IOSR Journal of VLSI and Signal Processing (IOSR-JVSP)
Volume 7, Issue 5, Ver. I (Sep.-Oct. 2017), PP 53-61
e-ISSN: 2319 – 4200, p-ISSN No. : 2319 – 4197
www.iosrjournals.org

Low power and area efficient Parallel chein search Architecture.

I.Kanaka Durgamma[1], [2] K.Murthy Raju

M.Tech -VLSID Student, Department of Electronics and Communications Engineering, Shri Vishnu Engineering College for Women (Autonomous), Bhimavaram, India)[1]

Sr. Associate Professor, Department of Electronics and Communications Engineering, Shri Vishnu Engineering College for Women (Autonomous), Bhimavaram, India)[2]

Corresponding Author: I.Kanaka Durgamma

Abstract*: The short horizontal Bose-chaudhuri-Hocquenghem (BCH) Chien search for signs of a new power-saving(CS) structure is proposed. For syndrome-based decoding, CS plays an important role in identifying the areas of error, but incurs a huge waste of exhaustive computation power consumption. The proposed architecture, the process of searching for the binary representation of the matrix is decomposed in two steps. This is neither new low power architecture for parallel CS provided. By reducing access to the second stage of the conventional CS to achieve significant power savings is decomposed in two steps. Error operate under the same ownership, the less energy the size of the CS in the construction sector in different configurations, and error correction capability of the horizontal factor compared to traditional construction. Power saving horizontal factor or increase the size of the field will become more and more important.Further this project is enhanced by replacing by multiplier architecture with radix8 modified booth multiplication algorithm for more power and area reduction. Radix4 modified booth encoding algorithm produces 50 percent reduction in partial products.*

Keywords: *Bose-Chaudhuri- Hocquenghem (BCH) codes, ChienSearch (CS), low power,two step approach,modified booth encoding.*

Date of Submission: 14-10-2017 Date of acceptance: 31-10-2017

I. Introduction

Communications and storage systems for various error correction codes are used to recover the corrupted code words, Bose-chaudhuri-Hocquenghem(BCH) code [1], [2] is the most widely used due to its powerful error correction performance and affordable hardware complexity is one of the algebraic signs. Binary BCH code is a solid-state storage such forward [3], [4] and optical fiber communication systems [5], most of the applications and the never-ending demand for high throughput decoding has been running ever larger error-correction capability of different systems. Satisfying the huge computational capacity of high throughput and strong error correction is inevitable, therefore, becomes more and more important power saving structure of the BCH decoding.

In general, a BCH decoder to correct the bits T at the peak of the three main blocks, namely, the syndrome calculation (SC), the key-equation solving (KES) has, and Chien search (CS) [1], [2]. Receiving a code word for a given R (x) Compute syndromes SC 2T and KES (X) using the syndromes of the error locator polynomial Λ(X). Finally, the error is E (X) is CS determined by the algorithm is based on the finding. In a parallel BCH decoder, CS main cause of power consumption and total electricity consumption [6] and can take up to a half. Numerous studies have demonstrated the ability to reduce the power consumption of CS proposed structures. Early termination of the methods presented in [6] and [7].After finding an error in the past to eliminate redundant computations are. An additional error counter is incremented when an error is found, and the counter KES downsides found in the CS is turned off matches. BCH decoder dealing with a small number of errors early in the implementation of the common and effective drug, though, when the power saving small insignificant error correction capability, [8], is a more effective method in polynomial order reduction (POR) when the error was found in the error locator polynomial of the proposed reform. Locator polynomial order one at a time, errors are detected by the decline and eventually becomes zero. POR [8] at a time, gradually power down circuitry associated with a polynomial factor makes it impossible for the CS. POR for serial BCH decoders are successful, however, because it is difficult to apply the technique of complex polynomial update parallel architecture. Furthermore, all of the previous power saving algorithms, including early termination, [6], [7] and the POR [8], depending on the position of the errors. For example, if faults at the end of the term of the code, as in the case of power savings is significant that in the beginning of errors. In this brief, we have a new approach, which is parallel to the CS proposed two stages of decomposition. In order to have access to each of the first step, but the first step to access the second stage will be activated only when a less successful result. The proposed two-step method [9] that is conceptually similar. The two-step approach, in general, lead to an increase in the critical path delay and delay, the losses can be solved simply by employing an efficient pipelined architecture. Unlike previous architectures [6]-[8], regardless of the error, the location of the proposed construction of the power consumption can be saved.

The rest of the paper is structured as follows: In section II Two-step cs architecture,. III PROPOSED architecture the following section explains simulation results. In section IV, performance comparisons with other architectures are given. The final conclusion of this paper is shown in section V.

