DECUS
PROGRAM LIBRARY

DECUS NO.	8-248
TITLE	SABR-CODED FAST FOURIER TRANSFORM SUBROUTINE
AUTHOR	Gerald N. Cederquist
COMPANY	Cooley Electronics Laboratory University of Michigan Ann Arbor, Michigan
DATE	April 10, 1970
SOURCE LANGUAGE	FORTRAN

A SABR-CODED FAST FOURIER TRANSFORM SUBROUTINE

Gerald N. Cederquist

Cooley Electronics Laboratory
University of Michigan
Ann Arbor, Michigan 48105

ABSTRACT

Using the Fast Fourier Transform algorithm, this subroutine computes in situ either the direct or inverse discrete Fourier transform of a pure-power-of-two number of complex points. Written for use with the DEC 8K FORTRAN system, it runs in less time and takes less core space than the same algorithm coded in FORTRAN. It has been extensively tested and checked; for example, the subroutine will do a direct followed by an inverse transform of 128 points in 47 seconds with a round trip root mean square error of 1.23 parts per million.

INTRODUCTION

The Acoustical Signal Processing Group at Cooley Laboratory is engaged in data acquisition and analysis of underwater sound signals. Due to constantly changing requirements in our data analyses, we write most of our analysis programs in FORTRAN and use the DEC 8K FORTRAN System to run these programs. We have found the Fast Fourier Transform subroutine documented here to be a useful tool in data analysis and wish to pass it along to others who can make use of it.

Note that by using FORTRAN, we are trading CPU time for the capability to make quick and easy modifications to our programs. We do not recommend using this FFT subroutine (or 8K FORTRAN for that matter) in computation-intensive programs; these programs should be written in assembly code or (preferably) run on a larger machine than the PDP-8.

REQUIREMENTS

Hardware

Required hardware to use the FFT subroutine is the same as the required hardware to use the DEC 8K FORTRAN system: a processor which will execute the PDP-8 order code and which has at least 8K of memory. Thus an 8K PDP-8 or an 8K LINC-8 or an 8K PDP-12 may be used. Note that the subroutine does no I/O and thus does not place any requirement on the I/O devices attached to the machine.

Storage

The FFT subroutine requires 5 pages of memory in any memory bank to store the instructions for computing the FFT. In addition the subroutine requires autoindex register 10 and locations 100-120 inclusive on page 0 of the memory bank in which FFT resides.

Note that the subroutine does not itself contain the data area holding the points to be transformed. Thus it is possible to transform up to 512 points (stored in COMMON) in an 8K machine and 1024 points in a 12K or larger machine.

Subroutines

The following subroutines from the 8K FORTRAN Library are required:

SIN, COS, FAD, FSB, FMP, FDV, FLOT, STO, CHS

USAGE

Loading

The FFT subroutine must be assembled using the SABR assembler. (See DEC manual DEC-08-ARXA-D, 8K SABR Assembler, available from the DEC Program Library.) The resulting binary tape may be loaded by the 8K Relocating Loader into any program which needs the FFT subroutine.

<u>Calling Sequence</u>

The FFT subroutine may be invoked via the FORTRAN statement

CALL FFT (A, M, ISIGN)

to compute either the direct transform when ISIGN is < 0:

$$F_k = \frac{1}{2^M} \sum_{j=1}^{2^M} A_j \exp(-2\pi i \cdot jk/2^M), \qquad k = 1, \ldots, 2^M$$

or the inverse transform when ISIGN is > 0:

$$F_k = \sum_{j=1}^{2^M} A_j \exp(2\pi i \cdot jk/2^M), \qquad k = 1, \ldots, 2^M$$

where $i = \sqrt{-1}$. (Though not indicated above, the transform is done in-situ, and needs no scratch storage for F.)

The calling parameters have the following significance:

"A" is an array of complex numbers to be transformed, stored in the format "real part, imaginary part, real part, imaginary part, ..." The numbers forming each complex pair are assumed to be in FORTRAN floating point form.

"M" is the logarithm to the base 2 of the number of points to be transformed, and must be a FORTRAN integer.

"ISIGN" is any positive integer if the sign on the exponent of the complex roots of unity to be used in the transform is to be positive, and is any negative integer if the sign on the exponent of the complex roots of unity is to be negative. Note that if ISIGN is negative, the transform will be taken, and then additionally, all values in the A array will be divided by 2**M (normalizing the direct transform).

For example, the direct transform of a 256 point complex-valued time series stored in the array "X" may be taken by the statement

```
CALL FFT(X, 8, -1)
```

and the inverse transform of a 32 point complex spectrum stored in the array "SPECT" may be taken by the statement

```
CALL FFT(SPECT, 5, 1)
```

EXECUTION TIME

The execution time of the FFT subroutine for both direct and inverse transforms is given in the table below. Note that the fourth column contains an estimate of the time needed to transform according to the definition of the Discrete Fourier Transform. The difference is the time saved by using the FFT technique.

