;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

;\* FUNCTION NAME: FIR \*

;\* \*

;\* Regs Modified : A0,A3,A4,A5,B0,B4,B5,B6,B7,SP,B31 \*

;\* Regs Used : A0,A3,A4,A5,B0,B3,B4,B5,B6,B7,SP,B31 \*

;\* Local Frame Size : 0 Args + 24 Auto + 0 Save = 24 byte \*

;\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**\_FIR:**

;\*\* --------------------------------------------------------------------------\*

.dwcfi cfa\_offset, 0

.dwcfi save\_reg\_to\_reg, 228, 19

**SUB** .D2 SP,24,SP ; |136|

.dwcfi cfa\_offset, 24

$C$DW$46 .dwtag DW\_TAG\_variable, DW\_AT\_name("index")

.dwattr $C$DW$46, DW\_AT\_TI\_symbol\_name("\_index")

.dwattr $C$DW$46, DW\_AT\_type(\*$C$DW$T$42)

.dwattr $C$DW$46, DW\_AT\_location[DW\_OP\_breg31 4]

$C$DW$47 .dwtag DW\_TAG\_variable, DW\_AT\_name("xBuff")

.dwattr $C$DW$47, DW\_AT\_TI\_symbol\_name("\_xBuff")

.dwattr $C$DW$47, DW\_AT\_type(\*$C$DW$T$45)

.dwattr $C$DW$47, DW\_AT\_location[DW\_OP\_breg31 8]

$C$DW$48 .dwtag DW\_TAG\_variable, DW\_AT\_name("yVal")

.dwattr $C$DW$48, DW\_AT\_TI\_symbol\_name("\_yVal")

.dwattr $C$DW$48, DW\_AT\_type(\*$C$DW$T$17)

.dwattr $C$DW$48, DW\_AT\_location[DW\_OP\_breg31 16]

$C$DW$49 .dwtag DW\_TAG\_variable, DW\_AT\_name("i")

.dwattr $C$DW$49, DW\_AT\_TI\_symbol\_name("\_i")

.dwattr $C$DW$49, DW\_AT\_type(\*$C$DW$T$10)

.dwattr $C$DW$49, DW\_AT\_location[DW\_OP\_breg31 24]

MV .L1X B4,A3 ; |136|

|| STB .D2T1 A4,\*+SP(4) ; |136|

STW .D2T1 A3,\*+SP(8) ; |136|

.dwpsn file "../lab\_files/lab2.c",line 137,column 12,is\_stmt

**ZERO** .L1 **A5:**A4 ; |137|

STDW .D2T1 **A5:**A4,\*+SP(16) ; |137|

.dwpsn file "../lab\_files/lab2.c",line 139,column 9,is\_stmt

**ZERO** .L2 B4 ; |139|

STW .D2T2 B4,\*+SP(24) ; |139|

.dwpsn file "../lab\_files/lab2.c",line 139,column 14,is\_stmt

MVK .S2 32,B5 ; |139|

CMPLT .L2 B4,B5,B0 ; |139|

[!B0] BNOP .S1 $C$L11,5 ; |139|

; BRANCHCC OCCURS {$C$L11} ; |139|

;\*----------------------------------------------------------------------------\*

;\* SOFTWARE PIPELINE INFORMATION

;\* Disqualified loop: Software pipelining disabled

;\*----------------------------------------------------------------------------\*

