DataGeneral

TECHNICAL STATEMENT

TEXT LISTING

068-001073-01

PROGRAM

ARRAY PROCESSOR EXERCISER-I

TEXT TAPE

097-001073-01

ABSTRACT

THESE PROGRAMS EXERCISE THE A.P. INSTRUCTIONS REAL RECURSIVE FILTER AND COMPLEX RECURSIVE FILTER. THE PROGRAM IS DESIGNED TO AID IN THE INITIAL CHECKOUT, AND MAINTENANCE OF THE ARRAY PROCESSOR. IT CAN ALSO BE CONSIDERED AN EXERCISER FOR P. M. PURPOSES.

```
 0001 .MAIN    MACRO REV 06.30          09:47:39 09/19/79
01                      ;*****************************************************************
02                      ;
03                      ; NAME: API.TX                    PART NUMBER 097-1073
04                      ;
05                      ;
06                      ; DESCRIPTION: TEST COMPLEX RECURSIVE FILTER INSTRUCTION.
07                      ;
08                      ;
09                      ; REVISION HISTORY:
10                      ;
11                      ;       REV.            DATE
12                      ;
13                      ;       00              03/29/78
14                      ;       01              12/29/78
15                      ;
16                      ;
17                      ; COPYRIGHT (C) DATA GENERAL CORPORATION, 1978
18                      ; ALL RIGHTS RESERVED.
19                      ;*****************************************************************
```

```
10002 .MAIN
01                      ;
02                      ;               API.TX          PART NUMBER:097-1073
03                      ;
04                      ;1.0    PROGRAM NAME
05                      ;
06                      ;       SOURCE FILE:    API.SR
07                      ;       DTOS FILE:      API EXER
08                      ;
09                      ;2.0    REVISION HISTORY
10                      ;
11                      ;       DATE            REVISION
12                      ;       ----            --------
13                      ;       XX/XX/78           00
14                      ;
15                      ;3.0    MACHINE REQUIREMENTS
16                      ;
17                      ;       1. ECLIPSE FAMILY CENTRAL PROCESSOR  (HOST OR IOP)
18                      ;          WITH AT LEAST 16.K READ/WRITE MEMORY
19                      ;       2. ARRAY PROCESSOR BOARDS AP1, AP2, AP3
20                      ;       3. BASIC I/O TELETYPE INTERFACE AND CONTROL
21                      ;
22                      ;4.0    TEST REQUIREMENTS
23                      ;
24                      ;       1. SAME AS MACHINE REQUIREMENTS
25                      ;       2. I/O TESTER OR DISC
26                      ;       3. MMPU1
27                      ;
28                      ;5. SUMMARY.
29                      ;   --------
30                      ;
31                      ;   THESE PROGRAMS EXERCISE THE A.P. INSTRUCTIONS
32                      ;   REAL RECURSIVE FILTER AND COMPLEX RECURSIVE FILTER.
33                      ;
34                      ;   THE PROGRAM IS DESIGNED TO AID IN THE INITIAL
35                      ;   CHECKOUT, AND MAINTENANCE OF THE ARRAY PROCESSOR
36                      ;   IT CAN ALSO BE CONSIDERED AN EXERCISER FOR P.M.
37                      ;   PURPOSES.
38                      ;
39                      ;6.0    RESTRICTIONS
40                      ;
41                      ;       THE PROGRAM ASSUMES
42                      ;       - THE SYSTEM EXCLUDING THE AP IS ERROR FREE
43                      ;       - THE HOST COMPUTER AND MEMORY/MAP SYSTEMS
44                      ;         ARE WORKING PROPERLY.
45                      ;       - THE 2 AP MAINTAINANCE DIAGNOSTICS USING
46                      ;         THE MAINTAINANCE INSTRUCTION SET (MIS)
47                      ;         HAVE BEEN SUCCESSFULLY RUN.
```

```
10003 .MAIN
01              ;7. PROGRAM DESCRIPTION/THEORY OF OPERATION.
02              ;   ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
03
04              ;   SPECIFICLY THIS PROGRAM EXERCISES THE HARDWARE AND
05              ;   MICRO-CODE THAT IMPLEMENT THE ARRAY PROCESSOR'S RECURSIVE
06              ;   FILTER INSTRUCTIONS. THE PROGRAM FILLS THE ROLES OF BOTH
07              ;   DIAGNOSTIC AND EXERCISER, FOR THESE INSTRUCTIONS.
08
09              ;   THIS DUAL ROLE REQUIRES A PROGRAM WHICH CAN EXECUTE
10              ;   VERY SIMPLE TESTS AND BUILD GRADUALLY TO GENERAL,
11              ;   RANDOM TESTS. THE FORMER HELPS IN BRINGING UP THE SYSTEM
12              ;   INITIALLY, WHILE PUMPING LOTS OF RANDOM NUMBERS THROUGH
13              ;   UNDER RANDOM CONDITIONS HELPS TO UNCOVER MORE SUBTLE
14              ;   FAULTS THAT MIGHT NOT BE EASY TO FORSEE.
15
16              ;   THE FILTER INSTRUCTIONS, AS THEIR NAMES IMPLY ACCEPT
17              ;   A VECTOR AS INPUT AND PRODUCE A VECTOR AS OUTPUT.