II. Two-Step Cs Architecture

As indicated in above, the p-parallel CS examines p error positions simultaneously, each of which generates a 1 × m binary matrix denoting a Galois field (GF) element by computing

$$Y(\alpha^{wp+i}) = \sum_{j=1}^{t} \mathrm{FFM}_{ij} = \sum_{j=1}^{t} \Omega_j A_{ij} = [\Omega_1\ \Omega_2 \cdots \Omega_t] \begin{bmatrix} A_{i1} \\ A_{i2} \\ \vdots \\ A_{it} \end{bmatrix}$$

Where i ranges from 1 to p. The CS determines the presence of an error when Y (α^{wp+i}) is 1, which implies that α^{wp+i} is a root of the error locator polynomial. In the GF of dimension m, the multiplicative identity element, α0 or α^{2m-1}, is defined as 1, i.e., $0_{(m-1:1)1(0)}$, more precisely. The main idea comes from the fact that the absence of errors is guaranteed if some bits of $Y^{(\alpha wp+i)}$ are not equal to those of $0_{(m-1:1)1(0).}$ In the case of GF (2^4), for example, no presence of errors is guaranteed if $Y^{(\alpha wp+i)}$(3:2) # 0. Two-step approach is employed for early detection.

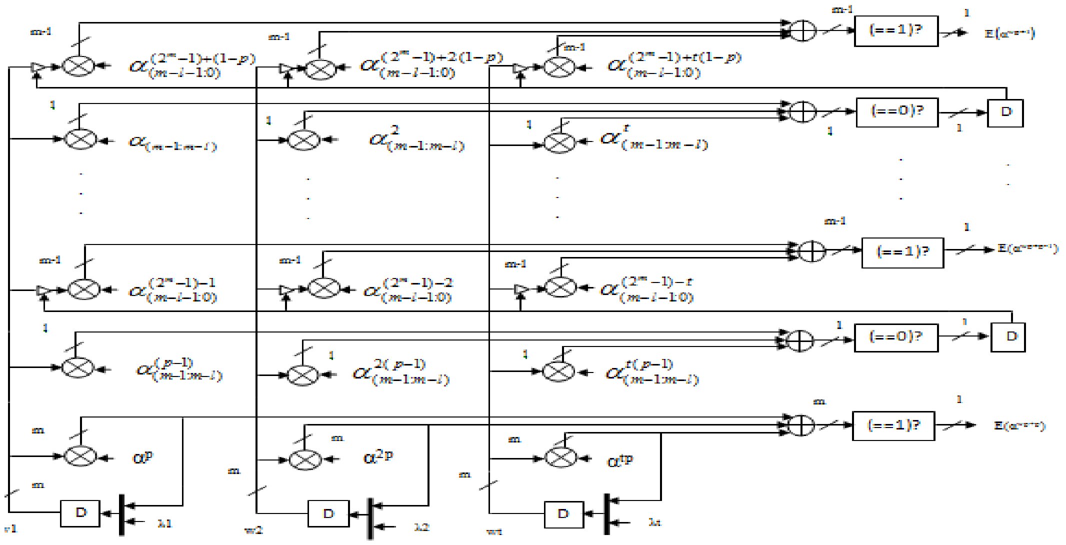

m-1
$\alpha^{(2^m-1)+(1-p)}_{(m-l-1:0)}$
$\alpha^{(2^m-1)+2(1-p)}_{(m-l-1:0)}$
$\alpha^{(2^m-1)+t(1-p)}_{(m-l-1:0)}$
(==1)?
E
(==0)?
D
1
$\alpha_{(m-1:m-l)}$
$\alpha^2_{(m-1:m-l)}$
$\alpha^t_{(m-1:m-l)}$
$\alpha^{(2^m-1)-1}_{(m-l-1:0)}$
$\alpha^{(2^m-1)-2}_{(m-l-1:0)}$
$\alpha^{(2^m-1)-t}_{(m-l-1:0)}$
$\alpha^{(p-1)}_{(m-1:m-l)}$
$\alpha^{2(p-1)}_{(m-1:m-l)}$
$\alpha^{t(p-1)}_{(m-1:m-l)}$
m
α^p
α^{2p}
α^{tp}
λ1
λ2
λt

Fig.1. Two step architecture for p-parallel CS.

Fig.1illustrates the low-power CS architecture based on the proposed two-step approach. According to, the m-bit FFMs in the conventional CS are replaced with the pipelined two partial FFMs except for those in the pth row.

Given the intermediate values from the registers, the first partial FFM processes the l MSBs and activates the second partial FFM responsible for the remaining m − l LSBs at the next clock cycle only when the output of the former is 0. Otherwise, we can reduce the dynamic switching power by disabling the latter partial FFMs. Since each intermediate register can hold one of all possible GF elements, the latter partial FFM is activated once every 2^l clock cycles on the average. Furthermore, it is worth noting that the sum of the hardware complexity for the former and the latter partial FFMs is almost the same as the conventional FFM. Therefore, additionally required in the proposed architecture are the p 1-bit registers and the (p − 1) t m-bit buffers.

