IJERST

International Journal of Engineering Research and Science & Technology

ISSN : 2319-5991
Vol. 4, No. 4
November 2015

www.ijerst.com

Email: editorijerst@gmail.com or editor@ijerst.com

International Journal of Engineering Research and Science & Technology

ISSN 2319-5991 www.ijerst.com
Vol. 4, No. 4, November 2015

Research Paper

IMPLEMENTATION OF POWER OPTIMIZED RAZOR-BASED MULTIPLIER

V L Prathyusha[1]* and N Suresh Babu[2]

*Corresponding Author: V L Prathyusha ✉ prathyusha.vakkalagadda@gmail.com

In this project a multiprecision (MP) reconfigurable multiplier that incorporates variable precision, Parallel Processing (PP), razor-based Dynamic Voltage Scaling (DVS), and dedicated MP operands scheduling to provide optimum performance for a variety of operating conditions. All of the building blocks of the proposed reconfigurable multiplier can either work as independent smaller-precision multipliers or work in parallel to perform higher-precision multiplications. Given the user's requirements (e.g., throughput), a dynamic voltage/frequency scaling management unit configures the multiplier to operate at the proper precision and frequency. Adapting to the run-time workload of the targeted application, razor flip-flops together with a dithering voltage unit then configure the multiplier to achieve the lowest power consumption. The single-switch dithering voltage unit and razor flip-flops help to reduce the voltage safety margins and overhead typically associated to DVS to the lowest level. The large silicon area and power overhead typically associated to reconfigurability features are removed. Finally, the proposed novel MP multiplier can further benefit from an operands scheduler that rearranges the input data, hence to determine the optimum voltage and frequency operating conditions for minimum power consumption.

Keywords: Razor multiplier, Clock domain, XILINX ISE, Verilog

INTRODUCTION

A binary multiplier is an electronic circuit used in digital electronics, such as a computer, to multiply two binary numbers. It is built using binary adders. A variety of computer arithmetic techniques can be used to implement a digital multiplier. Most techniques involve computing a set of *partial products*, and then summing the partial products together. This process is similar to the method taught to primary schoolchildren for conducting long multiplication on base-10 integers, but has been modified here for application to a base-2 (binary) numeral system.

Until the late 1970s, most minicomputers did not have a multiply instruction, and so programmers used a "multiply routine" which

[1] M.Tech Student, Department of E.C.E., Chirala Engineering College, Ramapuram Beach Rd, Chirala, Andhra Pradesh 523157, India.

[2] Associate Professor, Department of E.C.E., Chirala Engineering College, Ramapuram Beach Rd, Chirala, Andhra Pradesh 523157, India.

repeatedly shifts and accumulates partial results, often written using loop unwinding. Mainframe computers had multiply instructions, but they did the same sorts of shifts and adds as a "multiply routine". Early microprocessors also had no multiply instruction. The Motorola 6809, introduced in 1978, was one of the earliest microprocessors with a dedicated hardware multiply instruction. It did the same sorts of shifts and adds as a "multiply routine", but implemented in the microcode of the MUL instruction.The method taught in school for multiplying decimal numbers is based on calculating partial products, shifting them to the left and then adding them together. The most difficult part is to obtain the partial products,as that involves multiplying a long number by one digit (from 0 to 9):

123

x 456

=====

738 (this is 123 x 6)

615 (this is 123 x 5, shifted one position to the left)

+ 492 (this is 123 x 4, shifted two positions to the left)

=====

56088

A binary computer does exactly the same, but with binary numbers. In binary encoding each long number is multiplied by one digit (either 0 or 1), and that is much easier than in decimal, as the product by 0 or 1 is just 0 or the same number. Therefore, the multiplication of two binary numbers comes down to calculating partial products (which are 0 or the first number), shifting them left, and then adding them together (a binary addition, of course):

1011 (this is 11 in decimal)

x 1110 (this is 14 in decimal)

======

0000 (this is 1011 x 0)

1011 (this is 1011 x 2, shifted one position to the left)

1011 (this is 1011 x 4, shifted two positions to the left)

+ 1011 (this is 1011 x 8, shifted three positions to the left)

