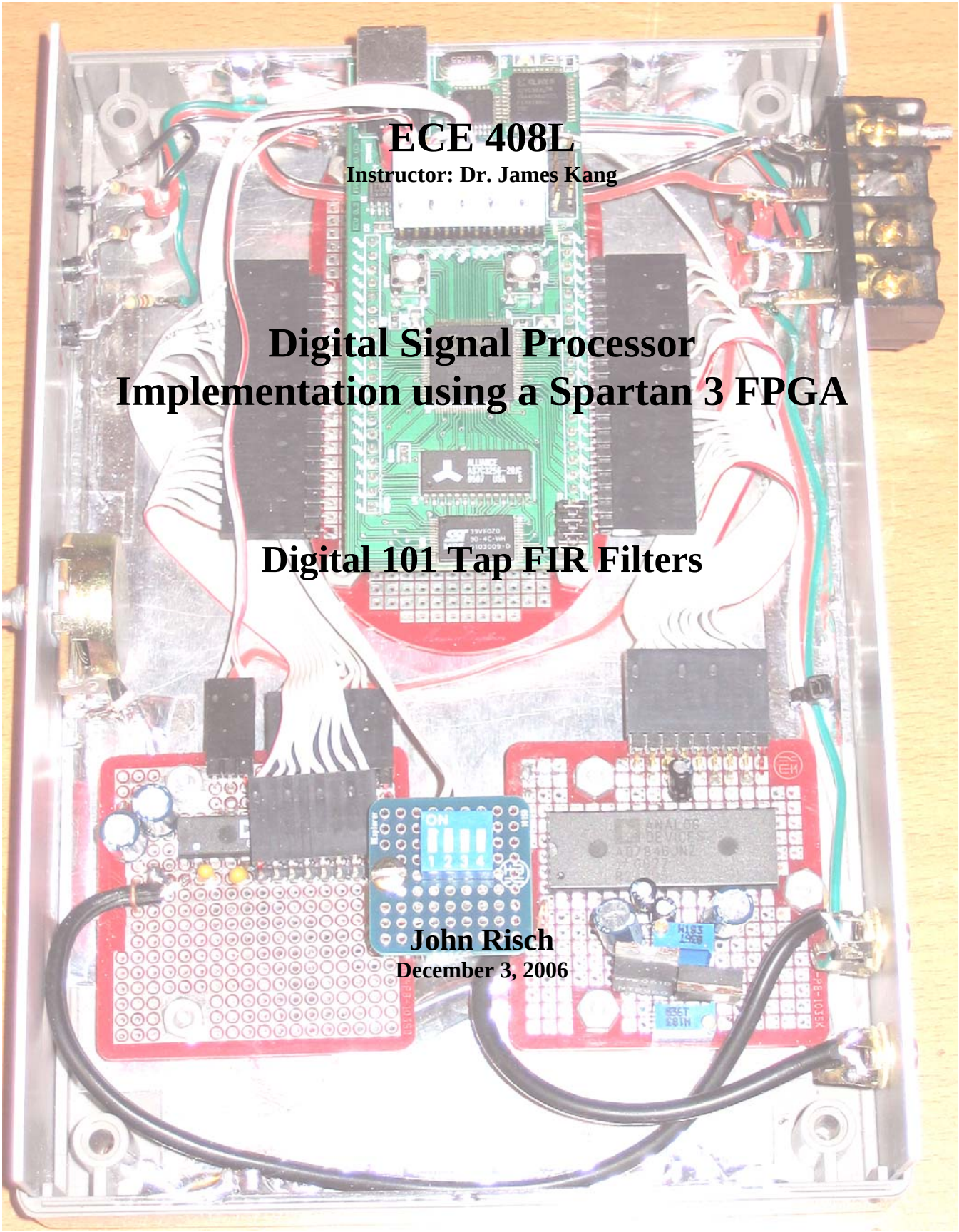
ECE 408L

Instructor: Dr. James Kang

Digital Signal Processor Implementation using a Spartan 3 FPGA

Digital 101 Tap FIR Filters



John Risch

December 3, 2006

Digital Filtering on an FPGA

John Risch
jdrisch@hotmail.com

Abstract

The objectives of this project are to implement a Digital Signal Processing System using a Xilinx Spartan 3 FPGA board. Then Implement several Digital Filters on the created Digital Signal Processor System.

1. Introduction

ECE 408L is the follow-up laboratory to ECE 408 Digital Filter Systems. The purpose of this laboratory is to explore implementations of different digital filtering techniques in a more hands-on environment. Dr. Kang has allowed several options, one of which is designing a DSP board via an FPGA. In such a design an ADC and DAC are the only external components required in addition to the FPGA chip. The Digital Signal Processing will all be handled within the FPGA hardware, which is created first as software code. Verilog HDL will be the programming language used in this project. The digital designing in Verilog will be responsible for interfacing, syncing, storing, and processing a single channel 16-bit analog signal.

2. Hardware Design

Dr. Kang has designated the use of a 200kSA/s 16-bit ADC, the AD976A, and a greater then 100kHz 16-bit DAC IC, the AD7846, to be used in the design. Both of these chips are 16-bit parallel configuration in a DIP package.

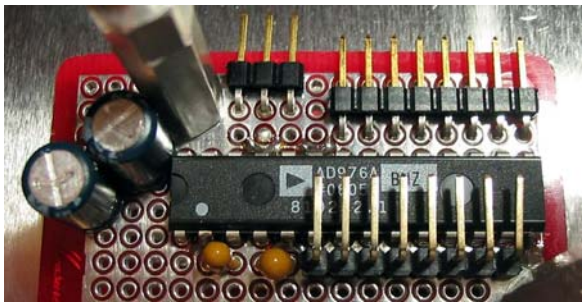


Figure 2.1

Figure 2.1 shows the AD976A ADC IC as it has been mounted to a 2" by 2" prototyping board.

The AD976A has a few different modes of operation, but after studying the datasheet and timing diagrams, the purposes of this laboratory a simple "always on" mode will be used. With the chip select pin tied low to enable the device as long as it is powered, along with a few other unneeded inputs disabled, the chip has been configured to take a single input in to initiate a conversion, and pass a single "conversion ready" signal output to tell the FPGA that the 16-bit data outputs contain valid data. This simple configuration will ease the design of the FPGA's ADC interface module.

Figure 2.3, shown on the next page, shows a schematic of the pinout connections for the AD976A IC. This IC is quite simple to use since it only needs a few external components to operate. A couple reference capacitors and a 200Ω protective resistor were the only components required by the datasheet. The AD976A also provides separate analog and digital power supply pins, which enable the ability to provide separate power capacitors and power supply voltage wires. This feature helps reduce the analog and digital crosstalk and improve the SNR.

Another reminder worth mentioning is that the ICs used in this design are 5V TTL devices and the Spartan 3 FPGA has 3.3LVTTL I/O ports. This means that the FPGA outputs will offer a "high" output at 3.3V which is acceptable to drive 5V TTL devices like the DAC. But 5V TTL outputs from the ADC IC should not be run directly into the FPGA, the extra current may damage the FPGA. In cases where 5V TTL signals are to be inputted into the FPGA simple resistive voltage dividers work great. There is typically no need for active signal conditioning into the FPGA, since the FPGA's inputs are CMOS devices which require little current to trigger.

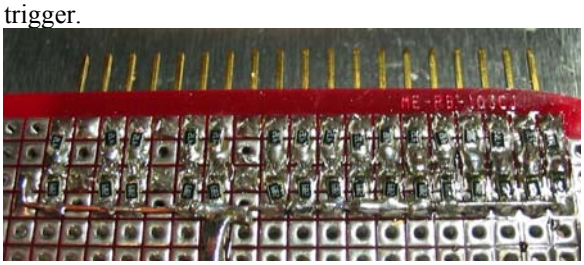


Figure 2.2 shows the 16 voltage dividers used on the ADC interface into the FPGA.
(0805 SMT resistors)

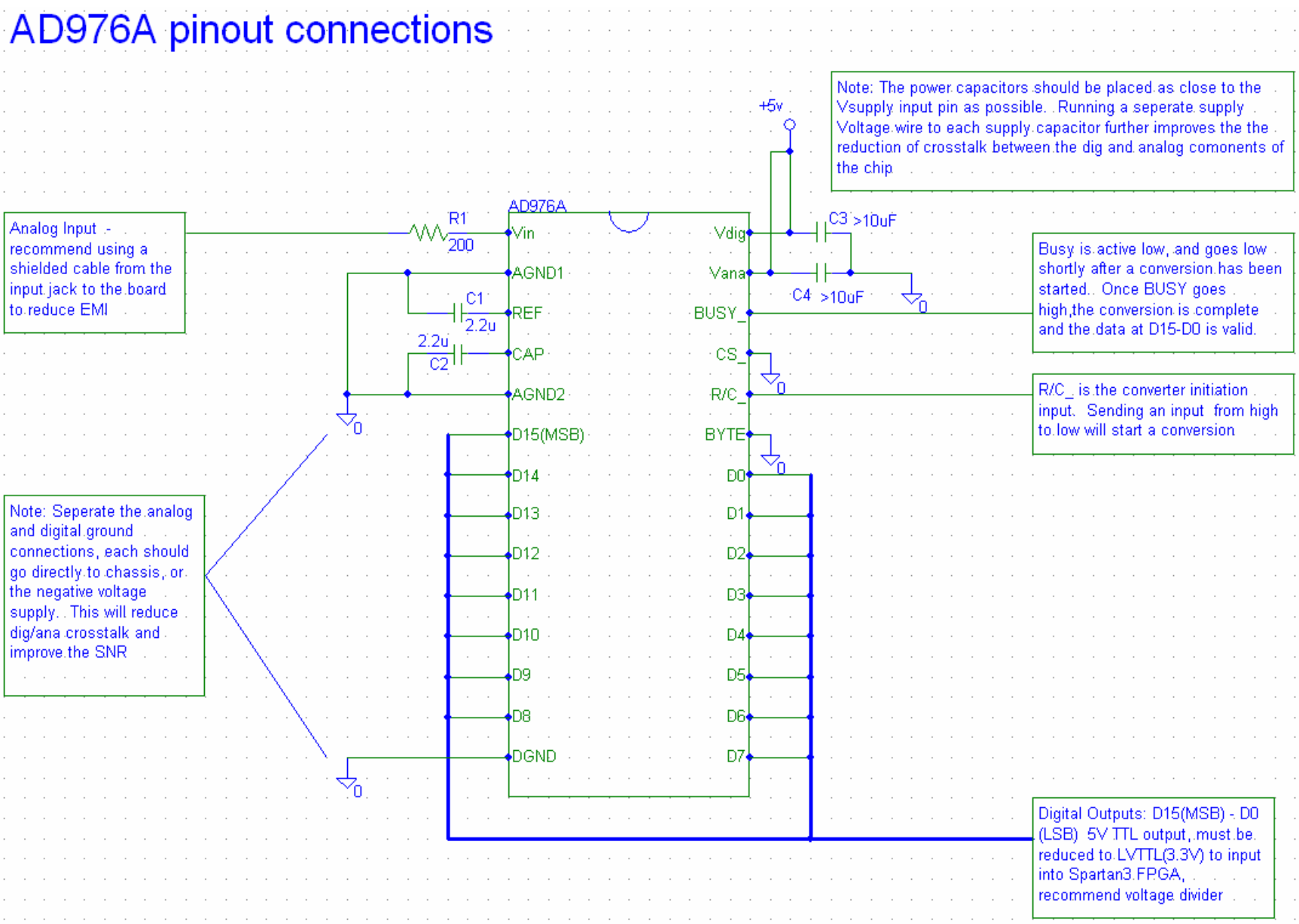

AD976A pinout connections
Note: The power capacitors should be placed as close to the Vsupply input pin as possible. Running a seperate supply Voltage wire to each supply capacitor further improves the the reduction of crosstalk between the dig and analog comonents of the chip
+5v
Analog Input - recommend using a shielded cable from the input jack to the board to reduce EMI
R1
200
AD976A
Vin
AGND1
C1
2.2u
REF
2.2u
C2
CAP
AGND2
D15(MSB)
D14
D13
D12
D11
D10
D9
D8
DGND
Vdig
Vana
BUSY_
CS_
R/C_
BYTE
D0
D1
D2
D3
D4
D5
D6
D7
C3 >10uF
C4 >10uF
Busy is active low, and goes low shortly after a conversion has been started. Once BUSY goes high,the conversion is complete and the data at D15-D0 is valid.
R/C_ is the converter initiation input. Sending an input from high to low will start a conversion
Note: Seperate the analog and digital ground connections, each should go directly to chassis, or the negative voltage supply. This will reduce dig/ana crosstalk and improve the SNR
Digital Outputs: D15(MSB) - D0 (LSB) 5V TTL output, must be reduced to LVTTL(3.3V) to input into Spartan3 FPGA, recommend voltage divider

Figure 2.3

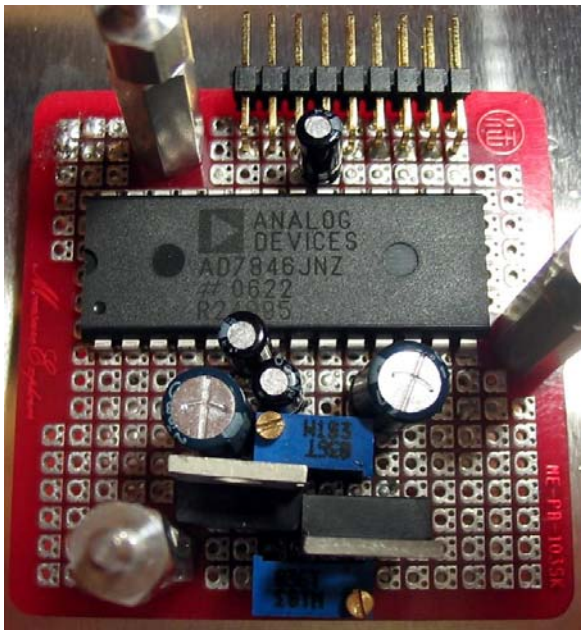


Figure 2.4

Figure 2.4 shows the AD7846 DAC IC as it was mounted to a 2" by 2" prototyping board. Also appearing below the IC are a pair of adjustable +/- 5V regulated power supplies, which provide the needed low-ripple +/- 5V reference voltages for the DAC IC.

