Journal of Theoretical and Applied Information Technology
10th July 2014. Vol. 65 No.1
© 2005 - 2014 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

MODIFIED UNIVERSAL SHIFT REGISTER BASED LOW POWER MULTIPLIER ARCHITECTURE

[1]S. P.VALAN ARASU, [2]Dr.S. BAULKANI
[1]A.P. (Senior Grade), Department of ECE
Dr. Sivanthi Aditanar College of Engineering, Tiruchendur -628215
[2]Associate Professor, Department of ECE,
Government College of Engineering, Tirunelveli – 627007.
E-mail: [1]valanarasu0568@gmail.com

ABSTRACT

In this article, a low power 8x8 shift-add multiplier architecture called Universal Shift Register (USR)-Multiplier is proposed. The proposed shift-add multiplier architecture concentrates on minimizing the switching activities of partial products in conventional shift-add multiplier. The switching activities are minimized by bypassing the multiplication operations of Zero's ('0') in Multiplier (A) with Multiplicand (B). This technique of bypassing reduces power consumed by the multiplier architecture. The proposed low power 8x8 multiplier adopts a Universal Shift Register (USR) with clock control unit such that the bypassing of zero's in 'B' is realized. Our proposed multiplier is designed and simulated using Xilinx ISE tools and compared with the power consumption of the conventional Shift-and Add multiplier, BZ-FAD shift-and add multiplier and ET shift-and add multiplier for the 8-bit multiplication results. Also, our multiplier is designed and simulated using Xilinx ISE tools and compared with the power consumption of the existing Booth multiplier, Wallace multiplier, Vedic Multiplier, BZ-FAD Shift-and add multiplier for the 16-bit multiplication results.

Advantages- Results show a reduction in switching activity and a reduction of more than 41% of the total power consumption comparing with conventional Shift-and Add multiplier.

Keywords: *Universal Shift Register, Shift-Add Multiplier, Switching activity, Partial product, BZFAD (Bypass Zero Feed A Directly).*

1. INTRODUCTION

Advances in VLSI technology have lead to the growth of more integrated signal processing systems. Among the arithmetic operations, multiplication plays an important role in DSP applications like arithmetic coding, wavelet transformation, digital filtering [1] and various other applications. Several computer arithmetic techniques exist to implement a digital multiplier and most among them involve computing a set of partial products, and then summing the partial products together to get the product of the two input values. There are more techniques for reducing the partial product in order to make the operation fast [2]. Numerous algorithms and methods are proposed for efficient multiplier implementation such as Booth's Algorithm [3], Braun and Baugh Wooley [4]. The Booth's algorithm works by multiplying two signed binary numbers using 2's complement notation. Braun is an array multiplier designed to meet the growing computational complexities in DSP applications.

The performance of a system is mostly contributed by the performance of the multiplier [5] because the multiplier is the slowest element in the system. Area and speed are usually inversely proportional to each other hence, improving the speed of the system results in an increase in area. The requirement for low-power VLSI system arises from two main factors. 1) due to the steady increase in operating frequency and processing capacity per chip, large amount of current has been used and this leads to large power consumption and hence the need for proper cooling technique for the removal of the generated heat arises. 2) the limited battery life in portable or mobile electronic devices and the continuous advancements in microelectronic technologies motivates the system designers to go for better use of energy, effective encoding of data, more reliable transmission of information, etc. Hence, in today's scenario, the system designers are more concerned about the low-power consumption to meet the requirements of various portable and mobile applications [10].

Among the various obstacles in designing an efficient multiplier the main problem it faces is the power consumption [19] [20]. Many researches have been done for reducing the power dissipation of different multipliers. The greatest and remarkable contribution for the total power consumption in a multiplier is the generation of partial product [6]. Among the multipliers, tree multipliers [7] plays an important role in high speed applications such as in the implementation of DSP filters, but their biggest drawback is the requirement of larger area. The carry-select-adder (CSA)-based radix multipliers [8] have lower area overhead, but this multiplier employs a greater number of active transistors for the multiplication operation leading to high power consumption. The shift-and-add multipliers have been used in various other applications because of their simplicity and small area requirement [12]. The design for higher-radix multipliers are faster but consume more power due to the usage of wider registers, and due to their more complex logic, require more silicon area. Power dissipation in most of the microelectronic systems is due to the switching activities. By reducing the switching activities overall power consumption can be reduced [9]. For the multiplier architecture proposed in this paper our main contribution includes 1) A modified Universal Shift Register and 2) A Johnson counter based USR clock control logic. Our contributions reduces the major switching activities in the conventional shift add multiplier.

The rest of the paper is organized as follows. Section 2 reviews some existing works related to our work. Section 3 highlights the research methodology for our proposed work arrived from three major existing methods. Section 4 explains our proposed multiplier architecture and the blocks included in it separately. Section 5 experiments and reports our proposed multiplier and compares our multipliers efficiency with some of the existing methods and finally the conclusion is presented in section 6.

