

A Radix 2 FFT Program for the HPC

National Semiconductor
Application Note 487
Ashok Krishnamurthy
April 1987



INTRODUCTION

This report describes the implementation of a radix-2, Decimation-in-time FFT algorithm on the HPC. The program, as presently set up can do FFTs of length 2, 4, 8, 16, 32, 64, 128 and 256. The program can be easily modified to work with higher FFT lengths by increasing the Twiddle Factor table.

FFT FUNDAMENTALS

If $x(n)$, $n = 0, 1, \dots, N-1$ are N samples of a time domain signal, its Discrete Fourier Transform (DFT) is defined as

$$X(k) = \sum_{n=0}^{N-1} x(n) W^{nk}, \quad k = 0, 1, \dots, N-1$$

where $W = e^{-j2\pi/N}$

The straight evaluation of the above equation requires on the order of N^2 complex multiplies. The FFT is nothing but a fast algorithm to compute the DFT that uses only on the order of $N \log(N)$ complex multiplies. Many different FFT algorithms exist (please see references 1, 2 and 3). The algorithm implemented for the HPC is the most common type of FFT — a radix-2, Decimation-in-time algorithm. This class of algorithms requires that the number of input samples, N , be a power of 2. This is usually not a problem, since the input data can be zero padded to achieve this. The development of this algorithm is described in references 1 and 2; the discussion here is brief and based on reference 1.

Separating the DFT summation above into the even-numbered points and odd-numbered points of $x(n)$, we can rewrite the above sum as:

$$X(k) = \sum_{n \text{ even}} x(n) W^{nk} + \sum_{n \text{ odd}} x(n) W^{nk}$$

Using $n = 2r$ for n even and $n = 2r + 1$ for n odd, we can further rewrite the above as:

$$X(k) = \sum_{r=0}^{N/2-1} x(2r) W^{2rk} + W^k \sum_{r=0}^{N/2-1} x(2r+1) W^{2rk}$$

If $G(k)$ is the $N/2$ point DFT of $x(2r)$ and $H(k)$ is the $N/2$ point DFT of $x(2r+1)$, the above equation can be written as:

$$X(k) = G(k) + W^k H(k)$$

This equation shows that a N point DFT can be written as the sum of two $N/2$ point DFTs. The $N/2$ point DFTs can be computed as the sum two $N/4$ point DFTs and so on until we are left with two point DFTs. The two point DFTs can be trivially evaluated by direct computation.

Figure 1, taken from reference 1, shows the decomposition for the case $N = 8$. With reference to this figure, we can note the following points.

1. If N is the number of points in the original sequence, where $N = 2^L$, then there are L stages in the DFT decomposition.

2. The basic computation unit is the so-called Butterfly, shown in Figure 2. Each stage involves the computation of $N/2$ butterflies.
3. The results from the computation in one stage are fed to the next stage after multiplication by some power of W . These powers of W are the so-called Twiddle Factors. Note that each power of W is really a complex number that can be represented by its real and imaginary parts. The real part of W^k is $\cos(2\pi k/N)$ and the imaginary part is $-\sin(2\pi k/N)$.
4. The number of distinct Twiddle Factors used in the first stage is 1, in the second stage is 2 etc., until the L^{th} stage that involves $2^L - 1 = N/2$ twiddle factors. Each twiddle factor in the first stage is involved in $N/2$ Butterflies, in the second stage with $N/4$ butterflies etc., until in the L^{th} stage each twiddle factor is involved with $N/(2^L) = 1$ butterfly.
5. The input data sequence needs to be suitably scrambled if the output sequence is to be in the proper order. This scrambling is easily accomplished by using the so-called Bit-Reverse counter as outlined in reference 2.
6. The outputs from each stage can be stored back again in the same storage area as the input sequence. This gives the algorithm the in-place property. Thus the final DFT results overwrite the initial data.

THE INVERSE FFT

If $X(k)$ $k = 0, 1, \dots, N-1$ is the DFT of a sequence, then its inverse DFT, $x(n)$, is defined as follows:

$$x(n) = \left(\frac{1}{N}\right) \sum_{k=0}^{N-1} X(k) W^{-nk} \quad n = 0, 1, \dots, N-1.$$

Thus the Inverse FFT is the same as the forward FFT except for the following: 1. Negative powers of W are used instead of positive powers; and 2. The final sequence is scaled by $1/N$. The basic FFT program can therefore be used to compute the inverse FFT with these two changes. This is the approach used in the HPC implementation.

TWIDDLE FACTOR TABLE

The brief description of the FFT in the previous section shows that the algorithm needs to use the Twiddle Factors W^k in the computation. The twiddle factors can either be computed as required, they can be computed using a recursive relation, or they can be obtained by looking up in a table (Ref. 2). The approach used in the HPC implementation is to construct a table containing the needed twiddle factors. This table is stored in ROM and values needed are looked up from this table. The length of the table needed is determined by the maximum FFT length that you want to use. The HPC FFT implementation is presently limited to a maximum length of 256. This requires that the twiddle factors $W^0, W^1, \dots, W^{255}$ be available, where

$W = e^{-j2\pi/256}$. Since $e^{jx} = \cos(x) + j\sin(x)$, the values stored in this table are $\cos(0)$, $\sin(0)$, $\cos(2\pi/256)$, $\sin(2\pi/256)$ etc., up to $\cos(2\pi \times 255/256)$, $\sin(2\pi \times 255/256)$. The table used in the implementation is organized as follows:

```
.WORD cos(0) × 214
.WORD sin(0) × 214
.WORD cos(2π/256) × 214
.WORD sin(2π/256) × 214
.
.
.
.WORD cos(2π255/256) × 214
.WORD sin(2π255/256) × 214
```

This table is available in the file TWDTBL.MAC and occupies 1024 bytes of storage.

DATA STORAGE

The data to be transformed, $x(0), \dots, x(N-1)$ are also regarded as complex numbers with a real and an imaginary part. Let $xr(i)$ be the real part of $x(i)$ and $xi(i)$ the imaginary part of $x(i)$. Then the data needs to be stored as follows:

```
.WORD xr(0)
.WORD xi(0)
.WORD xr(1)
.WORD xi(1)
.
.
.
.WORD xr(N-1)
.WORD xi(N-1)
```

The length of this storage area obviously depends on the number of data points to be transformed. Note that the FFT program itself does not use any base page user RAM. Also, only 8 words of stack are needed. Thus the base page user RAM can be used to store the data to be transformed. Since 192 bytes are available in this area, transforms of up to 32 point in length can be in the single chip mode with no external RAM.

USING THE FFT PROGRAM

The FFT program along with test data to test the program is provided in the files FFT.MAC, TSTDAT.MAC and TWDTBL.MAC. TSTDAT.MAC contains the test data, and the output from the FFT routines. TWDTBL.MAC contains the Twiddle Factors. The FFT computation involves the use

of 4 different subroutines: FFT, IFFT, BRNCNTR and SMULT. FFT does the forward FFT calculation, IFFT the Inverse FFT calculation, BRNCNTR implements the bit reversed counter, and SMULT does signed multiplication.

Two global symbols need to be defined by the user to use the FFT routines. The first, called TWSTAD should be set to the address of the start of the twiddle factor table. The second, called DTSTAD, should be set to the address of the start of the data area to be transformed. For details on the organization of these storage areas, see the preceding sections.

The actual number of data points to be transformed needs to be passed to the FFT routines. This is done as follows.

Two symbols that refer to words of on-chip RAM have been defined. The first is NUMB = W(01C0) and the second is L1 = W(01C2). Before calling the FFT routine, the user should load NUMB with N, the number of data points to be transformed, and L1 with L, $N = 2^L$.

To do a forward FFT, call FFT; to do an inverse FFT, call IFFT. In both cases, the output of the transform overwrites the input data.

INCREASING THE MAXIMUM TRANSFORM LENGTH

The maximum transform length for the FFT program is primarily limited by the size of the Twiddle Factor table. To increase the transform length, the following needs to be done.

