Efficient Serial Multiplier Design using Ripple Counters,Kogge-Stone Adder and Full Adder

Vishnupriya.A
P.G.Scholar-Department of Electronics and Communication Engineering
Avinashilingam University for Women
Coimbatore, India.

Sudarmani.R
Assistant Professor - Department of Electronics and Communication Engineering
Avinashilingam University for Women
Coimbatore, India.

ABSTRACT
High speed and low power multiplier circuits are highly demanded in VLSI design. In this paper, a new approach for high speed and low power multiplier design with less number of gate counts is proposed. In the Ripple Counter–based multiplier design, the number of computational clock cycles is reduced to n for n * n multiplication where n is the word length or the number of bits, which was 2n in the conventional CSAS multiplier. The Ripple Carry Adder (RCA) in the Counter-based design is replaced with Kogge-Stone adder (KSA) for reducing the average connection delay. For reducing the total equivalent gate count and power, the full adder with alternate logic is implemented along with KSA in the multiplier architecture. In our paper, 27.21% and 31.53% of the total power has been reduced for unsigned and signed number multiplication respectively.

Keywords
Binary multiplication, Kogge-stone adder, Partial product, Ripple counters, Serial multiplier.

1. INTRODUCTION
Multipliers are generally the slowest element in the system and the performance of the system or a chip is generally determined by the performance the multiplier. The speed of multiply operation is of great importance in digital signal processing as well as in the general purpose processors. Parallel multipliers are popular for their high speed of operation. The cost of hardware is the drawback in long word length multiplication using parallel multipliers. It also consumes more power in such a case. In public key cryptography like RSA encryption and decryption, integer multiplications of 1024 bits are specific [1].

A full precision bit-serial multiplier was introduced by Strader et al [2] for unsigned numbers. A full-precision scheme for 2's complement numbers has been presented in [3]. This multiplier for n-bit operands requires 2n clocks and 2n number of five-input adder modules. A serial multiplier and a squarer with no latency cycles are presented in [4] and algorithms for serial squarer's and serial/serial multipliers were derived in [5]. The signed multiplication technique was proposed in the year 1950 which is applicable for both the signs [6]. This is a technique where binary numbers of both the sign can be multiplied by a uniform process. Multiplication involves 2 basic operations: the generation of the partial product and their accumulation.Baugh-Wooley Multiplier is used for both unsigned and signed number multiplication [7], [8]. Serial accumulation of unsigned binary numbers by high speed 1's counters is proposed in [12]. Multiplication process consists of three stages of operation, the generation of partial products (PPs), the reduction of PPs and the final carry-propagation addition [13]. Unlike the conventional CSAS architecture, the critical path is found only along the AND gates [16]. The rows of the partial product can be generated in n clock cycles instead of 2n clock cycles and hence delay is reduced. Column compression technique is used for reducing the height of the partial product tree height. Thus, the numbers of computational cycles are reduced. Moreover, the counters change states only when input is '1', which leads to low switching power.

Adder is one of the most important components of a CPU (Central Processing Unit). Fast adders are necessary in ALUs, for computing memory addresses.Full-adders are important components in other applications such as digital signal processors (DSP) architectures and microprocessors.

Kogge-stone adder is called as parallel prefix adder. It is the common design for high-performance adders in industry. It is considered as the fastest adder as the computation time taken by this adder is O (log n) time [20].The logic-level implementation of the basic cells used in parallel-prefix adders is described in [22]. The basic structure of 8-bit radix-2 Kogge-Stone adder (KSA) described here operates on the principle of block propagate (p) and block generate (g).Adder is one of the most important components of a CPU (Central Processing Unit).

A number of fast adders like carry-skip adder, carry-select adder and carry look-ahead adders have been proposed in the past [25]-[28]. Optimization of adders can be done at logic-level or circuit level. Logic-level optimization involves rearranging the Boolean equations so that a faster or smaller even less power consumption circuit is obtained. Several full adder logic cells are compared and discussed in [34].The template, and replace the content with your own material.

