FAST MEMORY UCODE

HIGH=000342

SYMBOL TABLE'

SYMBOL	VALUE	
FIFFT	000000	
VFLT	000024	
CFFT	000050	CFFT
VSHFX	000037	
ILOG2	000113	
FFT2	000204	
FFT4	000224	FFT4
STSTAT	000075	
CLSTAT	000110	
ADV2	000124	
ADV4	000120	
BITREV	000130	

0

342

A FFT 02

~~APCODED~~

MPD > AEC > APCODE > FFT

ERROR - $127_8 \rightarrow 45670$

SPOOL
APAL
Ø
NAME
TO
SPOOL
1
Ø
1

Also I will include the uCode listing I found which seems to be the diagnostic that runs an FFT and IFFT and checks that the input and output are the same (within some tolerance). This was the test that was run overnight at extended voltages before the unit was shipped.

```
1APAL
 REV 1
 PASS 1
 PASS 2
                    "FFT IDENTITY TEST
                    "
                            $TITLE FIFFT
                            $ENTRY FIFFT,3
                            $EXT VFLT,CFFT,VSHFX,ILOG2
                    "
                    "COMPATABLE WITH 'FIFFT' CALLING PROGRAM IN THE HOST
                    "
                    "
                    "S-PAD PARAMETERS:
                    "
            000000          N $EQU 0        "NUMBER OF COMPLEX POINTS

            000001          M $EQU 1        "M=LOG2(N)   (IGNORED)

            000002          BASE $EQU 2     "BASE ADDRESS OF THE ARRAY

                    "
                    "
                    "SETUP FOR FLOAT
 000000     042023  FIFFT:  MOVL 0,4; REFR
            174000
            000000
            000000

 000001     040200          MOV 2,0
            000000
            000000
            000000

 000002     001604          LDSPI 1; DB=1
            000000
            002000
            000001

 000003     040114          MOV 1,3
            000000
            000000
            000000

 000004     011014          JSR VFLT                "CALL VFLT(BASE,1,BASE,1,N*2)
            000000
            000000
            177777

                    "SETUP FOR FORWARD COMPLEX FFT
 000005     046404          MOVR 4,1
            000000
            000000
            000000

 000006     001610          LDSPI 2; DB=1
```

```
                000000
                002000
                000001

000007  011014          JSR CFFT                "CALL CFFT(BASE,N,1)
                000000
                000000
                177777

                        "SETUP FOR INVERSE COMPLEX FFT
000010  001610          LDSPI 2; DB=-1
                000000
                002000
                177777

000011  011014          JSR CFFT                "CALL CFFT(BASE,N,-1)
                000000
                000000
                000007

                        "SETUP FOR SHIFT AND FIX
000012  040174          MOV 1,17
                000000
                000000
                000000

000013  011014          JSR ILOG2               "GET LOG2(N) INTO SP(17)
                000000
                000000
                177777

000014  001024          CLR 5
                000000
                000000
                000000

000015  031724          SUB 17,5                "SHIFT IS - POWER OF 2
                000000
                000000
                000000

000016  042120          MOVL 1,4
                000000
                000000
                000000

000017  040010          MOV 0,2
                000000
                000000
                000000

000020  001604          LDSPI 1; DB=1
                000000
                002000
                000001
```

```
000021  040114          MOV 1,3
        000000
        000000
        000000

000022  011014          JSR VSHFX   37      *CALL VSCL(BASE,1,BASE,1,N*2,0)
        000000
        000000
        177777

000023  000003          HALT
        170000
        000000
        000000

                        $END


****     0 ERRORS   ****


SYMBOL  VALUE
VFLT    000004 EXT ✓
CFFT    000011 EXT
VSHFX   000022 EXT✓
ILOG2   000013 EXT
N       000000
M       000001
BASE    000002
FIFFT   000000 ENT
1
```

```
1APAL
 REV 1
 PASS 1
 PASS 2
                "AP-120B SOURCE  VFLT    712-0034-000-A-13  VECTOR FLOAT
                        $TITLE VFLT
                        $ENTRY VFLT, 5
                "VECTOR FLOAT
                "
                "                --- ABSTRACT ---
                "FLOATS A VECTOR OF INTEGERS (28-BITS WIDTH MAX)
                "
                "                --- STATISTICS ---
                "LANGUAGE:       AP-120B ASSEMBLER (APAL)
                "EQUIPMENT:      AP-120B, 2 CYCLE MD
                "STORAGE:        PS - 9
                "                MD - FOR VECTORS
                "                TM - NONE
                "                DPX - 1 (SCRATCH)
                "                DPY - NONE
                "                SP - 5 (PARAMETERS), 4 (SCRATCH)
                "SPEED: 0.833 US. + .66 US. TO 1.0 US. PER POINT
                "AUTHOR:         A.E. CHARLESWORTH
                "DATE:           OCT. 75
                "VERSION 1.3  FEB 76  CHANGED $ENTRY
                "
                "                --- USEAGE ---
                "SAMPLE CALL:    JSR VFLT
                "ARGUEMENTS - SPAD:
                "        0        A        BASE ADDRESS OF VECTOR A
                "        1        I        INCREMENT FOR A
                "        2        C        BASE ADDRESS  VECTOR C
                "        3        K        INCREMENT FOR C
                "        4        N        NUMBER OF ELEMENTS IN C
                "
                "SCRATCH:        S-PAD 12.-15.
                "
                "                    --- ALGORITHM ---
                "1.      INPUT: 2'S COMPLEMENT INTEGERS IN THE MANTISSA PORTION OF
                "          MAIN DATA MEMORY WORDS
                "2.      FLOATING IS DONE BY ADDING THE INTEGERS TO 0.0, THE ADDER
                "          WILL NORMALIZE THE ANSWERS.  THE "MDPX" FLOATING ADD OPERAND
                "          IN THE INSTRUCTION AT LOCATION "LOOP" CAUSES THE MANTISSA FROM
                "          DPX (I.E. OUR INTEGER) TO BE COMBINED WITH SPFN AS THE
                "          EXPONENT TO GIVE A FLOATING POINT NUMBER (ALTHOUGH
                "          UN-NORMALIZED) TO ADD TO 0.0.  27 IS THE PROPER EXPONENT
                "          VALUE SINCE 27 LEFT SHIFTS WILL NORMALIZE AN INTEGER 1 (A 1
                "          IN THE LSB) INTO A NORMALIZED FRACTION OF 0.5

                "        S-PAD MNEMONICS:
000000                   A $EQU 0

000001                   I $EQU 1

000002                   C $EQU 2
```

```
                000003          K $EQU 3
                000004          N $EQU 4
                000014          CTR $EQU 14
                000015          T $EQU 15
                000016          APTR $EQU 16
                000017          CPTR $EQU 17
000000  040070  VFLT:   MOV A,APTR; SETMA          "FETCH A(0)
        000000
        000000
        000060

000001  001664          LDSPI  T ; DB=27.        "EXPONENT
        000000
        002000
        000033

000002  040460          MOV N,CTR
        000000
        000000
        000000

000003  020170          ADD I,APTR; SETMA;          "FETCH A(1)
        000000            DPX<MD                    "  SAVE A(0)
        045004
        000060

000004  040274           MOV C,CPTR
        000000
        000000
        000000

000005  030374          SUB K,CPTR
        000000
        000000
        000000

000006  041565  LOOP:   DPX<MD;                     "SAVE A(M+1)
        156000            FADD ZERO,MDPX;           "  FLOAT A(M)
        045404            MOV T,T                   "  SUPPLY EXPONENT
        000000

000007  020170          ADD I,APTR; SETMA           "FETCH A(M+2)
        000000
        000000
        000060

000010  001261          FADD;DEC CTR                "SEE IF DONE???
        100000
        000000
        000000
```

```
000011  020374          ADD K,CPTR; SETMA; MI<FA;  "STORE C(M)
        000655            BNE LOOP
        000000
        000160

000012  000000          RETURN                    "DONE
        000340
        000000
        000000

                        $END


****     0 ERRORS   ****


SYMBOL  VALUE
A       000000
I       000001
C       000002
K       000003
N       000004
CTR     000014
T       000015
APTR    000016
CPTR    000017
VFLT    000000 ENT
LOOP    000006
1
```