Number of Points	Direct FFT	Inverse FFT	By The Definition
4	< 1	< 1	1
8	1	1	3
16	2	2	8
32	5	4	29
64	11	10	112 (1.8 min)
128	24	23	430 (7.2 min)
256	53	51	1664 (27.7 min)
512	118	113	6571 (109.5 min)

Table 1. Execution Time of FFT in seconds

Note that it is only the FFT algorithm which makes computing discrete Fourier transforms feasible in 8K FORTRAN.

ERROR ANALYSIS

The FFT subroutine was extensively tested for the 128 point case on a time series of samples from a white Gaussian random process. The method used was to transform a time series into the frequency domain and back to time domain using FFT, and then to compute the root mean square error incurred during the round trip.

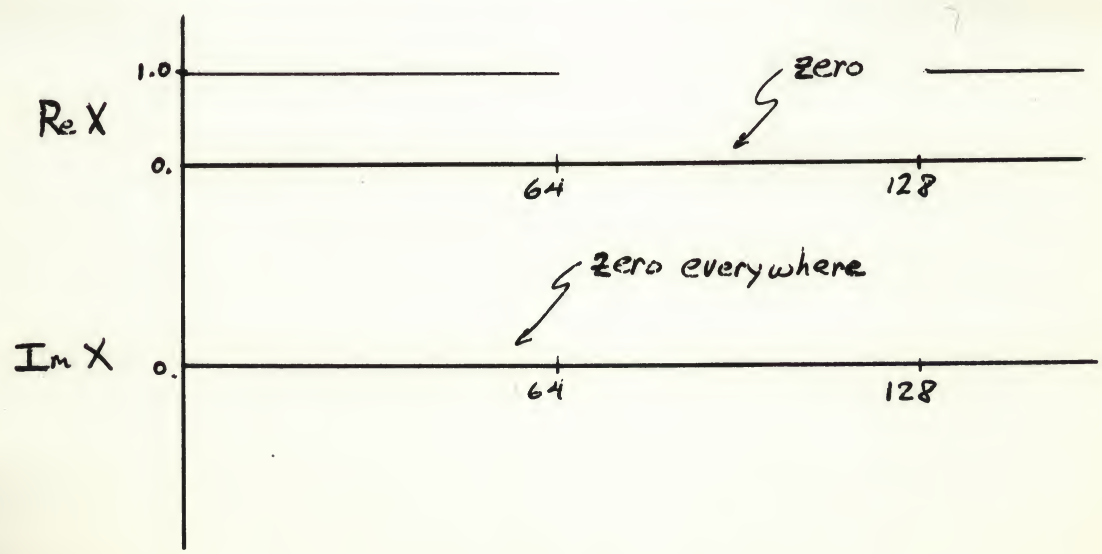
If the original time series is denoted by A_i and the doubly-transformed time series by B_i, then the RMS error E is defined to be

$$E = \left(\frac{\sum_{i=1}^{128} (A_i - B_i)^2}{\sum_{i=1}^{128} A_i^{\,2}} \right)^{\frac{1}{2}}$$

as suggested by the discussion in Ref. 2.

The value of E averaged over 512 trials on an integer valued time series having standard deviation of 249.6 in both real and imaginary components was 1.2380×10^{-6}. Increasing the standard deviation to slightly over one million leads to an E value of 1.2311×10^{6} for 256 trials. In either test, the maximum value of E did not exceed 1.47×10^{-6} in any one trial.

The value of the direct transform via the FFT subroutine was compared with the value obtained by analytical computation for the case of the 128 point waveform shown below:



The largest error found was 1.4×10^{-7}.

As a final test, the average D of the difference between a set of original and doubly-transformed points was computed for $M = 2$ through 9:

$$D = \frac{\sum_{i=1}^{2^M} (A_i - B_i)}{2^M}$$

The table below gives the values of D obtained on the previously-mentioned white Gaussian process with standard deviation 249.6:

M	2^M	D
2	4	0.E0
3	8	1.90735E-6
4	16	-2.86102E-6
5	32	-2.38418E-7
6	64	-4.76836E-7
7	128	1.90734E-6
8	256	-2.38418E-6
9	512	1.E-5 (Upper bound. True value not known due to memory limitations)

The conclusion we draw from all these tests is that the FFT subroutine is capable of giving slightly less than 6 figure accuracy over the range M = 2 through M = 8 and at least 5 figure accuracy for M = 9.

LISTING

The listing which follows this writeup was done on version 16 of the SABR assembler running within CPS, the local Cooley Laboratory operating system for the LINC-8 (see DECUS program L-80). Total assembly time excluding listing is 48 seconds.

REFERENCES

1. J. W. Cooley, P. A. Lewis and P. D. Welch, "The Fast Fourier Transform and Its Applications," IEEE Transactions on Education, Vol. 12, No. 1, pp. 27 ff, March 1969. The algorithm used in the SABR-Coded FFT described here is a direct copy of the one shown in FORTRAN IV on page 29 of this paper.

