An Optimized Fir Filter Implementation Using High Speed Compressors and Counter

Meenaakshi Sundhari R.P[1] Ramya S[2] Karthikeyan S[3] Fathimasaffana A[4]

[1]Professor Department of ECE, PA College of Engineering and Technology, Pollachi
[2]Assistant Professor Department of ECE, KGiSL Institute of Technology, Coimbatore
[3]Professor Department of ECE, KGiSL Institute of Technology, Coimbatore
[4]PG Scholar Department of ECE, Hindusthan Institute of Technology, Coimbatore
Email: [1] rpmeenaakshi@gmail.com [2] ramya.t@kgkite.ac.in [3] karthickmrem@gmail.com
[4]afathimasaffana@gmail.com

Abstract
Multipliers, which are widely involved in various digital signal processors, are the essential elements in computer arithmetic. In a variety of computing applications, high-speed multipliers are becoming more popular. Approximate computing had given a lot of attention as a promising paradigm for minimizing power consumption, delay and area with increasing the accuracy. There are three steps to multiplication: (i) Creating partial product (PPs), (ii) Diminishing partial product and (iii) Calculation of result . Compressors minimize number of PPs stages of multipliers with more critical path delay, whereas adders require more stages to reduce partial product reduction. Counters have been used to get partial products in the multiplication process with lower critical path delay in the proposed work. Counters adds the partial product without using the previous carry signal, so it is considerably faster than traditional adders and compressors. The code is written in VERILOG, then simulated with MODELSIM and result is synthesized in Xilinx 14.2 ISE.

Keywords: High speed multipliers, partial product reduction, compressors, counters.

I.INTRODUCTION

Multipliers are widely utilized for high speed devices and multimedia applications due to their improved efficiency. The multiplier is situated in the crucial path of signal transmission in the majority of CPUs. In recent years, practical design requirements have been established to reduce the latency and complexity of these devices. The multiplication process is separated into three phases in a parallel multiplier. PPs are created first. And this products were added & procedure is repeated till two rows , the point at which the remaining 2 rows can added utilizing (CPA) carry propagation adder, for example, The partial product reduction tree (PPRT) was frequently utilized for fast summing of product in second stage, after the production of PPs. When full adder (FA) is used as the primary element for arrangements of different multiplication, it builds the basis for a wide range of architecture.

However, the use of modern parallel multiplier architecture is degraded by the inherent drawback of FA based systems, which is the transmission latency for the cascaded units. Key considerations for designing a parallel multiplier are circuit size, power consumption, and the distribution of used gates across the architecture. A compressor network, rather than FA trees, should be used to overcome such constraints, especially in the PPRT.

Hence, boosting the efficiency of this stage can greatly increase throughput while decreasing power consumption.

The n 2 compressor, as a general principle, alters the connections between nearby cells during the consolidation stage through employing 1 as well as multiple horizontally channels denoted by the letters cout1, cout2.The ways will send their logic value to the adjacent compressor, which is referred to as Cin1, Cin2, and so on, and has a binary bit greater degree of significance for the later compressor. Carry rippling, on the other hand, will emerge as newer challenge for such architectures that must be considered.

The tactics employed to accomplish this feat are essentially an art form involving the addition of integers with the minimum possible carry propagation latency. The fundamental idea is to utilize an FA to reduce a set of three numbers to a set of two. A 4-2 compressor is the simplest way to create a n2 compressor. The usual method of construction involves cascading the attachment of two FAs. Three FAs can be combined in a similar fashion to make a 5-2 compressor. As there are three FA blocks stacked on top of one another, the structure should have a latency of at least five XOR logical gates. Four XOR gates' worth of delay was cut thanks to the optimizations report. Several 5-2 compressor circuits exist, each with its own advantages and disadvantages. All previous studies share the same defect, where the horizontal outputs (Cout1 or Cout2) ripple throughout three iterations. With the provided circuits, the true gate level delay will be the product of five XOR logic gates. Also, the arrangement's typical design does not provide any speed benefits. Thus, compressor equivalent delays will be greater than 6 XOR logical circuits. Compressor blocks of higher orders can also ripple. Improving the performance of a n 2 compressor relies heavily on mitigating carry rippling issues between adjacent compressor cells.