Shown in figure 2.6 on the next page is a schematic of the pinout connections for the AD7846 IC. A few additional features were implemented to improve the functionality of the DAC. A dedicated pair of voltage regulators were provided for this IC to offer low ripple reference voltage sources. The recommended regulators would be the LM7805 (+5V) and the LM7905(-5V) regulators, although an NTE equivalent the NTE960 was used instead of the LM7905. Each regulator is fed respectively from either a +15V or -15V source provided by an external power supply to the system.

EMI reduction

In any system where analog signals are used, EMI is always a major consideration. In the case of a DSP there is need for concern when considering the high frequency digital signals that travel along the digital buses, and how they may provide unwanted HF noise to nearby analog signal cables. Here is a list of EMI reducing tips when designing the layout of the analog and digital portions of a system.

EMI Shielding tips:

- Routing the digital signals away from the analog sections of the circuit, (always physically separate analog and digital circuits as much as possible)
- Providing separate power supplies and power capacitors for digital and analog circuitry,
- Routing all analog signals with shielded cable.
- Use aluminum plating as the grounding for each circuit board, using a long ground cable between circuit boards allows for noise to develop along the grounding cable. A noisy ground dictates the inherent noise on the connected circuits.
- Shielding plastic enclosures with aluminum tape, or other conductive material. Can reduce external noise from outside sources, as well as keeping noise generated by your circuit from effecting outside systems.
- Note: when shielding digital signals be careful to consider the capacitance created between the cable and the nearby ground, shielding high bandwidth cables to closely can harm high bandwidth performance.

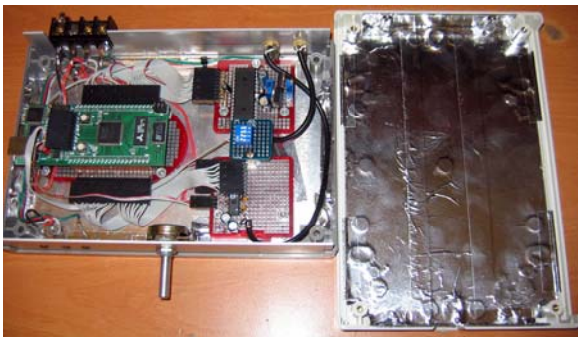


Figure 2.5

Figure 2.5 shows the DSP board's layout. Notice that the digital bus cables leave and travel to the left of the ADC and DAC chips, while the analog signals travel to the right side in shielded black cables.

The DAC Verilog module can be viewed in the Verilog Code section at the end of this report.

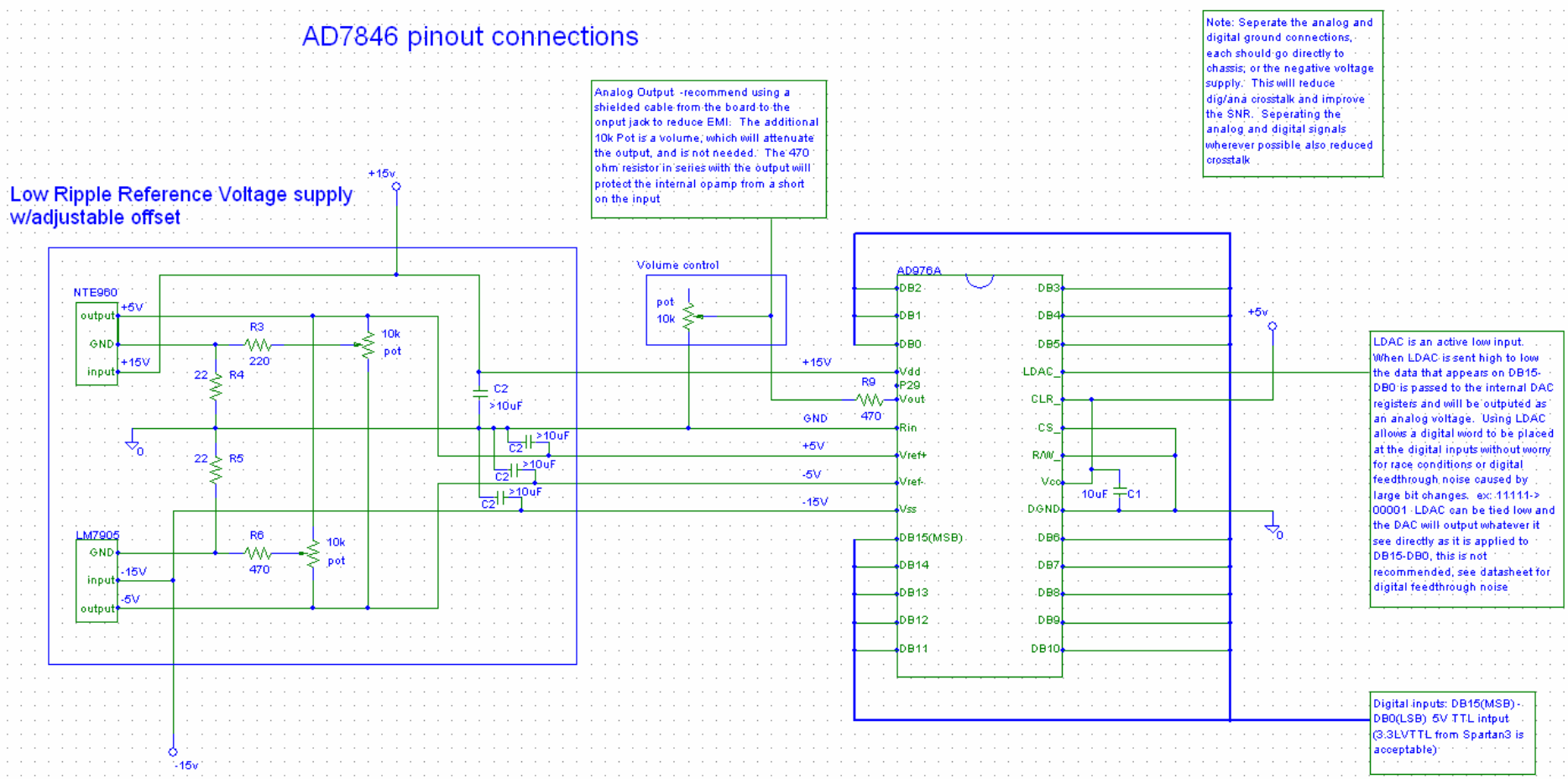

AD7846 pinout connections
Low Ripple Reference Voltage supply w/adjustable offset
Analog Output -recommend using a shielded cable from the board to the onput jack to reduce EMI. The additional 10k Pot is a volume; which will attenuate the output, and is not needed. The 470 ohm resistor in series with the output will protect the internal opamp from a short on the input
Note: Seperate the analog and digital ground connections, each should go directly to chassis; or the negative voltage supply. This will reduce dig/ana crosstalk and improve the SNR. Seperating the analog and digital signals wherever possible also reduced crosstalk
+15v
NTE960
output
GND
input
+5V
+15V
R3
220
10k
pot
22
R4
22
R5
0
C2
>10uF
LM7905
GND
input
output
-15V
-5V
R6
470
10k
pot
-15v
Volume control
pot
10k
AD976A
DB2
DB1
DB0
Vdd
P29
Vout
Rin
Vref+
Vref-
Vss
DB15(MSB)
DB14
DB13
DB12
DB11
DB3
DB4
DB5
LDAC_
CLR_
CS_
R/W_
Vcc
DGND
DB6
DB7
DB8
DB9
DB10
R9
470
GND
+5V
-5V
-15V
10uF
C1
LDAC is an active low input. When LDAC is sent high to low the data that appears on DB15-DB0 is passed to the internal DAC registers and will be outputed as an analog voltage. Using LDAC allows a digital word to be placed at the digital inputs without worry for race conditions or digital feedthrough noise caused by large bit changes. ex: 11111-> 00001 LDAC can be tied low and the DAC will output whatever it see directly as it is applied to DB15-DB0, this is not recommended, see datasheet for digital feedthrough noise
Digital inputs: DB15(MSB) - DB0(LSB) 5V TTL intput (3.3LVTTL from Spartan3 is acceptable)

Figure 2.6

3. Software Design

The software design for this project was done using Xilinx ISE 8.2i using Verilog HDL. Debugging outside of syntax errors and other error detection features of Xilinx ISE was done using ModelSim and Verilog test fixture modules. Test fixtures were written to simulate inputs into individual modules to ensure their functionality under predictable test stimuli and verify the module's expected outputs.

The first software task was to implement the ADC and DAC interface modules. Then verify through-put of the analog signal from ADC to FPGA and out of the DAC. I prefer a modular design approach when I write Verilog coding. By reducing individual tasks to separate modules, the debugging of each modules becomes much more feasible. If code is written in a single module, a single syntax error can resort to several cascaded errors, and this can make debugging beyond tedious and impractical.

DSP module, Top Module

The top module was named "DSP", I consider the top module to be the control and routing module. If there is a master clock or counter it is created or starts here. All other modules are called in this module and their signals routed to one another through this module. The next modules created were "ADC" and "DAC". ADC and DAC must be synchronized to execute the recording of the sampled inputs, the outputting of the processed signal, and of course allow time to process the signal. It was later decided to implement a master counter in the DSP module to ensure that each module would run based on a single timing signal.

ADC module

ADC needed to control the AD976A IC and record the samples collected by the ADC. The R/C_ signal was the initiate a conversion input for the ADC, and was the first output to consider. Once the ADC started a conversion it was a particular wait until a conversion was completed, then the BUSY_ output of the ADC would go high, and the ADC module would know that the 16-bit data input would be valid, and to pass that samples signal into the DSP module.

DAC module

DAC was required to feed the 16-bit output signal to the DAC and control the LDAC input on the DAC chip. The purpose of the LDAC input on the DAC was to reduce digital feedthrough noise caused when a large number of bits changed at once. For example, since the DAC operated on a signed binary offset numbering system, there was a "zero" voltage crossover from just below zero volts to zero volts that appeared as this:

0111 1111 1111 1111 → 1000 0000 0000 0000

This change of polarity for all 16 bits causes a voltage surge that would affect the output if the DAC was to immediately output the 16-bit data while LDAC was tied low. By controlling LDAC to go low a few moments after the 16-bit data has changed, the large bit change has very little effect on the output since the output latches of the DAC are not updated until LDAC is signaled low. Further details about the use of LDAC can be read in the datasheet for the AD7846 chip.

The completion of the throughput signal was achieved, and the following description will entail the remaining modules of the FIR system. The throughput implementation is simply the FIR system with the output of the ADC module being fed directly into the DAC module.

SignFixPt19 and UnSignFixPt19 module

A problem with Digital systems is that signal errors occur due to quantization factors during sampling and in the processing of the digital signal. To improve on the quantization error that resulted from the processing quantization, the 16-bit sampled data was converted into a 19-bit signed magnitude word with 12 fractional bits. 12 fractional bits was chosen because it was just more precise then the 16-bit signed binary input offered by the ADC, and thus there would not be a significant quantization loss during the conversion. And the 19-bit word would be capable of holding more information during the processing. Two modules were created to convert the ADC input to 19-bit number and then the 19-bit number back into a 16-bit signed binary offset number for the DAC. These were called: SignFixPt19 and UnSignFixPt19.

An example of the conversion using the maximum value input from the ADC would look like this:
10V→ADC→0111_1111_1111_1111→SignFixPt19 →0_001010_0000_0000_0000 The first bit of the 19-bit number is a sign bit, then 6 integer bits, and then 12 fractional bits allowing a calculation quantization of 244.24 μV during processing. The change to this 19-bit number was necessary to overcome the chances of data overflow during processing. Using only a 16-bit signed number during processing would mean immediate quantization loss during any addition or multiplication, since the largest number in the original 16-bit signed word was "10". 6 integer bits allows a maximum value range of "+/-63"

Once the 16-bit data from the ADC module was passed through SignFixPt19 and was converted to a 19-bit value, the data was passed into the FIR module.

FIR module

The Fir module was responsible for handle all of the data processing. Since this was a FIR filtering system, this would entail the mathematics of a difference equation.

The final version of the FIR module encompassed 4 101 Tap FIR filters: a Low-Pass, High-Pass, Band-Pass, and Band-Stop filter. Each requiring 101 coefficients and 100 19-bit storage registers for previous samples. The best implementation I came up with involved line by line execution of 203 consecutive calls for parallel math functions and register transfers which implemented a MAC, or Multiply and accumulate function. The reward would be the use of only a single 19 by 19 bit multiplier and a single 19 by 19 bit adder in hardware for the FIR calculations.

Tap Module

Another hardware conserving part of the design was to implement the "Tap" module. This module was designed to contain the 404 coefficients needed for the 4 different filter types. This module would input a mode, 0-3, and the coefficient suffix, and output the desired coefficient to the FIR module. This look-up-table design reduced the FIR module implementation to one hardware implementation for all 4 filters. A quantization improvement was also done during this part of the design to further improve quantization error. The coefficients of a FIR system are always less then 1, thus altering the coefficients to be 19-bit signed magnitude representation with 17 fractional bits greatly improved the quantization losses in the coefficient values.

Mul module

A multiplication module was created to handle the 19-bit numbers using "if" conditions and 18-bit unsigned multiplication. The two multiplied quantities were the system variable, a 19-bit signed magnitude number with 12 fractional bits, and the FIR coefficient from the Tap module, a 19-bit signed magnitude number with 17 fractional bits. The output was a 19-bit signed magnitude with 12 fractional bits.

