<u>Digital System Testing and Testable Design</u>

Project Report

“Noise Reduction in Electrocardiography using Low Pass Finite Impulse Filter“

Submitted by

Amar Vignesh S

EVD17I003

Department of Electronics and Communication Engineering

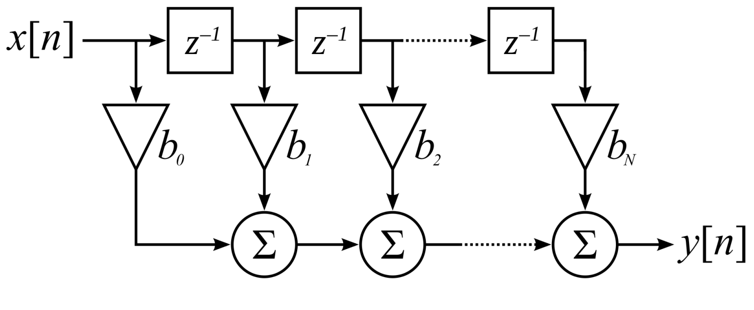
Indian Institute of Information Technology Design and Manufacturing, Kancheepuram

Introduction

Electrocardiogram (ECG) signal is a bio-electrical activity of the heart. It is a common routine and important cardiac diagnostic tool where in electrical signals are measured and recorded to know the functional status of heart, but ECG signal can be distorted with noise as, various artifacts corrupt the original ECG signal and reduces it quality Therefore, there is a need to remove such artifacts from the original signal and improve its quality for better interpretation. Digital filters are used to remove noise error from the low frequency ECG signal and improve the accuracy of the signal. Noise can be any interference due to motion artifacts or due to power equipment that are present where ECG has been taken. Thus, ECG signal processing has become a prevalent and effective tool for research and clinical practices.Low pass Filter can be used to remove such noise when the ecg signal has high frequency noise.

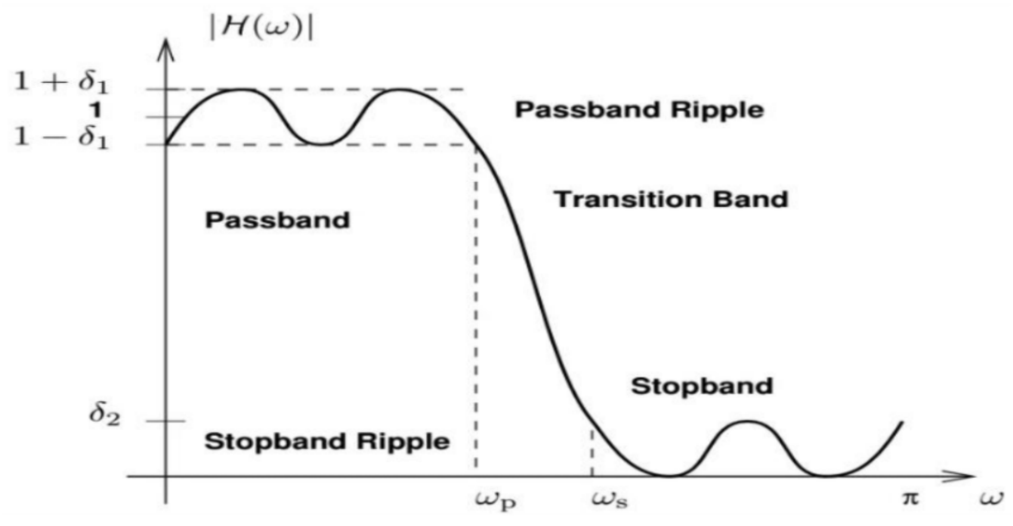
FIR Filter

A finite impulse response (FIR) filter is a filter whose impulse response (or response to any finite length input) is of finite duration.A FIR filter has a number of useful properties which sometimes make it preferable to an infinite impulse response (IIR) filter. FIR filters Require no feedback ,are inherently stable since the output is a sum of a finite number of finite multiples of the input values and can easily be designed to be linear phase by making the coefficient sequence symmetric.

