International Journal of Electrical, Electronics and Data Communication, ISSN(p): 2320-2084, ISSN(e): 2321-2950
Volume-6, Issue-10, Oct.-2018, http://iraj.in

A NOVEL DESIGN OF BIT-SLICE MATRIX MULTIPLIER FOR RSFQ USING SHADOW LATCH

[1]T. PRASAD BABU, [2]A. SIVA PRASAD, [3]T. INDIRA

[1]Head of the Dept., Dept of E.C.E, Bvsr Engineering College, Chimakurthy, Prakasam Dt, A.P
[2]Head of the Dept.,, Dept. Of ECE, Chintalapudi Engineering College, Ponnur, Guntur Dt, A.P
[3]Assistant Professor, Dept of E.C.E, Narasaraopet Engineering College, Narasaraopeta, Guntur Dt, A.P
E-mail: sbabafariddin@gmail.com

Abstract: In this paper, we demonstrated a high speed energy efficient approximate multiplier. We contemplate an approach which is to round the operands to nearest exponent to two. The proposed system is applied for both unsigned and signed multiplications. A 4-bit bit-slice matrix multiplier is exploited for 32-bit rapid multiple-flux-quantum (RMFQ) artificial intelligence processor is proposed in this paper. The multiplier mainly includes bit-slice multipliers which is 4-bit and 4-bit bit-slice adders. The unsigned integer matrixes multiplication is contrivance by control signals. The result shows that our method simplifies the circuit complexity, truncates the hardware costs and allows extending the matrix multiplier to a smaller or larger number of bits.

Keywords - Accuracy, approximate computing, energy efficient, error analysis, high speed, multiplier.

I. INTRODUCTION

One of the paramount design requirements is energy minimization which is consequential in any electronic systems like smart phones, tablets and various gadgets [1]. The blocks of Digital Signal Processing (DSP) are the key components to perceive the different multimedia applications. The computational core is the arithmetic logic unit (ALU) where multiplications have the prodigious share among all arithmetic operations which are performed in the DSP systems. Hence, speed and power/energy-efficiency characteristics of multipliers play a major role in maximizing the efficacy of processors.

Most of the DSP cores develop the algorithms of image and video processing and final outputs are either videos or images which are concocted for human consumptions. This fact enables us to utilize approximations to revamp the speed/energy efficiency. This originates from the limited perceptual capabilities of human beings for reconnoitre an image or a video. In addition to the video processing and image, there are some other areas where the arithmetic operations exactness is not crucial to the system functionality [2]. The arithmetic units can be performed at miscellaneous design abstraction levels which include circuit, logic and architecture levels, and algorithm and software layers [3].

The conjecture may be performed by utilizing various techniques like allowing some timing violations examples like voltage over scaling or over clocking and function approximation methods like modifying the Boolean function of a circuit or a combination of them [4]. In the methods of function approximation a number of approximating arithmetic building blocks like multipliers and adders at various levels of design have been suggested [5].

Superconducting rapid multiple-flux-quantum (RMFQ) circuits [6] are envisaged to be a future integrated circuits technology by right of its features: high speed and low power consumption [7].This paper focus on a high speed low energy/power approximate multiplier for error resilient applications of DSP. With the progress of RMFQ fabrication process technology, it has become viable to realize an RMFQ including tens of thousands of Josephson junctions (JJs) [8]. However, the calculation time by bit-serial architecture is very long to process 32-/64-bit data. Bit-parallel architecture is used in FLUX [9] processor. In this paper, a 4-bit bit-slice 4×4 matrix multiplier in [10], which is one of the important components for a 32-bit bit-slice RMFQ artificial processor, is proposed. A 4×4-bit and an 8×8-bit parallel RMFQ multiplier have been designed and fabricated [11], [12].