2. P. D. Welch, "A Fixed-Point Fast Fourier Transform Error Analysis," IEEE Transactions on Audio, Vol. 17, No. 2, p. 151, June 1969.

```
/SABR CODED FAST FOURIER TRANSFORM
/SUBROUTINE FOR A PURE-POWER-OF-2
/NUMBER OF COMPLEX POINTS
/
/THIS SUBROUTINE FOR USE IN 8K FORTRAN PROGRAMS
/GERALD N. CEDERQUIST
/COOLEY ELECTRONICS LABORATORY
/NORTH CAMPUS
/UNIVERSITY OF MICHIGAN
/ANN ARBOR, MICHIGAN  48105
/1 SEPTEMBER, 1969
/
ENTRY FFT
/
/CALLED WITH PARAMETERS A, M, AND ISIGN
/WHERE:
/
/"A" IS AN ARRAY OF COMPLEX NUMBERS TO BE
/TRANSFORMED, STORED IN THE FORMAT
/"REAL PART, IMAGINARY PART,..."
/THE NUMBERS FORMING EACH COMPLEX PAIR
/ARE ASSUMED TO BE IN FORTRAN FLOATING
/POINT FORM.
/
/"M" IS THE LOGARITHM TO THE BASE 2
/OF THE NUMBER OF POINTS TO BE
/TRANSFORMED, AND MUST BE A FORTRAN INTEGER.
/
/"ISIGN" IS ANY POSITIVE INTEGER IF
/THE SIGN ON THE EXPONENT OF THE COMPLEX ROOTS OF
/UNITY TO BE USED IN THE TRANSFORM IS TO
/BE POSITIVE, AND IS ANY NEGATIVE INTEGER
/IF THE SIGN ON THE EXPONENT OF THE COMPLEX ROOTS
/OF UNITY IS TO BE NEGATIVE.  NOTE THAT IF ISIGN
/IS NEGATIVE, THE TRANSFORM WILL BE TAKEN, AND THEN
 /ADDITIONALLY, ALL VALUES IN THE A ARRAY WILL
/BE DIVIDED BY 2**M
/
/REFERENCE:  COOLEY, LEWIS, AND WELCH, "THE
/       FAST FOURIER TRANSFORM AND ITS APPLICATIONS",
 /       IEEE TRANSACTIONS ON EDUCATION, VOL 12, #1,
 /       PAGE 27 FF, MARCH, 1969.
/
/STORAGE REQUIRED:
/       AUTOINDEX REGISTER 10
/       LOCNS 100-120 INCLUSIVE ON PAGE 0 OF THE
/       MEMORY BANK IN WHICH FFT RESIDES
/       5 PAGES ABOVE 200(OCTAL) FOR THE PROGRAM PROPER
/
/SUBROUTINES REQUIRED:
/       SIN, COS, FAD, FSB, FMP, FDV, FLOT, STO, CHS
/
/NOTE THAT THE COMMENTS CONCERNING THE CODE FOR
/THE FFT REFLECT THE FACT THAT IT WAS LIFTED FROM
/THE FORTRAN PROGRAM GIVEN IN THE REFERENCE.
/ALSO, THE FOLLOWING NOTATIONAL CONVENTIONS
```

```
/ARE USED:
/        "->" MEANS "POINTER TO"
/        .REAL. AND .IMAG. ARE THE REAL- AND
/        IMAGINARY-PART OPERATORS RESPECTIVELY, AND
/        HAVE PRECEDENCE > * OR /.
/THE FOLLOWING DECLARATIONS ARE IMPLICIT IN THE
/COMMENTS:
/        COMPLEX A(2**M), U, W, T
```

```
                /STORAGE ALLOCATION
                /
                /ABSOLUTE SYMBOLS
                /
                DUMMY ARGPTR /ARGUMENT POINTER
      0100      ABSYM T         100
      0100      ABSYM TREAL     100
      0101      ABSYM T1        101
      0101      ABSYM AIADR     101
      0102      ABSYM T2        102
      0102      ABSYM ARVADR    102
      0103      ABSYM TIMAG     103
      0103      ABSYM IREV      103
      0104      ABSYM CNT       104
      0105      ABSYM NM2       105
      0106      ABSYM M         106 /COPY OF CALLING PARAMETER
      0107      ABSYM ISIGN     107 / "
      0110      ABSYM N         110
      0111      ABSYM IP        111
      0112      ABSYM II        112
      0113      ABSYM AFIELD    113
      0114      ABSYM AADR      114
      0115      ABSYM L         115
      0116      ABSYM J         116
      0117      ABSYM LE        117
      0120      ABSYM LE1       120
                /RELOCATABLE SYMBOLS
                ARGPTR, /USED ONLY AT START OF PROGRAM
0200  0000      UREAL, BLOCK 3
0201  0000
0202  0000
0203  0000      UIMAG, BLOCK 3 /REAL AND IMAG PARTS OF U
0204  0000
0205  0000
0206  0000      WREAL, BLOCK 3
0207  0000
0210  0000
0211  0000      WIMAG, BLOCK 3 /REAL AND IMAG PARTS OF W
0212  0000
0213  0000
0214  2014      FPONE, 2014
0215  0000       0
0216  0000       0 /FL PT 1.0
0217  2026      PI, 2026
0220  2207       2207
0221  7325       7325 /FL PT 3.1415926535
0222  0200 01   ADRU, UREAL /SINCE NON-NUMERIC LITERALS ARE
0223  0206 01   ADRW, WREAL /NOT ALLOWED
                /
                /PROGRAM STARTS HERE WITH CODE TO
                /PICK UP ARGUMENTS
                /
0224  0000      FFT, BLOCK 2
0225  0000