18              ;   THESE VECTORS ARE ALL SITUATED IN A.P. RAM. THE
19              ;   RECURSIVE FILTER UTILIZES THE PAST STATE OF THE INPUT
20              ;   AND OUTPUT VECTORS,SUMMED WITH THE CURRENT STATE OF
21              ;   THE INPUT TO DETERMINE THE NEW OUTPUT, HENCE THE TERM
22              ;   RECURSIVE. THE RELATIONSHIPS BETWEEN THESE ITEMS ARE
23              ;   DETERMINED BY A SET OF CONSTANTS, REFERRED TO IN THE
24              ;   PROGRAM AS "COEFFICIENTS". THE INPUT VECTOR HAS A
25              ;   CHARACTERISTIC LENGTH, REFERRED TO IN THE PROGRAM AS "N".
26              ;   THE OUTPUT VECTOR ALSO HAS A CHARACTERISTIC LENGTH
27              ;   WHICH IS RELATED TO THE INPUT VECTOR BY A CONSTANT CALLED
28              ;   THE "DESAMPLE RATE."IT IS CALLED 'D' IN THE PROGRAM. THE
29              ;   INPUT VECTOR IS CALLED THE 'X' VECTOR AND THE OUTPUT VECTOR
30              ;   IS CALLED THE 'Z' VECTOR.
31
32              ;   ALL OF THESE PARAMATERS MUST BE ESTABLISHED BY THE
33              ;   PROGRAM BEFORE THE INSTRUCTION CAN BE EXECUTED.
34              ;   IN PRACTICE A TEST IS BROKEN INTO TWO COMPONENTS.
35              ;   THE FIRST COMPONENT GENERATES ALL THE DATA NECCESARY
36              ;   TO DRIVE THE A.P. AT THE END OF THIS SECTION THE
37              ;   SIMULATER IS CALLED TO GENERATE THE ANSWER TO THE
38              ;   TEST CASE.
39
40              ;   IN THE SECOND HALF OF THE TEST, STARTING WITH THE SETP1
41              ;   CALL, THE PARAMETERS ARE LOADED INTO THE A.P. AND THE
42              ;   INSTRUCTION UNDER TEST IS EXECUTED. THE A.P.'S RESULTS
43              ;   ARE CHECKED WITH THE SIMULATORS AND ANY ERRORS THAT SHOW
44              ;   UP ARE REPORTED.
45
46              ;   IF THERE WERE NO ERRORS THE PROGRAM RETURNS TO THE
47              ;   SETP1 CALL, LOADS THE DATA AGAIN, EXECUTES THE INSTRUCTION
48              ;   AGAIN ETC. THIS LOOP IS REPEATED 5 TIMES. IF THERE
49              ;   STILL ARE NO ERRORS THE PROGRAM CONTINUES TO THE
50              ;   NEXT TEST.
51
52              ;   IF AN ERROR OCCURS IN ANY PART OF THE LOOP, IT IS
53              ;   REPORTED AND THEN THE PROGRAM STAYS IN THE SETP1/
54              ;   LOOP LOOP. THE OPERATOR MAY FORCE THE PROGRAM OUT
55              ;   OF THE LOOP OR GATHER MORE DATA THROUGH THE SWITCH
56              ;   FUNCTIONS (SEE SWITCH PACK INFO)
```

```
10004 .MAIN
01              ;   NOTE THAT THE PARAMATERS ARE SETUP IN THE FIRST
02              ;   PART OF THE TEST AND THAT THEY ARE NOT CHANGED IN THE
03              ;   SECOND HALF. THE TEST LOOP ONLY LOADS THE A.P. IT DOES
04              ;   NOT GENERATE ANY NEW DATA. THIS INSURES THAT IN THE EVENT
05              ;   OF AN ERROR THE TEST IS LOOPING USING THE DATA THAT
06              ;   CAUSED THE FAILURE.
07
08              ;   THIS TEST FORMAT IS USED THROUGHOUT, THOUGH DIFFERENT
09              ;   PARAMATER ROUTINES ARE USED TO MAKE THE TESTS PROGRESSIVLY
10              ;   MORE COMPLICATED.
11
12              ;   AS THE MICRO-CODE FOR EACH TYPE OF FILTER IS RELATIVLY
13              ;   INDEPENDENT OF THE OTHERS, EACH TYPE IS RUN THROUGH
14              ;   THE COMPLETE RANGE OF TESTS INDEPENDENTLY. THUS WE TEST
15              ;   ONE POLE, NO ZERO FILTERS COMPLETLY, THEN ONE POLE,
16              ;   ONE ZERO, ONE POLE, TWO ZERO, TWO POLE, NO ZERO, AND
17              ;   SO ON. EACH TYPE IS PUT THROUGH A SEQUENCE OF 12 TESTS
18              ;   BEFORE PROCEEDING ON TO THE NEXT TYPE.
19
20              ;   IN THE FIRST TEST SO CALLED 'MATCHED' COEFFICIENTS
21              ;   ARE USED, WITH AN IMPULSE FUNCTION AS INPUT.
22              ;   THE VECTORS ARE OF RANDOM LENGTH, BUT DESAMPLING IS
23              ;   SET TO ONE SO THE OUTPUT VECTOR IS THE SAME LENGTH
24              ;   AS THE INPUT VECTOR. THE RAM ADDRESSES IN ALL TESTS
25              ;   ARE CHOSEN AT RANDOM, THOUGH WITHIN BOUNDS CHOSEN
26              ;   TO GUARANTEE SEPARATION OF THE VECTORS. THE INPUT VECTOR
27              ;   MAY BE EITHER ABOVE OR BELOW THE OUTPUT VECTOR.