=========

10011010 (this is 154 in decimal)

This is much simpler than in the decimal system, as there is no table of multiplication to remember: just shifts and adds. This method is mathematically correct and has the advantage that a small CPU may perform the multiplication by using the shift and add features of its arithmetic logic unit rather than a specialized circuit. The method is slow, however, as it involves many intermediate additions. These additions take a lot of time. Faster multipliers may be engineered in order to do fewer additions; a modern processor can multiply two 64-bit numbers with 6 additions (rather than 64), and can do several steps in parallel. The second problem is that the basic school method handles the sign with a separate rule ("+ with + yields +", "+ with - yields -", etc.). Modern computers embed the sign of the number in the number itself, usually in the two's complement representation. That forces the multiplication process to be adapted to handle two's complement numbers, and that complicates the process a bit more. Similarly, processors that use ones' complement, sign-and-magnitude, IEEE-754 or other binary representations require specific adjustments to the multiplication process

High speed arithmetic operations are very important in many signal processing applications. Speed of the Digital Signal Processor (DSP) is largely determined by the speed of its multipliers. In fact the multipliers are the most important part of all digital signal processors; they are very important in realizing many important functions such as fast Fourier transforms and convolutions. Since a processor spends considerable amount of time in performing multiplication, an improvement in multiplication speed can greatly improve system performance. Multiplication can be implemented using many algorithms such as array, booth, carry save,and wallace tree algorithm. The computational time required by the array-multiplier is less because the partial products are computed -independently in parallel. The delay associated with the array multiplier is the time taken by the signals to propagate through the gates that form the multiplication array.

Arrangement of adders is another way of improving-multiplication speed. There are two methods for this: Carry Save Array (CSA) method and Wallace tree method. In the CSA method, bits are processed one by one to supply a carry signal to an adder located at a one bit higher position. The CSA method has got its own limitations since the execution time depends on the number of bits of the multiplier. In the Wallace tree method, three bit signals are passed to a one bit full adder and the sum is supplied to the next stage full adder of the same bit and the carry output signal is passed to the next stage full adder of same number of bit and the then formed carry is supplied to the next stage of the full adder located at a one bit higher position. In this method, the circuit lay out is not easy.

Booth algorithm reduces the number of partial-products. However, large booth arrays are required for high speed multiplication and exponential operations which in turn require large partial sum and partial carry registers. Multiplication of two n-bit operands using a radix-4 booth recording multiplier requires approximately n/(2 m) clock cycles to generate the least significant half of the final product, where m is the number of booth recoded adder stages. Thus, a large propagation delay is associated with this case. The modified booth encoded Wallace tree multiplier uses modified booth algorithm to reduce the partial products and also faster additions are performed using the Wallace tree.

RAZOR BASED MULTIPLIER

CONSUMERS demand for increasingly portable yet high performance Multimedia and communication products imposes stringent constraints on the power consumption of individual internal components. Of these, multipliers perform one of the most frequently encountered arithmetic operations in Digital Signal Processors (DSPs). For embedded applications, it has become essential to design more power-aware multipliers. Given their fairly complex structure and interconnections, multipliers can exhibit a large number of unbalanced paths, resulting in substantial glitch generation and propagation.

Each pair of incoming operands is routed to the smallest multiplier that can compute the result to take advantage of the lower energy consumption of the smaller circuit. This ensemble of point systems is reported to consume the least power but this came at the cost of increased chip area given the used ensemble structure. To address this issue, proposed to share and reuse some functional modules within the ensemble. In, an 8-bit multiplier is reused for the 16-bit multiplication, adding scalability without large area penalty. Reference extended this method by implementing pipelining to further improve the multiplier's performance.

A more flexible approach is proposed in, with several multiplier elements grouped together to provide higher precisions and reconfigurability. Reference analyzed the overhead associated to such reconfigurable multipliers. This analysis showed that around 10%-20% of extra chip area is needed for 8-16 bits multipliers. Combining multiprecision (MP) with Dynamic Voltage Scaling (DVS) can provide a reduction in power consumption by adjusting the supply voltage according to circuit's run-time workload rather than fixing it to cater for the worst case scenario. When adjusting the voltage, the actual performance of the multiplier running under scaled voltage has to be characterized to guarantee a fail-safe operation. Conventional DVS techniques consist mainly of lookup table (LUT) and on-chip critical path replica approaches.