Add module

This module was created to handle the 19-bit signed magnitude summing using "if" conditions for the sign bit and unsigned 18-bit addition.

4. Filter Design

The first attempted filter designs were IIR filters. One of the IIR filter implementations included a Second Order Section (SOS) module which was designed to import SOS coefficients with pre-quantized coefficients straight out of Matlab. This module was desired to be implemented in a way that would make is cascadable to implement high order IIR filters using SOS in cascade. Unfortunately a stable IIR filter was never achieved and the IIR filter designs had to sit back seat to ensure that the inherently more stable FIR filters would be completed for the project.

In digital filtering FIR filters are typically used because they are simple to implement and far more stable then there IIR relatives. IIR filters are recursive and coupled with the quantization error of a digital system, resort in cascaded quantization that can grow and cause an unstable stable system. IIR filters can actually become oscillators because of this trait. For this reason non-recursive filters or FIR filters which pass a sample out of the system after the last order tap, remain far more stable. Wit the power of FPGAs growing the performance advantages of a lower order IIR filter will be overtaken by extremely high order FIR filters which can naturally be more stable. It is not to say that the extreme power of newer FPGAs cannot store and process 64-bit numbers or larger and minimize quantization error to the point that implementing IIR filters in an FPGA system remains more practical. The answer to this question is beyond the scope of this course.

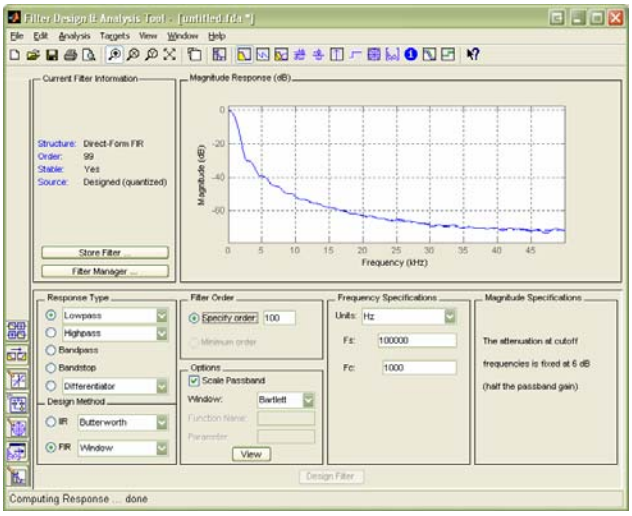


Figure 4.1 shows a view of the fdatool application window, this design instance is of the 101 Tap FIR Low-Pass Filter using a Bartlett window

FIR system

The final design of the DSP system implemented 4 101 Tap FIR filters on a single FPGA design. The 101 coefficients used were directly imported from Matlab with pre-quantized accuracy. Using Matlab's powerful fdatool to design the filters was extremely useful. Matlab's fdatool includes a feature which calculated coefficients and the respective filter design using a quantized coefficients setup to represent the system restrictions inputted by the user. In my case I was using 19-bit signed magnitude representation with 17 fractional bits, so inputting these specifications into the quantization rules, allowed fdatool to output the expected filter frequency response using the quantized coefficients. All of fdatool's design views including the magnitude and phase response, the impulse response, the step response, and even a final output prediction magnitude response all reflected the quantized coefficients. This useful feature removed some of the trail an error that would have been caused by not knowing how the quantized coefficients would affect the system's frequency response.

The final 4 101 Tap FIR filters were using window methods. The 4 selected 101 Tap filters used the following specifications: all fs: 100kHz sampling rate

- Low-Pass fc: 1khz
  - Window: Bartlett
- High-Pass fc: 3kHz
  - Window: Bartlett
- Band-Pass f1: 2.5khz f2: 3.0kHz
  - Window: Chebyshev
  - Sidelobe Attenuation: 25dB
- Band-Stop f1: 2.5kHz f2: 4.0kHz
  - Window: Kaiser
  - Beta: 17

5. Results

Low-Pass Filter results

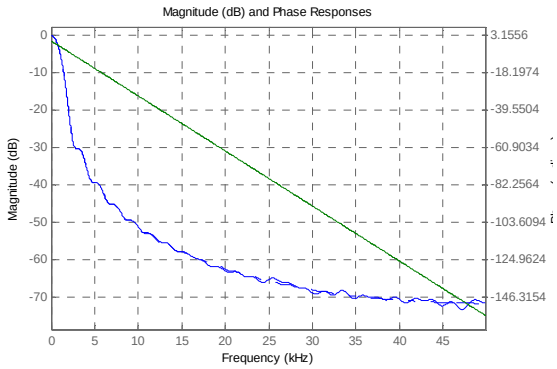


Figure 5.1 shows the magnitude and phase frequency response of the LP filter

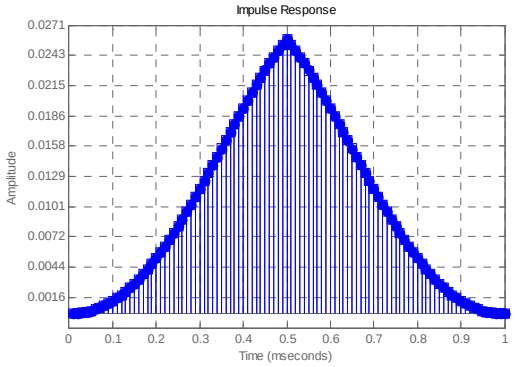


Figure 5.2 shows the impulse response of the LP filter

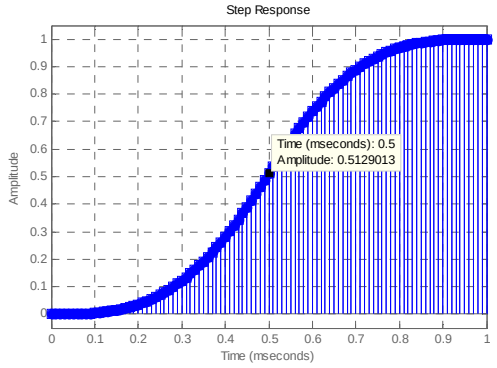


Figure 5.3 shows the step response of the LP filter

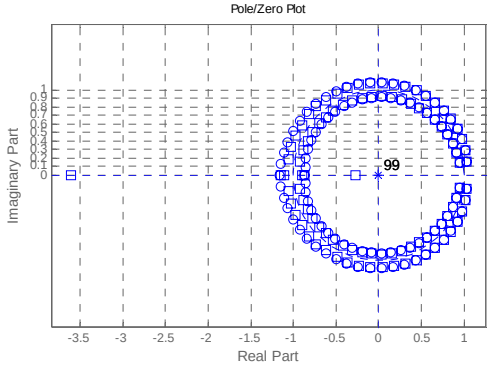


Figure 5.4 shows the Pole/Zero plot of the LP filter

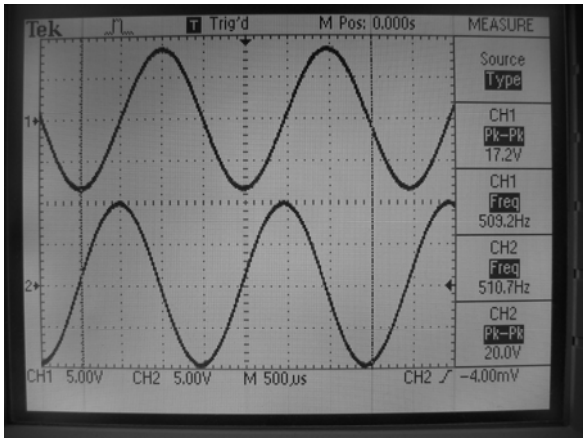


Figure 5.5

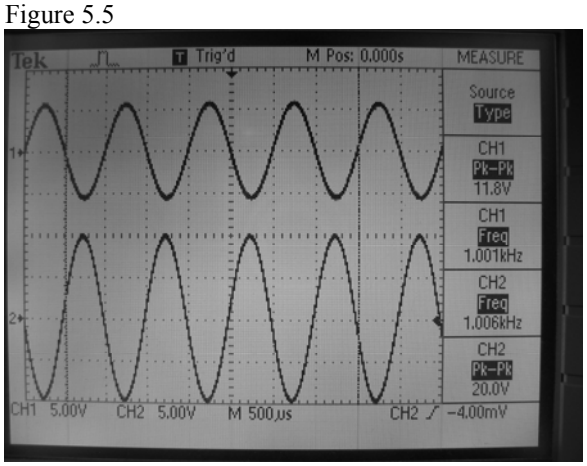


Figure 5.6

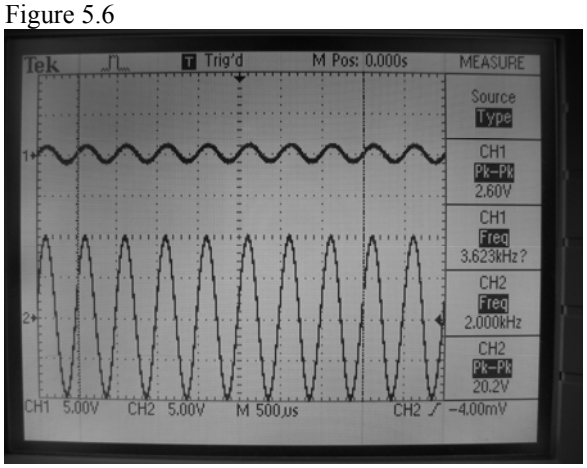


Figure 5.7

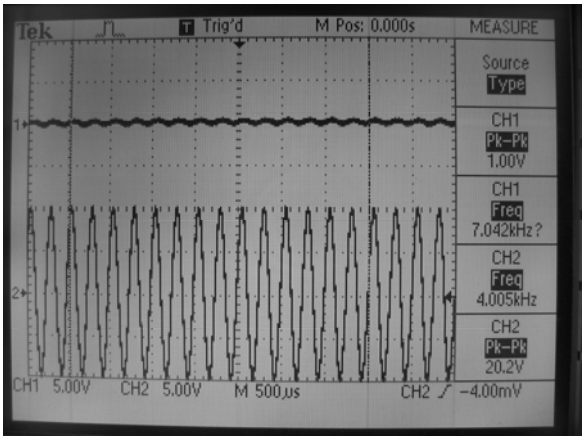


Figure 5.8

Figures 5.5, 5.6, 5.7, 5.8 show some sample results of inputs and outputs to the low-pass filter on a Tektronix TDS-220 DSO. Shown are several frequencies as the cut-off frequency is crossed. The input is CH2 and the output is CH 1.