28
29              ;   MATCHED COEFFICIENTS MEANS THAT THE A1 COEFFICIENT =
30              ;   -B1 COEFFICIENT AND THE A2 COEFFICIENT= -B2 COEFFICIENT
31              ;   THOUGH THE B1&B2 COEFFICIENTS WILL BE RANDOM #S.
32              ;   CHOOSING THESE COEFFICIENTS IN THIS FASHION MEANS THAT
33              ;   FOR THE 1POLE 1ZERO & 2POLE 2ZERO FILTERS THE OUTPUT
34              ;   WILL BE IDENTICAL WITH THE INPUT FUNCTION. FOR
35              ;   NONSYMECTRICAL FILTERS ONLY THE ZEROTH ELEMENT WILL
36              ;   BE THE SAME. THIS PROVIDES A CHECK ON THE SIMULATION
37              ;   TO GUARANTEE OUR 'CORRECT' ANSWERS ARE REALLY CORRECT!
38
39              ;   THE IMPULSE FUNCTION IS LIKEWISE CHOSEN FOR ITS
40              ;   SIMPLICITY. THE ENTIRE VECTOR IS SET TO ZERO, AND
41              ;   THEN THE ZEROTH ELEMENT IS SET TO A RANDOM VALUE.
42
43              ;   IN THE SECOND TEST INSTEAD OF USING MATCHED COEFFICIENTS
44              ;   EACH COEFFICIENT IS CHOSEN TO BE A RANDOM #, BUT THAT #
45              ;   IS BOUNDED TO MAKE IT LESS THAN ONE. THIS IS DONE TO
46              ;   PROMOTE UNDERFLOWS.
47
48              ;   LIKEWISE IN THE THIRD TEST THE COEFFICIENTS
49              ;   ARE CHOSEN FROM UNBOUNDED RANDOM NUMBERS TO
50              ;   PROMOTE OVERFLOWS.
51
52              ;   TESTS 4,5,&6 IN THE SEQUENCE REPEAT THE ABOVE
53              ;   SET ONLY USING A TOTALLY RANDOM VECTOR AS INPUT.
54              ;   LASTLY THE ENTIRE SET SO FAR IS RUN AGAIN WITH
55              ;   "D" SET TO A RANDOM INTEGER BETWEEN 1 & 32.
56
57              ;   THE PARAMETERS FOR A GIVEN TEST CAN BE DETER-
58              ;   MINED FROM THE LINE AT THE TOP OF THE PAGE
```

```
10005 .MAIN
01                      S?WPD 8
02
03
04              ;8.     SWITCH SETTINGS
05              ;
06              ;       LOCATION "SWREG" IS USED TO SELECT THE PROGRAM OPTIONS
07              ;       (NOT SYSTEM CONFIGURATION). WHILE RUNNING UNDER DTOS,
08              ;       THIS LOCATION WILL BE LOADED BY THE MONITOR.
09              ;       HOWEVER UNDER STAND ALONE AND PROGRAM LOAD MODES THIS
10              ;       LOCATION WILL BE SET ACCORDING TO THE ANSWERS SUPPLIED
11              ;       BY THE OPERATOR. IN ANY CASE THE OPTIONS CAN BE CHANGED
12              ;       OR VERIFIED BY USING ONE OF THE COMMANDS GIVEN IN SEC.
13              ;       8.2
14              ;
15              ;
16              ;8.1    SWITCH OPTIONS
17              ;       DIFFERENT BITS AND THEIR INTERPRETATION AT LOCATION
18              ;       "SWREG" IS AS FOLLOWS:
19              ;
20              ;       BIT     OCTAL   BINARY  INERPRETATION
21              ;               VALUE   VALUE
22              ;
23              ;       1               0       LOOP ON ERROR
24              ;               40000   1       SKIP LOOPING ON ERROR
25              ;
26              ;       2               0       PRINT TO CONSOLE
27              ;               20000   1       ABORT PRINT OUT TO CONSOLE
28              ;
29              ;       3               0       DO NOT PRINT % FAILURE
30              ;               10000   1       PRINT % FAILURE
31              ;
32              ;       4               0       ALLOW END OF PASS PRINT OUT
33              ;               04000   1       SUPPRESS END OF PASS PRINT OUT
34              ;
35              ;       5               0       DO NOT PRINT ON THE LINE PRINTER
36              ;               02000   1       PRINT ON THE LINE PRINTER
37              ;
38              ;       6               0       DO NOT HALT ON ERROR
39              ;               01000   1       HALT ON ERROR
40              ;
41              ;       7               0       DO NOT PRINT SUMMARY AND/OR
42              ;                               PASSING OF EACH SUBTEST
43              ;               00400   1       PRINT SUMMARY AND/OR
44              ;                               PASSING OF EACH SUBTEST
45              ;       8               0       PRINT ONLY THE FIRST ERROR
46              ;               00200   1       PRINT EVERY ERROR
47
48              ;8.2    SWITCH COMMANDS
49              ;       ONCE THE PROGRAM STARTS EXECUTING THE STATE OF ANY OF
50              ;       THE BITS CAN BE CHANGED BY HITTING KEYS 1-9, A-F. THE
51              ;       PROGRAM WILL CONTINUE RUNNING AFTER UPDATING THE OPTIONS.
