APPENDIX A

Fully Optimized Complex FFT

A.1 Optimized Complex FFT for the DSP96002

```
;*************************************************************************
;                                                                        *
;   RMAXS.ASM  : START PROGRAM FOR THE FFT MACRO RMAX.ASM.               *
;   THIS FILE SHOWS HOW TO CONFIGURE MOTOROLAS DSP96002                  *
;   TO USE THE FAST COMPLEX FFT.                                         *
;   TWIDDLE FACTORS IN R4TAB1.ASM                                        *
;                                                                        *
;   WRITTEN BY : KARL SCHWARZ, RAIMUND MEYER       10.11.89              *
;                                                                        *
;       LEHRSTUHL FUER NACHRICHTENTECHNIK                                *
;       UNIVERSITAET ERLANGEN-NUERNBERG                                  *
;                                                                        *
;*************************************************************************
    include 'r4tab1'
    include'rmax'

    points     equ  1024          ; FFT-length, only 1024 possible
    passes     equ  10            ; ld(points)
    data       equ  $800          ; input data, normal order
    odata      equ  $C00          ; output data, normal order
    tab4       equ  $1000         ; start of radix-4 twiddle factors
                                  ; 766 complex values)

    r4  tab1   tab4

        org    p:$100

        move   #$008A0000,x:$FFFFFFFD      ; zero wait states in BCRB
        move   #$008A0000,x:$FFFFFFFE      ; zero wait states in BCRA
        move   #$0000FFFF,x:$FFFFFFFC      ; X port A, Y and P port B in PSR
                          ;Upper three moves won't count for the benchmark,
                          ;only for initializing the simulator or the DSP.
                          ;They show how to configure the device.
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 1 of 20)

```
;   A T T E N T I O N   P L E A S E !!!!!!!

;   STEP THROUGH THE FIRST THREE LINES, THEN LOAD THE SIMULATOR NEW
;   WITH RMAXS AND INPUT VECTORS, THEN GET A NEW RUN

    rmaxpoints,passes,data,odata,tab4

    nop
    nop
    nop

        end
;*************************************************************************
;                                                                        *
;               COMPLEX, RADIX-2,4 DIT FFT  :  RMAX.ASM                  *
;            ------------------------------------------                  *
;                                                                        *
;   MACRO FOR A FAST LOOPED-CODE MIXED-RADIX DIT FFT COMPUTATION         *
;                        IN DSP96002                                     *
;                                                                        *
;    WRITTEN BY: KARL SCHWARZ, RAIMUND MEYER       10.11.89            *
;                                                                        *
;         LEHRSTUHL FUER NACHRICHTENTECHNIK                              *
;         UNIVERSITAET ERLANGEN-NUERNBERG                                *
;                                                                        *
;   REVISION : THIS PROGRAM IS SPEEDED UP FROM RMIX1.ASM                 *
;                                                                        *
;   PLEASE LOOK IN THE START FILE RMAXS.ASM HOW TO CONFIGURE THE DEVICE  *
;                                                                        *
;   FOR THIS PROGRAM THE FFTLENGTH IS 1024 POINTS                        *
;   SPECIAL FEATURES : RADIX-4 BUTTERFLY IN FIRST AND LAST TWO STAGES    *
;   SIMPLE RADIX-4 BUTTERFLY IN 1. TO 6. STAGE IF NO TWIDDLES ARE USED   *
;   TABLE IN USE : ONLY R4TAB1.ASM FOR RADIX-2 AND LAST RADIX-4 BUTTERFLY*
;   LOOK IN R4TAB1.M HOW TO BUILT A TABLE                                *
;                                                                        *
;*************************************************************************
;                                                                        *
;   EXAMPLE FOR THE 1024 POINT COMPLEX FFT (WITH BITREVERSAL) :          *
;                                                                        *
;   MEMORY SIZE :  PROGRAM   :  219 WORDS                                *
;                  DATA      : 4096 WORDS                                *
;          TWIDDLE FACTORS   : 1532 WORDS                                *
;                                                                        *
;   CYCLES PER BUTTERFLY :                                               *
;       1. AND 2. STAGE:     2                                           *
;       3. AND 4. STAGE:     3.5                                         *
;       5. AND 6. STAGE:     3.875                                       *
;       7.  STAGE   :        4                                           *
;       8.  STAGE   :        4.25                                        *
;       9. AND 10.STAGE :    4.25                                        *
;   AVERAGE CYCLES/BUTTERFLY:    3.55                                    *
;   TOTAL BUTTERFLYCYCLES :      18176                                   *
;   INITIALIZATION OVERHEAD:     715  =  3.8 % OF TOTAL TIME             *
;   TOTAL NUMBER OF INSTRUCTION CYCLES  : 18891                          *
;   TOTAL TIME FOR A 1024 POINT FFT:      1.399 msAT 27 MHz              *
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 2 of 20)

```
;*****************************************************************************
;                                                                            *
;   USED RADIX-2 BUTTERFLY                                                   *
;                                                                            *
;   AR + j AI ------------------------O-------------O--------- AR' + j AI'   *
;                                      \          / +                        *
;                                       \       /                          *
;                                        \ /                               *
;                                        / \                               *
;                                       /     \                            *
;                                      /         \ +                        *
;   BR + j BI ---- ( COS - j SIN ) --O-------------O--------- BR' + j BI'   *
;                                    -                                       *
;                                                                            *
;   TR = BR * COS + BI * SIN                                                 *
;   TI = BR * SIN - BI * COS                                                 *
;   AR'= AR + TR                                                             *
;   AI'= AI - TI                                                             *
;   BR'= AR - TR                                                             *
;   BI'= AI + TI                                                             *
;                                                                            *
;*****************************************************************************
;                                                                            *
;   USED RADIX-4 BUTTERFLY                                                   *
;                                                                            *
;   AR + j AI ----------O-----------O----------O-----O--- AR' + j AI'       *
;                        \         /            \ /                        *
;                         \   /                 / \                        *
;   BR + j BI ---(W1)---O-----X-----O----------O-----O--- BR' + j BI'       *
;                        \ /   \ /                -                         *
;                        / \   / \                                          *
;   CR + j CI ---(W2)---O-----X-----O----------O-----O--- CR' + j CI'       *
;                          /   \  -           \ /                          *
;                        /       \           / \                          *
;   DR + j DI ---(W3)---O-----------O---(-j)---O-----O--- DR' + j DI'       *
;                                   -             -                        *
;                                                                            *
; MIXING OF RADIX-2 AND RADIX-4 BUTTERFLIES IS POSSIBLE WITHOUT TROUBLE !    *
;                                                                            *
;*****************************************************************************

; ---> about 10 % faster than ICASSP 89 Paper 40.D9.7 by Kloker and Lindsley

rmaxmacropoints,passes,data,odata,tab4

;points  : FFT-length (power of 2)
;passes  : log2(points)
;data    : start address of input vector
;odata   : start address of output vector due to bitreversal
;tab4    : start address of radix-4 twiddle factors from file r4tab.asm

    pam5    equ     passes-5
    pg2     equ     points/2
    pg4     equ     points/4
    pg8     equ     points/8
```