35

36

```
1APAL
 REV 1
 PASS 1
 PASS 2
                "VECTOR SHIFT AND FIX,  RELEASE: REV 2
                        $TITLE VSHFX
                        $ENTRY VSHFX, 6

                "                       --- ABSTRACT ---
                "SHIFTS (MULTIPLIES BY A POWER OF 2) AND FIXES

                "                       --- STATISTICS ---
                "EQUIPMENT:             AP-120B
                "LANGUAGE:              AP-120B ASSEMBLER
                "AUTHOR:                A.E. CHARLESWORTH
                "DATE:                  SEPT 1976
                "SIZE:                  9 LOCATIONS
                "SPEED:                 2 MEMORY CYCLES PER POINT
                "SCRATCH:               S-PAD:  0,2,4,6

                "                       --- USEAGE ---

                "S-PAD PARAMETERS:
        000000          A $EQU 0        "BASE ADDRESS OF A

        000001          I $EQU 1        "INCREMENT FOR A

        000002          C $EQU 2        "BASE ADDRESS OF C

        000003          K $EQU 3        "INCREMENT FOR C

        000004          N $EQU 4        "ELEMENT COUNT FOR C

        000005          SHFT $EQU 5     "SHIFT (POWER OF 2 MULTIPLY) DESIRED BEFORE FIXING

                                        "  + IS LEFT SHIFT, - IS RIGHT SHIFT
                                        "  I.E. -2 WILL SHIFT RIGHT 2 (DIVIDE BY 4)
                                        "  BEFORE DOING A FIX (RIGHT JUSTIFY
                                        "  IN 28 BITS)

                "SCRATCH S-PAD:
        000006          T $EQU 6

000000  040000  VSHFX:  MOV A,A; SETMA          "FETCH FIRST A
        000000
        000000
        000060

000001  001630          LDSPI T; DB=
        000000
        002000
        000035

000002  030530          SUB SHFT,T              "THIS IS THE FSCALE VALUE
        000000
```

```
          000000
          000000

000003    030310          SUB K,C
          000000
          000000
          000000

000004    040630  SLOOP:  MOV T,T; FSCLT MD       "FIX INTEGER
          034000
          000000
          000000

000005    020101          ADD I,A; SETMA; FADD    "FETCH NEXT A
          100000
          000000
          000060

000006    001220          DEC N
          000000
          000000
          000000

000007    020310          MI<FA; ADD K,C; SETMA;  "STORE C
          000655            BNE SLOOP
          000000
          000160

000010    000000          RETURN
          000340
          000000
          000000

                          $END


****      0 ERRORS    ****


SYMBOL    VALUE
A         000000
I         000001
C         000002
K         000003
N         000004
SHFT      000005
T         000006
VSHFX     000000 ENT
SLOOP     000004
```

```
1APAL
 REV 1
 PASS 1
 PASS 2
                "MAINLINE FOR A COMPLEX FFT
                "
                "
                        $TITLE CFFT
                        $ENTRY CFFT,3
                "
                        $EXT FFT2,FFT4,STSTAT,CLSTAT,ADV2,ADV4,BITREV
                "
                "
                "
                "                                       --- ABSTRACT ---
                "DOES A COMPLEX FFT
                "
                "
                "                                       --- STATISTICS ---
                "EQUIPMENT:                             AP-120B
                "SIZE:                                  20 LOCATIONS
                "AUTHOR:                                A.E. CHARLESWORTH
                "DATE:                                  JUNE 1976
                "RELEASE:                               REV 2
                "
                "
                "                                       --- USEAGE ---
                "DOES A COMPLEX FFT
                "
                "S-PAD PARAMETERS:
                "       NAME      NUMBER          PURPOSE
000000          C    $EQU 0                    "BASE ADDRESS OF ARRAY

000001          N    $EQU 1                    "# OF COMPLEX POINTS IN THE ARRAY

000002          F    $EQU 2                    "DIRECTION: 1= FORWARD,  -1= INVERSE

                "
                "
                "                                       --- SCRATCH (OVERALL) ---
                "S-PAD:                                 2-17
                "DATA PAD X:                            -4 THROUGH +3 (RELATIVE TO DPA)
                "DATA PAD Y:                            -4 THROUGH +3 (RELATIVE TO DPA)
                "
                "
                "
                "
                "                                       --- ALGORITHM ---
                "
                "PROCEEDURE:
                "
                "1.     CALL 'STSTAT'       THIS TAKES 'N' AND 'F', AND SETS THE BIT-REVERSE
                "                           AND FFT-MODE BITS IN THE STATUS REGISTER,  AND
                "                           LEAVES 2**N IN SPAD(17)
                "2.     PARAMETER INITIALIZE  SET THE FOUR 'PASS VARIABLES'
```

```
                "                               WD = TBLSZE * 2
                "                               MDEL = 2
                "                               JCOUNT = 1
                "                               ICOUNT = N/4  IF N A POWER OF 4
                "                                      = N/2 IF N A POWER OF 2
                "                               AND SET N2 = 2
                "3.     DO THE FIRST PASS:      RADIX 2 (FFT2) IF N WAS A POWER OF 2
                "                                  AND ADVANCE (ADV2)
                "                               RADIX 4 (FFT4) IF N WAS A POWER OF 4
                "                                  AND ADVANCE (ADV4)
                "4.     DO THE REST OF THE RADIX 4 PASSES.
                "                               CALL FFT4  (TO DO THE NEXT FFT PASS)
                "                               CALL ADV4  (TO ADVANCE TO THE NEXT PASS)
                "                               TEST ICOUNT (SPFN AFTER ADV4) FOR ZERO TO TEST
                "                                 FOR DONE
                "5.     WHEN DONE, CALL 'CLSTAT' TO CLEAR THE BIT-REVERSE AND FFT-MODE
                "                                STATUS BITS
                "
                "
                "OTHER S-PAD NAMES:
000010          N2      $EQU 10         "CONTAINS 2
000014          WD      $EQU 14         "W DELTA
000015          MDEL    $EQU 15         "MEMORY DELTA
000016          ICOUNT  $EQU 16         "I-LOOP COUNT
000017          JCOUNT  $EQU 17         "J-LOOP COUNT

                "
                "
                "
                "SIZE OF INSTALLED FFT TABLE
004000          TBLSZE $EQU !FFTSZ

                "
                "
                "
                "SET UP FOR THE FFT
                "
                "INITIALIZE POINTERS:
                "
                "SET AP-STATUS FOR FORWARD OR INVERSE FFT
                "SET AP-STATUS FOR BIT-REVERSE SIZE
                "JCOUNT<1
                "WD<!FFTSZ*2
                "MDEL<2
                "N2<2
                "ICOUNT<N/4  IF 1ST PASS RADIX 4
                "ICOUNT<N/2  IF 1ST PASS RADIX 2
                "
000000  040270  CFFT:   MOV F,16                "SET-UP FOR STSTAT
        000000
        000000
        000000
```

51

```
000001  040174          MOV N,17
        000000
        000000
        000000

000002  011014          JSR STSTAT              "SET FFT STATUS BITS
        000000
        000000
        177777

000003  011014          JSR BITREV              "BIT-REVERSE THE ARRAY
        000000
        000000
        177777

54
000004  001660          LDSPI WD; DB=TBLSZE*2   "SET WD=!FFTSZ*2
        000000
        002000
        010000  → 20000

000005  001664          LDSPI MDEL; DB=2
        000000
        002000
        000002

000006  001640          LDSPI N2; DB=2          "PUT 2 INTO N2
        000000
        002000
        000002

000007  047774          MOVR 17,17              "WAS LOG2(N) ODD ?????
        000000
        000000
        000000

000010  010010          BNC R2                  "IF SO, DO A RADIX 2 PASS FIRST
        000023
        000000
        000000

                "
                "SET UP FOR FIRST RADIX 4 PASS
000011  044170  R4:     MOVRR N,ICOUNT          "SET ICOUNT=N/4
        000000
        000000
        000000

000012  001674          LDSPI JCOUNT; DB=1;     "SET COUNT=1
        000125             BR LOOP              "AND GO LOOP
        002000
        000001

                "
                "DO A RADIX 2 PASS FIRST
000013  046170  R2:     MOVR N,ICOUNT           "SET ICOUNT=N/2
```

```
                000000
                000000
                000000

000014  011014          JSR FFT2
                000000
                000000
                177777

000015  001674          LDSPI JCOUNT; DB=1      "SET JCOUNT=1
                000000
                002000
                000001

000016  011014          JSR ADV2                "RADIX 2 ADVANCE
                000000
                000000
                177777

                "
                "
                "LOOP HERE TO DO ALL THE PASES
000017  011014  LOOP:   JSR FFT4                "DO A RADIX 4 PASS
                000000
                000000
                177777

000020  011014          JSR ADV4                "RADIX 4 PASS ADVANCE
                000000
                000000
                177777

                "
                "DONE IF 'ADV' HAS SHIFTED ICOUNT TO ZERO
000021  000000          BNE LOOP                "IF NOT DONE, GO BACK FOR MORE
                000656
                000000
                000000

                "
                "
                "WHEN DONE, CLEAR FFT MODE BITS
                "
000022  011014          JSR CLSTAT              "CLEAR STATUS REGISTER
                000000
                000000
                177777

000023  000000          NOP
                000000
                000000
                000000

000024  000000          RETURN
                000340
                000000
```

```
        000000

                        $END


****     0 ERRORS   ****


SYMBOL  VALUE
FFT2    000014 EXT
FFT4    000017 EXT
STSTAT  000002 EXT
CLSTAT  000022 EXT
ADV2    000016 EXT
ADV4    000020 EXT
BITREV  000003 EXT
C       000000
N       000001
F       000002
N2      000010
WD      000014
MDEL    000015
ICOUNT  000016
JCOUNT  000017
TBLSZE  004000
CFFT    000000 ENT
 R4     000011
 R2     000013
 LOOP   000017
1
```

```
1APAL
 REV 1
 PASS 1
 PASS 2
                $TITLE STATUS
                $ENTRY STSTAT
                $ENTRY CLSTAT
                $ENTRY ILOG2
              "FFT UTILITY SUBROUTINES
              "
              "                    --- ABSTRACT ---
              "THESE SUBROUTINES PERFORM THREE TASKS COMMON TO FFT RELATED PROGRAMS
              "
              "                    --- STATISTICS ---
              "LANGUAGE:      AP-120B ASSEMBLER
              "EQUIPMENT:     AP-120B
              "STORAGE:       13 PROGRAM LOCATIONS
              "SPEED:         NEGLIGABLE
              "AUTHOR:        A.E.CHARLESWORTH
              "DATE:          JUNE 1976
              "
              "                    --- USEAGE ---
              "       STSTAT
              "SETS THE FFT-MODE AND BIT-REVERSE BITS IN THE AP-STATUS REGISTER
              "ARGUMENTS:
              "       SP(17)  N:  NUMBER OF COMPLEX POINTS
              "       SP(16)  FLAG: +1 FOR DIRECT, -1 FOR INVERSE TRANSFORM
              "                              OR 0 FOR NO TRANSFORM
              "
              "RESULTS:
              "       SP(17)  SET TO LOG2(N)
              "SCRATCH: SP(14-17)
              "
              "         CLSTAT
              "CLEARS THE FFT MODE BITS IN THE AP STATUS REGISTER
              "RESULTS:  AP-STATUS RETURNED IN SP(14)
              "SCRATCH: SP(14-15)

              "         ILOG2
              "COMPUTES  LOG2 ( SP(17) )
              "ARGUMENTS:  INPUT   SP(17)
              "            ANSWER  SP(17)
              "SCRATCH: SP(16-17)


               "STSTAT   ---   SET THE TWO FFT MODE STATUS BITS
              "       DEPENDING ON SP(16)
              "       AND THE BIT-REVERSE BITS DEPENDING UPON SP(17)
000000  011014  STSTAT: JSR CLSTAT                        "CLEAR THE FIVE BITS
        000000
        000000
        000013

000001  023670          ADDL 16,16              "SHIFT LEFT 2 PLACES
        000000
        000000
```

75

```
                000000

000002  001664          LDSPI 15; DB=14         "MASK FOR FFT-MODE BITS
        000000
        002000
        000014

000003  053570          ANDL 15,16              "PROPER FFT-MODE BITS
        000000
        000000
        000000

000004  061660          OR 16,14                "OR INTO STATUS REGISTER
        000000
        000000
        000000

000005  011014          JSR ILOG2               "GET LOG2( SP(17) )
        000000
        000000
        000011

000006  001670          LDSPI 16; DB=15.        "WE WANT 15.-LOG2(N)
        000000
        002000
        000017

000007  031770          SUB 17,16               "BIT REVERSE SHIFT
        000000
        000000
        000000

000010  001664          LDSPI 15; DB=7          "MASK FOR SHIFT
        000000
        002000
        000007

000011  051570          AND 15,16               "PROPER BIT-REVERSE BITS
        000000
        000000
        000000

000012  061663          OR 16,14; DB=SPFN;      "OR INTO STATUS REGISTER
        106340            LDAPS; RETURN         "AND RETURN
        006000
        000000


                "CLEAR THE TWO FFT MODE STATUS BITS

000013  001463  CLSTAT: RAPS; LDSPNL 14                "GET AP STATUS
        116000
        000000
        000000
```