52              ;       EACH KEY WILL COMPLEMENT THE STATE OF THE BIT AFFILIAT-
53              ;       ED WITH IT, THUS BIT 4 CAN BE ALTERED BY HITTING KEY 4.
54              ;       SETTING OF ANY BIT OF LOCATION "SWREG" WILL SET BIT 0.
55              ;       (DEFAULT MODE IS DEFINED AS ALL BITS OF SWREG SET TO 0)
56              ;       THE PROGRAM CAN BE LOCKED INTO SWITCH MODIFICATION MODE
57              ;       BY TYPING A 0, IN WHICH CASE MORE THAN ONE BIT CAN BE
58              ;       CHANGED BEFORE  CONTROL IS ALLOWED TO RETURN TO THE
59              ;       MAIN PROGRAM.
60              ;
```

```
 0006 .MAIN
01              ;8.2.1  OTHER COMMANDS
02              ;
03              ;       "CR"    A "RETURN" CAN BE TYPED TO CONTINUE THE PROGRA'
04              ;               AFTER ITS LOCKED IN A SWITCH MODIFICATION MODE
05              ;
06              ;       ^D      THIS COMMAND GIVEN AT ANY TIME WILL RESET "SWREG'
07              ;               TO DEFAULT MODE AND RESTART THE PROGRAM.
08              ;
09              ;       ^R      THIS COMMAND GIVEN AT ANY TIME WILL RESTART THE
10              ;               PROGRAM. SWITCHES ARE LEFT WITH THE VALUES THEY
11              ;               HAD BEFORE THE COMMAND WAS ISSUED.
12              ;
13              ;       ^O      THIS COMMAND GIVEN AT ANY TIME WILL CAUSE THE
14              ;               PROGRAM CONTROL TO GO TO ODT (NOTE: THIS IS AN
15              ;               OPTIONAL COMMAND AND IS AVAILBLE ONLY IF
16              ;               ODTPK IS PRESENT)
17              ;
18              ;       M       THIS COMMAND GIVEN AT ANY TIME WILL PRINT THE
19              ;               CURRENT OPERATING MODES.
20              ;
21
22              ;
```

```
10007 .MAIN
01              ;9.0    OPERATING PROCEDURE/OPERATOR INPUT
02              ;
03              ;       9.1     OPERATING PROCEDURE (STAND ALONE MODE)
04              ;               1. LOAD THE PROGRAM IN THE COMPUTER
05              ;                  USING THE BINARY LOADER.
06              ;               2. SET SWITCHES TO ONE OF THE FOLLOWING
07              ;                  STARTING ADDRESSES:
08              ;                                       200 OR 500
09              ;                  TO RUN ALL TESTS.
10              ;               3. PRESS START.
11              ;               4. THE PROGRAM PRINTS "PASS"
12              ;                  AFTER SUCCESSFUL COMPLETION
13              ;                  OF THE TESTS.
14              ;
15              ;       9.2     OPERATING PROCEDURE (UNDER DTOS)
16              ;               1. SET PROGRAM OPERATING MODE WITH DTOS
17              ;                  "SWREG" COMMAND. (SEE SWITCH PACK INFO).
18              ;               2. LOAD THE DESIRED PROGRAMS WITH A RUNALL,
19              ;                  SELECT OR LOAD, COMMAND.
20              ;               3. THE PROGRAM PRINTS "PASS"
21              ;                  AFTER THE SUCCESSFUL COMPLETION OF THE TESTS.
22              ;
```

```
10008 .MAIN
01              ;10. PROGRAM ERROR DESCRIPTION
02              ;    ------------------------
03              ;
04              ;   EXAMPLE ERROR PRINTOUT FOR REAL FILTER TEST.
05              ;
06              ;   ERROR AT LOC. 002112     IN PASS 127
07              ;
08              ;   CPU STATE:
09              ;   CRY     AC0     AC1     AC2     AC3
10              ;   0       177777 144000 012066 002113
11              ;
12              ;   POLES   ZEROS   DESAMP X LEN   Z LEN
13              ;   1       2       12      3103    240
14              ;
15              ;   COEFFICIENTS:
16              ;   B1:  0  2B  D158DE
17              ;   B2:  0  00  000000
18              ;   A1:  1  70  D128CD
19              ;   A2:  1  47  86A69E
20              ;
21              ;   X ADDR= 133     64266      Z ADDR= 3412    73024
22              ;
23              ;   INDEX    DATA FROM AP   DATA EXPECTED
24              ;   -2       0 0D 673CB9    0 0D 673CB9
25              ;   -1       0 7F FFFFFF    1 2D 291F37
26              ;   0        1 18 21A0C2    0 00 000000
27              ;
28              ;
29              ;   THIS IS THE OUTPUT FOR RECURSIVE FILTERS. THE CPU STATE
30              ;   IS CAPTURED JUST AFTER THE FILTER INSTRUCTION IS EXECUTED
31              ;   SO AC2 POINTS TO THE A.P. PARAMATER BLOCK LOCATION. ALL
32              ;   NUMBERS NOT IN F.P. FORMAT ARE IN OCTAL. FLOATING POINT
33              ;   NUMBERS ARE EXPRESSED IN HEX AS FOLLOWS. <SIGN><HEXPONENT>
34              ;   <MANTISSA> THE DATA GIVEN IN "X ADDRESS" AND "Z ADDRESS" IS
35              ;   THE RAM ADDRESS (IN 2 WORD MODE) AND THE ACTUAL
36              ;   MEMORY ADDRESS OF THE START OF THE VECTOR. IN
37              ;   ORDER TO ACCESS A VECTOR ELEMENT WITH THE DEBUGGER
38              ;   FOR EXAMPLE, TAKE THE VECTOR INDEX FOR THE ELEMENT,
39              ;   MULTIPLY BY 2 FOR REAL & 4 FOR COMPLEX (REMEMBER
40              ;   THESE ARE OCTAL #S) AND ADD THE RESULT
41              ;   TO THE MEMORY ADDRESS OF THE START OF THE VECTOR.