1. Increase the Twiddle Factor table. Thus, if the maximum transform length required is 1024, the table needs to store the cosine and sine of the angles

$$0, 2\pi/1024, 2\pi \times 2/1024, \dots, 2\pi \times 1023/1024$$

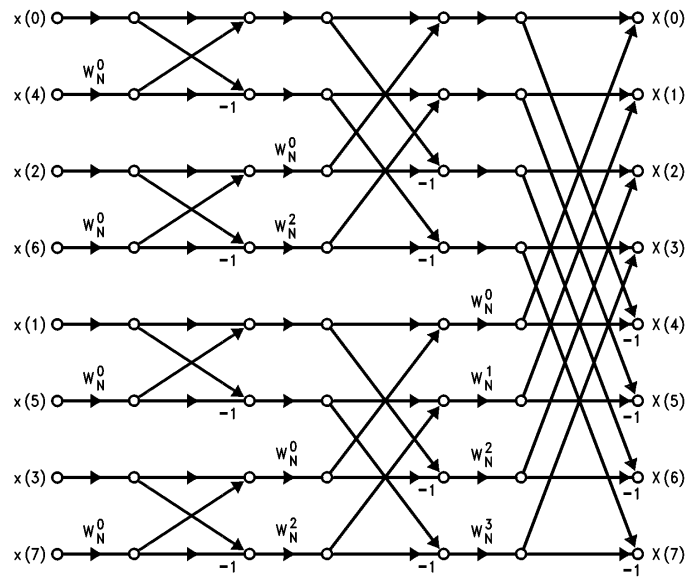
2. Change the global symbol LMAX such that the maximum transform length is 2^{LMAX} .

FFT/IFFT TEST PROGRAM

The data in the file TSTDAT.MAC can be used to test the FFT program. The data and its transform value is from reference 3. The program in reference 3 is for a Floating point FORTRAN FFT program. Since the HPC FFT program is a fixed point one, the input data needs to be suitably scaled. The scale factor chosen is 2^{10} . The file TSTDAT.MAC contains the scaled input data, and the expected transform. The input data is stored in memory words 200/27E and the expected transform is stored in memory words 280/2FE. To run the test program, do the following.

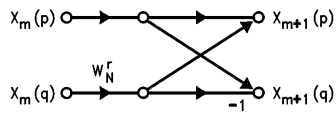
Set up the MOLE Development System with Blocks 0, 13, 14 and 15 mapped ON. Download the program to the MOLE. Set up a Breakpoint at F410. Run the program starting at F400. When the program is breakpointed, list memory words 200/27F and compare them with memory words 280/2FE.

Note that any difference between the expected DFT values and the DFT values actually computed is due to the fixed point computations in the FFT program.



TL/DD/9259-1

FIGURE 1. FFT Flow Graph for N = 8 Points



TL/DD/9259-2

FIGURE 2. The Butterfly—The Basic Computation Unit in the FFT

REFERENCES

1. A.V. Oppenheim and R.W. Schaffer, *Digital Signal Processing*, Prentice-Hall, New Jersey, 1975.
2. L.R. Rabiner and B. Gold, *Theory and Applications of Digital Signal Processing*, Prentice-Hall, New Jersey, 1975.
3. IEEE ASSP Society Digital Signal Processing Committee, *Programs for Digital Signal Processing*, IEEE Press, New York, 1979.

The code listed in this App Note is available on Dial-A-Helper.

Dial-A-Helper is a service provided by the Microcontroller Applications Group. The Dial-A-Helper system provides access to an automated information storage and retrieval system that may be accessed over standard dial-up telephone lines 24 hours a day. The system capabilities include a MESSAGE SECTION (electronic mail) for communicating to and from the Microcontroller Applications Group and a FILE SECTION mode that can be used to search out and retrieve application data about NSC Microcontrollers. The minimum system requirement is a dumb terminal, 300 or 1200 baud modem, and a telephone.

With a communications package and a PC, the code detailed in this App Note can be down loaded from the FILE SECTION to disk for later use. The Dial-A-Helper telephone lines are:

Modem (408) 739-1162
Voice (408) 721-5582

For Additional Information, Please Contact Factory

APPENDIX A

Listing of FFT Program Code

NATIONAL SEMICONDUCTOR CORPORATION
HPC CROSS ASSEMBLER, REV: C, 30 JUL 86

PAGE: 1

```

1          ;
2          ; THIS PROGRAM IMPLEMENTS A RADIX-2, DECIMATION IN TIME FFT ALGORITHM.
3          ;
4          ;
5 0008          LMAX = 08          ; MAXIMUM FFT LENGTH IS 2^LMAX.
6          ;
7 F000          TWSTAD = 0F000      ; TWIDDLE FACTOR TABLE START
8          ; ADDRESS.
9 0200          DTSTAD = 0200      ; DATA STORAGE AREA START
10         ; ADDRESS.
11         ;
12 01C0          NUMB = W(01C0)      ; NUMBER OF DATA POINTS TO BE
13         ; TRANSFORMED.
14 01C2          L1 = W(01C2)        ; NUMB IS 2^L1.
15 01C4          LSHIFT = W(01C4)    ; LSHIFT = LMAX - L1. IT IS A SHIFT
16         ; FACTOR NEEDED TO COMPUTE THE
17         ; ADDRESS REQUIRED FOR TWIDDLE
18         ; FACTOR LOOKUP.
19 01C6          NBFLY = W(01C6)     ; NUMBER OF BUTTERFLIES PER
20         ; TWIDDLE FACTOR PER STAGE.
21 01C8          ISTEP = W(01C8)     ; IF X(1) AND X(J) ARE INVOLVED
22         ; IN A BUTTERFLY, THEN J=I+ISTEP.
23         ; IT IS ALSO THE NUMBER OF TWIDDLE
24         ; FACTORS IN A STAGE.
25 01CA          ILEAP = W(01CA)     ; IF X(I) IS THE FIRST DATA VALUE
26         ; FOR THE FIRST BUTTERFLY FOR A
27         ; GIVEN TWIDDLE FACTOR, THEN THE
28         ; SUBSEQUENT BUTTERFLIES FOR THAT
29         ; TWIDDLE FACTOR HAVE AS THE FIRST
30         ; DATA VALUE X(I+N*ILEAP).
31 01CC          WESTEP = W(01CC)     ; TWIDDLE FACTOR EXPONENT STEP.
32         ; THE TWIDDLE FACTORS FOR A GIVEN
33         ; STAGE ARE W^I(WESTEP).
34 01CE          NSTG = W(01CE)      ; FFT STAGE BEING EVALUATED.
35         ;
36 01D0          ISTART = W(01D0)    ; INDEX OF THE FIRST DATA VALUE
37         ; FOR THE FIRST BUTTERFLY FOR A
38         ; GIVEN TWIDDLE FACTOR.
39 01D2          WEXP = W(01D2)      ; EXPONENT VALUE FOR A GIVEN
40         ; TWIDDLE FACTOR.
41 01D4          NTWD = W(01D4)      ; TWIDDLE FACTOR BEING EVALUATED.
42         ;
43 01D6          COSTH = W(01D6)     ; COSINE PART OF TWIDDLE FACTOR.
44         ;
45 01D8          SINTH = W(01D8)     ; SINE PART OF TWIDDLE FACTOR.
46         ;
47 01DA          M1 = W(01DA)        ; INDEX OF FIRST DATA VALUE FOR
48         ; A BUTTERFLY.
49 01DC          NBCNT = W(01DC)     ; BUTTERFLY BEING EVALUATED.
50         ;
51 01DE          RIADDR = W(01DE)    ; ADDRESS OF REAL PART OF FIRST