14 = WD

← 103

Bottom 3

105

110

```
000014  001664          LDSPI 15;  DB=177740            "GET MASK
        000000
        002000
        177740

000015  051563          AND 15,14;  DB=SPFN;            "MASK OUT THE UNWANTED BITS
        106340            LDAPS;  RETURN                "  RESET STATUS
        006000
        000000


                "ILOG2  ---   HERE WE GET LOG2 ( SP(17) )
                "NOTE:  IF THE INPUT IS NOT A POWER OF 2,  THE RESULT WILL BE THE
                "             NEXT LOWER POWER OF 2

000016  041770  ILOG2:  MOV 17,16
        000000
        000000
        000000

000017  001074          CLR 17                  "CLEAR OUR COUNTER
        000000
        000000
        000000

                "
000020  047670  LOOP:   MOVR 16,16               "DIVIDE BY 2
        000000
        000000
        000000

000021  001174          INC 17; BGT LOOP        "COUNT UP RESULT
        000757
        000000
        000000

000022  001274          DEC 17; RETURN          "DECREMENT ANSWER AND RETURN
        000340
        000000
        000000

                        $END


****     0 ERRORS   ****


SYMBOL  VALUE
STSTAT  000000 ENT
CLSTAT  000013 ENT
ILOG2   000016 ENT
LOOP    000020
```

111

115

SAY SP(17) = 1 ⇒ SP16 = 0

2 = 1

4 = 2

10 = 3

20 = 4

40 = 5

100 = 6

1000 = 9

16000 = 12

20000 = 13

40000 = 14

100000 = 15

```
1APAL
 REV 1
 PASS 1
 PASS 2
```

ADV4

```
"ADVANCE FROM A RADIX 4 OR RADIX 2 FFT PASS
"
        $TITLE ADV
        $ENTRY ADV4
        $ENTRY ADV2
"
"
"                                   --- ABSTRACT ---
"
"ADVANCES THE POINTERS USED IN AN FFT FROM ONE PASS TO ANOTHER
"
"
"                                   --- STATISTICS ---
"EQUIPMENT:                         AP-120B
"SIZE:                              8 LOCATIONS
"AUTHOR:                            A.E. CHARLESWORTH
"DATE:                              JUNE 1976
"RELEASE:                           REV 2
"
"
"                                   --- USEAGE ---
"THIS IS 'CALLED' AT THE END OF EACH PASS OF AN FFT,
"    ADV4    FOR A RADIX 4 PASS
"    ADV2    FOR A RADIX 2 PASS
"
"THE FOUR PARAMETERS WHICH DEFINE THE PASS OF AN FFT
"ARE UPDATED:
"
"        S-PAD #    NAME     PURPOSE
"        14         WD       W DELTA (INCREMENT BETWEEN SUCESSIVE W'S)
"        15         MDEL     MEMORY DELTA (INCREMENT BETWEEN COMPLEX PAIRS)
"        16         ICOUNT   I-LOOP COUNT
"        17         JCOUNT   J-LOOP COUNT
"
"
"THESE ARE ADVANCED AS FOLLOWS:
"
"        WD = WD / R
"        MDEL = MDEL * R
"        JCOUNT = JCOUNT * R
"        ICOUNT = ICOUNT / R
"
"WHERE 'R' IS THE RADIX OF THE PRECEEDING FFT PASS,
"    EITHER 2 OR 4.
"
"IN ADDITION, THE LAST 'SPFN' IS ICOUNT / R
"WHICH SHOULD BE TESTED BY THE CALLING PROGRAM IN THE
"INSTRUCTION FOLLOWING THE RETURN TO SEE WHETHER THE
"FFT IS DONE.
"
"THE FFT IS DONE WHEN ICOUNT IS SHIFTED DOWN TO ZERO
"
```

ADV4

```
                "
                "                                   --- SCRATCH ---
                "NONE
                "
                "
                "
                "S-PAD DEFINITIONS:
                "
        000014          WD      $EQU 14         "W DELTA
        000015          MDEL    $EQU 15         "MEMORY DELTA
        000016          ICOUNT  $EQU 16         "I-LOOP COUNT
        000017          JCOUNT  $EQU 17         "J-LOOP COUNT
                "
                "
                "ADVANCE POINTERS FROM ONE RADIX 4 PASS TO THE NEXT
                "
                "MDEL<MDEL*4
                "WD<WD/4
                "JCOUNT<JCONT*4
                "ICOUNT<ICOUNT/4
                "       .
                "
```

120

```
000000  023564  ADV4:   ADDL MDEL,MDEL          "MDEL=MDEL*4
        000000
        000000
        000000

000001  045460          MOVRR WD,WD             "WD=WD/4
        000000
        000000
        000000
        000000

000002  023774          ADDL JCOUNT,JCOUNT      "JCOUNT=JCOUNT*4
        000000
        000000
        000000

000003  045670          MOVRR ICOUNT,ICOUNT;    "ICOUNT=ICOUNT/4
        000340             RETURN
        000000
        000000
```

```
                "
                "
                "ADVANCE FROM A RADIX 2 PASS:
                "       MDEL<MDEL*2
                "       WD<WD/2
                "       JCOUNT=JCOUNT*2
                "       ICOUNT<ICOUNT/2
                "
```

124

```
000004  043564  ADV2:   MOVL MDEL,MDEL          "MDEL=MDEL*2
        000000
```

ADV4

```
                000000
                000000

000005  047460          MOVR WD,WD              "WD=WD/2
                000000
                000000
                000000

000006  043774          MOVL JCOUNT,JCOUNT      "JCOUNT=JCOUNT*2
                000000
                000000
                000000

000007  047670  45670   MOVR ICOUNT,ICOUNT;     "ICOUNT=ICOUNT/2
                000340      RETURN
                000000
                000000

                        $END


****     0 ERRORS   ****


SYMBOL  VALUE
WD      000014
MDEL    000015
ICOUNT  000016
JCOUNT  000017
ADV4    000000 ENT
ADV2    000004 ENT
1
```

