Digital System Design

we gather, we share, you learn..

www.digitalsystemdesign.in

Parallel and Pipeline Implementations of IIR Low Pass Filter on FPGA

Shirshendu Roy

May 9, 2020

Abstract

Majority of digital filters implemented in the digital systems are Finite Impulse Response (FIR) filters. Infinite Impulse Response (IIR) filters can produce same frequency response but with less co-efficients and delay elements compared to FIR filters. But use of IIR filters is limited to the low frequency applications. This is due to the fact that IIR filters have longer critical path due to their recursive nature. In this work, various parallel structures of IIR filters are implemented on Field Programmable Gate Array (FPGA) device. Also, pipeline implementation of IIR filters is investigated. All these implementations are implemented by taking an example of low pass filter. FPGA implementation performance of all the structures are compared. Root Mean Squared Error (RMSE) is used to estimate the performance.

Keywords: Field Programmable Gate Array (FPGA), Infinite Impulse Response (IIR) filters, Direct Form I, Direct Form II, Cascaded Structure, Parallel Structure, Look-ahead algorithm.

1 Introduction

There are two type of digital filters used in the signal processing applications which are Finite Impulse Response (FIR) filters and Infinite Impulse Response (IIR) filters. Majority of filters implemented in the digital domain are FIR filters due to their ease of implementation. On the other hand, IIR filters can achieve same frequency response with less number of co-efficients and delay elements. Also IIR filters are suitable to achieve narrow bandwidth.

Field Programmable Gate Array (FPGA) is a very popular platform to implement the digital filters for real time applications. Recent developments in the FPGA technology provides highly efficient DSP blocks [1] to implement digital filters. FIR filters can easily work at high frequencies with the help of these DSP blocks. On the other hand, use of IIR filters is limited. IIR filters are generally used at low frequencies. This due to the fact that the IIR filters are recursive means output is reused as input. The IIR filters have long critical path due the recursive property.

Earlier in [2], we have discussed about the FPGA implementation of the FIR filters. In this work, different parallel FPGA implementation of the IIR filters is discussed. The pipeline implementation of IIR filters is investigated here. A simple low pass filter is taken as example and this filter is implemented with every type of structures. A comparison of the performances of these structures is carried out in this work.

The manuscript is organized as follows: The section 2 describes the basic IIR low pass

filter. In the section 3, FPGA implementation different parallel structures are described. In the section 4, the pipeline implementation of the IIR filters is investigated. Performance of the implementation of the different structures is estimated in section 5. In the section 6, conclusions of on the finding of this work are made.

2 IIR Low Pass Filter

In this work, design of a low pass filter is taken as an example to illustrate the implementation of IIR filter. The frequency response of the low pass filter is specified as

$$\alpha_p = 0.01\,dB \quad \text{for}\, 0 \leq w \leq \frac{\pi}{3} \tag{1}$$

$$\alpha_s = 60\,dB \quad \text{for}\, \frac{\pi}{3} \leq w \leq \frac{\pi}{2} \tag{2}$$

Here, w is the normalized frequency. The parameter α_p and the α_s denotes the pass band and stop band attenuation. This low pass IIR filter can be realized using many techniques as illustrated in [3]. Here elliptical based design is followed.

The transfer function $H(z) = y(z)/x(z)$ of an IIR filter can be written as

$$H(z) = \frac{\sum_{n=0}^{M} b_n z^{-n}}{1 - \sum_{n=1}^{N} a_n z^{-n}} \tag{3}$$

Here, b_n is the co-efficients of the all zero transfer function and a_n denotes the co-efficient of the all pole transfer function. Here, $N = M = 6$ for the given low pass filter. The frequency plot of the low pass IIR filter is shown in Figure 1.