```

```

52                                     ; DATA VALUE INVOLVED IN A BUTTERFLY.
53      01E0      R2ADDR = W(01E0)      ; ADDRESS OF REAL PART OF SECOND
54                                     ; DATA VALUE INVOLVED IN A BUTTERFLY.
55      01E2      XR1 = W(01E2)          ; REAL PART OF FIRST DATA VALUE
56                                     ; INVOLVED IN A BUTTERFLY.
57      01E4      XI1= W(01E4)           ; IMAGINARY PART OF ABOVE.
58                                     ;
59      01E6      XR2 = W(01E6)          ; REAL PART OF SECOND DATA VALUE
60                                     ; INVOLVED IN A BUTTERFLY.
61      01E8      XI2 = W(01E8)         ; IMAGINARY PART OF ABOVE.
62                                     ;
63      01EA      TEMPR = W(01EA)        ; TEMPORARY STORAGE USED IN
64                                     ; A BUTTERFLY.
65      01EC      TEMPI = W(01EC)        ; SAME AS ABOVE.
66                                     ;
67      01EE      MTEMP = W(01EE)       ; TEMPORARY STORAGE USED IN SMULT.
68      ;
69      .INCLD TSTDAT.MAC
70      .INCLD TWDTBL.MAC
71      ;
72      ;
73      F400      . = 0F400
74      TSTFFT:
75      F400 B701FOC4      LD SP, 01F0
76      F404 832001COAB    LD NUMB, 020      ; 32 POINT FFT.
77      F409 830501C2AB    LD L1, 05        ; 32 = 2^5.
78      F40E 3049          JSR FFT           ; COMPUTE FFT.
79      F410 40            NOP
80      F411 31C4          JSR IFFT
81      F413 40            NOP
82      F414 61            JP .-1
83      ;
84      ;
85      ; THIS SUBROUTINE IMPLEMENTS A BIT REVERSED COUNTER AS NEEDED FOR
86      ; DATA SHUFFLING IN THE FFT ROUTINE. THE ALGORITHM IS BASED ON
87      ; THE DESCRIPTION IN:
88      ;   RABINER AND GOLD,
89      ;   THEORY AND APPLICATIONS OF DIGITAL SIGNAL PROCESSING,
90      ;   PRENTICE-HALL, 1975.
91      ;
92      ; ON INPUT, X CONTAINS THE PREVIOUS BIT REVERSED COUNTER VALUE.
93      ; THE NEXT BIT REVERSED OUTPUT IS RETURNED IN X.
94      ; A IS LOST, B AND K ARE PRESERVED.
95      ;
96      .LOCAL
97      BRCNTR:
98      F415 B601COA8      LD A, NUMB      ; GET NUMBER OF DATA SAMPLES
99                                     ; TO BE TRANSFORMED.
100     F419 C7             SHR A           ; DIVIDE BY 2.
101     $REPEAT:
102     F41A 96CEFD         IFGT A, X      ; IS BIT BEING TESTED A 0 ?

```

```

103 F41D 47          JP $FOUND          ; YES, SO STOP CHECKING.
104                                     ; GET HERE MEANS BIT BEING
105                                     ; CHECKED IS 1.
106 F41E 02          SET C
107 F41F A0C8CEE8     SUBC X, A          ; ZERO OUT THE BIT.
108 F423 C7          SHR A              ; UPDATE BIT LOCATOR.
109 F424 6A          JP $REPEAT
110                                     $FOUND:
111 F425 A0C8CEF8     ADD X, A
112 F429 3C          RET
113                  .LOCAL
114                  ;
115                  ;
116                  ;
117                  ; THIS SUBROUTINE MULTIPLIES TWO 16-BIT 2'S COMPLEMENT INTEGERS AND RETURNS
118                  ; THE UPPER HALF OF THE RESULT. THE MULTIPLICAND IS IN A, AND THE MULTIPLIER
119                  ; IN W(B). THE RESULT IS RETURNED IN A. ONE TEMPORARY WORD OF STORAGE,
120                  ; ADDRESSED AS MTEMP IS USED.
121                  ;
122                  SMULT:
123 F42A 830001EEAB     LD MTEMP, 0      ; CLEAR TEMPORARY STORAGE.
124 F42F A9CC          INC B              ; B NOW POINTS TO UPPER BYTE
125                                     ; OF MULTIPLIER.
126 F431 17          IF M(B).7          ; IS IT NEGATIVE ?
127 F432 B601EEAB     ST A, MTEMP        ; THEN SAVE MULTIPLICAND IN MTEMP.
128 F436 AACC          DECSZ B           ; B INTO WORD POINTER.
129 F438 40          NOP
130 F439 B601EEAE     X A, MTEMP         ; SWAP A AND MTEMP.
131 F430 B601EF17     IF M(($MTEMP)+1).7 ; IS MULTIPLICAND NEGATIVE ?
132 F441 F8          ADD A, W(B)         ; THEN ACCUMULATE MULTIPLIER.
133 F442 B601EEAE     X A, MTEMP
134 F446 FE          MULT A, W(B)        ; UNSIGNED MULTIPLY.
135 F447 AECE          X A, X            ; UPPER HALF IN A.
136 F449 02          SET C
137 F44A B601EEEB     SUBC A, MTEMP
138 F44E E7          SHL A
139 F44F 96CF17       IF H(X).7
140 F452 04          INC A
141 F453 E7          SHL A
142 F454 96CF16       IF H(X).6
143 F457 04          INC A
144 F458 3C          RET
145                  ;
146                  ;
147                  ;
148                  ; THIS SUBROUTINE IMPLEMENTS THE FIXED POINT RADIX-2 DECIMATION-IN-TIME
149                  ; FFT ALGORITHM. THE DATA IS INITIALLY PUT IN THE BIT REVERSED ORDER, AND
150                  ; THEN THE FFT IS COMPUTED. FOR THE THEORY BEHIND THE ALGORITHM, CONSULT:
151                  ;
152                  ; 1. OPPENHEIM AND SCHAFER, DIGITAL SIGNAL PROCESSING,
153                  ; PRENTICE-HALL.

```

```

154      ;
155      ;      2. RABINER AND GOLD, THEORY AND APPLICATIONS OF DIGITAL SIGNAL
156      ;      PROCESSING, PRENTICE-HALL, 1975.
157      ;
158      ; THE ALGORITHM USED CLOSELY FOLLOWS THE FORTRAN PROGRAM FOREA IN
159      ;
160      ;      3. PROGRAMS FOR DIGITAL SIGNAL PROCESSING, IEEE.
161      ;
162      ;
163      FFT:
164      ;
165      ; FIRST PUT THE DATA IN BIT REVERSED ORDER.
166      ;
167 F459 00      CLR A
168 F45A ABCC      ST A, B      ; SET UP NORMAL COUNTER.
169 F45C ABCE      ST A, X      ; SET UP BIT REVERSED COUNTER.
170 F45E A401COCAAB LD K, NUMB      ; K HAS NUMBER OF DATA POINTS.
171      REVLP:
172 F463 A0CCCEFD      IFGT X, B      ; IS BIT REV CNTR → NORM CNTR ?
173 F467 42      JP SWAP      ; YES, SO SWAP DATA.
174 F468 9421      JMP COUNT      ; NO SO INCREMENT COUNT.
175      SWAP:
176 F46A AFCC      PUSH B
177 F46C AFCE      PUSH X
178 F46E A8CC      LD A, B      ; INDEX VALUE I IS IN A.
179 F470 E7      SHL A
180 F471 E7      SHL A
181 F472 B80200      ADD A, DTSTAD      ; GET ADDR. OF XR(I).
182 F475 ABCC      ST A, B      ; SAVE IT IN B.
183 F477 A8CE      LD A, X      ; INDEX VALUE J IS IN A.
184 F479 E7      SHL A
185 F47A E7      SHL A
186 F47B B80200      ADD A, DISTAD      ; GET ADDR. OF XR(J).
187 F47E ABCE      ST A, X      ; SAVE IT IN X.
188 F480 E4      LD A, W(B)      ; A ← XR(I).
189 F481 F1      X A, W(X+)      ; A ← XR(J), XR(J) ← XR(I).
190 F482 E1      XS A, W(B+)      ; A ← XR(I), XR(I) ← XR(J).
191 F483 40      NOP
192 F484 E4      LD A, W(B)      ; A ← XI(I).
193 F485 F5      X A, W(X)      ; A ← XI(J), XI(J) ← XI(I).
194 F486 E6      ST A, W(B)      ; XI(I) ← XI(J).
195 F487 3FCE      POP X
196 F489 3FCC      POP B
197      ;
198      COUNT:
199      ;
200 F48B AACA      DECSZ K      ; DONE ?
201 F48D 41      JP UPIT      ; NO GO DO SOME MORE.
202 F48E 46      JP DOFFT
203      UPIT:
204 F48F 347A      JSR BRCNTR      ; COUNT UP ON BIT REV CNTR.