The rest of this paper is organized as follows Section II describes the background information about CSAS serial multiplier. Section III reviews the description of the Ripple Counter – based serial multiplier. Section IV explains the implementation of Kogge-stone adder in the Counter – based multiplier. Section V explains the logic structure of Full Adder and its implementation in the Ripple Counter - based multiplier. Section VI illustrates the implementation and simulation results arrived. Section VII gives the conclusion of the work done so far.

2. THE CSAS SERIAL MULTIPLIER
Data accumulation is carried out using carry save adder (CSA). These multipliers are based on carry-save add-and-shift (CSAS) architecture.Both signed and unsigned multiplication involves AND gates, full adders and D flip flops. AND gates are used to multiply the serially loaded data.

Full adders are employed for addition of PP. The critical path is along the D flip flop and the AND gates. DFFs below the FA are used for the storage of carry that is generated by the FA. The adjacent DFFs are used for shifting of the sum which is the result of the FA.

2.1 Unsigned multiplication

At the first clock cycle, y_0 is loaded to all the AND gates and $x_0,..,x_{n-1}$ are loaded to their respective AND gates. Addition is performed simultaneously by FA. The sum is stored in the adjacent DFF and in the next clock cycle, carry is saved in the DFF below. The same is repeated for all operand bits in the following clock cycles. After 2n clock cycles, the product P is obtained. A '0' at the MSB side is used for reset, before loading a new set of operands.

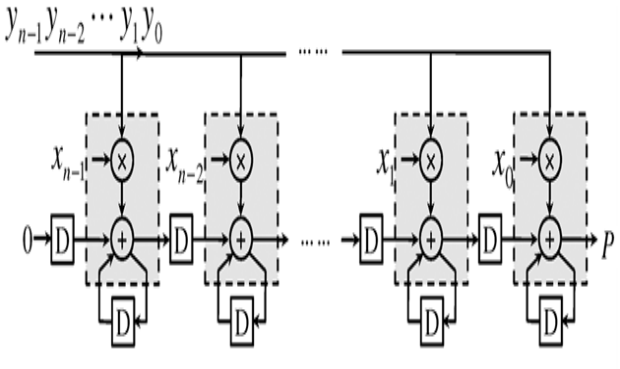


Fig 1 Unsigned CSAS serial-parallel multiplier [15]

2.2 Signed multiplication

The operation is similar to unsigned multiplication except that it involves a bit Q and another EX-OR gate.Q stores the MSB of the operands, which is the signed bit that is EX-ORed with the multiplicand. The result of EX-OR is then ANDed with y_0. Then the product P is obtained as in unsigned multiplication.In this method, data accumulation is done using CSA. The critical path is along the AND gates, FA's and DFF's. The disadvantage of this CSA based accumulation is it takes 2n computational cycles to compute the product.

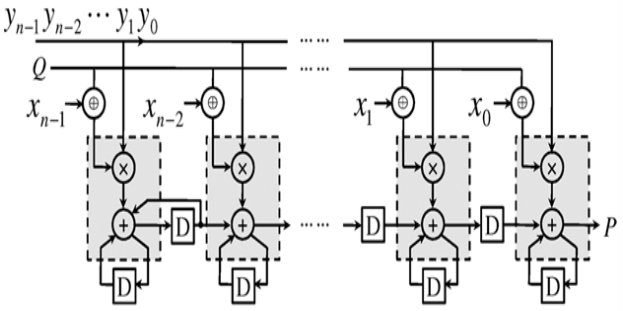


Fig 2 Signed CSAS serial-parallel multiplier [15]

3. THE RIPPLE COUNTER BASED SERIAL MULTIPLIER USING RIPPLE CARRY ADDER

Asynchronous counter-based data accumulation is introduced in our method to overcome the drawbacks of CSA based accumulation.