III. Proposed Modified Booth Algorithm For Two-Step Architecture

This modified booth multiplier is used to perform high-speed multiplications using modified booth algorithm. This modified booth multiplier's computation time and the logarithm of the word length of operands are proportional to each other is shown in Fig1 . We can reduce half the number of partial product. Radix-8 booth algorithm used here increases the speed of multiplier and reduces the area of multiplier circuit in Fig 2.

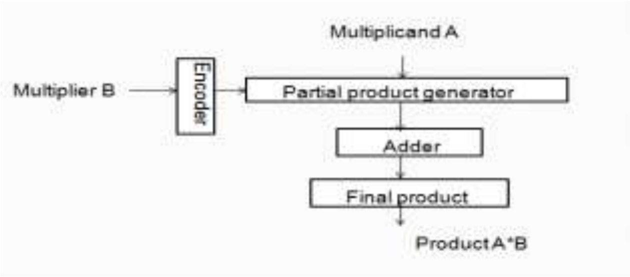


Fig.2. General block diagram of Booth multiplier.

Multiplier architecture comprise of two architectures, i.e., Modified Booth and Wallace tree. Based on the study of various multiplier architectures, we find that Modified Booth increases the speed because it reduces partial products to half. Further, the delay in multiplier can be reduced by using Wallace tree. Power consumption of Wallace tree multiplier is also less as compared to booth and array. Features of both multipliers can be combined to produce high speed and low power multiplier. Modified Booth multiplier consists of Modified Booth Recorder (MBR). MBR have two parts, i.e., Booth Encoder (BE) and Booth Selector (BS). The basic operation of BE is to decode the multiplier signal and output will be used by BS to generate the partial product. The partial products are then, added with the Wallace tree adders, similar to the carry save adder approach. The last row of carry and sum output is added together by Ripple carry adder.

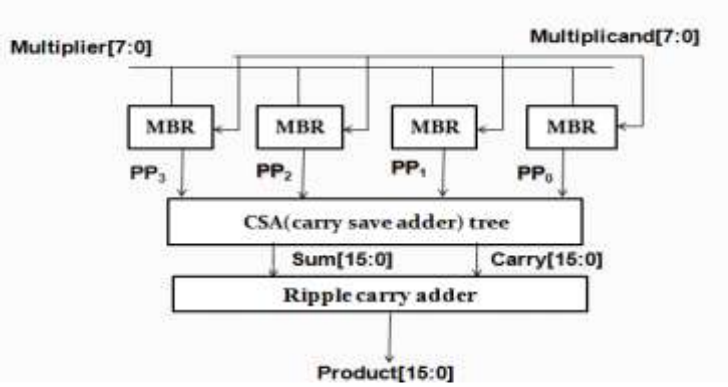


Fig.3. Block diagram of Radix-8 Booth algorithm.

Radix-8 Booth encoding is most often used to avoid variable size partial product arrays. Before designing Radix-8 BE, the multiplier has to be converted into a Radix-8 number by dividing them into four digits respectively according to Booth Encoder Table given after wards. Prior to convert the multiplier, a zero is appended into the Least Significant Bit (LSB) of the multiplier.

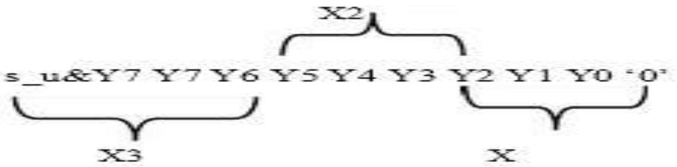


Fig.4. Grouping of bits.

Radix 8 Booth recoding applies the same algorithm as that of Radix 4, but now we take quartets of bits instead of triplets. Each quartet is codified as a signed digit using below Table. Radix 8 algorithm reduces the number of partial products to n/3, where n is the number of multiplier bit s. Thus it allows a ti me gain in the partial products summation Radix-8 recoding applies the same algorithm as radix-4, but now we take quartets of bits instead of triplets. Each quartet is codified as a signed-digit using the table.

Table.1. Partial product generator of Encoding table for radix-8 modified Booth multiplier

Group of multiplier bits	Operation to be perform on multiplicand

0000	**0**
0001	**1xMultiplicand**
0010	**1xMultiplicand**
0011	**2xMultiplicand**
0100	**2xMultiplicand**
0101	**3xMultiplicand**
0110	**3xMultiplicand**
0111	**4xMultiplicand**
1000	**-4xMultiplicand**
1001	**-3xMultiplicand**
1010	**-3xMultiplicand**
1011	**-2xMultiplicand**
1100	**-2xMultiplicand**
1101	**-1xMultiplicand**
1110	**-1xMultiplicand**
1111	**0**

Partial product generator

A product formed by multiplying the multiplicand by one digit of the multiplier when the multiplier has more than one digit. Partial products are used as intermediate steps in calculating larger products.