2. RELATED WORK

In this section some of the significant works based on Low Power Multiplier is discussed. Shiann-Rong Kuang and Jiun-Ping Wang [11] designed a power-efficient 16 times 16 Configurable Booth multiplier (CBM) that supported single 16-bit, single 8-bit, or twin parallel 8-bit multiplication operations. In order to efficiently reduce power consumption, a novel dynamic-range detector was developed to dynamically detect the effective dynamic ranges of two input operands. The detection result is used to not only pick the operand with smaller dynamic range for Booth encoding to increase the probability of partial products becoming zero but also to deactivate the redundant switching activities in ineffective ranges as much as possible. Moreover, the output product of their implemented multiplier could be truncated further thereby decreasing the power consumption by sacrificing a bit of output precision. For efficiently and correctly combining those techniques, some additional components, including a correcting-vector generator, an adjustor, a sign-bit generator, a modified error compensation circuit, etc., were also developed. Finally, three real-life applications were adopted for evaluating the power efficiency and error performance of the implemented multiplier. The results show that the designed multiplier is more complex than non-CBMs, but significant power and energy savings could be achieved. Furthermore, the designed multiplier maintained an acceptable output quality for those applications when truncation is performed.

M. Mottaghi-Dastjerdi *et.al* [12] designed a low-power structure called Bypass Zero, Feed A Directly (BZ-FAD) for shift-and-add multipliers. This architecture considerably reduced the switching activity of conventional multipliers. The modifications to the multiplier that multiplies ‘A’ by ‘B’ was done by the removing the shifting of the ‘B’ register, direct feeding of ‘A’ to the adder, bypassing the adder whenever possible, using a ring counter instead of a binary counter and by removing the partial product shift. The modified architecture used a low-power ring counter implemented by them. Simulation for 32-bit radix-2 multipliers resulted that the BZ-FAD architecture reduced the total switching activity up to 76% and power consumption up to 30% when compared to the conventional architecture. The designed multiplier could be used for low-power applications where the speed is not a primary design parameter.

Two reconfigurable recursive multipliers were designed by M. Azarmehr *et.al* [13]. Their architectures combined some of the flexibility of software with the high performance of hardware by implementing different levels of recursive

Journal of Theoretical and Applied Information Technology
10th July 2014. Vol. 65 No.1
© 2005 - 2014 JATIT & LLS. All rights reserved·
ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

multiplication schemes on a two-dimensional logarithmic number system (2DLNS) processing structure. The data were split into a number of smaller sections, where each section was converted to a 2-digit 2DLNS (2 bases) representation. The dynamic range reduction and logarithmic characteristics of computing with two orthogonal base exponents in that number system allowed multiplication to be implemented with simple parallel small adders. Their architectures were able to perform single and double precision multiplications, as well as fault tolerant and dual throughput single precision operations. The implementations demonstrate the efficiency of 2DLNS in multiplication intensive DSP applications and exhibited outstanding results in terms of operation delay and dynamic power consumption.

Ko-Chi Kuon and Chi-Wen Chou [14] designed a low power and high speed row bypassing multiplier. The primary power reductions were obtained by tuning off MOS components through multiplexers when the operands of multiplier are zero. Analysis of the conventional DSP applications showed that the average of zero input of operand in multiplier was 73.8 percent. Therefore, significant power consumption could be reduced by the designed bypassing multiplier. The implemented multiplier adopted ripple-carry adder with fewer additional hardware components. In addition, the implemented bypassing architecture could enhance operating speed by the additional parallel architecture to shorten the delay time of the designed multiplier. Both unsigned and signed operands of multiplier were developed. Post-layout simulations were performed with standard TSMC 0.18mm CMOS technology and 1.8 V supply voltage by Cadence Spectral simulation tools. Simulation results showed that the implemented design could reduce power consumption and operating speed compared to those of the conventional counterparts. For a 16*16 multiplier, the implemented design achieves 17 and 36 percent reduction in power consumption and delay, respectively, at the cost of 20 percent increase of chip area in comparison with those of conventional array multipliers. In addition, the implemented design achieved averages of 11 and 38 percent reduction in power consumption and delay with 46 percent less chip area in comparison with those counterparts for both unsigned and signed multipliers. The implemented design was suitable for low power and high speed arithmetic applications.