Figure A-1 Optimized Complex FFT for the DSP96002(sheet 3 of 20)

```
        pg 16       equ       points/16
        pg 32       equ       points/32
        pg 64       equ       points/64
        pg 128      equ       points/128
        pg 4m1      equ       points/4-1
        pg 16m1     equ       points/16-1
        pg 64m1     equ       points/64-1


;*************************************************************************
;  ------------ FIRST 2 STAGES AS RADIX-4 BUTTERFLY ----------------------- *
;*************************************************************************

        move   #-1,m0
        move   m0,m1
        move   m0,m2
        move   m0,m3
        move   m0,m4
        move   m0,m5
        move   m0,m6
        move   m0,m7
        move   #data,r0
        move   #(data+pg4),r1
        move   #(data+2*pg4),r2
        move   #(data+3*pg4),r3
        move   #2,n0
        move   n0,n6
        move   n0,n5
        move   n0,n7
        move   #pg4m1,n1
    jsr _sr4

;*************************************************************************
;  ------------ PARTS OF 3. AND 4. STAGE AS SPECIAL RADIX-4 BUTTERFLY ------ *
;*************************************************************************

        move   #data,r0
        move   #(data+pg16),r1
        move   #(data+2*pg16),r2
        move   #(data+3*pg16),r3
        move   #pg 16m1,n1
    jsr _sr4


;*************************************************************************
;  ------------ PARTS OF 5. AND 6. STAGE AS SPECIAL RADIX-4 BUTTERFLY ------ *
;*************************************************************************

        move   #data,r0
        move   #(data+pg64),r1
        move   #(data+2*pg64),r2
        move   #(data+3*pg64),r3
        move   #pg 64m1,n1
    jsr _sr4
```

Figure A-1 Optimized Complex FFT for the DSP96002(sheet 4 of 20)

```
;*************************************************************************
;
; ------------ REST OF 3. STAGE AS RADIX-2 BUTTERFLY ---------------------- *
;*************************************************************************
;
       move   #3,n6                  ; step for twiddle addressing in r4tab

       move   #(tab4+3),r6           ; address of sin cos table
       move   #(data+pg4),r0         ; input vector
       move   #(data+pg4+pg8),r1
       move   #3,n7                  ; still 3 r2 groups to calculate
       move   #(pg 8-3),r7           ; pg8 r2 butterflies in a group
       move   #(pg 8+1),n0           ; step to next group
   jsr _nr2

;*************************************************************************
;
; ------------ REST OF 4. STAGE AS RADIX-2 BUTTERFLY ---------------------- *
;*************************************************************************
;
       move   #(tab4+6),r6           ; address of sin cos table
       move   #(data+pg4),r0         ; input vector
       move   #(data+pg4+pg16),r1
       move   #6,n7                  ; still 6 r2 groups to calculate
       move   #(pg16-3),r7           ; pg16 r2 butterflies in a group
       move   #(pg16+1),n0           ; step to next group
   jsr _nr2

;*************************************************************************
;
; ------------ REST OF 5. STAGE AS RADIX-2 BUTTERFLY ---------------------- *
;*************************************************************************
;
       move   #(tab4+3),r6           ; address of sin cos table
       move   #(data+pg16),r0        ; input vector
       move   #(data+pg16+pg32),r1
       move   #15,n7                 ; still 15 r2 groups to calculate
       move   #(pg32-3),r7           ; pg32 r2 butterflies in a group
       move   #(pg32+1),n0           ; step to next group
   jsr _nr2

;*************************************************************************
;
; ------------ REST OF 6. STAGE AS RADIX-2 BUTTERFLY ---------------------- *
;*************************************************************************
;
       move   #(tab4+6),r6           ; address of sin cos table
       move   #(data+pg16),r0        ; input vector
       move   #(data+pg16+pg64),r1
       move   #30,n7                 ; still 30 r2 groups to calculate
       move   #(pg64-3),r7           ; pg64 r2 butterflies in a group
       move   #(pg64+1),n0           ; step to next group
   jsr _nr2

;*************************************************************************
;
; ------------ 7. STAGE AS RADIX-2 BUTTERFLY ------------------------------ *
;*************************************************************************
;
       move   #(tab4),r6             ; address of sin cos table
       move   #(data),r0             ; input vector
```

Figure A-1 Optimized Complex FFT for the DSP96002(sheet 5 of 20)

```
    move      #(data+pg128),r1
    move      #64,n7                ; still 64 r2 groups to calculate
    move      #(pg128-3),r7         ; pg128 r2 butterflies in a group
    move      #(pg128+1),n0         ; step to next group
jsr _nr2

;*************************************************************************
; ------------ 8. STAGE AS RADIX-2 BUTTERFLY ---------------------------- *
;*************************************************************************

    move      #5,n0
    move      #(data+4),r1
    move      #tab4,r6
    move      #data,r0
    move      r1,r5
    mover     0,r4
    move      n0,n1
    move      n0,n4
    move      n0,n5

    move              x:(r6),d9.s       y:,d8.s
    move              x:(r1)+,d0.s      y:,d1.s
    move              x:(r0),d4.s       y:(r6),d2.s
    move                                y:(r1),d7.s
    faddsub.s d4,d0   x:(r1)+,d6.s      y:(r6)+n6,d3.s

do       #pg8,_end3                           ; loop of groups

    fmpy      d9,d6,d0      fsub.s    d1,d2    d0.s,x:(r4)   y:(r0)+,d5.s
    fmpy      d9,d7,d1      faddsub.s d5,d2    d4.s,x:(r5)   y:(r1),d7.s
    fmpy      d8,d6,d2      fadd.s    d3,d0    x:(r0),d4.s   d2.s,y:(r5)+
    fmpy      d8,d7,d3      faddsub.s d4,d0    x:(r1)+,d6.s  d5.s,y:(r4)+

    fmpy      d9,d6,d0      fsub.s    d1,d2    d0.s,x:(r4)   y:(r0)+,d5.s
    fmpy      d9,d7,d1      faddsub.s d5,d2    d4.s,x:(r5)   y:(r1),d7.s
    fmpy      d8,d6,d2      fadd.s    d3,d0    x:(r0),d4.s   d2.s,y:(r5)+

    fmpy      d8,d7,d3      faddsub.s d4,d0    x:(r1)+n1,d6.s    d5.s,y:(r4)+
    fmpy      d9,d6,d0      fsub.s    d1,d2    d0.s,x:(r4)   y:(r0)+,d5.s
    fmpy      d9,d7,d1      faddsub.s d5,d2    d4.s,x:(r5)   y:(r1),d7.s
    fmpy      d8,d6,d2      fadd.s    d3,d0    x:(r0),d4.s   d2.s,y:(r5)+
    move                                       x:(r6)+n6,d9.s      y:,d8.s
    fmpy      d8,d7,d3      faddsub.s d4,d0    x:(r1)+,d6.s  d5.s,y:(r4)+
    fmpy      d9,d6,d0      fsub.s    d1,d2    d0.s,x:(r4)   y:(r0)+n0,d5.s
    fmpy      d9,d7,d1      faddsub.s d5,d2    d4.s,x:(r5)   y:(r1),d7.s
    fmpy      d8,d6,d2      fadd.s    d3,d0    x:(r0),d4.s   d2.s,y:(r5)+n5

    fmpy      d8,d7,d3      faddsub.s d4,d0    x:(r1)+,d6.s  d5.s,y:(r4)+n4
_end3

;*************************************************************************
; ------------ LAST TWO STAGES AS RADIX-4 BUTTERFLY ---------------------- *
;*************************************************************************

    move      #$0,m2
    move      m2,m3
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 6 of 20)