The proposed approximate multiplier is also an area efficacy and it is constructed by modifying the conventional multiplication proposal at the level of algorithm which assuming values of rounded input. This is known as Rounding-based approximate multiplier. The proposed multiplier approach is applicable for both unsigned and signed multiplications in which three optimized architectures are presented. The efficacies of these structures are procured by comparing the delays, power and energy consumptions, Energy Delay Products (EDPs), areas with those of some approximate and accurate (exact) multipliers.

II.EXISTED SYSTEM

The block diagram of the existed system is shown in above fig (1). The inputs are represented in two's complement format. Initially, the signs of the inputs

are resolved and for each negative value, the absolute value is generated.

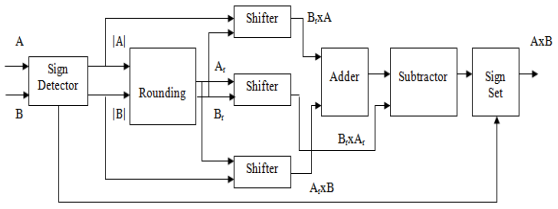

FIG 1. EXISTED SYSTEM

Then the rounding block elicits the nearest value for each value which is absolute in the form of two to the power of n (2^n). The bit width of the output of this block is n that is most significant bit of the absolute value of an n-bit number in the two's complement format is zero. Having determined the rounding values, by utilizing blocks of three barrel shifter the products are deliberated. Hence, depending up on the operand the amount of shifting is determined. Therefore, the shifter blocks input bit width is n, while their outputs are 2n.

III. 4-BIT BIT-SLICE MATRIX MULTIPLIER

A. Algorithm and architecture: A and B are 4×4 matrixes. Each of the 32-bit elements in the two matrixes is demarcated into eight slices of 4 bits each. The eight pairs are input one by one from the least significant one. The multiplier performs unsigned integer matrix multiplication. The matrix produce C=AB is also a 4×4 matrix. Each of the 64-bit elements in C matrix is divided into sixteen 4-bit slices which are output one by one from the least significant one.

Fig. (2) Shows a block diagram of the proposed 4-bit bit slice 4×4 matrix multiplier. It embraces of four 4-bit bit-slice multipliers and three 4-bit bit-slice adders. Fig. (3) Shows a diagram of a 4-bit bit-slice 4×4 matrix multiplication. The matrix A and B are segregated into four rows and four columns respectively. The matrix produce C is divided into four rows.

A 4-bit bit-slice 4×4 matrix multiplication is carried out via 259 steps. The i-th (i=1 to 8) pair of the 32-bit elements in A and B matrixes is input to the matrix multiplier at i-th step. The i-th slice of a00, a01, a02, and a03 in A matrix is fed to input ports A0, A1, A2, and A3 at the i-th step and fed back to the thriving 4-bit bit-slice multiplier with 8-step delay. The i-th slice of b00, b10, b20, and b30 in B matrix is fed to input ports B0, B1, B2, and B3 at the i-th step and fed back to the succeeding 4-bit bit-slice multiplier with 32-step delay.

The succeeding 4-bit bit-slice multiplier enumerates the products of the i-th slices of a00b00, a01b10, a02b20, and a03b30. The two succeeding 4-bit bit-slice adders calculate the sums of the i-th slices of a00b00+a01b10 and a02b20+a03b30, respectively. The last 4-bit bit-slice adder calculates the j-th (j=1 to 16) slice of c00 at steps 4 to 19.