Aloke Saha *et.al* [15] designed a low-power, high-speed, layout-efficient 8b×8b unsigned parallel multiplier based on pair-wise algorithm with wave-pipelining. Simplified interconnection and data propagation in forward direction with no feedback in pair-wise multiplication technique was the key to achieve high-performance wave-pipelined multiplier. In the implemented design, normal process complementary pass-transistor logic was used to build all the leaf cells of combinational block. The input/output registers were designed with high-performance pulse-triggered true single-phase clocking flip flop. Post-layout simulation with Taiwan Semiconductor Manufacturing Company Limited 0.18 μm single-poly double-metal complementary metal oxide semiconductor technology using Tanner EDA V.13 showed that the designed multiplier works at 6.25 GHz clock frequency and achieves the throughput of 6.25 billion multiplications per second with average power dissipation of 18.54 mW and overall latency of 3.24 ns at 25°C temperature and at 2 V supply rail.

Jin-Hao Tu and Lan-Da Van [16] designed a pipelined reconfigurable fixed-width Baugh-Wooley multiplier design framework that provided four configuration modes (CMs): n*n fixed-width multiplier, two n/2 *n/2 fixed-width multipliers, n/2*n/2 full-precision multiplier, and two n/4*n/4 full-precision multipliers. Furthermore, low-power schemes including gated clock and zero input techniques were employed to achieve the power-efficient pipelined reconfigurable design. The designed power-efficient pipelined reconfigurable fixed width multiplier design not only generated a family of widely used multipliers but also leads to 10.59, 21.7, 28.84, and 31.58 percent power saving, on average, for n=8,16,24 and 32 respectively, compared with that of the pipelined reconfigurable fixed-width multiplier without using the low-power schemes. On the other hand, compared with non-reconfigurable pipelined multiplier, we could save 0.81, 12.46, 17.93, and 23.2 percent power consumption, respectively, for n=8, 16, 24 and 32.

Chu Yu *et.al* [17] designed the efficient implementation of a pipeline FFT/IFFT processor for OFDM applications. Their design adopted a single-path delay feedback style as the implemented hardware architecture. To eliminate the read-only memories (ROM's) used to store the twiddle factors, their design architecture applied a reconfigurable complex multiplier and bit-parallel multipliers to achieve a ROM-less FFT/IFFT processor, thus consuming lower power than the existing works. Their design required about 33.6K gates, and its power consumption was about 9.8mW at 20MHz.

Shiann-Rong Kuang *et.al* [18] designed a variable-latency floating-point multiplier architecture, that was compliant with IEEE 754-1985 and suitable for low-power applications. Their designed architecture separated the significant multiplier into the upper and lower parts, and predicted the carry bit, sticky bit, and significant product from the upper part. In the case of correct prediction, the computation of lower part was disabled and the rounding operation was significantly simplified so that the floating-point multiplication could be made to consume less power, and be completed early while maintaining the correct IEEE rounding and product. Their experimental results showed that the designed multiplier could save respectable power and energy when compared to the fast multiplier at the expense of slight area and acceptable delay overheads.

3. RESEARCH METHODOLOGY

Array multiplier is commonly known for its regular structure. Its circuit is mainly based on add and shift algorithm. By multiplying the multiplicand with single multiplier bit each partial product is generated. According to their bit position the partial product are shifted and added. The addition operation is done by a common carry propagate adder. A total of N-1 adders are needed where 'N' is the length of multiplier. The final multiplication product is attained in the last adder by using any fast adder like carry ripple adder. In this type of multiplication we need to add each and every partial product as there are multiplier bits. The main disadvantage of the array multiplier is the delay in computation with respect to the width of the multiplier. Hence, the speed of the multiplier is low also the overall power consumption and area is low. Booth recoding algorithm is the most common method for partial product generation in a multiplier. This algorithm works by reducing the number of partial products thereby compressing the adder tree. The key to Booth's technique is to divide the groups bit of multiplier into 3 parts: the beginning, the middle, or the end of a run of 1's thereby compressing the partial product to one half.

Shift-and-add multiplication is same as that of basic multiplication performed mathematically. In this method the multiplicand 'A' is added to itself 'B' times, where 'B' is the multiplier. Mathematically to multiply two numbers, the algorithm is to take the single digit of the multiplier from right to left one at a time. Then, the multiplicand is multiplied by a single digit of the multiplier and the obtained intermediate product is placed to the left of the past results in separate positions. For performing the complete operations to obtain the Quotient, the conventional architecture for shift and add multipliers undergo many switching activities. Because of these switching activities the dynamic power consumption is high in conventional architecture. In order to reduce the switching activity in the conventional multiplier architecture, low power architecture can be designed.

In this paper we have proposed a modified Universal Shift register based shift and add multiplier with reduced dynamic power consumption. Here, a low power 8 bit * 8bit multiplier in which a modified low power universal shifts register for loading the multiplicand will be used, this register will be used to output the shifted multiplicand bits to the adder when the bit value of the multiplier is '1' otherwise the USR is only simply shifted .This is realized by using a Johnson counter based logic gate control. Reduction of overall power consumption is achieved by our proposed multiplier.