**$C$L10:**

**$C$DW$L$\_FIR$2$B:**

.dwpsn file "../lab\_files/lab2.c",line 141,column 9,is\_stmt

LDB .D2T2 \*+SP(4),B5 ; |141|

LDW .D2T2 \*+SP(8),B6 ; |141|

MVKL .S2 \_filterCoeff,B31

MVKH .S2 \_filterCoeff,B31

LDW .D2T2 \*+B31[B4],B4 ; |141|

**NOP** 1

LDH .D2T2 \*+B6[B5],B6 ; |141|

**NOP** 4

INTSP .L2 B6,B5 ; |141|

LDDW .D2T2 \*+SP(16),**B7:**B6 ; |141|

**NOP** 2

MPYSP .M2 B4,B5,B4 ; |141|

**NOP** 3

SPDP .S2 B4,**B5:**B4 ; |141|

**NOP** 1

ADDDP .L2 **B5:**B4,**B7:**B6,**B5:**B4 ; |141|

**NOP** 6

STDW .D2T2 **B5:**B4,\*+SP(16) ; |141|

.dwpsn file "../lab\_files/lab2.c",line 139,column 29,is\_stmt

LDW .D2T2 \*+SP(24),B5 ; |139|

LDB .D2T2 \*+SP(4),B4 ; |139|

**NOP** 4

**ADD** .L2 1,B5,B4 ; |139|

|| SUB .S2 B4,1,B5 ; |139|

STW .D2T2 B4,\*+SP(24) ; |139|

|| EXTU .S2 B5,27,27,B4 ; |139|

STB .D2T2 B4,\*+SP(4) ; |139|

.dwpsn file "../lab\_files/lab2.c",line 139,column 14,is\_stmt

LDW .D2T2 \*+SP(24),B4 ; |139|

MVK .S1 32,A3 ; |139|

**NOP** 3

CMPLT .L1X B4,A3,A0 ; |139|

[ A0] BNOP .S1 $C$L10,5 ; |139|

; BRANCHCC OCCURS {$C$L10} ; |139|

**$C$DW$L$\_FIR$2$E:**

;\*\* --------------------------------------------------------------------------\*

**$C$L11:**

.dwpsn file "../lab\_files/lab2.c",line 143,column 5,is\_stmt

LDDW .D2T1 \*+SP(16),**A5:**A4 ; |143|

**NOP** 4

DPSP .L1 **A5:**A4,A4 ; |143|

**NOP** 3

.dwpsn file "../lab\_files/lab2.c",line 144,column 1,is\_stmt

ADDK .S2 24,SP ; |144|

.dwcfi cfa\_offset, 0

.dwcfi cfa\_offset, 0

$C$DW$50 .dwtag DW\_TAG\_TI\_branch

.dwattr $C$DW$50, DW\_AT\_low\_pc(0x00)

.dwattr $C$DW$50, DW\_AT\_TI\_return

RETNOP .S2 B3,5 ; |144|

; BRANCH OCCURS {B3} ; |144|

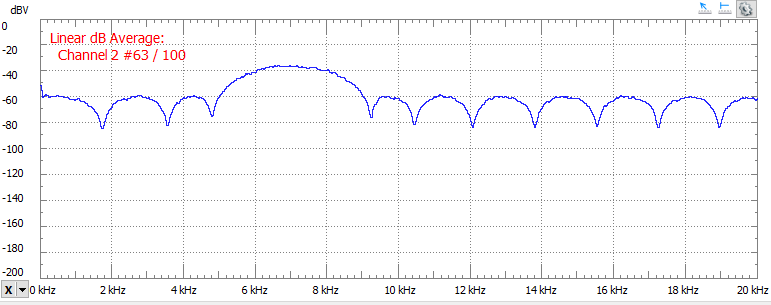


Figure 3 Ladder FIR FFT

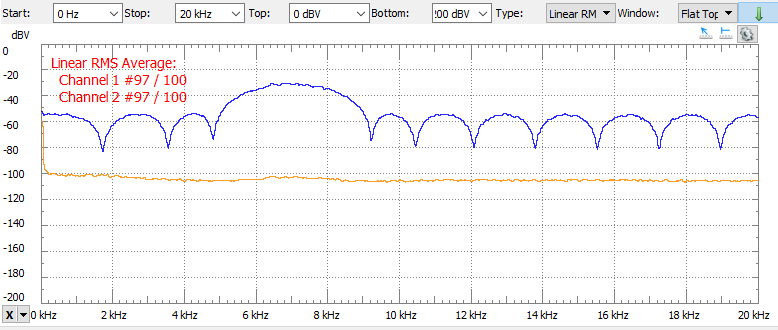
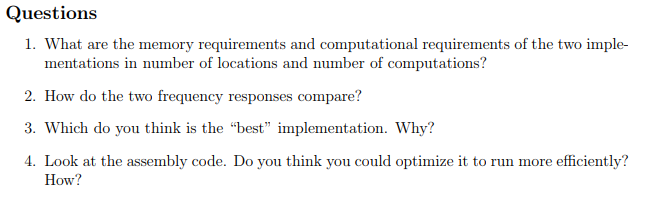
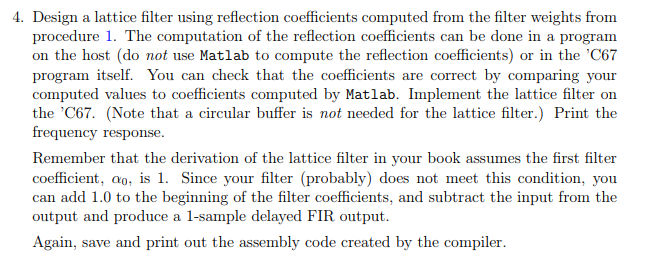