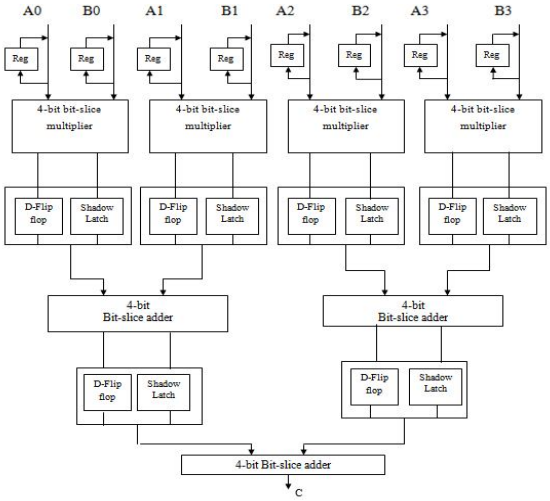

FIG. 2 THE BLOCK DIAGRAM OF A 4-BIT BIT-SLICE 4×4 MATRIX MULTIPLIER

$$\begin{pmatrix} a00 & a01 & a02 & a03 \\ a10 & a11 & a12 & a13 \\ a20 & a21 & a22 & a23 \\ a30 & a31 & a32 & a33 \end{pmatrix} X \begin{pmatrix} b00 & b01 & b02 & b03 \\ b10 & b11 & b12 & b13 \\ b20 & b21 & b22 & b23 \\ b30 & b31 & b32 & b33 \end{pmatrix} = \begin{pmatrix} C00 & C01 & C02 & C03 \\ C10 & C11 & C12 & C13 \\ C20 & C21 & C22 & C23 \\ C30 & C31 & C32 & C33 \end{pmatrix}$$

FIG. 3 THE DIAGRAM OF A 4-BIT BIT-SLICE 4×4 MATRIX MULTIPLICATION

After 16-step delay, the i-th slice of b01, b11, b21, and b31 in B matrix is fed to input ports B0, B1, B2, and B3 at the (i+8)-th step and fed back to the succeeding bit-slice multiplier which is 4-bit with 32-step delay. The succeeding 4-bit bit-slice multiplier calculates the products of the i-th slices of a00b01, a01b11, a02b21, and a03b31. The two succeeding 4-bit bit-slice adders calculate the sums of the i-th slices of a00b01+a01b11 and a02b21+a03b31, respectively. The last 4-bit bit-slice adder calculates the (j+16)-th slice of c01 at steps 36 to 51.

After 16-step delay, the i-th slice of b02, b12, b22, and b32 in B matrix is fed to input ports B0, B1, B2, and B3 at the (i+16)-th step and fed back to the succeeding bit-slice multiplier which is 4-bit with 32-step delay. The succeeding 4-bit bit-slice multiplier calculates the products of the i-th slices of a00b02, a01b12, a02b22, and a03b32. The two succeeding 4-bit bit-slice adders calculate the sums of the i-th slices of a00b02+a01b12 and a02b22+a03b32, respectively. The last 4-bit bit-slice adder calculates the (j+32)-th slice of c02 at steps 67 to 82.

After 16-step delay, the i-th slice of b03, b13, b23, and b33 in B matrix is fed to input ports B0, B1, B2,

and B3 at the (i+24)-th step and fed back to the succeeding 4-bit bit-slice multiplier with 32-step delay. The succeeding 4-bit bit-slice multiplier calculates the products of the i-th slices of a00b03, a01b13, a02b23, and a03b33. The two succeeding 4-bit bit-slice adders calculate the sums of the i-th slices of a00b03+a01b13 and a02b23+a03b33, respectively. The last 4-bit bit-slice adder calculates the (j+48)-th slice of c03 at steps 98 to 113.

Comparably, the i-th slice of a10, a11, a12, and a13 in A matrix is fed to input ports A0, A1, A2, and A3 at the (i+32)-th step and fed back to the succeeding 4-bit bit-slice multiplier with 8-step delay. The sixteen slices of c10, c11, c12, and c13 are calculated from steps 114 to 241. The i-th slice of a20, a21, a22, and a23 in A matrix is fed to input ports A0, A1, A2, and A3 at the (i+64)-th step and fed back to the succeeding 4-bit bit-slice multiplier with 8-step delay. The sixteen slices of c20, c21, c22, and c23 are calculated from steps 242 to 369. The i-th slice of a30, a31, a32, and a33 in A matrix is fed to input ports A0, A1, A2, and A3 at the (i+96)-th step. The sixteen slices of c30, c31, c32, and c33 are calculated from steps 370 to 497.