```
    movem     2,m5
    movem     2,m7
    move      #data,r0
    move      #(data+1),r4
    move      #(data+2),r1
    move      #(tab4+1),r6
    move      #2,n4
    move      #4,n0
    move      n0,n1
    move      #odata,r5
    move      #(odata+pg2),r2
    move      #(odata+pg4),r7
    move      #(odata+pg4*3),r3
    move      #pg8,n5
    move      n5,n2
    move      n5,n7
    move      n5,n3

    move              x:(r4)+n4,d3.s     y:,d5.s   ;d3=Br,d5=Bi
    move              x:(r4)+n4,d1.s     y:,d2.s   ;d1=Dr,d2=Di
    faddsub.s d1,d3   x:(r0),d7.s                  ;d3=Br+Dr,d1=Br-Dr,d7=Ar
    faddsub.s d5,d2   x:(r1),d0.sd1.s,   y:(r7)    ;d5=Bi+di,d2=Bi-Di,d0=Cr,
                                                   ;temp store Br-Dr
    faddsub.s d7,d0   d3.s,d4.s     y:(r1)+n1,d1.s     ;d0=Ar+Cr,d7=Ar-
                                                       ;Cr,d4=Br+Dr,d1=Ci
    faddsub.s d7,d5   x:(r4),d6.s   y:(r0)+n0,d3.s     ;d7=Ar-Cr-(bi+Di)
    faddsub.s d0,d4   d7.s,x:(r3)   y:(r4)+n4,d7.s

    do  #pg4m1,_er4
    faddsub.s d3,d1         x:(r6)+,d9.sy:,d8.s
    fmpy.s    d6,d9,d5      d5.s,x:(r7)
    fmpy      d7,d8,d3      faddsub.s     d1,d2d4.s,    x:(r5)d3.s,d4.s
    fmpy      d6,d8,d1      fadd.s        d5,d3d0.s,    x:(r2)+n2d1.s,y:
    fmpy.s    d7,d9,d5                    x:(r6)+,d9.s  y:,d8.s
    fsub.s    d1,d5                       x:(r4)+n4,d6.sy:,d7.s
    fmpy.s    d6,d9,d1                                  y:(r7),d0.s
    fmpy      d7,d8,d2      faddsub.s     d4,d0    d2.s,y:(r5)+n5
    fmpy      d6,d8,d0      fadd.s        d2,d1    x:(r1),d6.sd0.s,y:(r7)+n7
    fmpy      d7,d9,d2      faddsub.s     d1,d3    x:(r6)+,d9.sy:,d8.s
    fmpy      d6,d9,d0      fsub.s        d0,d2    y:(r1)+n1,d7.s
    fmpy      d7,d8,d3      faddsub.s     d5,d2    d3.s,d4.d4.s,y:(r3)+n3
    fmpy      d7,d9,d1      fadd.s        d3,d0    x:(r0),d7.sd1.s,y:(r7)
    fmpy      d6,d8,d3      faddsub.s     d7,d0
    faddsub.s d7,d5
    faddsub.s d0,d4d7.s,          x:(r3)        y:(r4),d7.s
    fsub.s    d3,d1               x:(r4)+n4,d6.sy:(r0)+n0,d3.s
_er4
    faddsub.s      d3,d1d5.s,     x:(r7)
    faddsub.s      d1,d2                        y:(r7),d6.s
    move           d0.s,          x:(r2)d1.s,   y:
    faddsub.s      d3,d6d4.s,     x:(r5)d2.s,   y:
    move           d6.s,                        y:(r7)
    move           d3.s,                        y:(r3)
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 7 of 20)

```
;_ponrjmp_ponr              ; REMOVE THIS COMMAND AND APPEND YOUR OWN JOB
        nop
        nop
        jmp *

;*************************************************************************
;
*
; ------------ END OF FFT ------------------------------------------------ *
;*************************************************************************
;
*

; SUBROUTINES FOLLOWING

;*************************************************************************
;
*
; ------------ SPECIAL RADIX-4 BUTTERFLY WITH SIMPLE TWIDDLES ------------ *
;*************************************************************************
;
*

_sr4
    move      r0,r4
    move      r1,r5
    move      r3,r7
    move      r2,r6

    move                                  y:(r5)+,d1.s
    move                x:(r0)+,d0.s      y:(r7)+,d3.s
    faddsub.s d1,d3     x:(r2),d2.s
    faddsub.s d0,d2                       y:(r4),d5.s
    faddsub.s d0,d1     x:(r1),d4.s       y:(r6)+,d7.s
    faddsub.s d5,d7     d1.s,x:(r2)+
    faddsub.s d7,d3     x:(r3),d6.s       y:(r5)-,d1.s
    faddsub.s d6,d4     d0.s,x:(r3)+      d3.s,y:(r4)+
    faddsub.s d2,d4     x:(r0)-,d0.s      d7.s,y:(r5)+n5
    faddsub.s d5,d6     d2.s,x:(r1)+      y:(r7)-,d3.s

do n1,_st2
    faddsub.s d1,d3     x:(r2),d2.s       d5.s,y:(r7)+n7
    faddsub.s d0,d2     d4.s,x:(r0)+n0    y:(r4),d5.s
    faddsub.s d0,d1     x:(r1),d4.s       y:(r6)-,d7.s
    faddsub.s d5,d7     d1.s,x:(r2)+      d6.s,y:(r6)+n6
    faddsub.s d7,d3     x:(r3),d6.s       y:(r5)-,d1.s
    faddsub.s d6,d4     d0.s,x:(r3)+      d3.s,y:(r4)+
    faddsub.s d2,d4     x:(r0)-,d0.s      d7.s,y:(r5)+n5
    faddsub.s d5,d6     d2.s,x:(r1)+      y:(r7)-,d3.s
_st2
    move                d4.s,x:(r0)       d5.s,y:(r7)
    move                                  d6.s,y:-(r6)
    rts


;*************************************************************************
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 8 of 20)

```
; ------------ NORMAL RADIX-2 BUTTERFLY ---------------------------------- *
;*****************************************************************************
;
**