0226  4067       TAD I FFT /CDF FOR A ARRAY
0227  0224 01
0230  1407
```

```
0231  3113        DCA AFIELD
0232  2225        INC FFT#
0233  4067        TAD I FFT
0234  0224 01
0235  1407
0236  3114        DCA AADR /ADR OF A ARRAY
0237  2225        INC FFT#
0240  4067        TAD I FFT /CDF FOR M
0241  0224 01
0242  1407
0243  3200        DCA ARGPTR
0244  2225        INC FFT#
0245  4067        TAD I FFT /ADR OF M
0246  0224 01
0247  1407
0250  3201        DCA ARGPTR#
0251  2225        INC FFT#
0252  4067        TAD I ARGPTR /VALUE OF M
0253  0200 01
0254  1407
0255  3106        DCA M
0256  4067        TAD I FFT /CDF FOR ISIGN
0257  0224 01
0260  1407
0261  3200        DCA ARGPTR
0262  2225        INC FFT#
0263  4067        TAD I FFT /ADR OF ISIGN
0264  0224 01
0265  1407
0266  3201        DCA ARGPTR#
0267  2225        INC FFT#
0270  4067        TAD I ARGPTR /VALUE OF ISIGN
0271  0200 01
0272  1407
0273  3107        DCA ISIGN
                 /
                 /SET UP DATA FIELD PART OF ARG PSEUDO-OPS
                 /WHICH REFER TO THE "A" ARRAY LATER IN THE PROGRAM.
                 /
0274  1113        TAD AFIELD
0275  6201 05     DCA AI1
0276  3776
0277  1113        TAD AFIELD
0300  3775        DCA AR1
0301  1113        TAD AFIELD
0302  3774        DCA AR2
0303  1113        TAD AFIELD
0304  3773        DCA AI2
0305  1113        TAD AFIELD
0306  3772        DCA AIR
0307  1113        TAD AFIELD
0310  3771        DCA AIPR
0311  1113        TAD AFIELD
0312  3770        DCA AII
0313  1113        TAD AFIELD
0314  3767        DCA AIPI
0315  1113        TAD AFIELD
```

```
0316  3766        DCA AI1R
0317  1113        TAD AFIELD
0320  3765        DCA AI2R
0321  1113        TAD AFIELD
0322  3764        DCA AI1I
0323  1113        TAD AFIELD
0324  3763        DCA AI2I
0325  1113        TAD AFIELD
0326  3762        DCA A1
0327  1113        TAD AFIELD
0330  3761        DCA A2
                 /
                 /       N=2**M
                 /
0331  1106        TAD M
0332  4760        JMS EXPON
0333  3110        DCA N
                 /
                 /       NM2=N - 2
                 /
0334  7344        STA CLL RAL /AC=-2
0335  1110        TAD N
0336  3105        DCA NM2
                 /
                 /       DO 7 I = 1, NM2
                 /
0337  7201        CLA IAC
0340  3112        DCA II
0341  1113        TAD AFIELD /FOOL ASSEMBLER BY OVERLAYING ITS CDF
0342  3757        DCA EXCHNG /WITH CDF FOR THE FIELD IN WHICH "A" RESIDES
                 DO7AS, /START OF DO 7 LOOP SCOPE
                 /
                 /       IREV = BIT REVERSE OF I
                 /
0343  3103        DCA IREV
0344  1112        TAD II
0345  3101        DCA T1
0346  1106        TAD M
0347  7141        CIA CLL
0350  3102        DCA T2
0351  1101       RVLOOP, TAD T1
0352  7110        CLL RAR
0353  3101        DCA T1
0354  1103        TAD IREV
0355  7004        RAL
0356  5377
0357  0423 01
0360  1260 01
0361  1246 01
0362  1236 01
0363  1032 01
0364  1026 01
0365  1016 01
0366  1012 01
0367  0760 01
0370  0750 01
0371  0744 01
```

```
PAGE 06


0372  0734 01
0373  0702 01
0374  0666 01
0375  0646 01
0376  0632 01
0377  7000
0400  3103          DCA IREV
0401  2102          ISZ T2
0402  4045          JMP RVLOOP
0403  7410
0404  5776
                   /
                   /        IF (IREV <= I) GO TO 7
                   /
0405  1103          TAD IREV
0406  7041          CIA
0407  1112          TAD II
0410  7700          SMA CLA
0411  5236          JMP DO7AF
                   /
                   /        T = A(I)
                   /        A(I) = A(IREV)
                   /        A(IREV) = T
                   /
0412  1112          TAD II
0413  6201 05       JMS SUBSC
0414  4775
0415  3101          DCA AIADR
0416  1103          TAD IREV
0417  4775          JMS SUBSC
0420  3102          DCA ARVADR
0421  1374          TAD (-6
