

CHAPTER 5

FIR FILTER DESIGN

5.1 INTRODUCTION

A Finite Impulse Response (FIR) filter is a type of a digital filter. The impulse response of an N^{th} -order FIR filter lasts for $N+1$ sample, and then dies to zero. The difference equation that defines the output of an FIR filter in terms of its input is given in Equation (5.1).

$$y(n) = \sum_{n=0}^{N-1} x(n) h(N-n) \quad (5.1)$$

Where,

$x[n]$ is the input signal, $y[n]$ is the output signal

b_i are the filter coefficients and N is the filter order – an N^{th} -order filter has $(N + 1)$ terms on the right-hand side. These are commonly referred to as taps.

Equation (5.1) can also be expressed as a convolution of the coefficient sequence b_i with the input signal. That is, the filter output is weighed as the sum of the current and a finite number of previous values of the input. A FIR "tap" is simply a coefficient/delay pair. The number of FIR taps, (often designated as " N ") is an indication of

1. The amount of memory required to implement the filter
2. The number of calculations required and
3. The amount of "filtering" the filter can do

There are a few terms used to describe the behavior and performance of FIR filter including the following:

- Filter Coefficients - a set of constants, also called tap weights, used to multiply against delayed sample values. For an FIR filter, the filter coefficients are, by definition, the impulse response of the filter
- Impulse Response - a filter's time domain output sequence when the input is an impulse. An impulse is a single unity-valued sample followed and preceded by zero-valued samples. For an FIR filter the impulse response of a FIR filter is the set of filter coefficients. The impulse response, often denoted $H(z)$ or $h(n)$ is a measurement of how a filter will respond to the Kronecker delta function.
- Tap - The number of FIR taps, typically N , tells a couple things about the filter. Most importantly, it tells the amount of memory needed, the number of calculations required, and the amount of "filtering" that it can do. Basically, the more taps in a filter results in better stop band attenuation (less of the part filtered out), less rippling (less variations in the pass band), and steeper roll off (a shorter transition between the pass band and the stop band).

Multiply-Accumulate (MAC): In the context of FIR Filters, a "MAC" refers to the operation of multiplying a coefficient by the corresponding delayed data sample and accumulating the result. There is usually one MAC per tap. FIR filters have a transfer function of a polynomial in z - and is an all-zero filter in the sense that the zeroes in the z -plane determine the frequency response magnitude characteristic. The Z-transform of the impulse response yields the transfer function of the FIR filter.

The z transform of a N -point FIR filter is given in Equation (5.2)

$$H(z) = \sum_{n=0}^{N-1} h(n)z^{-n} \quad (5.2)$$

FIR filters are particularly useful for applications where exact linear phase response is required. The FIR filter is generally implemented in a non-recursive way which guarantees a stable filter.

FIR filter design essentially consists of two parts:

- i. Approximation problem
- ii. Realization problem

The approximation stage takes the specification and gives a transfer function through four steps. They are as follows:

- i. A desired or ideal response is chosen, usually in the frequency domain.

- ii. An allowed class of filters is chosen (e. g. the length N for a FIR filters).
- iii. A measure of the quality of approximation is chosen.
- iv. A method or algorithm has been selected to find the best filter transfer function.

The realization part deals with choosing the structure to implement the transfer function which may be in the form of circuit diagram or in the form of a program. There are essentially three well-known methods for FIR filter design, namely

- The window method
- The frequency sampling technique
- Optimal filter design method

Among the three methods the window method is always preferred because of its relative simplicity as compared to other methods and ease of use. The fact that well-defined equations are often available for calculating the window coefficients has made this method successful and also straight forward to obtain the filter response with minimal computational effort.

5.1.1 Algorithm for Design of FIR Filter

- | | |
|--------|---|
| Step 1 | Obtain the difference equation for filter output in terms of input and order of filter. |
|--------|---|

- | | |
|--------|--|
| Step 2 | Determination of filter coefficient or tap weights used to multiply against delayed sample values. |
| Step 3 | Selection of number of taps based on the stop band attenuation, less rippling and steeper roll off. |
| Step 4 | Selection of appropriate windowing method for calculating the window coefficient with minimal computational effort. |
| Step 5 | Determine the value of filter coefficient and windowing function based on the frequency response and cutoff frequency. |
| Step 6 | Scaling of filter coefficient using the scaling factor. |
| Step 7 | Design of optimized PDP n bit delay element, adder and multiplier for producing the output with low power and delay. |

5.2 WINDOWING METHOD

In signal processing, a window function (also known as an apodization function or tapering function) is a function that is zero-valued outside of some chosen interval. This multiplication or windowing has the effect of making the filter impulse response finite in duration (hence FIR), but also has the effect of smearing the desired frequency response. In this method from the desired frequency response specification $H_d(w)$, corresponding unit sample response $h_d(n)$ is determined using the Equations (5.3) and (5.4).

$$h_d(n) = \frac{1}{2\pi} \int_{-\pi}^{+\pi} H_d(w) e^{jwn} dw \quad (5.3)$$

where

$$H_d(w) = \sum_{n=-\infty}^{\infty} h_d(n) e^{-jwn} \quad (5.4)$$

In general, unit sample response $h_d(n)$ obtained from the above relation is infinite in duration, so it must be truncated at some point say $n = N-1$ to yield an FIR filter of length N (i.e. $0 \leq n \leq N-1$).

This truncation of $h_d(n)$ to length $N-1$ is same as multiplying $h_d(n)$ by the rectangular window as per the Equation (5.5).

$$\begin{aligned} w(n) &= 1 \quad 0 \leq n \leq N-1 \\ &= 0 \quad \text{otherwise} \end{aligned} \quad (5.5)$$

Among the different windowing techniques Hamming window is preferred because of its ability to reduce its side lobes while maintaining the main lobe width as $8\pi/N$. When compared with other techniques, the maximum side lobe is about one-fifth of the Hanning window which is raised with the cosine of simpler coefficients given by Equation (5.6).

$$w(n) = 0.54 + 0.46 \cos\left(\frac{2\pi n}{N-1}\right) \quad (5.6)$$

Thus the unit sample response of the FIR filter becomes Equation (5.7).

$$h(n) = h_d(n) w(n) \quad (5.7)$$

$$= h_d(n) \quad 0 \leq n \leq N-1$$

$$= 0 \quad \text{otherwise}$$

Now, the multiplication of the window function $w(n)$ with $h_d(n)$ is equivalent to convolution of $H_d(w)$ with $W(w)$, where $W(w)$ is the frequency domain representation of the window function given in Equation (5.8).

$$W(w) = \sum_{n=0}^{N-1} w(n) e^{-jwn} \quad (5.8)$$

Thus the convolution of $H_d(w)$ with $W(w)$ yields the frequency response of the truncated FIR filter as given in Equation (5.9).

$$H(w) = \frac{1}{2\pi} \int_{-\pi}^{+\pi} H_d(v) W(w-v) dv \quad (5.9)$$

The frequency response can also be obtained using the Equation (5.10).

$$H(w) = \sum_{n=0}^{N-1} h(n) e^{-jwn} \quad (5.10)$$

But direct truncation of $h_d(n)$ to N terms to obtain $h(n)$ leads to the Gibbs phenomenon effect. It manifests itself as a fixed percentage overshoot and ripple before and after an approximated discontinuity in the frequency response. This is due to the non-uniform convergence of the fourier series at a discontinuity. Thus the frequency response obtained contains ripples in the frequency domain. In order to reduce the ripples, instead of multiplying $h_d(n)$ with a rectangular window $w(n)$, $h_d(n)$ is

multiplied with a window function that contains a taper and decays toward zero gradually, instead of abruptly as it occurs in a rectangular window. As multiplication of sequences $h_d(n)$ and $w(n)$ in time domain is equivalent to convolution of $H_d(w)$ and $W(w)$ in the frequency domain, it has the effect of smoothing $H_d(w)$.

The several effects of windowing the Fourier coefficients of the filter on the result of the frequency response of the filter are as follows:

- i. A major effect is that discontinuities in $H(w)$ become transition bands between values on either side of the discontinuity.
- ii. The width of the transition bands depends on the width of the main lobe of the frequency response of the window function, $w(n)$ i.e. $W(w)$.
- iii. Since the filter frequency response is obtained via a convolution relation, it is clear that the resulting filters are never optimal in any sense.
- iv. As N (the length of the window function) increases, the main lobe width of $W(w)$ is reduced. It reduces the width of the transition band, but this also introduces more ripple in the frequency response.
- v. The window function eliminates the ringing effects at the band edge and does result in lower side lobes at the expense of an increase in the width of the transition band of the filter. Thus the $h(n)$ can be obtained for the desired value of cut off frequency.

Procedural steps to determine the filter coefficients are given below :

- Step 1** Determine the frequency response for the required filter using the Equation (5.3).
- Step 2** Determine the window function for the corresponding filter as per the Equation (5.6).
- Step 3** Finally determine the value of the filter coefficients using the Equation (5.7).

The low pass FIR filter designed for cut off frequency and sampling frequency values are 2 KHz and 16 KHz respectively.