3.1 Ripple counter-based accumulation

Accumulator is an adder that adds the current input with the value that is already stored in its internal register. Mathematically, accumulation of n integers x_i for i=0, 1, n-1, can be given by

$$S = \sum_{i=0}^{n-1} x(i)$$

where S is the sum.Asynchronous counters are also called as ripple counters. Here, the DFFs are used for storage. The clock input to the first DFF in a 3 bit 1's counter is the ANDed output of the clock signal and input. The clock for the successive DFF's is driven by the output of its preceding DFF.When a new operand is loaded, the counter gets incremented if a '1' is present in the bit position, corresponding to that column. At the end of n clock cycles, the counters outputs are accumulated in the latching register. These 1's counters are called as accumulators.

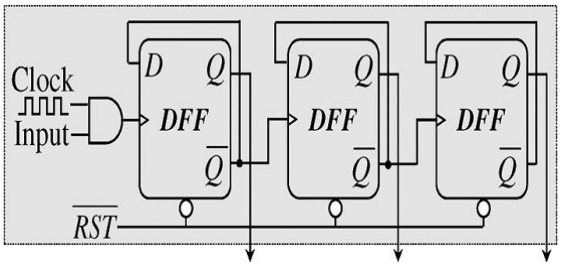


Fig 3 Architecture of 3-bit 1's counter

3.2 Ripple Counter-based multiplication scheme

Two serial inputs are considered, one starting from the LSB and the other from the MSB, for generating the individual row of partial products. The product P of two n-bit unsigned binary numbers X and Y is obtained as follows:

$$P = X.Y = \sum_{i=0}^{n-1}\sum_{j=0}^{n-1} x_i y_i 2^{i+j}$$

Where x_i and $y_i$are the *i*th and *j*th bits of X and Y respectively, with 0 being the LSB.Toform the PP matrix, the multiplier and the multiplicand is ANDed. To reduce the height of the PP matrix, column to row transformation is performed using 1's counters.The length of the counters varies according to the column height of the PP matrix which gets incremented from 1 to n and then reduces to 1. One row of PPs is generated in each cycle.The partial products so formed are counted column wise for the number of ones.

x and y are the multiplier and multiplicand bits respectively that are loaded serially. Two clocks, clk1 and clk2 are employed to synchronize the data flow.Clk2 is derived from clk1 to drive the latching register.

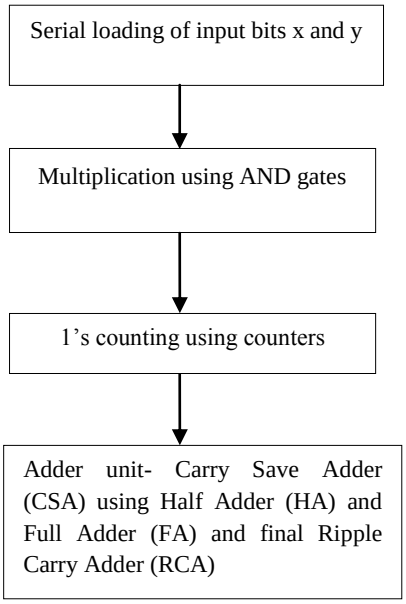


Fig 4 Counter based multiplication scheme

Clk1 is given to AND gates in order to make sure that their outputs have no glitches. The latching register is used in between the counter and the adder stages to prevent spurious transitions. There are 3 stages in addition: half adder (HA), full adder (FA) and ripple carry adder (RCA).

The first 2 adders form the carry save adder (CSA) unit. The product of multiplication is produced in parallel by the final RCA.The architecture of Counter-based architecture for serial unsigned multiplication is shown in Fig *7*.x and y bits are loaded serially and are shifted left and right respectively at the successive clock cycles using DFFs.

The ANDed output is then given as input to its corresponding 1's counter. If a '1' is at the output of the AND gate, the counter changes its state at the rising edge of the clock. The counter outputs are accumulated in the latching register at the end of n clock cycles. Now, the PP matrix should further be reduced which is performed by the adder blocks present below the latching register to obtain the product.

