AREA AND POWER OPTIMIZED CHIEN SEARCH ARCHITECTURE USING MODIFIED BOOTH ENCODING

E.Ramakrishna Reddy[1],M.Rammohan Rao[2] ,K.Babu Rao[3] ,L.Prasanthi[4]

[1]Asst. Professor., Electronics & Communication Engg., Usha Rama College Of *Engineering*
[2] Assot.Professor., Electronics & Communication Engg., Usha Rama College Of *Engineering*
[3] Assot.Professor., Electronics & Communication Engg., Usha Rama College Of *Engineering*
[4] M.Tech ,Student Electronics & Communication Engg., Usha Rama College Of *Engineering*

ABSTRACT: *This* brief proposes a new power-efficient Chien Search(CS) architecture for parallel Bose–Chaudhuri–Hocquenghem (BCH) codes. For syndrome-based decoding, the CS plays a significant role in finding error locations, but exhaustive computation incurs a huge waste of power consumption. In the proposed architecture, the searching process is decomposed into two steps based on the binary matrix representation. Unlike the first step accessed every cycle, the second step is activated only when the first step is successful, resulting in remarkable power saving. Furthermore, an efficient architecture is presented to avoid the area and delay increase in critical paths caused by the two-step approach. Further this project is enhanced by replacing by multiplier architecture with radix16 modified booth multiplication algorithm saves power consumption by up to 50% compared with the conventional architecture and reduce area.

***Key Words*:** Chien Search, Bose–Chaudhuri–Hocquenghem (BCH), Polynomial Order Reduction (POR), Radix-16 Booth multiplier.

1. INTRODUCTION

Communications and storage systems for various error correction codes are used to recover the corrupted code words, Bose-Caudhuri 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. Finally, the error is E (X) Λ sources (X) 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 order polynomial 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 become 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 and reduces the total partial products by radix-16 booth multiplier. 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 and save area and power consumption. 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.

Issue 7 volume 3 May-June2017
ISSN 2249-9954

II TWO-STEP CS ARCHITECTURE

Let us consider a binary BCH (*n*, *k*, *t*) code over GF (2*m*), Where *n* is the code length, *k* is the message length, and *t* is the maximal number of correctable error bits. More precisely, $n = k + mt$, where is the field dimension that satisfies $2m - 1 \geq n$. During the syndrome-based decoding, the error Locator polynomial delivered by the KES is expressed as

$$\Lambda(x) = \sum_{j=1}^{t} \lambda_j x^j + 1 = Y(x) + 1.$$

the *p*-parallel CS examines *p* error positions simultaneously, each of which generates a $1 \times m$ 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 $Y(\alpha^{wp+i})$ 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, α^0 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 $Y(\alpha^{wp+i})_{(3:2)} \neq 0$. Similar to, a two-step approach is employed for early detection. In other words, the possibility of error presence is found by searching only the *l* MSBs rather than the entire *m* bits. Using this property, can be decomposed into two matrix multiplications as

$$Y(\alpha^{wp+i}) = [\Omega_1\ \Omega_2\ \cdots\ \Omega_t] \times \left(\left[\begin{array}{c|c} A_{i1,(m-1:m-l)} & 0_{(m-l-1:0)} \\ A_{i2,(m-1:m-l)} & 0_{(m-l-1:0)} \\ \vdots & \vdots \\ A_{it,(m-1:m-l)} & 0_{(m-l-1:0)} \end{array}\right] + \left[\begin{array}{c|c} 0_{(m-1:m-l)} & A_{i1,(m-l-1:0)} \\ 0_{(m-1:m-l)} & A_{i2,(m-l-1:0)} \\ \vdots & \vdots \\ 0_{(m-1:m-l)} & A_{it,(m-l-1:0)} \end{array}\right] \right)$$

$$= \mathrm{concat}\left\{ \Omega(w) \begin{bmatrix} A_{i1,(m-1:m-l)} \\ A_{i2,(m-1:m-l)} \\ \vdots \\ A_{it,(m-1:m-l)} \end{bmatrix}, \Omega(w) \begin{bmatrix} A_{i1,(m-l-1:0)} \\ A_{i2,(m-l-1:0)} \\ \vdots \\ A_{it,(m-l-1:0)} \end{bmatrix} \right\}.$$

Similar to [9], a two-step approach is employed for early detection.