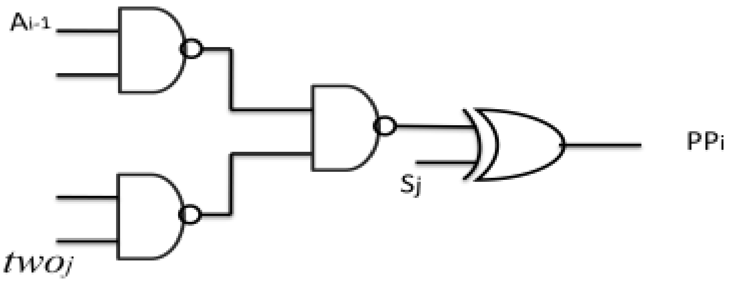


Fig.5. Block diagram of partial product generator.

Partial product generator is designed to produce the product by multiplying the multiplicand A by 0, 1, -1, 2, -2,-3,-4, 3, 4. For product generator, multiply by zero means the multiplicand is multiplied by “0”.Multiply by “1” means the product still remains the same as the multiplicand value. Multiply by “-1” means that the product is the two’s complement form of the number. Multiply by “-2” is to shift left one bit the two’s complement of the multiplicand value and multiply by “2” means just shift left the multiplicand by one place.

SIGN EXTENSION CORRECTOR

Sign Extension Corrector is designed to enhance the ability of the booth multiplier to multiply not only the unsigned number but as well as the signed number. The working principle of sign extension that converts signed multiplier signed unsigned multiplier as follows. One bit control signal called signed unsigned(s_u) bit is used to indicate whether the multiplication operation is signed number or unsigned number .when sign unsign s_u=0, it indicates unsigned number multiplication and when s_u=1, it indicates signed number multiplication.

Table.2. Sign extension corrector

Sign- unsigned	Type of operation
0	Unsigned multiplication
1	Signed multiplication

RIPPLE CARRY ADDER

A ripple carry adder is a digital circuit that produces the arithmetic sum of two binary numbers. It can be constructed with full adders connected in cascaded, with the carry output from each full adder connected to the carry input of the next full adder in the chain. Fig 6 shows the interconnection of four full adder (FA) circuits to provide a 4-bit ripple carry adder. Notice from below Figure that the input is from the right side because the first cell traditionally represents the least significant bit (LSB). Bits a0 and b0 in the figure represent the least significant bits of the numbers to be added. The sum output is represented by the bits S0-S3.

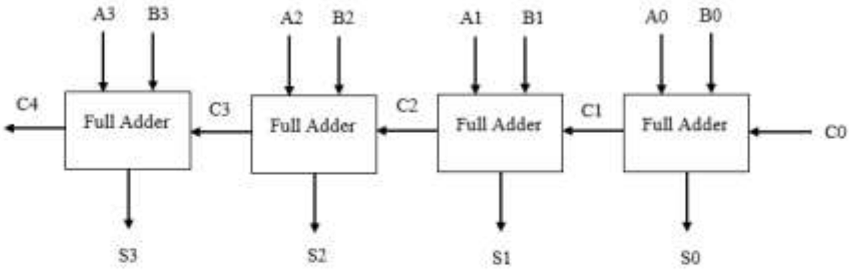


Fig.6. Ripple carry adder.

CARRY-SAVE ADDER

It is a type of digital adder, used in computer micro architecture to compute the sum of three or more *n*-bit numbers in binary. It differs from other digital adders in that it outputs two numbers of the same dimensions as the inputs, one which is a sequence of partial sum bits and another which is a sequence of carry bits.

$$ps_i = a_i \oplus b_i \oplus c_i sc_i = (a_i \wedge b_i) \vee (a_i \wedge c_i) \vee (b_i \wedge c_i)$$

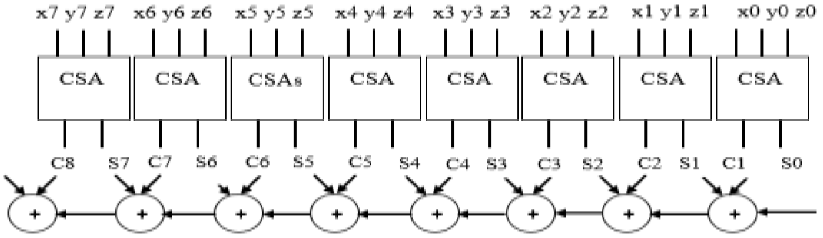


Fig.7. Architecture of Carry Save Adder.

IV. Simulation Results

The Two step structure for parallel CS has been designed. The programming language used in this is Verilog HDL and simulated using Xilinx ISE 14.7 and ISIM simulator. Design properties are Spartan 6family, FGG900 package, XC6SLX150T device with a speed grade --3.

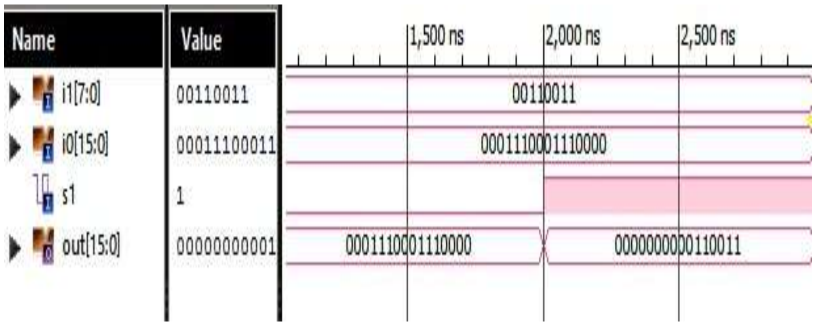


Fig .8. Simulation results for 2x1 multiplexer.