High-Pass Filter results

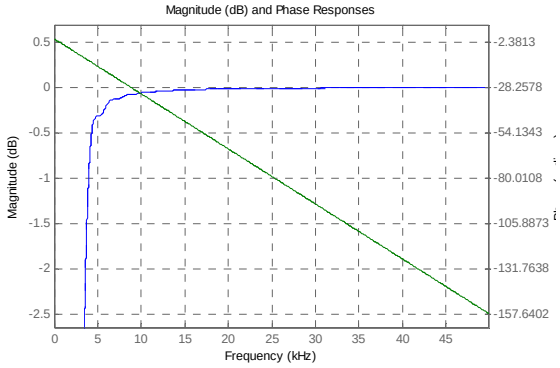


Figure 5.9 shows the magnitude and phase frequency response of the HP filter

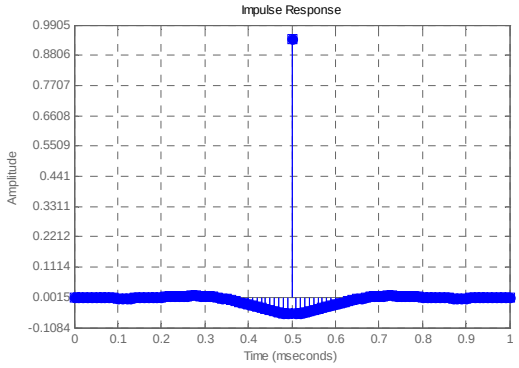


Figure 5.10 shows the impulse response of the HP filter

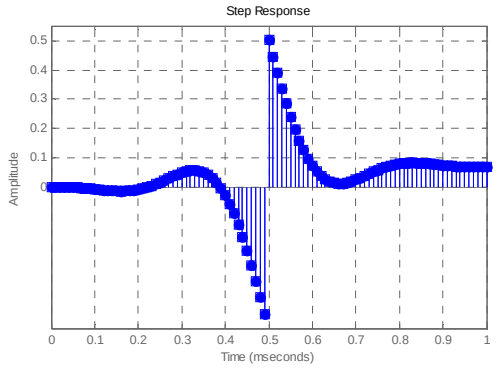


Figure 5.11 shows the step response of the HP filter

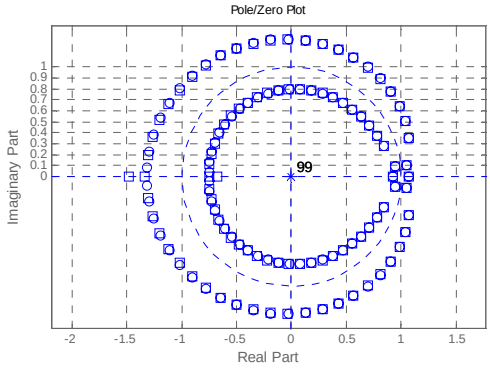


Figure 5.12 shows the Pole/Zero plot of the HP filter

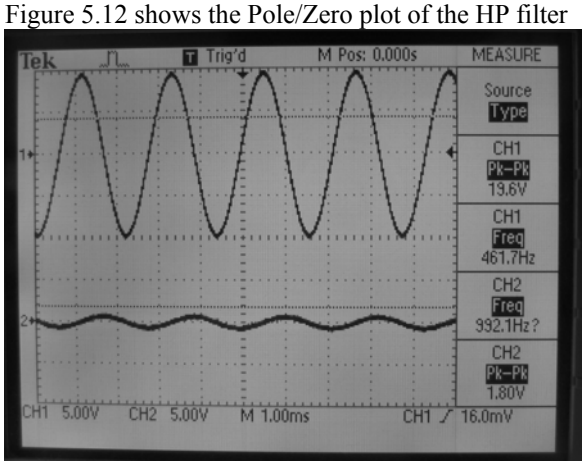


Figure 5.13

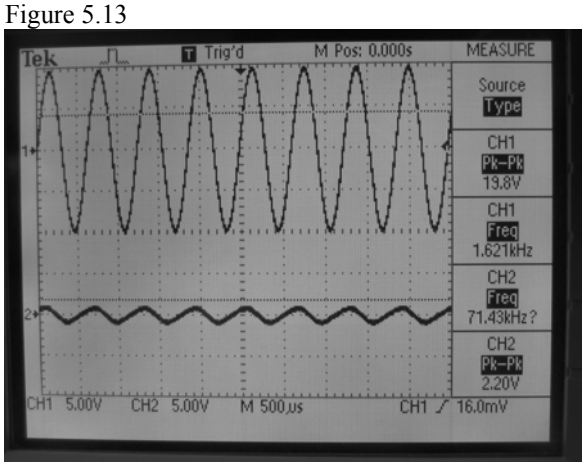


Figure 5.14

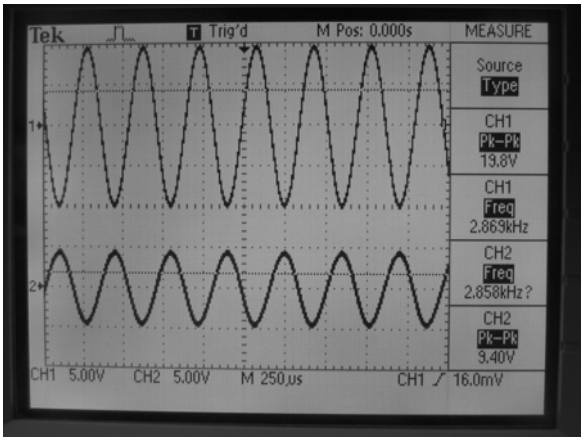


Figure 5.15

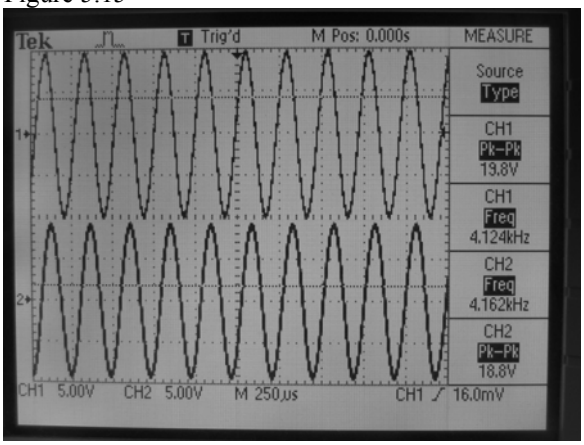


Figure 5.16

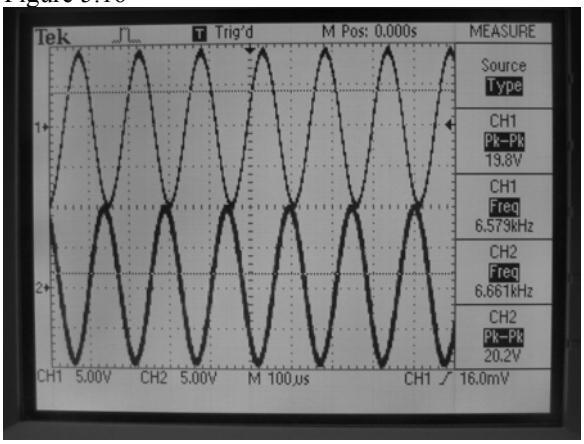


Figure 5.17

Figures 5.13, 5.14, 5.15, 5.16, 5.17 show some sample waveform inputs and outputs of the HP FIR filter on a Tektronix TDS-220 DSO for several frequencies crossing the high-pass cut-off frequency. The input is on CH1 and the output is on CH2.

Band-Pass Filter results

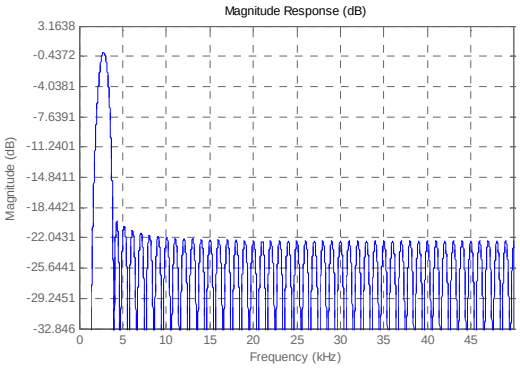


Figure 5.18 shows the magnitude and phase frequency response of the BP filter

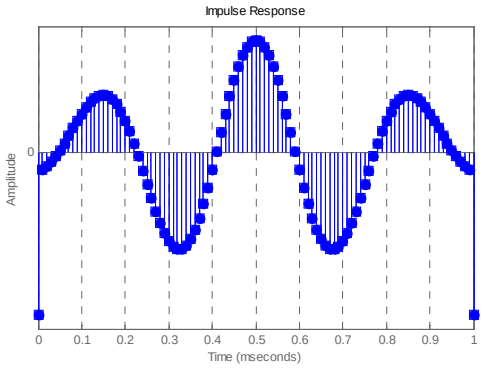


Figure 5.19 shows the impulse response of the BP filter

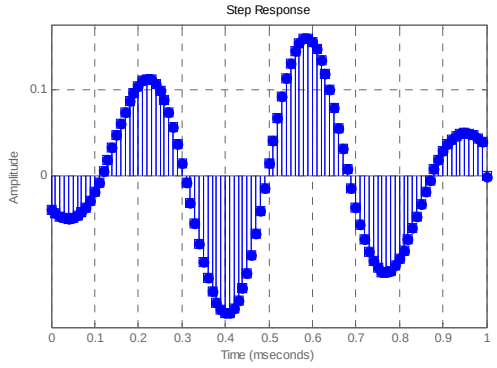


Figure 5.20 shows the step response of the BP filter

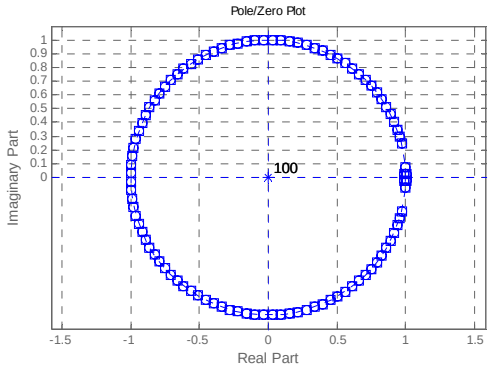


Figure 5.21 shows the Pole/Zero plot of the BP filter

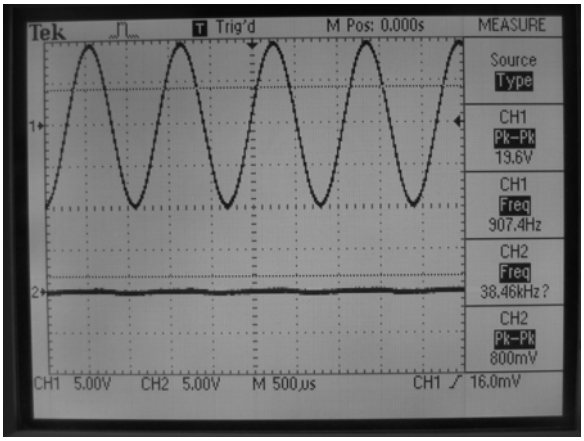


Figure 5.22

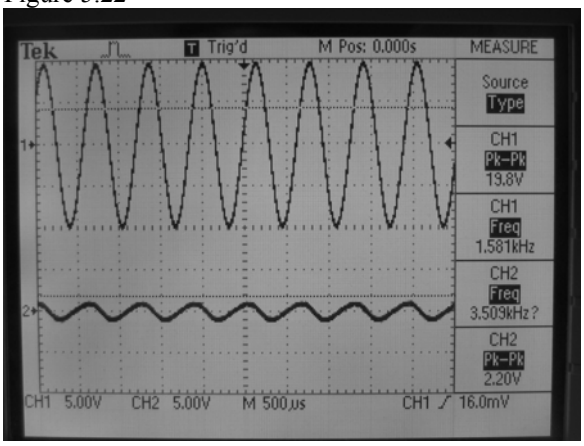


Figure 5.23

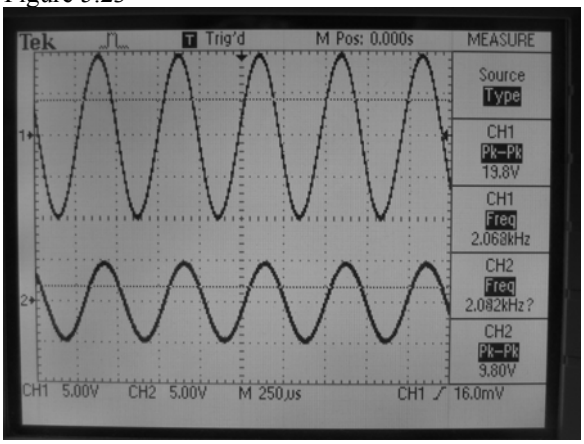


Figure 5.24

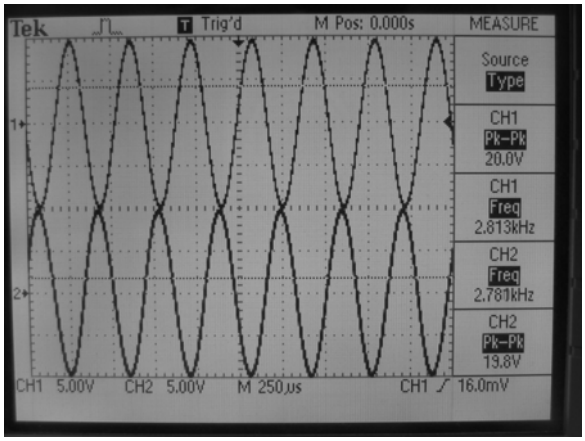


Figure 5.25

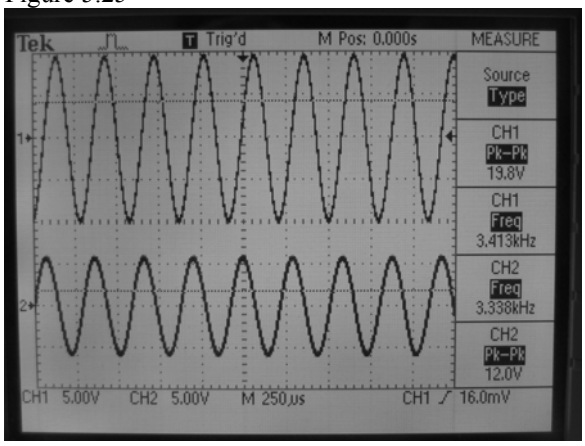


Figure 5.26

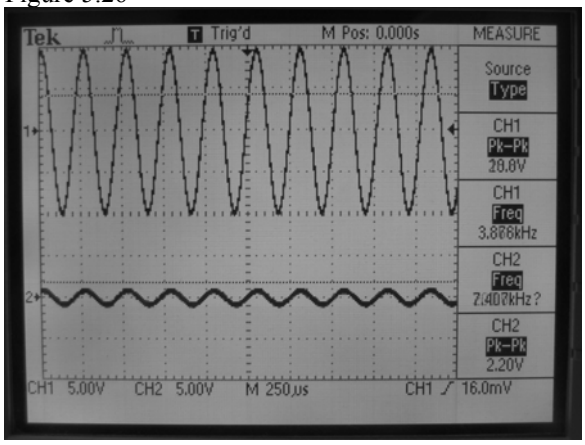


Figure 5.27

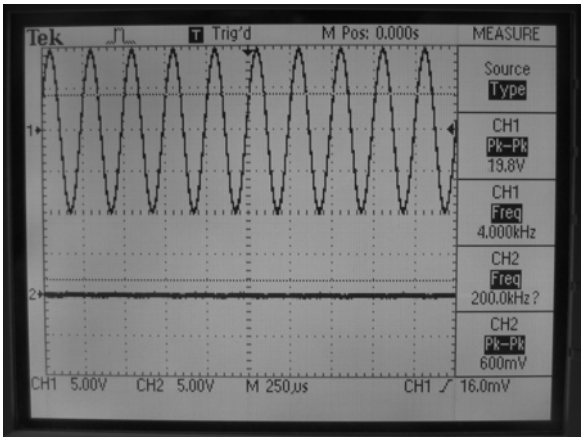


Figure 5.28

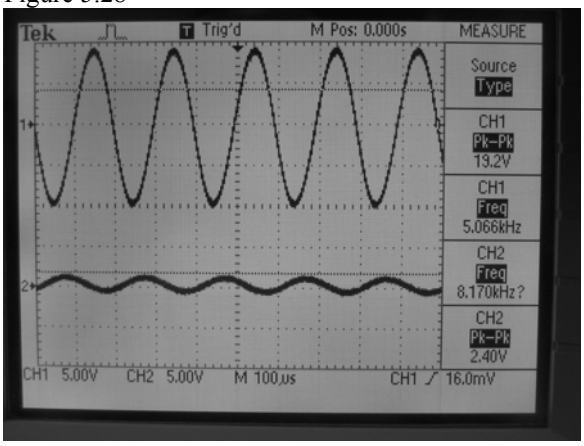


Figure 5.29

Figures 5.22 through 5.29 show some sample results taken at several frequencies crossing the cut-off frequencies of the BP filter. These images were taken on a Tektronix TDS-220 DSO. Notice that figure 5.29 is showing a peak during one of the attenuated lobes which can be seen on the BP frequency response in figure 5.18.