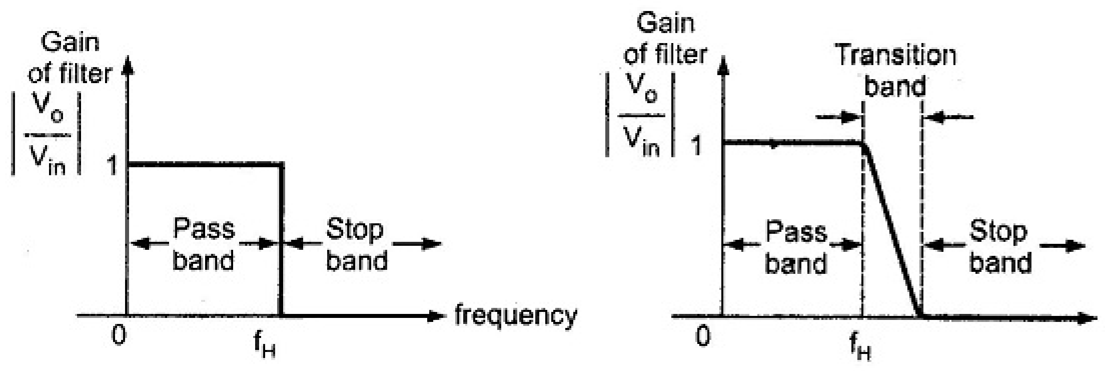


<u>Low Pass Filter</u>

A Low Pass Filter is a circuit that can be designed to modify, reshape or reject all unwanted high frequencies of an electrical signal and accept or pass only those signals wanted by the circuits designer.The exact frequency response of the filter depends on the filter design.



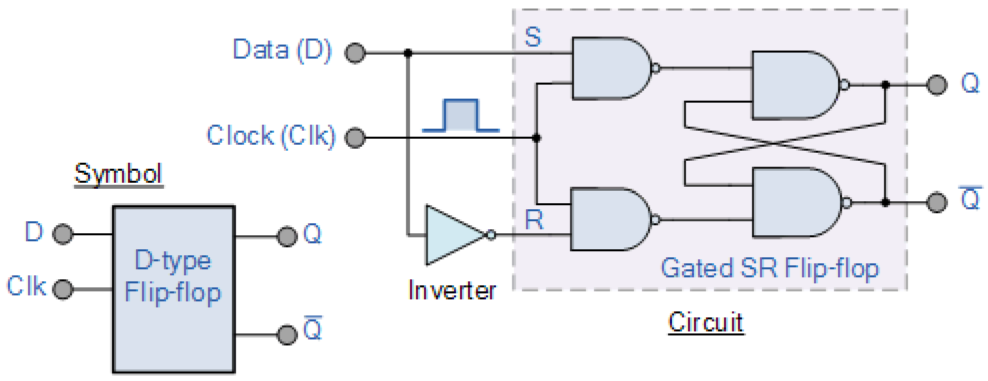
Frequency response of a practical lowpass filter