```

```
!0009 .MAIN
01              :EXAMPLE ERROR OUTPUT FOR COMPLEX FILTER
02
03              ;   ERROR AT LOC. 00553    DURING PASS 1
04              ;
05              ;   CPU STATE:
06              ;   CRY     AC0     AC1     AC2     AC3
07              ;   1       000554 000000 012145 000554
08              ;
09              ;   POLES   ZEROS   DESAMP X LEN   Z LEN
10              ;   1       0       1       465     465
11              ;
12              ;   COEFFICIENTS:
13              ;   B1: 0 04 7D831E       0 4F B7FE10
14              ;   B2: 0 00 000000       0 00 000000
15              ;   A1: 0 00 000000       0 00 000000
16              ;   A2: 0 00 000000       0 00 000000
17              ;
18              ;   X ADDR= 1124    70520       Z ADDR= 404    66020
19              ;
20              ;   INDEX   DATA FROM AP                DATA FROM SIM
21              ;   -2      0 00 000000   0 00 000000 0 00 000000    0 00 000000
22              ;   -1      0 00 000000   0 00 000000 0 00 000000    0 00 000000
23              ;   0       0 00 000000   0 00 000000 1 1C CD8739    1 69 6C2A03
24
25
26              ;   THE ADDRESS OF THE INCORRECT VECTOR ELEMENT IS
27              ;   FOLLOWED BY THE VALUE OF THE ELEMENT IN ERROR
28              ;   AS WELL AS THE TWO ELEMENTS PRECEDING THE BAD
29              ;   ELEMENT. THE ELEMENTS ARE PRINTED IN ORDER
30              ;   BY THEIR INDEX, THE FIRST NUMBER IS THE ELEMENTS
31              ;   INDEX # IN OCTAL. THE RESULT FROM THE A.P. FOLLOWS
32              ;   WITH THE REAL DATA FIRST, THEN IN THE CASE OF A
33              ;   COMPLEX FILTER, THE IMAGINARY COMPONENT. THIS
34              ;   SAME FORMAT IS MAINTAINED FOR THE DATA EXPECTED
35              ;   PRINTOUT.
36
37              ;   ELEMENTS OF THE RESULT AFTER THE FAULTY ELEMENT ARE
38              ;   NOT CHECKED, BECAUSE THE RECURSIVE NATURE OF THE
39              ;   INSTRUCTION, (PAST OUTPUTS ARE USED TO GENERATE FUTURE INPUTS),
40              ;   PRETTY WELL GUARANTEES THE OUTPUTS WILL BE BAD.
41
42              ;   IT SHOULD BE NOTED THAT IN THIS OUTPUT THE FIRST TWO
43              ;   ELEMENTS PRINTED WILL MATCH THE EXPECTED INFO PRODUCED
44              ;   BY THE SIMULATOR. THESE DATA POINTS ARE PROVIDED TO
45              ;   INDICATE THE TREND THE OUTPUT WAS FOLLOWING AT THE TIME
46              ;   OF THE ERROR, (IF THE VALUES WERE GETTING SMALLER AND
47              ;   SMALLER, AND WE GET AN ERROR, WE MIGHT SUSPECT AN
48              ;   UNDERFLOW PROBLEM FOR EXAMPLE). THESE VALUES WILL ALSO
49              ;   BE USED IF WE HAVE TO CALCULATE, (HORRORS), THE RESULTS
50              ;   BY HAND.
51
52              ;   AN EXCEPTION IS MADE FOR ELEMENTS -1 AND -2.
53              ;   THESE ARE THE INITIAL CONDITIONS PROVIDED AT THE
54              ;   START OF THE INSTRUCTION. THE A.P. USES THESE VECTOR
55              ;   ELEMENTS AS SCRATCH DURING INSTRUCTION EXECUTION,
56              ;   WHILE THE SIMULATOR DOES NOT. THEREFORE THE RESULTS
57              ;   MAY NOT MATCH IF THE ELEMENT AT FAULT IS THE
58              ;   ZEROTH OR FIRST ELEMENT. FOR THE -1 OR -2 ELEMENTS
59              ;   THEN, USE THE NUMBERS IN THE "DATA FROM SIM" OR
60              ;   "DATA EXPECTED" COLUMNS FOR CALCULATIONS.
```

```
 0010 .MAIN
01              ;11. DEBUG HELP.
02              ;    ~~~~~~~~~~
03
04              ;   THIS PROGRAM HAS THE STANDARD BLOCK AT
05              ;   200. LOCATION 201 CONTINUES THE ADDRESS OF THE
06              ;   TEST CURRENTLY BEING EXECUTED. IT HELPS TO
07              ;   MONITOR THIS TO MAKE SURE THE PROGRAM IS STILL
08              ;   RUNNING.