Band-Stop Filter results

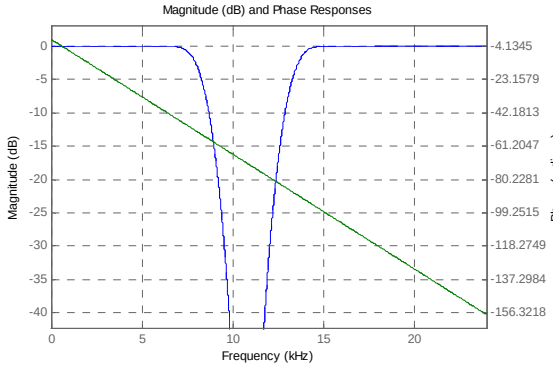


Figure 5.30 shows the magnitude and phase frequency response of the BS filter

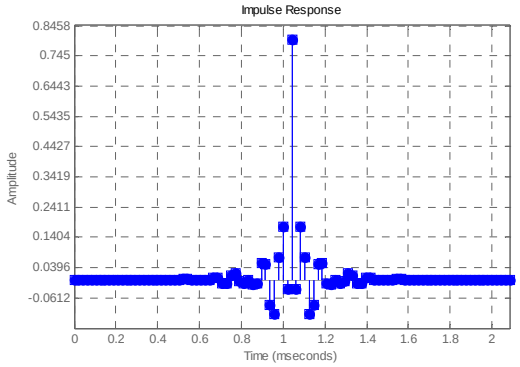


Figure 5.31 shows the impulse response of the BS filter

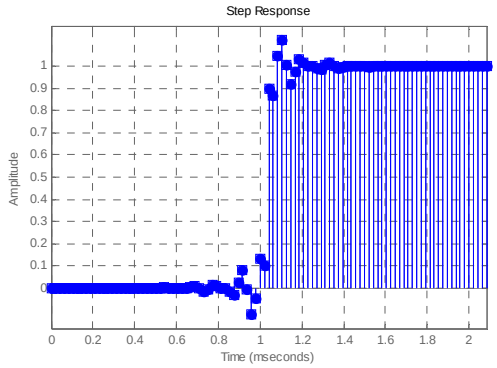


Figure 5.32 shows the step response of the BS filter

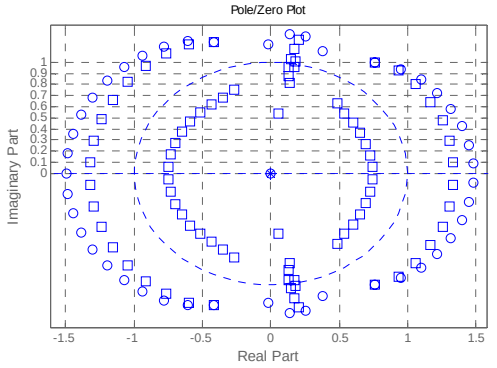


Figure 5.33 shows the Pole/Zero plot of the BS filter, but only the most significant appear, some distant poles and zeros extend as far as 4x10^15.

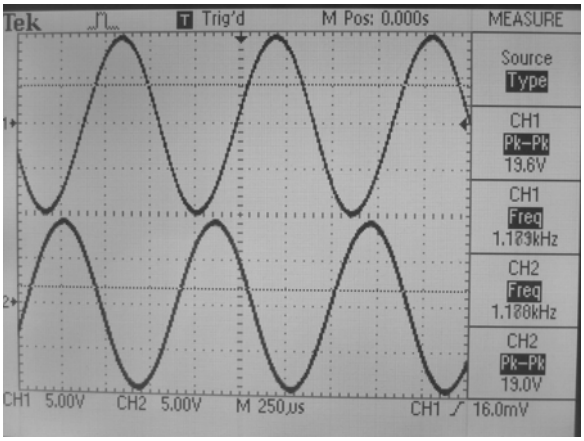


Figure 5.34

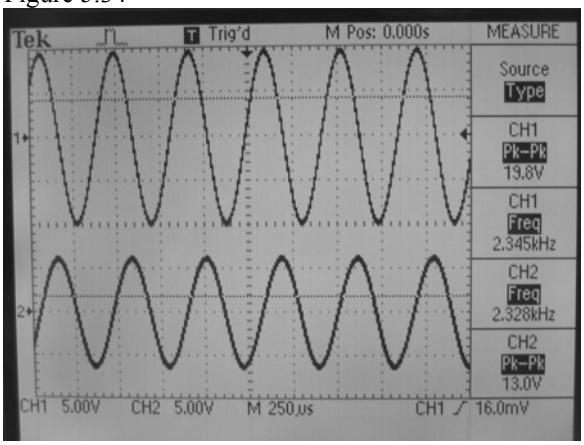


Figure 5.35

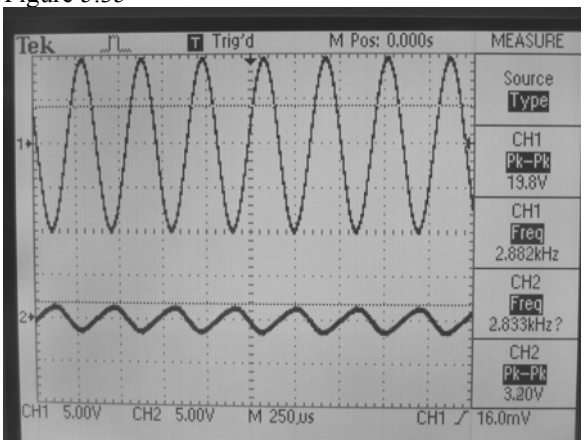


Figure 5.36

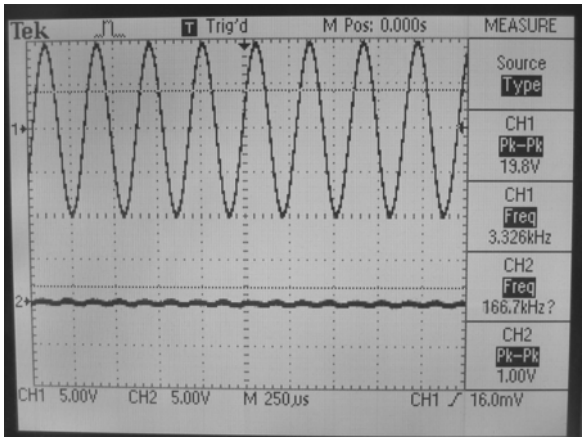


Figure 5.37

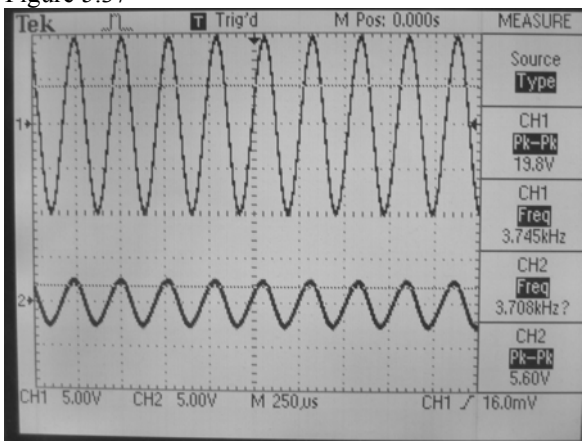


Figure 5.38

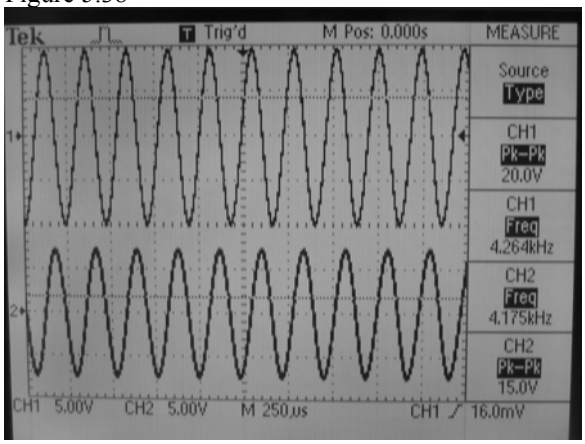


Figure 5.39

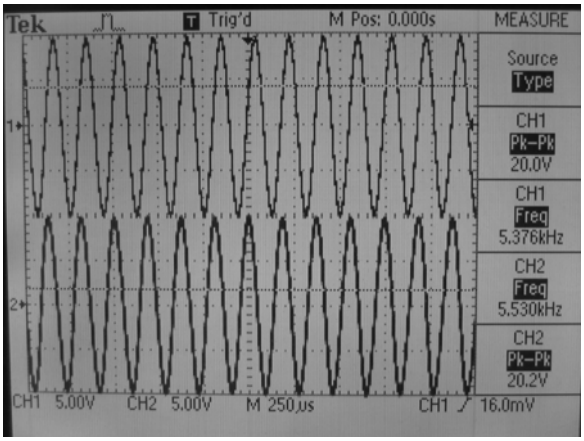


Figure 5.40

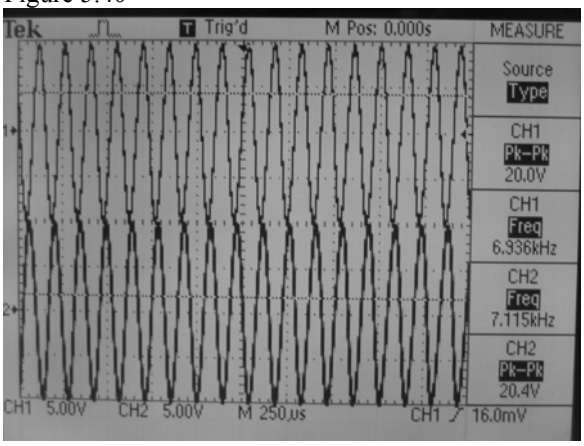


Figure 5.41

Figures 5.34 through 5.41 show several sample results of the BS filter at several frequencies crossing through the cut-off frequencies on a Tektronix TDS-220 DSO.

6. Verilog HDL Code

Code Index

DSP module

```
module DSP(clk, adc, busy, rc, ldac, dac, reset, led,
                    mode, sram_ce, flash_ce);
          //,adcout,twosComp,sFixPt19,FIRo);
   input clk,reset;
   input [1:0]mode;
   output led, sram_ce, flash_ce;
   wire clk,reset;
   wire [1:0]mode;

   reg [7:0] count;

//ADC
   input [15:0] adc;
   input busy;
   output rc;
   wire busy;


//DAC
   output [15:0] dac;
   output ldac;

          //test fixture
//        output [15:0]adcout,twosComp;
//        output [18:0] sFixPt19,FIRo;

   wire [15:0] adc,dac,adcout,twosComp;
   wire [18:0] sFixPt19,FIRo;

assign sram_ce = 1;
assign flash_ce = 1;
assign led = ~reset;

ADC     adc1(clk, reset,count, adc, rc, busy, adcout);

SignFixPt19 conv1(adcout, sFixPt19);

FIR     filter(clk,reset,count,mode, sFixPt19, FIRo);

UnSignFixPt19 conv2(FIRo,twosComp);

DAC     dac1(clk, reset,count, twosComp, dac, ldac);

//100kHz =  250 clks at 25Mhz
always@(posedge clk)
        begin
        if(~reset)
                count<=0;
        else
                begin
                count<= count+1;
                if (count>=249)
                        count<= 0;
                end
        end
endmodule
```

ADC module

```
module ADC(clk, reset,count, adc, rc, busy, adcout);
   input [15:0] adc;
   input [7:0] count;
   input clk, reset, busy;
   output rc;
   output [15:0] adcout;
   reg rc;
   reg [15:0] adcout;
   wire [7:0] count;
   wire [15:0] adc;

//100kHz =  250 clks at 25Mhz

always@(posedge clk)
        begin
        if(~reset)
                begin
                adcout<=0;
                end
        else
                begin
                if(count==145)
                        rc<=0;
                if(count==147)
                        rc<=1;
                if((count==249) && (busy))
                        adcout<=adc;
                end
        end
endmodule
```

SignFixPt19 module

```
module SignFixPt19(twosComp16, sFixPt19);
   input [15:0] twosComp16;
   output [18:0] sFixPt19;

   reg [14:0]untwos;
   reg [17:0]mul1;
   reg [18:0]sFixPt19;
   wire [15:0] twosComp16;

always@(twosComp16 or untwos or mul1 or
sFixPt19 )
   begin
  if(twosComp16[15])
        begin
        untwos = ~twosComp16[14:0] + 1;
        mul1 = 3'b101 * untwos[14:0];
        sFixPt19 = {3'b100,mul1[17:2]};
        end
   else
        begin
        mul1 = 3'b101 * twosComp16[14:0];
        sFixPt19 = {3'b000,mul1[17:2]};
        end
   end
endmodule
```