0422  3104          DCA CNT
0423  6201 05      EXCHNG, TAD I AIADR /NOTE CDF CHANGED BY ABOVE CODE
0424  1501
0425  3100          DCA T
0426  1502          TAD I ARVADR
0427  3501          DCA I AIADR
0430  1100          TAD T
0431  3502          DCA I ARVADR
0432  2101          INC AIADR
0433  2102          INC ARVADR
0434  2104          ISZ CNT
0435  5223          JMP EXCHNG
                   /
                   /7       CONTINUE
                   /
0436  2112         DO7AF, INC II
0437  1112          TAD II
0440  7041          CIA
0441  1105          TAD NM2
0442  7700          SMA CLA
0443  4045          JMP DO7AS /REENTER LOOP
0444  7410
0445  5773

                   /
                   /NOW THAT BIT REVERSAL IS DONE, DIDDLE AADR
```

```
                    /TO REFLECT THE FACT THAT WE WILL BE USING
                    /FORTRAN-TYPE SUBSCRIPTS FROM NOW ON, WHEREIN
                    /THE ARRAYS START FROM THE FIRST ELEMENT, NOT
                    /THE ZEROTH ELEMENT.
                    /
0446  1114           TAD AADR
0447  1374           TAD (-6
0450  3114           DCA AADR
                    /
                    /        DO 20 L = 1, M
                    /
0451  7201           CLA IAC
0452  3115           DCA L
                    DO20AS, /START OF SCOPE OF OUTERMOST DO 20 LOOP
                    /
                    /        LE = 2**L
                    /
0453  1115           TAD L
0454  6201 05        JMS EXPON
0455  4772
0456  3117           DCA LE
                    /
                    /        LE1 = LE / 2
                    /
0457  1117           TAD LE
0460  7110           CLL RAR
0461  3120           DCA LE1
                    /
                    /        U = (1.0, 0.0)
                    /
0462  7240           STA
0463  1771           TAD ADRU
0464  3010           DCA 10
0465  4770           JMS ONESET
0466  4767           JMS ZERSET
                    /
                    /        W = COMPLX( COS(PI/LE1), SIN(PI/LE1) )
                    /
0467  1120           TAD LE1
0470  1366           TAD (-1
0471  7440           SZA
0472  5277           JMP TWOTST
0473  4765           JMS SETW
0474  4764           JMS MONESET /IF LE1=1, W = (-1., 0.)
0475  4767           JMS ZERSET
0476  5763           JMP WMERGE
0477  1366          TWOTST, TAD (-1
0500  7640           SZA CLA
0501  5315           JMP GOTRIG
0502  6201 05        JMS SETW
0503  4765
0504  4767           JMS ZERSET /IF LE1=2, W = (0., SGN(ISIGN)*1.0 )
0505  1107           TAD ISIGN
0506  7700           SMA CLA
0507  5312           JMP PLUS
0510  4764           JMS MONESET
0511  5763           JMP WMERGE
```

```
0512  6201 05   PLUS, JMS ONESET
0513  4770
0514  5763       JMP WMERGE
0515  1120      GOTRIG, TAD LE1 /CALL UPON TRIG FUNCTION TO EVALUATE W
0516  4033       CALL 0,FLOT
0517  0002 06
0520  4033       CALL 1,STO /LE1 IN FL PT
0521  0103 06
0522  6201 05    ARG T
0523  0100
0524  4033       CALL 1,FAD
0525  0104 06
0526  6201 05    ARG PI
0527  0217 01
0530  4033       CALL 1,FDV
0531  0105 06
0532  6201 05    ARG T
0533  0100
0534  4033       CALL 1,STO
0535  0103 06
0536  6201 05    ARG T /T=PI/LE1
0537  0100
0540  4033       CALL 1,COS
0541  0106 06
0542  6201 05    ARG T
0543  0100
0544  4033       CALL 1,STO
0545  0103 06
0546  6201 05    ARG WREAL /WREAL=COS(T)
0547  0206 01
0550  4033       CALL 1,SIN
0551  0107 06
0552  6201 05    ARG T
0553  0100
0554  1107       TAD ISIGN /TEST FOR SIGN REVERSAL
0555  7700       SMA CLA
0556  5762       JMP WISTO
0557  4033       CALL 0,CHS
0560  0010 06
0561  5377
0562  0600 01
0563  0604 01
0564  1320 01
0565  1327 01
0566  7777
0567  1303 01
0570  1311 01
0571  0222 01
0572  1260 01
0573  0343 01
0574  7772
0575  1271 01
0576  0351 01
0577  7000
0600  4033      WISTO, CALL 1,STO
0601  0103 06
0602  6201 05    ARG WIMAG /WIMAG=SGN(ISIGN)*SIN(T)
```

