1526  1545  1586  1607  1632  1658  1683
         1707  1731  1755  1781  1807
SWRSU     141#   473#
TA0001      1#
TA0002      1#
TA00F1      1#
TA00F2      1#
TA00R1      1#
TA00R2      1#
TRMTRP   3173#
TYPBIN    141#
TYPDEC    141#
TYPNAM    141#
TYPNUM    141#
TYPOCS    141#
TYPOCT    141#
TYPTXT    141#
UPCODE      1#  3212
$$CMRE    218#   260   261   262   263   264   265   266   267
$$CMTM    218#   268   269   270   271   272   273   274   275
$$ESCA    141#
$$NEWT    141#   626   665   692   745   780   817   868   891   932   981  1041  1096  1159  1190
         1232  1261  1290  1350  1383  1450  1519  1579  1619  1645  1671  1696  1720  1744  1768
         1794  1821  1840  1859  1878  1897  1916  1935  1978  1997  2016  2035  2054  2073  2092
         2111  2130  2149  2168  2187  2220  2235  2250  2273  2316  2341  2361  2379  2451
$$SET    3173#  3182  3183  3184
$$SETM    489#
$$SKIP    141#  1833  1852  1871  1890  1909  1928  1961  1990  2009  2028  2047  2066  2085  2104
         2123  2142  2161  2180  2213
.EQUAT      1#    31
.HEADE      1#     2
.SBPAS      1#  2475
.SETUP      1#   444
.STPAS      1#   531
.$ACT1      1#   184
.$APTB      1#   283#
```

DQFPCB.P11 04-MAY-77 19:34 CROSS REFERENCE TABLE -- MACRO NAMES

```
.$APTH      1#    195
.$APTY      1#   3016
.$CATC      1#    172
.$CMTA      1#    218
.$EOP       1#   2513
.$ERRO      1#   2808
.$POWE      1#   3187
.$SCOP      1#   2745
.$TRAP      1#   3150
.$TYER      1#   2869
.$TYPE      1#   2937
.$TYPO      1#   3073
```

```
. ABS.  021012     000
```

```
 ERRORS DETECTED:  0
 DEFAULT GLOBALS GENERATED:  0

 DQFPCB.BIN,DQFPCB.LST/CRF/SOL/P/DOC/CPU:70/EX/EN:WRP/NL:TTM=DQFPCB.MAC,DQFPCB.P11
 RUN-TIME: 19 14 1 SECONDS
 RUN-TIME RATIO: 68/35=1.9
 CORE USED:  24K  (48 PAGES)

 DOCUMENT PAGES:  99
 WRAP-AROUND:  0%

 USER SYMBOLS:  540
 MACRO  NAMES:  143
 UNDF SYMBOLS:  14
 DISK BLOCKS  READ:  984
 DISK BLKS WRITTEN:  428
 KILO CORE SECONDS:  1248
```