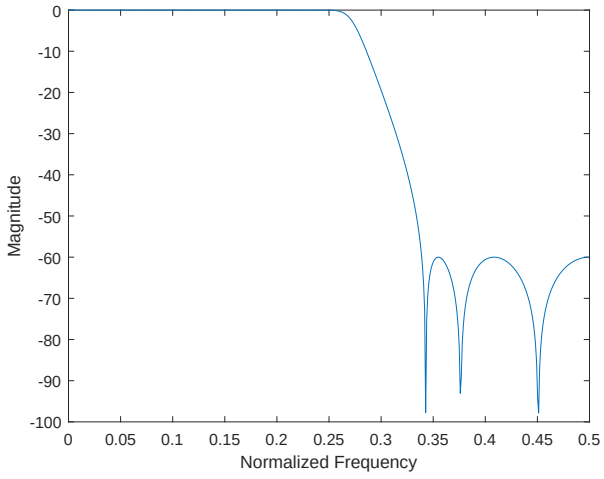
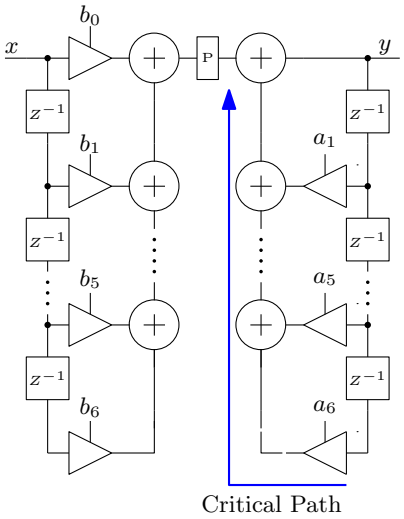


Figure 1: Frequency Spectrum of the Low Pass Filter.

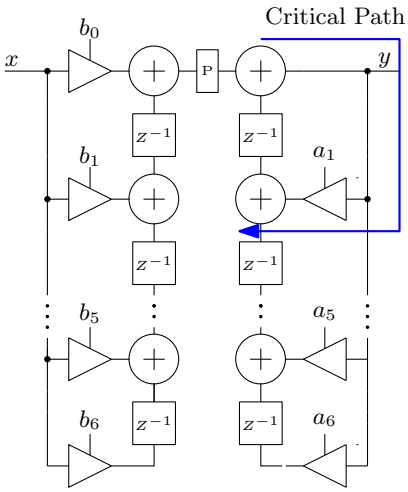
3 Different IIR Filter Structures

In literature, many different structures [3, 4] to implement the IIR filters are reported and this structures are

1. Direct Form I Structures
2. Direct Form II Structures



(a) Direct form I structure of IIR filters.



(b) Transposed structure of the direct form I.

Figure 2: Direct form I structures for IIR filters.

3. Cascaded Structures

4. Parallel Structures

5. Lattice Structures

In this work, a low pass filter is implemented using Direct Form I, Direct Form II, Cascaded Form and Parallel Form. Lattice structures are extensively used in speech processing and are costly. Thus it is not discussed here but other structures are discussed in details with their FPGA implementation.

3.1 Direct Form I Structures

The time difference equation of the IIR filter can be expressed as

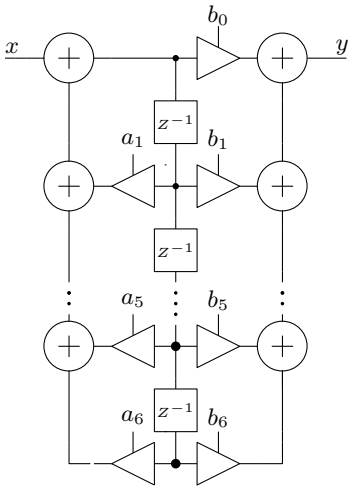
$$y(n) = \sum_{k=1}^{N} a_n y(n-k) + \sum_{n=0}^{M} b_n x(n-k) \tag{4}$$

The direct form I structures directly implements the IIR filters. The basic direct form I structure is shown in Figure 2(a). The implementation of this kind of architecture is not suitable for implementation as this structure has very high critical path in the all pole transfer function section or in the recursive section. If the pipeline registers are added, the result will be changed as inclusion of extra registers will implement a different filter. An alternative direct form I architecture is shown in Figure 2(b) which is called transposed direct form I structure. Here the critical path is reduced.