B. Rapid Multiple-Flux-Quantum (RMFQ) Logic Design: We exert concurrent flow clocking to design fully pipelined synchronous RMFQ logic circuits of the proposed matrix multiplier. Scilicet, each pipeline stage consists of a row of RMFQ clocked logic gates.

Each register (Reg) for holding a slice of A0-A3 is implemented using eight D flip-flops (DFFs). Each Reg for holding a slice of B0-B3 is contraption using thirty-two DFFs. We can use the bit-slice multiplier and the bit-slice Sklansky adder for the bit-slice multiplier which is 4-bit and the 4-bit bit-slice adder, respectively. The matrix multiplier constitute of 45 stages in total. There is only one 1-stage feed-back loop which is in a 4-bit bit-slice adder.

Eight pairs of each element slices in the initial row of matrix A and in the initial column of matrix B are fed at the first to eighth clock cycles. After sixteen clock cycles, Eight pairs of each element slices in the first row of matrix A are held and eight pairs of each element slices in the second column of matrix B are fed at the twenty-fifth to thirty-second clock cycles, and so on. Sixteen pairs of each row element slices in matrix produce C are output at the 46th to 566th clock cycles.

Flip flops and latches are the basic elements and these are utilized for storing the information. Single latch and flip flop can storing the single bit of data. The main difference among the latches and flip flops is that, a latch continuously checks the input and the output changes when there is a change in the input. But, flip flop is a combination of clock and latch which continuously checks the input and changes the output time which is adjusted by the clock.

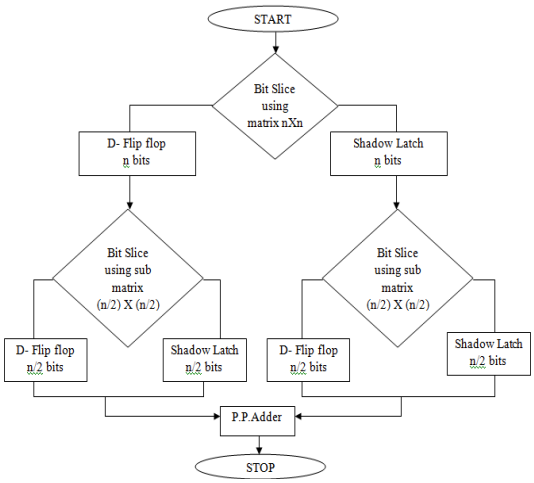

FIG 4. ALGORITHM

In D- flipflop, The i/p D which is given directly into the input S and the complement of the input D which is given to the input R. During the existence of a clock pulse, the input D is sampled. If it is one (1), then the flip-flop is in the state of set. If it is zero (0), then the flip-flop is in the state of clear.

A device of latch is provided with a shadow latch and a driver. The driver has an input for accepting input signal of a binary driver, an input for accepting a clock signal, and an input for accepting a shadow-Q signal. The driver has an output for supplying a binary Q signal which is equal to the inverse of the driver input signal, in response to the driver input signal, the shadow-Q signal, and the clock signal. The shadow latch has an input for accepting the driver input signal, and an input to accept the clock signal. The shadow latch has an output for supplying the shadow-Q signal equal to the inverted Q signal, in response to the driver input signal and clock signal.