```

```

205 F491 A9CC          INC B          ; COUNT UP ON NORMAL CNTR.
206 F493 9530          JMP REVLP
207                   DOFFT:
208                   ;
209                   ; DATA IS NOW STORED IN THE BIT REVERSED ORDER. COMPUTE THE FFT.
210                   ;
211 F495 9008          LD A, LMAX          ; A HAS MAX FFT EXPONENT.
212 F497 04          INC A
213 F498 04          INC A
214 F499 02          SET C
215 F49A B601C2EB      SUBC A, L1          ; COMPUTE LSHIFT.
216 F49E B601C4AB      ST A, LSHIFT
217 F4A2 B601C0A8      LD A, NUMB
218 F4A6 C7          SHR A
219 F4A7 B601C6AB      ST A, NBFLY          ; INITIALIZE NBFLY.
220 F4AB B601CCAB      ST A, WESTEP          ; INITIALIZE WESTEP.
221 F4AF 830101C8AB     LD ISTEP, 01          ; INITIALIZE ISTEP.
222 F4B4 830201CAAB     LD ILEAP, 02          ; INITIALIZE ILEAP.
223                   ;
224                   ; SET UP L1 STAGES OF BUTTERFLIES.
225                   ;
226 F4B9 B601C2AB      LD A, L1
227 F4BD B601CEAB      ST A, NSTG          ; LOOP L1 TIMES.
228                   LOOP1:
229                   ;
230 F4C1 00          CLR A
231 F4C2 B601D0AB      ST A, ISTART          ; INITIALIZE ISTART FOR EACH STAGE.
232 F4C6 B601D2AB      ST A, WEXP          ; INITIALIZE WEXP.
233                   ;
234                   ; SET UP ISTEP LOOPS OF TWIDDLE FACTORS.
235                   ;
236 F4CA B601C8A8      LD A, ISTEP
237 F4CE B601D4AB      ST A, NTWD          ; LOOP ISTEP TIMES.
238                   LOOP2:
239                   ;
240                   ; LOOK UP THE TWIDDLE FACTOR.
241                   ;
242 F4D2 A401C4CAAB     LD K, LSHIFT          ; SHIFT LEFT LSHIFT TIMES.
243 F4D7 B601D2A8      LD A, WEXP
244                   GADLP:
245 F4DB E7          SHL A
246 F4DC AACA          DECSZ K          ; DONE SHIFTING ?
247 F4DE 63          JP GADLP          ; NO SO DO MORE.
248 F4DF B8F000        ADD A, TWSTD          ; ADD STARTING ADDR OF TWIDDLE
249                   ; FACTOR TABLE.
250 F4E2 ABCE          ST A, X          ; TWIDDLE FACTOR ADDR IN X.
251 F4E4 F0          LD A, W(X+)          ; GET COS(THETA).
252 F4E5 B601D6AB      ST A, COSTH
253 F4E9 F4          LD A, W(X)          ; GET SIN(THETA).
254 F4EA 01          COMP A
255 F4EB 04          INC A          ; MAKE IT NEGATIVE.

```

```

256 F4EC B601D8AB      ST A, SINTH
257                    ;
258 F4F0 A501D001DAAB  LD M1, ISTART      ; INITIALIZE M1.
259                    ;
260                    ; SET UP NBFLY BUTTERFLIES FOR THIS TWIDDLE FACTOR.
261                    ;
262 F4F6 A501C601DCAB  LD NBCNT, NBFLY      ; LOOP NBFLY TIMES.
263                    LOOP3:
264 F4FC B601DAAB      LD A, M1              ; GET INDEX OF X(I).
265 F500 E7            SHL A
266 F501 E7            SHL A
267 F502 B80200        ADD A, DTSTAD          ; ADDR. OF XR(I).
268 F505 B601DEAB      ST A, R1ADDR
269 F509 ABCE          ST A, X
270 F50B F0            LD A, W(X+)            ; A ← XR(I).
271 F50C B601E2AB      ST A, XR1              ; STORE IN XR1.
272 F510 F4            LD A, W(X)            ; A ← XI(I).
273 F511 B601E4AB      ST A, XI1              ; STORE IN XI1.
274 F515 B601DAA8      LD A, M1
275 F519 B601C8F8      ADD A, ISTEP          ; GET INDEX OF X(J).
276 F51D E7            SHL A
277 F51E E7            SHL A
278 F51F B80200        ADD A, DISTAD          ; ADDR. OF XR(J).
279 F522 B601E0AB      ST A, R2ADDR
280 F526 ABCE          ST A, X
281 F528 F0            LD A, W(X+)            ; A ← XR(J).
282 F529 B601E6AB      ST A, XR2              ; STORE IN XR2.
283 F520 F4            LD A, W(X)            ; A ← XI(J).
284 F52E B601E8AB      ST A, XI2              ; STORE IN XI2.
285                    ;
286 F532 B201E6        LD B, #XR2            ; B ← ADDR(XR2).
287 F535 B601D6A8      LD A, COSTH           ; A ← COS(THETA).
288 F539 350F          JSR SMULT              ; COMPUTE XR(J)*COS(THETA).
289 F53B B601EAA8      ST A, TEMPR           ; SAVE IN TEMPR.
290 F53F B601D8A8      LD A, SINTH           ; A ← SIN(THETA).
291 F543 3519          JSR SMULT              ; COMPUTE XR(J)*SIN(THETA).
292 F545 B601ECAB      ST A, TEMPI           ; SAVE IN TEMPI.
293 F549 B201E8        LD B, #XI2            ; B ← ADDR(XI2).
294 F54C B601D8A8      LD A, SINTH           ; A ← SIN(THETA).
295 F550 3526          JSR SMULT              ; COMPUTE XI(J)*SIN(THETA).
296 F552 01            COMP A
297 F553 04            INC A
298 F554 B601EAF8      ADD A, TEMPR          ; COMPUTE XR(J)*COS(THETA) -
299                    ; XI(J)*SIN(THETA).
300 F558 B601EAA8      ST A, TEMPR
301 F55C B601D6A8      LD A, COSIN           ; A ← COS(THETA).
302 F560 3536          JSR SMULT              ; COMPUTE XI(J)*COS(THETA).
303 F562 B601ECF8      ADD A, TEMPI          ; COMPUTE XR(J)*SIN(THETA) +
304                    ; XI(J)*COS(THETA).
305 F566 B601ECAB      ST A, TEMPI
306                    ;

```

```

307
308 F56A A401DECEAB ; LD X, R1ADDR ; X ← ADDR(XR(I)).
309 F56F A401EOCCAB LD B, R2ADDR ; B ← ADDR(XR(J)).
310 F574 F0 LD A, W(X+) ; A ← XR(I).
311 F575 02 SET C
312 F576 B601EAEB SUBC A, TEMPR ; A ← XR(I) - TEMPR.
313 F57A E1 XS A, W(B+)
314 F57B 40 NOP
315 F57C F2 LD A, W(X-) ; A ← XI(I).
316 F57D 02 SET C
317 F57E B601ECEB SUBC A, TEMPI ; A ← XI(J) - TEMPI.
318 F582 E6 ST A, W(B)
319 F583 F4 LD A, W(X) ; A ← XR(I).
320 F584 B601EAF8 ADD A, TEMPR ; A ← XR(I) + TEMPR.
321 F588 F1 X A, W(X+)
322 F589 F4 LD A, W(X) ; A ← XI(I).
323 F58A B601ECF8 ADD A, TEMPI ; A ← XI(I) + TEMPI.
324 F58E F6 ST A, W(X)
325
326 F58F A501CA01DAF8 ; ADD M1, ILEAP ; UPDATE M1 FOR NEXT LOOP.
327
328 F595 B601DCAA DECSZ NBCNT ; DONE WITH ALL BUTTERFLIES
329 ; FOR THIS TWIDDLE FACTOR ?
330 F599 959D JMP LOOP3 ; NO, SO GO DO SOME MORE.
331
332
333 F59B B601DOA9 INC ISTART ; SET UP STARTING INDEX FOR
334 ; NEXT TWIDDLE FACTOR.
335 F59F A501CC01D2F8 ADD WEXP, WESTEP ; UPDATE TWIDDLE FACTOR
336 ; EXPONENT VALUE.
337
338 F5A5 B601D4AA DECSZ NTWD ; DONE WITH ALL TWIDDLES
339 ; FOR THIS STAGE ?
340 F5A9 95D7 JMP LOOP2 ; NO, SO GO DO SOME MORE.
341
342
343 F5AB B601CAA8 LD A, ILEAP
344 F5AF E7 SHL A
345 F5B0 B601CAAB ST A, ILEAP ; UPDATE ILEAP FOR NEXT STAGE.
346 F5B4 B601C8A8 LD A, ISTEP
347 F5B8 E7 SHL A
348 F5B9 B601C8AB ST A, ISTEP ; UPDATE ISTEP FOR NEXT STAGE.
349 F5BD B601C6A8 LD A, NBFLY
350 F5C1 C7 SHR A
351 F5C2 B601C6AB ST A, NBFLY ; UPDATE NBFLY FOR NEXT STAGE.
352 F5C6 B601CCA8 LD A, WESTEP
353 F5CA C7 SHR A
354 F5CB B601CCAB ST A, WESTEP ; UPDATE WESTEP FOR NEXT STAGE.
355
356 F5CF B601CEAA DECSZ NSTG ; DONE WITH ALL STAGES ?
357 F5D3 B4FEEB + JMP LOOP1 ; NO SO GO DO SOME MORE.