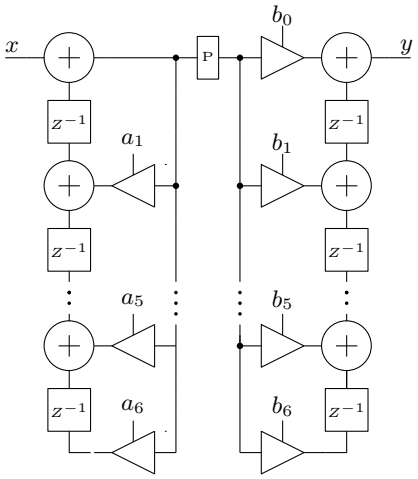
3.2 Direct Form II Structures

The direct form II structures, implements the difference function based on the fact that the filter transfer function can be represented as

$$H(z) = \frac{y(z)}{w(z)} . \frac{w(z)}{x(z)} \tag{5}$$



(a) Direct form II structure of IIR filters.



(b) Transposed structure of the direct form II.

Figure 3: Direct form II structures for IIR filters.

The basic direct form II structure is shown in Figure 3(a). This structure uses less number of delay elements compared to the direct form II structure but has the same critical path. But this structure is also not suitable for the high frequency applications. The transposed architecture is shown in Figure 3(b). This architecture reduces the critical path. All though transposed structures for direct form I and direct form II are same, direct form II structures are not preferred. In direct form II structures, a high gain all pole network is followed by a all zero network. This increases the size of the adders and multipliers.

3.3 Cascaded Form

An alternative way to implement the IIR filters is to break the transfer function into smaller transfer functions and cascading them. A IIR filter transfer function can be written as

$$H(z) = H_1(z).H_2(z)...H_k(z) \tag{6}$$

Here, $H(z)$ is written in terms of k smaller transfer functions. These smaller transfer functions can be 1st order IIR transfer function or second order IIR transfer function. The second order IIR section is popularly known as Biquad structure. The Biquad transfer function is

$$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2}}{1 - a_1 z^{-1} - a_2 z^{-2}} \tag{7}$$

The Biquad structure for the low pass filter considered here is shown in Figure 4. The Biquad structures can be implemented using either direct form I or direct form II structure. The cascaded structure of the IIR low pass filter is shown in Figure 5.

3.4 Parallel Forms

The parallel forms of IIR filters gained more popularity than the cascaded forms as parallel forms provide immunity to the co-efficient quantization and also provide parallelism in the design. In the parallel form, the IIR transfer function is written as summation of 1st order or second order sections. This is usually achieved using the partial fraction procedure. Generally Biquad structures are preferred as individual sections. The IIR low pass filter transfer function considered here can be written as

$$H(z) = FIR + \sum_{i=1}^{L} \frac{d_{i1} + d_{i2} z^{-1}}{1 - a_{i1} z^{-1} - a_{i2} z^{-2}} \tag{8}$$

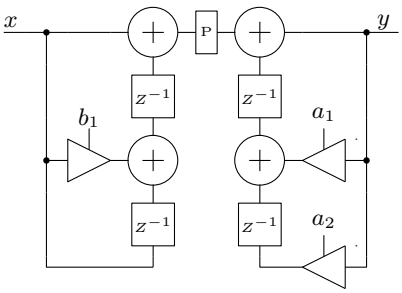


Figure 4: Architecture for the Biquad block for cascaded IIR filter.

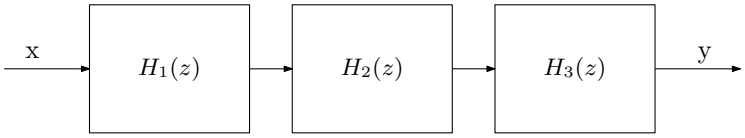


Figure 5: Cascaded structure of the IIR LPF.

Here the FIR is a simple constant and L denotes the number of parallel sections. The above equation is famously known as non-delayed architecture [5]. The delayed input signal is some times provided to the all pole section when the order of numerator is greater than the order of denominator. This form is known as delayed parallel structure [6]. In this case, FIR part can be a FIR like transfer function. The second order section for the parallel structure of the low pass filter considered here is shown in Figure 6. The non-delayed and delayed parallel structure is shown in Figure 7 and 8 respectively.

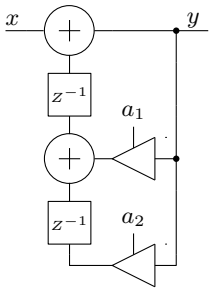


Figure 6: Architecture for the Biquad block for Cascaded IIR filter.