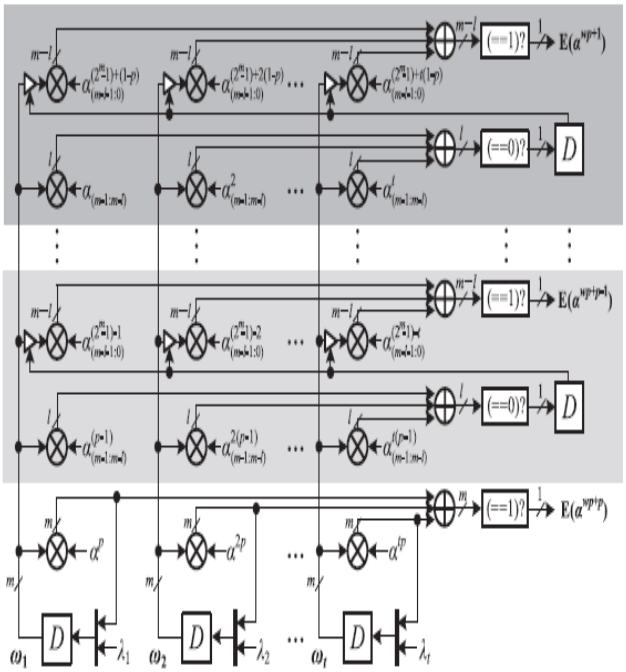


Fig: Two-step structure for *p*-parallel CS

The above figure illustrates the low-power CS architecture based on the proposed two-step approach. According to (8), 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 2l 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. It is important to decide how many *l* bits are appropriate for the former partial FFMs. The more bits are examined in the former, the latter partial FFMs will be accessed less resulting in a large power reduction, but the former partial FFM will suffer from the increased power dissipation, and vice versa.

III RADIX-16 MODIFIED BOOTH ENCODING ALGORITHM

Booth multiplier is mostly used multiplier. The number of partial products rows that must be added to give the multiplication's result can be reduced by using Booth decoding. In Booth multiplier, the numbers of reduced partial products rows are depend on the grouping done at multiplier bits. These groups of multiplier perform the selected operation on multiplicand. In booth multiplier grouping is done by 2 bits, 3 bits, 4 bits and so on. Higher order booth decoding reduces the number of partial product rows by a greater by decoding larger groups of multiplier bits.

This multiplication process is completed in 3 steps. First step: multiplier bits are divided in groups then these groups are fed to decoder at where it will indicate that which operation is to perform on multiplicand. Second step: here indicated operation performs on the multiplicand and it will generate the partial products. Third step: Now generated partial products are adding with adders

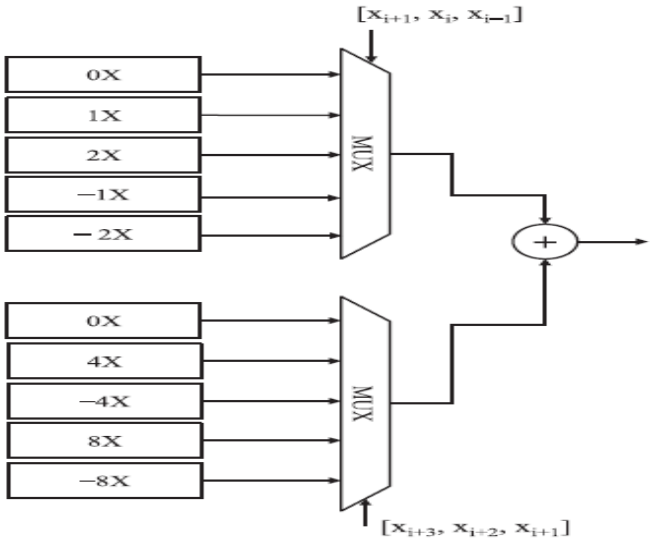


Fig: Radix16 modified booth encoder

Booth encoding was originally introduced when multiplication was implemented using a series of shift-add operations. By recoding, the number of 1's in the multiplier could be reduced, and thereby the number of additions. The core idea is as follows: Introduce new variables: $b\hat{}i=-b_i+b_{i-1}$ for $i=0\ldots n-1$ (assume $b-1=0$).

Compute $p = \sum_{i=0}^{n-1} (b\hat{}i \times 2i \times A)$

Although this looks very similar to the normal product, note that any time $bi=bi-1$ there is no addition to be performed as the partial product will be zero. In the case of serial addition, these steps can be skipped, thus saving computation.