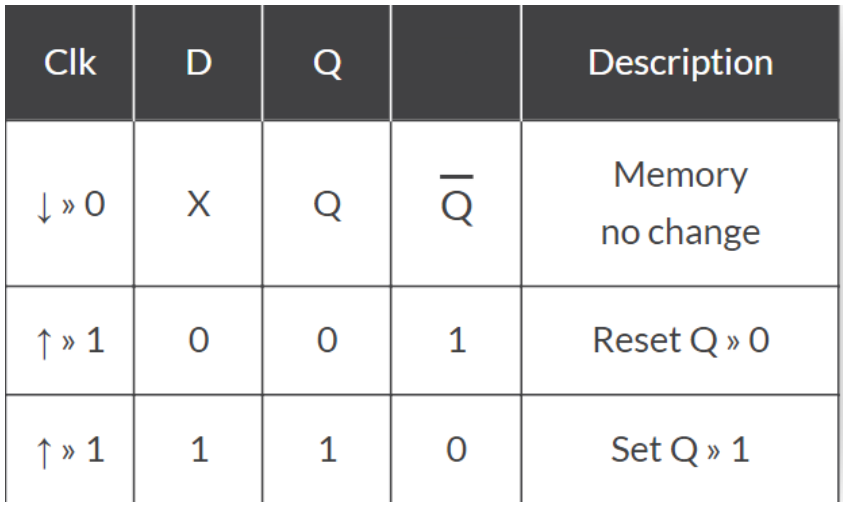


(a) Ideal **Fig. 2.71 Low pass filter** **(b) Practical**

<u>Delay</u>

Delay is implemented using D Flip Flop

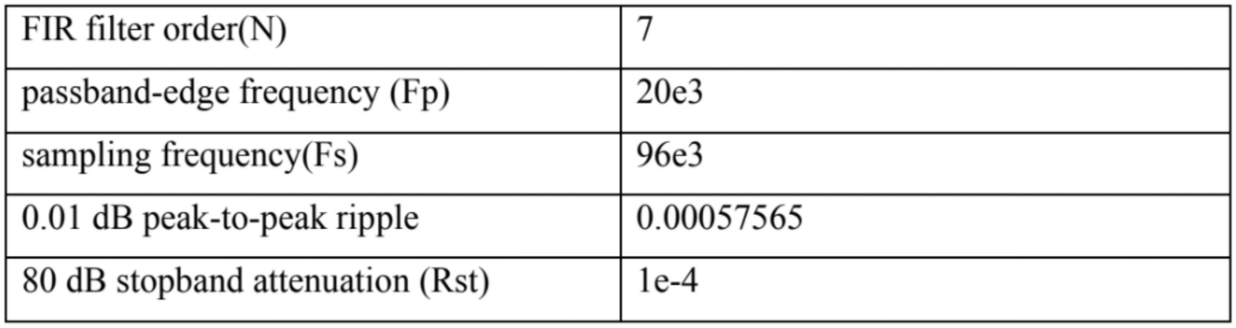




Clk	D	Q		Description
↓ » 0	X	Q	$\overline{Q}$	Memory no change
↑ » 1	0	0	1	Reset Q » 0
↑ » 1	1	1	0	Set Q » 1

8 Tap Filter

Coefficients for 8 tap filter used for removing noise from 100 16bit ecg samples are obtained using firceqrip(n,Fo,DEV) function which designs an order n filter (filter length equal n + 1) lowpass FIR filter with linear phase.The input argument Fo specifies the frequency at the upper edge of the passband in normalized frequency (0<Fo<1). The two-element vector dev specifies the peak or maximum error allowed in the passband and stopband. [d1 d2] are entered for dev where d1 sets the passband error and d2 sets the stopband error.

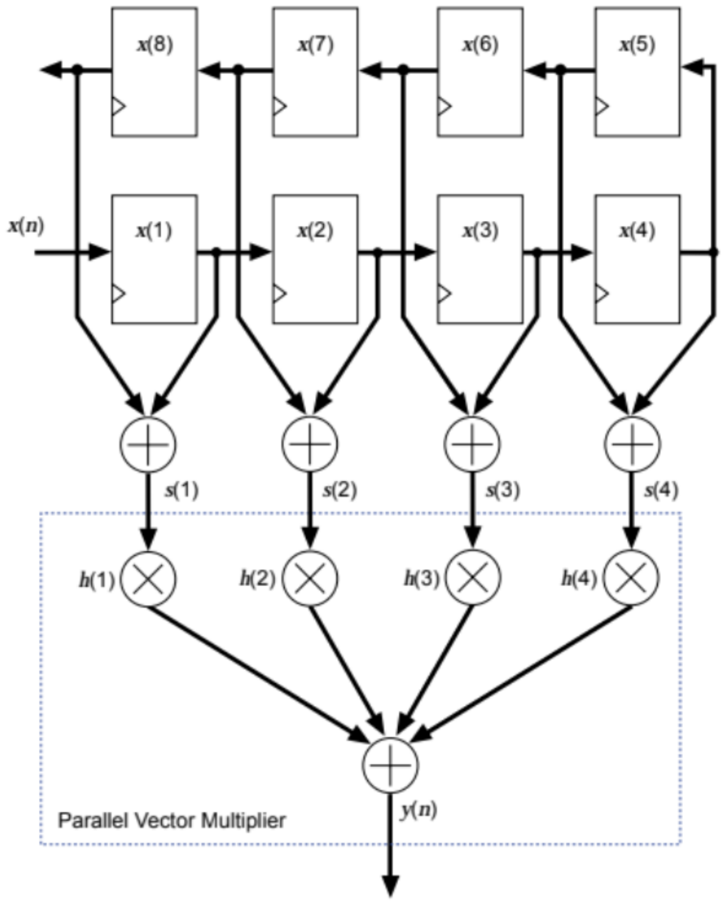


FIR filter order(N)	7
passband-edge frequency (Fp)	20e3
sampling frequency(Fs)	96e3
0.01 dB peak-to-peak ripple	0.00057565
80 dB stopband attenuation (Rst)	1e-4

```
eqnum = firceqrip(N,Fp/(Fs/2),[Rp Rst],'passedge')
```