The procedural steps for calculating the filter co-efficient for the desired filter are as follows :

Step 1: The frequency response for the design of low pass FIR filter with cut off frequency, $f_c = 2$ KHz, $f_s = 16$ KHz is given by Equation (5.11).

$$H_d(\omega) = e^{-j\omega\alpha} - (\pi/4) \leq \omega_c \leq (\pi/4) \quad (5.11)$$

$$= 0 \quad \text{otherwise}$$

Where normalization of cut-off frequency

$$\omega_c = 2\pi * (f_c / f_s)$$

$$= 2 \times 3.14 \times (2000/16000) = (\pi/4) \text{ radians}$$

The desired filter coefficients are obtained by Equation (5.12)

$$h_d(n) = \frac{1}{2\pi} \int_{-\omega_c}^{+\omega_c} H_d(\omega) e^{j\omega n} d\omega \quad (5.12)$$

Substituting the value for $H_d(\omega)$, helps to get

$$h_d(n) = \frac{1}{2\pi} \int_{-(\pi/4)}^{(\pi/4)} e^{-j\omega\alpha} e^{j\omega n} d\omega$$

$$h_d(n) = \frac{1}{2\pi} \int_{-(\pi/4)}^{(\pi/4)} e^{j\omega n} d\omega$$

$$= \frac{1}{2\pi} \left(\frac{e^{j\omega n}}{jn} \right)_{-(\pi/4)}^{(\pi/4)}$$

$$= \frac{1}{\pi n} \left(\frac{e^{j(\pi/4)n} - e^{-j(\pi/4)n}}{2j} \right)$$

$$= \frac{1}{\pi n} \sin \frac{n\pi}{4}$$

$$\sin x = \left(\frac{e^{jx} - e^{-jx}}{2j} \right)$$

$$= \frac{1}{4} \frac{\sin(n\pi/4)}{(n\pi/4)}$$

Calculation of filter coefficients

$$h_d(0) = \frac{1}{4} \frac{\sin 0}{0} = 0.25$$

$$h_d(1) = \frac{1}{4} \frac{\sin(\pi/4)}{(\pi/4)} = 0.225$$

$$h_d(2) = \frac{1}{4} \frac{\sin(\pi/2)}{(\pi/2)} = 0.159$$

$$h_d(3) = \frac{1}{4} \frac{\sin(3\pi/4)}{(3\pi/4)} = 0.075$$

Step 2: Generally the window function $w(n)$ is given by Equation (5.13).

$$w(n) = \alpha + (1 - \alpha) \cos\left(\frac{2\pi n}{N-1}\right) \quad (5.13)$$

For hamming window, $\alpha = 0.54$, thus it is obtained as

$$w(n) = 0.54 + 0.46 \cos\left(\frac{2\pi n}{N-1}\right) \quad (5.14)$$

Substituting the different values for n , one gets

$$w(0) = 1$$

$$w(1) = 0.31$$

$$w(2) = 0.31$$

$$w(3) = 1$$

Step 3: By window function, the filter coefficients are,

$$h(0) = h_d(0) * w(0) = 0.250$$

$$h(1) = h_d(1) * w(1) = 0.0697$$

$$h(2) = h_d(2) * w(2) = 0.0493$$

$$h(3) = h_d(3) * w(3) = 0.0750$$

5.3 SCALING OF COEFFICIENTS

$$\text{Scale factor, } h = \lceil \log_2 ((2^{M-1}-1)/\max(|h_i|)) \rceil$$

$$M = \text{coefficient word size in bits (M = 16)}$$

$$h = \lceil \log_2 ((2^{15}-1)/0.250) \rceil$$

$$h = \log_2 2^{17} = 17$$

$$h = 17$$

Scaled coefficients can be obtained by

$$h' = \text{round}(h * 2^b)$$

$$h_0' = \text{round}(0.250 * 2^{17}) = 32768_{10} = \mathbf{8000_H}$$

$$h_1' = \text{round}(0.0697 * 2^{17}) = 9145_{10} = \mathbf{23B9_H}$$

$$h_2' = \text{round}(0.0493 * 2^{17}) = 25867_{10} = \mathbf{650B_H}$$

$$h_3' = \text{round}(0.0750 * 2^{17}) = 9833_{10} = \mathbf{2669_H}$$

After the design of the filter, it must be realized by developing a signal flow diagram that describes the filter in terms of operations on sample sequences. The System function of an FIR filter given in the Equation (5.1) can be realized using direct form realization of filters. For designing a N tap filter the direct form realization or transversal structure requires N Multipliers, N-1 Adders and N-1 Delay elements.

The direct form realization of the 4 tap low pass FIR filter is shown in Figure 5.1.

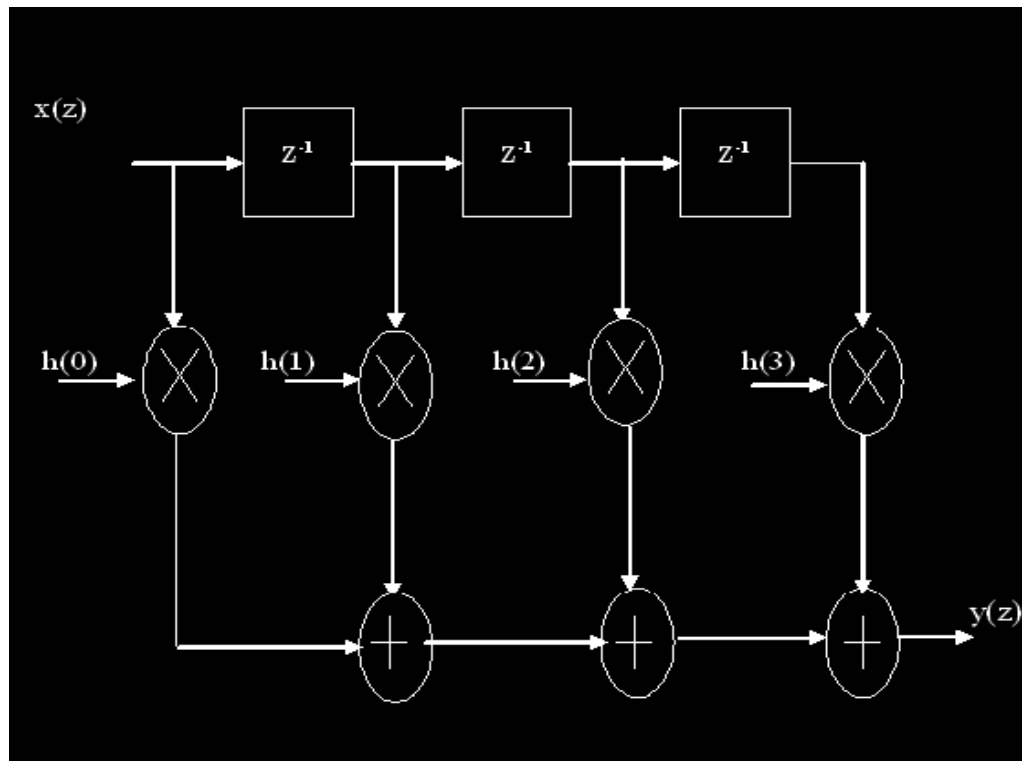


Figure 5.1 Direct form realization of 4 tap low pass FIR filter

The schematic diagram of 4-tap low pass FIR filter designed with Tanner Spice is illustrated in Figure 5.2.

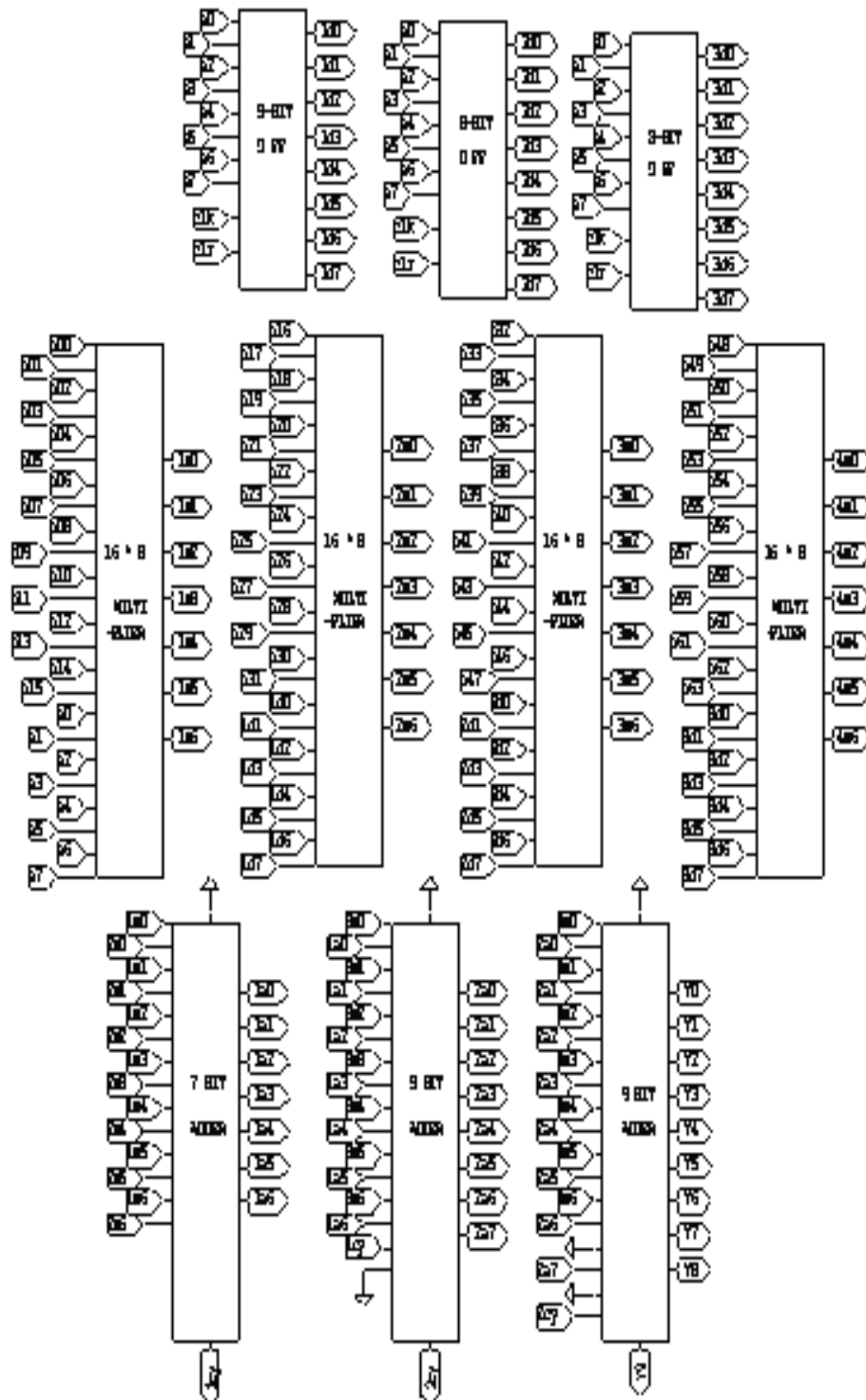


Figure 5.2 Schematic diagram of 4-tap low Pass FIR filter

5.4 DELAY ELEMENT

The delay flip-flop (DFF) is unique in that it only has one external input along with a clock input. The logic symbol for this flip-flop is given in Figure 5.3.