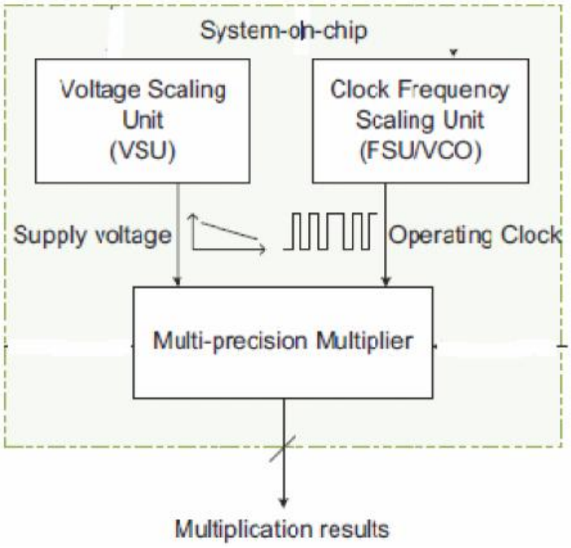
The LUT approach tunes the supply voltage according to a predefined voltage-frequency relationship stored in a LUT, which is formed considering worst case conditions (process variations, power supply droops, temperature hot-spots, coupling noise, and many more). Therefore, large margins are necessarily added, which in turn significantly decrease the effectiveness of the DVS technique. The critical path replica approach typically involves an on-chip critical path replica to approximate the actual critical path. Therefore, voltage could be scaled to the extent that the replica fails to meet the timing.However, safety margins are still needed to compensate for the intradie delay mismatch and address fast-changing transient effects. In addition, the critical path may change as a result of the varying supply voltage or process or temperature variations. If this occurs, computations will completely fail regardless of the safety margins. The aforementioned limitations of conventional DVS techniques motivated recent research efforts into error-tolerant DVS approaches, which can run-time operate the circuit even at a voltage level at which timing errors occur. A recovery mechanism is then applied to detect error occurrences and restore the correct data. Because it completely removes worst case safety margins, error-tolerant DVS techniques can further aggressively reduce power consumption. In this paper, we propose a low power reconfigurable multiplier architecture that combines MP with an error-tolerant DVS approach based on razor flip-flops .The main contributions of this paper can be summarized follows.

- A novel MP multiplier architecture featuring, respectively, 28.2% and 15.8% reduction in silicon area and power consumption compared with its conventional 32 × 32 bit fixed-width multiplier counterpart. All reported multipliers trade silicon area/power consumption for MP. In this paper, silicon area is optimized by applying an operation reduction technique that replaces a multiplier by adders/subtractions.
- A silicon implementation of this MP multiplier integrating an error-tolerant razor-based dynamic DVS approach. The fabricated chip demonstrates run-time adaptation to the actual workload by operating at the minimum supply voltage level and minimum clock frequency while meeting throughput requirements. Prior works combining MP with

DVS have only considered a limited number of offline simulated precision-voltage pairs, with unnecessary large safety margins added to cater for critical paths.

- A novel dedicated operand scheduler that rearranges operations on input operands so as to reduce the number of transitions of the supply voltage and, in turn, minimize the overall power consumption of the multiplier. Unlike reported scheduling works, the function of the proposed scheduler is not task scheduling rather input operands scheduling for the proposed MP multiplier.

