

4C1 Integrated Systems Design

REPORT: Lab-3 Part-2 FIR Filter Implementation

-Pulkit Agarwal (19323939)

.....

Configuring the FIR:

On setting the input sampling frequency and clock frequency to 100 MHz, which is same as the Basys3 Board rate, the following observations were made:

1. Keeping clock frequency as constant, the input sampling Frequency is proportional to the number of DSP Slices.
2. Keeping Input Sampling Frequency as constant, the Clock Frequency is inversely proportional to the number of DSP Slices.
3. In cases where Input Sampling Frequency is greater than the Clock Frequency, the DSP slices do not apply.

This brings us to the conclusion that the number of DSP Slices do not affect the performance and is dominantly dependent on the Sampling Frequency.

Now, we need to configure the quantisation to meet the spectral requirements of the filter with minimum possible resources. For this, I had to reduce the cost and the area by minimising the number of bits that hold the filter coefficients.

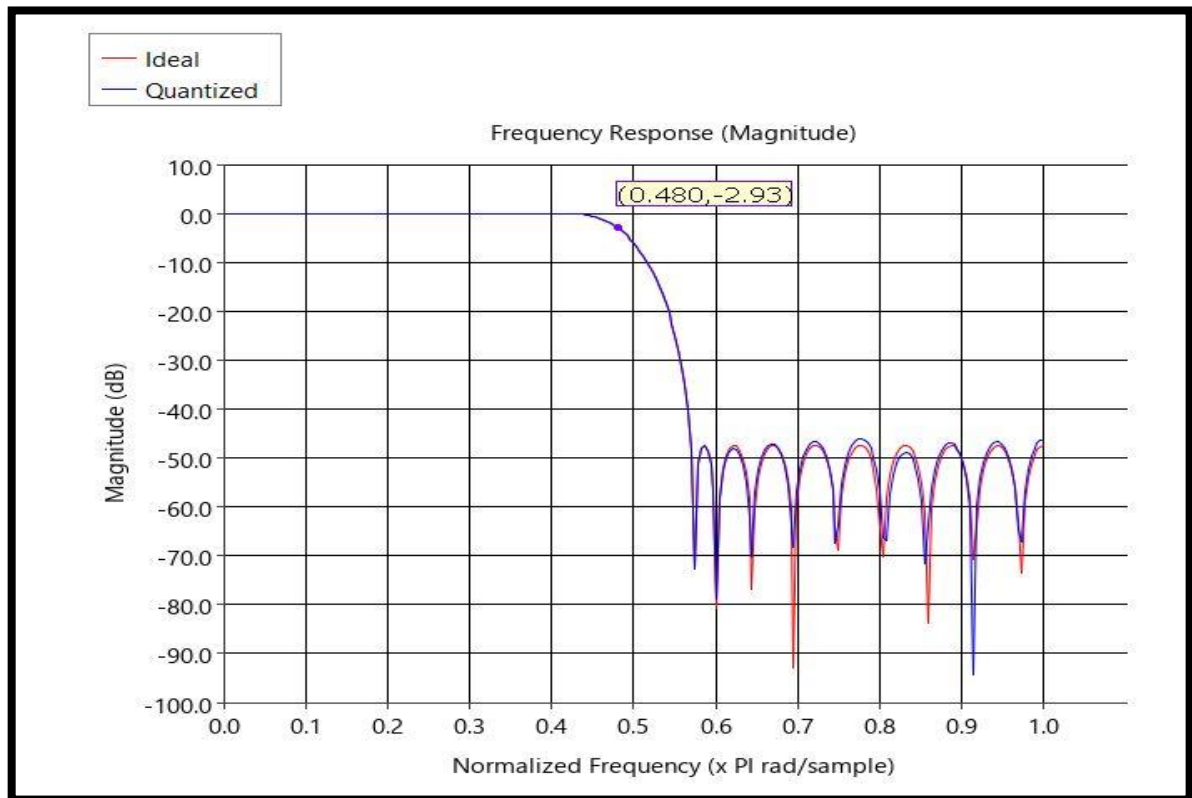
It was observed that the number of DSP Slices was directly proportional to the Width Bit because fewer bits required smaller hamming weights which in turn required less taps.

Also, for width less than 9 bit seems to be under-quantized. This was because there was a drastic change in the spectral performance in this range.

On the other hand, for width greater than 15 bits, they seemed to be over-quantized.

Therefore, the appropriate width bit ranged from 9 to 13 bits seemed to be appropriately quantized. It also had less impact on the frequency response as compared to the ideal response.

Displayed below is the Frequency Response Graph where I used 13 bit width for the filter coefficients.



Simulation and Testing the System:

For simulation and testing, I created a task "*fcw_values*" to calculate the Fcw values at different frequencies.