However, for the 5-2 and 7-2 compressors, the researchers relied completely on that output quality due to the increasing complexity of the circuit-level design making computations more challenging. Changing the standard truth table by one cell in the compressor is the most effective solution to the carry rippling problem. The procedure begins with a new evaluation of the truth table's neutral states. Logic values can be assigned to these states to create the desired partitions in the table. This allows for a logical simplification of the pertinent output throughout the critical path of signal propagation.

II.BACKGROUND STUDY

In 0.35m CMOS technology, a unique 4-2 compressor with high speed and static,pass-transistor logic was proposed. Some modifications were done to the traditional 4-2 compressor truth table in order for decreasing delay and increase speed, which resulting in reducing parameter logic function. As a result, loss of energy was reduced. Furthermore, the latencies are the same because the pathways from all inputs to outputs are similar.

D.Balobas “Low-power high-performance CMOS 5-2 compressor with 58 transistors,” proposed Compressors are

essential components in CMOS multipliers' partial product reduction stage. A CMOS with 5-2 compressor design having total transistors of 58 was presented. In comparison to previous published methods,modeling outcomes shows suggested work having 5-2 compressor has dramatically enhanced effectiveness of power delay.

A. Najafi "High-speed energy-efficient 5:2 compressor," proposed Multipliers are essential components in determining effectiveness of arithmetic circuits as a whole.Essential part for multiplier is compressor. 5:2 compressor architecture was proposed which is based on modifying some internal equations. Suggested work utilizes few transistors. For comparison, three 5:2 compressors are considered.Suggested work is better in all aspects. Under a 1 V voltage supply, designs are modelled in 90-nm CMOS technology.

M. Tohidi, "CMOS implementation of a new high speed, glitch-free 5-2 compressor for fast arithmetic operations," It has been suggested to develop a revolutionary, high-speed, low-power 5-2 compressor that combines static logics and pass transistor logics in a creative way. some modifications to the original truth table was done to create new truth table. As a result, simple structures with lower middle-stage capacitances are obtained. As a result, there is a decrease in power dissipation and a decrease in total delay. In addition, input and internal driving issues have been drastically decreased.Not needed of additional buffering delay paths based on similar paths from inputs to outputs. As a result, power consumption will be reduced, & output waveforms will be glitch-free. Using voltage full swing logics in these systems has also improved the speed of cascaded operations.Suggested 5-2 compressor seems to have total latency of 302 ps and a power dissipation of 248.62 μw. It utilises TSMC 0.18 m CMOS technology and is simulated using HSPICE.

S. Mehrabi, K. Navi, "Performance comparison of high-speed high-order (n:2) and (n:3) CNFET-based compressors," presented a Compressor as one of the Processing Components, which is the main building block for collecting partial products during the computation . In this study, the designs and hardware specifications for different high-speed high-order compressors, such as 6:2 and 7:2, are compared with 6:3 and 7:3 compressors.These were analyzed and compared in latency and Power-Delay Product to enhance their performance (PDP). The comparison has been made by utilizing CNFET technology, which was recently implemented Full-Adder cell design at the transistor level. Synopsys HSPICE was used to run the simulations, which uses 32nm CNTFET technology and a 0.65 voltage supply. According to the simulation, the n:3structures have an advantage over other architectures with regards both transmission latency and energy usage by about 41% and 8%, respectively.

III. IMPLEMENTATION OF 5-2 AND 7-2 COMPRESSOR

The novel circuit is based on a modified truth table of the classic 5-2 compressor. A revised truth table is shown in Fig. 3.1.

Based on the logic state of Cin1 and Cin2, three different tables will be generated. Each table is split into quadrants reflecting the four distinct permutations of I1, I2, and I3 inputs indicated by the hatched regions.

In a straightforward design, Cin1 and Cin2 will have no bearing on the speed performance of Cout1 and Cout2 across all states. When Cout1 and Cout2 are produced, carry rippling is reduced because Cin1 and Cin2 originate from neighboring compressor cells. According to the truth table in Fig. 3.1, Cout1 will always increase to high voltage levels when at least two of the input signals have logic "1" values. This leads to the formula Cout1 = I12 + I13 + I23. The following formula can also be used to determine Cout2:

Cout2 =I4 or I5 I4 = I5

I1 xor I2 xor I3 I4 /= I5.