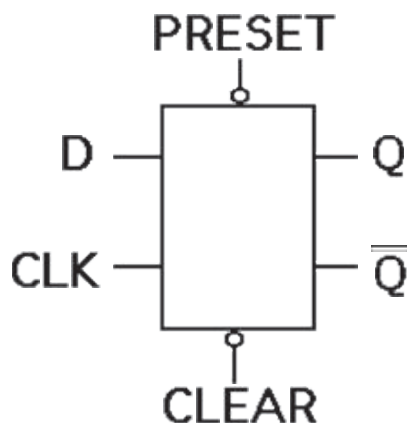


Figure 5.3 Logic symbol of D flip-flop

Here the two asynchronous inputs, PRESET and CLEAR enable the flip-flop to be set to a predetermined state, independent of the CLOCK. Note the invert bubble on these lines which indicates that these lines are normally held at 1 and that the function (CLEAR or PRESET) is performed by taking the line to 0. The delay flip-flop transfers whatever is at the external input D to the output Q. This does not happen immediately however and only happen on a rising clock pulse (i.e. as CLK goes from 0 to 1). The input is thus delayed by up to a clock pulse before appearing at the output. This is illustrated in the timing diagram shown in Figure 5.4. The DFF is an edge-triggered device which means that the change of state occurs on a clock transition (in this case, the rising clock pulse as it goes from 0 to 1).

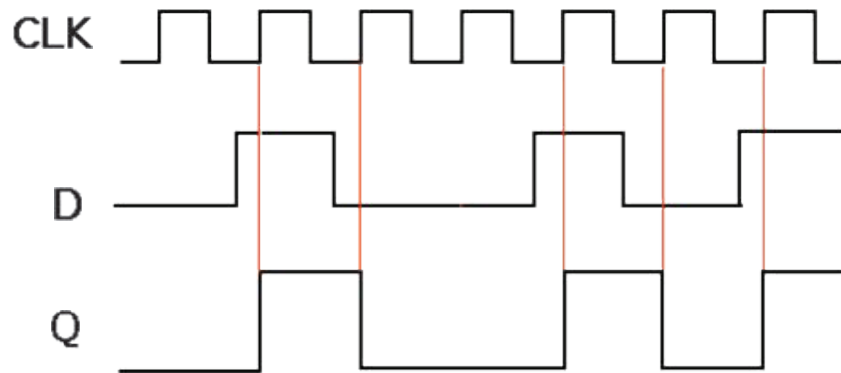


Figure 5.4 Waveforms of D flip-flop

Table 5.1 Transition table of D flip-flop

Present State	Delay Input	Next state
Q	D	Q+1
0	0	0
0	1	1
1	0	0
1	1	1

The characteristic equation of the D flip-flop is given in Equation (5.14).

$$\mathbf{Q + 1 = D} \quad (5.14)$$

5.5 ANALYSIS OF VARIOUS FLIP-FLOPS

5.5.1 Hybrid Latch Flip-Flop Circuit

The hybrid latch flip-flop circuit shown in Figure 5.5 was implemented with 20 transistors. It has the special set-up transparency window by inserting three inverters and it consumes more power than the MUX based flip-flop.

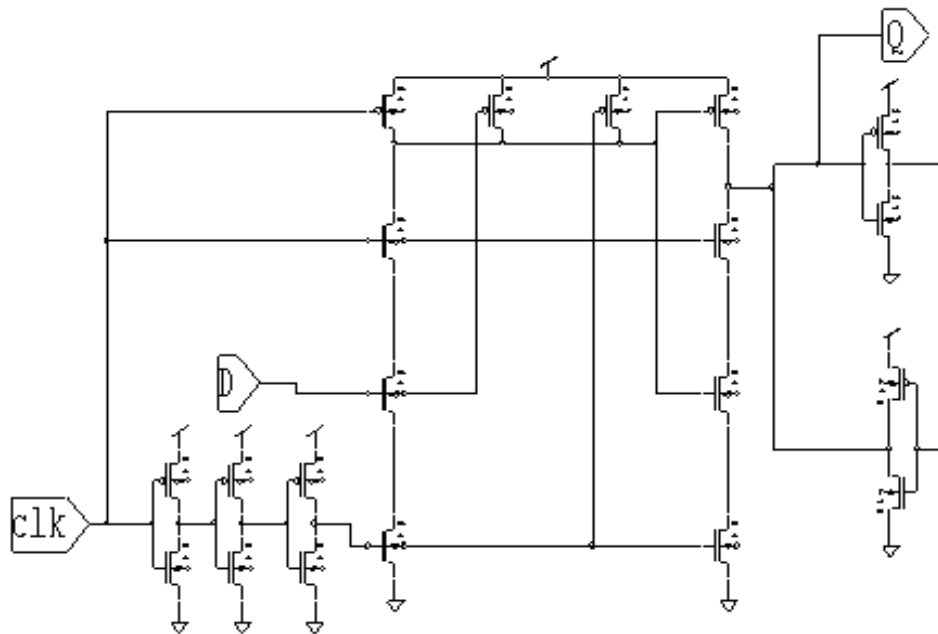


Figure 5.5 Hybrid latch flip-flop

Transparency window was defined by the propagation time of three inverters. It reduced the complexity of locking mechanism and results in small area and small delay. The HLFF circuit consumes considerable power because of the circuit's high internal activity even when input activity is small. Also, increased output activity occurred due to glitches

when the output was at the high level which contributed to the total power dissipation.

The first stage of the circuit conditionally generates the glitch in the transparency window based on the level of input signal. It can be easily noticed that it was formed by static 3-input CMOS NAND gate, which has the functionality for wanted glitch generation.

Every time when input is high the glitch was generated, regardless of the previous state of the output. Also, the circuit suffers from an unwanted glitch at the output, generated by the race condition.

5.5.2 Conditional Pre-Charge Flip-Flop Circuit

Another circuit for the Conditional pre-charge flip-flop is shown in Figure 5.6 and it was implemented with 23 transistors. The power consumption was less than that of HLFF and its performance comparison is given in Table 5.7.

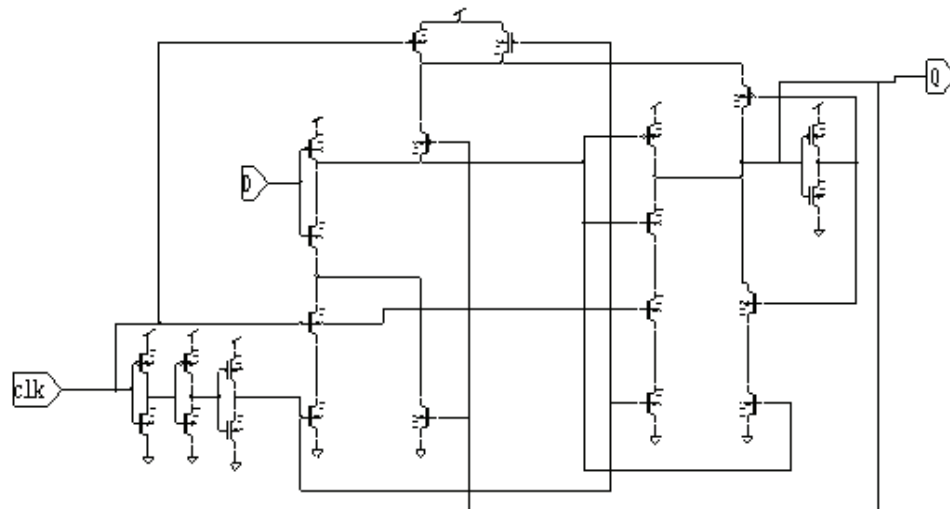


Figure 5.6 Conditional pre-charge flip-flops

One of the most important contributions of this work is related to abandoning unconditional pre-charge operation of the internal node as in the hybrid latch flip-flop previously designed. This is accomplished by controlling the return of internal node to inactive (high) state using the information about previous flip-flop state, allowing the internal node to stay at low level until input the condition was changed.

This approach was efficiently eliminated the unnecessary transitions of the internal node as well as race condition at the output. There were two main disadvantages of this approach in terms of propagation time. It

1. introduces the critical path for low input level capture and
2. increases the output load due to the feedback.

5.5.3 NOR Based Flip-Flop Using CMOS Logic

A NOR based flip-flop design using CMOS logic was implemented with 18 transistors and it is shown in Figure 5.7. The power consumption was less than the CPFF and its performance comparison is given in Table 5.7. This structure is designed with NOR gate and is realized with CMOS logic. The delay in producing the output is more than NAND based flip-flop and MUX based flip-flop.