4. PROPOSED METHOD

The key to our proposed low power multiplier architecture is a modified Universal Shift Register which swings between two modes of operations (Load and Shift Left). The overall block diagram of our proposed multiplier architecture is shown in Figure 1.

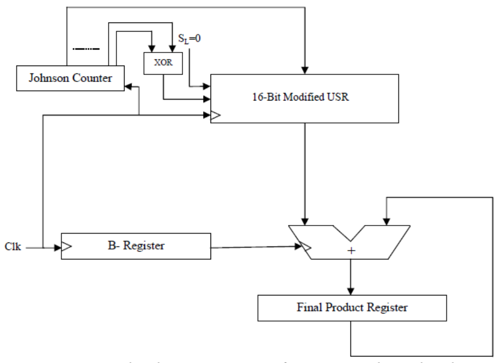


Figure 1. Block Diagram Of Proposed Multiplier

Proposed circuit design

Let us consider '*A'* is the 8 bit multiplicand and '*B'* is the 8 bit multiplier. In our architecture for shift add multiplication we adopt the left shift operation. The mathematical expression for our proposed multiplier expressed as below.

$$A = A_n A_{n-1} \ldots . A_0 \quad (1)$$

$$B = B_n B_{n-1} \ldots . B_0 \quad (2)$$

$$P = A \times B \quad (3)$$

Partial Product with left shift and add is given as

$$P^{(m+1)} = 2P^m + B_{n-m-1} A \quad (4)$$

Where,

P^m and P^{m+1} are the present and next partial Product obtained with respect to B_{n-m-1}

For our proposed technique we eliminate the addition operation when $B_{n-m-1} = 0$. Hence, in this case the partial product is given as

$$P^{(m+1)} = 2P^m \quad (5)$$

The final product is the partial product when $m+1=n$

Conventional Universal Shift Register

A universal shift register is an integrated logic circuit that can store binary data and can shift the data left or right by applying a clock signal [22]. It can transfer data in all possible modes such as serial-in serial-out (SISO), serial-in parallel-out (SIPO), parallel-in serial-out (PISO), parallel-in parallel-out (PIPO).

The function of the USR with respect to various input configurations is shown in Table 1 and Figure 2 shows the logic diagram of a conventional USR. For our work, we have modified this logic with the essential gate configurations in order to meet our requirement. The response of the input and output with respect to the clock pulse applied can be analyzed from the timing diagram shown in Figure 3.

Table 1.Function Table Of USR

Inputs							Outputs				
	Mode			Serial		Parallel					
$\overline{\text{MR}}$	M0	M1	Clk	Left	Right	X0…X7	Y0	Y 1	……..	Y 6	Y 7
0	x	x	x	x	x	x	0	0	….	0	0
1	x	x	0	x	x	x	Y 00	Y 10	….	Y 60	Y 70
1	1	1	↑	x	x	X0…X7	X0	X1	….	X6	X 0
1	1	0	↑	x	1	x	1	Y 0n	….	Y 5n	Y 6n
1	1	0	↑	x	0	x	0	Y 0n	….	Y 5n	Y 6n
1	0	1	↑	1	x	x	Y 1n	Y 2n	….	Y 7n	1
1	0	1	↑	0	x	x	Y 1n	Y 2n	….	Y 7n	0
1	0	0	x	x	x	x	Y 00	Y 10	….	Y 60	Y 70

Conventional USR consists of two selection modes M_0 and M_1 and it is configured with appropriate input so as to perform desired operation. The different mode of operation for USR is given in Table 2. When M_0M_1 = 00, the current value of the register is fed to D inputs of the flip-flops. This case forms a data transmission from the output of each flip-flop into the input of the same, so that the output recycle to the input in this mode of operation.

Table 2 .Modes Of Operation For USR

Mode		Operation
M_0	M_1	
0	0	No change
0	1	Shift Left
1	0	Shift Right
1	1	Parallel Load