$C_{in1}=0$, $C_{in2}=0$

	n*	C_{out2}	C_{out1}	Carry	Sum		C_{out2}	C_{out1}	Carry	Sum	
$I_4=0$ $I_5=0$	0	0	0	0	0		0	0	0	1	$I_4=0$ $I_5=1$
	1	0	0	0	1		1	0	0	0	
	2	0	1	0	0		0	1	0	1	
	3	0	1	0	1		1	1	0	0	
$I_4=1$ $I_5=0$	0	0	0	0	1		1	0	0	0	$I_4=1$ $I_5=1$
	1	1	0	0	0		1	0	0	1	
	2	0	1	0	1		1	1	0	0	
	3	1	1	0	0		1	1	0	1	

$C_{in1}=0$, $C_{in2}=1$ / $C_{in1}=1$, $C_{in2}=0$

	n*	C_{out2}	C_{out1}	Carry	Sum		C_{out2}	C_{out1}	Carry	Sum	
$I_4=0$ $I_5=0$	0	0	0	0	1		0	0	1	0	$I_4=0$ $I_5=1$
	1	0	0	1	0		0	0	0	1	
	2	0	1	0	1		0	1	1	0	
	3	0	1	1	0		0	1	0	1	
$I_4=1$ $I_5=0$	0	0	0	1	0		1	0	0	1	$I_4=1$ $I_5=1$
	1	1	0	0	1		1	0	1	0	
	2	0	1	1	0		1	1	0	1	
	3	1	1	0	1		1	1	1	0	

$C_{in1}=1$, $C_{in2}=1$

	n*	C_{out2}	C_{out1}	Carry	Sum		C_{out2}	C_{out1}	Carry	Sum	
$I_4=0$ $I_5=0$	0	0	0	1	0		0	0	1	1	$I_4=0$ $I_5=1$
	1	0	0	1	1		1	0	1	0	
	2	0	1	1	0		0	1	1	1	
	3	0	1	1	1		1	1	1	0	
$I_4=1$ $I_5=0$	0	0	0	1	1		1	0	1	0	$I_4=1$ $I_5=1$
	1	1	0	1	0		1	0	1	1	
	2	0	1	1	1		1	1	1	0	
	3	1	1	1	0		1	1	1	1	

* n is number of I_1 , I_2 and I_3 inputs in which they are equal to logic 1.

Fig. 3.1. Truth table that is suggested for use with the 5-2 compressor.

According to (21), If an odd number of inputs have a logic "1" value, then the Sum output will only attain a high level voltage.

Sum = I1 xor I2 xor I3 xor I4 xor I5 xor Cin1 xor Cin2. (11)

To cover all possible Cin1 and Cin2 combinations, three alternative states must be examined for Carry output production.

If Cin1 = Cin2 = 1, output of carry was "1" as shown in Fig. 3.1.Output of carry was 0 when Cin1 = Cin2 = 0. But if Cin1 /= Cin2, then we have

Carry =I1 xor I2 xor I3 I4 = I5

I1 xor I2 xor I3 I4 /= I5

It shows that the value of Carry will be determined by the logic XOR/XNOR of the inputs I1, I2, and I3.