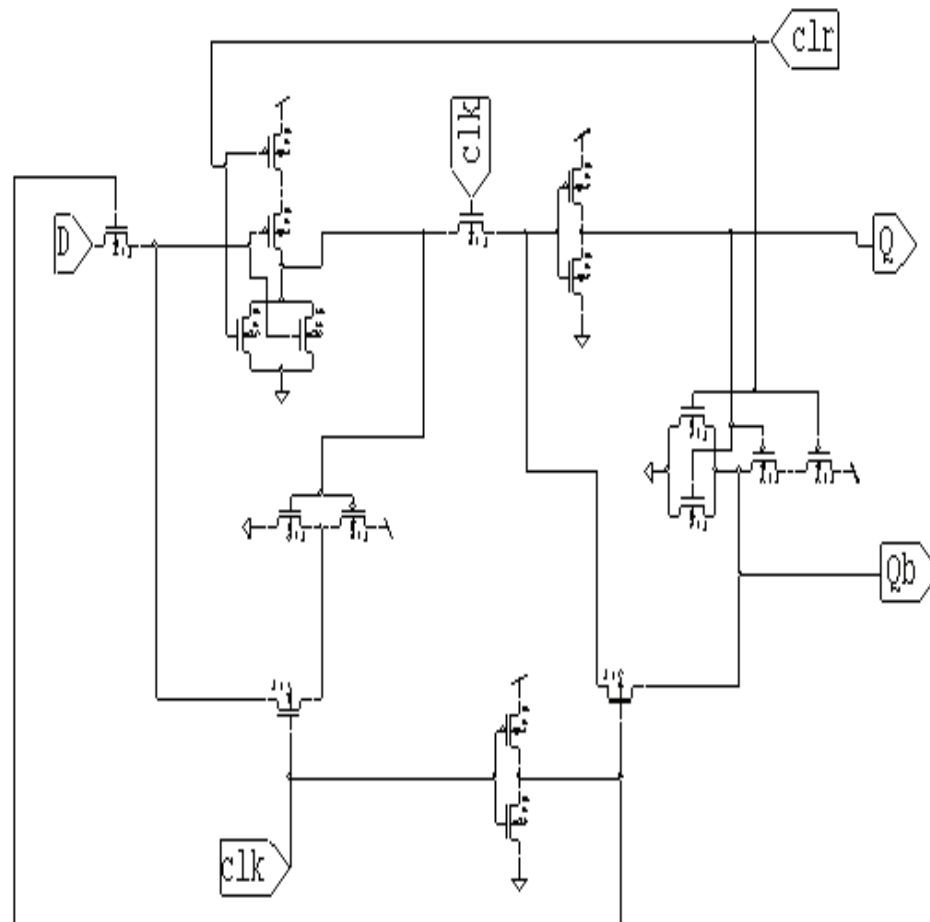


Figure 5.7 NOR based flip-flop circuit using CMOS logic

5.5.4 NAND Based Flip-Flop Circuit

The NAND based flip-flop circuit using CMOS logic is shown in Figure 5.8 and it was implemented with 18 transistors. A comparison of its power consumption with the earlier discussed flip-flops shows that, it consumes less power consumption as shown in Table 5.6. This flip-flop is designed with NAND gate to produce the Q output and Q bar output. The NAND gates are realized with CMOS logic and it require less delay to produce the output.

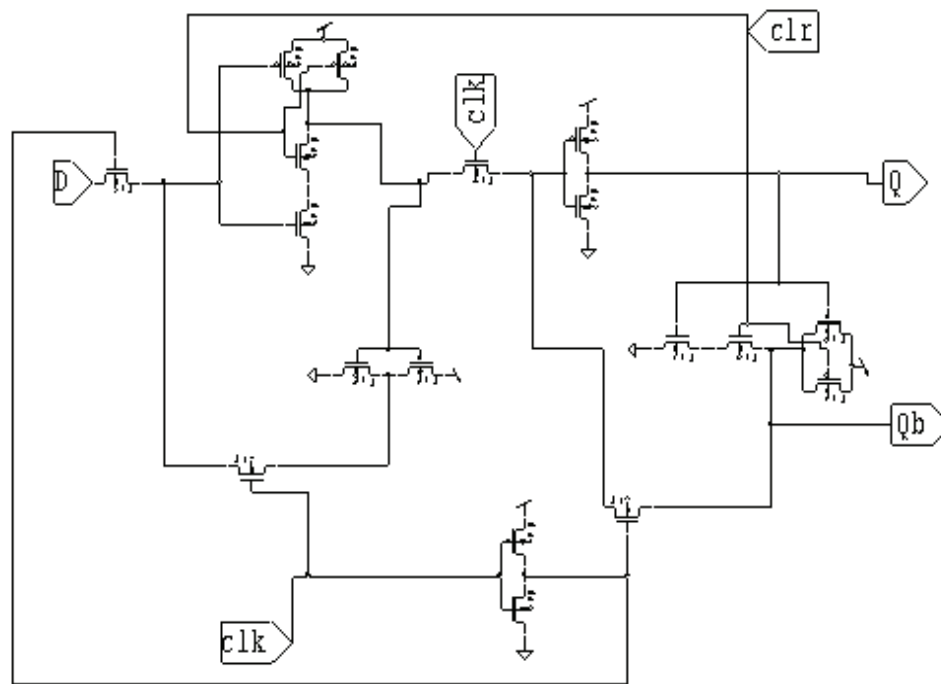


Figure 5.8 NAND based flip-flop

5.5.5 Multiplexer Based Proposed Flip-Flop

The multiplexers are implemented using NMOS and PMOS pass transistors. NMOS transistor passes the signal only when the control signal is '1' and PMOS transistor passes the signal only when the control signal is '0'. NMOS Transistor is generally efficient for the transmission of '0', while PMOS transistor is efficient for the transmission of '1'. In this regard, a 2-to-1 multiplexer is designed using NMOS and PMOS pass transistor. Depending on the select lines the corresponding input will be selected. If the select input is '0', then the output will be from A signal. If select input is '1', then the output will be from B signal. The schematic structure is manifested in Figure 5.9.

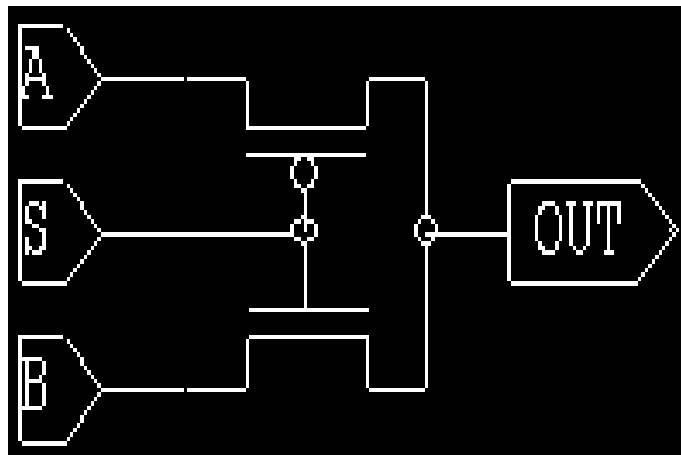


Figure 5.9 2-to-1 multiplexer using pass transistor

The 2-to-1 multiplexer has two inputs and one select input. The truth table of the 2-to-1 multiplexer is shown in Table 5.2.

Table 5.2 Truth table of 2-to-1 multiplexer

S.No.	Select Input	Output
1	0	A
2	1	B

The MUX based flip-flop is implemented with 18 transistors and it is shown in Figure 5.10. It requires less PDP than other flip-flop architectures discussed earlier and it is used as delay element in FIR filter design. The gates are realized with 2-to-1 MUX and are faster to produce the Q and Q bar output. The power consumption of the structure is less than the other flip-flop architectures and the performance comparison is given in Table 5.7.

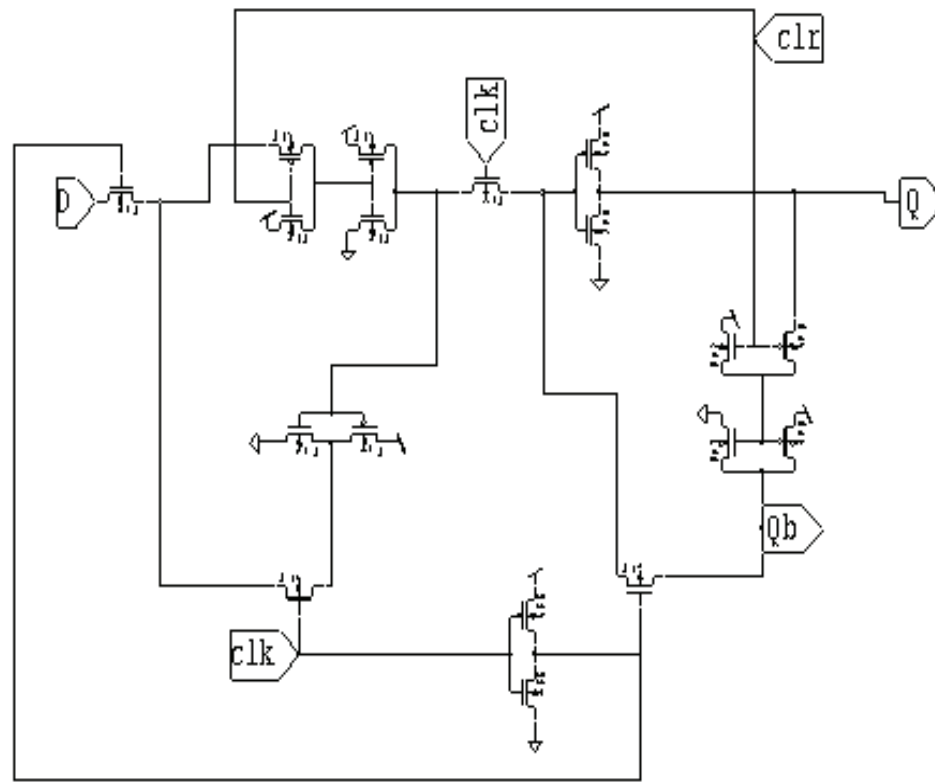


Figure 5.10 Multiplexer based proposed flip-flop

5.6 SEVEN BIT ADDER

To design a 4-tap low pass FIR filter, one 7-bit adder is required. The 7-bit adder is constructed using MUX based adder in ripple carry structure is shown in Figure 5.11. The input is multiplied with $h(n)$ values and the output is truncated to 7 bits. The truncated output from the multipliers is given as input to the adder.