09
10              ;   THERE ARE A COUPLE OF OTHER LOCATIONS WHICH
11              ;   MIGHT BE USEFUL. ALL OF THE PARAMATERS TO BE USED
12              ;   FOR THE A.P. ARE ASSEMBLED IN A BLOCK LABLED
13              ;   RRF/CRF PARAMATER BLOCK (FOR APH1/APH2 RESPECTIVLY)
14              ;   THIS BLOCK CONTAINS LABLED LOCATIONS WHERE THE GENERATED
15              ;   COEFFICIENTS, ADDRESSES ETC. ARE STORED. IF THE THING
16              ;   JUST GOES TO LUNCH YOU MIGHT GET SOME IDEA
17              ;   OF WHAT WAS HAPPENING BY LOOKING HERE. THIS BLOCK
18              ;   IS IN PAGE ZERO AROUND LOCATION 240.
19
20              ;   THE SECOND BLOCK OF INTEREST IS AT THE VERY
21              ;   END OF THE PROGRAM. THE LOCATION LABLED
22              ;   P.B.A. IS THE PARAMATER BLOCK AREA. BESIDES THE
23              ;   FILTER INSTRUCTIONS THIS PROGRAM USES THE A.P.
24              ;   INSTRUCTIONS LDR TO LOAD VECTORS INTO RAM, AND LSC TO LOAD
25              ;   THE COEFFICIENTS INTO SCRATCH. ALL A.P. INSTRUCTIONS
26              ;   USE THE P.B.A. BLOCK AS THE PARAMATER BLOCK, THERFORE
27              ;   ONE MUST BE CAREFUL TO BE AWARE OF WHO USED THE
28              ;   BLOCK LAST BEFORE MAKING RASH DECISIONS BASED ON
29              ;   ITS CONTENTS.
30
31              ;   LASTLY THERE ARE TWO BUFFERS AT THE END OF THE
32              ;   PROGRAM TO HOLD THE INPUT VECTOR, AND THE RESULT
33              ;   VECTOR GENERATED BY THE SIMULATOR. THE FORMER IS
34              ;   LABLED RRFXB/CRFXB, WHILE THE LAST ODDLY ENOUGH IS LABLED
35              ;   RRFZB/CRFZB. THE DATA IN THE XB BUFFER IS LOADED
36              ;   INTO APRAM AT THE PREDECIDED ADDRESS JUST BEFORE
37              ;   EXECUTION OF THE TEST INSTRUCTIONS. THIS VERY SAME DATA
38              ;   IS USED BY THE SIMULATOR. THIS ALLOWS A VERY USEFUL
39              ;   TECHNIQUE.
40
41              ;   BY BREAKPOINTING ON THE SIMULATOR CALL (RRFS/CRFS)
42              ;   ONE CAN GO OUT AND MODIFY THE INPUT VECTOR
43              ;   AND/OR COEFFICIENTS; DESAMPLE RATE ETC. WHEN
44              ;   YOU PROCEED THE SIMULATOR CALCULATES THE RESULT VECTOR
45              ;   USING THE MODIFIED PARAMATERS/INPUTS. THESE SAME
46              ;   MODIFIED INPUTS ARE THEN LOADED FOR EXECUTION
47              ;   INTO THE A.P.
48
49              ;   WE USED THIS TECHNIQUE WHHILE BRINGING UP THE
50              ;   PROTOTYPE TO CHANGE IMPULSES & COEFFICIENTS INTO
51              ;   SIMPLE NUMBERS LIKE 1,2,4 ETC. TO MAKE IT
52              ;   EASIER TO SEE WHAT WAS GOING ON. THE POINT TO
53              ;   BE MADE IS THAT IF YOU BREAK ON THE SIMULATOR CALL
54              ;   YOU CAN MODIFY ANY PARAMATER AND IT WILL STAY
55              ;   MODIFIED, NO ROUTINE FURTHER DOWN IN THE TEST WILL
56              ;   MODIFY THE DATA.
```

```
!0011 .MAIN
01                ;       11.4    SEQUENCE OF TESTING
02                ;
03                ;               IT IS IMPORTANT THAT THE PROGRAMS BE
04                ;               EXECUTED IN A SPECIFIC SEQUENCE:
05                ;
06                ;
07                ;               APIS DIAG (CPU/AP ONLY)
08                ;               APFP DIAG (CPU/AP ONLY)
09                ;               APA EXER
10                ;               APB EXER
11                ;               APC EXER
12                ;               APD EXER
13                ;               APE EXER
14                ;               APF EXER
15                ;               APG EXER
16                ;               APH EXER
17                ;               API EXER
18                ;               APJ EXER
19                ;               APK EXER
```

```
!0012 .MAIN
01                      O?DTO 12
02
03                ;12.  OCTAL DEBUG TOOL (ODT)
04                ;
05                ;       THE DIAGNOSTIC IS EQUIPED WITH A BUILT IN ODT WHICH CAN
06                ;       BE ACCESSED BY HITTING CONTROL O (^O) AT ANY TIME DURING
07                ;       THE EXECUTION OF THE PROGRAM (AFTER SETTING THE PARA-
08                ;       METERS).
09                ;       ON ENTERING ODT THE ADDRESS OF THE LOCATION HAVING THE
10                ;       NEXT INSTRUCTION TO BE EXECUTED WILL BE TYPED-OUT.