```

```

358      ;
359 F5D6 3C      RET      ; ALL OVER.
360      ;
361      ; THE CODE BELOW IS FOR THE INVERSE FFT. THE ONLY DIFFERENCE IS THAT
362      ; THE TWIDDLE FACTORS ARE USED A LITTLE DIFFERENTLY, AND A FINAL SCALING BY
363      ; 1/NUMB IS DONE.
364      IFFT:
365      ;
366      ; FIRST PUT THE DATA IN BIT REVERSED ORDER.
367      ;
368 F5D7 00      CLR A
369 F5D8 ABCC     ST A, B      ; SET UP NORMAL COUNTER.
370 F5DA ABCE     ST A, X      ; SET UP BIT REVERSED COUNTER.
371 F5DC A401COCAAB LD K, NUMB ; K HAS NUMBER OF DATA POINTS.
372      IREVLV:
373 F5E1 A0CCCEFD IFGT X, B      ; IS BIT REV CNTR → NORM CNTR ?
374 F5E5 42      JP ISWAP      ; YES, SO SWAP DATA.
375 F5E6 9421     JMP ICOUNT     ; NO SO INCREMENT COUNT.
376      ISWAP:
377 F5E8 AFCC     PUSH B
378 F5EA AFCE     PUSH X
379 F5EC A8CC     LD A, B      ; INDEX VALUE I IS IN A.
380 F5EE E7      SHL A
381 F5EF E7      SHL A
382 F5F0 B80200   ADD A, DTSTAD ; GET ADDR. OF XR(I).
383 F5F3 ABCC     ST A, B      ; SAVE IT IN B.
384 F5F5 A8CE     LD A, X      ; INDEX VALUE J IS IN A.
385 F5F7 E7      SHL A
386 F5F8 E7      SHL A
387 F5F9 B80200   ADD A, DTSTAD ; GET ADDR. OF XR(J).
388 F5FC ABCE     ST A, X      ; SAVE IT IN X.
389 F5FE E4      LD A, W(B)    ; A ← XR(I).
390 F5FF F1      X A, W(X+)    ; A ← XR(J), XR(J) ← XR(I).
391 F600 E1      XS A, W(B+)   ; A ← XR(I), XR(I) ← XR(J).
392 F601 40      NOP
393 F602 E4      LD A, W(B)    ; A ← XI(I).
394 F603 F5      X A, W(X)    ; A ← XI(J), XI(J) ← XI(I).
395 F604 E6      ST A, W(B)   ; XI(I) ← XI(J).
396 F605 3FCE     POP X
397 F607 3FCC     POP B
398      ;
399      ICOUNT:
400      ;
401 F609 AACA     DECSZ K      ; DONE ?
402 F60B 41      JP IUPIT     ; NO GO DO SOME MORE.
403 F60C 46      JP DOIFFT
404      IUPIT:
405 F60D 35F8     JSR BRCNTR   ; COUNT UP ON BIT REV CNTR.
406 F60F A9CC     INC B      ; COUNT UP ON NORMAL CNTR.
407 F611 9530     JMP IREVLV
408      DOIFFT:

```

```

409          ;
410          ; DATA IS NOW STORED IN THE BIT REVERSED ORDER. COMPUTE THE FFT.
411          ;
412 F613 9008          LD A, LMAX          ; A HAS MAX FFT EXPONENT.
413 F615 04          INC A
414 F616 04          INC A
415 F617 02          SET C
416 F618 B601C2EB     SUBC A, L1          ; COMPUTE LSHIFT.
417 F61C B601C4AB     ST A, LSHIFT
418 F620 B601C0A8     LD A, NUMB
419 F624 C7          SHR A
420 F625 B601C6AB     ST A, NBFLY          ; INITIALIZE NBFLY.
421 F629 B601CCAB     ST A, WESTEP          ; INITIALIZE WESTEP.
422 F62D 830101C8AB   LD ISTEP, 01          ; INITIALIZE ISTEP.
423 F632 830201CAAB   LD ILEAP, 02          ; INITIALIZE ILEAP.
424          ;
425          ; SET UP L1 STAGES OF BUTTERFLIES.
426          ;
427 F637 B601C2AB     LD A, L1
428 F63B B601CEAB     ST A, NSTG          ; LOOP L1 TIMES.
429          ILOOP1:
430          ;
431 F63F 00          CLR A
432 F640 B601DOAB     ST A, ISTART          ; INITIALIZE ISTART FOR EACH STAGE.
433 F644 B601D2AB     ST A, WEXP          ; INITIALIZE WEXP.
434          ;
435          ; SET UP ISTEP LOOPS OF TWIDDLE FACTORS.
436          ;
437 F648 B601C8A8     LD A, ISTEP
438 F64C B601D4AB     ST A, NTWD          ; LOOP ISTEP TIMES.
439          ILOOP2:
440          ;
441          ; LOOK UP THE TWIDDLE FACTOR.
442          ;
443 F650 A401C4CAAB   LD K, LSHIFT          ; SHIFT LEFT LSHIFT TIMES.
444 F655 B601D2AB     LD A, WEXP
445          IGADLP:
446 F659 E7          SHL A
447 F65A AACA          DECSZ K          ; DONE SHIFTING ?
448 F65C 63          JP IGADLP          ; NO DO SOME MORE.
449 F65D B8F000       ADD A, TWSTAD          ; ADD STARTING ADDR OF TWIDDLE
450          ;          ; FACTOR TABLE.
451 F660 ABCE          ST A, X          ; TWIDDLE FACTOR ADDR IN X.
452 F662 F0          LD A, W(X+)          ; GET COS(THETA).
453 F663 B601D6AB     ST A, COSTH
454 F667 F4          LD A, W(X)          ; GET SIN(THETA).
455 F668 B601D8AB     ST A, SINTH
456          ;
457 F66C A501D001DAAB LD M1, ISTART          ; INITIALIZE M1.
458          ;
459          ; SET UP NBFLY BUTTERFLIES FOR THIS TWIDDLE FACTOR.

```

```

460                                     ;
461 F672 A501C601DCAB                LD NBCNT, NBFLY          ; LOOP NBFLY TIMES.
462                                     IL00P3:
463 F678 B601DAA8                     LD A, M1                ; GET INDEX OF X(I).
464 F67C E7                           SHL A
465 F67D E7                           SHL A
466 F67E B80200                       ADD A, DTSTAD           ; ADDR. OD XR(I).
467 F681 B601DEA8                     ST A, R1ADDR
468 F685 ABCE                         ST A, X
469 F687 F0                           LD A, W(X+)             ; A ← XR(I).
470 F688 B601E2AB                     ST A, XR1              ; STORE IN XR1.
471 F68C F4                           LD A, W(X)             ; A ← XI(I).
472 F68D B601E4AB                     ST A, XI1              ; STORE IN XI1.
473 F691 B601DAA8                     LD A, M1
474 F695 B601C8F8                     ADD A, ISTEP           ; GET INDEX OF X(J).
475 F699 E7                           SHL A
476 F69A E7                           SHL A
477 F69B B80200                       ADD A, DTSTAD           ; ADDR. OF XR(J).
478 F69E B601EOAB                     ST A, R2ADDR
479 F6A2 ABCE                         ST A, X
480 F6A4 F0                           LD A, W(X+)             ; A ← XR(J).
481 F6A5 B601E6AB                     ST A, XR2              ; STORE IN XR2.
482 F6A9 F4                           LD A, W(X)             ; A ← XI(J).
483 F6AA B601E8AB                     ST A, XI2              ; STORE IN XI2.
484                                     ;
485 F6A8 B201E6                       LD B, #XR2            ; B ← ADDR(XR2).
486 F6B1 B601D6A8                     LD A, COSTH           ; A ← COS(THETA).
487 F6B5 368B                         JSR SMULT              ; COMPUTE XR(J)*COS(THETA).
488 F6B7 B601EAAB                     ST A, TEMPR          ; SAVE IN TEMPR.
489 F6BB B601D8A8                     LD A, SINTH          ; A ← SIN(THETA).
490 F6BF 3695                         JSR SMULT              ; COMPUTE XR(J)*SIN(THETA).
491 F6C1 B601ECAB                     ST A, TEMPI          ; SAVE IN TEMPI.
492 F6C5 B201E8                       LD B, #XI2            ; B ← ADDR(XI2).
493 F6C8 B601D8A8                     LD A, SINTH          ; A ← SIN(THETA).
494 F6CC 36A2                         JSR SMULT              ; COMPUTE XI(J)*SIN(THETA).
495 F6CE 01                           COMP A
496 F6CF 04                           INC A
497 F6D0 B601EAF8                     ADD A, TEMPR          ; COMPUTE XR(J)*COS(THETA) -
498                                     ; XI(J)*SIN(THETA).
499 F6D4 B601EAAB                     ST A, TEMPR
500 F6D8 B601D6A8                     LD A, COSTH           ; A ← COS(THETA).
501 F6DC 36B2                         JSR SMULT              ; COMPUTE XI(J)*COS(THETA).
502 F6DE B601ECF8                     ADD A, TEMPI          ; COMPUTE XR(J)*SIN(THETA) +
503                                     ; XI(J)*COS(THETA).
504 F6E2 B601ECAB                     ST A, TEMPI
505                                     ;
506                                     ;
507 F6E6 A401DECEAB                   LD X, R1ADDR          ; X ← ADDR(XR(I)).
508 F6EB A401EOCCAB                   LD B, R2ADDR          ; B ← ADDR(XR(J)).
509 F6F0 F0                           LD A, W(X+)             ; A ← XR(I).
510 F6F1 02                           SET C