For a linear phase response FIR filter, the coefficients are symmetric around the center values. This symmetry allows the symmetric taps to be added together before they are multiplied by the coefficients. See Figure 2. Taking advantage of the symmetry lowers the number of multiplies from eight to four, which reduces the circuitry required to implement the filter.

Figure 2. Adding Symmetric Taps Before Multiplication



MatLab is used to generate the ecg samples using the inbuilt function present in the signal processing toolbox of Matlab.

HO is the coefficient multiplied with the current sample ,H1 by sample delayed by one unit,H2 by sample delayed by two units and so on.

Each input sample is multiplied by the all the coefficients and stored in a register

```
MCM3 = H3*Xin;
MCM2 = H2*Xin;
MCM1 = H1*Xin;
MCM0 = H0*Xin;
```

Delay is implemented using D FlipFlop

```
task DFLIP;
input signed [31:0]D;
output reg signed [31:0]Q;
Q = D;
```

Flipflop instantiations

```
DFF1 dff1 (Clk,MCM3,Q1);
DFF1 dff2 (Clk,add_out1,Q2);
DFF1 dff3 (Clk,add_out2,Q3);
```

All the samples are multiplied with corresponding coefficients and added together to get the output.

```
add_out1 = Q1 + MCM2;
add_out2 = Q2 + MCM1;
add_out3 = Q3 + MCM0;
```

ECG Signal Analysis

ECG signal is obtained from Matlab ecg() inbuilt function from the signal processing toolbox (add-on) .

The samples are multiplied by 10000 to avoid floating point operations in system verilog.

```
ecg_signal = 10000*ecg(1000);
```

Coefficients were obtained using firrcos function from the same signal processing toolbox.

b= firrcos(n,F0,df,fs,'bandwidth') returns an order n low pass linear phase FIR filter with a raised cosine transition band.

The filter has cutoff frequency F0, transition bandwidth df and sampling frequency fs, all in hertz.df must be small enough so that F0 ± df/2 is between 0 and fs/2.

The coefficients in b are normalized so that the nominal passband gain is always equal to 1.
Specify fs as the empty vector [] to use the default value fs = 2.

cutoff frequency, corner frequency, or break frequency is a boundary in a system's frequency response at which energy flowing through the system begins to be reduced (attenuated or reflected) rather than passing through.

The transition band is a range of frequencies that allows a transition between a passband and a stopband of a signal processing filter. The transition band is defined by a passband and a stopband cutoff frequency or corner frequency.

```
coeff=floor(10000*firrcos(101,15,10,1000))
binary_ecg=dec2bin(typecast(int16(ecg_signal),'uint16'))
binary_coeff=dec2bin(typecast(int16(coeff),'uint16'))
```

Samples are read in from the .txt file and processed and the graph is plotted back in matlab to display the output .

Code:

MatLab code for generating ecg signals and storing in .txt file

```
clc;

close all;

clear all;

ecg_signal=10000*ecg(1000);

coeff=floor(10000*firrcos(100,15,10,1000))

binary_ecg=dec2bin(typecast(int16(ecg_signal),'uint16'))

binary_coeff=dec2bin(typecast(int16(coeff),'uint16'))

%reading array into .txt

fileID1 = fopen('D:\FIR\binary_ecg.txt','w');
```

```
fprintf(fileID1,'%s \n',string(binary_ecg));

fileID2 = fopen('D:\FIR\binary_coeff.txt','w');

fprintf(fileID2,'%s \n',string(binary_coeff));

fclose(fileID1);

fclose(fileID2);
```