11                ;
12                ;12.1 CONVENTIONS AND SYMBOLS
13                ;       THE FOLLOWING CONVENTIONS ARE USED BY THE ODT:
14                ;       ?       PRESSING ANY ILLEGAL KEY CAUSES THE ODT TO RES-
15                ;               POND WITH A "?".
16                ;       @       ODT IS READY AND AT YOUR SERVICE.
17                ;
18                ;12.2 COMMAND STRUCTURE
19                ;       AN ODT COMMAND HAS THE FOLLOWING FORMAT:
20                ;               [ARGUMENT][COMMAND]
21                ;       AN ARGUMENT MAY BE ONE OF THE FOLLOWING:
22                ;       "EXP"   AN OCTAL EXPRESSION CONSISTING OF OCTAL NUMBERS
23                ;               SEPARATED BY PLUS (+) OR MINUS (-) SIGNS. LEAD-
24                ;               ING ZEROS NEED NOT BE TYPED.
25                ;       "ADR"   AN ADDRESS IS THE SAME AS AN EXPRESSION EXCEPT
26                ;               THAT BIT 0 IS NEGLECTED.
27                ;       A COMMAND IS A SINGLE TELETYPE CHARACTER
28                ;
29                ;12.3 ODT COMMANDS
30                ;       THE LOCATIONS THAT CAN BE EXAMINED AND MODIFIED BY THE
31                ;       USER ARE CALLED CELLS. THESE CELLS ARE OF TWO TYPES:
32                ;       INTERNAL CPU CELLS AND MEMORY LOCATIONS.
33                ;
34                ;12.3.1 OPENING INTERNAL CELLS
35                ;       THE COMMAND TO OPEN ONE OF THE INTERNAL REGISTERS IS OF
36                ;       THE FORM "NA" WHERE N IS ANY OCTAL EXPRESSION BETWEEN
37                ;       0 AND 7
38                ;       0-3     FOR ACCUMULATORS 0-3
39                ;       4       FOR PC OF THE NEXT INSTRUCTION TO BE EXECUTED IN
40                ;               THE EVENT OF A "P" COMMAND.
41                ;       5       CPU AND TTO STATUS
42                ;               BIT     INTERPRETATION
43                ;               15      STATUS OF TTO DONE FLAG
44                ;               14      STATUS OF INTERRUPTS (ION FLAG)
45                ;               13      STATUS OF CARRY BIT
46                ;       6       ADDRESS OF THE LOCATION HAVING THE BREAK POINT (IF
47                ;               ANY)
48                ;       7       INSTRUCTION AT THE BREAK POINT LOCATION
49                ;
50                ;       OTHER COMMANDS TO OPEN CELLS ARE:
51                ;
52                ;       "ADR"/  OPEN THE CELL AND PRINT ITS CONTENTS
53                ;       ./      OPEN THE CELL CURRENTLY POINTED TO BY THE POINTER
54                ;               AND PRINT ITS CONTENTS.
55                ;       .+"ADR"/        ADD "ADR" TO THE POINTER, OPEN THE CELL
56                ;               AND PRINT ITS CONTENTS.
57                ;       .-"ADR"/        SUBTRACT "ADR" FROM THE POINTER, OPEN
58                ;               THE CELL AND PRINT ITS CONTENTS.
59                ;       "CR"    THE RETURN KEY IS USED TO CLOSE THE OPEN CELL
60                ;               WITH OR WITHOUT MODIFICATION.
```

```
 0013 .MAIN
01              ;       "LF"    LINE FEED IS USED TO CLOSE THE OPEN CELL WITH OR
02              ;               WITHOUT MODIFICATION AND TO OPEN THE SUCCEEDING
03              ;               CELL.
04              ;       ^       CLOSE THE OPEN CELL WITH OR WITHOUT MODIFICATION
05              ;               AND OPEN THE PRECEDING CELL
06              ;       /       CLOSE THE OPEN CELL WITHOUT MODIFICATION, AND
07              ;               OPEN THE CELL POINTED TO BY ITS CONTENTS.
08              ;       +"ADR"/ CLOSE THE OPEN CELL WITHOUT MODIFICATION, AND
09              ;               OPEN THE CELL POINTED TO BY ITS CONTENTS + "ADDR".
10              ;       -"ADR"/ CLOSE THE OPEN CELL WITHOUT MODIFICATION, AND
11              ;               OPEN THE CELL POINTED TO BY ITS CONTENTS - "ADR".
12              ;
13              ;12.3.2 MODIFICATION OF A CELL
14              ;       ONCE A CELL HAS BEEN OPENED ITS CONTENTS CAN BE MODIFIED
15              ;       BY TYPING THE NEW VALUE THE CELL IS TO CONTAIN IN THE
16              ;       FORM OF AN OCTAL EXPRESSION FOLLOWED BY "CR" OR "LF".
17              ;       IF A + OR - IS TYPED AS THE FIRST CHARACTER OF THE EX-
18              ;       PRESSION THEN THE VALUE OF THE EXPRESSION IS ADDED TO OR
19              ;       SUBTRACTED FROM THE OLD CONTENTS OF THE CELL. THE
20              ;       ADDRESS ITSELF OR AN EXPRESION RELATIVE TO THE ADDRESS
21              ;       CAN BE DEPOSITED BY TYPING A "." OR ".+/-OCTAL EXPRESS-
22              ;       ION". A RUBOUT COMMAND GIVEN RIGHT AFTER OPENING A CELL
23              ;       ALLOWS THE MODIFICATION OF ITS CONTENTS AS IF THEY WERE
24              ;       TYPED IN JUST BEFORE THE COMMAND WAS ISSUED.