```

```

511 F6F2 B601EAEB      SUBC A, TEMPR      ; A ← XR(I) - TEMPR.
512 F6F6 E1            XS A, W(B+)
513 F6F7 40            NOP
514 F6F8 F2            LD A, W(X-)      ; A ← XI(I).
515 F6F9 02            SET C
516 F6FA B601ECEB      SUBC A, TEMPI      ; A ← XI(J) - TEMPI.
517 F6FE E6            ST A, W(B)
518 F6FF F4            LD A, W(X)      ; A ← XR(I).
519 F700 B601EAF8      ADD A, TEMPR      ; A ← XR(I) + TEMPR.
520 F704 F1            X A, W(X+)
521 F705 F4            LD A, W(X)      ; A ← XI(I).
522 F706 B601ECF8      ADD A, TEMPI      ; A ← XI(I) + TEMPI.
523 F70A F6            ST A, W(X)
524                    ;
525 F70B A501CA01DAF8  ADD M1, ILEAP      ; UPDATE M1 FOR NEXT LOOP.
526                    ;
527 F711 B601DCAA      DECSZ NBCNT      ; DONE WITH ALL BUTTERFLIES
528                    ;               ; FOR THIS TWIDDLE FACTOR ?
529 F715 959D          JMP ILOOP3      ; NO, SO GO DO SOME MORE.
530                    ;
531                    ;
532 F717 B601D0A9      INC ISTART      ; SET UP STARTING INDEX FOR
533                    ;               ; NEXT TWIDDLE FACTOR.
534 F71B A501CC01D2F8  ADD WEXP, WESTEP  ; UPDATE TWIDDLE FACTOR
535                    ;               ; EXPONENT VALUE.
536                    ;
537 F721 B601D4AA      DECSZ NTWD      ; DONE WITH ALL TWIDDLES
538                    ;               ; FOR THIS STAGE ?
539 F725 95D5          JMP ILOOP2      ; NO, SO GO DO SOME MORE.
540                    ;
541                    ;
542 F727 B601CAA8      LD A, ILEAP
543 F72B E7            SHL A
544 F72C B601CAAB      ST A, ILEAP      ; UPDATE ILEAP FOR NEXT STAGE.
545 F730 B601C8A8      LD A, ISTEP
546 F734 E7            SHL A
547 F735 B601C8AB      ST A, ISTEP      ; UPDATE ISTEP FOR NEXT STAGE.
548 F739 B601C6A8      LD A, NBFLY
549 F73D C7            SHR A
550 F73E B601C6AB      ST A, NBFLY      ; UPDATE NBFLY FOR NEXT STAGE.
551 F742 B601CCA8      LD A, WESTEP
552 F746 C7            SHR A
553 F747 B601CCAB      ST A, WESTEP      ; UPDATE WESTEP FOR NEXT STAGE.
554                    ;
555 F748 B601CEAA      DECSZ NSTG      ; DONE WITH ALL STAGES ?
556 F74F B4FEED      + JMP ILOOP1      ; NO SO GO DO SOME MORE.
557                    ;
558                    ;
559                    ; DO THE FINAL SCALING OF THE DATA BY 1/NUMB.
560                    ;
561 F752 B04000          LD A, 04000      ; A ← 1.0

```

```

562 F755 A401C2CAAB      LD K, L1          ; K ← L1.
563                      SCALLP:
564 F75A C7              SHR A              ; DIVIDE BY 2.
565 F75B AACA            DECSZ K
566 F75D 63              JP SCALLP
567                      ;
568                      ; GET HERE MEANS A IS 1/(2*L1).
569 F75E B601EAAB        ST A, TEMPR        ; SAVE IT IN TEMPR.
570 F762 A501C001DCAB    LD NBCNT, NUMB    ; LOOP COUNTER.
571 F768 B20200          LD B, DTSTAD       ; B ← ADDR(XR(0)).
572                      SCALIT:
573 F76B B601EAA8        LD A, TEMPR
574 F76F 3745            JSR SMULT          ; A ← XR(I)*(1/NUMB).
575 F771 E1              XS A, W(B+)        ; XR(I) ← XR(I)*(1/NUMB).
576 F772 40              NOP
577 F773 B601EAA8        LD A, TEMPR        ; A ← 1/NUMB.
578 F777 374D            JSR SMULT          ; A ← X1(I)*(1/NUMB).
579 F779 E1              XS A, W(B+)        ; X1(I) ← X1(I)*(1/NUMB).
580 F77A 40              NOP
581 F77B B601DCAA        DECSZ NBCNT        ; DONE ?
582 F77F 74              JP SCALIT          ; NO DO SOME MORE.
583 F780 3C              RET                ; ALL OVER.
584                      ;
585 FFFE 00F4            .END TSIFFT
NATIONAL SEMICONDUCTOR CORPORATION      PAGE: 13
HPC CROSS ASSEMBLER,REV:C,30 JUL 86

```

SYMBOL TABLE

A	00C8 W	B	00CC W	BRCNTR F415	COSTN 01D6 W
COUNT F48B		DOFFT F495		DOIFFT F613	DTSTAD 0200
FFT F459		GADLP F40B		ICOUNT F609	IFFT F507
IGADLP F659		ILEAP 01CA W		IL00P1 F63F	IL00P2 F650
IL00P3 F678		IREVLP F5E1		ISTART 01D0 W	ISTEP 01C8 W
ISWAP F5E8		IUPIT F600		K 00CA W	L1 01C2 W
LMAX 0008		LOOP1 F4C1		LOOP2 F4D2	LOOP3 F4FC
LSHIFT 01C4 W		M1 01DA W		MTEMP 01EE W	NBCNT 01DC W
NBFLY 01C6 W		NSTG 01CE W		NTWD 01D4 W	NUMB 01C0 W
PC 00C6 W		RIADDR 01DE W		R2ADDR 01E0 W	REVLP F463
SCALIT F76B		SCALLP F75A		SINTH 01D8 W	SMULT F42A
SP 00C4 W		SWAP F46A		TEMPI 01EC W	TEMPR 01EA W
TSIFFT F400		TWSTAD F000		UPIT F48F	WESTEP 01CC W
WEXP 01D2 W		X 00CE W		XI1 01E4 W	XI2 01E8 W
XR1 01E2 W		XR2 01E6 W		\$FOUND F425	\$REPEA F41A