I tried taking Fcw on the basis that they lied in all the regions of the Frequency Response Graph. They were as follows:

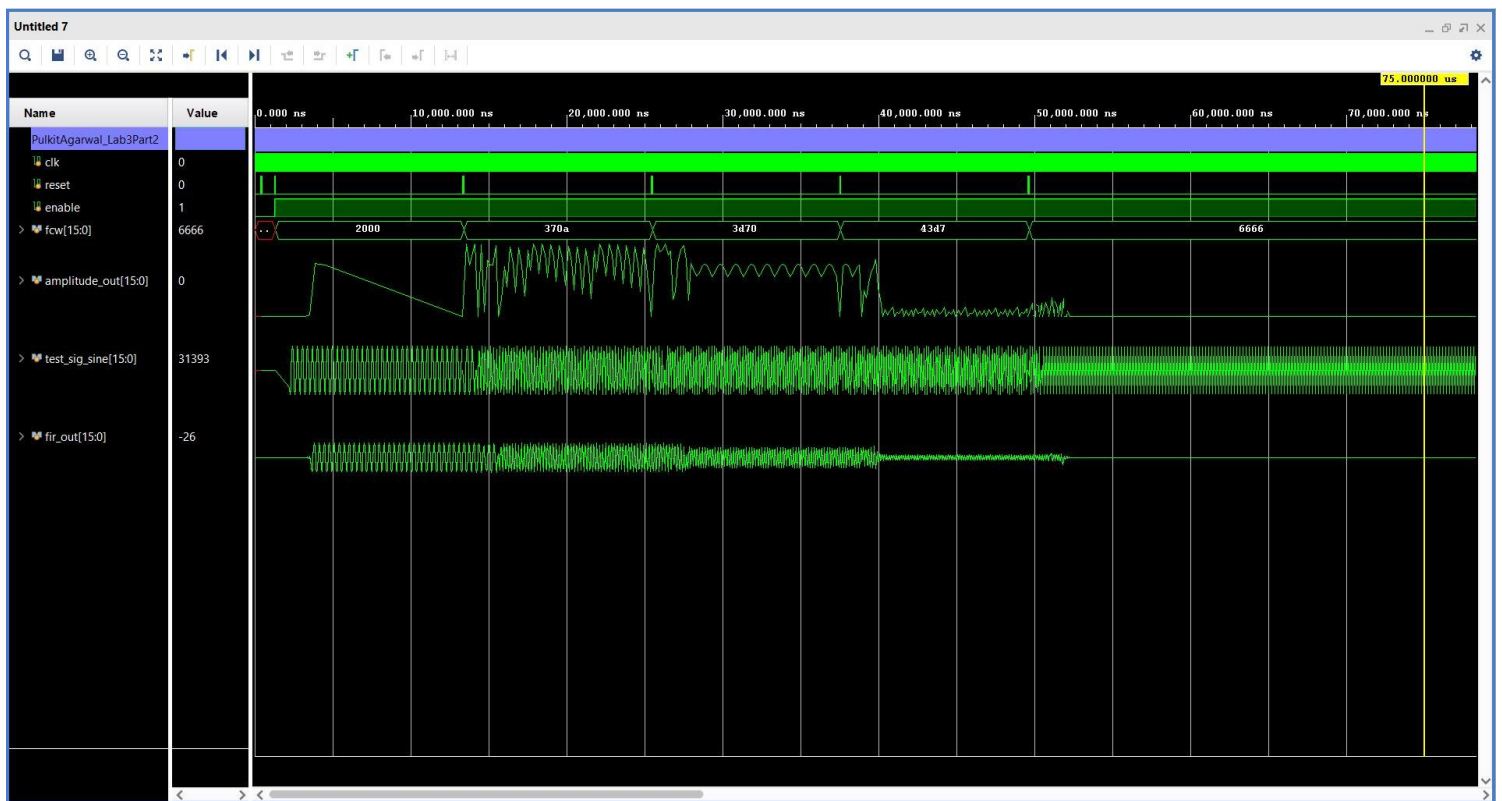
1. In the passband range 0 to 0.43.
2. At the passband cut-off frequency 0.43.
3. In the transition band 0.43 to 0.53.
4. At the stopband cut-off frequency 0.53.
5. In the stopband range 0.53 to 1.

The following formula was used to calculate the Fcw values for the frequency response:

$$\mathbf{Fcw = value*0.5*100*655.36}$$

(where $0 < \text{value} < 1$)

Displayed below is the simulation graph showing Tested Sine Signal and FIR Output Signal:



On comparing the tested sine signal and the FIR output signal from the waveform above, it can be clearly seen that both these waveforms are divided into five sections. These five sections (passband, passband cut-off, transition band, stopband cut-off, stopband) are all the regions described above.