4 Pipeline Implementation of IIR Filters

In the previous section, different structures of IIR filters are discussed. The direct forms directly implements the IIR difference equation. The direct form structures are not generally suitable for high frequency applications due to their long critical path. The transposed architectures remove this drawback of the direct forms. The longer critical path is still a concern. Insertion of pipeline registers is not possible as it will implement any other filter. Thus special algorithms are reported to implement pipeline IIR filters.

Let's consider the case of a simple 1st order filter. The transfer function of the 1st order filter is

$$H(z) = \frac{1}{1 - az^{-1}} \tag{9}$$

This filter can be implemented in two ways. One is direct form and another is transposed form. These structures are shown in Figure 9. In both the cases, the maximum frequency is limited by the combinational delay of an adder and a multiplier. This delay can not be reduced by

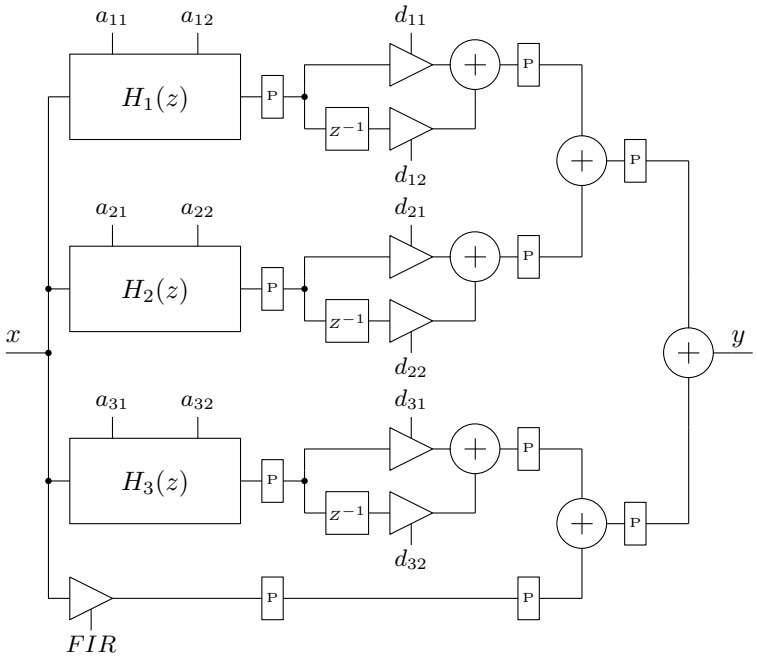


Figure 7: Non delayed parallel structure of IIR filter.

inserting pipeline registers.

The Look-ahead pipelining techniques [7] are very popular to insert pipeline registers in the IIR filters. The original transfer function has a single pole at $z = a$. The P-stage pipeline implementation is derived by adding $(P-1)$ poles and zeros at identical locations. The pipeline implementation of the 1st order filter is derived by the following equation

$$H(z) = \frac{\prod_{i=0}^{log_2^{P-1}}(1 + a^{2^i} z^{-2^i})}{1 - a^P z^{-P}} \tag{10}$$

The transfer function for the three stage pipeline implementation of the 1st order filter is shown below

$$H(z) = \frac{1 + az^{-1} + a^2 z^{-2}}{1 - a^3 z^{-3}} \tag{11}$$

The pipeline structure of the 1st order IIR filter is shown in Figure 10. Here, the critical path is obviously reduced by atleast three times. Thus this circuit can be operated at high frequencies. But the hardware complexity of the implementation increases.

There exists two Look-ahead techniques to implement pipeline IIR filters which are

1. Clustered Look-ahead Technique
2. Scattered Look-ahead Technique

In both the techniques, pipeline implementation of 1st order IIR filter is same. A P stage pipeline implementation using the Clustered Look-ahead technique for the higher filter is obtained by multiplying the numerator and denominator by