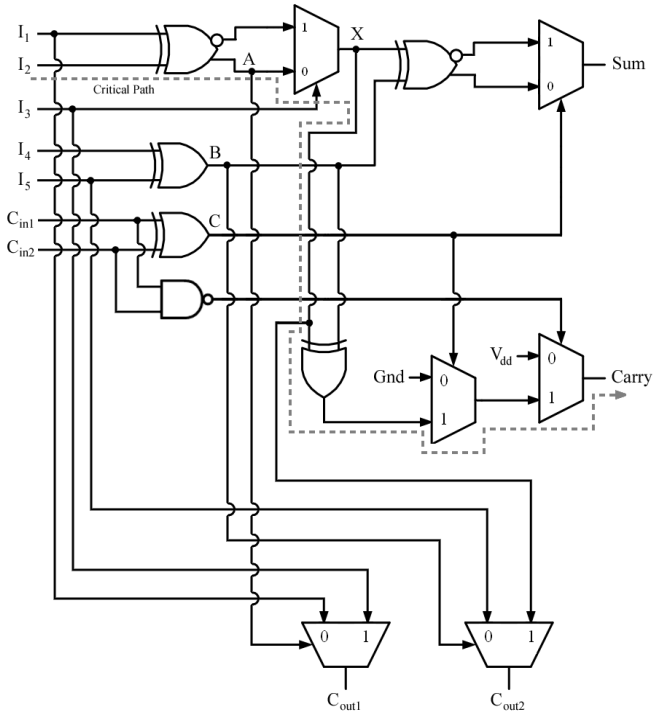


Fig. 3.2. Proposed 5-2 compressor

In Fig. 3.2, the proposed new circuit for a 5-2 compressor is shown. As the carry rippling concludes in the second compressor

block, two neighbouring 5-2 compressor cells must be coupled together for better realisation of latency in distinct pathways from inputs to outputs. This topic is covered in detail in the simulation section. Except for the MUX gates that are fed by Gnd and those that output Cout1, all of the MUX gates in Fig. 3.2 are channel-ready gates. They will thus encounter a 0.25 delta lag. The non-full swing XOR gate that creates B also employs an inverter at its output terminal to create Bbar. Yet, it has no impact on the overall latency of the system. At the TG's input stage, the NMOS transistor for high logic values will charge the output capacitor to Vdd-Vth. Simultaneously, the output of the inverter gate is connected to Bbar. When the output capacitor is charged to Vdd with the help of Bbar, the PMOS transistor is turned on.

The dual output XNOR-XOR gates produce two outputs simultaneously. Differences in their outputs have no bearing on the delay and glitch effect for TG waveforms because their paths are so similar. Static CMOS has also been used to create the architectures of NAND and NOR gates. The AND/OR gates do not impact system latency since they are placed outside of the critical path. If the outputs of the AND/OR gates were reversed, the gates would perform as intended. T crucial path, as was previously said, refers to the path that begins at I1 and I2 and finishes at the output of carry.

Literature review indicates that 7-2 compressor circuits have been designed with less attention paid to optimizing design concerns. In addition, the carry rippling issue is still present at three levels deep. Even though the crucial path uses nearly six cascaded XOR gates due to careless design, only Daniela Di Sclafan and Giuseppe Caruso were able to solve the rippling issue. The 7-2 compressor's improved architecture has made it possible to employ the same method as the 5-2 compressor. Carry ripple between adjacent blocks is often reduced by one level compared to prior designs, which is correlated with increased speed. As can be seen in Fig. 3.3, the latency between inputs and outputs is minimized by using less than four XOR logic gates.