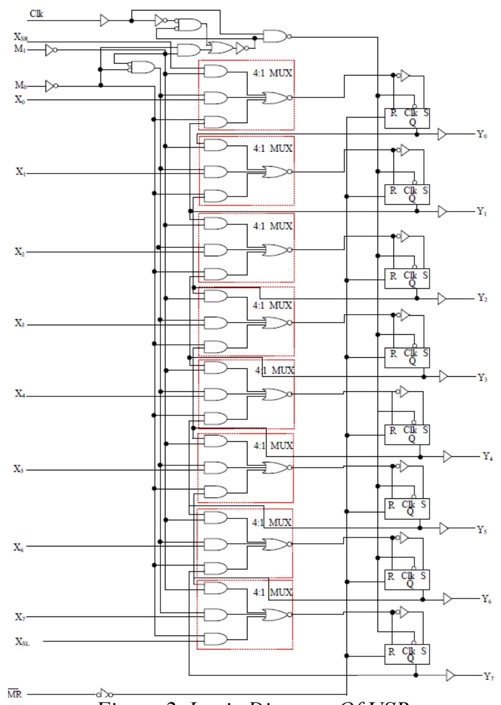


Figure 2. Logic Diagram Of USR

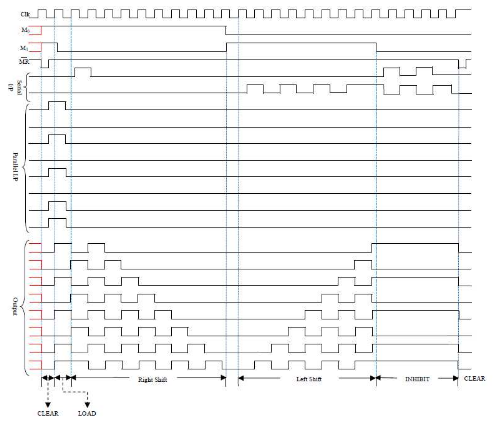


Figure 3. USR Timing Diagram

When M_0M_1 = 01, left shift operation with the serial input transferred to the flip-flop is performed. When M_0M_1 = 10, shift right operation is performed with other serial input going into the flip-flop. In the final case when M_0M_1 = 11, binary data from parallel input is transferred simultaneously into the register during next clock signal.

For our work we have modified the USR in such way it is low power, perform only left shift and Load and reduced area when compare to the conventional architecture. The architecture of the modified USR is shown in Figure 4. In our proposed modified USR the mode selector line '*S*' is controlled by the XOR output of 'J_0' and 'J_1' bits of the Johnson counter. The 4:1 multiplexer used in conventional USR are replaced with 2:1 multiplexers. The modes of operation for our proposed USR are shown in Table 3. These multiplexers helps in switching between two modes depending on the selector line '*S*'. When *S=0* the mode of operation is *load* and when *S=1* mode of operation becomes *Shift left*. The clocks for the first half of flip-flops (8-bits) are interconnected to tri-state buffer and then fed to the flip-flops. The buffers are controlled by the outputs from Johnson counter as shown in figure. Hence, depending on the Johnson counter output the flip-flops are enabled and this gated clock logic contributes to the reduction of switching activity. This gated clock logic holds the unnecessary flip-flops in the first half of the USR from shifting thereby reducing the switching activity.

Table 3. Mode Of Operation For Proposed USR

S	Operation
0	Load
1	Shift Left

The 'S_L' line is supplied with a low signal so that when shift left operation is performed '0' is filled in from the LSB. When comparing with the implementation of a multiplier with conventional USR the proposed 16-bit architecture of USR require only an 8 multiplexers thereby reducing the total area of the multiplier. The Johnson counter performs the control logic operation of the overall multiplication process. This counter is the modified form of ring counter, with the inverse output of the most significant flip-flop is fed to the input of the least significant flip-flop. The architecture of basic Johnson counter is shown in Figure 5.

The truth table of Johnson counter is shown in Table 3. Initially the counter is set to all '0' state by applying the clear signal .The flip-flops used here is a positive triggered D-flip-flop. From the table we can note that there is only one bit change from one state to another state for each clock pulse. Every bit starting from J_0 goes high after one state till J_7. When J_7 become high the J_0 bit will be '0'. This is because in the circuit the inverted output of J_7 is given as the feedback input to flip-flop J_0.