$$\sum_{i=0}^{P-1} r_i z^{-i} \tag{12}$$

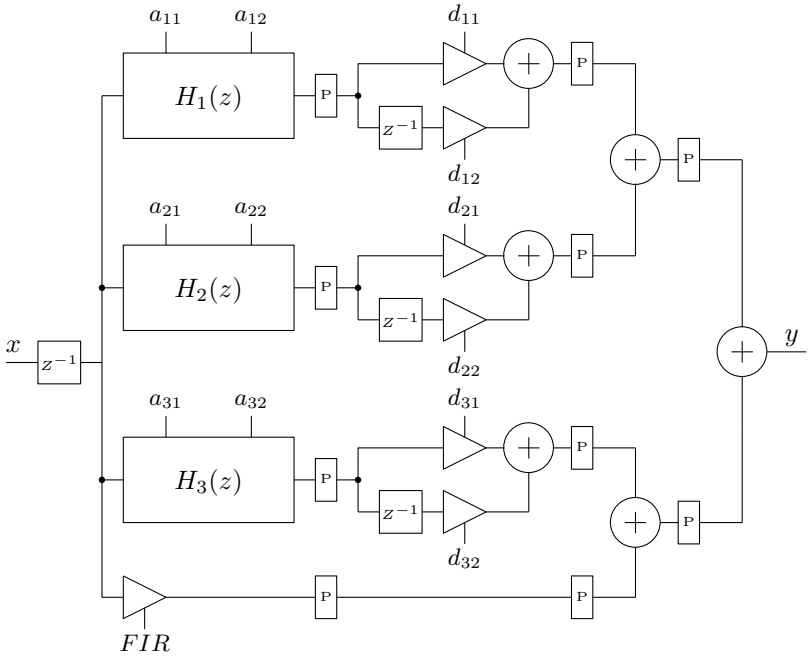


Figure 8: Delayed parallel structure of IIR filter.

(a) Structure of a typical first order IIR filter. (b) Transposed structure of first order IIR filter.

Figure 9: First order IIR filter structures.

where r_i is evaluated as follows

$$r_i = 0, for\, i = 0, 1, 2, ..., (N-1) \tag{13}$$

$$r_0 = 0 \tag{14}$$

$$r_i = \sum_{k=1}^{N} a_k r_{i-k},\ i > 0 \tag{15}$$

Here $(P-1)$ additional canceling poles and zeros are inserted. This method suffers from the stability problem. The pipeline implementation obtained from this method can be unstable even though the original transfer function was stable. Stable implementation can be obtained by inserting higher pipeline stages.

Other technique to obtain pipeline implementation is scattered Look-ahead technique. Lets, consider the denominator of a transfer function is represented as

$$D(z) = \prod_{i=1}^{P}(1 - p_i z^{-1}) \tag{16}$$

Then the P-stage pipeline implementation in this technique is obtained by the following equation

$$H(z) = \frac{N(z)}{D(z)} = \frac{N(z)\prod_{i=1}^{N}\prod_{k=1}^{P-1}(1 - p_i e^{j2\pi k/P} z^{-1})}{\prod_{i=1}^{N}\prod_{k=0}^{P-1}(1 - p_i e^{j2\pi k/P} z^{-1})} \tag{17}$$

This technique guarantees stability with less number of pipeline stages in comparison to the clustered Look-ahead technique. Both the techniques implements pipeline IIR filters with

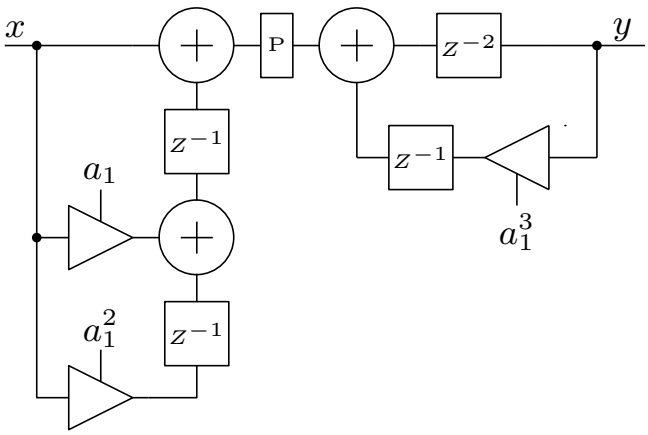


Figure 10: Pipelined version of the First order filter.

increased hardware complexity. It is better to apply this technique to the second order transfer functions instead of applying directly to the main transfer function. The second order transfer function $H(z) = 1/(1 - a_1 z^{-1} - a_2 z^{-2})$ is transformed into the following transfer function

$$H(z) = \frac{1 + a_1 z^{-1} + (a_1^2 + a_2) z^{-2} - a_1 a_2 z^{-3} + a_2^2 z^{-4}}{1 - (a_1^3 + a_1 a_2) z^{-3} - a_2^3 z^{-6}} \tag{18}$$

In this work, pipeline implementation of the IIR low pass filter is presented. The parallel IIR low pass filter is chosen here for pipeline implementation. The scattered Look-ahead technique is applied to the second order Biquads. The modified Biquad structure is shown in Figure 11. The resulting pipeline parallel IIR low pass filter is shown in Figure 12. Here the star marked co-efficients are the co-efficients of pipelined version of the IIR filter.