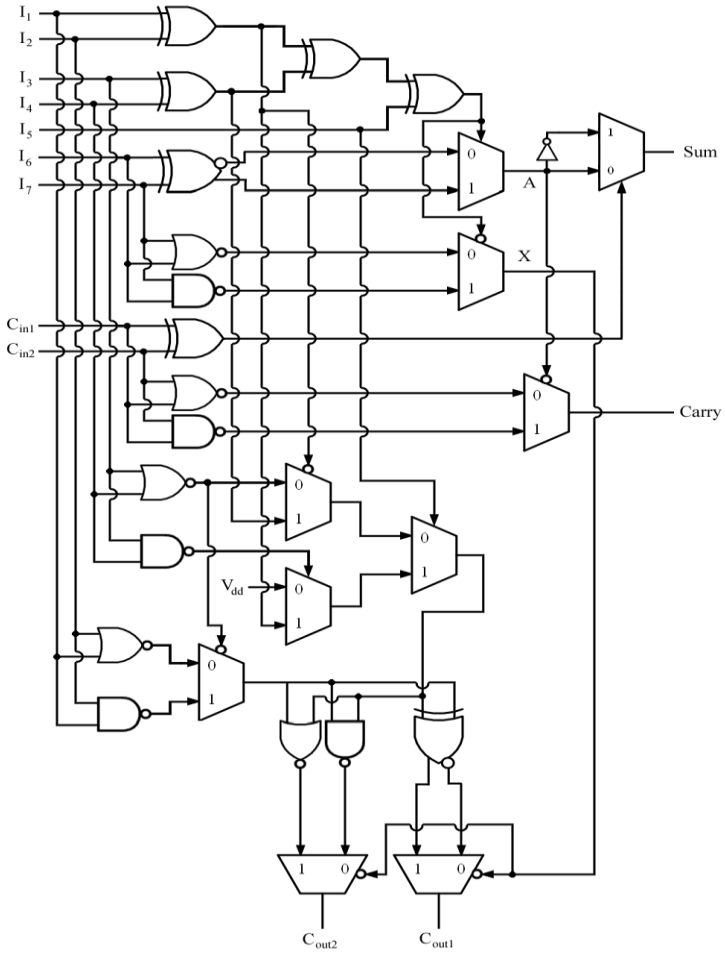


Fig. 3.3. Newly proposed 7-2 compressor.

The typical application platform 16x16 bit multiplier architecture was designed.. We made this architectural decision in order to

properly implement the suggested 5-2 and 7-2 compressor blocks in the body of PPRT. The suggested architecture is shown in Fig. 3.4. With AND gates, the PPs are created in their general form. The PPs will then be divided into two rows using a two-step approach. The ultimate outcome has been obtained using a 32-bit carry-lookahead adder , which is taken from. The PPRT of the multiplier was first implemented using our created blocks for a fair comparison analysis, and then it was modified using the previously published works for 5-2 and 7-2 structures.

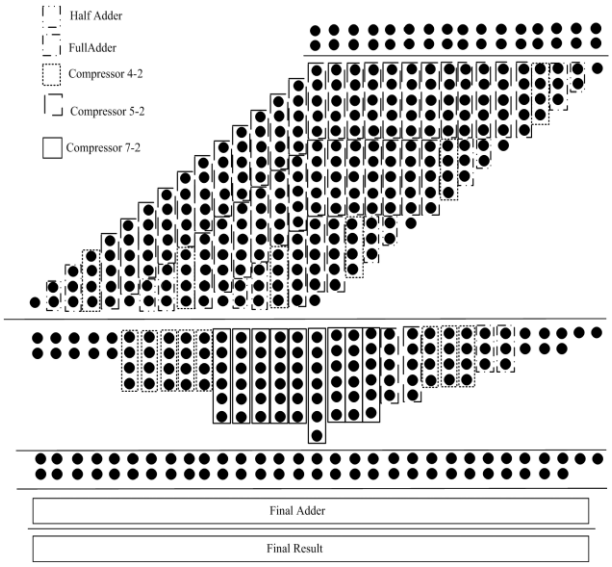


Fig. 3.4 The suggested 16-bit multiplier's architecture.

IV. SUGGESTED SYSTEM

4.1. BIT STACKING BASED ON SYMMETRIC METHOD

A 6:3 counter was generated by stacking each input bit so that all "1" bits were in the same place. By stacking the input bits and then translating the stack to a binary count, the 6-bit count can be generated. Initially, 3-bit stacks were constructed using tiny stacking circuits. Two 3-bit stacks were connected symmetrically to form a 6-bit stack, which required additional circuitry..

4.2. CIRCUIT OF 3-BIT STACKING

A 3-bit stacker circuit with X0, X1, and X2 inputs can produce Y0, Y1, and Y2 as outputs with the same amount 1 bits with in outputs as the inputs, but 1 bits were clustered along a left following with 0 bits. The outputs, it is evident,

$Y0 = X0 + X1 + X2$

$Y1 = X0X1 + X0X2 + X1X2$

$Y2 = X0X1X2.$

When one input is "1",1st result yield "1",when 2 inputs are one,2nd result yield "1",similarly when 3 inputs are one ,3rd result will yield "1".A output Y1 seems to be dominant expression that was constructed with just 1 CMOS gate.

Figure 1 depicts circuit of 3-bit stacking.