BITREV

```
1APAL
 REV 1
 PASS 1
 PASS 2
```

AT 130

```
        $TITLE BITREV
        $ENTRY BITREV
"BIT REVERSE ORDER AN ARRAY

"                   --- ABSTRACT ---
"TAKES A COMPLEX DATA ARRAY AND PUTS IT IN BIT-REVERSED (RADIX 2) ORDER

"                   --- STATISTICS ---
"LANGUAGE:          AP-120B ASSEMBLER
"EQUIPMENT:         AP-120B WITH 1 CYCLE (FAST) MEMORY
"SIZE:
"SPEED:             0.92 US PER POINT
"AUTHOR:            A.E. CHARLESWORTH
"DATE:              AUGUST 1976
"RELEASE:           REV 2
"                   --- USEAGE ---
"S-PAD PARAMETERS:
"       0 BASE                          BASE ADDRESS OF ARRAY
"       1 N                             NUMBER OF COMPLEX ELEMENTS
"
"ALSO: THE STATUS REGISTER BITREVERSE BITS SET PROPERLY TO
"      IAND(15-LOG2(N),7)

"                   --- SCRATCH ---
"S-PAD:               13-16 (OCTAL)
"DATA PAD:          X - 0,  Y - 0   (RELATIVE TO DPA)

"                   --- ALGORITHM ---
"       WE USE THE ADDRESS BIT-REVERSING BUILT INTO THE S-PAD, WHICH
"       BIT-REVERSES THE SUBSCRIPT TO A COMPLEX DATA ARRAY (ELEMENTS
"         OCCUPY PAIRS OF MEMORY LOCATIONS SO THE LSB OF THE SUBSCRIPT
"         IS TAKEN TO BE 0.
"       WE SPECIFY THE ARRAY LENGTH (MUST BE A POWER OF 2) IN THE
"         STATUS REGISTER BITS 13-15: 15-LOG2(N)
"WE HAVE TWO SEPARATE LOOPS, DEPENDING UPON WHETHER N IS LESS THAN 256.
"       IF N IS SMALL WE SET THE STATUS BITS TO 7-LOG2(N), AND THEN
"         SHIFT THE SUBSCRIPT LEFT 8 PLACES BEFORE BIT-REVERSING, WHICH
"         TOGETHER GIVES THE PROPER BIT-REVERSED SUBSCRIPT.

"S - PAD REGISTER MNEMONICS
000000          BASE $EQU 0         "ARRAY BASE ADDRESS

000001          N $EQU 1            "N (NUMBER OF COMPLEX POINTS)

"
000013          I $EQU 13           "ARRAY POINTER

000014          T $EQU 14           "SHIFTED ARRAY POINTER

000015          DT $EQU 15                      "DELTA FOR T (256.)

000016          DI $EQU 16                      "DELTA FOR I (2.)
```

```
                        "
                        "
000000  042154  BITREV: MOVL N,I                "I=N*2   (I.E. IT'S A COMPLEX ARRAY)
        000000
        000000
        000000

000001  001660          LDSPI T; DB=256.
        000000
        002000
        000400

000002  031404          SUB# T,N                 "SEE IF N .GE. 256
        000040
        000000
        000000

000003  001670          BGE BIG;                 "YES, A 'BIG' ARRAY
        000722            LDSPI DI; DB=2.        "SET DELTA I TO 2
        002000
        000002

000004  011010          JMP SMALL                "NO, A SMALL ARRAY
        000000
        000000
        000021

                "ARRAY 256 ELEMENTS OR LARGER
                "BIT-REVERSE BITS OF STATUS REGISTER SET TO 15.-LOG2(N)
000005  131354  BIG:    SUB# &I,I                "TEST BITREV(I) AGAINST I
        000040
        000000
        000000

000006  031654          SUB DI,I;                "DECREMENT I ONE LOCATION
        000717            BGE BIG                "BRANCH IF BITREV(I).LT.I
        000000
        000000

                "WE COME HERE IF WE HAVE TO SWAP ELEMENTS.....................
000007  021654          ADD DI,I                 "BACK UP ONE PLACE
        000000
        000000
        000000

000010  021300          ADD# I,BASE; SETMA       "FETCH REAL(I)
        000040
        000000
        000060

000011  000000          INCMA                    "FETCH IMAG(I)
        000000
        000000
        000020
```

```
000012  121300          ADD# &I,BASE; SETMA              "FETCH REAL(BITREV(I))
        000040
        000000
        000060

000013  031655          INCMA;                           "FETCH IMAG(BITREV(I))
        154040            FADD ZERO,MD;                  "SAVE REAL(I)
        000000            SUB# DI,I                      "BACK UP T ONE PLACE
        000020

000014  000001          FADD ZERO,MD;                    "SAVE IMAG(I)
        154625            BEQ DONBIG                     "BRANCH IF DONE (T DOWN TO 0)
        000000
        000000

000015  121301          ADD# &I,BASE; SETMA; MI<FA;       "STORE NEW REAL(BITREV(I))
        154040            FADD ZERO,MD                    "SAVE OLD REAL(BITREV(I))
        000000
        000160

000016  000001          INCMA; MI<FA;                    "STORE NEW IMAG(BITREV(I))
        154000            FADD ZERO,MD                   "SAVE OLD IMAG(BITREV(I))
        000000
        000120

000017  021301          ADD# I,BASE; SETMA; MI<FA;       "STORE NEW REAL(I)
        100040            FADD
        000000
        000160

000020  031654          INCMA; MI<FA;                     "STORE NEW IMAG(I)
        000105            SUB DI,I; BR BIG                "RESTORE I AND LOOP
        000000
        000120

                "
                "FINISH LAST LOOP
000021  121301  DONBIG: ADD# &I,BASE; SETMA; MI<FA; FADD ZERO,MD
        154040
        000000
        000160

000022  000001          INCMA; MI<FA; FADD ZERO,MD
        154000
        000000
        000120

000023  021301          ADD# I,BASE; SETMA; MI<FA; FADD
        100040
        000000
        000160

000024  000000          INCMA; MI<FA; RETURN
        000340
        000000
```

```
                000120
                
                "
                "ARRAY SMALLER THAN 256 ELEMENTS
                "BIT-REVERSE FIELD OF STATUS REGISTER SET TO 7-LOG2(N)
000025  001664  SMALL:  LDSPI DT; DB=512.                  "DELTA FOR T
        000000
        002000
        001000

000026  043360          MOVL I,T                  "SHIFT I LEFT 8 PLACES TO GET T
        000000
        000000
        000000

000027  023460          ADDL T,T                          "***
        000000
        000000
        000000

000030  023460          ADDL T,T                          "***
        000000
        000000
        000000

000031  023460          ADDL T,T                          "***
        000000
        000000
        000000

000032  043460          MOVL T,T
        000000
        000000
        000000

000033  031560          SUB DT,T                          "DECREMENT T
        000000
        000000
        000000

000034  031654  LOOPS:  SUB DI,I                         "DECREMENT I
        000000
        000000
        000000

000035  131454          SUB# &T,I                         "COMPARE BITREV(T) AND I
        000040
        000000
        000000

000036  031560          SUB DT,T; BGE LOOPS               "BRANCH IF BITREV(T).LT.I)
        000716
        000000
        000000
```

```
                "SWAP ELEMENTS ..........
000037  021300          ADD# I,BASE; SETMA          "FETCH REAL(I)
        000040
        000000
        000060

000040  021560          INCMA; ADD DT,T             "FETCH IMAG(I), DECREMENT T
        000000
        000000
        000020

000041  121400          ADD# &T,BASE; SETMA         "FETCH REAL(BITREV(I))
        000040
        000000
        000060

000042  031655          INCMA;                      "FETCH IMAG(BITREV(I))
        154040            FADD ZERO,MD;             "SAVE REAL(I))
        000000            SUB# DI,I                 "TEST FOR DONE (I DOWN TO ZERO)
        000020

000043  000001          FADD ZERO,MD;               "SAVE IMAG(I)
        154625            BEQ DONSML                "BRANCH IF DONE
        000000
        000000

000044  121401          ADD# &T,BASE; SETMA; MI<FA; "STORE NEW REAL(BITREV(I))
        154040            FADD ZERO,MD              "SAVE OLD REAL(BITREV(I))
        000000
        000160

000045  000001          INCMA; MI<FA;               "STORE NEW IMAG(BITREV(I))
        154000            FADD ZERO,MD              "SAVE OLD IMAG(BITREV(I))
        000000
        000120

000046  021301          ADD# I,BASE; SETMA; MI<FA;  "STORE REAL(I)
        100040            FADD
        000000
        000160

000047  031560          INCMA; MI<FA;               "STORE IMAG(I)
        000105            SUB DT,T; BR LOOPS        "LOOP BACK
        000000
        000120

                "
                "FINISH OFF LAST LOOP
000050  121401  DONSML: ADD# &T,BASE; SETMA; MI<FA; FADD ZERO,MD
        154040
        000000
        000160

000051  000001          INCMA; MI<FA; FADD ZERO,MD
        154000
        000000
```

```
          000120

000052    021301          ADD# I,BASE; SETMA; MI<FA; FADD
          100040
          000000
          000160

000053    000000          INCMA; MI<FA; RETURN
          000340
          000000
          000120

                          $END


****      0 ERRORS    ****


SYMBOL    VALUE
BASE      000000
N         000001
I         000013
T         000014
DT        000015
DI        000016
BITREV    000000 ENT
BIG       000005
DONBIG    000021
SMALL     000025
LOOPS     000034
DONSML    000050
1
```

```
1APAL
 REV 1
 PASS 1
 PASS 2
                "RADIX 2 FIRST PASS FOR FAST (1 CYCLE) MEMORY
                "
                        $TITLE FFT2
                        $ENTRY FFT2
                "
                "                                       --- ABSTRACT ---
                "DOES THE FIRST RADIX 2 FFT PASS WHERE THE DATA ARRAY HAS A ODD
                "POWER OF TWO NUMBER OF POINTS
                "
                "
                "                                       --- STATISTICS ---
                "LANGIAGE:                              APAL
                "EQUIPMENT:                             AP-120B WITH FAST (1-CYCLE MEMORY)
                "SIZE:                                  16 PROGRAM LOCATIONS
                "AUTHOR:                                A.E. CHRLESWORTH
                "DATE:                                  JUNE 1976
                "REV 2
                "
                "
                "                                       --- USEAGE ---
                "       DOES THE FIRST PASS OF A COMPLEX FFT WHERE A RADIX 2 PASS IS NEEDED
                "
                "INPUT PARAMETERS:
                "       SPAD #    NAME    PURPOSE
                "       0         BASE    BASE ADDRESS OF ARRAY
                "       15        MDEL    MEMORY DELTA (BETWEEN COMPLEX POINTS)
                "       16        ICOUNT # OF PAIRS OF COMPLEX POINTS
                "
                "SCRATCH:
                "S-PAD:   3-5
                "DATA PAD X:  0-1  (RELATIVE TO DPA)
                "DATA PAD Y:  0-3  (RELATIVE TO DPA)
                "
                "
                "                                       --- ALGORITHM ---
                "
                "SINCE THIS IS THE FIRST PASS OF THE FFT, THE TWIDDLE FACTORS ARE 0.0,
                "  AND HENCE NO MULTIPLIES NEED BE DONE.
                "
                "THUS, THE 'BUTTERFLY' IS:
                "
                "       A' = A + C
                "       C' = A - C
                "
                "WHERE A AND C ARE COMPLEX POINTS
                "
                "
                "
                "DATA PAD NAMES:
                "       DATA PAD X:
000000                  CR $EQU 0               "C REAL
```

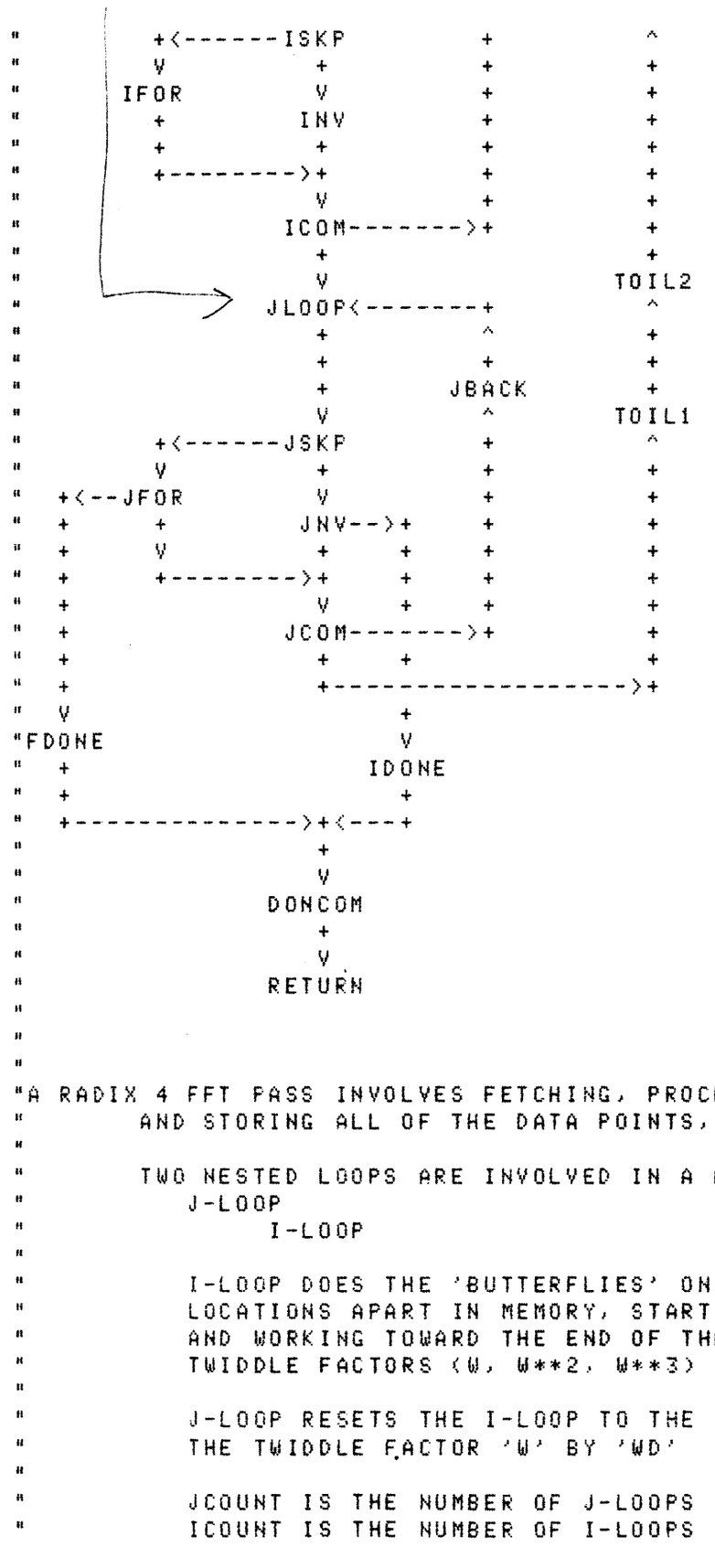
```
000001          CI $EQU 1               "C IMAGINARY
                "
                "       DATA PAD Y:
000000          AR $EQU 0               "A REAL
000001          AI $EQU 1               "A IMAGINARY
000002          AR1 $EQU 2
000003          AI1 $EQU 3
                "
                "
                "S-PAD NAMES:
000000          BASE $EQU 0             "BASE ADDRESS OF ARRAY
000003          READ $EQU 3             "READ POINTER
000004          WRITE $EQU 4            "WRITE POINTER
000005          ICTR $EQU 5             "LOOP COUNTER
000015          MDEL $EQU 15            "MEMORY DELTA
000016          ICOUNT $EQU 16          "LOOP COUNT
                "
                "
                "
                "INTRO TO THE LOOP ...
                "INITIALIZE POINTERS:
                "READ<BASE
                "WRITE<BASE-MDEL
                "ICTR<ICOUNT
                "
000000  040014  FFT2:   MOV BASE,READ; SETMA             "1. FETCH AR
        000000
        000000
        000060
000001  040020          INCMA;                          "2. FETCH AI
        000000          MOV BASE,WRITE
        000000
        000020
000002  021514          ADD MDEL,READ; SETMA            "3. FETCH CR
        000000
        000000
        000060
000003  031520          INCMA;                          "4. FETCH CI
        000000          DPY(AR1)<MD;
        015000          SUB MDEL,WRITE                  "SET WRITE=BASE-MDEL
        140020
```