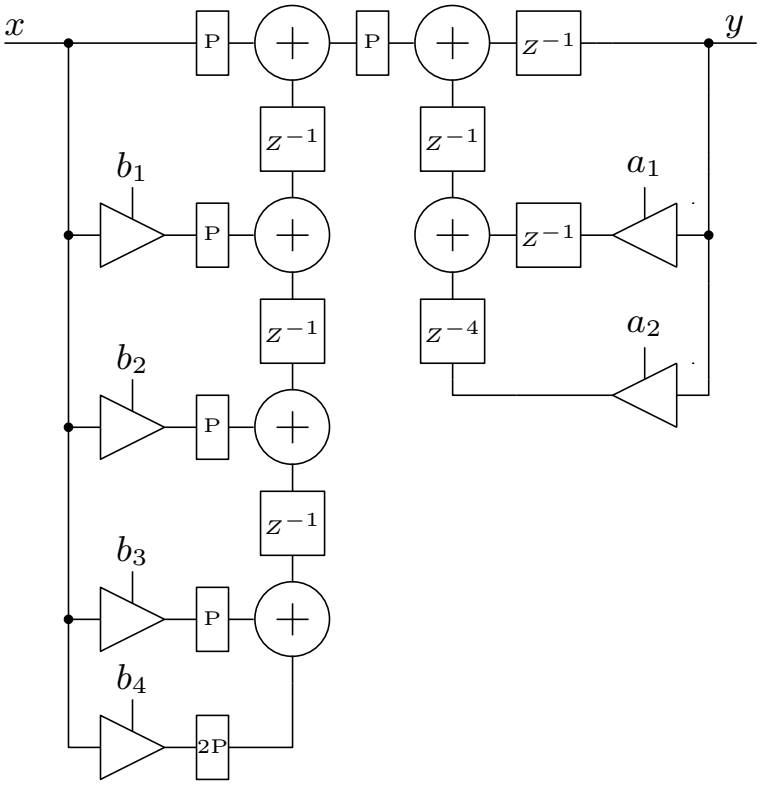


Figure 11: Pipelined version of the Biquad for parallel IIR filter.