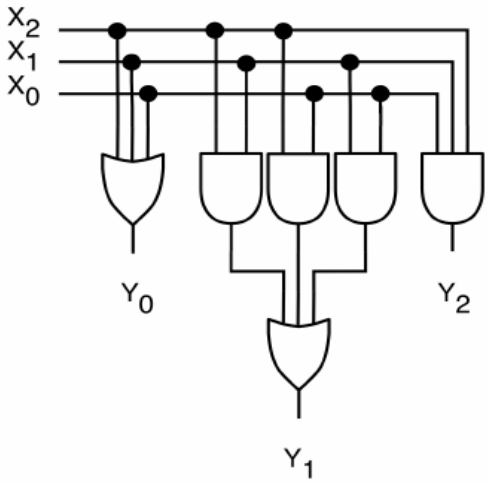


Fig.4.2 . Circuit of 3-bit stacking

4.3. MERGING STACKS

The 6-bit stacking circuit was developed by expanding the 3-bit stacking circuit. Using 3-bit stacking circuits, we split the six inputs X0,..., X5 into two sets of three bits. The X0, X1, and X2 signals are stacked as H0, H1, and H2, whereas the X3, X4, and X5 signals are I0, I1, and I2. Now, by inverting the outputs of the first stacker, we have a look at a six-bit representation of H2, H1, H0, I0, I1, and I 2. You can see an example of this method in action in the upper part of Fig. These six bits include a sequence of "1" bits, which are bordered by "0" bits. The sequence of "1" bits in a stack should begin with the leftmost bit. Two more 3-bit vectors of bits labeled J0, J1, J2,K0, K1, K2 are produced to create the necessary 6-bit stack. The J vector must be fully populated with ones before moving on to the K vector. Then

J0 = H2 + I0
J1 = H1 + I1
J2 = H0 + I2.

1st 3 "1" bits of train were assured for filling J bits in this manner, even if they are not stacked correctly. As assure that zero bits were listed separately, a K bits were created with identical inputs as before, but this time with logic AND gates.

K0 = H2 I0
K1 = H1 I1
K2 = H0 I2.

As the inputs to the AND gate were three positions apart, every bit k was zero unless the string of ones was no more than three positions longer. A few AND gates would get two inputs as "1" s if the train were longer than three positions, but this wouldn't happen if the inputs were only three positions apart. The number of "1" s in the train is three less than the number of "AND" gates with this feature. In this case, the J bits would be filled with ones before the K bits, even if both groups of bits started off with the same number of "1" bits. You may now stack J0 J1 J2 and K0 K1 K0 thanks to the addition of two more 3-bit stacking circuits. The outputs of these two circuits can be stacked together to get the values Y5,..., Y0.

4.4. BIT STACK TO BINARY NUMBER CONVERSION

Above mentioned 6 bit stack can translated as bit no to create a 6:3 counter circuit. utilising H, I, and K as middle values can rapidly calculate every final bit is not need last layer of stackers for a quicker, better effective count. C2, C1, and S are binary forms for no of "1" input bits, and C2, C1, S was binary expression for no of 1 input bitsTo calculate S, we only need to determine the parity of the results from the top layer of 3-bit stackers. If X0, X1, and X2 contain zero or two "1" bits, even parity occurs in the H. He and Ie, thus, signify even parity in the H and I bits,

He = + H1
Ie = + I1

Given that S has odd parity for all of the input bits and that the sum of two values with different parities is odd, we can calculate B0. S = He ^ Ie.

Despite the fact that there is 1 X-OR gate delay, this really isn't on a significant cricitical path. Whenever count is 2, 3, or 6, we use C1 = 1 to calculate C1. As a

result,2 scenarios exist. To begin, first determine if have at least 2 or 3 total inputs. For this H, I, and K intermediate vectors can be used. To check for at least two inputs, we need two stacks of length one, which is provided by either a top-level stacker or two stacks of length one.