Main Module

```
module fir_8tap(input Clk,input signed [15:0]Xin,output reg signed [31:0] Yout);

//Internal variables.

reg signed[15:0]H[100:0];

reg signed [31:0]MCM[100:0];

reg signed [31:0]Q[99:0],add_out[99:0];

integer i;

//flipflop instantiations (for introducing a delay).

task DFLIP;

input signed [31:0]D;

output reg signed [31:0]Q;

Q = D;

endtask

//reading coefficients from binary_coeff.txt file to H vector

initial
```

```
$readmemb("D:/FIR/binary_coeff.txt", H);

always @(posedge Clk)

begin

for (i=100;i>=0;i=i-1)

MCM[i] = H[i]*Xin;

for (i=0;i<100;i=i+1)

add_out[i]=Q[i]+MCM[100-(i+1)];

DFLIP(MCM[100],Q[0]);

for (i=0;i<99;i=i+1)

DFLIP(add_out[i],Q[i+1]);

Yout = add_out[99];

$display("%d",Yout);

$writememb("D:/FIR/binary_out.txt",Yout);

end

endmodule
```

Testbench

```
//Testbench for the low pass FIR filter:

module tb1;

reg [15:0]a[999:0];

integer i;
```

```
// Inputs

reg Clk;

reg [15:0] Xin;

// Outputs

wire [31:0] Yout;

// Instantiate the Unit Under Test (UUT)

fir_8tap LPF (Clk,Xin,Yout);

//Generate a clock with 10 ns clock period.

initial Clk = 0;

always #5 Clk =~Clk;

// reading ecg signals from binary.txt file ,here 100 samples of ecg signals are read

initial

begin

$readmemb("D:/FIR/binary_ecg.txt", a);

for (i=0; i<1000; i=i+1)

begin

Xin=a[i];#10;

end

end

initial
```

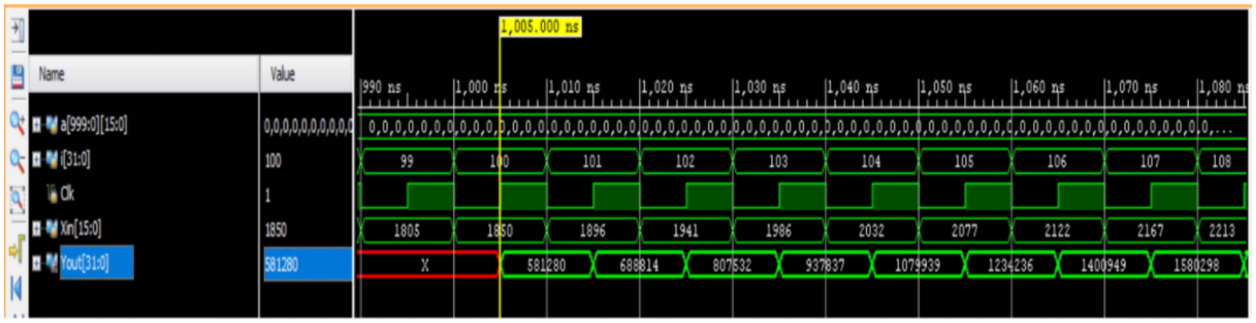
```
#10000 $stop;

endmodule
```

MatLab code for displaying output

```
clc;

clear all;

close all;

file0=fopen('D:\FIR\binary_ecg.txt','r')

form='%d'

x0=fscanf(file0,form)

x=x0./10000

fclose(file0);

file1=fopen('D:\FIR\binary_out.txt','r')

form='%d'

y0=fscanf(file1,form)

y=y0./10^8

fclose(file1);

subplot(2,1,1);plot(abs(fft(x)));

title("ecg signal,1000 samples")

xlabel("freq")

ylabel("DFT of ecg signal")
```