FIR module

```
module FIR(clk,reset,count,mode, FIRi, FIRo);//,ys,x,y,a,b,p,q);
   input [18:0] FIRi;
   input [7:0] count;
   input [1:0]mode;
   input clk, reset;
   output [18:0] FIRo;//,ys,x,y,a,b,p,q;
   wire [7:0] count;
   wire [1:0]mode;

          reg [18:0] FIRo,x,y,a,b,ys;
   reg [6:0] tapN;
          reg [18:0] w[0:100];
          wire [18:0] h,p,q;


Tap f1(mode,tapN,h);
Mul m1(x, y, p);
Add ad1(a, b, q);

always@(posedge clk)
          begin
          if(~reset)
                    begin
                    w[0]<=0; w[1]<=0; w[2]<=0; w[3]<=0; w[4]<=0; w[5]<=0; w[6]<=0; w[7]<=0; w[8]<=0;
w[9]<=0;
                    w[10]<=0; w[11]<=0; w[12]<=0; w[13]<=0; w[14]<=0; w[15]<=0; w[16]<=0; w[17]<=0;
w[18]<=0; w[19]<=0;
                    w[20]<=0; w[21]<=0; w[22]<=0; w[23]<=0; w[24]<=0; w[25]<=0; w[26]<=0; w[27]<=0;
w[28]<=0; w[29]<=0;
                    w[30]<=0; w[31]<=0; w[32]<=0; w[33]<=0; w[34]<=0; w[35]<=0; w[36]<=0; w[37]<=0;
w[38]<=0; w[39]<=0;
                    w[40]<=0; w[41]<=0; w[42]<=0; w[43]<=0; w[44]<=0; w[45]<=0; w[46]<=0; w[47]<=0;
w[48]<=0; w[49]<=0;
                    w[50]<=0; w[51]<=0; w[52]<=0; w[53]<=0; w[54]<=0; w[55]<=0; w[56]<=0; w[57]<=0;
w[58]<=0; w[59]<=0;
```

```
                w[60]<=0; w[61]<=0; w[62]<=0; w[63]<=0; w[64]<=0; w[65]<=0; w[66]<=0; w[67]<=0;
w[68]<=0; w[69]<=0;
                w[70]<=0; w[71]<=0; w[72]<=0; w[73]<=0; w[74]<=0; w[75]<=0; w[76]<=0; w[77]<=0;
w[78]<=0; w[79]<=0;
                w[80]<=0; w[81]<=0; w[82]<=0; w[83]<=0; w[84]<=0; w[85]<=0; w[86]<=0; w[87]<=0;
w[88]<=0; w[89]<=0;
                w[90]<=0; w[91]<=0; w[92]<=0; w[93]<=0; w[94]<=0; w[95]<=0; w[96]<=0; w[97]<=0;
w[98]<=0; w[99]<=0;
                w[100]<=0;
                FIRo<=0; a<=0; b<=0;      ys<=0; x<=0; y<=0; tapN<=0;
                end
        else
                begin
// yo = h0*w0 + h1*w1 + h2*w2 + h3*w3 + h4*w4 + h5*w5 + h6*w6 ...  FIRi=w0

if(count==0)
        begin
        w[0]<=FIRi; w[1]<=w[0]; w[2]<=w[1]; w[3]<=w[2]; w[4]<=w[3]; w[5]<=w[4]; w[6]<=w[5]; w[7]<=w[6];
w[8]<=w[7]; w[9]<=w[8];
        w[10]<=w[9]; w[11]<=w[10]; w[12]<=w[11]; w[13]<=w[12]; w[14]<=w[13]; w[15]<=w[14];
w[16]<=w[15]; w[17]<=w[16]; w[18]<=w[17]; w[19]<=w[18];
        w[20]<=w[19]; w[21]<=w[20]; w[22]<=w[21]; w[23]<=w[22]; w[24]<=w[23]; w[25]<=w[24];
w[26]<=w[25]; w[27]<=w[26]; w[28]<=w[27]; w[29]<=w[28];
        w[30]<=w[29]; w[31]<=w[30]; w[32]<=w[31]; w[33]<=w[32]; w[34]<=w[33]; w[35]<=w[34];
w[36]<=w[35]; w[37]<=w[36]; w[38]<=w[37]; w[39]<=w[38];
        w[40]<=w[39]; w[41]<=w[40]; w[42]<=w[41]; w[43]<=w[42]; w[44]<=w[43]; w[45]<=w[44];
w[46]<=w[45]; w[47]<=w[46]; w[48]<=w[47]; w[49]<=w[48];
        w[50]<=w[49]; w[51]<=w[50]; w[52]<=w[51]; w[53]<=w[52]; w[54]<=w[53]; w[55]<=w[54];
w[56]<=w[55]; w[57]<=w[56]; w[58]<=w[57]; w[59]<=w[58];
        w[60]<=w[59]; w[61]<=w[60]; w[62]<=w[61]; w[63]<=w[62]; w[64]<=w[63]; w[65]<=w[64];
w[66]<=w[65]; w[67]<=w[66]; w[68]<=w[67]; w[69]<=w[68];
        w[70]<=w[69]; w[71]<=w[70]; w[72]<=w[71]; w[73]<=w[72]; w[74]<=w[73]; w[75]<=w[74];
w[76]<=w[75]; w[77]<=w[76]; w[78]<=w[77]; w[79]<=w[78];
        w[80]<=w[79]; w[81]<=w[80]; w[82]<=w[81]; w[83]<=w[82]; w[84]<=w[83]; w[85]<=w[84];
w[86]<=w[85]; w[87]<=w[86]; w[88]<=w[87]; w[89]<=w[88];
        w[90]<=w[89]; w[91]<=w[90]; w[92]<=w[91]; w[93]<=w[92]; w[94]<=w[93]; w[95]<=w[94];
w[96]<=w[95]; w[97]<=w[96]; w[98]<=w[97]; w[99]<=w[98];
        w[100]<=w[99];
        tapN<=0;
        end
if(count==1)
                        begin x<=h; y<=w[0]; end
                if(count==2)
                        begin a<=ys; b<=p; tapN<=1;  end
                if(count==3)
                        begin ys<=q; x<=h; y<=w[1]; end
                if(count==4)
                        begin a<=ys; b<=p; tapN<=2; end
                if(count==5)
                        begin ys<=q; x<=h; y<=w[2]; end
                if(count==6)
                        begin a<=ys; b<=p; tapN<=3; end
                if(count==7)
                        begin ys<=q; x<=h; y<=w[3]; end
                if(count==8)
                        begin a<=ys; b<=p; tapN<=4; end
                if(count==9)
```

```
            begin ys<=q; x<=h; y<=w[4]; end
if(count==10)
            begin a<=ys; b<=p; tapN<=5; end
if(count==11)
            begin ys<=q; x<=h; y<=w[5]; end
if(count==12)
            begin a<=ys; b<=p; tapN<=6; end
if(count==13)
            begin ys<=q; x<=h; y<=w[6]; end
if(count==14)
            begin a<=ys; b<=p; tapN<=7; end
if(count==15)
            begin ys<=q; x<=h; y<=w[7]; end
if(count==16)
            begin a<=ys; b<=p; tapN<=8; end
if(count==17)
            begin ys<=q; x<=h; y<=w[8]; end
if(count==18)
            begin a<=ys; b<=p; tapN<=9; end
if(count==19)
            begin ys<=q; x<=h; y<=w[9]; end
if(count==20)
            begin a<=ys; b<=p; tapN<=10; end
if(count==21)
            begin ys<=q; x<=h; y<=w[10]; end
if(count==22)
            begin a<=ys; b<=p; tapN<=11;  end
if(count==23)
            begin ys<=q; x<=h; y<=w[11]; end
if(count==24)
            begin a<=ys; b<=p; tapN<=12; end
if(count==25)
            begin ys<=q; x<=h; y<=w[12]; end
if(count==26)
            begin a<=ys; b<=p; tapN<=13; end
if(count==27)
            begin ys<=q; x<=h; y<=w[13]; end
if(count==28)
            begin a<=ys; b<=p; tapN<=14; end
if(count==29)
            begin ys<=q; x<=h; y<=w[14]; end
if(count==30)
            begin a<=ys; b<=p; tapN<=15; end
if(count==31)
            begin ys<=q; x<=h; y<=w[15]; end
if(count==32)
            begin a<=ys; b<=p; tapN<=16; end
if(count==33)
            begin ys<=q; x<=h; y<=w[16]; end
if(count==34)
            begin a<=ys; b<=p; tapN<=17; end
if(count==35)
            begin ys<=q; x<=h; y<=w[17]; end
if(count==36)
            begin a<=ys; b<=p; tapN<=18; end
if(count==37)
            begin ys<=q; x<=h; y<=w[18]; end
```

```
if(count==38)
        begin a<=ys; b<=p; tapN<=19; end
if(count==39)
        begin ys<=q; x<=h; y<=w[19]; end
if(count==40)
        begin a<=ys; b<=p; tapN<=20; end
if(count==41)
        begin ys<=q; x<=h; y<=w[20]; end
if(count==42)
        begin a<=ys; b<=p; tapN<=21;  end
if(count==43)
        begin ys<=q; x<=h; y<=w[21]; end
if(count==44)
        begin a<=ys; b<=p; tapN<=22; end
if(count==45)
        begin ys<=q; x<=h; y<=w[22]; end
if(count==46)
        begin a<=ys; b<=p; tapN<=23; end
if(count==47)
        begin ys<=q; x<=h; y<=w[23]; end
if(count==48)
        begin a<=ys; b<=p; tapN<=24; end
if(count==49)
        begin ys<=q; x<=h; y<=w[24]; end
if(count==50)
        begin a<=ys; b<=p; tapN<=25; end
if(count==51)
        begin ys<=q; x<=h; y<=w[25]; end
if(count==52)
        begin a<=ys; b<=p; tapN<=26; end
if(count==53)
        begin ys<=q; x<=h; y<=w[26]; end
if(count==54)
        begin a<=ys; b<=p; tapN<=27; end
if(count==55)
        begin ys<=q; x<=h; y<=w[27]; end
if(count==56)
        begin a<=ys; b<=p; tapN<=28; end
if(count==57)
        begin ys<=q; x<=h; y<=w[28]; end
if(count==58)
        begin a<=ys; b<=p; tapN<=29; end
if(count==59)
        begin ys<=q; x<=h; y<=w[29]; end
if(count==60)
        begin a<=ys; b<=p; tapN<=30; end
if(count==61)
        begin ys<=q; x<=h; y<=w[30]; end
if(count==62)
        begin a<=ys; b<=p; tapN<=31;  end
if(count==63)
        begin ys<=q; x<=h; y<=w[31]; end
if(count==64)
        begin a<=ys; b<=p; tapN<=32; end
if(count==65)
        begin ys<=q; x<=h; y<=w[32]; end
if(count==66)
```

```
            begin a<=ys; b<=p; tapN<=33; end
if(count==67)
            begin ys<=q; x<=h; y<=w[33]; end
if(count==68)
            begin a<=ys; b<=p; tapN<=34; end
if(count==69)
            begin ys<=q; x<=h; y<=w[34]; end
if(count==70)
            begin a<=ys; b<=p; tapN<=35; end
if(count==71)
            begin ys<=q; x<=h; y<=w[35]; end
if(count==72)
            begin a<=ys; b<=p; tapN<=36; end
if(count==73)
            begin ys<=q; x<=h; y<=w[36]; end
if(count==74)
            begin a<=ys; b<=p; tapN<=37; end
if(count==75)
            begin ys<=q; x<=h; y<=w[37]; end
if(count==76)
            begin a<=ys; b<=p; tapN<=38; end
if(count==77)
            begin ys<=q; x<=h; y<=w[38]; end
if(count==78)
            begin a<=ys; b<=p; tapN<=39; end
if(count==79)
            begin ys<=q; x<=h; y<=w[39]; end
if(count==80)
            begin a<=ys; b<=p; tapN<=40; end
if(count==81)
            begin ys<=q; x<=h; y<=w[40]; end
if(count==82)
            begin a<=ys; b<=p; tapN<=41;  end
if(count==83)
            begin ys<=q; x<=h; y<=w[41]; end
if(count==84)
            begin a<=ys; b<=p; tapN<=42; end
if(count==85)
            begin ys<=q; x<=h; y<=w[42]; end
if(count==86)
            begin a<=ys; b<=p; tapN<=43; end
if(count==87)
            begin ys<=q; x<=h; y<=w[43]; end
if(count==88)
            begin a<=ys; b<=p; tapN<=44; end
if(count==89)
            begin ys<=q; x<=h; y<=w[44]; end
if(count==90)
            begin a<=ys; b<=p; tapN<=45; end
if(count==91)
            begin ys<=q; x<=h; y<=w[45]; end
if(count==92)
            begin a<=ys; b<=p; tapN<=46; end
if(count==93)
            begin ys<=q; x<=h; y<=w[46]; end
if(count==94)
            begin a<=ys; b<=p; tapN<=47; end
```

```
if(count==95)
        begin ys<=q; x<=h; y<=w[47]; end
if(count==96)
        begin a<=ys; b<=p; tapN<=48; end
if(count==97)
        begin ys<=q; x<=h; y<=w[48]; end
if(count==98)
        begin a<=ys; b<=p; tapN<=49; end
if(count==99)
        begin ys<=q; x<=h; y<=w[49]; end
if(count==100)
        begin a<=ys; b<=p; tapN<=50; end
if(count==101)
        begin ys<=q; x<=h; y<=w[50]; end
if(count==102)
        begin a<=ys; b<=p; tapN<=51;  end
if(count==103)
        begin ys<=q; x<=h; y<=w[51]; end
if(count==104)
        begin a<=ys; b<=p; tapN<=52; end
if(count==105)
        begin ys<=q; x<=h; y<=w[52]; end
if(count==106)
        begin a<=ys; b<=p; tapN<=53; end
if(count==107)
        begin ys<=q; x<=h; y<=w[53]; end
if(count==108)
        begin a<=ys; b<=p; tapN<=54; end
if(count==109)
        begin ys<=q; x<=h; y<=w[54]; end
if(count==110)
        begin a<=ys; b<=p; tapN<=55; end
if(count==111)
        begin ys<=q; x<=h; y<=w[55]; end
if(count==112)
        begin a<=ys; b<=p; tapN<=56; end
if(count==113)
        begin ys<=q; x<=h; y<=w[56]; end
if(count==114)
        begin a<=ys; b<=p; tapN<=57; end
if(count==115)
        begin ys<=q; x<=h; y<=w[57]; end
if(count==116)
        begin a<=ys; b<=p; tapN<=58; end
if(count==117)
        begin ys<=q; x<=h; y<=w[58]; end
if(count==118)
        begin a<=ys; b<=p; tapN<=59; end
if(count==119)
        begin ys<=q; x<=h; y<=w[59]; end
if(count==120)
        begin a<=ys; b<=p; tapN<=60; end
if(count==121)
        begin ys<=q; x<=h; y<=w[60]; end
if(count==122)
        begin a<=ys; b<=p; tapN<=61;  end
if(count==123)
```

```
            begin ys<=q; x<=h; y<=w[61]; end
if(count==124)
            begin a<=ys; b<=p; tapN<=62; end
if(count==125)
            begin ys<=q; x<=h; y<=w[62]; end
if(count==126)
            begin a<=ys; b<=p; tapN<=63; end
if(count==127)
            begin ys<=q; x<=h; y<=w[63]; end
if(count==128)
            begin a<=ys; b<=p; tapN<=64; end
if(count==129)
            begin ys<=q; x<=h; y<=w[64]; end
if(count==130)
            begin a<=ys; b<=p; tapN<=65; end
if(count==131)
            begin ys<=q; x<=h; y<=w[65]; end
if(count==132)
            begin a<=ys; b<=p; tapN<=66; end
if(count==133)
            begin ys<=q; x<=h; y<=w[66]; end
if(count==134)
            begin a<=ys; b<=p; tapN<=67; end
if(count==135)
            begin ys<=q; x<=h; y<=w[67]; end
if(count==136)
            begin a<=ys; b<=p; tapN<=68; end
if(count==137)
            begin ys<=q; x<=h; y<=w[68]; end
if(count==138)
            begin a<=ys; b<=p; tapN<=69; end
if(count==139)
            begin ys<=q; x<=h; y<=w[69]; end
if(count==140)
            begin a<=ys; b<=p; tapN<=70; end
if(count==141)
            begin ys<=q; x<=h; y<=w[70]; end
if(count==142)
            begin a<=ys; b<=p; tapN<=71;  end
if(count==143)
            begin ys<=q; x<=h; y<=w[71]; end
if(count==144)
            begin a<=ys; b<=p; tapN<=72; end
if(count==145)
            begin ys<=q; x<=h; y<=w[72]; end
if(count==146)
            begin a<=ys; b<=p; tapN<=73; end
if(count==147)
            begin ys<=q; x<=h; y<=w[73]; end
if(count==148)
            begin a<=ys; b<=p; tapN<=74; end
if(count==149)
            begin ys<=q; x<=h; y<=w[74]; end
if(count==150)
            begin a<=ys; b<=p; tapN<=75; end
if(count==151)
            begin ys<=q; x<=h; y<=w[75]; end
```

```
if(count==152)
        begin a<=ys; b<=p; tapN<=76; end
if(count==153)
        begin ys<=q; x<=h; y<=w[76]; end
if(count==154)
        begin a<=ys; b<=p; tapN<=77; end
if(count==155)
        begin ys<=q; x<=h; y<=w[77]; end
if(count==156)
        begin a<=ys; b<=p; tapN<=78; end
if(count==157)
        begin ys<=q; x<=h; y<=w[78]; end
if(count==158)
        begin a<=ys; b<=p; tapN<=79; end
if(count==159)
        begin ys<=q; x<=h; y<=w[79]; end
if(count==160)
        begin a<=ys; b<=p; tapN<=80; end
if(count==161)
        begin ys<=q; x<=h; y<=w[80]; end
if(count==162)
        begin a<=ys; b<=p; tapN<=81;  end
if(count==163)
        begin ys<=q; x<=h; y<=w[81]; end
if(count==164)
        begin a<=ys; b<=p; tapN<=82; end
if(count==165)
        begin ys<=q; x<=h; y<=w[82]; end
if(count==166)
        begin a<=ys; b<=p; tapN<=83; end
if(count==167)
        begin ys<=q; x<=h; y<=w[83]; end
if(count==168)
        begin a<=ys; b<=p; tapN<=84; end
if(count==169)
        begin ys<=q; x<=h; y<=w[84]; end
if(count==170)
        begin a<=ys; b<=p; tapN<=85; end
if(count==171)
        begin ys<=q; x<=h; y<=w[85]; end
if(count==172)
        begin a<=ys; b<=p; tapN<=86; end
if(count==173)
        begin ys<=q; x<=h; y<=w[86]; end
if(count==174)
        begin a<=ys; b<=p; tapN<=87; end
if(count==175)
        begin ys<=q; x<=h; y<=w[87]; end
if(count==176)
        begin a<=ys; b<=p; tapN<=88; end
if(count==177)
        begin ys<=q; x<=h; y<=w[88]; end
if(count==178)
        begin a<=ys; b<=p; tapN<=89; end
if(count==179)
        begin ys<=q; x<=h; y<=w[89]; end
if(count==180)
```

```
                begin a<=ys; b<=p; tapN<=90; end
        if(count==181)
                begin ys<=q; x<=h; y<=w[90]; end
        if(count==182)
                begin a<=ys; b<=p; tapN<=91;  end
        if(count==183)
                begin ys<=q; x<=h; y<=w[91]; end
        if(count==184)
                begin a<=ys; b<=p; tapN<=92; end
        if(count==185)
                begin ys<=q; x<=h; y<=w[92]; end
        if(count==186)
                begin a<=ys; b<=p; tapN<=93; end
        if(count==187)
                begin ys<=q; x<=h; y<=w[93]; end
        if(count==188)
                begin a<=ys; b<=p; tapN<=94; end
        if(count==189)
                begin ys<=q; x<=h; y<=w[94]; end
        if(count==190)
                begin a<=ys; b<=p; tapN<=95; end
        if(count==191)
                begin ys<=q; x<=h; y<=w[95]; end
        if(count==192)
                begin a<=ys; b<=p; tapN<=96; end
        if(count==193)
                begin ys<=q; x<=h; y<=w[96]; end
        if(count==194)
                begin a<=ys; b<=p; tapN<=97; end
        if(count==195)
                begin ys<=q; x<=h; y<=w[97]; end
        if(count==196)
                begin a<=ys; b<=p; tapN<=98; end
        if(count==197)
                begin ys<=q; x<=h; y<=w[98]; end
        if(count==198)
                begin a<=ys; b<=p; tapN<=99; end
        if(count==199)
                begin ys<=q; x<=h; y<=w[99]; end
        if(count==200)
                begin a<=ys; b<=p; tapN<=100; end
        if(count==201)
                begin ys<=q; x<=h; y<=w[100]; end
//~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~
        if(count==202)
                begin a<=ys; b<=p;  end
        if(count==203)
                begin FIRo<=q; ys<=0; end
        end
    end

endmodule
```