IV.RESULTS

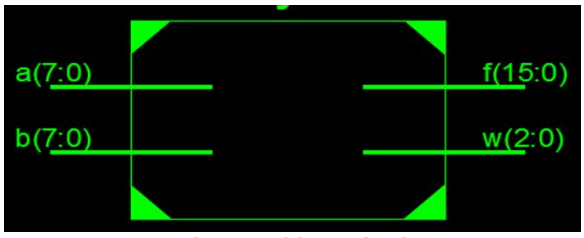

FIG 5. RTL SCHEMATIC

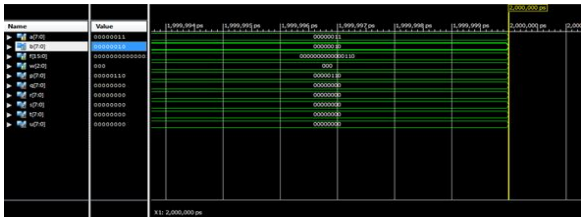

FIG 6. OUTPUT WAVEFORM

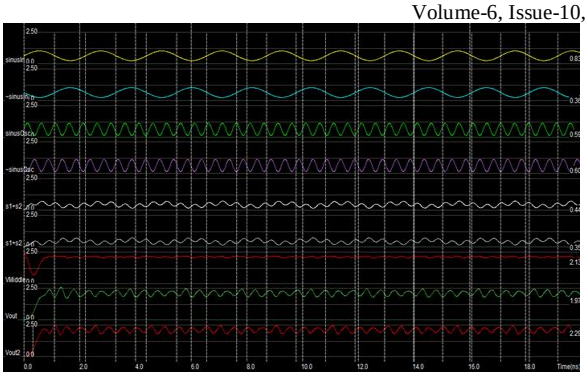

FIG 7. SIMULATED ANALOG OUTPUT USING BACKEND TOOLS

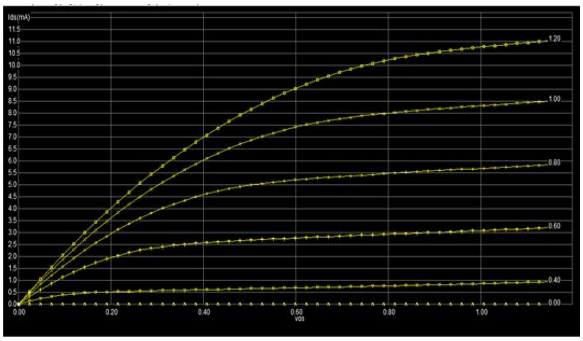

FIG 8. DRAIN CURRENT (I_D) VS GATE VOLTAGE (V_G)

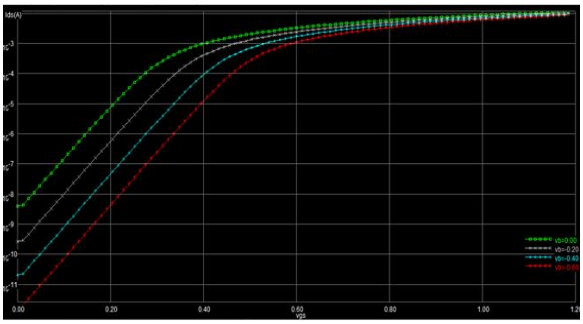

FIG 9. LOG (DRAIN CURRENT (I_D)) VS GATE VOLTAGE (V_G)

binary8mul Project Status			
Project File:	NSMUL.xise	Parser Errors:	No Errors
Module Name:	binary8mul	Implementation State:	Synthesized
Target Device:	xc3s100e-5vq100	• Errors:	No Errors
Product Version:	ISE 14.7	• Warnings:	18 Warnings (0 new)
Design Goal:	Balanced	• Routing Results:	
Design Strategy:	Xilinx Default (unlocked)	• Timing Constraints:	
Environment:	System Settings	• Final Timing Score:	

Device Utilization Summary (estimated values)			
Logic Utilization	Used	Available	Utilization
Number of Slices	93	960	9%
Number of 4 input LUTs	164	1920	8%
Number of bonded IOBs	35	66	53%

Detailed Reports					
Report Name	Status	Generated	Errors	Warnings	Infos
Synthesis Report	Current	Sun 29. Apr 11:39:28 2018	0	18 Warnings (0 new)	0
Translation Report					
Map Report					

FIG 10. REPORT

CONCLUSION

We have proposed a 4-bit bit-slice 4×4 matrix multiplier for 32-bit RMFQ artificial intelligence processors. The proposed multiplier has the proficiency to operate multiplication for both signed and unsigned values. The results divulged that, in most all cases, the multiplier architectures surmounted the corresponding approximate (exact) multipliers. The proposed 4-bit bit-slice multiplier proffers high performance and popularizes the complexity of circuit.