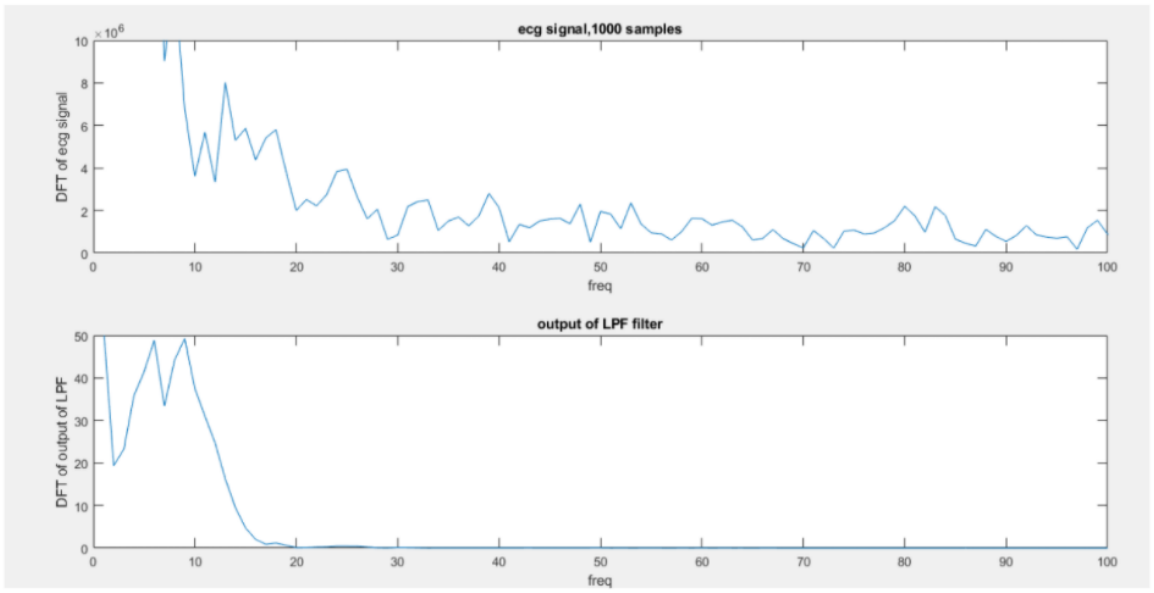
```
axis([0 100 0 10^7])

subplot(2,1,2);plot(abs(fft(y)));

title("output of LPF filter")

axis([0 100 0 6000])

xlabel("freq")

ylabel("DFT of output of LPF")
```

Results



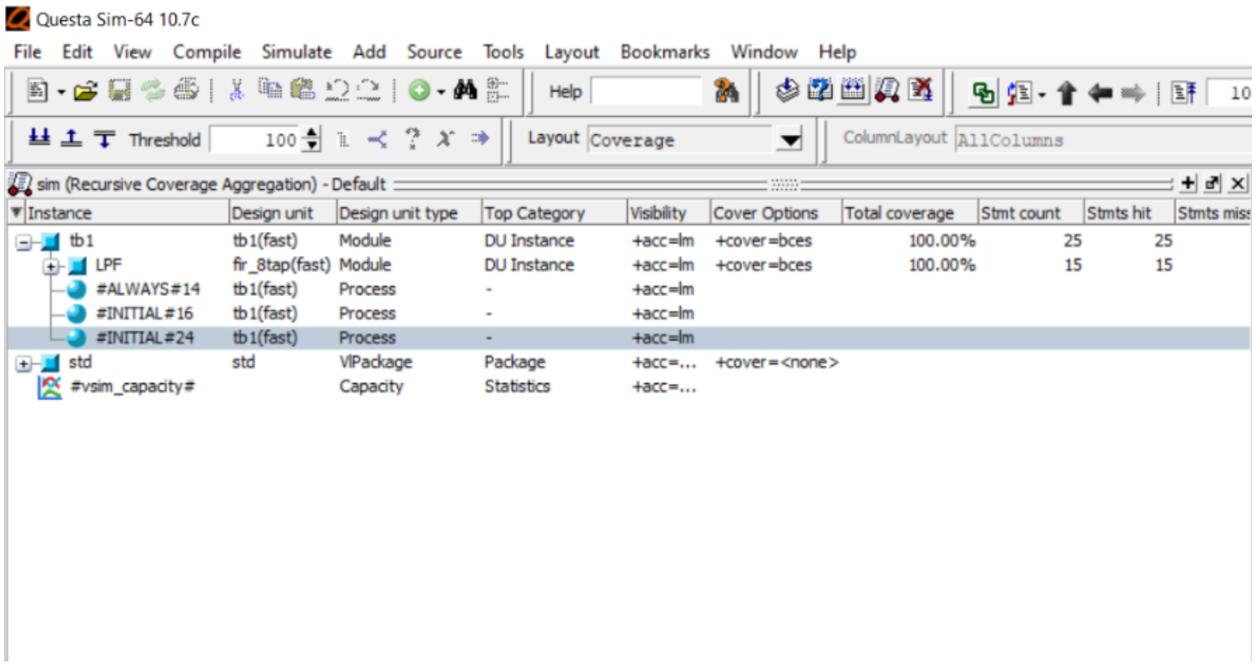
Output Waveform

DFT of ECG signal before and after passing from Low pass filter FIR Filter



In the output picture it can be seen that noise(above 20 hertz) is completely removed