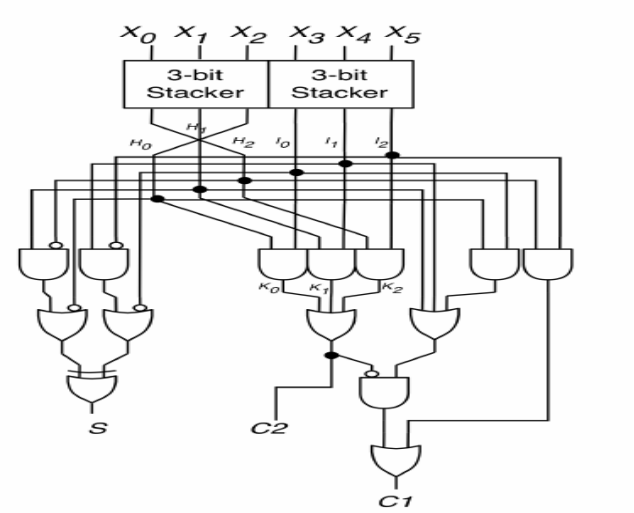


Fig.4.4 Symmetrically stacked 6:3 counter

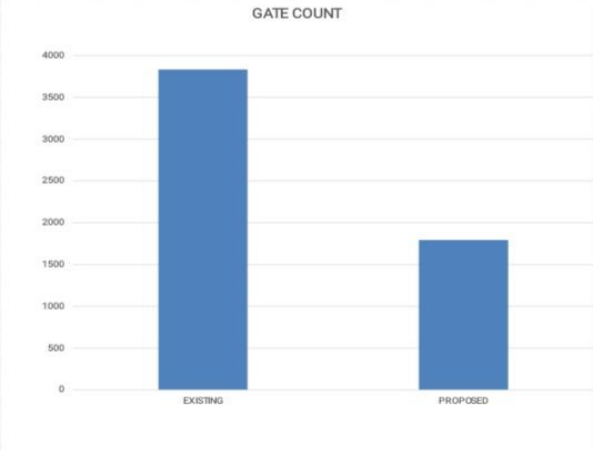
A complete 6:3 counter circuit might be constructed, as shown in Fig.3.6. By utilising larger CMOS gates, the critical route time is reduced by seven fundamental gates. A lack of X-OR gates on the critical route allows the 6:3 counter to outperform traditional architectures. Since there are 0 X-OR gates on its critical path, the proposed 6:3 counter based on bit stacking runs about 30% faster than any other counter designs. It is possible to create a counter with a considerable performance gain while using less power thanks to this novel method of counting that utilises bit stacking.

V. RESULTS AND DISCUSSION

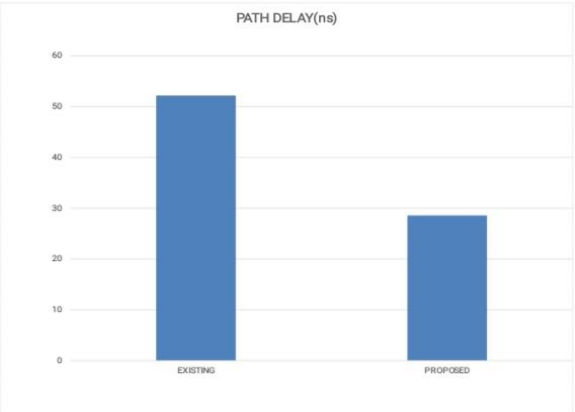
TABLE 5.1. PARAMETER COMPARISON

WORK	EXISTING	PROPOSED
Gate Count	3828	1788
Path Delay	52.119ns	28.520ns
Power Consumption	298mW	274mW

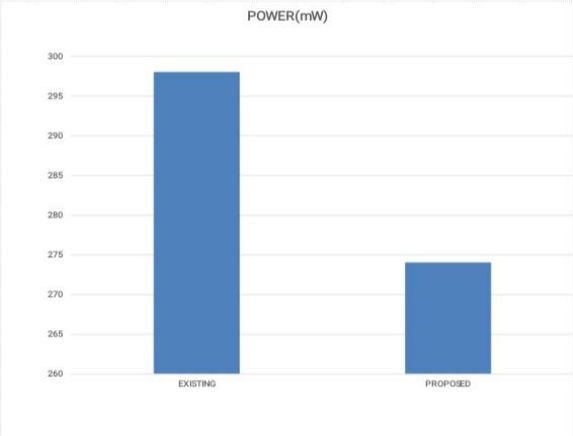
5.2. COMPARISON CHART FOR GATE COUNT:



5.3 COMPARISON CHART FOR PATH DELAY



5.4. COMPARISON CHART FOR POWER CONSUMPTION



VI. IMPLEMENTATION:

6.1. FIR FILTER

In signal processing, a filter is said to have a finite impulse response (FIR) if its impulse response decays to zero in a finite time. Every desired digital frequency response can be implemented with a FIR filter. These filters typically generate their output with the help of a multiplier, an addition circuit, and a series of delays. Filter design is the procedure by which the filter's length and coefficients are determined. In this study, we present a multiplier, an adder, and a delay for use in the design of a fir filter that achieves state-of-the-art performance.

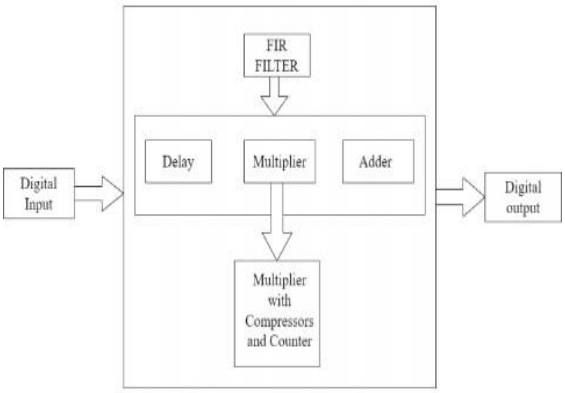


Fig.6.1. Proposed Fir Filter

VI.CONCLUSION AND FUTURE ENHANCEMENT:

The proposed work has been implemented using ModelSim and Xilinx.This process has been executed in FIR Filter, the features such as TIME, POWER and AREA has been reduced by use of Compressor along with counter.Furthermore, the hardware implementation in FPGA has been verified.

REFERENCES

1. Amir Fathi,Behbood Mashoufi (2020) “Very Fast, High Performance 5-2 and 7-2 Compressors in CMOS Process for Rapid Parallel Accumulations” IEEETrans.Electron.,vols.E28,no.6,pp.1403-1412.
2. C.-H. Chang, J. Gu (2004) “Ultra low-voltage low-power CMOS 4-2 and 5-2 compressors for fast arithmetic circuits,” IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 51, no. 10, pp. 1985–1997.
3. P. Saha, P. Samanta (2016) “4:2 and 5:2 decimal compressors,” in Proc. 7th Int. Conf. Intell. Syst., Modelling Simulation (ISMS).
4. C. Pan, Z. Wang (2013) “High speed and power efficient compression of partial products and vectors,” J. Algorithms Optim., Oct., vol. 1, no. 1, pp. 39–54.
5. V. G. Oklobdzija, D. Villeger (1996) “A method for speed optimized

parttial product reduction and generation of fast parallel multipliers using an algorithmic approach," IEEE Trans. Comput., vol. 45, no. 3,pp. 294–306.

6. M. Rouholamini, O. Kavehie (2007) "A new design for 7:2 compressors," in Proc. IEEE/ACS Int. Conf. Comput.Syst. Appl. (AICCSA).
7. Jiungmin Gu and Chip-Hong Chung(2003) "ultra low voltage, low power 4-2 compressor for high speed multiplications" IEEE International Symposium on Circuits and Systems (ISCAS), 2003, pp. II-661
8. R.P Meenaakshi Sundhari(2019) "An efficient implementation of low-power approximate compressor–based multiplier for cognitive communication systems'',International Journal of communication Systems,2019, vol. 35,
9. Zhongde Wang,G.A Jullien and W.C Miller(1995),"A New Design Technique for column compression multipliers," in IEEE Transaction on Computers,vol.44,no.8,pp.962-970.
10. P.Mokrian,G.M.Howard,G. Jullien and M.Ahma di(2003),"On the use of 4:2 compressors for partial product reduction," Canadian Conference on Electrical and Computer Engineering. Toward a Caring and Humane Technology (CCECE), vol.1. ,pp.121-124.