_nr2
    move      r0,r4
    move      r1,r5
    move      n0,n1
    move      n0,n4
    move      n0,n5
    move                          x:(r6)+n6,d9.s     y:,d8.s
    move                                             y:(r1),d7.s

    fmpy.s    d8,d7,d3            x:(r1)+,d6.s
    fmpy.s    d9,d6,d0
    fmpy.s    d9,d7,d1                               y:(r1),d7.s
    fmpy      d8,d6,d2            fadd.s    d3,d0    x:(r0),d4.s

    fmpy      d8,d7,d3            faddsub.s d4,d0    x:(r1)+,d6.s

    do    n7,_endgrp                                 ; loop of groups
    do    r7,_bfly                                   ; butterflyloop

    fmpy  d9,d6,d0 fsub.s         d1,d2    d0.s,x:(r4)        y:(r0)+,d5.s
    fmpy  d9,d7,d1 faddsub.s      d5,d2    d4.s,x:(r5)        y:(r1),d7.s
    fmpy  d8,d6,d2 fadd.s         d3,d0    x:(r0),d4.s        d2.s,y:(r5)+
    fmpy  d8,d7,d3 faddsub.s      d4,d0    x:(r1)+,d6.s       d5.s,y:(r4)+
_bfly
    fmpy  d9,d6,d0 fsub.s         d1,d2    d0.s,x:(r4)        y:(r0)+,d5.s
    fmpy  d9,d7,d1 faddsub.s      d5,d2    d4.s,x:(r5)        y:(r1),d7.s
    fmpy  d8,d6,d2 fadd.s         d3,d0    x:(r0),d4.s        d2.s,y:(r5)+

    fmpy  d8,d7,d3 faddsub.s      d4,d0    x:(r1)+n1,d6.s     d5.s,y:(r4)+
    fmpy  d9,d6,d0 fsub.s         d1,d2    d0.s,x:(r4)        y:(r0)+,d5.s
    fmpy  d9,d7,d1 faddsub.s      d5,d2    d4.s,x:(r5)        y:(r1),d7.s
    fmpy  d8,d6,d2 fadd.s         d3,d0    x:(r0),d4.s        d2.s,y:(r5)+

    move                                   x:(r6)+n6,d9.s     y:,d8.s

    fmpy  d8,d7,d3 faddsub.s      d4,d0    x:(r1)+,d6.s       d5.s,y:(r4)+
    fmpy  d9,d6,d0 fsub.s         d1,d2    d0.s,x:(r4)        y:(r0)+n0,d5.s
    fmpy  d9,d7,d1 faddsub.s      d5,d2    d4.s,x:(r5)        y:(r1),d7.s
    fmpy  d8,d6,d2 fadd.s         d3,d0    x:(r0),d4.s        d2.s,y:(r5)+n5
    fmpy  d8,d7,d3 faddsub.s      d4,d0    x:(r1)+,d6.s       d5.s,y:(r4)+n4
_endgrp
    rts

    endm
%   MATLAB-File to generate the radix-4 twiddle factor table for the
%   fast FFT-program RMIX1.ASM .
%   By increasing the variable fftlength you can make tables for higher
%   FFT-lengths than 1024.
%
%   Karl Schwarz, Raimund Meyer       17.10.1989
%   Lehrstuhl fuer Nachrichtentechnik
%   Universitaet   Erlangen-Nuernberg
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 9 of 20)

```
    fftlength=1024          ;
    fg4=fftlength/4         ;
    fg4m1=fg4-1             ;
    x=0:fg4m1               ;
    a=bitrev(x)             ;
    a=a(2:fg4)              ;
    i=1                     ;
    c(i)=1                  ;
    s(i)=0                  ;
    i=i+1                   ;
    kon=2*pi/fftlength      ;
    for k=1:fg4m1
    c(i)=cos(kon*a(k))      ;
    s(i)=sin(kon*a(k))      ;
    i=i+1                   ;
    c(i)=cos(kon*a(k)*3)    ;
    s(i)=sin(kon*a(k)*3)    ;
    i=i+1                   ;
    c(i)=cos(kon*a(k)*2)    ;
    s(i)=sin(kon*a(k)*2)    ;
    i=i+1                   ;
end
    c=c'                    ;% real part of twiddle factor (cos)
    s=s'                    ;% imaginary part of twiddle factor (sin)
end

Optimized Complex FFT for the DSP56001/2
        page    132,60
        opt     nomd,mex,loc,nocex,mu

        include 'sincosc'
        include 'bitrevtwd56'
        include 'gen56'
        include 'cfft56'

; Latest revision - 14-Oct.-92

    reset     equ       0
    start     equ       $40
    POINTS    equ       512
    IDATA     equ       $0
    COEF      equ       $800
    ODATA     equ       $1000

    sincosc   POINTS,COEF
    gen56     POINTS,IDATA

    opt       mex
    org       p:reset
    jmp       start

    org       p:start
    movep     #0,X:$FFFE                ;0 wait states
    bitrevtwd56 POINTS,COEF
    CFFT56    IDATA,COEF,POINTS,ODATA
```

Figure A-1 Optimized Complex FFT for the DSP96002(sheet 10 of 20)

```
        nop
        nop
        jmp *
        end
;
; Sine-Cosine Table Generator for rfft56.asm and cfft56.asm
;
; Last Update 10/28/92
;
sincosc  macro   points,coef
sincosc  ident   1,2
;
;       sincosc  -      macro to generate sine and cosine coefficient
;                       lookup tables for Decimation in Time complex FFT
;                       twiddle factors. Only points/4 coefficients
;                       are generted. For real FFT another points/4
;                       coefficients with higher freq. are created.
;
;       points  -       number of points (2 - 32768, power of 2)
;       coef    -       base address of sine/cosine table
;                       positive cosine value in X memory
;                       positive sine value in Y memory
;
;    8/12/92

    pi    equ       3.141592654
;freq    equ       2.0*pi/@cvf(points*2)

;         org       x:coef-points/2
;count    set       0
;         dup       points/2
;         dc        @cos(@cvf(count)*freq)
;count    set       count+1
;         endm
;
;         org       y:coef-points/2
;count    set       0
;         dup       points/2
;         dc        -@sin(@cvf(count)*freq)
;count    set       count+1
;         endm

freq1    equ       2.0*pi/@cvf(points)

;  int i,j=1,k,tmp=0;
;  k=1<<(length-1);
;  for(i=0;i<length;i++){
;     if (integer&j) tmp=tmp|k;
;     j=j<<1;
;     k=k>>1;
         org     x:coef
count    set     0
         dup     points/4
         dc      @cos(@cvf(count)*freq1)
count    set     count+1
         endm
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 11 of 20)