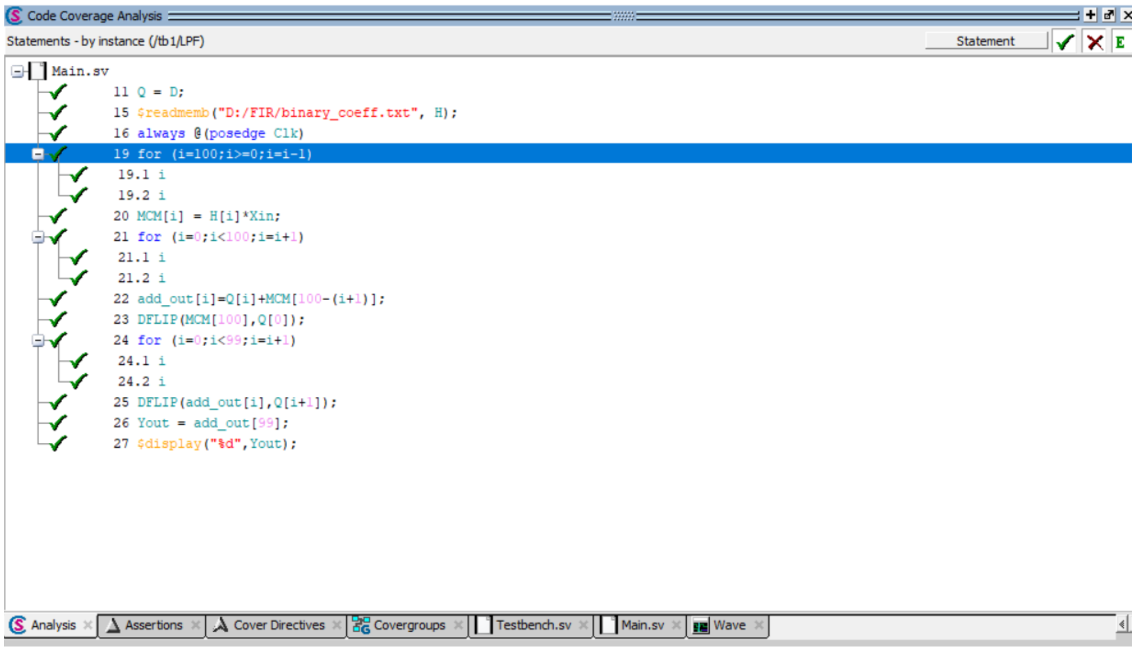
Code Coverage Analysis


Questa Sim-64 10.7c
File Edit View Compile Simulate Add Source Tools Layout Bookmarks Window Help
Help
Threshold 100
Layout Coverage
ColumnLayout AllColumns

sim (Recursive Coverage Aggregation) - Default

Instance	Design unit	Design unit type	Top Category	Visibility	Cover Options	Total coverage	Stmt count	Stmts hit	Stmts miss
tb1	tb1(fast)	Module	DU Instance	+acc=lm	+cover=bces	100.00%	25	25	
LPF	fir_8tap(fast)	Module	DU Instance	+acc=lm	+cover=bces	100.00%	15	15	
#ALWAYS#14	tb1(fast)	Process	-	+acc=lm					
#INITIAL#16	tb1(fast)	Process	-	+acc=lm					
#INITIAL#24	tb1(fast)	Process	-	+acc=lm					
std	std	VIPackage	Package	+acc=...	+cover=<none>				
#vsim_capacity#		Capacity	Statistics	+acc=...					