NATIONAL SEMICONDUCTOR CORPORATION
HPC CROSS ASSEMBLER,REV:C,30 JUL 86

PAGE: 14

MACRO TABLE

NO WARNING LINES

NO ERROR LINES

2307 ROM BYTES USED

SOURCE CHECKSUM = E9FC
OBJECT CHECKSUM = 28FC

INPUT FILE C:FFT.MAC
LISTING FILE C:FFT.PRN
OBJECT FILE C:FFT.LM

APPENDIX B

Twiddle Factor Table

```

;
; TWIDDLE FACTOR TABLE FOR USE IN THE FFT ROUTINES.
;
; TABLE SET FOR MAX FFT LENGTH OF 256.
;
; TABLE STARTS AT F000 AND OCCUPIES 1024 BYTES OF STORAGE.
;

      . = 0F000
      .WORD 16384, 0
      .WORD 16379, 402
      .WORD 16364, 804
      .WORD 16340, 1205
      .WORD 16305, 1606
      .WORD 16261, 2006
      .WORD 16207, 2404
      .WORD 16143, 2801
      .WORD 16069, 3196
      .WORD 15986, 3590
      .WORD 15893, 3981
      .WORD 15791, 4370
      .WORD 15679, 4756
      .WORD 15557, 5139
      .WORD 15426, 5520
      .WORD 15286, 5897
      .WORD 15137, 6270
      .WORD 14978, 6639
      .WORD 14811, 7005
      .WORD 14635, 7366
      .WORD 14449, 7723
      .WORD 14256, 8076
      .WORD 14053, 8423
      .WORD 13842, 8765
      .WORD 13623, 9102
      .WORD 13395, 9434
      .WORD 13160, 9760
      .WORD 12916, 10080
      .WORD 12665, 10394
      .WORD 12406, 10702
      .WORD 12140, 11003
      .WORD 11866, 11297
      .WORD 11585, 11585
      .WORD 11297, 11866
      .WORD 11003, 12140
      .WORD 10702, 12406
      .WORD 10394, 12665
      .WORD 10080, 12916
      .WORD 9760, 13160
      .WORD 9434, 13395
      .WORD 9102, 13623
      .WORD 8765, 13842
      .WORD 8423, 14053
      .WORD 8076, 14256
      .WORD 7723, 14449
      .WORD 7366, 14635
      .WORD 7005, 14811

      .WORD 6639, 14978
      .WORD 6270, 15137
      .WORD 5897, 15286
      .WORD 5520, 15426
      .WORD 5139, 15557
      .WORD 4756, 15679
      .WORD 4370, 15791
      .WORD 3981, 15893
      .WORD 3590, 15986
      .WORD 3196, 16069
      .WORD 2801, 16143
      .WORD 2404, 16207
      .WORD 2006, 16261
      .WORD 1606, 16305
      .WORD 1205, 16340
      .WORD 804, 16364
      .WORD 402, 16379
      .WORD 0, 16384
      .WORD -402, 16379
      .WORD -804, 16364
      .WORD -1205, 16340
      .WORD -1606, 16305
      .WORD -2006, 16261
      .WORD -2404, 16207
      .WORD -2801, 16143
      .WORD -3196, 16069
      .WORD -3590, 15986
      .WORD -3981, 15893
      .WORD -4370, 15791
      .WORD -4756, 15679
      .WORD -5139, 15557
      .WORD -5520, 15426
      .WORD -5897, 15286
      .WORD -6270, 15137
      .WORD -6639, 14978
      .WORD -7005, 14811
      .WORD -7366, 14635
      .WORD -7723, 14449
      .WORD -8076, 14256
      .WORD -8423, 14053
      .WORD -8765, 13842
      .WORD -9102, 13623
      .WORD -9434, 13395
      .WORD -9760, 13160
      .WORD -10080, 12916
      .WORD -10394, 12665
      .WORD -10702, 12406
      .WORD -11003, 12140
      .WORD -11297, 11866
      .WORD -11585, 11585
      .WORD -11866, 11297
      .WORD -12140, 11003
      .WORD -12406, 10702
      .WORD -12665, 10394
      .WORD -12916, 10080

```

.WORD -13160, 9760
 .WORD -13395, 9434
 .WORD -13623, 9102
 .WORD -13842, 8765
 .WORD -14053, 8423
 .WORD -14256, 8076
 .WORD -14449, 7723
 .WORD -14635, 7366
 .WORD -14811, 7005
 .WORD -14978, 6639
 .WORD -15137, 6270
 .WORD -15286, 5897
 .WORD -15426, 5520
 .WORD -15557, 5139
 .WORD -15679, 4756
 .WORD -15791, 4370
 .WORD -15893, 3981
 .WORD -15986, 3590
 .WORD -16069, 3196
 .WORD -16143, 2801
 .WORD -16207, 2404
 .WORD -16261, 2006
 .WORD -16305, 1606
 .WORD -16340, 1205
 .WORD -16364, 804
 .WORD -16379, 402
 .WORD -16384, 0

.WORD -16379, -402
 .WORD -16364, -804
 .WORD -16340, -1205
 .WORD -16305, -1606
 .WORD -16261, -2006
 .WORD -16207, -2404
 .WORD -16143, -2801
 .WORD -16069, -3196
 .WORD -15986, -3590
 .WORD -15893, -3981
 .WORD -15791, -4370
 .WORD -15679, -4756
 .WORD -15557, -5139
 .WORD -15426, -5520
 .WORD -15286, -5897
 .WORD -15137, -6270
 .WORD -14978, -6639
 .WORD -14811, -7005
 .WORD -14635, -7366
 .WORD -14449, -7723
 .WORD -14256, -8076
 .WORD -14053, -8423
 .WORD -13842, -8765
 .WORD -13623, -9102
 .WORD -13395, -9434
 .WORD -13160, -9760

.WORD -12916, -10080
 .WORD -12665, -10394
 .WORD -12406, -10702
 .WORD -12140, -11003
 .WORD -11866, -11297
 .WORD -11585, -11585
 .WORD -11297, -11866
 .WORD -11003, -12140
 .WORD -10702, -12406
 .WORD -10394, -12665
 .WORD -10080, -12916
 .WORD -9760, -13160
 .WORD -9434, -13395
 .WORD -9102, -13623
 .WORD -8765, -13842
 .WORD -8423, -14053
 .WORD -8076, -14256
 .WORD -7723, -14449
 .WORD -7366, -14635
 .WORD -7005, -14811
 .WORD -6639, -14978
 .WORD -6270, -15137
 .WORD -5897, -15286
 .WORD -5520, -15426
 .WORD -5139, -15557
 .WORD -4756, -15679
 .WORD -4370, -15791
 .WORD -3981, -15893
 .WORD -3590, -15986
 .WORD -3196, -16069
 .WORD -2801, -16143
 .WORD -2404, -16207
 .WORD -2006, -16261
 .WORD -1606, -16305
 .WORD -1205, -16340
 .WORD -804, -16364
 .WORD -402, -16379
 .WORD 0, -16384
 .WORD 402, -16379
 .WORD 804, -16364
 .WORD 1205, -16340
 .WORD 1606, -16305
 .WORD 2006, -16261
 .WORD 2404, -16207
 .WORD 2801, -16143
 .WORD 3196, -16069
 .WORD 3590, -15986
 .WORD 3981, -15893
 .WORD 4370, -15791
 .WORD 4756, -15679
 .WORD 5139, -15557
 .WORD 5520, -15426
 .WORD 5897, -15286
 .WORD 6270, -15137
 .WORD 6639, -14978

.WORD 7005, -14811
.WORD 7366, -14635
.WORD 7723, -14449
.WORD 8076, -14256
.WORD 8423, -14053
.WORD 8765, -13842
.WORD 9102, -13623
.WORD 9434, -13395
.WORD 9760, -13160
.WORD 10080, -12916
.WORD 10394, -12665
.WORD 10702, -12406
.WORD 11003, -12140
.WORD 11297, -11866
.WORD 11585, -11585
.WORD 11866, -11297
.WORD 12140, -11003
.WORD 12406, -10702
.WORD 12665, -10394
.WORD 12916, -10080
.WORD 13160, -9760
.WORD 13395, -9434
.WORD 13623, -9102
.WORD 13842, -8765
.WORD 14053, -8423
.WORD 14256, -8076
.WORD 14449, -7723
.WORD 14635, -7366
.WORD 14811, -7005
.WORD 14978, -6639
.WORD 15137, -6270
.WORD 15286, -5897
.WORD 15426, -5520
.WORD 15557, -5139
.WORD 15679, -4756
.WORD 15791, -4370
.WORD 15893, -3981
.WORD 15986, -3590
.WORD 16069, -3196
.WORD 16143, -2801
.WORD 16207, -2404
.WORD 16261, -2006
.WORD 16305, -1606
.WORD 16340, -1205
.WORD 16364, -804
.WORD 16379, -402