Figure 4 Lattice FIR FFT





**int** **main**(**void**)

{

int16\_t xBuff[FILTER\_SIZE] = { 0 };

int16\_t newVal=0;

**float** gDelayed[FILTER\_SIZE] = {0};

**float** g[FILTER\_SIZE]={0};

// int8\_t xIndex = 0x0;

**float** yVal = 0; // Sine Sample

**float** k,gDel;

int16\_t temp;

int8\_t i=0;

**#ifdef** LATTICE

**float** alpha[FILTER\_SIZE], beta[FILTER\_SIZE];

**float** kVal[FILTER\_SIZE] = { 0 };

memcpy(alpha,filterCoeff,**sizeof**(filterCoeff));

ComputeK(alpha, FILTER\_SIZE-1,beta, kVal);

**#endif**

Init();

// Infinite loop: Each loop reads/writes one sample to the left and right channels.

**while** (1)

{

//write out y value

**#ifndef** LATTICE

xBuff[x0Index] = newVal;

yVal = FIR(x0Index, xBuff);

x0Index = (x0Index + 0x1) & 0x1F;

**#else**

g[0]=newVal;

yVal=newVal;

//

**for**(i=1; i<FILTER\_SIZE/2; i++){

k=kVal[i];

gDel=gDelayed[i-1];

g[i]=k\*yVal+gDel;

yVal=yVal+k\*gDel;

}

**#endif**

**while** (!CHKBIT(MCASP->SRCTL12, RRDY)) { }

temp = MCASP->RBUF12; // read next value from left channel

**while** (!CHKBIT(MCASP->SRCTL11, XRDY)) { }

MCASP->XBUF11 = 0;

// write output to left channel

**#ifdef** LATTICE

**for**(i=FILTER\_SIZE/2; i<FILTER\_SIZE; i++){

k=kVal[i];

gDel=gDelayed[i-1];

g[i]=k\*yVal+gDel;

yVal=yVal+k\*gDel;

}

memcpy(gDelayed,g,**sizeof**(g));

yVal=yVal-newVal;

**#endif**

**while** (!CHKBIT(MCASP->SRCTL11, XRDY)) { }

MCASP->XBUF11 = (int16\_t) yVal; //(int16\_t) xBuff[xIndex];

**while** (!CHKBIT(MCASP->SRCTL12, RRDY)) { }

newVal=MCASP->RBUF12;

// write 0 to right channel

}

}

**void** **ComputeK**(**float** \*alpha, **int** m, **float** \*beta, **float** \*k)

{

**int** i;

k[m] = alpha[m];

**if** (m == 0)

{

**return**;

}

//reverse coeff for beta

**for**(i=0;i<=m;i++){

beta[i]=alpha[m-i];

}

// beta=memcpy(&beta[0],&alpha[0],FILTER\_SIZE\*sizeof(float));

//computes z^i coefficients of A\_(m-1)

**for** (i = 0; i < m; i++)

{

alpha[i] = (alpha[i] - k[m] \* beta[i])

/ (1 - k[m] \* k[m]);

}

//compute K\_(m-1)

ComputeK(alpha, m - 1,beta,k);

}