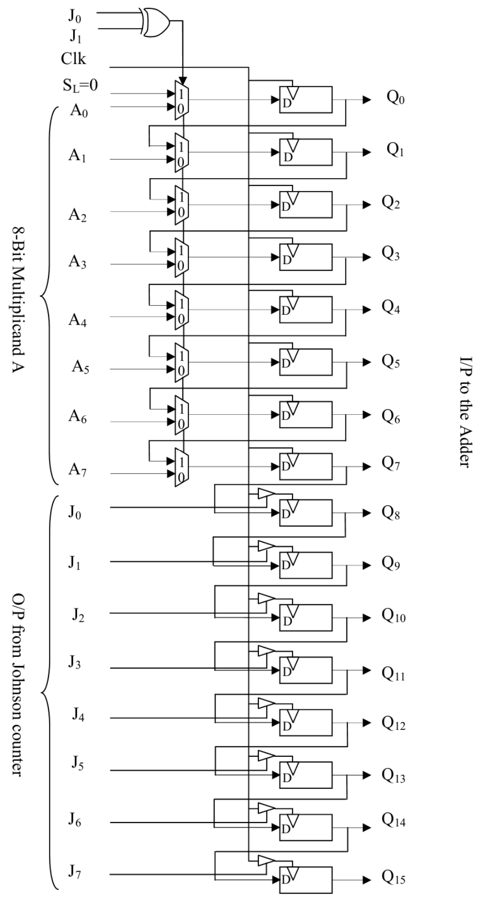


Figure 4. Proposed Modified Universal Shift Register

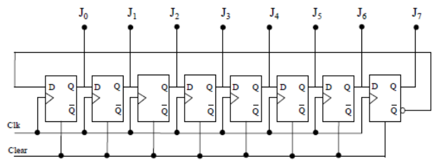


Figure 5.8-Bit Johnson Counter

Let A be the 8-Bit multiplicand and B is the 8-Bit multiplier. Initially A is given as input to the USR and B is loaded in the 8-bit B-register. The Johnson counter is initialized to all 0 states. When the first clock signal is applied, the selector line (*S*)

Table 4. Truth Table Of Johnson Counter

Clk	J_7	J_6	J_5	J_4	J_3	J_2	J_1	J_0
ω_0	0	0	0	0	0	0	0	0
ω_1	0	0	0	0	0	0	0	1
ω_2	0	0	0	0	0	0	1	1
ω_3	0	0	0	0	0	1	1	1
ω_4	0	0	0	0	1	1	1	1
ω_5	0	0	0	1	1	1	1	1
ω_6	0	0	1	1	1	1	1	1
ω_7	0	1	1	1	1	1	1	1
ω_8	1	1	1	1	1	1	1	1

Table 5. Illustration Of The Proposed Multiplier

Clock	$S=J_0+J_1$	Proposed USR	B-Register output	Operation	Final Product
1	1	0000000011010110	1	Load A,ADD	0000000011010110
2	0	0000000110101100	0	Shift Left	0000000011010110
3	0	0000001101011000	1	Shift Left ,ADD	000010000101110
4	0	0000011010110000	1	Shift Left ,ADD	000101011011110
5	0	0000110101100000	0	Shift Left	000101011011110
6	0	0001101011000000	1	Shift Left ,ADD	010 010110011110
7	0	0011010110000000	0	Shift Left	010 010110011110
8	0	0110101100000000	0	Shift Left	010 010110011110

of USR becomes 1 since $J_0=1$ and $J_1=0$. When $S=1$ all the A inputs are loaded to the Least significant USR flip-flops. At this clock pulse the remaining 8 flip-flops in the USR are clock gated and disabled as the control from the Johnson counter is 0. In the mean time, the B (0) value enables or disables the adder. i.e.) If B (0) =1, adder is enabled and if B(0)=0, adder is disabled and the addition operation is not included to the multiplication process. This leads to a reduction in overall dynamic power

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

consumption. The whole process is repeated for every bit value of B and when the Johnson counter shows all '1' state, the multiplication operation completes and the value stored in the result register gives the final product of the given inputs.

5. EXPERIMENTATION AND RESULTS

The multiplication process of our proposed multiplier is shown in Table.5 with the help of an example.

Let us consider,

A=11010110 (214)

B=00101101(45)

Initially, the USR is fed with 8-Bit multiplicand '*A*' and the B-register is loaded with 8-Bit multiplier '*B*' and the final product register is reset to '0'. When the first clock pulse triggers the Johnson counter, its J_0 become '1' and J_1 is '0' so the selector signal now becomes '1'. Since S=1 now the USR is in *Load* mode and the 'A' values are loaded to the USR. In the mean time the B-register which is provided with a synchronous clock is shifted right to make the LSB to pass to the adder circuit. In this case for first clock pulse, the output of B-register is '1' hence adder is enabled and the final product register value and USR values are added and the result is stored again in the final product register. The process is repeated for 8 clock pulses and the final value in the product register is '010 010110011110 (9630)'. The screenshot of simulation result for the above mentioned example is shown in Figure 6.

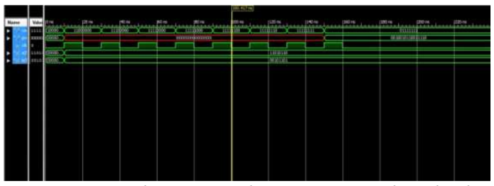
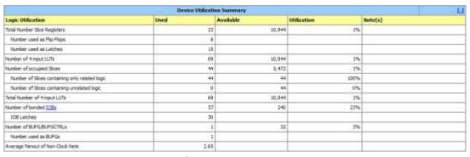


Figure 6. Simulation Result For Proposed Multiplier

Figure 7 exhibits the device utilization summary for our proposed multiplier.