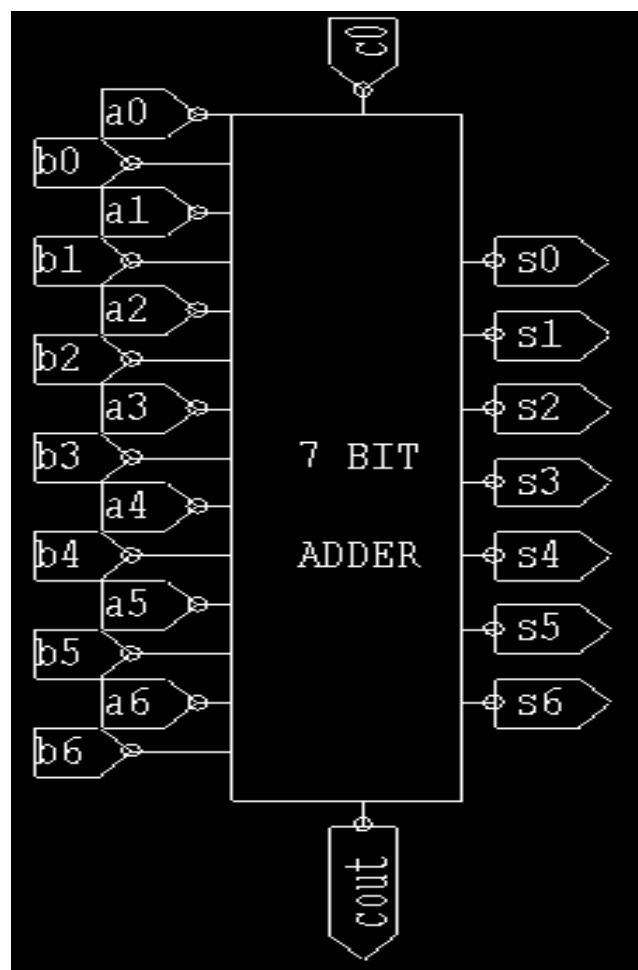
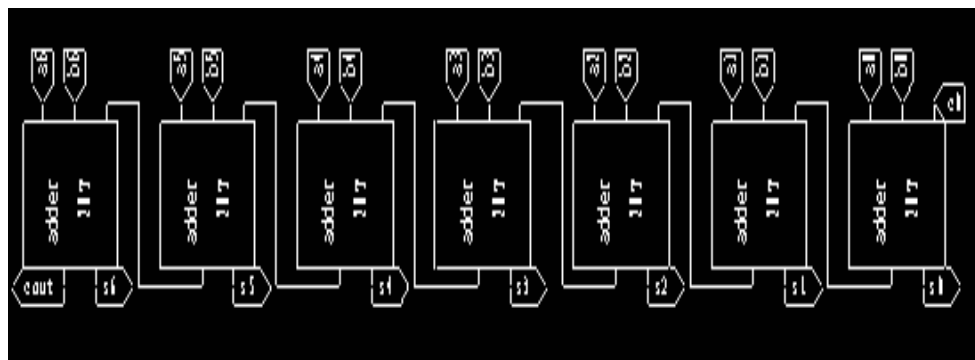


Figure 5.11 7-bit adder

5.7 MULTIPLIER ARCHITECTURE

The design of 4-tap low pass FIR filter needs, a 16×8 multiplier. The binary multiplication was based on the basic operation of AND logical operation. The proposed binary tree based multiplier, presented in Chapter 4, and array multiplier architecture are used for multiplication. The multiplier accepts the multiplier and multiplicand and uses an array of cells to calculate the bit products $a_j \cdot b_k$ individually in a parallel manner. First calculate the bit product $a_j \cdot b_k$ and then add it to other contributions in column $i = (j + k)$. The bit product is determined by AND gates. The general structure for the 16×8 array multiplier is shown in Figure 5.12. In general, an array multiplier for n bit words requires $n(n-2)$ full adders, n half adders and n^2 AND gates. For the design of 16×8 array multiplier, it requires 40 AND gates, 26 full adders and 5 half adders.

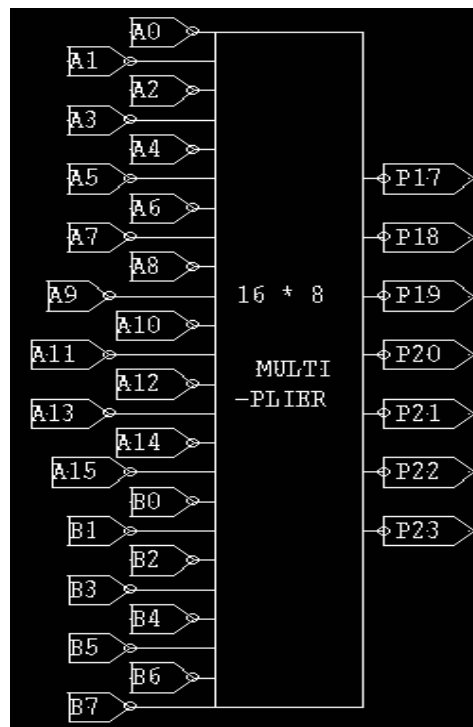


Figure 5.12 Symbol mode of 16×8 multiplier

5.8 RESULTS AND DISCUSSION

5.8.1 Power Consumption of Hybrid Latch Flip-Flop

The hybrid latch flip-flop circuit is designed and simulated with 0.18 μm CMOS technology using Tanner Spice. The simulated output waveform is revealed in Figure 5.13 and its power consumption from input sources is given in Table 5.3.

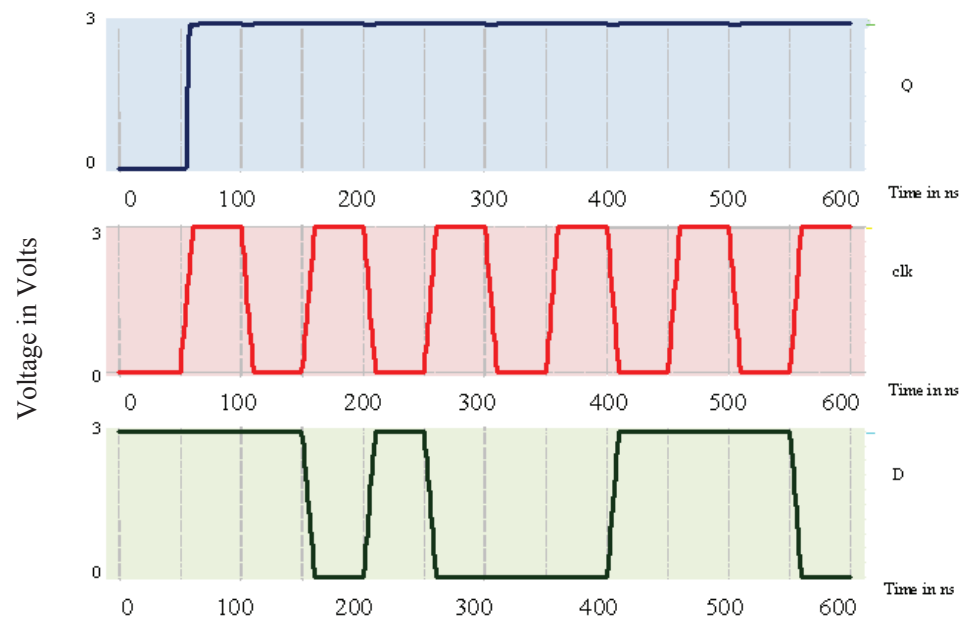


Figure 5.13 Simulated waveform of hybrid latch flip-flop

Table 5.3 Power consumption of hybrid latch flip-flop

S.No.	Name of the Source	Average Power Consumption (W)
1	D	8.23e-07
2	CLK	1.10e-06
3	VDD	8.30e-04
Total Dynamic Power		8.39e-04

5.8.2 Power Consumption of Conditional Pre-Charge Flip-Flop

The conditional pre-charge flip-flop is simulated using Tanner Spice with an input of 'D' and clock. The simulated 'Q' output waveform is shown in Figure 5.14 and its power consumption from input sources is furnished in Table 5.4.