204

```
000004  041624          DPY(AI)<MD;                                 "5.
        000000          MOV ICOUNT,ICTR                             "SET LOOP COUNTER
        015000
        120000

000005  000000          DPX(CR)<MD                                  "6.
        000000
        045004
        000000

000006  021514          ADD MDEL,READ; SETMA;
        000000                                  DPX(CI)<MD          "1. FETCH AR
        045005
        000060

                "
                "
                "THE LOOP .....
000007  000000  LOOP:    INCMA;                                      "2. FETCH AI
        000000                                  DPY(AI1)<DPY(AI)
        014050
        160020

000010  021515          ADD MDEL,READ; SETMA;                       "3. FETCH CR
        132000                                  FADD DPY(AR1),DPX(CR)   "AR'=AR+CR
        000460
        000060

000011  000001          INCMA;                                      "4. FETCH CI
        132000          DPY(AR)<MD;
        015570                                  FADD DPY(AI1),DPX(CI)   "AI'=AI+CI
        100020

000012  021521          DPY(AI)<MD;                                 "5.
        032000                                  FSUB DPY(AR1),DPX(CR);  "CR'=AR-CR
        015460                                  MI<FA; ADD MDEL,WRITE; SETMA "STORE AR'
        120160

000013  000001          DPX(CR)<MD;                                 "6.
        032000                                  FSUB DPY(AI1),DPX(CI);  "CI'=AI-CI
        045574                                  MI<FA; INCMA            "STORE AI'
        000120

000014  021521          DPX(CI)<MD;                                 "7.
        100000                                  MI<FA; ADD MDEL,WRITE; SETMA;"STORE CR'
        045005                                  FADD
        000160

000015  001224                                  MI<FA; INCMA;           "8. STORE CI'
        000000                                  DEC ICTR                "DONE ???
        000000
        000120

000016  021514          ADD MDEL,READ; SETMA;                       "1. FETCH AR
        000651                                  DPY(AR1)<DPY(AR);
        014040                                  BNE LOOP                 "GO BACK
```

```
         140060

                  *
                  *
                  "DONE WITH LOOP .....
000017   000000           RETURN
         000340
         000000
         000000

                           $END


****      0 ERRORS   ****


SYMBOL   VALUE
CR       000000
CI       000001
AR       000000
AI       000001
AR1      000002
AI1      000003
BASE     000000
READ     000003
WRITE    000004
ICTR     000005
MDEL     000015
ICOUNT   000016
FFT2     000000 ENT
LOOP     000007
```

```
1APAL
 REV 1
 PASS 1
 PASS 2
                "RADIX 4 FFT PASS FOR FAST (1-CYCLE MEMORY)
                "
                        $TITLE FFT4
                        $ENTRY FFT4
                "
                "
                "
                "                               --- ABSTRACT ---
                "DOES ONE RADIX 4 COMPLEX FFT PASS
                "
                "
                "                               --- STASTICS ---
                "LANGUAGE:                      APAL
                "EQUIPMENT:                     AP-120B WITH FAST (1 CYCLE) MEMORY
                "PROGRAM SIZE:                  81 LOCATIONS
                "AUTHOR:                        A.E. CHARLESWORTH
                "DATE:  JUNE 76
                "REVISION 2
                "
                "
                "                               --- USEAGE ---
                "
                "DOES ONE RADIX 4 FFT PASS, AS DETERMINED BY:
                "
                "       S-PAD REG    NAME       PURPOSE
                "
                "       14           WD         W (TWIDDLE FACTOR) DELTA
                "       15           MDEL       MEMORY DELTA
                "       16           ICOUNT     NUMBER OF I-LOOPS IN THIS PASS
                "       17           JCOUNT     NUMBER OF J-LOOPS IN THIS PASS
                "
                "
                "SCRATCH:
                "       S-PAD:  2 TO 12
                "       DATA PAD X:  -4 TO +3 (RELATVE TO DPA)
                "       DATA PAD Y:  -4 TO +3
                "
                "
                "                                   --- ALGORITHM ---
                "
                "PROGRAM FLOW:
                "
                "                   FFT4
                "                     +
                "                     V
                "                  ILOOP<--------+
                "                     +          ^
                "                     V          +
                "                  ILOOP4<-----------------+
                "                     +          +         ^
                "                     +       IBACK        +
                "                     V          ^       TOIL3
```



"A RADIX 4 FFT PASS INVOLVES FETCHING, PROCESSING,
" AND STORING ALL OF THE DATA POINTS, 4 COMPLEX POINTS AT A TIME.
"
" TWO NESTED LOOPS ARE INVOLVED IN A PASS:
" J-LOOP
" I-LOOP
"
" I-LOOP DOES THE 'BUTTERFLIES' ON QUADS OF POINTS, 'MDEL'
" LOCATIONS APART IN MEMORY, STARTING AT THE START OF THE ARRAY
" AND WORKING TOWARD THE END OF THE ARRAY. A FIXED SET OF
" TWIDDLE FACTORS (W, W**2, W**3) IS USED THROUGHOUT AN I-LOOP
"
" J-LOOP RESETS THE I-LOOP TO THE TOP OF THE ARRAY AND ADVANCES
" THE TWIDDLE FACTOR 'W' BY 'WD'
"
" JCOUNT IS THE NUMBER OF J-LOOPS
" ICOUNT IS THE NUMBER OF I-LOOPS

```
"
"
"
"THE RADIX 4 BUTTERFLY IS AS FOLLOWS:
"
"1.      FOUR COMPLEX POINTS:  A, C, B, D
"        ARE FETCHED IN THAT ORDER FROM MEMORY
"
"2.      THREE COMPLEX MULTIPLIES ARE DONE:
"        B = B * W
"        C = C * W**2
"        D = D * W**3
"
"
"3.      THEN THE FOLLOWING SET OF COMPLEX ADDS & SUBTRACTS ARE DONE
"        A' = A +  B + C +  D
"        B' = A -+JB - C +-JD
"        C' = A -  B + C -  D
"        D' = A +-JB - C -+JD
"
"            -+ MEANS - FOR FORWARD FFT, + FOR INVERSE FFT
"            +- MEANS + FOR FORWARD FFT, - FOR INVERSE FFT
"
"
"4.      A', B', C', D' ARE THEN STORED BACK INTO MEMORY
"
"
"TO MINIMIZE COMPLEX ADDS THE COMPUTATION IN THE BUTTERFLY IS
"        FACTORED AS FOLLOWS:
"
"
"        APC = A + C
"        AMC = A - C
"        BPD = B + D
"        JBMD - J * (B - D) = (DI-BI, BR-DR)
"
"        A' = APC + BPD
"        C' = APC - BPD
"
"DIRECT FFT:
"        B' = AMC - JBMD
"        D' = AMC + JBMD
"
"INVERSE FFT:
"        B' = AMC + JBMD
"        D' = AMC - JBMD
"
"
"THIS GIVES A TOTAL OF 22 REAL ADDS AND 12 REAL MULTIPLIES
"
"
"
"OUTLINE OF THE INNER LOOP COMPUTATION:
"
"1.      FETCH AR                    AMCR=AR-CR
"                                    SAVE APCR
```

```
"
"2.     FETCH AI                AMCI=AI-CI
"                               SAVE APCI
"
"3.     FETCH CR                BR=BRR-BII
"                               SAVE AMCR
"
"4.     FETCH W2R               BI=BRI+BIR
"       SAVE AR                 SAVE AMCI
"
"5.     FETCH CI                DR=DRR-DII
"       FETCH W2I               SAVE BR
"       SAVE AI
"
"6.     CRR=W2R*CR              DI=DRI+DIR
"                               SAVE BI
"
"7.     FETCH BR                BPDR=BR+DR
"       FETCH W2R               SAVE DR
"       CIR=W2I*CR
"
"8.     CII=W2I*CI              BPDI=BI+DI
"       FETCH W1R               SAVE DI
"
"9.     FETCH BI                JBMDR=DI-BI
"       CRI=W2R*CI              SAVE BPDR
"       FETCH W1I
"       SAVE CRR
"
"10.    BRR=W1R*BR              JBMDI=BR-DR
"       SAVE CIR                SAVE BPDI
"
"11.    FETCH DR                SAVE JBMDR
"       BIR=W1I*BR
"       CR=CRR-CII
"       FETCH W1R
"
"12.    BII=W1I*BI              SAVE JBMDI
"       CI=CRI+CIR
"       FETCH W3R
"
"13.    FETCH DI                AR'=APCR+BPDR
"       BRI=W1R*BI
"       FETCH W3I
"       SAVE BRR
"       SAVE CR
"
"14.    DRR=W3R*DR              AI'=APCI+BPDI
"       SAVE BIR
"       SAVE CI
"
"15.    DIR=W3I*DR              BR'=AMCR-+JBMDR
"       FETCH W3R               STORE AR
"       SAVE BII
"
"16.    DII=W3I*DI              BI'=AMCI-+JBMDI
```

```
                "       SAVE BRI                STORE AI'
                "
                "17.    DRI=W3R*DI              CR'=APCR-BPDR
                "       SAVE DRR                STORE BR'
                "
                "18.    SAVE DIR                CI'=APCI-BPDI
                "                               STORE BI'
                "
                "19.    SAVE DII                DR'=AMCR+-JBDMR
                "                               STORE CR'
                "
                "20.    SAVE DRI                DI'=AMCI+-JBMDI
                "                               STORE CI'
                "
                "21.    APCR=AR+CR              STORE DR'
                "
                "22.    APCI=AI+CI              STORE DI'
                "
                "
                "
                "
                "DATA PAD X NAMES:
177774          BPDR $EQU -4            "9 THROUGH 17     BR + DR

177774          DRR  $EQU -4            "17 THROUGH 5     W3R * DR

177775          BPDI $EQU -3            "10 THROUGH 18    BI + DI

177775          DIR  $EQU -3            "18 THROUGH 6     W3I * DR

177776          AMCR $EQU -2            "                  AR - CR

177777          AMCI $EQU -1            "                  AI - CI

000000          CR   $EQU  0            "                  C REAL

000001          CI   $EQU  1            "                  C IMAGINARY

000002          BBR  $EQU  2            "5 THROUGH 10      B REAL

000002          BII  $EQU  2            "15 THROUGH 3      W1I * BI

000003          BI   $EQU  3            "6 THROUGH 9       B IMAGINARY

000003          BRI  $EQU  3            "14 THROUGH 4      W1R * BI


        "DATA PAD Y NAMES:
177774          APCR $EQU -4            "                  AR + CR

177775          APCI $EQU -3            "                  AI + CI

177776          JBMDR $EQU -2           "11 THROUGH 19     J * (BR - DR)
```

```
177776          DII  $EQU -2          "19 THROUGH 5     W3I * DI
177776          CRR  $EQU -2          "9 THROUGH 11     W2R * CR
177777          JBMDI $EQU -1         "12 THROUGH 20    J * (BI - DR)
177777          DRI  $EQU -1          "20 THROUGH 6     W3R * DI
177777          CIR  $EQU -1          "10 THROUGH 12    W2I * CR
000000          AR   $EQU  0          "                 A REAL
000001          AI   $EQU  1          "                 A IMAGINARY
000002          DR   $EQU  2          "7 THROUGH 10     D REAL
000002          BRR  $EQU  2          "13 THROUGH 3     W1R * BR
000003          DI   $EQU  3          "8 THROUGH 9      D IMAGINARY
000003          BIR  $EQU  3          "16 THROUGH 4     W1I * BR


        "S-PAD NAMES:
000000          BASE   $EQU  0        "BASE ADDRESS OF ARRAY
000001          N      $EQU  1        "# OF COMPLEX POINTS
000002          JBASE  $EQU  2        "BASE FOR EACH J-LOOP
000003          READ   $EQU  3        "READ POINTER
000004          WRITE  $EQU  4        "WRITE POINTER
000005          ICTR   $EQU  5        "I-LOOP COUNTER
000006          JCTR   $EQU  6        "J-LOOP COUNTER
000007          TEMP   $EQU  7        "TEMPORARY
000010          N2     $EQU  10       "CONTAINS 2
000011          W1     $EQU 11        "W**1 POINTER
000012          W2     $EQU 12        "W**2 POINTER
000013          W3     $EQU 13        "W**3 POINTER
000014          WD     $EQU 14        "W DELTA
000015          MDEL   $EQU 15        "MEMORY DELTA
000016          ICOUNT $EQU 16        "I-LOOP COUNT
```

```
000017          JCOUNT $EQU 17          "J-LOOP COUNT


                "INTRO TO ILOOP
                "
                "INITIALIZE POINTERS:
                "
                "JBASE<BASE
                "READ<JBASE
                "WRITE<JBASE-MDEL
                "ICTR<ICOUNT
                "JCTR<JCOUNT
                "W1<0
                "W2<0
                "W3<0
                "
                "
                "START COMPUTATION:
                "
                "FETCH AR,AI,CR,CI
                "SET: BRR=BR, BRI=BI, DRR=DR, DRI=DI
                "SET: BIR=0.0, BII=0.0, DIR=0.0, DII=0.0
                "
                "
000000  040010  FFT4:   MOV BASE,JBASE                  "SET JBASE
        000000
        000000
        000000

000001  041624          MOV ICOUNT,ICTR                 "SET ICTR
        000000
        000000
        000000

000002  040214          MOV JBASE,READ; SETMA           "FETCH AR
        000000
        000000
        000060

000003  040220          INCMA;                          "FETCH AI
        000000             MOV JBASE,WRITE;
        050006             DPX(BII)<ZERO;               "BII=0.0
        160020             DPY(BIR)<ZERO                "BIR=0.0

000004  021514          ADD MDEL,READ; SETMA;           "FETCH CR
        000000             DPX(DIR)<ZERO;               "DIR=0.0
        050001             DPY(DII)<ZERO                "DII=0.0
        040060

000005  031520          INCMA;                          "FETCH CI
        000000             DPY(AR)<MD;
        015000             SUB MDEL,WRITE               "SET WRITE
        100020
```