```
        org     y:coef
count   set     0
        dup     points/4
        dc      @sin(@cvf(count)*freq1)
count   set     count+1
        endm

        endm                                    ;end of sincosr  macro


bitrevtwd56   macro     POINTS,COEF
bitrevtwd56   ident     1,2
;
;       bitrevtwd -     macro to sort sine and cosine coefficient
;                       lookup tables in bit reverse order for 56156
;
;       POINTS  -       number of points (2 - 32768, power of 2)
;       COEF    -       base address of sine/cosine table
;                       negative cosine (Wr) and negative sine (Wi) in X memory
;
; Wei Chen
; July-28, 1992
;


        move    #COEF,r1                        ;twiddle factor start address
        move    #0,m0                           ;bit reverse address
        move    #POINTS/8,n0                    ;sincosr use N/4 points data,
                                                ;offset for bit rev. is N/8
        move    #POINTS/4-1,n2
        move    r1,r0                           ;r1 ptr to normal order data
        move    (r1)+                           ;no swap on 1st data
        move    (r0)+n0                         ;r0 ptr to bitrev
        do      n2,_end_bit                     ;does N/4-1 points swap
        move    r1,x0
        move    r0,b
        cmp     x0,b
        jgt     _swap
        move    (r1)+                           ;no swap but update points
        move    (r0)+n0
        jmp     _nothing
_swap
        move    r1,r5
        move    r0,r4
        move    x:(r1),x0       y:(r5),y0
        move    x:(r0),a        y:(r4),b
        move    x0,x:(r0)+n0    y0,y:(r4)
        move    a,x:(r1)+       b,y:(r5)
_nothing
        nop
_end_bit
        endm                                    ;end of bitrevtwd macro
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 12 of 20)

```
; Input signal for FFT rfft56.asm and cfft56.asm
;
; Last Update 10/28/92
;
gen56    macro   POINTS,IDATA
;
;   gen56 -    macro to generate input signal for FFT test on 56001
;              2000 Hz sinewave with scaling factor POINTS in X and Y memory
;
;       POINTS -    number of points (2 - 32768, power of 2)
;       IDATA  -    base address of signal
;
srate   set    44100    ;Hz
ffreq   set    2000     ;Hz
ppi     equ    3.141592654
freq2   equ    2.0*ppi*ffreq/@cvf(srate)

        org    x:IDATA
count   set    0
        dup    POINTS
        dc     @sin(@cvf(count)*freq2)/POINTS
count   set    count+1
        endm

        org    y:IDATA
count   set    0
        dup    POINTS
        dc     @sin(@cvf(count)*freq2)/POINTS
count   set    count+1
        endm

        endm    ;end of gen56  macro




;
; 512-Point, 28174 clock cycles Non-In-Place FFT.
;
; Sept. 11 92   Version 1.0
;
CFFT56  macro   IDATA,COEF,POINTS,ODATA
CFFT56  ident   1,0
;
; 512 Point Complex Fast Fourier Transform Routine
; using the Radix 2, Decimation in Time, Cooley-Tukey FFT algorithm.
;
; This routine performs a 512 point complex FFT by taking advantages of
;   1).   internal memory access by starting first half data at location 0,
;         avoid cycle stretching;
;   2).   using N/4 complex twiddle factors based on the fact that two
;         consectivetwiddle factors in DIT FFT has a difference -j
;   3).   trivial twiddle factors (1,0) and (0,-1)  are utilized.
;
```

Figure A-1 Optimized Complex FFT for the DSP96002(sheet 13 of 20)

```
;
;    Complex input and output data
;        Real data in X memory
;        Imaginary data in Y memory
;    Normally ordered input data
;    Bit reversed output data for 1024 real input FFT
;       Coefficient lookup table
;        +Cosine values in X memory
;        -Sine values in Y memory
;
;
; Address pointers are organized as follows:
;
;   r0 = ar,ai input pointer     n0 = group offset      m0 = modulo (points)
;   r1 = br,bi input pointer     n1 = group offset      m1 = modulo (points)
;   r2 = ext. data base address  n2 = groups per pass   m2 = 256 pt fft counter
;   r3 = coef. offset each pass  n3 = coefficient base addr. m3 = linear
;   r4 = ar',ai' output pointer  n4 = group offset      m4 = modulo (points)
;   r5 = br',bi' output pointer  n5 = group offset      m5 = modulo (points)
;   r6 = wr,wi input pointer     n6 = coef. offset      m6 = bit reversed
;   r7 = not used (*)            n7 = not used (*)      m7 = not used (*)
;
;       * - r7, n7 and m7 are typically reserved for a user stack pointer.
;
; Alters Data ALU Registers
;       x1      x0      y1      y0
;       a2      a1      a0      a
;       b2      b1      b0      b
;
; Alters Address Registers
;       r0      n0      m0
;       r1      n1      m1
;       r2      n2      m2
;       r3      n3      m3
;       r4      n4      m4
;       r5      n5      m5
;       r6      n6      m6
;
; Alters Program Control Registers
;       pc      sr
;
; Uses 8 locations on System Stack
;
;


;-------------------------------------------------------------------------;
; Initialize pointers to r0->Ar,r1->Cr,r4->Bi,r5->Di, and r3->temp location  ;
; r0,r1,r4, and r5 are modular addressing with modulo N/2                    ;
;-------------------------------------------------------------------------;

        move    #IDATA,r0       ;r0 -> Ar
        move    r0,n3
        move    #ODATA,r3       ;r3 always has ODATA
        move    #COEF+1,n6      ;n6 always has COEF,(0,1) is not used
        move    #POINTS/4,n0    ;offset and butterflies per group
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 14 of 20)

```
        move   #POINTS/2-1,m0          ;modulo addressing
        move   r0,r6                   ;r6=0 flag reg. for trivial groups

        do     #3,_end_trivial         ;do three R4 passes
        move   n0,n1                   ;pointer offset
        move   n0,n4                   ;pointer offset
        move   n0,n5                   ;
        lea    (r0)+n0,r4              ;r4 -> Bi
        move   m0,m5
        lea    (r4)+n4,r1              ;r1 -> Cr
        move   m0,m1                   ;
        move   m0,m4
        lea    (r1)+n1,r5              ;r5 -> Di
;-------------------------------------------------------------------------;
;   First two passes are combined into a R4 pass without multiplication   ;
;   because Wr=1,Wi=0 in first R2 pass and Wr=0, Wi=-1 in 2nd R2 pass     ;
;                                                                         ;
; Ar'=Ar+Cr+(Br+Dr)  Br'=Ar+Cr-(Br+Dr)  Cr'=(Ar-Cr)+(Bi-Di)   Dr'=(Ar-Cr)-(Bi-Di);
; Ai'=Ai+Ci+(Bi+Di)  Bi'=Ai+Ci-(Bi+Di)  Ci'=(Ai-Ci)+(Dr-Br)   Di'=(Ai-Ci)-(Dr-Br);
;                                                                         ;
; This two passes fully ultilize internal memory by storing input data at location 0;
; For 1024-point complex FFTs, only 256-point in internal, rest of them in ;
; external, 17+2 instructions are needed for one butterfly because first and next to;
; the last instruction in the loop takes two Icycles. Other parallel move seems to;
; take two  cycles, but one of the two moves is internal, only one cycle is needed. ;
; 4.75 Icycles per R2 butterfly in the fisrt two passes.                  ;
;                                                                         ;
; For 512-point complex FFT, 17+1 instructoins are used because first instruction in
;the loop takes only one Icycle. 4.5 Icycles per R2 butterfly.            ;
;                                                                         ;
; For 256 or less point complex FFT, 17 Icycles are needed. 4.25 Icycles/bfly.
;
;-------------------------------------------------------------------------;
    move      x:(r0)+n0,a               ; a= Ar r0 -> Br
    move      x:(r1)+n1,b               ; b= Cr,r1 -> Dr