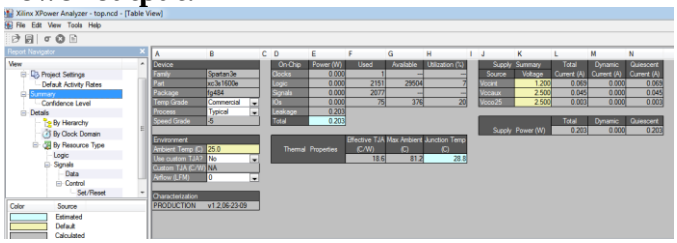
X_{i+3}	X_{i+2}	X_{i+1}	X_i	X_{i-1}	PP
0	0	0	0	0	0Y
0	0	0	0	1	1Y
0	0	0	1	0	1Y
0	0	0	1	1	2Y
0	0	1	0	0	2Y
0	0	1	0	1	3Y
0	0	1	1	0	3Y
0	0	1	1	1	4Y
0	1	0	0	0	4Y
0	1	0	0	1	5Y
0	1	0	1	0	5Y
0	1	0	1	1	6Y
0	1	1	0	0	6Y
0	1	1	0	1	7Y
0	1	1	1	0	7Y
0	1	1	1	1	8Y
1	0	0	0	0	-8Y
1	0	0	0	1	-7Y
1	0	0	1	0	-7Y
1	0	0	1	1	-6Y
1	0	1	0	0	-6Y
1	0	1	0	1	-5Y
1	0	1	1	0	-5Y
1	0	1	1	1	-4Y
1	1	0	0	0	-4Y
1	1	0	0	1	-3Y
1	1	0	1	0	-3Y
1	1	0	1	1	-2Y
1	1	1	0	0	-2Y
1	1	1	0	1	-1Y
1	1	1	1	0	-1Y
1	1	1	1	1	0Y

Fig: Radix16 modified booth encoding table

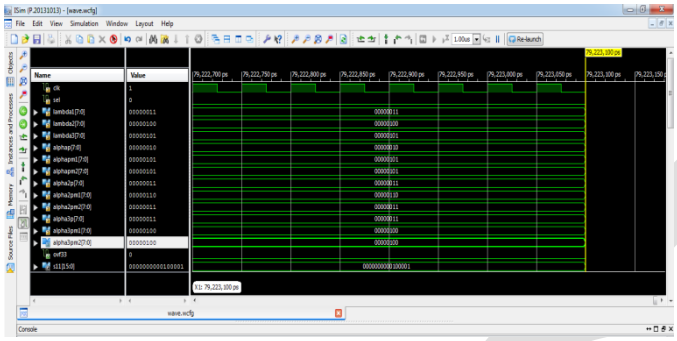
To reduce the number of partial products added while multiplying the multiplicand higher radix Booth Encoding algorithm is one of the most well known techniques used .Radix 16 Booth algorithm which scan strings of five bits with the algorithm given below: (1) Extend the sign bit 1 position if necessary to ensure that n is even. (2) Append a 0 to the right of the LSB of the multiplier (3) According to the value of each vector , each partial product will be 0, y, +2y, +3y, +4y, +5y,+6y,+7y,+8y, -8Y, - 7y, - 6y, - 5y, - 4y, -3y, - 2Y, - Y. The multiplication of y is done by shifting y by one bit to the left. Thus, in any case, in designing n bit parallel multipliers, only n/4 partial products are generated.

IV Simulation Results

Power output:



Synthesis Result:



Device utilization summary:

Name of the Device	No of Gates	% of Utilization
Number of Slices	1142 out of 14752	7%
Number of Slice Flip Flops	44 out of 29504	0%
Number of 4 input LUTs	2117 out of 29504	7%
Number of IOs	99	
Number of bonded IOBs	75 out of 376	19%
Number of GCLKs	1 out of 24	4%

V CONCLUSION

This is 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 16 modified booth encoding algorithm yields reduction in density and power consumption.

REFERENCES:

[1] Jiun-Ping Wang, Shiann-Rong Kuang, Shish Chang Liang, "High Accuracy Fixed Width Modified Booth Multipliers for Lossy Applications", IEEE Trans 2011.
[2] Pouya Asadi and Keivan Navi, "A new low power 32×32 bit multiplier", World Applied Sciences Journal 2 (4): 341-347,2007.
[3] A.D. Booth, "A signed binary multiplication technique", Quart. J. Mech. Appl. Marh, vol.4, pp. 236-240, 1951. (Reprinted in [8, pp. 100-104])
[4] Razaidi Hussin, Ali Yeon Md. Shakaff, Norina Idris, Zaliman Sauli, Rizala fande CheIIsmail, and Afzan Kamaraudin, "An efficient modified Booth multiplier architecture", International Conference on Electronics Design, 978-1-4244-2315-6/08, 2008 IEEE.
[5] Ramya Muralidharan, Chip Hong Chang," Radix-4 and Radix-8 Booth encoded multi-modulas multipliers" IEEE Trans, 2013.
[6] Vignesh Kumar R., Kamala J., "High Accuracy Fixed Width Multipliers using Modified Booth Algorithm",International Conference on Modeling Optimization and Computing, Procedia Engineering 38(2012) 2491-2498.
[7] Y. Wu, "Low power decoding of BCH codes," in Proc. IEEE ISCAS, May 2004, pp. II-369–II-372.
[8] 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.
[9] 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.
[10] 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.
[11] 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.
[12] 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.
[13] 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.