Baugh-Wooley algorithm is followed for signed multiplication. In this algorithm, the MSB of each row and the last row in PP matrix are complemented, except the MSB of the last row. The architecture of Counter-based architecture for serial signed multiplication is shown in Fig *6*.A '1' is added to the immediateleft column of middle column. An inverter for complementing some PP is presentin the multiplier block. Now the PPs are added as usual and the other operations are same as unsigned multiplier.

4. THE RIPPLE COUNTER BASED SERIAL MULTIPLIER USING KOGGE-STONE ADDER

In both the above multiplier architectures the final product is computed by the Ripple carry adder (RCA). A number of fast adders like carry-skip adder, carry-select adder and carry look-ahead adders have been proposed in the past [21], [23]-[28]. Kogge-Stone adders (KSA) are popular choice in high speed ALU design due to its faster operation, regular structure and balanced loading in internal nodes compared to other sparse tree adders.

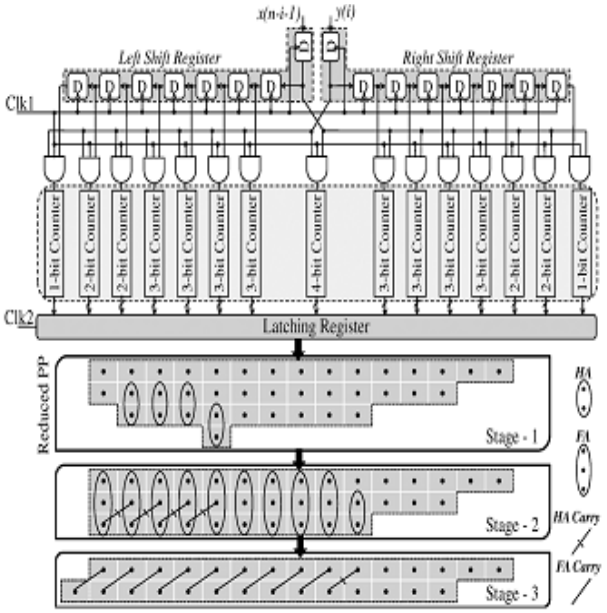


Fig5 Architecture for counter based serial unsigned multiplier [16]

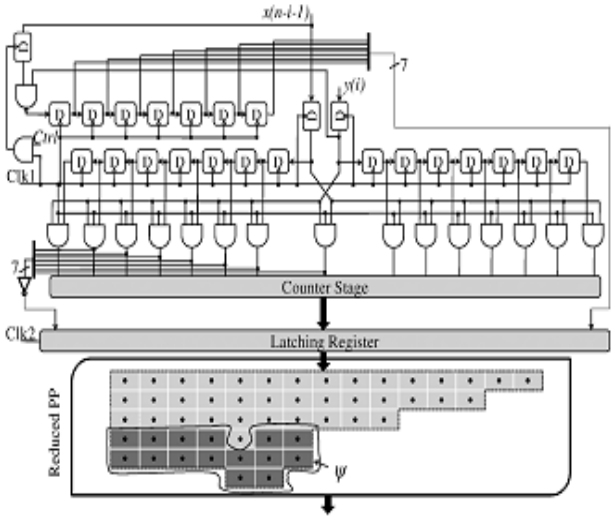


Fig 6 Architecture for counter based serial signed multiplier[16]

It is one of the parallel prefix adder structures. In Fig *4*, RCA is replaced by KSA for improved performance. In this section, first we briefly discuss the design, operation and general properties of KSA. In KSA, the carries are computed fast by computing them in parallel.

KSA belongs to the family of fastest parallel prefix adders with complexity of $\log_2 N$ (where N is the width of the adder) meaning thereby that the addition can be done in $\log_2 N$ stages. The basic structure of 8-bit radix-2 Kogge-Stone adder [22] is illustrated in Fig 7. It operates on the principle of block propagate (p) and block generate (g) [23]. The block propagate determines whether the input carry can propagate through the block of bits or not. The block generate determines if the block of bits can generate a carry or not.