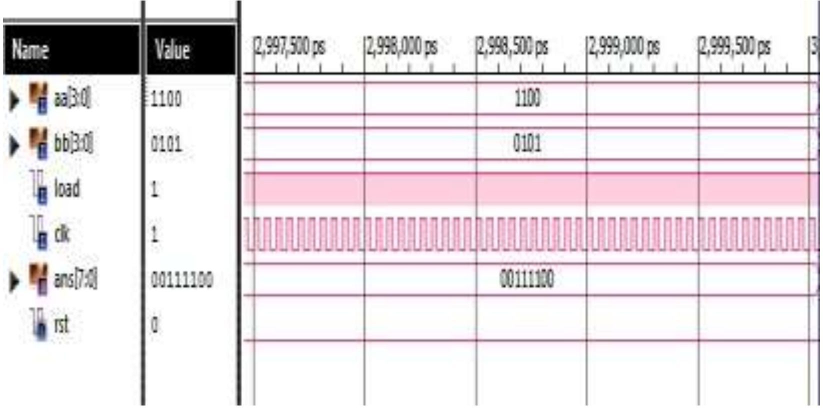
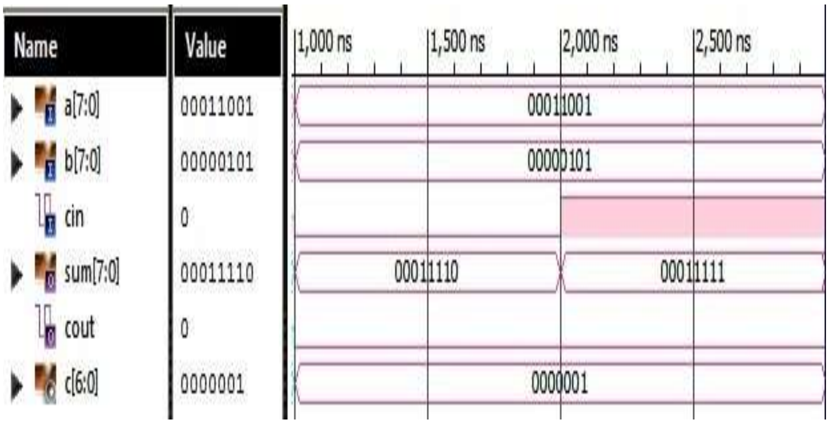


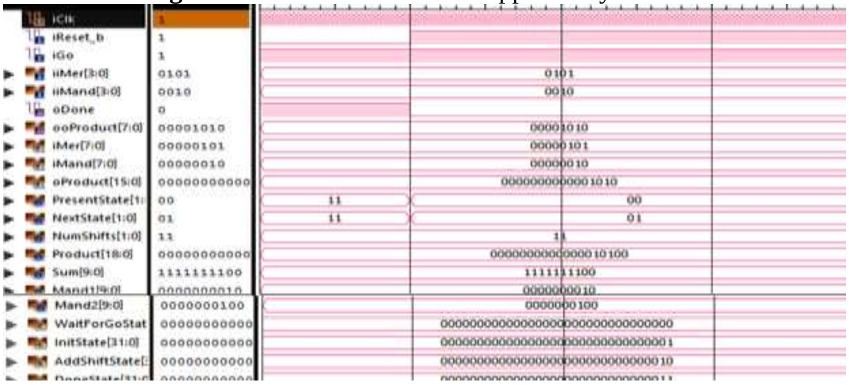
Fig .9. Simulation results for four bit serial multiplier.

Fig 8 shows the simulation result for 2x1 multiplexer ,in this having two inputs i0,i1 depending on selection line outputs are produced. Fig 9 shows the simulation result for 4-bit serial multiplier here multiplication operation is performed. Fig 10 shows the simulation result for Ripple carry adder based on c_{in} value get the final outputs . Fig 11 shows the simulation result for Modified Booth Multiplier, in this following algorithm to perform multiplication operation. Fig 12 and Fig 13 shows the simulation result for conventional Two-Step structure for Parallel Chien Search architecture using modified Booth multiplier ,where it gives eight outputs by adding two inputs and Fig 14 shows the simulation result for The RTL Schematic of two-step structure for parallel CS using modified Booth multiplier.