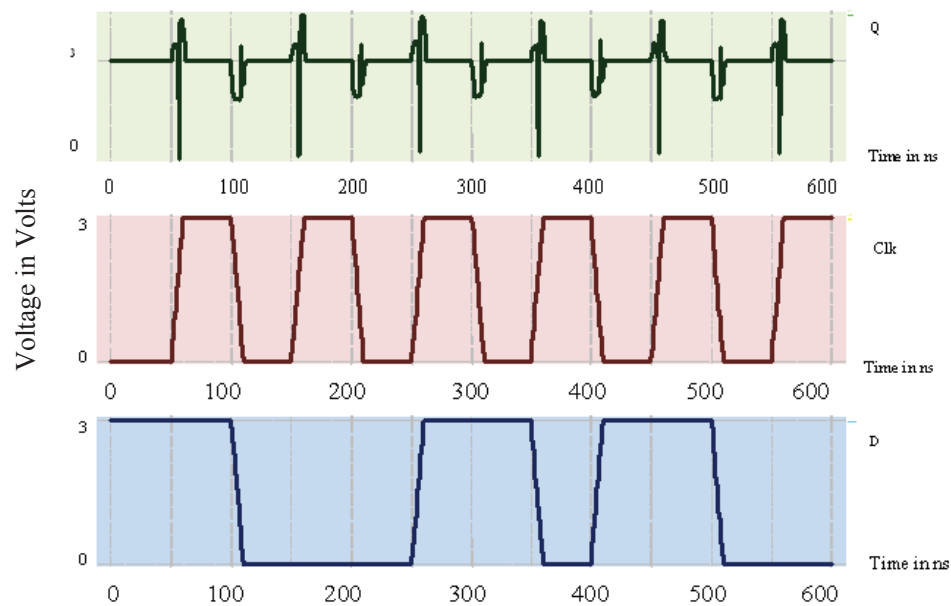


Figure 5.14 Simulated waveform of CPFF

Table 5.4 Power consumption of CPFF

S.No.	Name of the Source	Average Power Consumption (W)
1	D	0
2	CLK	1.61e-04
3	VDD	1.72e-04
Total Dynamic Power		3.33e-04

5.8.3 Power Consumption of NOR Based Flip-Flop Using CMOS Logic

The NOR based flip-flop is simulated with inputs 'D', clear, clock and its 'Q' and Q bar output waveform is shown in Figure 5.15. The power consumption from the input sources is given in Table 5.5.

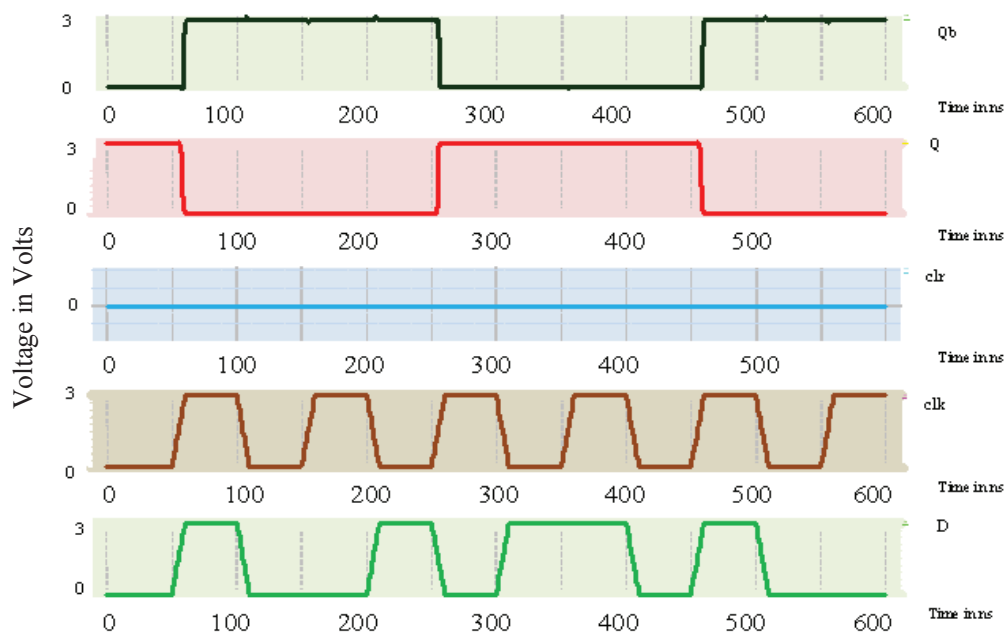


Figure 5.15 Simulated waveform of NOR CMOS based flip-flop

Table 5.5 Power consumption of NOR CMOS based flip-flop

S.No.	Name of the Source	Average Power Consumption (W)
1	D	0
2	CLK	0.68e-04
3	CLR	0
4	VDD	0.84e-04
Total Dynamic Power		1.52e-04

5.8.4 Power Consumption of NAND Based Flip-Flop

The NAND based flip-flop is simulated with inputs 'D', the clock and its 'Q' output are shown in Figure 5.16. The power consumption from the input sources is given in Table 5.6.



Figure 5.16 Simulated waveform of NAND based flip-flop

Table 5.6 Power consumption of NAND based flip-flop

S.No.	Name of the Source	Average Power Consumption (W)
1	D	0
2	CLK	0.09e-04
3	CLR	0
4	VDD	4.52e-04
Total Dynamic Power		4.61e-04

5.8.5 Power Consumption of MUX Based Flip-Flop

The proposed MUX based flip-flop is simulated with inputs 'D', clear, clock and its 'Q' and Q bar output waveform are shown in Figure 5.17. The power consumption from the input sources is given in Table 5.7.

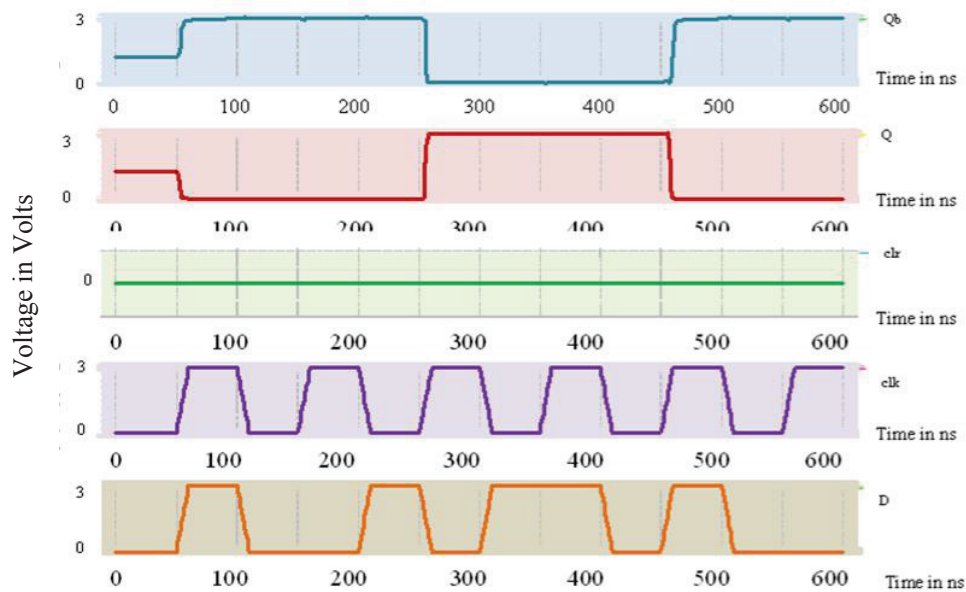


Figure 5.17 Simulated waveform of proposed MUX based flip-flop

Table 5.7 Power consumption of proposed MUX based flip-flop

S.No.	Name of the Source	Average Power Consumption (W)
1	D	0
2	CLK	0.095e-04
3	CLR	0
4	VDD	1.33e-04
Total Dynamic Power		1.35e-04

The performance comparison of various flip flop architectures focused earlier is shown in Table 5.8 and it is also represented as graphs from Figures 5.18 to 5.20.

Table 5.8 Performance comparison of various flip-flops

S.No.	Name of the Flip-flop	Avg. Power Consumption (W)	Delay (nS)	PDP (WS)
1	Hybrid-Latch flip-flop (Nedovic et al 2000)	8.39e-04	1.8	15.1e-13
2	Conditional pre-charge flip-flop (Zhang et al 2000)	3.33e-04	2.1	6.99e-13
3	NOR-CMOS based flip-flop (Knepper 2001)	1.52e-04	1.6	2.43e-13
4	NAND-CMOS based flip-flop (Knepper 2001)	4.61e-04	1.4	6.45e-13
5	Proposed MUX based flip-flop	1.35e-04	1.4	1.89e-13