If a and b are input operands of the adder, the propagate/generate and carry in (Ci)/carry out (Ci+1) are related as follows

$$P_i = a_i \; XOR \; b_i; \quad g_i = a_i \bullet b_i; \quad C_{i+1} = g_i + P_i C_i$$

In absence of carry input to the adder core (i.e., C-1= 0), the generate signal become the carry inputs for the intermediate carry computations. In 8-bit KSA, the carry is computed in k=3stages. The logic-level implementation of thebasic cells used in parallel-prefix adders are shown in Fig 8.

The average connection delay in the multiplier architecture due to KSA is reduced and is expressed in terms of Nano seconds.

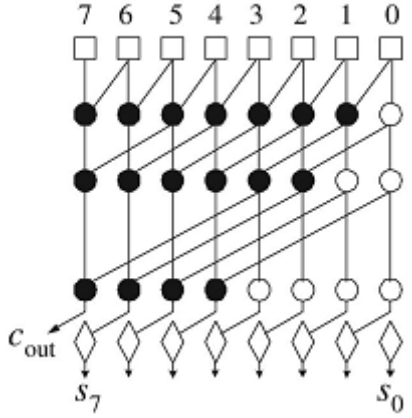
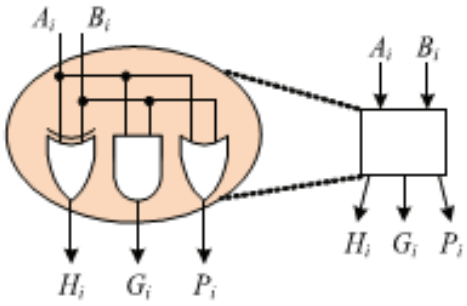
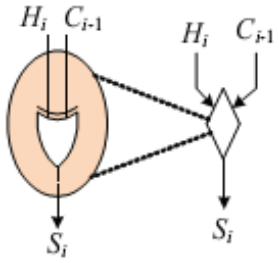


Fig 7 The basic structure of 8-bit radix-2 Kogge-Stone adder [21]





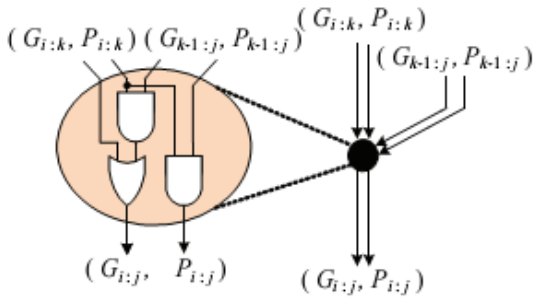


Fig 8 The logic-level implementation of the basic cells used in parallel-prefix adders [21]

5. FULL ADDER IMPLEMENTATION IN THE RIPPLE COUNTER-BASED SERIAL MULTIPLIER USING KOGGE-STONE ADDER

Optimization of adders can be done at logic-level or circuit level. Logic-level optimization involves rearranging the Boolean equations so that a faster or smaller even less power consumption circuit is obtained. Several full adder logic cells are compared and discussed.

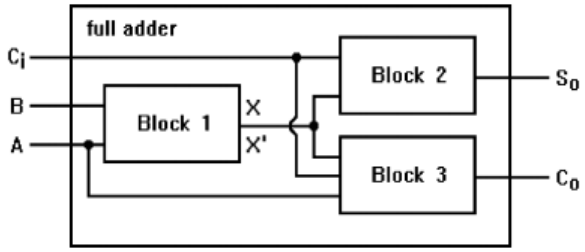


Fig 9 Full-adder cell formed by three main logical blocks [34]

In Fig 9, the adder module isformed by three main logical blocks: aXOR-XNORgate to obtain A xor B, Axnor B (Block 1), and XOR blocks or multiplexers to obtain theSUM (So) and CARRY (Co) outputs (Blocks 2 and 3).