REFERENCES

[1] M. Alioto, "Ultra-low power VLSI circuit design demystified and explained: A tutorial," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 59, no. 1, pp. 3–29, Jan. 2012.

[2] H. R. Mahdiani, A. Ahmadi, S. M. Fakhraie, and C. Lucas, "Bio-inspired imprecise computational blocks for efficient VLSI implementation of soft-computing applications," IEEE Trans. Circuits Syst. I, Reg. Papers, vol. 57, no. 4, pp. 850–862, Apr. 2010.

[3] V. Gupta, D. Mohapatra, A. Raghunathan, and K. Roy, "Low-power digital signal processing using approximate adders," IEEE Trans. Comput.-Aided Design Integr. Circuits Syst., vol. 32, no. 1, pp. 124–137, Jan. 2013.

[4] F. Farshchi, M. S. Abrishami, and S. M. Fakhraie, "New approximate multiplier for low power digital signal processing," in Proc. 17th Int. Symp. Comput. Archit. Digit. Syst. (CADS), Oct. 2013, pp. 25–30.

[5] D. R. Kelly, B. J. Phillips, and S. Al-Sarawi, "Approximate signed binary integer multipliers for arithmetic data value speculation," in Proc. Conf. Design Archit. Signal Image Process., 2009, pp. 97–104.

[6] K. K. Likharev and V. K. Semenov, "RMFQ logic/memory family: a new Josephson-junction technology for sub-terahertz-clock-frequency digital systems," IEEE Trans. Appl. Supercond., vol. 1, no. 1, pp. 3-28, Mar. 1991.

[7] N. Takagi and M. Tanaka, "Comparisons of synchronous-clocking SFQ adders," IEICE Trans. Electron., vol. E91-C, no. 4, pp. 429-434, Apr. 2010

[8] Y. Yamanashi, T. Kainuma, N. Yoshikawa, I. Kataeva, H. Akaike, A. Fujimaki, M. Tanaka, N. Takagi, S. Nagasawa and M. Hidaka, "100 GHz demonstrations based on the single-flux-quantum cell library for the 10 kA/cm2 Nb multi-layer process," IEICE Trans. Electron., Vol. E93-C, No. 4, pp. 440-444, Apr. 2010.

[9] M. Dorojevets, P. Bunyk, and D. Zinoviev, "FLUX chip: design of a 20-GHz 16-bit ultrapipelined RMFQ processor prototype based on 1.75-μm LTS technology," IEEE Trans. Appl. Supercond., vol. 11, no. 1, pp. 326–332, Mar. 2001.

[10] G. Tang, K. Takagi, and N. Takagi, "32 × 32-Bit 4-Bit Bit-Slice Integer Multiplier for RMFQ Microprocessors," IEEE Trans. Appl. Supercond., vol. 27, no. 3, Apr. 2017, Art. no. 1301005.

[11] I. Kataeva, H. Engseth, and A. Kidiyarova-Shevchenko, "New design of an RMFQ parallel multiply accumulate unit," Supercond. Sci. Technol., vol. 19, pp. 381–387, May 2006.

[12] M. Dorojevets, A. K. Kasperek, N. Yoshikawa, and A. Fujimaki, "20- GHz 8 × 8-bit Parallel Carry-Save Pipelined RMFQ Multiplier," IEEE Trans. Appl. Supercond., vol. 23, no. 3, pp. 1300104, Jun. 2013.

[13] A. K. Kasperek, "32-bit superconductor integer and floating-point multiplier," Ph.D. dissertation, Dept. Comput. Eng., Stony Brook Univ., Stony Brook, NY, 2012.

[14] Nangate 45nm Open Cell Library, accessed on 2010. [Online]. Available: http://www.nangate.com/

[15] G. Tang, K. Takagi, and N. Takagi, "RMFQ 4-bit bit-slice integer multiplier," IEICE Trans. Electron., vol. E99-C, no. 6, pp. 697–702, Jun. 2016.

★★★