    do        n0,_twopass
    add   a,b  x:(r0)+n0,x1  y:(r5)+n5,y1  ;b=Ar+Cr,x1=Br,y1=Di,r0->Ar,r5->Ci
    subl  b,a  b,x:(r0)      y:(r4),b      ;a=Ar-Cr,save Ar+Cr temp in Ar,b=Bi
    add   y1,b a,x0          y:(r4)+n4,a   ;b=Bi+Di,x0=Ar-Cr,a=Bi again,
                                           ;save Ar-Cr in Dr,r4->Ai
    sub   y1,a b,x:(r3)x0,b           ;a=Bi-Di,store Bi+Di temp in x:ODATA,b=Ar-Cr
    sub   a,b  x:(r1),x0              ;b=Ar-Cr-(Bi-Di)=Dr',x0=Dr,r0 -> Ar
    addl  b,a  b,x:(r1)+n1   x0,b     ;a=Ar-Cr+(Bi-Di)=Cr',save Dr',b=Dr,r1->Cr
    sub   x1,b a,x:(r1)+     x0,a     ;b=Dr-Br,save Cr',a=Dr,r1->nCr
    add   x1,a x:(r0)+n0,b   b,y1     ;a=Dr+Br,b=Ar+Cr, y1=Dr-Br,r0->Br
    sub   a,b                y:(r5),y0;a=Ar+Cr-(Dr+Br)=Br',y0=Ci
    addl  b,a  b,x:(r0)+n0   y:(r4),b ;a=Ar+Cr+(Dr+Br)=Ar',save Br',r0->Ar,b=Ai
    sub   y0,b a,x:(r0)+     y:(r4),a ;b=Ai-Ci,a=Ai again, save Ar',r0->nAr
    add   y0,a x:(r3),b      b,y0     ;a=Ai+Ci,y0=Ai-Ci,b=Bi+Di, r5->Ci
    add   a,b                         ;b=Ai+Ci+(Bi+Di)=Ai'
    subl  b,a  y0,b     b,y:(r4)+n4   ;a=Ai+Ci-(Bi+Di)=Bi',b=Ai-Ci,save Ai'
    add   y1,b y0,a     a,y:(r4)+ ;b=Ai-Ci+(Dr-Br)=Ci',a=Ai-Ci,save Bi',r4->nBi
    sub   y1,a x:(r1)+n1,b   b,y:(r5)+n5;a=Ai-Ci-(Dr-Br)=Di',b=nCr,  save Ci',
    move  x:(r0)+n0,a        a,y:(r5)+ ;a=nAr,  save Di',r5->nDi
_twopass
;-------------------------------------------------------------------------;
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 15 of 20)

```
; Do rest of trivial group by 5 Icyc butterfly
;-------------------------------------------------------------------------------;
    move      n5,a                          ;n5 contains ptr to Ar already
    asr       a      n5,r1                  ;r1->Ar
    move      a,n1                          ;get offset
    move      r1,r5                         ;r5->Ai
    lea       (r1)+n1,r4                    ;r4->Bi
    move      #2,n4                         ;for pointer
    move      r4,r0                         ;r0->Br
    move      x:(r1),a  y:(r4)+,b           ;a=Ar,b=Bi,r4->nBi
    do        n1,_no_more                   ;w=(0,-1), R2 butterfly
    add       a,b  x:(r0),x0 y:(r4)-,y0     ;b=Ar+Bi=Ar',x0=Br,y0=nBi
    subl      b,a  b,x:(r1)+ y:(r5),b       ;a=Ar-Bi=Br',save Ar',b=Ai
    add       x0,b a,x:(r0)+ y:(r5),a       ;b=Ai+Br=Bi',save Br',a=Ai again
    subl      b,a  y0,b      b,y:(r4)+n4    ;a=Ai-Br=Ai',save Bi',b=nBi
    move      x:(r1),a  a,y:(r5)+           ;a=nAr,save Ai'
_no_more
    move      n0,a
    asr       a    n3,r0                    ;r0->IDATA
    asr       a    a,r2
    move      a,n0                          ;(points in a group)/4 after a radix
4 pass
    move      x:(r2)-,b      ;dec r2
    move      r2,m0
_end_trivial
    move      r0,r4                              ;output pointer
    move      n1,r1                              ;r1->Br
    move      r1,r5                              ;r5->Bi
    move      x:(r0),a                           ;a=Ar
    move      x:(r1),b                           ;b=Br
    do        n1,_extra                          ;w=(1,0)
    add       a,b                 y:(r5),y0      ;b=Ar+Br=Ar', y0=Bi
    subl      b,a  b,x:(r0)+      y:(r4),b       ;a=Ar-Br=Br',save Ar',b=Ai
    add       y0,b a,x:(r1)+      y:(r4),a       ;b=Ai+Bi=Ai',save Br',a=Ai
    sub       y0,a x:(r1),b       b,y:(r4)+      ;a=Ai-Bi=Bi',save Ai',b=nBr
    move      x:(r0),a            a,y:(r5)+      ;a=nAr,save Bi'
_extra

;-------------------------------------------------------------------------------;
;   Remaining passes are broken down to POINTS/256 sets,                        ;
;   each set has 256-point R2 FFT                                               ;
;   and runs on internal data and externalcoefficients.                         ;
;In each pass, first two groups takes advantages of trivial twiddle factors and;
;no multiplication is carried out. Remaining groups use complex twiddle factors.;
;
;                                                                               ;
; Radix 2, Decimation In Time Cooley-Tukey FFT algorithm                        ;
;              ___________                                                      ;
;             |           |       Ar'= Ar + Wr*Br - Wi*Bi                      ;
; Ar,Ai ----> |  Radix-2  |---->  Ai'= Ai + Wi*Br + Wr*Bi                      ;
; Br,Bi ----> | Butterfly |---->  Br'= Ar - Wr*Br + Wi*Bi = 2*Ar - Ar'         ;
;             |___________|       Bi'= Ai - Wi*Br - Wr*Bi = 2*Ai - Ai'         ;
;                   ^                                                         ;
;                   |                                                         ;
;                W= Wr-jWi                                                    ;
;                                                                             ;
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 16 of 20)