Analysing Fig 10, it can be seen that when C=0, the output S_O= A XOR B and when C=1, the output S_O= A XNOR B. Thus, a multiplexer can be used to obtain the respective value taking the C input as the selection signal.Similarly, when C=0, the output C_O= A AND B and when C=0, the output C_O= A OR B.

C	B	A	S_O	C_O
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Fig 10 Truth Table of Full Adder

Hence, an alternative logic scheme [35] to design a full-addercell can be formed by a logic block to obtain the XOR, XNOR, AND, OR logic and two multiplexers being driven by the input to generate the SoandCooutputs, as shown in Fig 11.

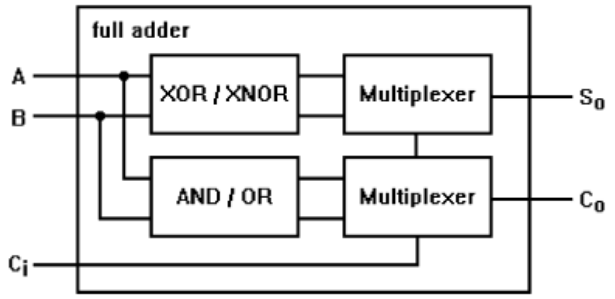


Fig 11Alternative logic scheme for designing full-adder cell [35]

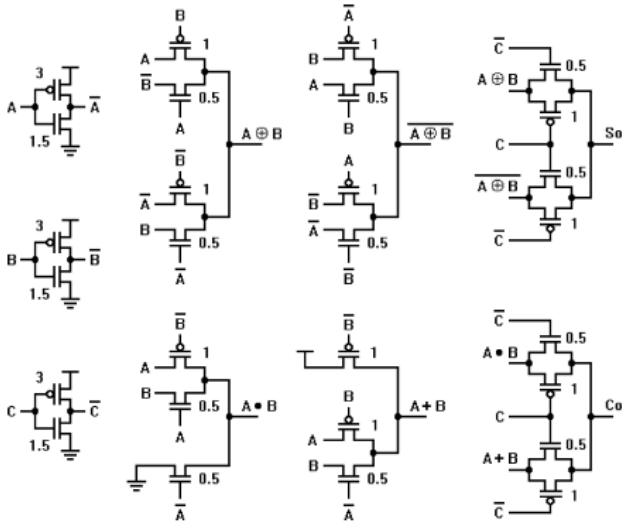


Fig 12Full-adder designed with the alternate logic structure [35]

The full adder unit in Fig 4is replaced by the full adder designed with alternate logic structure shown in Fig 12. Both the full adder structures are implemented in the Ripple Counter – based multiplier using Kogge-stone adder and the results are compared.

6. RESULTS AND DISCUSSION

In the CSAS multiplier, the inputs has to pass through AND gate, Full adder and DFF. The critical path delay exists along these units. This technique requires 2n clock cycles for computation. In the counter-based multiplier, the critical path delay exists in only one AND gate of the counter. Hence it requires n clock cycles for computation [16]. 8*8 multiplication has been performed for both the techniques. The simulation results have been compared using Modelsim 10.0b. The average connection delay in the multiplier architecture due to KSA is reduced and is expressed in terms of Nano seconds. The delay report is obtained by simulating the code in Xilinx ISE simulator.The total equivalent gate count and the total power consumed in milli Watt using the full adder structure in Kogge Stone adder based multiplier unit are analyzed using Xilinx ISE simulator.

Table 1 Comparison results of number of clock cycles

Parameter	CSAS multiplier		Ripple Counter-based multiplier	
Number of clock cycles	Unsigned	Signed	Unsigned	Signed
	16	16	8	8

Table 2Comparison results of*average connection delay*

Parameter	Ripple Counter based multiplier with RCA		Ripple Counter based multiplier with KSA	
Average connection delay(nano seconds)	Unsigned	Signed	Unsigned	Signed
	1.597	1.364	1.462	1.343