PAGE 09

```
0603  0211 01
                   WMERGE,
                   /
                   /       DO 20 J = 1, LE1
                   /
0604  7201          CLA IAC
0605  3116          DCA J
                   DO20BS, /START OF SCOPE OF INNER DO 20 LOOP
                   /
                   /       DO 10 I = J, N, LE
                   /
0606  1116          TAD J
0607  3112          DCA II
                   DO10AS, /START OF SCOPE OF DO 10 LOOP
                   /
                   /       IP = I + LE1
                   /
0610  1112          TAD II
0611  1120          TAD LE1
0612  3111          DCA IP
                   /
                   /       T=A(IP) * U
                   /
0613  1111          TAD IP
0614  6201 05       JMS SUBSC
0615  4776
0616  3247          DCA AR1# /SET UP ARG PSEUDO OP
0617  1247          TAD AR1# /ADDRESS PARTS FOR A(IP)
0620  3267          DCA AR2# /REAL PART
0621  1247          TAD AR1# /NOW IMAG PART
0622  1375          TAD (3
0623  3233          DCA AI1#
0624  1233          TAD AI1#
0625  3303          DCA AI2#
0626  4033          CALL 0,CLEAR
0627  0011 06
0630  4033          CALL 1,FAD
0631  0104 06
0632  6211         AI1, ARG 0 /-> .IMAG. A(IP)
0633  0000
0634  4033          CALL 1,FMP
0635  0112 06
0636  6201 05       ARG UIMAG
0637  0203 01
0640  4033          CALL 1,STO
0641  0103 06
0642  6201 05       ARG TREAL /TREAL=.IMAG. A(IP) * UIMAG
0643  0100
0644  4033          CALL 1,FAD
0645  0104 06
0646  6211         AR1, ARG 0 /-> .REAL. A(IP)
0647  0000
0650  4033          CALL 1,FMP
0651  0112 06
0652  6201 05       ARG UREAL
0653  0200 01
0654  4033          CALL 1,FSB
```

```
0655  0113 06
0656  6201 05     ARG TREAL
0657  0100
0660  4033        CALL 1,STO
0661  0103 06
0662  6201 05     ARG TREAL /TREAL=.REAL.A(IP)*UREAL - .IMAG.A(IP)*UIMAG
0663  0100
0664  4033        CALL 1,FAD
0665  0104 06
0666  6211       AR2, ARG 0 /-> .REAL. A(IP)
0667  0000
0670  4033        CALL 1,FMP
0671  0112 06
0672  6201 05     ARG UIMAG
0673  0203 01
0674  4033        CALL 1,STO
0675  0103 06
0676  6201 05     ARG TIMAG
0677  0103
0700  4033        CALL 1,FAD
0701  0104 06
0702  6211       AI2, ARG 0 /-> .IMAG. A(IP)
0703  0000
0704  4033        CALL 1,FMP
0705  0112 06
0706  6201 05     ARG UREAL
0707  0200 01
0710  4033        CALL 1,FAD
0711  0104 06
0712  6201 05     ARG TIMAG
0713  0103
0714  4033        CALL 1,STO
0715  0103 06
0716  6201 05     ARG TIMAG /TIMAG=.REAL.A(IP)*UIMAG + .IMAG.A(IP)*UREAL
0717  0103
                 /
                 /         A(IP) = A(I) - T
                 /
0720  1247        TAD AR1# /-> .REAL. A(IP)
0721  3345        DCA AIPR#
0722  1233        TAD AI1#
0723  3361        DCA AIPI# /-> .IMAG. A(IP)
0724  1112        TAD II
0725  4776        JMS SUBSC
0726  3335        DCA AIR# /-> .REAL. A(I)
0727  1335        TAD AIR#
0730  1375        TAD (3
0731  3351        DCA AII# /-> .IMAG. A(I)
0732  4033        CALL 1,FAD
0733  0104 06
0734  6211       AIR, ARG 0 /-> .REAL. A(I)
0735  0000
0736  4033        CALL 1,FSB
0737  0113 06
0740  6201 05     ARG TREAL
0741  0100
0742  4033        CALL 1,STO
```