Tap Module

```
module Tap(mode,tapN, h);
   input [6:0] tapN;
   input [1:0]mode;
   output [18:0] h;
   reg [18:0] h;
   wire [6:0] tapN;
   wire [1:0]mode;

always@(mode or tapN)
        begin
        case(~mode)
        0: begin
                case(tapN)
// 101 Tap FIR Low-Pass 1kHz 100kSA/s Bartlett
Window
                0:h=19'b0000000000000000000;
                1:h=19'b0000000000000000001;
                2:h=19'b0000000000000000110;
                3:h=19'b0000000000000001101;
                4:h=19'b0000000000000010111;
                5:h=19'b0000000000000100101;
                6:h=19'b0000000000000110110;
                7:h=19'b0000000000001001011;
                8:h=19'b0000000000001100011;
                9:h=19'b0000000000001111111;
                10:h=19'b0000000000010011110;
                11:h=19'b0000000000011000010;
                12:h=19'b0000000000011101001;
                13:h=19'b0000000000100010100;
                14:h=19'b0000000000101000011;
                15:h=19'b0000000000101110110;
                16:h=19'b0000000000110101100;
                17:h=19'b0000000000111100111;
                18:h=19'b0000000001000100101;
                19:h=19'b0000000001001100111;
                20:h=19'b0000000001010101100;
                21:h=19'b0000000001011110100;
                22:h=19'b0000000001101000000;
                23:h=19'b0000000001110001111;
                24:h=19'b0000000001111100001;
                25:h=19'b0000000010000110110;
                26:h=19'b0000000010010001110;
                27:h=19'b0000000010011101000;
                28:h=19'b0000000010101000100;
                29:h=19'b0000000010110100010;
                30:h=19'b0000000011000000010;
                31:h=19'b0000000011001100100;
                32:h=19'b0000000011011000111;
                33:h=19'b0000000011100101010;
                34:h=19'b0000000011110001111;
                35:h=19'b0000000011111110100;
                36:h=19'b0000000100001011001;
                37:h=19'b0000000100010111101;
                38:h=19'b0000000100100100010;
                39:h=19'b0000000100110000101;
                40:h=19'b0000000100111100111;
                41:h=19'b0000000101001001000;
                42:h=19'b0000000101010101000;
                43:h=19'b0000000101100000101;
                44:h=19'b0000000101101011111;
                45:h=19'b0000000101110110111;
                46:h=19'b0000000110000001100;
                47:h=19'b0000000110001011110;
                48:h=19'b0000000110010101100;
                49:h=19'b0000000110011110110;
                50:h=19'b0000000110100111100;
                51:h=19'b0000000110011110110;
                52:h=19'b0000000110010101100;
                53:h=19'b0000000110001011110;
                54:h=19'b0000000110000001100;
                55:h=19'b0000000101110110111;
                56:h=19'b0000000101101011111;
                57:h=19'b0000000101100000101;
                58:h=19'b0000000101010101000;
                59:h=19'b0000000101001001000;
                60:h=19'b0000000100111100111;
                61:h=19'b0000000100110000101;
                62:h=19'b0000000100100100010;
                63:h=19'b0000000100010111101;
                64:h=19'b0000000100001011001;
                65:h=19'b0000000011111110100;
                66:h=19'b0000000011110001111;
                67:h=19'b0000000011100101010;
                68:h=19'b0000000011011000111;
                69:h=19'b0000000011001100100;
                70:h=19'b0000000011000000010;
                71:h=19'b0000000010110100010;
                72:h=19'b0000000010101000100;
                73:h=19'b0000000010011101000;
                74:h=19'b0000000010010001110;
                75:h=19'b0000000010000110110;
                76:h=19'b0000000001111100001;
                77:h=19'b0000000001110001111;
                78:h=19'b0000000001101000000;
                79:h=19'b0000000001011110100;
                80:h=19'b0000000001010101100;
                81:h=19'b0000000001001100111;
                82:h=19'b0000000001000100101;
                83:h=19'b0000000000111100111;
                84:h=19'b0000000000110101100;
                85:h=19'b0000000000101110110;
                86:h=19'b0000000000101000011;
                87:h=19'b0000000000100010100;
                88:h=19'b0000000000011101001;
                89:h=19'b0000000000011000010;
                90:h=19'b0000000000010011110;
                91:h=19'b0000000000001111111;
                92:h=19'b0000000000001100011;
                93:h=19'b0000000000001001011;
                94:h=19'b0000000000000110110;
                95:h=19'b0000000000000100101;
```

```
                96:h=19'b0000000000000010111;
                97:h=19'b0000000000000001101;
                98:h=19'b0000000000000000110;
                99:h=19'b0000000000000000001;
                100:h=19'b0000000000000000000;

        default:h=19'b00000000000000000;
                endcase
                end
        1: begin
                case(tapN)
// 101 Tap FIR High-Pass 3kHz 100kSA/s Bartlett
Window
                0:h=19'b0000000000000000000;
                1:h=19'b1000000000000000011;
                2:h=19'b1000000000000001100;
                3:h=19'b1000000000000011100;
                4:h=19'b1000000000000110001;
                5:h=19'b1000000000001001011;
                6:h=19'b1000000000001100111;
                7:h=19'b1000000000010000011;
                8:h=19'b1000000000010011110;
                9:h=19'b1000000000010110101;
                10:h=19'b1000000000011000110;
                11:h=19'b1000000000011001110;
                12:h=19'b1000000000011001011;
                13:h=19'b1000000000010111010;
                14:h=19'b1000000000010011100;
                15:h=19'b1000000000001101110;
                16:h=19'b1000000000000110001;
                17:h=19'b0000000000000011011;
                18:h=19'b0000000000001110100;
                19:h=19'b0000000000011011001;
                20:h=19'b0000000000101000111;
                21:h=19'b0000000000110111000;
                22:h=19'b0000000001000101001;
                23:h=19'b0000000001010010101;
                24:h=19'b0000000001011110101;
                25:h=19'b0000000001101000010;
                26:h=19'b0000000001101111000;
                27:h=19'b0000000001110001111;
                28:h=19'b0000000001110000001;
                29:h=19'b0000000001101001000;
                30:h=19'b0000000001011100000;
                31:h=19'b0000000001001000100;
                32:h=19'b0000000000101110001;
                33:h=19'b0000000000001100101;
                34:h=19'b1000000000011011110;
                35:h=19'b1000000001001011010;
                36:h=19'b1000000010000001010;
                37:h=19'b1000000010111101010;
                38:h=19'b1000000011111110101;
                39:h=19'b1000000101000100010;
                40:h=19'b1000000110001101000;
                41:h=19'b1000000111010111101;
                42:h=19'b1000001000100010110;
                43:h=19'b1000001001101100111;
                44:h=19'b1000001010110100100;
                45:h=19'b1000001011110111111;
                46:h=19'b1000001100110101100;
                47:h=19'b1000001101101100000;
                48:h=19'b1000001110011010000;
                49:h=19'b1000001110111110010;
                50:h=19'b0011110000110010001;
                51:h=19'b1000001110111110010;
                52:h=19'b1000001110011010000;
                53:h=19'b1000001101101100000;
                54:h=19'b1000001100110101100;
                55:h=19'b1000001011110111111;
                56:h=19'b1000001010110100100;
                57:h=19'b1000001001101100111;
                58:h=19'b1000001000100010110;
                59:h=19'b1000000111010111101;
                60:h=19'b1000000110001101000;
                61:h=19'b1000000101000100010;
                62:h=19'b1000000011111110101;
                63:h=19'b1000000010111101010;
                64:h=19'b1000000010000001010;
                65:h=19'b1000000001001011010;
                66:h=19'b1000000000011011110;
                67:h=19'b0000000000001100101;
                68:h=19'b0000000000101110001;
                69:h=19'b0000000001001000100;
                70:h=19'b0000000001011100000;
                71:h=19'b0000000001101001000;
                72:h=19'b0000000001110000001;
                73:h=19'b0000000001110001111;
                74:h=19'b0000000001101111000;
                75:h=19'b0000000001101000010;
                76:h=19'b0000000001011110101;
                77:h=19'b0000000001010010101;
                78:h=19'b0000000001000101001;
                79:h=19'b0000000000110111000;
                80:h=19'b0000000000101000111;
                81:h=19'b0000000000011011001;
                82:h=19'b0000000000001110100;
                83:h=19'b0000000000000011011;
                84:h=19'b1000000000000110001;
                85:h=19'b1000000000001101110;
                86:h=19'b1000000000010011100;
                87:h=19'b1000000000010111010;
                88:h=19'b1000000000011001011;
                89:h=19'b1000000000011001110;
                90:h=19'b1000000000011000110;
                91:h=19'b1000000000010110101;
                92:h=19'b1000000000010011110;
                93:h=19'b1000000000010000011;
                94:h=19'b1000000000001100111;
                95:h=19'b1000000000001001011;
                96:h=19'b1000000000000110001;
                97:h=19'b1000000000000011100;
                98:h=19'b1000000000000001100;
                99:h=19'b1000000000000000011;
```

```
                100:h=19'b0000000000000000000;


        default:h=19'b000000000000000000;
                endcase
                end
        2: begin
                case(tapN)
// 101 Tap FIR Band-Pass f1:2.5kHz f2:3.0kHz
100kSA/s Window:Chebyshev SideLobe attn:25dB

                0:h=19'b1000001010001111100;
                1:h=19'b1000000001000100001;
                2:h=19'b1000000000110101110;
                3:h=19'b1000000000100011011;
                4:h=19'b1000000000001101011;
                5:h=19'b0000000000001011110;
                6:h=19'b0000000000100111101;
                7:h=19'b0000000001000101010;
                8:h=19'b0000000001100011101;
                9:h=19'b0000000010000001100;
                10:h=19'b0000000010011110000;
                11:h=19'b0000000010110111100;
                12:h=19'b0000000011001101001;
                13:h=19'b0000000011011101101;
                14:h=19'b0000000011100111111;
                15:h=19'b0000000011101011000;
                16:h=19'b0000000011100110100;
                17:h=19'b0000000011011001110;
                18:h=19'b0000000011000100110;
                19:h=19'b0000000010100111101;
                20:h=19'b0000000010000011000;
                21:h=19'b0000000001010111100;
                22:h=19'b0000000000100110011;
                23:h=19'b1000000000001110111;
                24:h=19'b1000000001000110110;
                25:h=19'b1000000001111111101;
                26:h=19'b1000000010110111010;
                27:h=19'b1000000011101011110;
                28:h=19'b1000000100011011010;
                29:h=19'b1000000101000011111;
                30:h=19'b1000000101100011111;
                31:h=19'b1000000101111001110;
                32:h=19'b1000000110000100100;
                33:h=19'b1000000110000011011;
                34:h=19'b1000000101110110000;
                35:h=19'b1000000101011100010;
                36:h=19'b1000000100110110110;
                37:h=19'b1000000100000110100;
                38:h=19'b1000000011001100110;
                39:h=19'b1000000010001011010;
                40:h=19'b1000000001000011111;
                41:h=19'b0000000000000110111;
                42:h=19'b0000000001010010110;
                43:h=19'b0000000010011101011;
                44:h=19'b0000000011100100001;
                45:h=19'b0000000100100100101;
                46:h=19'b0000000101011100110;
                47:h=19'b0000000110001010100;
                48:h=19'b0000000110101100011;
                49:h=19'b0000000111000001010;
                50:h=19'b0000000111001000010;
                51:h=19'b0000000111000001010;
                52:h=19'b0000000110101100011;
                53:h=19'b0000000110001010100;
                54:h=19'b0000000101011100110;
                55:h=19'b0000000100100100101;
                56:h=19'b0000000011100100001;
                57:h=19'b0000000010011101011;
                58:h=19'b0000000001010010110;
                59:h=19'b0000000000000110111;
                60:h=19'b1000000001000011111;
                61:h=19'b1000000010001011010;
                62:h=19'b1000000011001100110;
                63:h=19'b1000001000000110100;
                64:h=19'b1000001001101101110;
                65:h=19'b1000001010111000010;
                66:h=19'b1000001011101100000;
                67:h=19'b1000001110000011011;
                68:h=19'b1000001110000100100;
                69:h=19'b1000001011111001110;
                70:h=19'b1000001011000111111;
                71:h=19'b1000001010000111111;
                72:h=19'b1000001000011011010;
                73:h=19'b1000000111010111110;
                74:h=19'b1000000101101110010;
                75:h=19'b1000000001111111101;
                76:h=19'b1000000001000110110;
                77:h=19'b1000000000001110111;
                78:h=19'b0000000000100110011;
                79:h=19'b0000000001010111100;
                80:h=19'b0000000010000011000;
                81:h=19'b0000000010100111101;
                82:h=19'b0000000011000100110;
                83:h=19'b0000000011011001110;
                84:h=19'b0000000011100110100;
                85:h=19'b0000000011101011000;
                86:h=19'b0000000011100111111;
                87:h=19'b0000000011011101101;
                88:h=19'b0000000011001101001;
                89:h=19'b0000000010110111100;
                90:h=19'b0000000010011110000;
                91:h=19'b0000000010000001100;
                92:h=19'b0000000001100011101;
                93:h=19'b0000000001000101010;
                94:h=19'b0000000000100111101;
                95:h=19'b0000000000000110111;
                96:h=19'b1000000000001101011;
                97:h=19'b1000000000100011011;
                98:h=19'b1000000000110101110;
                99:h=19'b1000000001000100001;
                100:h=19'b1000001010001111100;

        default:h=19'b000000000000000000;
```

```
                endcase
                end
        3: begin
                case(tapN)
// 101 Tap FIR Band-Stop f1:2.5kHz f2:4.0kHz
100kSA/s  Window:Kaiser  Beta: 17

                0:h=19'b0000000000111110000;
                1:h=19'b0000000001010000000;
                2:h=19'b0000000001100000101;
                3:h=19'b0000000001101110111;
                4:h=19'b0000000001111001010;
                5:h=19'b0000000001111110110;
                6:h=19'b0000000001111110011;
                7:h=19'b0000000001110111010;
                8:h=19'b0000000001101001000;
                9:h=19'b0000000001010011101;
                10:h=19'b0000000000110111010;
                11:h=19'b0000000000010100110;
                12:h=19'b1000000000010010110;
                13:h=19'b1000000000111101101;
                14:h=19'b1000000001101010001;
                15:h=19'b1000000010010110000;
                16:h=19'b1000000010111111000;
                17:h=19'b1000000011100010110;
                18:h=19'b1000000011111110111;
                19:h=19'b1000000100010001101;
                20:h=19'b1000000100011001000;
                21:h=19'b1000000100010100001;
                22:h=19'b1000000100000010001;
                23:h=19'b1000000011100011000;
                24:h=19'b1000000010110111100;
                25:h=19'b1000000010000000110;
                26:h=19'b1000000001000000111;
                27:h=19'b0000000000000101100;
                28:h=19'b0000000001001111110;
                29:h=19'b0000000010011010011;
                30:h=19'b0000000011100001110;
                31:h=19'b0000000100100010101;
                32:h=19'b0000000101011001011;
                33:h=19'b0000000110000011011;
                34:h=19'b0000000110011101111;
                35:h=19'b0000000110100111010;
                36:h=19'b0000000110011110011;
                37:h=19'b0000000110000011010;
                38:h=19'b0000000101010110011;
                39:h=19'b0000000100011001011;
                40:h=19'b0000000011001110110;
                41:h=19'b0000000001111001011;
                42:h=19'b0000000000011101001;
                43:h=19'b1000000001000010000;
                44:h=19'b1000000010011111110;
                45:h=19'b1000000011111000000;
                46:h=19'b1000000101000110100;
                47:h=19'b1000000110000111110;
                48:h=19'b1000000110111000110;
                49:h=19'b1000000111010111000;
                50:h=19'b0011110011001010100;
                51:h=19'b1000000111010111000;
                52:h=19'b1000000110111000110;
                53:h=19'b1000000110000111110;
                54:h=19'b1000000101000110100;
                55:h=19'b1000000011111000000;
                56:h=19'b1000000010011111110;
                57:h=19'b1000000001000010000;
                58:h=19'b0000000000011101001;
                59:h=19'b0000000001111001011;
                60:h=19'b0000000011001110110;
                61:h=19'b0000000100011001011;
                62:h=19'b0000000101010110011;
                63:h=19'b0000000110000011010;
                64:h=19'b0000000110011110011;
                65:h=19'b0000000110100111010;
                66:h=19'b0000000110011101111;
                67:h=19'b0000000110000011011;
                68:h=19'b0000000101011001011;
                69:h=19'b0000000100100010101;
                70:h=19'b0000000011100001110;
                71:h=19'b0000000010011010011;
                72:h=19'b0000000001001111110;
                73:h=19'b0000000000000101100;
                74:h=19'b1000000001000000111;
                75:h=19'b1000000010000000110;
                76:h=19'b1000000010110111100;
                77:h=19'b1000000011100011000;
                78:h=19'b1000000100000010001;
                79:h=19'b1000000100010100001;
                80:h=19'b1000001100011001000;
                81:h=19'b1000001100010001101;
                82:h=19'b1000000111111110111;
                83:h=19'b1000000111100010110;
                84:h=19'b1000000010111111000;
                85:h=19'b1000000010010110000;
                86:h=19'b1000000001101010001;
                87:h=19'b1000000000111101101;
                88:h=19'b1000000000010010110;
                89:h=19'b0000000000010100110;
                90:h=19'b0000000000110111010;
                91:h=19'b0000000001010011101;
                92:h=19'b0000000001101001000;
                93:h=19'b0000000001110111010;
                94:h=19'b0000000001111110011;
                95:h=19'b0000000001111110110;
                96:h=19'b0000000001111001010;
                97:h=19'b0000000001101110111;
                98:h=19'b0000000001100000101;
                99:h=19'b0000000001010000000;
                100:h=19'b0000000000111110000;
        default:h=19'b0000000000000000000;
                endcase
                end
        endcase
        end
endmodule
```