Code Coverage analysis of Main Module

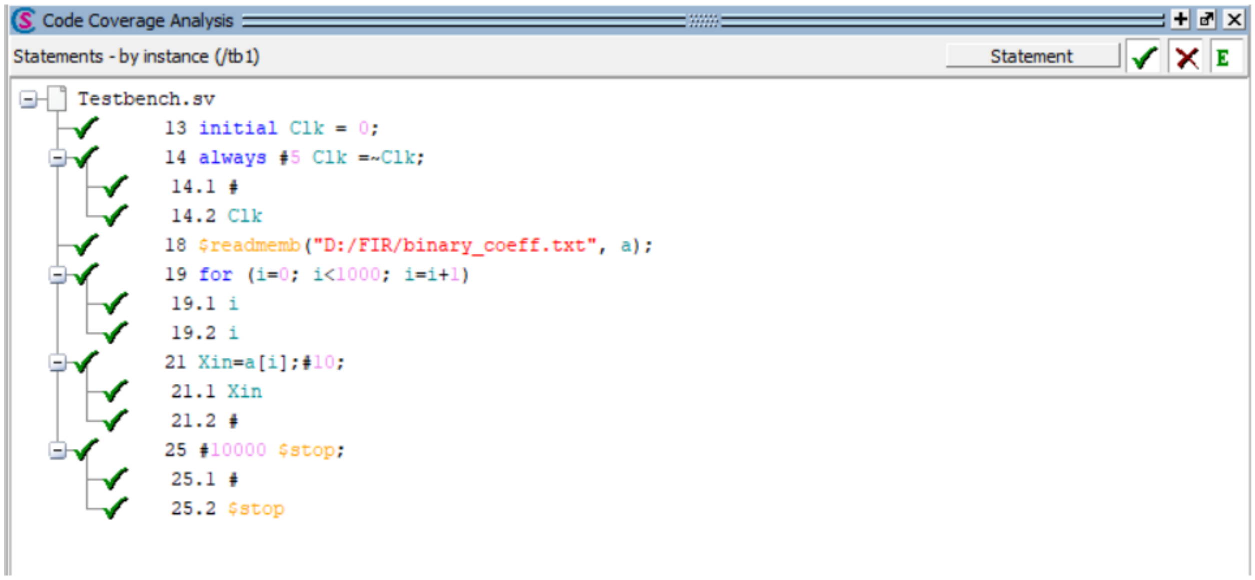


Code Coverage Analysis

Statements - by instance (/tb1/LPF)

```
Main.sv
 ✓ 11 Q = D;
 ✓ 15 $readmemb("D:/FIR/binary_coeff.txt", H);
 ✓ 16 always @(posedge Clk)
 ✓ 19 for (i=100;i>=0;i=i-1)
   ✓ 19.1 i
   ✓ 19.2 i
 ✓ 20 MCM[i] = H[i]*Xin;
 ✓ 21 for (i=0;i<100;i=i+1)
   ✓ 21.1 i
   ✓ 21.2 i
 ✓ 22 add_out[i]=Q[i]+MCM[100-(i+1)];
 ✓ 23 DFLIP(MCM[100],Q[0]);
 ✓ 24 for (i=0;i<99;i=i+1)
   ✓ 24.1 i
   ✓ 24.2 i
 ✓ 25 DFLIP(add_out[i],Q[i+1]);
 ✓ 26 Yout = add_out[99];
 ✓ 27 $display("%d",Yout);
```

Code Coverage Analysis of Testbench



Code Coverage Analysis

Statements - by instance (/tb1)

```
Testbench.sv
 ✓ 13 initial Clk = 0;
 ✓ 14 always #5 Clk =~Clk;
   ✓ 14.1 #
   ✓ 14.2 Clk
 ✓ 18 $readmemb("D:/FIR/binary_coeff.txt", a);
 ✓ 19 for (i=0; i<1000; i=i+1)
   ✓ 19.1 i
   ✓ 19.2 i
 ✓ 21 Xin=a[i];#10;
   ✓ 21.1 Xin
   ✓ 21.2 #
 ✓ 25 #10000 $stop;
   ✓ 25.1 #
   ✓ 25.2 $stop
```

References

Implementing FIR Filters in FLEX Devices by ALTERA

Denoising of ECG Signals Using FIR & IIR Filter: A Performance Analysis
By Ms. Chhavi Saxena1 , Mr. Vivek Upadhyaya2 , Dr. Hemant Kumar Gupta3,
Dr. Avinash Sharma4

Signal Processing Techniques for Removing Noise from ECG Signals by Rahul Kher*

Conclusion

Implemented Low pass filter for ecg signal which was generated using matlab function and high frequency unwanted components of the real time signal was eliminated using the low pass filter by changing order and type of filter required.