Table 3Comparison results oftotal equivalent gate count and power

Parameter	Ripple Counter based multiplier with KSA		Full adder implementation in the Ripple Counter-based multiplier	
	Unsigned	Signed	Unsigned	Signed
Total equivalent gate count for the design	7098	11203	5886	9067
Total power consumption (mW)	136	203	99	139

Initially the design is implemented in ModelSim 10.0b using VHDL code and it is observed from the above comparison Table I that the number of computational clock cycles has been reduced from 16 to 8 for an 8*8 multiplication. Then the code is simulated using Xilinx ISE Simulator and the delay report is extracted. Now from Table II, it is clear that the average connection delay is reduced to 1.462ns and 1.343ns from 1.597ns and 1.374ns for unsigned and signed operations of ripple counter-based multiplier with RCA and KSA respectively. Table III describes the area and power reduction using Full Adder. From the results compared in Table III, it can be concluded that the total equivalent gate count and the total power consumed are reduced.

7. CONCLUSION

In our paper, 8 * 8 multiplication is performed. The product is obtained in 8 computational clock cycles. Using Kogge-stone adder, it can be concluded that the average connection delay of the design is reduced. The total equivalent gate count for the design is also reduced using the alternate logic of full adder. 27.21% and 31.53% of the total power has been reduced for unsigned and signed number multiplication respectively. Thus high speed and low power serial multiplier with less number of gate counts has been designed. n * n multiplication can also be performed.

8. REFERENCES

[1] A. K. Lenstra and E. Verheul, "Selecting cryptographic key sizes," *J.Cryptology*, vol. 14, no. 4, pp. 255–293, 2001.

[2] N. R. Strader and V. T. Rhyne, "A canonical bit-sequential multiplier," *IEEE Trans. Comput.*, vol. C-31, no. 8, pp. 791–795, August 1982.

[3] R. Gnanasekaran, "On a bit-serial input and bit-serial output multiplier" *IEEE Trans. Comput.*, vol. C-32, no. 9, pp. 878–880, 1983.

[4] P. Ienne and M. A Viredaz, "Bit-serial multipliers and squarers," *IEEE Trans. Comput.*, vol. 43, no. 12, pp. 1445–1450, December 1994.

[5] Mark Vesterbacka, Kent Palmkvist, and Lars Wanhammar, "Serial squarers and serial/serial multipliers".

[6] A. D. Booth, "A signed binary multiplication technique," *Quarterly J.Mechan. Appl. Math.*, vol. 4, no. 2, pp. 236–240, August 1951.

[7] C.P. Lerogue, P.Gerard, and J.S. Colardelle, " A fast 16-bit NMOS parallel multiplier", *IEEE Journal of Solid-state circuits*, Vol. 19, no.3, pp. 338-342, Mar 1984.

[8] Jin-HaoTu and Lan-Da Van, "Power-Efficient Pipelined Reconfigurable Fixed-Width Baugh-Wooley Multipliers" *IEEE Transactions on computers*, vol. 58, No. 10, October 2009.

[9] A. Dandapat, S. Ghosal, P. Sarkar, D. Mukhopadhyay (2009), "A 1.2- ns16×16-Bit Binary Multiplier Using High Speed Compressors", *International Journal of Electrical, Computer, and Systems Engineering*, pp. 234-239, 2009.

[10] William J. Stenzel, William J. Kumitz , Gilles H.Garcia, " Compact High- Speed Parallel Multiplication Scheme," *IEEE Trans. Comput*., C-26:948-957, 1977.

[11] S. Haynal and B. Parhami, "Arithmetic structures for inner-product and other computations based on a latency-free bit-serial multiplier design," presented at the 13th *Asilomar Conf. Signals, Syst. Comput., Geneva*, Switzerland, 1996.

[12] ManasRanjanMeher, Ching-ChuenJong,Chip-Hong Chang, "High-Speed and Low-Power Serial Accumulator for Serial/Parallel Multiplier" *IEEE Xplore*, 2010.

[13] B. Parhami, Computer Arithmetic: Algorithms and Hardware Designs. New York: Oxford Univ. Press, 2009.