```
;   r0->A,r1->B,r4->A',r5->B',r6->TF,n0=offset for B pointer,
;   n2=numberof bflies in a group ;
;   n3=number of groups in a pass, m3=number of pass. r2=n3 or n3+1        ;
;-------------------------------------------------------------------------------;
    move      m2,m0             ;linear address
    move      m2,m1
    move      m2,m4
    move      m2,m5
    move      #POINTS/4,r0      ;start location of a pass
    move      #4,m3             ;4 passes in first 256-point
    move      #POINTS/16,n0     ;offset to point to Br and Bi
    move      n0,n1
    move      n0,n4
    move      n0,n5
    move      n6,r6             ;r6->COEF
    move      n0,n2             ;number of bflies in the first pass=R2 bfies/4
    lea       (r0)+n0,r1        ;r1->Br
    move      r0,r4             ;r4->Ai
    lea       (r1)-,r5          ;r5->Bi-1 for pointer reason
    jsr       _body

;-------------------------------------------------------------------------------;
; The second 256-point FFT has no any trivial twiddle factors,                  ;
; three nested loops do it                                                      ;
;-------------------------------------------------------------------------------;
    move      #256,r0           ;start location of first pass in 2nd 256
    move      #5,m3             ;5 passes in second 256-point
    move      #POINTS/8,n0      ;offset to point to Br and Bi
    move      #COEF+1,r6        ;twiddle factor pointer
    move      n0,n1
    move      n0,n4
    move      #IDATA,r4         ;r4->A' =IDATA
    move      n0,n5
    lea       (r4)+n4,r5        ;r5->B'
    lea       (r0)+n0,r1        ;r1->B
    move      x:(r5)-,a         ;r5->B'-1 for pointer reason
    jsr       _body
    jmp       _end_FFT

;-------------------------------------------------------------------------------;
; All subroutines
;-------------------------------------------------------------------------------;
_body
    move      #1,n3         ;number of groups in a pass
    move      n3,r2         ;copy of n3
    jset      #0,m3,_set_grp;first 256-point has number of group 1,3,7,15,..
    move      #2,r2
_set_grp
    do        m3,_inner_loop
    jsr       _inner_pass
    move      n0,a
    asr       a        #IDATA,r0;r0=IDATA
    move      a,n0              ;n0=offset of B
    move      r2,a
    asl       a        n0,n1
    move  a,r2                  ;r2=r2*2
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 17 of 20)

```
    asr   b         r2,n3     ;n3=number of groups in second 256
    jset #0,m3,_inner_set     ;set up start address,for 2nd 256-point r0 is already ok
    lea   (r2)-,n3            ;n3=number of groups in first 256
    move  n4,a
    asl   a         n6,r6     ;for 1st 256, TF always starts at first location
    move  a,r0                ;r0=start location of first 256-point
_inner_set
    move  n0,n4
    move  n0,n5
    move  n0,r4
    lea   (r0)+n0,r1          ;r1->B
    move  r0,r4               ;r4->A'
    lea   (r1)-,r5            ;r5->B'
_inner_loop

;---------------------------------------------------------------------------;
;  End inner loop
;---------------------------------------------------------------------------;
    move  #IDATA,r0
    move  #32,n2                        ;n2=number of groups in the next to last
                                        ;pass for 1st 256
    jset  #0,m3,_no_set                 ;set up start address of TF,for 2nd
                                        ;256-point r6 is already ok
    move  #COEF,n6                      ;now n6 -> COEF
    move  n6,r6                         ;r6=COEF
    lea   (r0)+n0,r1                    ;r1->Br
    move  r0,r4                         ;r4->Ai
_no_set
    move  #-1,r5                        ;r5->Bi
    move  #3,n0
    move  n0,n1
    move  n0,n4
    move  n0,n5
    jsr  _next_last                     ;do the pass next to last
;---------------------------------------------------------------------------;
;  End _next_last pass
;---------------------------------------------------------------------------
;
    move      #IDATA,r0                 ;r0->IDATA
    move      r3,r4                     ;r4->A',output ptr -> external memory
    jclr      #0,m3,_add_offset         ;set up output address for 2nd 256-point
    move      #256,n3
    move      r6,n6                     ;start address of TF for 2nd 256
    lea       (r3)+n3,r4
_add_offset
    lea       (r0)+,r1                  ;r1->B
    lea       (r4)-,r5                  ;r5->B'
    move      #64,n2                    ;number of blies in the last pass
    move      #2,n0
    move      n0,n1
    move      n0,n4
    move      n0,n5
    move      n6,r6                     ;r6=COEF
    jsr       _last
    rts
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 18 of 20)

```
_inner_pass
    do    n3,_end_grp                                  ;do groups in a pass
    move        x:(r5),a                               ;for pointer reason,a=something
    move        x:(r6),x0     y:(r0),b                 ;x0=Wr,b=Ai
    move        x:(r1),x1     y:(r6)+,y0               ;x1=Br,y0=Wi
    do    n0,_end_bfy1                                 ;Radix 2 DIT butterfly kernel
                                                       ;with y0=Wi,x0=Wr
    mac   -x1,y0,b  y:(r1)+,y1                         ;b=Ai-BrWi,y1=Bi, r1->nBi
    macr  x0,y1,b   a,x:(r5)+     y:(r0),a             ;b=Ai-BrWi+BiWr=Ai',
                                                       ;save prev.Br',a=Ai
    subl  b,a       x:(r0),b      b,y:(r4)             ;a=2Ai-Ai'=Bi',b=Ar,save Ai'
    mac   x1,x0,b   x:(r0)+,a     a,y:(r5)             ;b=Ar+BrWr,a=Ar,save Bi',r0->nAi
    macr  y1,y0,b   x:(r1),x1                          ;b=Ar+BrWr+BiWi=Ar',x1=nBr
    subl  b,a       b,x:(r4)+     y:(r0),b             ;a=2Ar-Ar'=Br',
                                                       ;save Ar',b=nAi,r4->nAr
_end_bfy1
    move            a,x:(r5)+n5   y:(r1)+n1,b          ;save preve. Br' inc r5 and r1
    move            x:(r4)+n4,a   y:(r0)+n0,b          ;inc r0,r4
    move            x:(r1),x1                          ;x1=nGBr
    move            x:(r5),a      y:(r0),b             ;for pointer reason,
                                                       ;a=something,b=nGAr
    do    n0,_end_bfy2                                                ;W=-j*W
    mac   -x1,x0,b  y:(r1)+,y1                         ;b=Ai-BrWr,y1=Bi,r1->nBi
    macr  -y0,y1,b  a,x:(r5)+y:(r0),a                  ;b=Ai-BrWr-BiWi=Ai',
                                                       ;save prev. Br',a=Ai
    subl  b,a       x:(r0),b      b,y:(r4)             ;a=2Ai-Ai'=Bi',b=Ar,save Ai'
    mac   -x1,y0,b  x:(r0)+,a     a,y:(r5)             ;b=Ar-BrWi,a=Ar,save Bi',r0->nAi
    macr  y1,x0,b   x:(r1),x1                          ;b=Ar-BrWi+BiWr=Ar',x1=nBr
    subl  b,a       b,x:(r4)+     y:(r0),b             ;a=2Ar-Ar'=Br',
                                                       ;save Ar',b=nAi,r4->nAr
_end_bfy2
    move            a,x:(r5)+n5   y:(r1)+n1,b          ;save preve. Br' inc r5 and r1
    move            x:(r4)+n4,a   y:(r0)+n0,b          ;inc r0,r4
_end_grp
    rts