.END

APPENDIX C

Test Data and Expected Results

NATIONAL SEMICONDUCTOR CORPORATION
HPC CROSS ASSEMBLER,REV:C,30 JUL 86

PAGE: 1

```
1          ;
2          ;
3          ; TEST DATA FOR FFT ROUTINES.
4          ; OBTAINED FROM : PROGRAMS FOR DIGITAL SIGNAL PROCESSING, IEEE PRESS,
5          ; CHAPTER 1 BY GOLD.
6          ;
7          ;
8      0200          . = 0200
9 0200 0004          .WORD 1024, 0
      0202 0000
10 0204 9A03          .WORD 922, 307
      0206 3301
11 0208 E102          .WORD 737, 553
      020A 2902
12 020C F201          .WORD 498, 719
      020E CF02
13 0210 E800          .WORD 232, 796
      0212 1C03
14 0214 E2FF          .WORD -30, 786
      0216 1203
15 0218 F9FE          .WORD -263, 699
      021A BB02
16 021C 42FE          .WORD -446, 550
      021E 2602
17 0220 C9FD          .WORD -567, 361
      0222 6901
18 0224 96FD          .WORD -618, 155
      0226 9800
19 0228 A5FD          .WORD -603, -46
      022A D2FF
20 022C EFFF          .WORD -529, -222
      022E 22FF
21 0230 67FE          .WORD -409, -359
      0232 99FE
22 0234 FBFE          .WORD -261, -446
      0236 42FE
23 0238 9BFF          .WORD -101, -479
      023A 21FE
24 023C 3500          .WORD 53, -462
      023E 32FE
25 0240 BA00          .WORD 186, -400
      0242 70FE
26 0244 1F01          .WORD 287, -304
      0246 D0FE
27 0248 5E01          .WORD 350, -187
      024A 45FF
28 024C 7301          .WORD 371, -64
      024E C0FF
29 0250 6101          .WORD 353, 54
      0252 3600
30 0254 2001          .WORD 301, 154
```

```

0256 9A00
31 0258 E100          .WORD 225, 229
025A E500
32 025C 8600          .WORD 134, 274
025E 1201
33 0260 2600          .WORD 38, 287
0262 1F01
34 0264 CCFF          .WORD -52, 269
0266 OD01
35 0268 81FF          .WORD -127, 227
026A E300
36 026C 49FF          .WORD -183, 166
026E A600
37 0270 2AFF          .WORD -214, 95
0272 5F00
38 0274 23FF          .WORD -221, 21
0276 1500
39 0278 33FF          .WORD -205, -48
027A DDFD
40 027C 55FF          .WORD -171, -104
027E 98FF
41
42      ;
43      ; THESE ARE THE EXPECTED DFT RESULTS.
44      ;
45 0280 C702          .WORD 711, 3584
0282 000E
46 0284 2B0B          .WORD 2859, 8244
0286 3420
47 0288 9D25          .WORD 9629, -9354
028A 76DB
48 028C 7707          .WORD 1911, -3926
028E AAFO
49 0290 8704          .WORD 1159, -2288
0292 10F7
50 0294 9F03          .WORD 927, -1571
0296 DDF9
51 0298 3303          .WORD 819, -1167
029A 71FB
52 029C F502          .WORD 757, -903
029E 79FC
53 02A0 CE02          .WORD 718, -715
02A2 35FD
54 02A4 B202          .WORD 690, -572
02A6 C4FD
55 02A8 9D02          .WORD 669, -457
02AA 37FE
56 02AC 8C02          .WORD 652, -361
02AE 97FE
57 02B0 7F02          .WORD 639, -279
02B2 E9FE
02B4 7302          .WORD 627, -205

```

```

0286 33FF
58 02B8 6902          .WORD 617, -139
02BA 75FF
59 02BC 6002          .WORD 608, -77
02BE B3FF
60 02C0 5802          .WORD 600, -18
02C2 EEFF
61 02C4 5102          .WORD 593, 39
02C6 2700
62 02C8 4A02          .WORD 586, 95
02CA 5F00
63 02CC 4302          .WORD 579, 152
02CE 9800
64 02D0 3C02          .WORD 572, 210
02D2 0200
65 02D4 3502          .WORD 565, 271
02D6 0F01
66 02D8 2E02          .WORD 558, 336
02DA 5001
67 02DC 2702          .WORD 551, 408
02DE 9801
68 02E0 2002          .WORD 544, 488
02E2 EB01
69 02E4 1802          .WORD 536, 581
02E6 4502
70 02E8 1002          .WORD 528, 691
02EA B302
71 02EC 0702          .WORD 519, 827
02EE 3B03
72 02F0 FE01          .WORD 510, 1004
02F2 EC03
73 02F4 F701          .WORD 503, 1248
02F6 E004
74 02F8 F701          .WORD 503, 1615
02FA 4F06
75 02FC 1202          .WORD 530, 2241
02FE C108
76          ;
77          ;
78          ; TEST DATA FOR FFT ROUTINES.
79          ; OBTAINED FROM :
80          ;
81          ;
82 0300 0004          .WORD 1024, 0
0302 0000
83 0304 9A03          .WORD 922, 307
0306 3301
84 0308 E102          .WORD 737, 553
030A 2902
85 030C F201          .WORD 498, 719
030E CF02

```

86 0310 E800	.WORD 232, 796
0312 1C03	
87 0314 E2FF	.WORD -30, 786
0316 1203	
88 0318 F9FE	.WORD -263, 699
031A BB02	
89 031C 42FE	.WORD -446, 550
031E 2602	
90 0320 C9FD	.WORD -567, 361
0322 6901	
91 0324 96FD	.WORD -618, 155
0326 9B00	
92 0328 A5FD	.WORD -603, -46
032A D2FF	
93 032C EFFD	.WORD -529, -222
032E 22FF	
94 0330 67FE	.WORD -409, -359
0332 99FE	
95 0334 FBFE	.WORD -261, -446
0336 42FE	
96 0338 9BFF	.WORD -101, -479
033A 21FE	
97 033C 3500	.WORD 53, -462
033E 32FE	
98 0340 BA00	.WORD 186, -400
0342 70FE	
99 0344 1F01	.WORD 287, -304
0346 D0FE	
100 0348 5E01	.WORD 350, -187
034A 45FF	
101 034C 7301	.WORD 371, -64
034E C0FF	
102 0350 6101	.WORD 353, 54
0352 3600	
103 0354 2D01	.WORD 301, 154
0356 9A00	
104 0358 E100	.WORD 225, 229
035A E500	
105 035C 8600	.WORD 134, 274
035E 1201	
106 0360 2600	.WORD 38, 287
0362 1F01	
107 0364 CCFF	.WORD -52, 269
0366 OD01	
108 0368 81FF	.WORD -127, 227
036A E300	
109 036C 49FF	.WORD -183, 166
036E A600	
110 0370 2AFF	.WORD -214, 95
0372 5F00	
111 0374 23FF	.WORD -221, 21

NATIONAL SEMICONDUCTOR CORPORATION
HPC CROSS ASSEMBLER,REV:C,30 JUL 86

PAGE: 5

0376 1500
112 0378 33FF .WORD -205, -48
037A D0FF
113 037C 55FF .WORD -171, -104
037E 98FF
114 ;
115 .END
@
ERROR, OPERAND MUST BE SINGLE VALID SYMBOL NAME

NATIONAL SEMICONDUCTOR CORPORATION
HPC CROSS ASSEMBLER,REV:C,30 JUL 86

PAGE: 6

SYMBOL TABLE

A	00C8 W	B	00CC W	K	00CA W	PC	00C6 W
SP	00C4 W	X	00CE W				

MACRO TABLE

NO WARNING LINES

1 ERROR LINES

384 ROM BYTES USED

SOURCE CHECKSUM = 7A03
OBJECT CHECKSUM = 0705

INPUT FILE C:TSTDAT.MAC
LISTING FILE C:TSTDAT.PRN

LIFE SUPPORT POLICY

NATIONAL'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF NATIONAL SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and whose failure to perform, when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.

2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

 National Semiconductor Corporation 1111 West Bardin Road Arlington, TX 76017 Tel: 1(800) 272-9959 Fax: 1(800) 737-7018	National Semiconductor Europe Fax: (+49) 0-180-530 85 86 Email: cnjwge@tevm2.nsc.com Deutsch Tel: (+49) 0-180-530 85 85 English Tel: (+49) 0-180-532 78 32 Français Tel: (+49) 0-180-532 93 58 Italiano Tel: (+49) 0-180-534 16 80	National Semiconductor Hong Kong Ltd. 19th Floor, Straight Block, Ocean Centre, 5 Canton Rd. Tsimshatsui, Kowloon Hong Kong Tel: (852) 2737-1600 Fax: (852) 2736-9960	National Semiconductor Japan Ltd. Tel: 81-043-299-2309 Fax: 81-043-299-2408
---	---	--	--

National does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied and National reserves the right at any time without notice to change said circuitry and specifications.