Device Utilization Summary				
Logic Utilization	Used	Available	Utilization	Note(s)
Total Number Slice Registers	23	10,944	1%	
Number used as Flip Flops	8			
Number used as Latches	15			
Number of 4 input LUTs	69	10,944	1%	
Number of occupied Slices	44	5,472	1%	
Number of Slices containing only related logic	44	44	100%	
Number of Slices containing unrelated logic	0	44	0%	
Total Number of 4 input LUTs	69	10,944	1%	
Number of bonded IOBs	37	240	15%	
IOB Latches	30			
Number of BUFG/BUFGCTRLs	1	32	3%	
Number used as BUFGs	1			
Average Fanout of Non-Clock Nets	2.83			

Figure 7. Device Utilization Summary For Proposed Multiplier

Table 6. Comparison Between Power Consumption

Multiplier	Power(mW)	Saving (%)
Conventional Shift and Add Multiplier [21]	295	-
BZ-FAD Shift and Add Multiplier [21]	271	8.14
ET Shift and Add Multiplier [21]	228	22.71
Proposed Shift and Add Multiplier	172	41.69

From table 6 we can understand that switching activity of low power shift-and add multiplier saves 41.69% of power when compared to conventional shift and add multiplier and saves 36.53 % compared to BZ-FAD multiplier and when compared with ET Shift and Add Multiplier our method saves 24.54% of power. The major reductions in power consumption are due to the reduction in number of switching activities in the proposed shift- and add multiplier. A constant saving in leakage power is achieved since the adder is disabled during zero bit value of multiplier. Our design also contributes to reduction in delay by disabling the adder whenever the multiplier bit value is zero. Figure 8 shows a graph plotting the power consumption and power saved by different multipliers in Table 5.

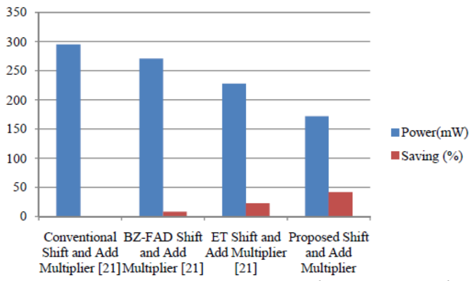


Figure 8. Power Consumption And Power Saved By Different Multipliers

Table 7 figure out the comparison of various existing 16×16 multiplier with our proposed multiplier for 16×16.Our method reduces the power consumption by 39.59% when compared with the array multiplier while the existing Booth multiplier, Wallace multiplier, Vedic Multiplier, BZ-FAD Shift-and add multiplier reduces it by 35.49%, 32.42%, 22.18% and 7.51% respectively.

Table 7. Comparison Between Power Consumption For 16-Bit Multipliers

Multiplier(16×16)	Power(mW)	Saving (%)
Array Multiplier [23]	293	-
BZ-FAD Shift-and add multiplier [23]	271	7.51
Vedic Multiplier [23]	228	22.18
Wallace multiplier [23]	198	32.42
Booth Multiplier [23]	189	35.49
Proposed Shift and Add Multiplier	177	39.59

6. CONCLUSION

Our proposed shift and add multiplier architecture using USR exhibits a low power consumption when compared with some of the existing architectures. The modified USR with reduced multiplier count and the logic for clock gating using Johnson counter contributed for reduced switching activity and in turn lead to reduced power consumption. The blocking of adder when the multiplier bit value is zero also contributed for the overall power consumption. The design can be verified using Xilinx software with Verilog code and power consumption is analyzed using Xilinx xpower analyzer tool. The proposed USR based multiplier exhibits a power reduction of more than 41% of the total power comparing with conventional Shift-and Add multiplier and also exhibits a power reduction of 36.53 % and 24.54% when compared to that of BZ-FAD multiplier and ET Shift and Add Multiplier. For 16 ×16 multiplier a power reduction of 39.59% when compared with the array multiplier and 35.49%, 32.42%, 22.18%, 7.51% comparing with that of existing Booth multiplier, Wallace multiplier, Vedic Multiplier, BZ-FAD Shift-and add multiplier was achieved.

REFERENCES

[1] Jer Min Jou and Shiann Rong Kuang "Design of low-error fixed-width multiplier for DSP applications", *Electronics Letters*, Vol. 33, No. 19, pp: 1597-1598, September 1997.

[2] Vojin G. Oklobdzija, David Villeger and Simon S. Liu, "A Method for Speed Optimized Partial Product Reduction and Generation of Fast Parallel Multipliers Using an Algorithmic Approach", *IEEE Transactions On Computers*, Vol. 45, No. 3, pp: 294-306, March 1996.

[3] Louis P. Rubinfield, "A Proof of the Modified Booth's Algorithm for Multiplication", *IEEE Transactions on Computers*, pp: 1014 and1015, October 1975.

[4] A. Bermak , D. Martinez and J.-L. Noullet, "High density 16/8/4-bit configurable multiplier", *IEE Proceedings Circuits Devices System,* Vol. 144, No. 5, pp: 272-276, October I997.