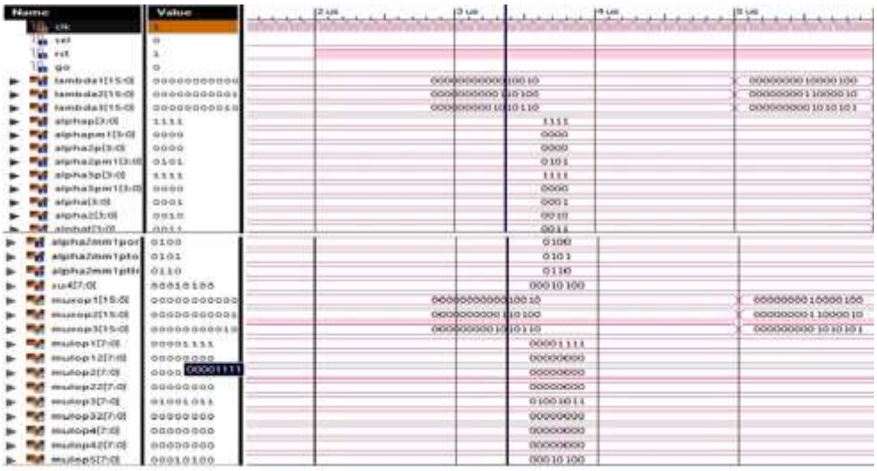
Name	Value
a[7:0]	00011001
b[7:0]	00000101
cin	0
sum[7:0]	00011110
cout	0
c[6:0]	0000001

Fig .10. Simulation results for Ripple Carry Adder.



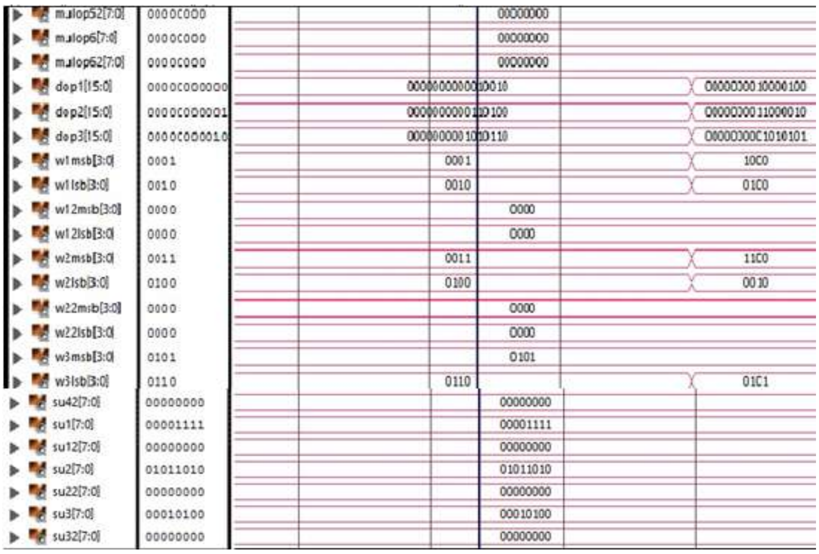
Name	Value
iClk	1
iReset_b	1
iGo	1
iiMer[3:0]	0101
iiMand[3:0]	0010
oDone	0
ooProduct[7:0]	00001010
iMer[7:0]	00000101
iMand[7:0]	00000010
oProduct[15:0]	00000000000
PresentState[1:	00
NextState[1:0]	01
NumShifts[1:0]	11
Product[18:0]	00000000000
Sum[9:0]	1111111100
Mand1[9:0]	0000000010
Mand2[9:0]	0000000100
WaitForGoStat	00000000000
InitState[31:0]	00000000000
AddShiftState[:	00000000000
DoneState[31:0	00000000000

Fig .11. Simulation results for Modified Booth multiplier.



Name	Value
clk	1
sel	0
rst	1
go	0
lambda1[15:0]	00000000000
lambda2[15:0]	00000000001
lambda3[15:0]	00000000010
alphap[3:0]	1111
alphapm1[3:0]	0000
alpha2p[3:0]	0000
alpha2pm1[3:0]	0101
alpha3p[3:0]	1111
alpha3pm1[3:0]	0000
alpha[3:0]	0001
alpha2[3:0]	0010
alpha3[3:0]	0011
alpha2mm1po	0100
alpha2mm1pto	0101
alpha2mm1pth	0110
su4[7:0]	00010100
muxop1[15:0]	00000000000
muxop2[15:0]	00000000001
muxop3[15:0]	00000000010
mulop1[7:0]	00001111
mulop12[7:0]	00000000
mulop2[7:0]	0000 00001111
mulop22[7:0]	00000000
mulop3[7:0]	01001011
mulop32[7:0]	00000000
mulop4[7:0]	00000000
mulop42[7:0]	00000000
mulop5[7:0]	00010100

Fig .12. Two step architecture using Modified Booth multiplier for p-parallel CS.



mulop52[7:0]	0000C000
mulop6[7:0]	0000C000
mulop62[7:0]	0000C000
dop1[15:0]	0000C00000000
dop2[15:0]	0000C00000001
dop3[15:0]	0000C00000010
w1msb[3:0]	0001
w1lsb[3:0]	0010
w12msb[3:0]	0000
w12lsb[3:0]	0000
w2msb[3:0]	0011
w2lsb[3:0]	0100
w22msb[3:0]	0000
w22lsb[3:0]	0000
w3msb[3:0]	0101
w3lsb[3:0]	0110
su42[7:0]	00000000
su1[7:0]	00001111
su12[7:0]	00000000
su2[7:0]	01011010
su22[7:0]	00000000
su3[7:0]	00010100
su32[7:0]	00000000

Fig .13. Two step architecture using Modified Booth multiplier for p-parallel CS contd.