_next_last
    move            x:(r5),a      y:(r0),b             ;a=something,b=Ai
    move            x:(r1),x1     y:(r6),y0            ;x1=Br,y0=Wi
    do    n2,_n_last                                   ;do the pass next to last,
                                                       ;internal to internal
    mac   -x1,y0,b  x:(r6)+,x0    y:(r1)+,y            ;b=Ai-BrWi,x0=Wr,y1=Bi, r1->nBi
    macr  x0,y1,b   a,x:(r5)+n5   y:(r0),a             ;b=Ai-BrWi+BiWr=Ai',
                                                       ;save prev. Br',a=Ai
    subl  b,a       x:(r0),b      b,y:(r4)             ;a=2Ai-Ai'=Bi',b=Ar,save Ai'
    mac   x1,x0,b   x:(r0)+,a     a,y:(r5)             ;b=Ar+BrWr,a=Ar,save Bi',r0->nAi
    macr  y1,y0,b   x:(r1),x1                          ;b=Ar+BrWr+BiWi=Ar',x1=nBr
    subl  b,a       b,x:(r4)+     y:(r0),b             ;a=2Ar-Ar'=Br',
                                                       ;save Ar',b=nAi,r4->nAr

    mac   -x1,y0,b                y:(r1)+n1,y1         ;b=Ai-BrWi,y1=Bi, r1->nGBi
    macr  x0,y1,b   a,x:(r5)+     y:(r0),a             ;b=Ai-BrWi+BiWr=Ai',
                                                       ;save prev. Br',a=Ai
    subl  b,a       x:(r0),b      b,y:(r4)             ;a=2Ai-Ai'=Bi',b=Ar,save Ai'
    mac   x1,x0,b   x:(r0)+n0,a   a,y:(r5)             ;b=Ar+BrWr,a=Ar,
                                                       ;save Bi',r0->nGAi
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 19 of 20)

```
    macr  y1,y0,b   x:(r1),x1              ;b=Ar+BrWr+BiWi=Ar',x1=nGBr
    subl  b,a       b,x:(r4)+n4   y:(r0),b ;a=2Ar-Ar'=Br',save Ar',
                                           ;b=nGAi,r4->nGAr

    mac   -x1,x0,b           y:(r1)+,y1    ;b=Ai-BrWr,y1=Bi,r1->nGBi
    macr  -y0,y1,b  a,x:(r5)+n5y:(r0),a    ;b=Ai-BrWr-BiWi=Ai',
                                           ;save prev. Br',a=Ai,r5->nGBi
    subl  b,a       x:(r0),b      b,y:(r4) ;a=2Ai-Ai'=Bi',b=Ar,save Ai'
    mac   -x1,y0,b  x:(r0)+,      a,y:(r5) ;b=Ar-BrWi,a=Ar,save Bi',r0->nGAi
    macr  y1,x0,b   x:(r1),x1              ;b=Ar-BrWi+BiWr=Ar',x1=nBr
    subl  b,a       b,x:(r4)+     y:(r0),b      ;a=2Ar-Ar'=Br',
                                                ;save Ar',b=nAi,r4->nGAr

    mac   -x1,x0,b                y:(r1)+n1,y1  ;b=Ai-BrWr,y1=Bi,r1->nGBi
    macr  -y0,y1,b  a,x:(r5)+     y:(r0),a      ;b=Ai-BrWi-BiWr=Ai',
                                                ;save prev. Br',a=Ai,r5->Bi
    subl  b,a       x:(r0),b      b,y:(r4)      ;a=2Ai-Ai'=Bi',b=Ar,save Ai'
    mac   -x1,y0,b  x:(r0)+n0,a   a,y:(r5)      ;b=Ar-BrWi,a=Ar,
                                                ;save Bi',r0->nGAi
    macr  y1,x0,b   x:(r1),x1     y:(r6),y0     ;b=Ar-BrWi+BiWr=Ar',
                                                ;x1=nBr,y0=nWi
    subl  b,a       b,x:(r4)+n4   y:(r0),b      ;a=2Ar-Ar'=Br',save Ar',
                                                ;b=nAi,r4->nGAr

_n_last
    move            a,x:(r5)
    rts

_last
    move  x:(r5),a  y:(r0),b              ;a=something,b=Ai
    move  x:(r1),x1 y:(r6),y0             ;x1=Br,y0=Wi
    do    n2,_end_last                    ;do last pass, internal to external
    mac   -x1,y0,b  x:(r6)+,x0    y:(r1)+n1,y1  ;b=Ai-BrWi,x0=Wr,y1=Bi, r1->nGBi
    macr  x0,y1,b   a,x:(r5)+n5   y:(r0),a ;b=Ai-BrWi+BiWr=Ai',
                                           ;save prev. Br',a=Ai
    subl  b,a       x:(r0),b      b,y:(r4) ;a=2Ai-Ai'=Bi',b=Ar,save Ai'
    mac   x1,x0,b   x:(r0)+n0,a   a,y:(r5) ;b=Ar+BrWr,a=Ar,save Bi',r0->nGAi
    macr  y1,y0,b   x:(r1),x1              ;b=Ar+BrWr+BiWi=Ar',x1=nGBr
    subl  b,a       b,x:(r4)+n4   y:(r0),b ;a=2Ar-Ar'=Br',
                                           ;save Ar',b=nGAi,r4->nGAr

    mac   -x1,x0,b                y:(r1)+n1,y1  ;b=Ai-BrWr,y1=Bi,r1->nGBi
    macr  -y0,y1,b  a,x:(r5)+n5   y:(r0),a ;b=Ai-BrWr-BiWi=Ai',
                                           ;save prev. Br',a=Ai,r5->Bi
    subl  b,a       x:(r0),b      b,y:(r4) ;a=2Ai-Ai'=Bi',b=Ar,save Ai'
    mac   -x1,y0,b  x:(r0)+n0,a   a,y:(r5) ;b=Ar-BrWi,a=Ar,save Bi',r0->nGAi
    macr  y1,x0,b   x:(r1),x1     y:(r6),y0;b=Ar-BrWi+BiWr=Ar',
                                           ;x1=nBr,y0=nWi
    subl  b,a       b,x:(r4)+n4   y:(r0),b ;a=2Ar-Ar'=Br',
                                           ;save Ar',b=nAi,r4->nGAr
_end_last
    move            a,x:(r5)
    rts
_end_FFT
    endm
```

Figure A-1 Optimized Complex FFT for the DSP96002 (sheet 20 of 20)