Figure 1: Overall Multiplier System Architecture


System-on-chip
Voltage Scaling Unit (VSU)
Clock Frequency Scaling Unit (FSU/VCO)
Supply voltage
Operating Clock
Multi-precision Multiplier
Multiplication results

The proposed MP multiplier system (Figure 1) comprises three different modules that are as follows:

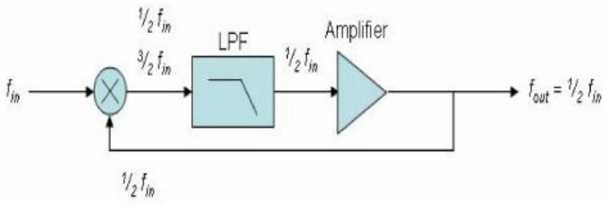
- The MP multiplier;
- Tthe Voltage Scaling Unit (VSU) implemented using a voltage dithering technique to limit silicon area overhead. Its function is to dynamically generate the supply voltage so as to minimize power consumption;
- The clock Frequency Scaling Unit (FSU) implemented using a frequency division technique to limit clock rate. It function is to dynamically generate the different levels of frequencies so as to minimize the clock;

ANALOG DIVIDERS

Regenerative Frequency Divider

A regenerative frequency divider, also known as a Miller frequency divider, mixes the input signal with the feedback signal from the mixer. The feedback signal is $f_{in}/2$. This produces sum and difference frequencies $f_{in}/2$, $3f_{in}/2$ at the output of the mixer. A low pass filter removes the higher frequency and the $f_{in}/2$ frequency is amplified and fed back into mixer. In order to establish a stable 1/2 frequency feedback, the amplifier gain at the half frequency must be greater than unity. The phase shift must also be an integer multiple of 2*pi*.

Figure 2: Regenerative Frequency Divider



Injection-Locked Frequency Divider

A free-running oscillator which has a small amount of a higher-frequency signal fed to it will tend to oscillate in step with the input signal. Such frequency dividers were essential in the development of television.It operates similarly to an injection locked oscillator. In an injection locked frequency divider, the frequency of the input signal is a multiple (or fraction) of the free-running frequency of the oscillator. While these frequency dividers tend to be lower power than broadband static (or flip-flop based) frequency dividers, the drawback is their low locking range. The ILFD locking range is inversely proportional to the quality factor (Q) of the oscillator tank. In integrated circuit designs, this makes an ILFD sensitive to process variations. Care must be taken to ensure the tuning range of the driving circuit (for example, a voltage-controlled oscillator) must fall within the input locking range of the ILFD.

Digital Dividers

For power-of-2 integer division, a simple binary counter can be used, clocked by the input signal. The least-significant output bit alternates at the same rate as the input, the next bit is the 1/2 the rate, the third bit is 1/4 the rate, etc. An arrangement of flipflops are a classic method for integer-n division. Such division is frequency and phase coherent to the source over environmental variations including temperature. The easiest configuration is a series where each flip-flop is a divide-by-2. For a series of three of these, such system would be a divide-by-8. By adding additional logic gates to the chain of flip flops, other division ratios can be obtained. Integrated circuit logic families can provide a single chip solution for some common division ratios.

Another popular circuit to divide a digital signal by an even integer multiple is a Johnson counter. This is a type of shift register network that is clocked by the input signal. The last register's complemented output is fed back to the first register's input. The output signal is derived from one or more of the register outputs. For example, a divide-by-6 divider can be constructed with a 3-register Johnson counter. The three valid values for each register are 000, 100, 110, 111, 011, and 001. This pattern repeats each time the network is clocked by the input signal. The output of each register is a f/6 square wave with 60° of phase shift between registers. Additional registers can be added to provide additional integer divisors.

Mixed Signal Division

An arrangement of D flip-flops are a classic method for integer-n division. Such division is frequency and phase coherent to the source over environmental variations including temperature. The easiest configuration is a series where each D flip-flop is a divide-by-2. For a series of three of these, such system would be a divide-by-8. More complicated configurations have been found that generate odd factors such as a divide-by-5. Standard, classic logic chips that implement this or similar frequency division functions include the 7456, 7457, 74292, and 74294.

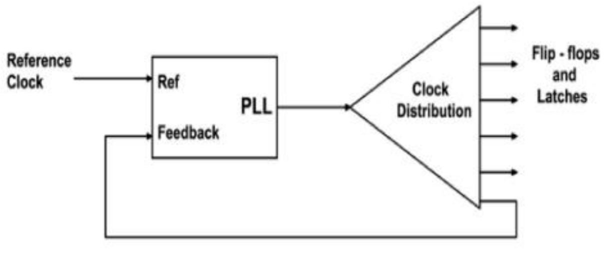
Fractional-n Dividers

A fractional-n frequency synthesizer can be constructed using two integer dividers, a divide-by-n and a divide-by-(n + 1) frequency divider. With a modulus controller, n is toggled between the two values so that the VCO alternates between one locked frequency and the other. The VCO stabilizes at a frequency that is the time average of the two locked frequencies. By varying the percentage of time the frequency divider spends at the two divider values, the frequency of the locked VCO can be selected with very fine granularity.