PAGE 11

```
0743  0103 06
0744  6211       AIPR, ARG 0 /-> .REAL. A(IP)
0745  0000
0746  4033        CALL 1,FAD
0747  0104 06
0750  6211       AII, ARG 0 /-> .IMAG. A(I)
0751  0000
0752  4033        CALL 1,FSB
0753  0113 06
0754  6201 05     ARG TIMAG
0755  0103
0756  4033        CALL 1,STO
0757  0103 06
0760  6211       AIPI, ARG 0 /-> .IMAG. A(IP)
0761  0000
                 /
                 /       A(I) = A(I) + T
                 /
0762  1335        TAD AIR#
0763  6201 05     DCA AI1R#
0764  3774
0765  1335        TAD AIR#
0766  3773        DCA AI2R#
0767  1351        TAD AII#
0770  3772        DCA AI1I#
0771  5377
0772  1027 01
0773  1017 01
0774  1013 01
0775  0003
0776  1271 01
0777  7000
1000  4045        TAD AII#
1001  7410
1002  1776
1003  3233        DCA AI2I# /ABOVE SETS UP ADDRESSES FOR ARGS
1004  4033        CALL 1,FAD
1005  0104 06
1006  6201 05     ARG TREAL
1007  0100
1010  4033        CALL 1,FAD
1011  0104 06
1012  6211       AI1R, ARG 0 /-> .REAL. A(I)
1013  0000
1014  4033        CALL 1,STO
1015  0103 06
1016  6211       AI2R, ARG 0 /-> .REAL. A(I)
1017  0000
1020  4033        CALL 1,FAD
1021  0104 06
1022  6201 05     ARG TIMAG
1023  0103
1024  4033        CALL 1,FAD
1025  0104 06
1026  6211       AI1I, ARG 0 /-> .IMAG. A(I)
1027  0000
1030  4033        CALL 1,STO
```

```
1031  0103 06
1032  6211        AI2I, ARG 0 /-> .IMAG. A(I)
1033  0000
                  /
                  /10     CONTINUE
                  /
                  DO10AF, /END OF DO 10 LOOP SCOPE
1034  1112         TAD II
1035  1117         TAD LE
1036  3112         DCA II
1037  1112         TAD II
1040  7041         CIA
1041  1110         TAD N
1042  7700         SMA CLA
1043  4045         JMP DO10AS /LOOP AGAIN
1044  7410
1045  5775
                  /
                  /       U = U * W
                  /
1046  4033         CALL 1,FAD
1047  0104 06
1050  6201 05      ARG UIMAG
1051  0203 01
1052  4033         CALL 1,FMP
1053  0112 06
1054  6201 05      ARG WIMAG
1055  0211 01
1056  4033         CALL 1,STO
1057  0103 06
1060  6201 05      ARG TREAL /TREAL=UIMAG*WIMAG
1061  0100
1062  4033         CALL 1,FAD
1063  0104 06
1064  6201 05      ARG UREAL
1065  0200 01
1066  4033         CALL 1,FMP
1067  0112 06
1070  6201 05      ARG WIMAG
1071  0211 01
1072  4033         CALL 1,STO
1073  0103 06
1074  6201 05      ARG TIMAG /TIMAG=UREAL*WIMAG
1075  0103
1076  4033         CALL 1,FAD
1077  0104 06
1100  6201 05      ARG UREAL
1101  0200 01
1102  4033         CALL 1,FMP
1103  0112 06
1104  6201 05      ARG WREAL
1105  0206 01
1106  4033         CALL 1,FSB
1107  0113 06
1110  6201 05      ARG TREAL
1111  0100
1112  4033         CALL 1,STO
```

```
1113  0103 06
1114  6201 05     ARG UREAL /UREAL=UREAL*WREAL - UIMAG*WIMAG
1115  0200 01
1116  4033        CALL 1,FAD
1117  0104 06
1120  6201 05     ARG UIMAG
1121  0203 01
1122  4033        CALL 1,FMP
1123  0112 06
1124  6201 05     ARG WREAL
1125  0206 01
1126  4033        CALL 1,FAD
1127  0104 06
1130  6201 05     ARG TIMAG
1131  0103
1132  4033        CALL 1,STO
1133  0103 06
1134  6201 05     ARG UIMAG /UIMAG=UREAL*WIMAG + UIMAG*WREAL
1135  0203 01
                 /
                 /20      CONTINUE
                 /
                 D020BF, /END OF SCOPE OF INNER DO 20 LOOP
1136  2116        INC J
1137  1116        TAD J
1140  7041        CIA
1141  1120        TAD LE1
1142  7700        SMA CLA
1143  4045        JMP D020BS /LOOP AGAIN
1144  7410
1145  5774
                 D020AF, /END OF SCOPE OF OUTER DO 20 LOOP
1146  2115        INC L
1147  1115        TAD L
1150  7041        CIA
1151  1106        TAD M
1152  7700        SMA CLA
1153  4045        JMP D020AS /LOOP AGAIN
1154  7410
1155  5773

                 /
                 /        IF (ISIGN >= 0 ) RETURN
                 /
1156  1107        TAD ISIGN
1157  7710        SPA CLA
1160  5363        JMP NORMLZ
1161  4040        RETRN FFT
1162  0001 06
                 /
                 /        M = 2*N
                 /
1163  1110       NORMLZ, TAD N
1164  7104        CLL RAL
1165  3106        DCA M
                 /
                 /        T = 1. / N
                 /
```

```
1166  1110        TAD N
1167  4033        CALL 0,FLOT
1170  0002 06
1171  5377
1173  0453 01
1174  0606 01
1175  0610 01
1176  0751 01
1177  7000
1200  4033        CALL 1,STO
1201  0103 06
1202  6201 05     ARG TIMAG
1203  0103
1204  4033        CALL 1,FAD
1205  0104 06
1206  6201 05     ARG FPONE
1207  0214 01
1210  4033        CALL 1,FDV
1211  0105 06
1212  6201 05     ARG TIMAG
1213  0103
1214  4033        CALL 1,STO
1215  0103 06
1216  6201 05     ARG T
1217  0100
                 /
                 /       DO 40 I = 1, M
                 /
1220  7201        CLA IAC
1221  3112        DCA II
                 /NOW WE WILL SEQUENCE THROUGH THE A ARRAY JUST AS
                 /IF IT WERE A REAL ARRAY INSTEAD OF COMPLEX.
                 /THUS WE NEED TO DIDDLE THE ADDR OF A AGAIN
1222  1114        TAD AADR
1223  1377        TAD (3
1224  3114        DCA AADR
                 DO40AS, /START OF SCOPE OF DO 40 LOOP
                 /
                 /       AA(I) = AA(I) * T
                 /
                 /WHERE THE FOLLOWING IS ASSUMED
                 /       EQUIVALENCE (A(1), AA(1))
                 /       REAL AA
                 /
1225  1112        TAD II
1226  7104        CLL RAL /COMPUTE SUBSCRIPT
1227  1112        TAD II
1230  1114        TAD AADR
1231  3237        DCA A1#
1232  1237        TAD A1#
1233  3247        DCA A2#
1234  4033        CALL 1,FAD
1235  0104 06
1236  6211       A1, ARG 0 /-> AA(I)
1237  0000
1240  4033        CALL 1,FMP
1241  0112 06
```