25              ;
26              ;12.3.3 OTHER ODT COMMANDS
27              ;       RUBOUT  THIS KEY IS USED TO DELETE ERRONEOUSLY TYPED
28              ;               DIGITS. EACH TIME THE KEY IS PRESSED THE RIGHT MOST
29              ;               DIGIT IS DELETED AND ECHOED ON THE TERMINAL. IF
30              ;               THE RUBOUT KEY IS PRESSED RIGHT AFTER OPENING A
31              ;               CELL THEN IT DELETES THE RIGHT MOST DIGIT OF THE CELLS
32              ;               CONTENTS. THIS ALLOWS THE MODIFICATION OF THE CELL
33              ;               AS IF ITS CONTENTS WERE TYPED IN JUST BEFORE THE
34              ;               KEY WAS PRESSED.
35              ;       "ADR"B  INSERT A BREAK POINT AT LOCATION "ADR".
36              ;               ONLY ONE BREAK POINT CAN BE INSERTED AND ANY
37              ;               ENTRY TO ODT AFTER EXECUTING A BREAK POINT WILL
38              ;               CAUSE IT TO BE DELETED.
39              ;       D       DELETE THE BREAK POINT IF ANY.
40              ;       P       RESTART THE EXECUTION OF THE PROGRAM AT LOCATION
41              ;               POINTED BY 4A.
42              ;       "ADR"R  START EXECUTING THE PROGRAM AT "ADR" AFTER AN
43              ;               IO-RESET.
44              ;       K       KILL THE STRING TYPED SO FAR. THE ODT RESPONDS
45              ;               WITH A "?" AND THE OPEN CELL IS CLOSED WITHOUT
46              ;               MODIFICATION.
47              ;       =       PRINT THE OCTAL VALUE OF THE INPUT ONLY.
48              ;               THIS WILL CLOSE ANY OPEN CELLS WITHOUT
49              ;               MODIFICATION AND WILL NOT OPEN A CELL
50              ;
51              ;
52              ;       NOTE:   IN PROGRAMS WHICH RELOCATE THEMSELVES THE
53              ;               THE USER SHOULD PLACE BREAK POINTS ONLY IN THE
54              ;               THE ORIGINAL PROGRAM AREA. IF A BREAK POINT IS
55              ;               PLACED OUTSIDE THIS AREA THE RESULTS WILL
56              ;               BE UNPREDICTABLE.
57              ;
58              ;
```

```
!0014 .MAIN
01              ;13.    SPECIAL NOTES/SPECIAL FEATURES
02              ;
03              ;       13.1    FOR A COMPLETE TEST ALL PROGRAMS
04              ;               SHOULD BE EXECUTED WITH CAT/KITTEN.
05              ;
06              ;       13.2    A NOTE ABOUT AP ADDRESSING
07              ;
08              ;               ADDRESSES IN THE AP CAN BE OF SEVERAL
09              ;               MODES:
10              ;
11              ;                   1) ONE WORD MODE (1E) SAME AS STANDARD
12              ;                      ADDRESSING
13              ;                   2) TWO WORD MODE (2WM) - EACH 32 BITS IS
14              ;                      NOW ONE ADDRESS SPACE.  THIS IS USED
15              ;                      IN THE AP TO SIMPLIFY REAL NUMBER
16              ;                      ADDRESSING.
17              ;                   3) FOUR WORD MODE (4WM) - EACH 64 BITS IS
18              ;                      NOW ONE ADDRESS SPACE.  THIS IS USED
19              ;                      IN THE AP TO SIMPLIFY COMPLEX NUMBER
20              ;                      ADDRESSING.
21              ;
22              ;               THE AP ACCESSES AN ADDRESS RELATIVE TO THE START OF
23              ;               THE AP RAM.  THUS AP RAM LOC 0 WOULD BE THE
24              ;               FIRST LOCATION THAT IS IN THE AP.  HOWEVER,
25              ;               AS FAR AS THE ECLIPSE CPU IS CONCERNED, AP
26              ;               LOC 0 IS CONTAINED AT LOCATION LABEL
27              ;               "RAMPT". (IN PAGE ZERO.  SO, IF RAMPT CON-
28              ;               TAINS 64000, THEN 2000 2WM (AP RAM) IS REALLY
29              ;               64000+2000+2000=70000.
30              ;
31              ;               NOTE: "STOP ON STORE" OR "STOP ON ADDRESS" IN AP RAM
32              ;               SPACE WILL NOT WORK IF THE AP IS USING
33              ;               THE INTERNAL AP ADDRESS LINES TO ACCESS AP RAM.
```

```
10015 .MAIN
01              ;14.0   RUN TIME
02              ;
03              ;       14.1    PASS 1                  50 SEC
04              ;
05              ;       14.2    SUBSEQUENT PASSES       55 SEC
06              ;

**00000 TOTAL ERRORS, 00000 PASS 1 ERRORS
```

```
 0016 .MAIN

O?DTD 001551  MC    12/01
S?WPD 001075  MC     5/01
```