224

```
000006  021514          ADD MDEL,READ; SETMA;                          "FETCH BR
        000000             DPY(AI)<MD
        015000
        120060

000007  001044          INCMA;                                         "FETCH BI
        000000             DPX(CR)<MD;
        045004             CLR W1                                      "CLEAR W1
        000020

000010  021514          ADD MDEL,READ; SETMA;                          "FETCH DR
        000000             DPX(CI)<MD
        045005
        000060

000011  001050          INCMA;                                         "FETCH DI
        000000             DPY(BRR)<MD;
        015000             CLR W2                                      "CLEAR W2
        140020

000012  001054          DPX(BRI)<MD;
        000000             CLR W3                                      "CLEAR W3
        045007
        000000

000013  001225          DPX(DRR)<MD;
        132000             FADD DPY(AR),DPX(CR);                       "APCR=AR+CR
        045440             DEC ICTR                                    "GO TO J-LOOP ???
        000000

000014  041731          DPY(DRI)<MD;
        132662             FADD DPY(AI),DPX(CI);                       "APCI=AI+CI
        015550             MOV JCOUNT,JCTR;                            "SET JCTR
        060000                                                         "TO I-LOOP
                           BNE ILOOP                                   "IF NOT YET ZERO

                "
                "
                "IF ICTR ALREADY ZERO, GO TO J-LOOP
000015  011010          JMP JLOOP
        000000
        000000
        000040

                "
                "
                "
000016  021515  ILOOP:  ADD MDEL,READ; SETMA;                          "1. FETCH AR
        032000                                FSUB DPY(AR),DPX(CR);    "AMCR=AR-CR
        020440                                DPY(APCR)<FA
        000060

000017  000001          INCMA;                                         "2. FETCH AI
        032000                                FSUB DPY(AI),DPX(CI);    "AMCI=AI-CI
        020550                                DPY(APCI)<FA
        020020
```

```
000020  021515          ADD MDEL,READ; SETMA;                                "3. FETCH CR
        032000                                FSUB DPY(BRR),DPX(BII); "BR=BRR-BII
        100662                                DPX(AMCR)<FA
        000060

000021  041251  ILOOP4: MOV W2,W2; SETTMA;                                  "4. FETCH W2R
        123000          DPY(AR)<MD;
        115773                                FADD DPX(BRI),DPY(BIR); "BI=BRI+BIR
        100003                                DPX(AMCI)<FA

000022  000001          INCMA;                                               "5. FETCH CI
        023000          INCTMA;                                              "FETCH W2I
        115026          DPY(AI)<MD;
        120021                                FSUB DPX(DRR),DPY(DII); "DR=DRR-DII
                                              DPX(BBR)<FA

000023  000001          FMUL TM,MD;                                          "6. W2R*CR
        132000                                FADD DPY(DRI),DPX(DIR); "DI=DRI+DIR
        100137                                DPX(BI)<FA
        017400

000024  021515          ADD MDEL,READ; SETMA;                                "7. FETCH BR
        121000          DECTMA;                                              "FETCH W2R
        020600          FMUL TM,MD;                                          "CIR=W2I*CR
        157462                                FADD DPX(BBR),FA;       "BPDR=BR+DR
                                              DPY(DR)<FA

000025  041145          MOV W1,W1; SETTMA;                                   "8. FETCH W1R
        121000          FMUL TM,MD;                                          "CII=W2I*CI
        020700                                FADD DPX(BI),FA;        "BPDI=BI+DI
        177403                                DPY(DI)<FA

000026  000001          INCMA;                                               "9. FETCH BI
        032000          INCTMA;                                              "FETCH W1I
        130770          FMUL TM,MD;                                          "CRI=W2R*CI
        057421          DPY(CRR)<FM;
                                              FSUB DPY(DI),DPX(BI);   "JBMDR=DI-BI
                                              DPX(BPDR)<FA

000027  000001          FMUL TM,MD;                                          "10. BRR=W1R*BR
        023000          DPY(CIR)<FM;
        130661                                FSUB DPX(BBR),DPY(DR);  "JBMDI=BR-DR
        077400                                DPX(BPDI)<FA

000030  021514          ADD MDEL,READ; SETMA;                                "11. FETCH DR
        113000          DECTMA;                                              "FETCH W1R
        020020          FMUL TM,MD;                                          "BIR=W1I*BR
        057462          FSUBR FM,DPY(CRR);                                   "CR=CRR-CII
                                              DPY(JBMDR)<FA

000031  041355          MOV W3,W3; SETTMA;                                   "12. FETCH W3R
        113000          FMUL TM,MD;                                          "BII=W1I*BI
        020030          FADD FM,DPY(CIR);                                    "CI=CRI+CIR
        077403                                DPY(JBMDI)<FA
```

```
000032  000001          INCMA;                                          "13. FETCH DI
        132123          INCTMA;                                         "FETCH W3I
        130004          FMUL TM,MD;                                     "BRI=W1R*BI
        157421          DPY(BRR)<FM;
                        DPX(CR)<FA;
                                        FADD DPY(APCR),DPX(BPDR);"AR'=APCR+BPDR
                                        BR ISKP                 "SKIP AROUND

                "
                "
                "
                ".....................................................................
                "THIS INSTRUCTION IS ACTUALLY THE LAST
                "INSTRUCTION OF ILOOP, IT IS INSERTED HERE SO THAT THE LOOP CAN
                "BE CLOSED WITH 'BRANCH' OP-CODES, WHICH HAVE A MAXIMUM
                "RANGE OF -16 TO + 15
                "
000033  000001  IBACK:  FADD DPY(AI),DPX(CI);                          "22. APCI=AI+CI
        132103                          INCMA; MI<FA;           "STORE DI'
        000550                          BR ILOOP                "LOOP AGAIN
        000120

                "
                "
                "HERE IS TOIL3 ON THE WAY TO ILOOP AT ILOOP4
                "
000034  021515  TOIL3:  ADD MDEL,READ; SETMA;                          "3. FETCH CR
        032105                          FSUB DPY(BRR),DPX(BII); "BR=BRR-BII
        100662                          DPX(AMCR)<FA;
        000060                          BR ILOOP4

                "
                ".....................................................................
                "
                "
                "
                "
000035  010031  ISKP:   FMUL TM,MD;                                    "14. DRR=W3R*DR
        132027          DPY(BIR)<FM;
        130115          DPX(CI)<FA;
        177400                          FADD DPY(APCI),DPX(BPDI);"AI'=APCI+BPDI
                                        BIFN INV                "INVERSE FFT???

                "
                "FORWARD FFT, COMPUTE B', C', AND D'
000036  021521  IFOR:   DECTMA;                                       "15. FETCH W3R
        023000          FMUL TM,MD;                                     "DIR=W3I*DR
        140226          DPX(BII)<FM;
        017562                          FSUB DPX(AMCR),DPY(JBMDR); "BR'=AMCR-JBMDR
                                        ADD MDEL,WRITE; SETMA; MI<FA "STORE AR'

000037  000001          FMUL TM,MD;                                     "16. DII=W3I*DI
        023000          DPX(BRI)<FM;
        140337                          FSUB DPX(AMCI),DPY(JBMDI); "BI'=AMCI-JBMDI
        017520                          INCMA; MI<FA            "STORE AI'
```

```
000040  021521          FMUL TM,MD;                                 "17. DRI=W3R*DI
        032000          DPX(DRR)<FM;
        140000                              FSUB DPY(APCR),DPX(BPDR); "CR'=APCR-BPDR
        017560                              ADD MDEL,WRITE; SETMA; MI<FA "STORE BR'

000041  000001          DPX(DIR)<FM;                                "18.
        032000          FMUL;                                       "PUSH
        140111                              FSUB DPY(APCI),DPX(BPDI); "CI'=APCI-BPDI
        010120                              INCMA; MI<FA            "STORE BI'

000042  021521          DPY(DII)<FM;                                "19.
        123000          FMUL;                                       "PUSH
        030220                              FADD DPX(AMCR),DPY(JBMDR); "DR'=AMCR+JBMDR
        050160                              ADD MDEL,WRITE; SETMA; MI<FA "STORE CR

000043  001225          DPY(DRI)<FM;                                "20.
        123127                              FADD DPX(AMCI),DPY(JBMDI); "DI'=AMCI+JBMDI
        030330                              INCMA; MI<FA;           "STORE CI'
        060120                              DEC ICTR;               "ILOOP DONE???
                                            BR ICOM                 "BACK TOGETHER

                "
                "
                "INVERSE FFT: COMPUTE B',C', AND D'
000044  021521  INV:    DECTMA;                                     "15. FETCH W3R
        123000          FMUL TM,MD;                                 "DIR=W3I*DR
        140226          DPX(BII)<FM;
        017562                              FADD DPX(AMCR),DPY(JBMDR); "BR'=AMCR+JBMDR
                                            ADD MDEL,WRITE; SETMA; MI<FA "STORE AR'

000045  000001          FMUL TM,MD;                                 "16. DII=W3I*DI
        123000          DPX(BRI)<FM;
        140337                              FADD DPX(AMCI),DPY(JBMDI); "BI'=AMCI+JBMDI
        017520                              INCMA; MI<FA            "STORE AI'

000046  021521          FMUL TM,MD;                                 "17. DRI=W3R*DI
        032000          DPX(DRR)<FM;
        140000                              FSUB DPY(APCR),DPX(BPDR); "CR'=APCR-BPDR
        017560                              ADD MDEL,WRITE; SETMA; MI<FA "STORE BR'

000047  000001          DPX(DIR)<FM;                                "18.
        032000          FMUL;                                       "PUSH
        140111                              FSUB DPY(APCI),DPX(BPDI); "CI'=APCI-BPDI
        010120                              INCMA; MI<FA            "STORE BI'

000050  021521          DPY(DII)<FM;                                "19.
        023000          FMUL;                                       "PUSH
        030220                              FSUB DPX(AMCR),DPY(JBMDR); "DR'=AMCR-JBMDR
        050160                              ADD MDEL,WRITE; SETMA; MI<FA "STORE CR

000051  001225          DPY(DRI)<FM;                                "20.
        023000                              FSUB DPX(AMCI),DPY(JBMDI); "DI'=AMCI-JBMDI
        030330                              INCMA; MI<FA;           "STORE CI'
        060120                              DEC ICTR                "ILOOP DONE ???

                "
```

```
                    "
                    "BACK TOGETHER AGAIN
000052  021521      ICOM:   FADD DPY(AR),DPX(CR);                       "21. APCR=AR+CR
        132641                                  ADD MDEL,WRITE; SETMA; MI(FA;"STORE DR'
        000440                                  BNE IBACK               "BRANCH IF NOT DONE
        000160
                    "
                    "
                    "
                    "COME HERE WHEN DONE WITH ILOOP,
                    "       DO #22 AND THEN FALL INTO JLOOP
                    "
000053  000001              FADD DPY(AI),DPX(CI);                       "22. APCI=AI+CI
        132122                                  INCMA; MI(FA;           "STORE DI'
        000550                                  BR JLOOP
        000120
                    "
                    "
                    "
                    "...............................................................
                    "
                    "HERE IS TOIL2 ON THE WAY BACK TO ILOOP
                    "
000054  000001      TOIL2:  INCMA;                                      "2. FETCH AI
        032100                                  FSUB DPY(AI),DPX(CI);   "AMCI=AI-CI
        020550                                  DPY(APCI)(FA;
        020020                                  BR TOIL3
                    "
                    "...............................................................
                    "
                    "
                    "
                    "
                    "RESETING OF POINTERS FOR J-LOOP
                    "
                    "JBASE(JBASE+2
                    "READ(JBASE
                    "W1(W1+WD
                    "W2(W1*2
                    "W3(W1*3
                    "ICTR(ICOUNT
                    "
                    "
                    "WRITE(JBASE-MDEL
                    "
000055  021011      JLOOP:  ADD N2,JBASE;  SETMA;                       "1. FETCH AR
        032000                                                          "ADVANCE JBASE
        020440                                  FSUB DPY(AR),DPX(CR);   "AMCR=AR-CR
        000060                                  DPY(APCR)(FA

000056  021445              INCMA;                                      "2. FETCH AI
        032000              ADD WD,W1;                                  "ADVANCE W1
        020550                                  FSUB DPY(AI),DPX(CI);   "AMCI=AI-CI
```

```
        020020                                  DPY(APCI)<FA

000057  021511          ADD MDEL,JBASE; SETMA;  
        032000                                  "3. FETCH CR
        100662                                  "ADVANCE JBASE ONE
        000060                                  "STEP TOO FAR
                                FSUB DPY(BRR),DPX(BII); "BR=BRR-BII
                                DPX(AMCR)<FA

000060  043151          MOVL W1,W2; SETTMA;
        123000                                  "4. FETCH W2R
        115773          DPY(AR)<MD;             "W2=W1*2
        100003
                                FADD DPX(BRI),DPY(BIR); "BI=BRI+BIR
                                DPX(AMCI)<FA

000061  040215          INCMA;
        023000          MOV JBASE,READ;         
        115026          INCTMA;                 "5. FETCH CI
        120021          DPY(AI)<MD;             "SET READ=JBASE
                                                "FETCH W2I
                                FSUB DPX(DRR),DPY(DII); "DR=DRR-DII
                                DPX(BBR)<FA

000062  031511          FMUL TM,MD;
        132000          SUB MDEL,JBASE;         "6. W2R*CR
        100137                                  "RESET JBASE
                                FADD DPY(DRI),DPX(DIR); "DI=DRI+DIR
                                DPX(BI)<FA

000063  021515          ADD MDEL,READ; SETMA;
        121000          DECTMA;                 "7. FETCH BR
        020600          FMUL TM,MD;             "FETCH W2R
        157462                                  "CIR=W2I*CR
                                FADD DPX(BBR),FA;       "BPDR=BR+DR
                                DPY(DR)<FA

000064  041145          MOV W1,W1; SETTMA;
        121000                                  "8. FETCH W1R
        020700          FMUL TM,MD;             "CII=W2I*CI
        177403                                  "BPDI=BI+DI
                                FADD DPX(BI),FA;
                                DPY(DI)<FA

000065  040235          INCMA;
        032000          MOV JBASE,TEMP;         "9. FETCH BI
        130770          INCTMA;                 "TO SETUP WRITE
        057421          FMUL TM,MD;             "FETCH W1I
                        DPY(CRR)<FM;            "CRI=W2R*CI
                                FSUB DPY(DI),DPX(BI);   "JBMDR=DI-BI
                                DPX(BPDR)<FA

000066  041255          FMUL TM,MD;
        023000          MOV W2,W3;              "10. BRR=W1R*BR
        130661                                  "TO SETUP W3
        077400          DPY(CIR)<FM;            "TO W1+W2
                                FSUB DPX(BBR),DPY(DR);  "JBMDI=BR-DR
                                DPX(BPDI)<FA

000067  021514          ADD MDEL,READ; SETMA;
        113000          DECTMA;                 "11. FETCH DR
        020020          FMUL TM,MD;             "FETCH W1R
        057462          FSUBR FM,DPY(CRR);      "BIR=W1I*BR
                                                "CR=CRR-CII
```

```
                                                DPY(JBMDR)<FA

000070  021155          ADD W1,W3; SETTMA;                              "12. FETCH W3R
        113000                                                          "W3=W1*3
        020030          FMUL TM,MD;                                     "BII=W1I*BI
        077403          FADD FM,DPY(CIR);                               "CI=CRI+CIR
                                                DPY(JBMDI)<FA

000071  031535          INCMA;                                          "13. FETCH DI
        132123          SUB MDEL,TEMP;                                  "SETUP WRITE
        130004          INCTMA;                                         "FETCH W3I
        157421          FMUL TM,MD;                                     "BRI=W1R*BI
                        DPY(BRR)<FM;
                        DPX(CR)<FA;
                                        FADD DPY(APCR),DPX(BPDR);"AR'=APCR+BPDR
                                        BR JSKP                         "SKIP AROUND

                "
                "
                "
                "..............................................................................
                "THIS INSTRUCTION IS ACTUALLY THE LAST
                "INSTRUCTION OF JLOOP, IT IS INSERTED HERE SO THAT THE LOOP CAN
                "BE CLOSED WITH 'BRANCH' OP-CODES, WHICH HAVE A MAXIMUM
                "RANGE OF -16 TO + 15
                "
000072  040721  JBACK:  FADD DPY(AI),DPX(CI);                           "22. APCI=AI+CI
        132103          MOV TEMP,WRITE;                                 "WRITE=JBACK-MDEL
        000550                                  INCMA; MI<FA;           "STORE DI'
        000120                                  BR JLOOP                "LOOP AGAIN

                "
                "
                "HERE IS TOIL1 ON THE WAY BACK TO ILOOP
                "
000073  021515  TOIL1:  ADD MDEL,READ; SETMA;                           "1. FETCH AR
        032101                                  FSUB DPY(AR),DPX(CR);   "AMCR=AR-CR
        020440                                  DPY(APCR)<FA;
        000060                                  BR TOIL2

                "
                "..............................................................................
                "
                "
                "
                "
000074  010031  JSKP:    FMUL TM,MD;                                   "14. DRR=W3R*DR
        132027          DPY(BIR)<FM;
        130115          DPX(CI)<FA;
        177400                                  FADD DPY(APCI),DPX(BPDI);"AI'=APCI+BPDI
                                                BIFN JNV                "INVERSE FFT???

                "
                "FORWARD FFT, COMPUTE B', C', AND D'
000075  021521  JFOR:    DECTMA;                                       "15. FETCH W3R
        023000          FMUL TM,MD;                                     "DIR=W3I*DR
```

```
                140226          DPX(BII)<FM;
                017562                                  FSUB DPX(AMCR),DPY(JBMDR); "BR'=AMCR-JBMDR
                                                        ADD MDEL,WRITE; SETMA; MI<FA "STORE AR'

000076          041625          FMUL TM,MD;                                     "16. DII=W3I*DI
                023000          MOV ICOUNT,ICTR;                                "RESET ICTR=ICOUNT
                140337          DPX(BRI)<FM;
                017520                                  FSUB DPX(AMCI),DPY(JBMDI); "BI'=AMCI-JBMDI
                                                        INCMA; MI<FA            "STORE AI'

000077          021521          FMUL TM,MD;                                     "17. DRI=W3R*DI
                032000          DPX(DRR)<FM;
                140000                                  FSUB DPY(APCR),DPX(BPDR); "CR'=APCR-BPDR
                017560                                  ADD MDEL,WRITE; SETMA; MI<FA "STORE BR'

000100          001231          DPX(DIR)<FM;                                    "18.
                032000          FMUL;                                           "PUSH
                140111                                  FSUB DPY(APCI),DPX(BPDI); "CI'=APCI-BPDI
                010120                                  INCMA; MI<FA;           "STORE BI'
                                                        DEC JCTR                "JLOOP DONE ???

000101          021521          DPY(DII)<FM;                                    "19.
                123632          FMUL;                                           "PUSH
                030220                                  FADD DPX(AMCR),DPY(JBMDR); "DR'=AMCR+JBMDR
                050160                                  ADD MDEL,WRITE; SETMA; MI<FA;"STORE CR
                                                        BEQ FDONE               "BRANCH IF JLOOP DONE

000102          001225          DPY(DRI)<FM;                                    "20.
                123127                                  FADD DPX(AMCI),DPY(JBMDI); "DI'=AMCI+JBMDI
                030330                                  INCMA; MI<FA;           "STORE CI'
                060120                                  DEC ICTR;               "ILOOP DONE???
                                                        BR JCOM                 "BACK TOGETHER

                "
                "
                "INVERSE FFT: COMPUTE B',C', AND D'
000103          021521  JNV:    DECTMA;                                         "15. FETCH W3R
                123000          FMUL TM,MD;                                     "DIR=W3I*DR
                140226          DPX(BII)<FM;
                017562                                  FADD DPX(AMCR),DPY(JBMDR); "BR'=AMCR+JBMDR
                                                        ADD MDEL,WRITE; SETMA; MI<FA "STORE AR'

000104          041625          FMUL TM,MD;                                     "16. DII=W3I*DI
                123000          MOV ICOUNT,ICTR;                                "RESET ICTR=ICOUNT
                140337          DPX(BRI)<FM;
                017520                                  FADD DPX(AMCI),DPY(JBMDI); "BI'=AMCI+JBMDI
                                                        INCMA; MI<FA            "STORE AI'

000105          021521          FMUL TM,MD;                                     "17. DRI=W3R*DI
                032000          DPX(DRR)<FM;
                140000                                  FSUB DPY(APCR),DPX(BPDR); "CR'=APCR-BPDR
                017560                                  ADD MDEL,WRITE; SETMA; MI<FA "STORE BR'

000106          001231          DPX(DIR)<FM;                                    "18.
                032000          FMUL;                                           "PUSH
                140111                                  FSUB DPY(APCI),DPX(BPDI); "CI'=APCI-BPDI
```

```
         010120                                  INCMA; MI<FA;           "STORE BI'
                                                 DEC JCTR                "JLOOP DONE ???

000107   021521          DPY(DII)<FM;                                    "19.
         023625          FMUL;                                           "PUSH
         030220                                 FSUB DPX(AMCR),DPY(JBMDR); "DR'=AMCR-JBMDR
         050160                                 ADD MDEL,WRITE; SETMA; MI<FA;"STORE CR
                                                BEQ IDONE               "BRANCH IF JLOOP DONE

000110   001225          DPY(DRI)<FM;                                    "20.
         023000                                 FSUB DPX(AMCI),DPY(JBMDI); "DI'=AMCI-JBMDI
         030330                                 INCMA; MI<FA;           "STORE CI'
         060120                                 DEC ICTR                "ILOOP DONE ???

                 "
                 "
                 "BACK TOGETHER AGAIN
000111   021521  JCOM:   FADD DPY(AR),DPX(CR);                          "21. APCR=AR+CR
         132601                                 ADD MDEL,WRITE; SETMA; MI<FA;"STORE DR'
         000440                                 BEQ JBACK               "BRANCH IF WE MUST
         000160

                                                                        "STAY IN JLOOP
                 "
                 "
                 "COME HERE IF WE MUST LOOP BACK TO ILOOP . . . WHICH WE DO
                 "       IN THREE STEPS
                 "
000112   040721          FADD DPY(AI),DPX(CI);                          "22. APCI=AI+CI
         132101          MOV TEMP,WRITE;                                "WRITE=JBASE-MDEL
         000550                                 INCMA; MI<FA;           "STORE DI'
         000120                                 BR TOIL1

                 "
                 "
                 "
                 "COME HERE WHEN DONE WITH A FORWARD FFT PASS
                 "
000113   000001  FDONE:                         FADD DPX(AMCI),DPY(JBMDI); "20. DI'=AMCI+JBMDI
         123122                                 INCMA; MI<FA;            "STORE CI'
         000330                                 BR DONCOM
         000120

                 "
                 "
                 "COME HERE WHEN DONE WITH AN INVERSE FFT PASS
                 "
000114   000001  IDONE:                         FSUB DPX(AMCI),DPY(JBMDI); "20. DI'=AMCI-JBMDI
         023000                                 INCMA; MI<FA            "STORE CI'
         000330
         000120

                 "
                 "
                 "COMMON WHEN DONE WITH A PASS
000115   021521  DONCOM:                        FADD;                   "21. PUSH
```

```
        100000                          ADD MDEL,WRITE; SETMA; MI<FA "STORE DR'
        000000
        000160

000116  000000                          INCMA; MI<FA;           "22. STORE DI'
        000340                          RETURN
        000000
        000120

                        $END


****     0 ERRORS   ****


SYMBOL  VALUE
BPDR    177774
DRR     177774
BPDI    177775
DIR     177775
AMCR    177776
AMCI    177777
CR      000000
CI      000001
BBR     000002
BII     000002
BI      000003
BRI     000003
APCR    177774
APCI    177775
JBMDR   177776
DII     177776
CRR     177776
JBMDI   177777
DRI     177777
CIR     177777
AR      000000
AI      000001
DR      000002
BRR     000002
DI      000003
BIR     000003
BASE    000000
N       000001
JBASE   000002
READ    000003
WRITE   000004
ICTR    000005
JCTR    000006
TEMP    000007
N2      000010
W1      000011
W2      000012
W3      000013
```

```
 WD       000014
 MDEL     000015
 ICOUNT   000016
 JCOUNT   000017
 FFT4     000000 ENT
 ILOOP    000016
 ILOOP4   000021
 IBACK    000033
 TOIL3    000034
 ISKP     000035
 IFOR     000036
 INV      000044
 ICOM     000052
 TOIL2    000054
 JLOOP    000055
 JBACK    000072
 TOIL1    000073
 JSKP     000074
 JFOR     000075
 JNV      000103
 JCOM     000111
 FDONE    000113
 IDONE    000114
 DONCOM   000115
1
```

```
1APAL
 REV 1
 PASS 1
 PASS 2
                        "TBLGEN MICROCODE GENERATES
                        "FFT TABLE USING COSINE MICROCODE
                        "SP(5) = TMA
                        "SP(12) = COUNT = 2048
                        "SP(15) = TMA COEFF ADDR
                        "SR(16) = PS COEFF ADDR
                       "SP(17) = COEFF COUNTER
                        "DPX(0) = ARGUMENT
                        "DPX(2) = MULTIPLIER
                        "DPY(2) = PI/4096
                        "DPY(3) = 1.0
            0140        COS $EQU 500

   0000     0000 DELTH: $VAL 0,37,63110,77325
            001F
            6648
            7ED5

   0001     0000 TBL5:   $FP 0.6366197724              "2/PI
            0020
            0517
            CC1B

   0002     0000         $FP 0.79689678946E-1          "C
            001F
            D519
            A28D

   0003     0000         $FP 0.9999999925              "FRACTION MASK
            0020
            07FF
            FFFF

   0004     0000 FPONE:  $FP 1.0                       "ODD BIT MASK
            0020
            1400
            0000

   0005     0000         $FP 1.57079631844             "PI/2
            0020
            1648
            7ED5

   0006     0000         $FP 0.151485129E-3            "E
            001F
            44F6
            BFD6

   0007     0000         $FP -.645963710599            "B
            0020
            0AD5
            10FA
```