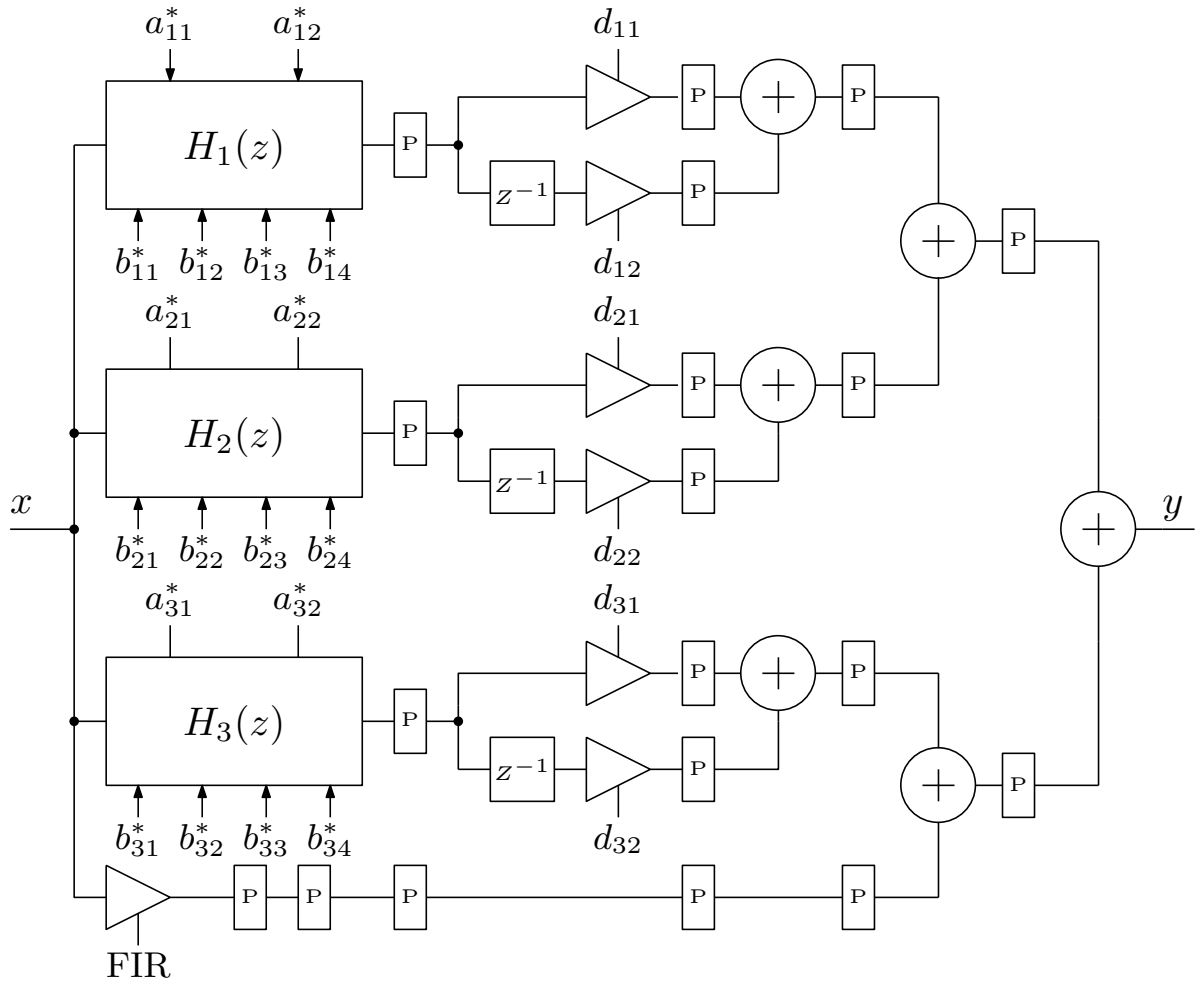


Figure 12: Pipelined version of the parallel IIR low pass filter.

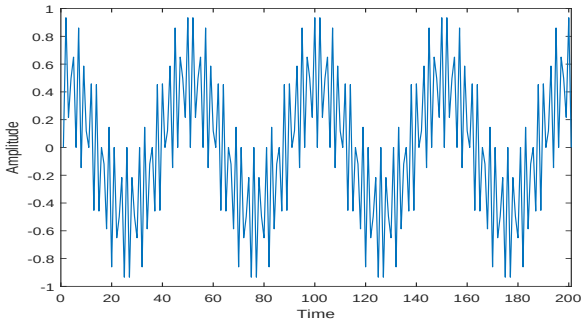
5 Performance Estimation

5.1 Implementation Issues

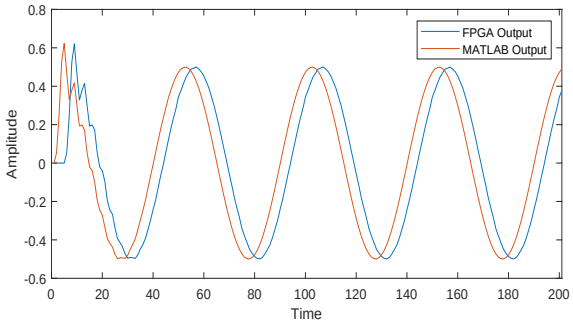
An implementation of the IIR filter can be serial or parallel. In this work, all the possible parallel architectures are mentioned. But in some applications the serial implementations may be useful. The major advantage of the IIR filters over FIR filters is that less number of filter order is required to achieve same frequency response. But at high frequency the IIR filters becomes inferior to the FIR filters. This is because of the high critical path involved in the IIR filter. This why the pipeline registers are inserted in the IIR filters to support the higher frequencies using Look-ahead techniques. The Look-ahead techniques are promising to insert pipeline registers. But these techniques increases hardware complexity and also inexact pole-zero cancellation may occur due to finite word-length. On the other hand, IIR filters can be designed in such a way that the transfer function is pipelinable. These methods are called constrained filter design techniques. These methods result hardware efficient pipelinable transfer functions.