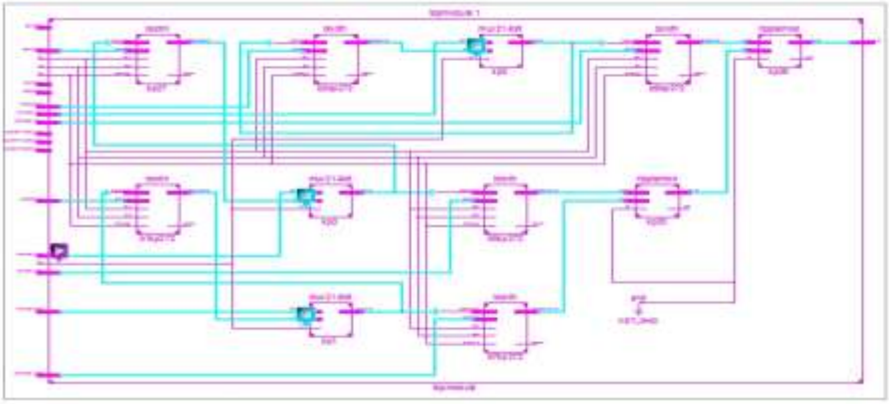


Fig .14. RTL Schematic for Two step architecture using Modified Booth multiplier for p-parallel CS.

IV. Comparisons

Table.3. Power comparison table for Two step architecture using serial multiplier for p-parallel CS and Two step architecture using Modified Booth multiplier for p- parallel CS.

Power report of the system	**Power in watts**
Two step architecture using serial multiplier for p-parallel CS	2.043
Two step architecture using Modified Booth multiplier for p- parallel CS	1.455

Table.3. shows the Power comparison table for Two step architecture using serial multiplier for p-parallel CS and Two step architecture using Modified Booth multiplier for p- parallel CS. in this Two step architecture using Modified Booth multiplier for p- parallel CS reduces o.588W than Two step architecture using serial multiplier for p-parallel CS.

Table.4. Comparison between Two step architecture using serial multiplier for p-parallel CS and Two step architecture using Modified Booth multiplier for p- parallel CS.

S.No	Parameters	Two step architecture using serial multiplier for p-parallel CS	Two step architecture using modified Booth algorithm for p-parallel CS
1	Number of slice registers(in %)	3	1
2	Number of fully used LUT-FF pairs (in %)	22	17
3	Number of bonded IOBs	65	11
4	Dynamic Power	1.822	1.276
5	Quiescent Power	0.221	0.178
6	Total Power	2.043	1.455

Two step architecture using Modified Booth multiplier for p- parallel CS is compared with Two step architecture using serial multiplier for p-parallel CS in various parameters like dynamic power, quiescent power, number of slice registers, number of fully used LUT-FF pairs, and number of bonded IOBs. The implementation results give the same outputs, but in power consumption and area is less compared with Two step architecture using serial multiplier for p-parallel CS.

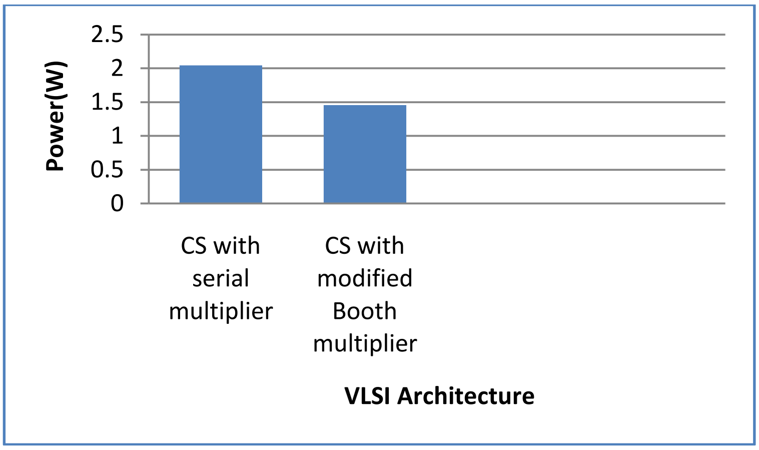


Fig.15. Power comparison of CS with serial multiplier and CS with modified Booth multiplier.

From Fig.15 we understood the power of CS with modified Booth multiplier is less compared with CS with serial multiplier. It says that fast output is appeared at CS with modified Booth multiplier.