Mul Module

```
module Mul(x, y, po);
   input [18:0] x;
   input [18:0] y;
   output [18:0] po;
//     output [35:0] p;
   reg [18:0] po;
   reg [35:0] p;
            wire [18:0] x,y;

            always@(x or y or p )
                     begin
                     if( ((x[18])&&(y[18])) |
((~x[18])&&(~y[18])))
                              begin
                              p= x[17:0]*y[17:0];
                              po={1'b0,p[34:17]};
                              end
                     else
                              begin
                              p= x[17:0]*y[17:0];
                              po={1'b1,p[34:17]};//
                              end
                     end
endmodule
```

Add Module

```
module Add(a, b, q);
   input [18:0] a;
   input [18:0] b;
   output [18:0] q;
   reg [18:0] q;
            wire [18:0] a,b;

always@(a or b )
   begin
   if(~a[18] && ~b[18])  //a.pos b.pos
            q = {1'b0,(a[17:0] + b[17:0])};

  else if(~a[18] && b[18])  //a.pos b.neg
            begin
            if(a[17:0] >= b[17:0])
                     begin
                     q = {1'b0,(a[17:0] - b[17:0])};
                     end
            else
                     q = {1'b1,(b[17:0] - a[17:0])};
            end

  else if(a[18] && ~b[18])  //a.neg b.pos
            begin
            if(a[17:0] > b[17:0])
                     begin
                     q = {1'b1,(a[17:0] - b[17:0])};
                     end
            else
                     q = {1'b0,(b[17:0] - a[17:0])};
            end

  else if(a[18] && b[18])  //a.neg b.neg
            q = {1'b1,(a[17:0] + b[17:0])};
  else
            q=19'b111_1111_1111_1111_1111;
end
endmodule
```

UnSignFixPt19 Module

```
module UnSignFixPt19(sFixPt19,twosComp);
   input [18:0] sFixPt19;
            output [15:0]twosComp;
            reg [15:0]twosComp;
            reg [23:0]mul1;
            wire [18:0] sFixPt19;

always@(sFixPt19 or mul1)
  begin
  if(sFixPt19[18])
            begin
            mul1 = 6'b110011 * sFixPt19[17:0];
            twosComp = {1'b1,~(mul1[20:6]-1)};
            end
  else
            begin
            mul1 = 6'b110011 * sFixPt19[17:0];
            twosComp = {1'b0,mul1[20:6]};
            end
  end
endmodule
```

DAC Module

```
module DAC(clk, reset,count, DACin, DACo, ldac);
   input  clk, reset;
   input [7:0] count;
            input  [15:0] DACin;
            output [15:0] DACo;
   output ldac;

            reg [15:0] DACo;//, DAC;
   reg ldac;
            wire [7:0] count;
            wire [15:0]DACin;

always@(posedge clk)
  begin
  if(~reset)
            begin
            DACo<=16'h8000;
            ldac<=0;
            end
  else
            begin
            if(count==240)
               DACo<={~DACin[15],DACin[14:0]};
            if(count==246 )
                     ldac<=0;
            if(count>=248)
                     ldac<=1;
            end
  end
endmodule
```