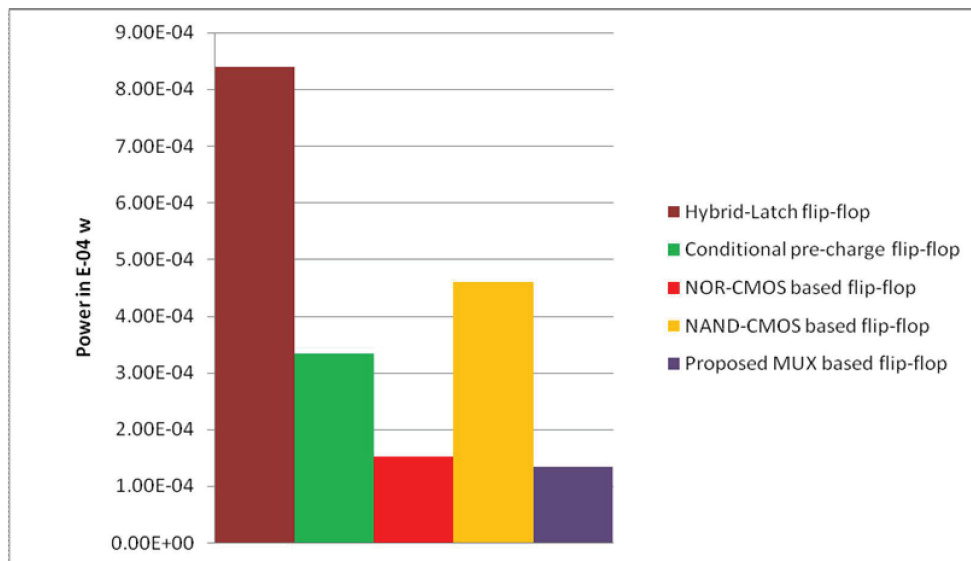


Figure 5.18 Power consumption of various flip-flops

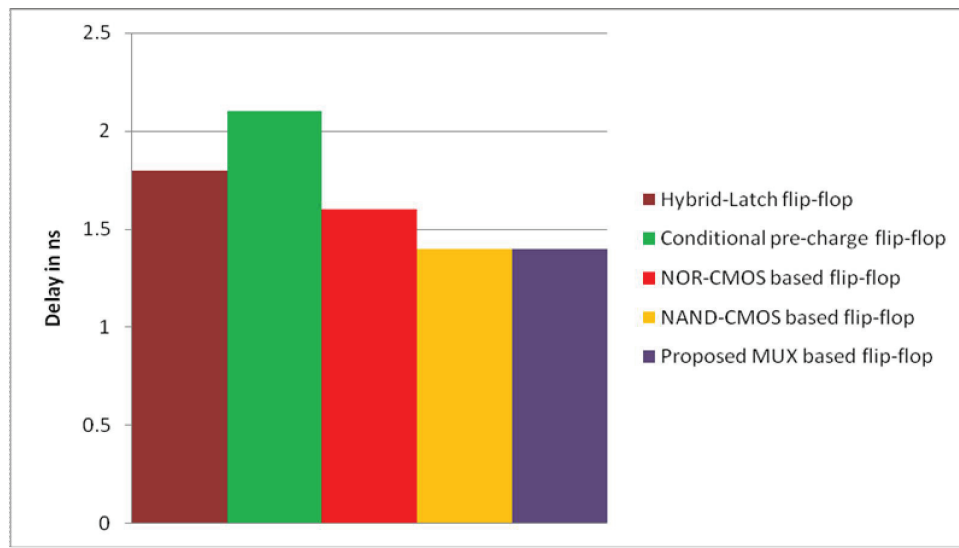


Figure 5.19 Delay of various flip-flops

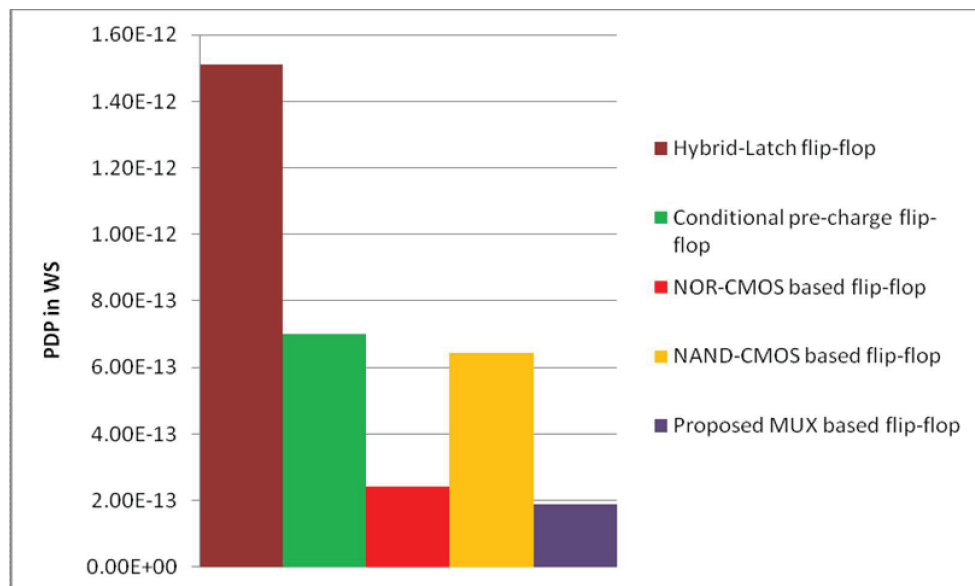


Figure 5.20 PDP of various flip-flops

5.9 OUTPUT WAVEFORM OF FIR FILTER USING PROPOSED ADDER AND MULTIPLIER ARCHITECTURE

The FIR filter is designed with low PDP adder, BT multiplier and MUX based flip-flop and is simulated with Tanner Spice with 0.18 μm technology. The filter is designed for the cut off frequency of 2 KHz with the sampling rate of 16 KHz. The digital input corresponding to 2 KHz is fed to the FIR filter and the output is obtained from the final adder. The simulated output waveform is exhibited in Figure 5.21 (a) and (b).

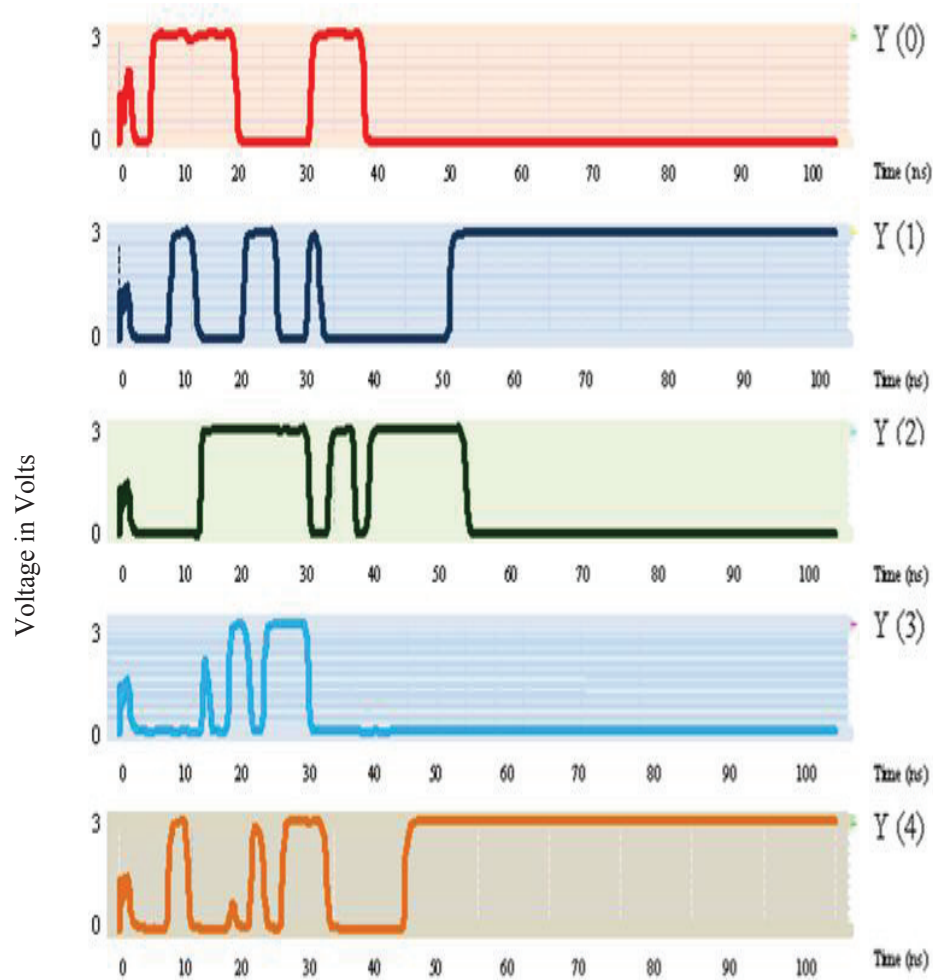


Figure 5.21(a) Output waveform of FIR filter

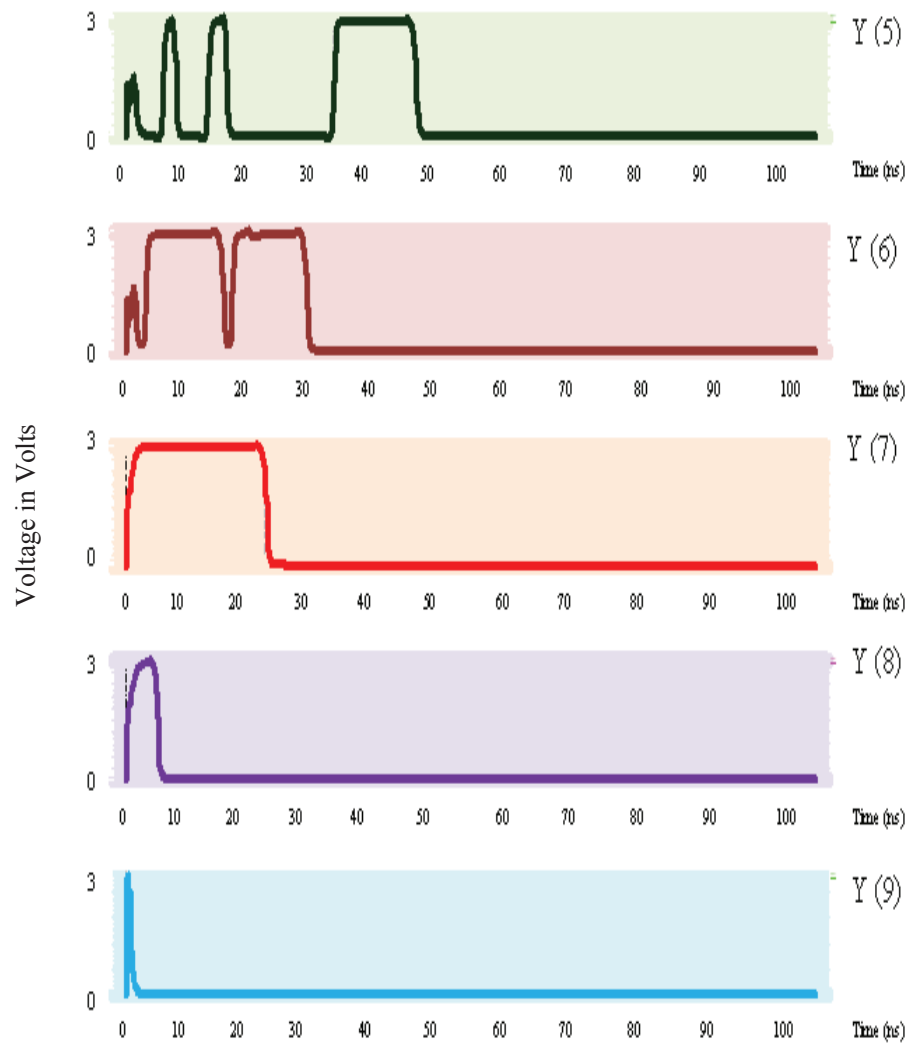


Figure 5.21(b) Output waveform of FIR filter

The power consumption of the FIR filter with low PDP adders and CMOS adder with proposed multiplier and array multiplier are shown in Table 5.9. The same input pattern is applied to the FIR structure designed with different multipliers and adders. The average power consumption and delay are computed through Tanner Spice.