Clock Distribution

Typically, the reference clock enters the chip and drives a Phase Locked Loop (PLL), which then drives the system's clock distribution. The clock distribution is usually balanced so that the clock arrives at every endpoint simultaneously. One of those endpoints is the PLL's feedback input. The function of the PLL is to compare the distributed clock to the incoming reference clock, and vary the phase and frequency of its output until the

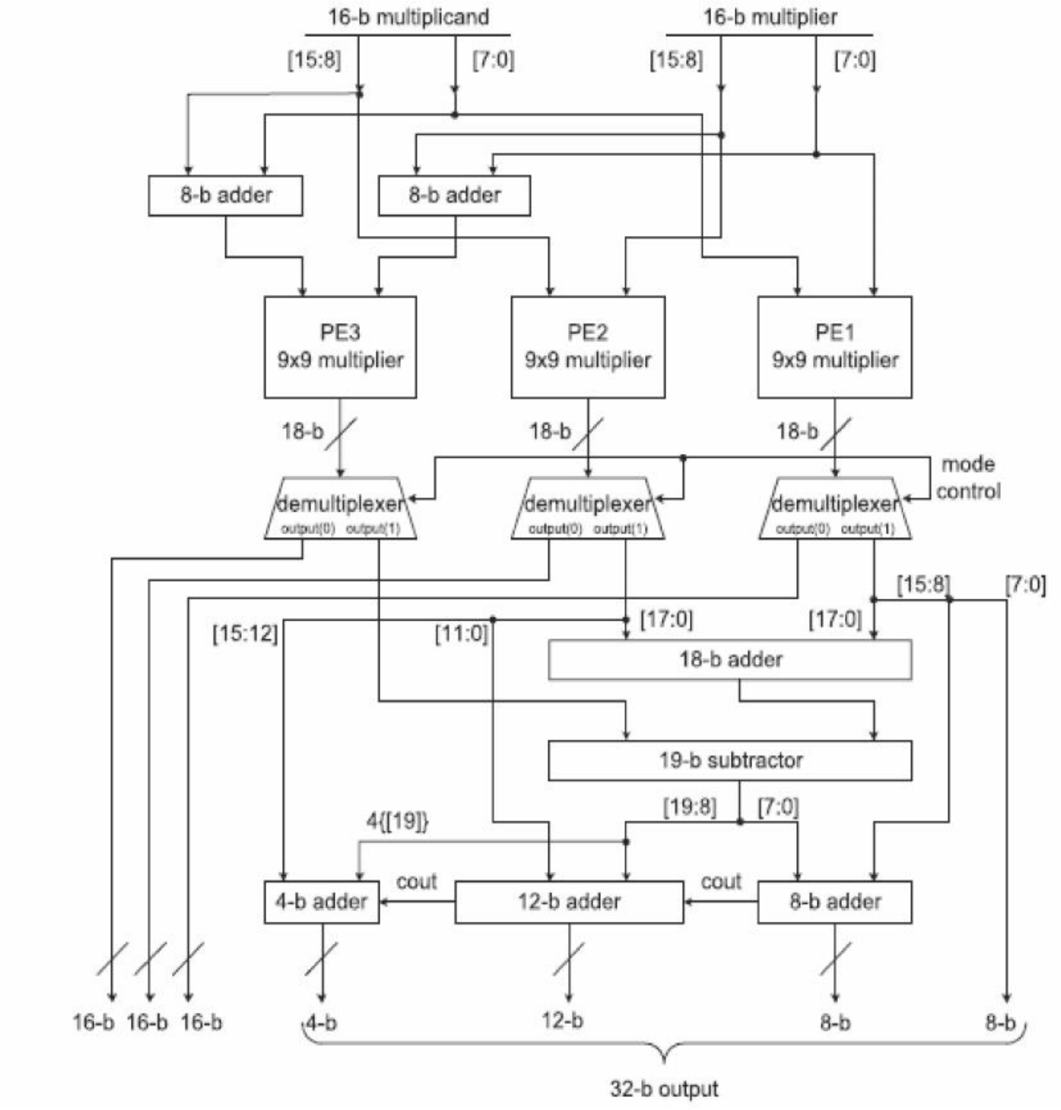
Figure 3: Clock Distribution Network Circuit



reference and feedback clocks are phase and frequency matched.