PAGE 15

```
1242  6201 05    ARG T
1243  0100
1244  4033       CALL 1,STO
1245  0103 06
1246  6211      A2, ARG 0 /-> AA(I)
1247  0000
                /
                /40      CONTINUE
                /
                DO40AF, /END OF SCOPE OF DO 40 LOOP
1250  2112       INC II
1251  1112       TAD II
1252  7041       CIA
1253  1106       TAD M
1254  7700       SMA CLA
1255  5225       JMP DO40AS /LOOP AGAIN
1256  4040       RETRN FFT
1257  0001 06
                /
                /END OF COMPUTATIONAL PART OF SUBROUTINE
                /UTILITY PROGRAMS FOLLOW
```

```
                  /UTILITY SUBROUTINES FOR FFT SUBROUTINE
                  /
                  /RAISE 2 TO THE INTEGER POWER IN THE AC
                  /RETURN INTEGER ANSWER IN AC
                  /
1260  0000        EXPON, 0
1261  7041         CIA
1262  3100         DCA T
1263  7001         IAC
1264  7104        EXL, CLL RAL
1265  2100         ISZ T
1266  5264         JMP EXL
1267  6201 05      JMP I EXPON
1270  5660
                  /
                  /COMPUTE THE ADDRESS IN THE A ARRAY
                  /CORRESPONDING TO THE INTEGER SUBSCRIPT
                  /IN THE AC.  RETURN ADDRESS IN AC
                  /
1271  0000        SUBSC, 0
1272  3302         DCA SUTEMP
1273  1302         TAD SUTEMP
1274  7104         CLL RAL
1275  1302         TAD SUTEMP
1276  7104         CLL RAL
1277  1114         TAD AADR
1300  6201 05      JMP I SUBSC
1301  5671
1302  0000        SUTEMP, 0
                  /
                  /MOVE FL PT ZERO INTO WRDS -> TO BY AX10
                  /
1303  0000        ZERSET, 0
1304  6201 05      DCA I 10
1305  3410
1306  3410         DCA I 10
1307  3410         DCA I 10
1310  5703         JMP I ZERSET
                  /
                  /MOVE FL PT 1.0 INTO WRDS -> TO BY AX10
                  /
1311  0000        ONESET, 0
1312  6201 05      TAD FPONE
1313  1776
1314  3410         DCA I 10
1315  3410         DCA I 10
1316  3410         DCA I 10
1317  5711         JMP I ONESET
                  /
                  /MOVE FL PT -1.0 INTO WRDS -> TO BY AX10
                  /
1320  0000        MONESET, 0
1321  1375         TAD (6014
1322  6201 05      DCA I 10
1323  3410
1324  3410         DCA I 10
1325  3410         DCA I 10
```

PAGE 17

```
1326  5720        JMP I MONESET
                  /
                  /SET AX REG 10 TO POINT TO W
                  /
1327  0000        SETW, 0
1330  7240         STA
1331  6201 05      TAD ADRW
1332  1774
1333  3010         DCA 10
1334  5727         JMP I SETW
1374  0223 01
1375  6014
1376  0214 01
1377  0003
                   END
```