Table 5.9 Performance comparison of 4 tap FIR filter with various adders and multiplier architectures

S.No.	Type of Adder used in FIR filter	Type of Multiplier used in FIR filter	Power Consumption (W)	Delay (μ s)	PDP (WS)	Area (No. of Transistors)
1	CMOS Adder (Jiang et al 2008)	Array Structure (Leonardo et al 2004)	0.98	1.5	1.47e-06	9752
2	CMOS Adder (Jiang et al 2008)	Binary Tree Architecture (Proposed)	0.873	1.1	0.96e-06	10536
3	MUX 12T adder (Proposed)	Array Structure (Leonardo et al 2004)	0.18	1.2	0.23e-06	4246
4	MUX 12T adder (Proposed)	Binary Tree Architecture (Proposed)	0.12	0.8	0.10e-06	4582
5	4T XOR adder (Wang et al 2004)	Array Structure (Leonardo et al 2004)	0.15	1.1	0.16e-06	3560
6	4T MUX based XOR adder (Wang et al 2004)	Binary Tree Architecture (Proposed)	0.14	0.8	0.11e-06	3840

The performance comparison of 4 tap FIR filter with various adder and multiplier architecture is shown in Table 5.8. The designed low pass FIR filter with low PDP adder, the proposed BT multiplier architecture and MUX based flip-flop required less power and time to produce the output than the existing conventional architecture. The average power consumption of FIR filter is shown in Figure 5.22.

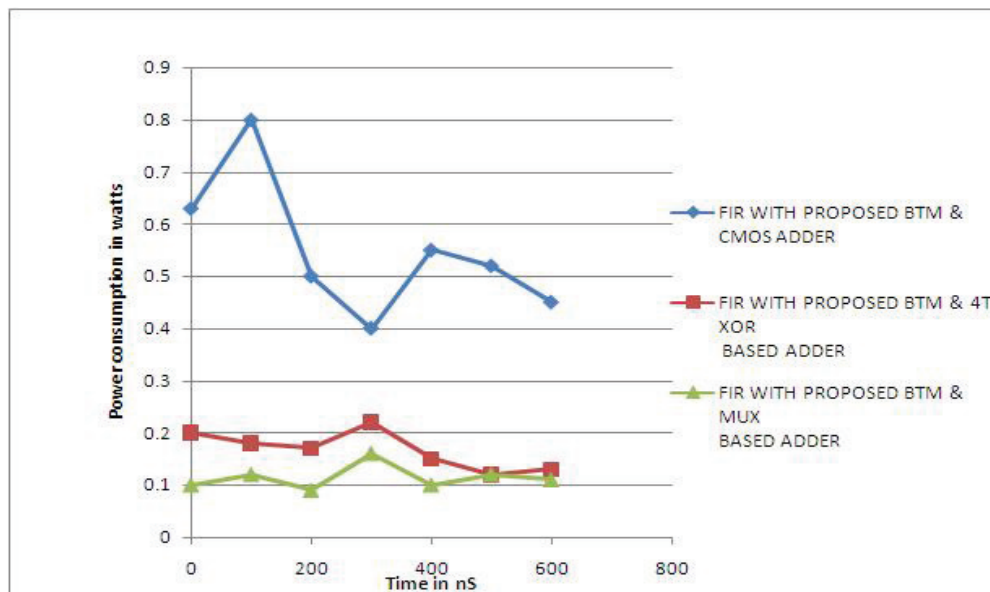


Figure 5.22 Average power consumption of FIR filter with different adders with BT multiplier structure

The number of transistors required and PDP of designed FIR filter with various adders and multiplier architecture are shown in Figure 5.23 and Figure 5.24. It is observed that FIR filter with the proposed adder and multiplier required 50% less PDP and optimum number of transistors than CMOS adder with array multiplier implemented in the FIR filter.

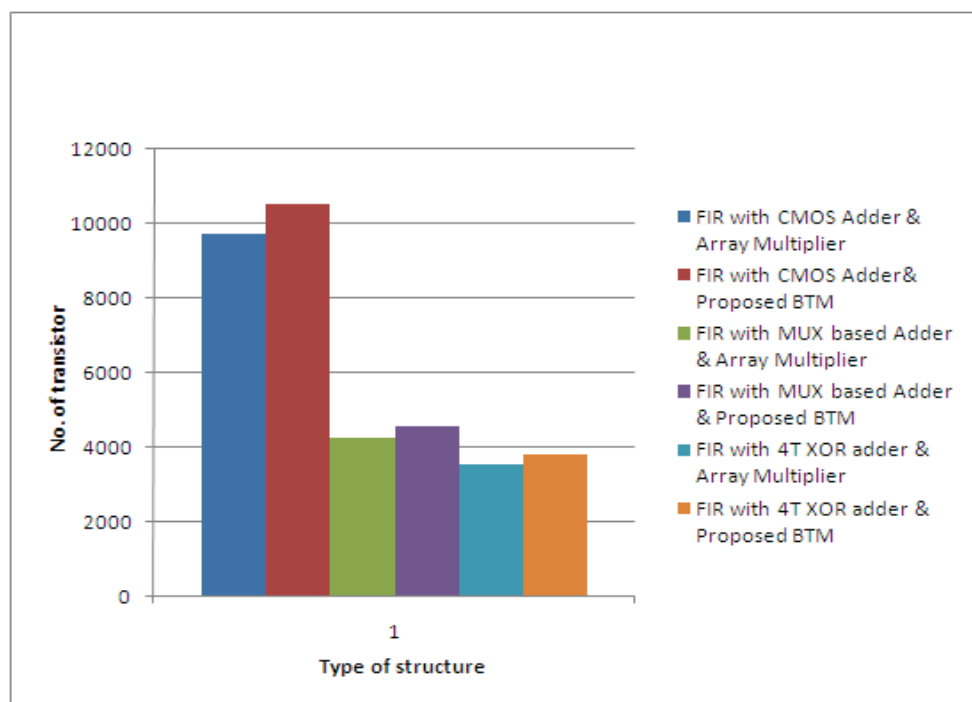


Figure 5.23 Number of transistors required for FIR filter

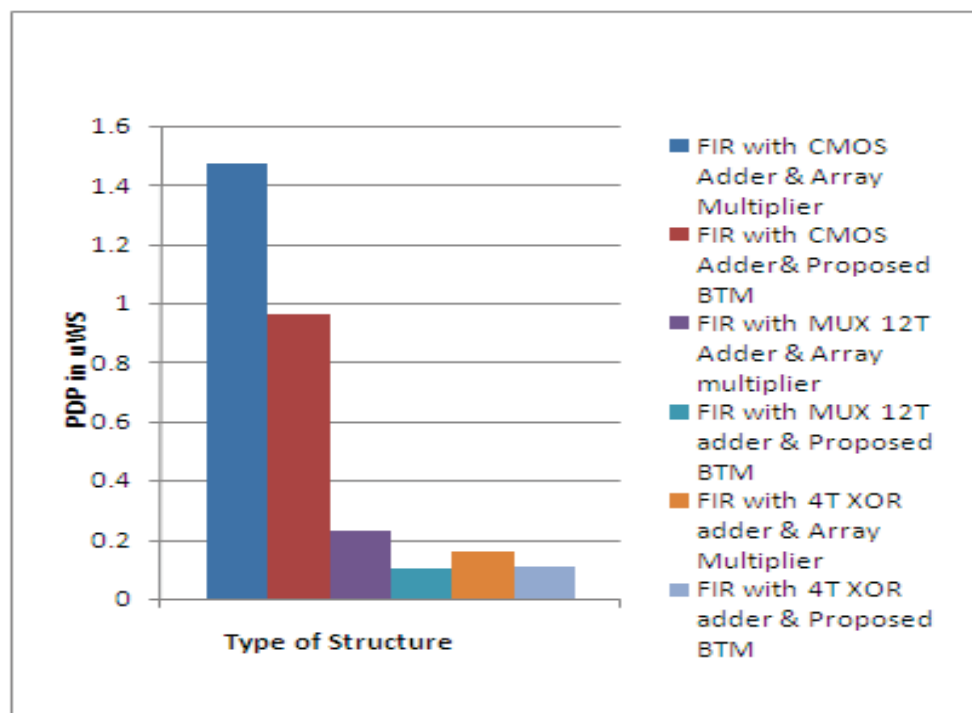


Figure 5.24 PDP of designed FIR filter

5.10 SUMMARY

The 4 tap low pass FIR filter was designed for 2 KHz signal with the sampling frequency of 16 KHz. The performance of various delay elements was compared with the proposed MUX based flip-flop and is used as delay element. Array multiplier and proposed binary tree multipliers are used for multiplication with ripple carry adders for addition. The FIR with array multiplier structure requires less transistors than the binary tree multiplier architecture, but it exhibits higher delay to produce the output. FIR with binary tree multiplier requires 50% less PDP with 7% rise in area. Hence the designed low pass FIR filter structure has an optimized circuit for high speed applications.