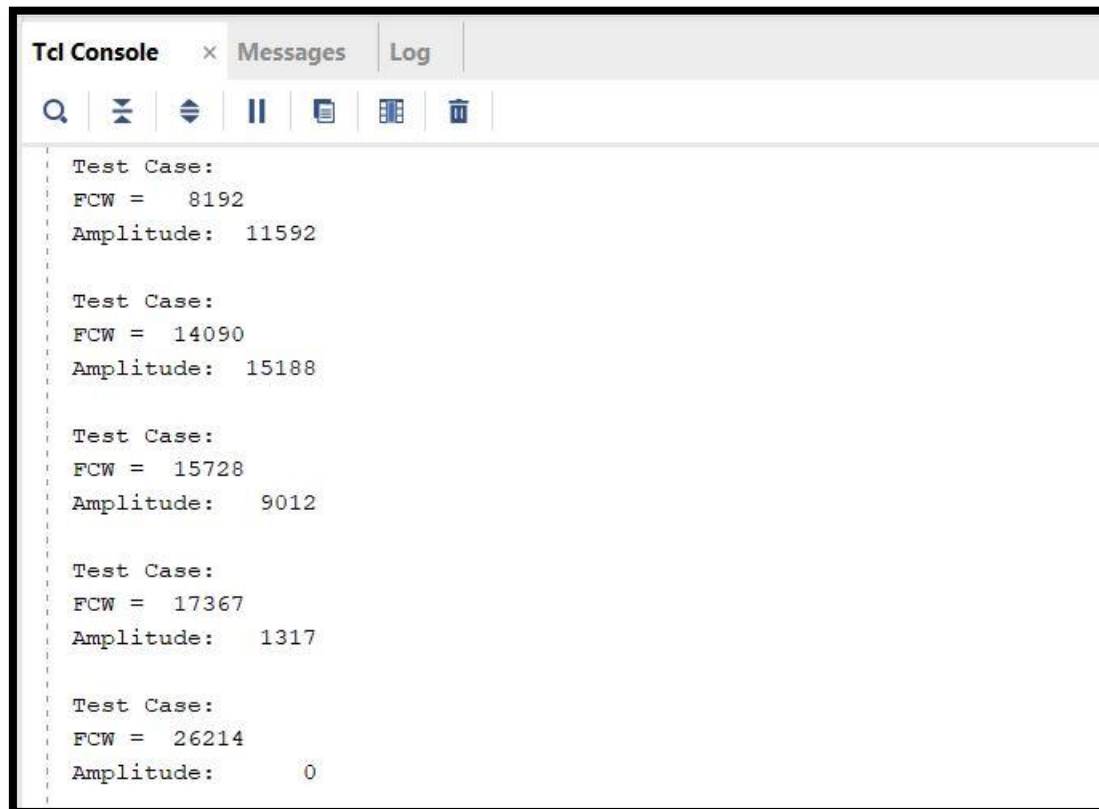
It can be observed that the signal is filtering in these parts as expected i.e. from no filtering to almost complete filtering as we proceed.

Also, the amplitude for the output signal is calculated using the formula:

$$20 \log_{10} V_{peak} / V_{max}$$

It has also been displayed in the above displayed waveform.

Here is a snippet of the TCL Console calculating the output amplitude and the expected frequency response.



```
Tcl Console x Messages Log
[Search] [Zoom In] [Zoom Out] [Run] [Copy] [Paste] [Delete]

Test Case:
FCW = 8192
Amplitude: 11592

Test Case:
FCW = 14090
Amplitude: 15188

Test Case:
FCW = 15728
Amplitude: 9012

Test Case:
FCW = 17367
Amplitude: 1317

Test Case:
FCW = 26214
Amplitude: 0
```

The calculations confirm that the implemented FIR is dishing out the desired performance.

Synthesis and Implementation:

To get realistic timing report data, an XDC file was added defining my clock frequency. Here is a code snippet confirming the same:

```
## Clock signal
set_property PACKAGE_PIN W5 [get_ports clk]
set_property IOSTANDARD LVCMOS33 [get_ports clk]
create_clock -add -name sys_clk_pin -period 10.00 -waveform {0 5} [get_ports clk]
```

Furthermore, here is a snippet of the WNS (Worse Negative Slack) from the project summary:

Timing	Setup	Hold	Pulse Width
Worst Negative Slack (WNS):	3.01 ns		
Total Negative Slack (TNS):	0 ns		
Number of Failing Endpoints:	0		
Total Number of Endpoints:	3773		
Implemented Timing Report			

The WNS is 3.01ns with the configuration.

To calculate **Fmax**:

$$Fmax = \frac{1}{(Tclk - WNS)}$$

Therefore,

$$Fmax = \frac{1}{(10 - 3.01ns)}$$

$Fmax = 143 \text{ MHz}$

To improve the Fmax value, the following strategies could be kept in mind when implementing a FIR:

1. Parallelism could be used. This helps increase the throughput by duplicating functions to operate different signals at once. Hence increasing the Fmax value.
2. Pipelining could also be used to increase the Fmax value as this method allows different functions to run concurrently.