The above Figure 4 shows how three 8 × 8 bit PEs are used to realize a 16 × 16 bit multiplier. The 32 × 32 bit multiplier is constructed using a similar approach but requires 3 × 3 PEs. A 3-bit control word defines which PEs work concurrently and which PEs are disabled. Whenever the full precision (32 × 32 bit) is not exercised, the supply voltage and the clock frequency may be scaled down according to the actual workload.

Figure 4: PEs Combined to Form 16 × 16 Bit Multiplier



RESULTS

RTL Schematic

The RTL SCHEMATIC gives the information about the user view of the design. The internal blocks contains the basic gate representation of the logic. These basic gate realization is purely depend upon the corresponding FPGA selection and the internal database information.

Waveform

In the waveform which is shown above, clk signal represents clock, rst signal represents reset, a, b signal represents the inputs which we are applying to the design. Similarly prod (product) is the output signal for the design. Here clock signal is generated for the positive edge.

Initially the reset signal should be force to logic 1 and after one clock cycle made it to logic 0 for performing the corresponding functional operation. To obtain the required outputs force the inputs logic with the required values. Here the output product value is obtained by multiplying the inputs a and b.

Figure 5: RTL Schematic for Razor Multiplier

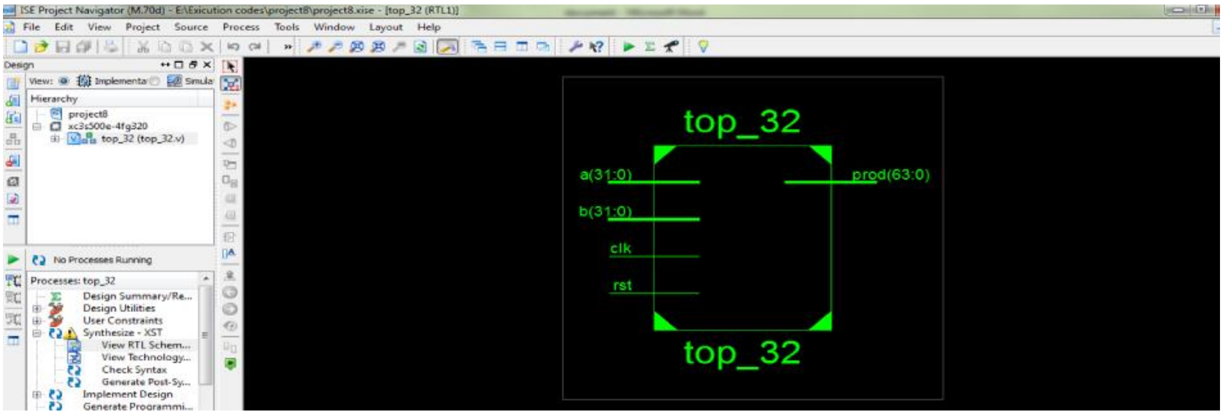


Figure 6: Internal View of RTL Schematic

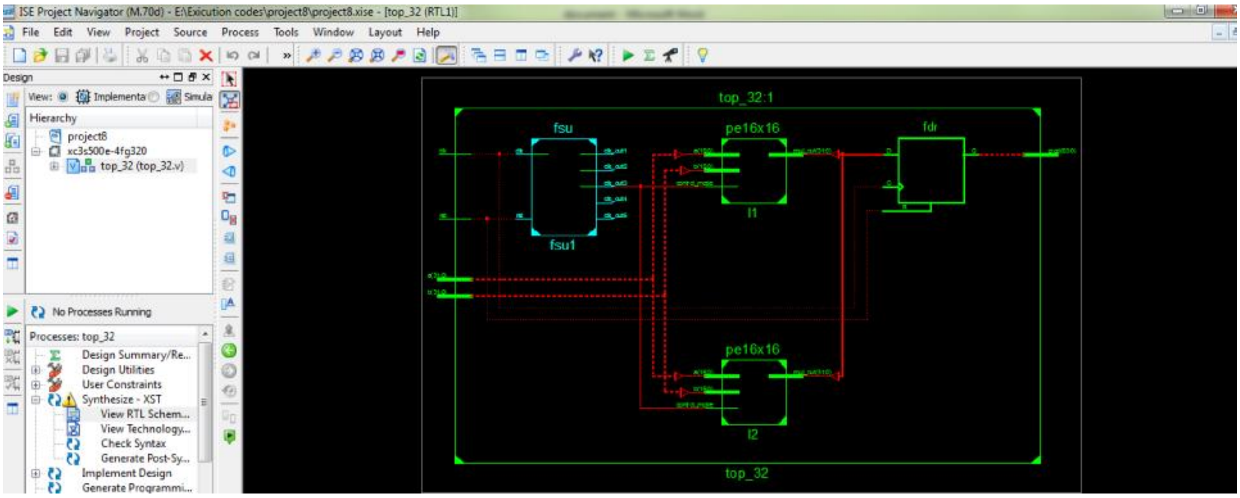
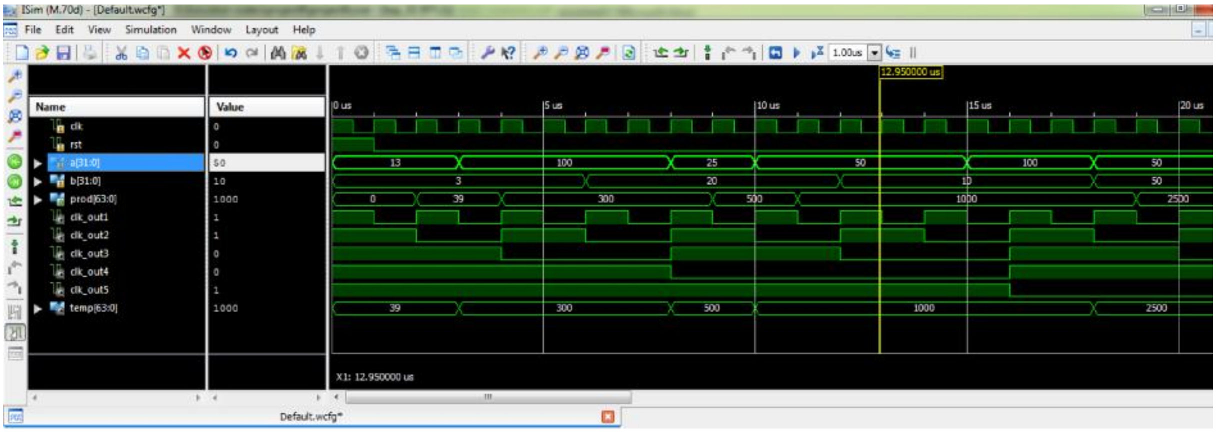
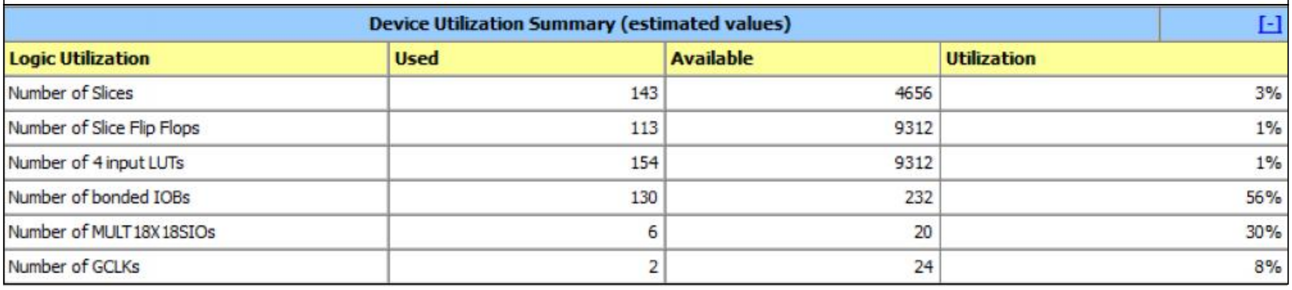