[14] R. Menon and D. Radhakrishnan, "High performance 5:2 compressor architectures," *IEE Proc-Circuits Devices Syst.*, vol. 153, no. 5, pp.447–452, Oct. 2006.

[15] R. Gnanasekaran, "A fast serial-parallel binary multiplier,"*IEEE Trans. Comput.*, vol. C-34, no. 8, pp. 741–744, August 1985.

[16] ManasRanjanMeher,ChingChuen Jong, Chip-Hong Chang, "A High Bit Rate Serial-Serial Multiplier with On-the-Fly Accumulation by Asynchronous Counters,"*IEEE transactions on Very Large Scale Integration (VLSI) systems*, 2011.

[17] Y.Kim and L.S .Kim, "64-bit carry-select adder with reduced area," *Electron. Lett.*, vol.37, no.10, May 2001.

[18] R.P.Brent and H.T.Kung, "ARegular Layout for Parallel Adders," *IEEE Transactions on Computers*, vol. C-31, no.3, pp.260–264, 1982.

[19] J. Sklansky, "Conditional-sum addition logic," *IRE Transactions on Electronic Computers*, vol. 9, pp. 226–231, 1960.

[20] http://en.wikipedia.org/wiki/Kogge%E2%80%93Stone_adder

[21] Haridimos T. Vergos and GiorgosDimitrakopoulos, "On Modulo 2^n+1 Adder Design", *IEEE transactions on computers*, vol. 61, no. 2, February 2012.

[22] J.Rabaey, "Digital Integrated Circuits: A Design Perspective", Prentice Hill, Second Edition, 2003.

[23] S. Ghosh et al., "Tolerance to small delay defects by adaptive clock stretches," *IOLTS*, 2007.

[24] P. Ndai et al., "Fine-grained redundancy in adders," *ISQED*, 2007.

[25] W. J. Townsend et al., "Quadruple time redundancy adders," *DFT*, 2003.

[26] B. W. Johnson, "Design and analysis of fault tolerant digital systems," *Addison Wesley Publishing Company*, 1989.

[27] J. H. Patel et al., "Concurrent error detection in ALUs by recomputing with shifted operands," *Trans. Comput*., 1982.

[28] K. Wu et al., "Algorithm level Re-computing with shifted operands - a register transfer level concurrent error detection technique," *ITC*, 2000.

[29] Aguirre, M., M. Linares, "An Alternative Logic Approach to Implement High-Speed Low-Power Full Adder Cells", *In:Proceedings of 18th annual symposium on Integrated circuits and system design*, pp. 166-171, 2005.

[30] Junming, L et al, "A Novel 10-Transistor Low-Power High-Speed Full Adder Cell", *In: Proceedings of IEEE 6th International Conf. Solid-State & Integrated-Circuit Technology, Shanghai Jiao Tong Univ., China*, 2001.

[31] Shalem, R., E. John, L. K. John, "A novel low power energy recovery full adder cell", *In: Proceedings of The Ninth Great Lakes Symposium on VLSI, Ann Arbor, MI, USA*, 1999.

[32] Fayed, A. A., M. A. Bayoumi, "A Low Power 10-Transistor Full Adder Cell for Embedded Architectures", *In: IEEE International Symposium on Circuits & Systems Design, Sydney, Australia*, pp. 226-229, 2001.

[33] Radhakrishnan, D., A. P. Preethy, "Low-power CMOS pass logic 4-2 compressor for high-speed multiplication", *In: Proceedings of the 43rd IEEE Midwest Symposium Circuits & Systems*, pp. 1296-1298, 2000.

[34] N. Zhuang and H. Wu, "A new design of the CMOS full adder," *IEEE J. Solid-State Circuits*, vol. 27, no. 5, May 1992.

[35] Mariano Aguirre-Hernandez and Monico Linares-Aranda, "CMOS Full-Adders for Energy-Efficient Arithmetic Applications" *IEEE transactions on very large scale integration (VLSI) systems*, Vol. 19, no. 4, April 2011.