FOR TM RAM (OBSOLETE)

```
0008  0000          $FP -.467376661E-2            "D
      001F
      9B36
      CCD5

0009  0000          $FP 2.0                       "NEXT ODD BIT MASK
      0020
      2400
      0000

000A  0394  START:   LDSPI 5; DB=4000             "2K POINTS
      0000
      0400
      0800

000B  0003           LDDA; DB=5                    "SET DEVICE ADDRESS
      8E00
      0400
      0005

000C  03B4           LDSPI 15; DB=!SNCS-1          "SET 1ST TMA
      0000
      0400
      08C5

000D  03B8           LDSPI 16; DB=TBL5             "SET PS POINTER
      0000
      0400
      0001

000E  03BC           LDSPI 17; DB=9.               "SET UP COEFF COUNT
      0000
      0400
      0009

000F  12D4  LP5:     RPSFT; DPX<DB                 "FETCH NEXT COEFF
      0000
      4004
      0000

0010  0277           INC 15; SETTMA; OUT; DB=DPX   "DUMP TO TMRAM
      C000
      0700
      0003

0011  02BC           DEC 17                        "DONE?
      0000
      0000
      0000

0012  0278           BNE LP5; INC 16; SETTMA       "SET NEXT PSA
      01AD
      0000
      0003
```