Figure 7: Test Bench Waveform



Name	Value
clk	0
rst	0
a[31:0]	50
b[31:0]	10
prod[63:0]	1000
clk_out1	1
clk_out2	1
clk_out3	0
clk_out4	0
clk_out5	1
temp[63:0]	1000

Table 1: Design and Summary Reports



Device Utilization Summary (estimated values)			
Logic Utilization	**Used**	**Available**	**Utilization**
Number of Slices	143	4656	3%
Number of Slice Flip Flops	113	9312	1%
Number of 4 input LUTs	154	9312	1%
Number of bonded IOBs	130	232	56%
Number of MULT18X18SIOs	6	20	30%
Number of GCLKs	2	24	8%

CONCLUSION

A novel MP multiplier architecture featuring, respectively, its 32 × 32 bit conventional fixed-width multiplier counterpart. When designing this MP multiplier architecture the power consumption is 1.2 mw and delay is 4.28 achieved which shows the power optimization of the proposed design Therefore the run-time adaptation to the actual workload is so much reduced while meeting throughput requirements. The HDL is developed based on the verilog language. The RTL is simulated and synthesized in the XILINX ISE.

REFERENCES

1. Bhardwaj M, Min R and Chandrakasan A (2001), "Quantifying and Enhancing Power Awareness of VLSI Systems", *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, Vol. 9, No. 6, pp. 757-772.
2. Carbognani F, Buergin F, Felber N, Kaeslin H and Fichtner W (2008), "Transmission Gates Combined with Level-Restoring CMOS Gates Reduce Glitches in Low-Power Low-Frequency Multipliers", *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, Vol. 16, No. 7, pp. 830-836.
3. Kuang S-R and Wang J-P (2010), "Design of Power-Efficient Configurable Booth Multiplier", *IEEE Trans. Circuits Syst. I, Reg. Papers*, Vol. 57, No. 3, pp. 568-580.
4. Kuroda T (1999), "Low Power CMOS Digital Design for Multimedia Processors", in *Proc. Int. Conf. VLSI CAD*, October, pp. 359-367.

5. Lee H (2004), "A Power-Aware Scalable Pipelined Booth Multiplier", in Proc. IEEE Int. SOC Conf., September, pp. 123-126.

6. Ling W and Savaria Y (2004), "Variable-Precision Multiplier for Equalizer with Adaptive Modulation", in Proc. 47th Midwest Symp. Circuits Syst., Vol. 1, July, pp. 1553-1556.

7. Min R, Bhardwaj M, Cho S-H, Ickes N, Shih E, Sinha A, Wang A and Chandrakasan A (2002), "Energy-Centric Enabling Technologies for Wireless Sensor Networks", *IEEE Wirel. Commun.*, Vol. 9, No. 4, pp. 28-39.

8. Pfander O A, Hacker R and Pfleiderer H-J (2004), "A Multiplexer-Based Concept for Reconfigurable Multiplier Arrays", in *Proc. Int. Conf. Field Program. Logic Appl.*, Vol. 3203, September, pp. 938-942.

9. Wang A and Chandrakasan A (2003), "Energy-Aware Architectures for a Realvalued FFT Implementation", in *Proc. IEEE Int. Symp. Low Power Electron. Design*, August, pp. 360-365.

10. Yamanaka T and Moshnyaga V G (2004), "Reducing Multiplier Energy by Data-Driven Voltage Variation", in *Proc. IEEE Int. Symp. Circuits Syst.*, May, pp. 285-288.

International Journal of Engineering Research and Science & Technology
Hyderabad, INDIA. Ph: +91-09441351700, 09059645577
E-mail: editorijlerst@gmail.com or editor@ijerst.com
Website: www.ijerst.com

ISSN 2319-5991