5.2 Design Performance

In this work, low pass IIR filter of order 6 is implemented on NEXYS DDR2 artix7 FPGA device (xc7a100t-3csg324). The low pass filter is verified by taking two sinusoidal signals of frequencies $22KHz$ and $20KHz$. These two signals are multiplied and output of the multiplier



(a) Input signal to the IIR filter



(b) Output of the low pass filter

Figure 13: Input and output signal of the filter

Table 1: Performance Comparison of Different Structures.

Structures	CLK_{min}	Slice Reg	Slice LUT	DSP48	Occupied Slices	Power
Transposed Direct Form I	6.9 ns	222	228	10	71	0.029 W
Transposed Direct Form II	6.9 ns	240	231	10	85	0.027 W
Cascaded Form	6.9 ns	226	228	13	67	0.030 W
Parallel Form	6.9 ns	226	228	13	67	0.0584 W
Pipelined Parallel Form	4 ns	591	527	22	206	0.067 W

is given to the low pass filter. The sampling frequency is taken as $100KHz$ and thus the low pass filter filters out the signals whose frequency greater than $25KHz$. The output of the filter is a tone of $2KHz$ which is shown in Figure 13(b). The original output signal obtained from MATLAB and the FPGA based filtered output (delayed version) is compared in Figure 13. Here, 18-bit fixed point data width is chosen for implementation where 10-bit is reserved for fractional part. Here, Root Mean Squared Error (RMSE) is used to measure the design performance. RMSE is computed as

$$RMSE = \frac{\|(\hat{y} - y)\|_2}{\|y\|_2} \tag{19}$$

Here y is MATLAB based filtered output and $\hat{y}$ is FPGA output. A RMSE of 0.0059 is achieved using 18-bits of word length for parallel IIR implementation. A RMSE of 0.00584 is achieved when the same parallel filter is implemented for higher frequency application.

5.3 Comparison of Different IIR Filter Structures

The performance of the FPGA implementation of different IIR low pass filter structures are compared and this comparison is shown in Table 1. The performance of the transposed direct form structures are almost same. The cascaded structures are preferred over direct forms due to its immunity towards the quantization noise. The cascaded structures consume slightly higher resources and have higher latency. The parallel structures are more popular than any other structures. They have similar immunity towards the quantization noise and also introduces parallelism to the design. Thus latency is reduced but consumes more hardware than direct forms. All these structures are not suitable for higher frequency applications as they have longer critical path. This critical path is due to the delay of two adders and one multiplier. This critical path is reduced in the pipelined parallel IIR filter. Higher maximum frequency is achieved with the cost of extra hardware.

6 Conclusion

In this work, FPGA implementation of the different IIR filter structures is presented and a comparison of the performances is presented here. Here, a low pass filter is designed using different structures to demonstrate difference in implementation. Direct form structures directly implement the IIR transfer functions and these are systolic architectures. Transposed structures are very popular as they have shorter critical path. In fixed point implementation, cascaded and parallel structures provide less quantization noise compared to the direct forms. All this structures suffer from the long critical path and thus not suitable for high frequency applications. Look ahead techniques are very popular to implement pipeline IIR filters. The parallel IIR filter is implemented using the scattered look ahead technique and high frequency is achieved. RMSE of both type of parallel filters are compared. This work, covers almost all the IIR filter structures and design aspects.

References

[1] XILINX, "7 series dsp48e1 slice," vol. UG479 (v1.10), march 2018. [Online]. Available: https://www.xilinx.com/support/documentation/user_guides/ug479_7Series_DSP48E1.pdf

[2] S. Roy, "Parallel fpga implementation of fir filters." digitalsystemdesign.in, April 2020.

[3] P. R. Babu, *Digital Signal Processing.* SCITECH, 2009.

[4] M. Francis, "Infinite impulse response filter structures in xilinx fpgas," in *XILINX*, 2009.

[5] B. Bank, "Direct design of parallel second-order filters for instrument body modeling." in *ICMC*, 2007.

[6] B. Bank and J. O. Smith, "A delayed parallel filter structure with an fir part having improved numerical properties," 2014.

[7] K. K. Parhi, *VLSI digital signal processing systems: design and implementation.* John Wiley & Sons, 2007.