RAM

```
0013  0003          LDAPS; DB=0            "CLEAR AP STATUS
      8C00
      0400
      0000

0014  03AB          DPX(2)<DB; DB=0;LDTMA; LDSPI 12  "SET TMA = 0
      8600
      4406
      0000

0015  12CC          DPY(2)<DB; RPSF DELTH          "GET ANGLE
      0000
      1006
      FFEB

0016  12CF          DPY(3)<DB; RPSF FPONE; OUT     "GET COS(0)=1.0
      C000
      1007
      FFEE

0017  0295  LOOP:   FADD DPY(3),DPX(2);DEC 5       "GENERATE POINTER
      B400
      01B8
      0000

0018  0001          FADD; BEQ DONE                 "PUSH
      8196
      0000

0019  0000          FMUL DPY(2),FA; DPX(2)<FA      "GENERATE ANGLE
      0000
      8036
      1800

001A  0000          FMUL                           "PUSH
      0000
      0000
      1000

001B  0000          FMUL                           "PUSH
      0000
      0000
      1000

001C  1204          DPX(0)<FM; JSRA COS            "GET COSINE
      0000
      C004
      0140

001D  026B          OUT; DB=DPX(0); INC 12; SETTMA; BR LOOP "BUMP ADR, PUT IN TMA
      C04A
      0700
      0003

001E  0003  DONE:   HALT
```

```
          F000
          0000
          0000

                        $END


 ****     0 ERRORS   ****


 SYMBOL  VALUE
 COS       0140
 DELTH     0000
 TBL5      0001
 FPONE     0004
 START     000A
 LP5       000F
 LOOP      0017
 DONE      001E
1
```