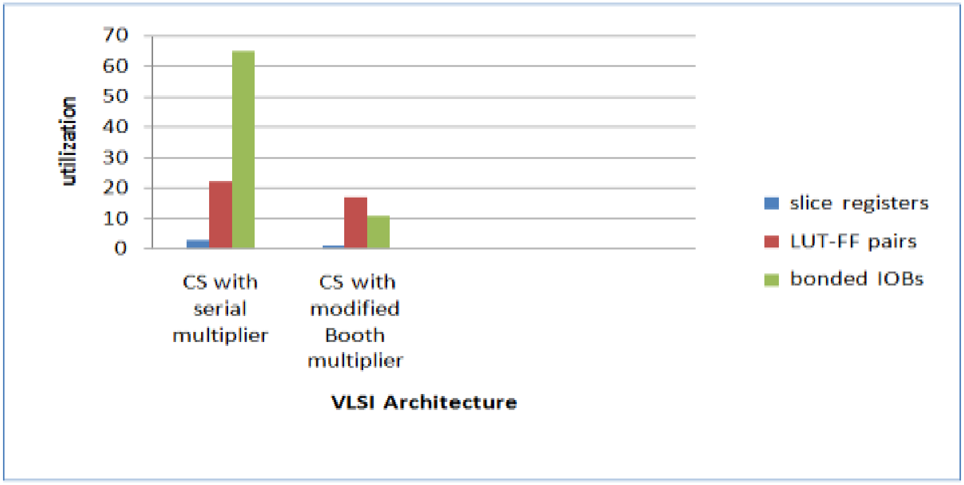


Fig.16. Comparison of area of CS with serial multiplier and CS with modified Booth multiplier.

From Fig.16 we understood the area of CS with modified Booth multiplier is less compared with CS with serial multiplier. It says that fast output is appeared at CS with modified Booth multiplier.

V. Conclusion

This is a new low-power architecture for parallel CS provided. By reducing access to the second stage of the conventional CS to achieve significant power savings is decomposed in two steps. Error operate under the same ownership, the less energy the size of the CS in the construction sector in different configurations, and error-correction capability of the horizontal factor compared to traditional construction. Final implementation with Radix 8 modified booth encoding algorithm yields reduction in density and power consumption.

Acknowledgements

The authors would like to thank the our beloved Principal Dr. G. Srinivasa Rao and Vice Principal Dr. P. Srinivasa Raju of Shri Vishnu Engineering College for Women for guidance and encouragement to do research by providing facilities and also like to acknowledge the anonymous reviewers for their suggestions that helps to improve the presentation of this paper.

References

[1]. Y. Lee, H. Yoo, and I.-C. Park, “High-throughput and low-complexity BCH architecture for solid- state drives,” IEEE Trans. Very Large Scale Integr. Syst., vol. 22, no. 5, pp. 1183–1187, May 2014.

[2]. Y. Lin, C. Yang, C. Hsu, H. Chang, and C. Lee, “A MPCN-based parallel architecture in BCH decoder for NAND Flash memory devices,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 58, no. 10, pp. 682– 686,Oct. 2011.

[3]. Y. Lee, H. Yoo, and I.-C. Park, “Low-complexity parallel Chien search structure using two-dimensional optimization,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 58, no. 8, pp. 522–526, Aug. 2011.

[4]. J. Cho and W. Sung, “Strength-reduced parallel Chien search architecture for strong BCH codes,” IEEE Trans. Circuits Syst. II, Exp. Briefs, vol. 55, no. 5, pp. 427–431, May 2008.

[5]. S. Wong, C. Chen, and Q. M. Wu, “Low power Chien search for BCH decoder using RT-level power management,” IEEE Trans. Very Large Scale Integr. Syst., vol. 19, no. 2, pp. 338–341, Feb. 2011.

[6]. Y. Wu, “Low power decoding of BCH codes,” in Proc. IEEE ISCAS, May 2004, pp. II-369–II-372.

[7]. Y. Chen and K. K. Parhi, “Small area parallel Chien search architectures for long BCH codes,” IEEE Trans. Very Large Scale Integr. Syst., vol. 12, no. 5, pp. 545–549 May 2004.

[8]. H. Weingarten, E. Sterin, O. A. Kanter, and M. Katz, “Low Power Chien-Search Based BCH/RS Decoding System for Flash Memory, Mobile Communications Devices and Other Applications,” U.S. Patent 2010 013 1831 A1, May 27, 2010.

[9]. H. Choi, W. Liu, and W. Sung, “VLSI implementation of BCH error correction for multilevel cell NAND Flash memory,” IEEE Trans. Very Large Scale Integr. Syst., vol. 18, no. 5, pp. 843–847, May 2010.

IOSR Journal of VLSI and Signal Processing (IOSR-JVSP) is UGC approved Journal with Sl. No. 5081, Journal no. 49363.

I.Kanaka Durgamma “Low power and area efficient parallel chein search architecture.” IOSR Journal of VLSI and Signal Processing (IOSR-JVSP) , vol. 7, no. 5, 2017, pp. 53-61.