[5] C. Senthilpari, Ajay Kumar Singh and K. Diwakar, "Design of a low-power, high performance, 8*8 bit multiplier using a Shannon-based adder cell", *Microelectronics Journal*, Vol. 39, No 5, pp 812–821,May 2008.

[6] Peter-Michael Seidel, D. McFearin and David W. Matula , " Secondary Radix Recodings for Higher Radix Multipliers*", IEEE Transactions On Computers,* Vol. 54, No. 2, pp: 111-123, February 2005.

[7] P.G.McCrea and W.S.Matheson, "Design of high-speed fully serial tree multiplier", *IEE Proceedings on Computers and Digital Techniques*, Vol. 128 , No 1,pp:13 - 20, January 1981.

[8] Yajuan He and Chip-Hong Chang ,"A Power-Delay Efficient Hybrid Carry-Lookahead/Carry-Select Based Redundant Binary to Two's Complement Converter" , *IEEE Transactions On Circuits And Systems*, Vol. 55, No. 1, pp: 336-346, February 2008.

[9] Oscal T.-C. Chen, Sandy Wang, and Yi-Wen Wu, Minimization of Switching Activities of Partial Products for Designing Low-Power Multipliers IEEE Transactions on Very Large Scale Integration (VLSI) Systems, Vol. 11, No. 3, pp:418-433, June 2003.

[10] Kuan-Hung Chen and Yuan-Sun Chu, "A Low-Power Multiplier With the Spurious Power Suppression Technique", *IEEE Transactions On Very Large Scale Integration (VLSI) Systems*, Vol. 15, No. 7, pp:846-850, July 2007.

[11] Shiann-Rong Kuang and Jiun-Ping Wang, "Design of Power-Efficient Configurable Booth Multiplier", *IEEE Transactions On Circuits And Systems-I,* Vol. 57, No. 3, pp.568-580, March 2010.

[12] M. Mottaghi-Dastjerdi, A. Afzali-Kusha, and M. Pedram, "BZ-FAD: A Low-Power Low-Area Multiplier Based on Shift-and-Add Architecture", *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, Vol. 17, No. 2, pp.302-306, February 2009.

[13] M. Azarmehr M. Ahmadi G.A. Jullien and R. Muscedere,"High-speed and low-power reconfigurable architectures of 2-digit two-

dimensional logarithmic number system-based recursive multipliers", *IET Circuits Devices Syst*., Vol. 4, No. 5, pp. 374–381, 2010.

[14] Ko-Chi Kuon and Chi-Wen Chou,"Low power and high speed multiplier design with row bypassing and parallel architecture", *Elsevier Science Publishers, Microelectronics Journal,* Vol 41, No 10, pp.639-650, October, 2010.

[15] Aloke Saha1, Dipankar Pal andMahesh Chandra, "Low-power 6-GHz wave-pipelined 8b×8bmultiplier", *IET Circuits Devices Syst*., 2013, Vol. 7, No. 3, pp. 124–140,October2012.

[16] Jin-Hao Tu and Lan-Da Van, "Power-Efficient Pipelined Reconfigurable Fixed-Width Baugh-Wooley Multipliers", *IEEE Transactions on Computers*, Vol. 58, No. 10, pp.1346-1355, October 2009.

[17] Chu Yu, Mao-Hsu Yen, Pao-Ann Hsiung, Sao-Jie Chen A Low-Power 64-point Pipeline FFT/IFFT Processor for OFDM Applications *IEEE Transactions on Consumer Electronics*, Vol. 57, No. 1,pp.40-45, February 2011.

[18] Shiann-Rong Kuang, Jiun-Ping Wang, and Hua-Yi Hong, "Variable-Latency Floating-Point Multipliers for Low-Power Applications", *IEEE Transactions On Very Large Scale Integration (VLSI) Systems*, Vol. 18, No. 10, pp.1493-1497, October 2010.

[19] Manuel de la Guia Solaz, WeiHan, Member and Richard Conway, "A Flexible Low Power DSP With a Programmable Truncated Multiplier" , *IEEE Transactions On Circuits And Systems*, Vol. 59, No. 11,pp:2555-2568, November 2012.

[20] Brian S. Cherkauer and Eby G. Friedman, "A Hybrid Radix-4/Radix-8 Low Power Signed Multiplier Architecture", *IEEE Transactions on Circuits and Systems, Analog and Digital Signal Processing*, Vol. 44, No. 8, pp: 656-659, August 1997.

[21] K.N. Vijeyakumar, V. Sumathy ,Sriram Komanduri and C. Chrisjin Gnana Suji, "Design of Low- Power High-Speed Error Tolerant Shift and Add Multiplier", *Journal of Computer Science,* Vol. 12, No.7,pp:1839-1845,2011.

[22] Philips Semiconductors, "Datasheet for 74F198, A 8-bit bidirectional universal shift register", October 1987.