Multiple Error Correction Using BCH and Cross Parity Codes in Reduced Time Over GF(2^m)

V.Sharmila Raj[1] , K.Abjith[2]

Associate professor, Dept. of ECE, SSM College of Engineering, Erode, Tamilnadu, India[1]

PG Student [Applied Electronics], Dept. of ECE, SSM College of Engineering, Erode, Tamilnadu, India[2]

Abstract—Here presents a novel low-complexity cross parity code, with a wide range of multiple bit error correction capability at a lower overhead, in circuits over GF(2^m) for improving the reliability. For an m input circuit, the proposed scheme can correct m D_w $3^{m=2}$ 1 multiple error combinations out of all the possible 2^m 1 errors, which is superior to many existing approaches. Tests on 80-bit parallel and, for the first time, on 163-bit Federal Information Processing Standard/National Institute of Standards and Technology (FIPS/NIST) standard word-level Galois field (GF) multipliers, suggest that it requires only 106% and 170% area overheads, respectively, which is lower than the existing approaches.

Keywords—Galois field, Error correction

I. INTRODUCTION

Hardware based cryptography has gained significant popularity in applications such as bank automated teller machines, mobile communications, and security applications. Transient error-based fault attacks, single event upset, multiple event upset, along with manufacturing faults and noisy operating environment, possess a real threat to security infrastructures. Several approaches have been proposed to mitigate errors to improve the reliability of hardware circuits, among which on-line error detection and correction is noteworthy .

In such a condition have to reduce errors in all means and hence can obtain perfect system. So here introduced the error correction topic, which has done in many ways. Different coding decoding techniques are used to do this. They are capable of correcting the errors in a larger manner, but some of the errors are detected and can not be located, if it is located , can not be corrected. The existing system clears a lot errors in it, using the BCH coding, decoding techniques. The BCH codes have better error detection coverage than the Hamming codes , which prefers here for the scheme.

II. BCH CODES

The Bose, Chaudhuri, and Hocquenghem (BCH) codes form a large class of powerful random error-correcting cyclic codes.This class of codes is a remarkable generalization of the Hamming code for multiple error correction. Given a prime power q and positive integers m and d with d qm 1, a primitive narrowsense BCH code over the finite field GF(q) with code length n = qm 1 and distance at least d is constructed by the following method. Let be a primitive element of GF (qm). For any positive integer i, let mi(x) be the minimal polynomial of i over GF(q). The generator polynomial of the BCH code is defined as the least common multiple g(x) = lcm(m1(x); ; md 1(x)). It can be seen that g(x) is a polynomial with coefficients in GF(q) and divides xn 1. Therefore, the polynomial code defined by g(x) is a cyclic code.

A. Primitive binary BCH codes

For any positive integers m 3 and t < $2^{m\ 1}$, there exists a binary BCH code with the following parameters:

$$\text{Blocklength} : n = 2^m \quad 1 \qquad (1)$$

$$\text{N umberof parity checkdigits} : n \ k \quad mt \qquad (2)$$

M inimumdistance : d_{min} 2t + 1 (3)

Can call this code a t-error-correcting BCH code.Let α be a primitive element in GF(2^m). The generator polynomial g(x) of the t-error-correcting BCH code of length $2^m - 1$ is the lowest-degree polynomial over GF(2^m) which has
α; α^2; α^3; α^{2t} as its roots.
$g(\alpha^i) = 0$ for $1 \leq i \leq 2t$ and g(x) has α; α^2; α^3; α^{2t} and their conjugates as all its roots.

B. Generator polynomial of binary BCH codes

The generator polynomial g(x) of a t-error-correcting primitive BCH codes of length 2^m 1 is given by

$$g(x) = LCM \{\varphi_1(x); \varphi_3(x); :::; \varphi_{2t-1}(x)\} \quad (4)$$

Note that the degree of g(x) is mt or less. Hence the number of parity-check bits, n-k, of the code is at most mt. Note that the generator polynomial of the binary BCH code is originally found to be the least common multiple of the minimum polynomials φ_1; φ_2; :::; φ_{2t-1}

$$\text{i:e: } g(x) = LCM \{\varphi_1(x); \varphi_3(x); :::; \varphi_{2t-1}(x), \varphi_{2t}(x)\} \quad (5)$$

However, generally, every even power of a in GF(2m) has the same minimal polynomial as some preceding odd power of a in GF(2m). As a consequence, the generator polynomial of the t-error-correcting binary BCH code can be reduced to

$$g(x) = LCMf_1(x); \ _3(x); :::; \ _{2t-1}(x)g \quad (6)$$

III. CROSS CODE PARITY

To detect multiple or t faults (t ¿ 1) also with even multiplicity, the calculation of the Cross-parity is introduced. It is based on the calculation of (at least) three parity vectors: the row-parity rX, the column-parity cY, and the diagonal-parity dZ.

Consider first the observation of an imaginable register file. If Y is the bitwidth of a register and X is the number of registers, the Cross-Parity is composed from the row-parity vector r[x (X 1) : 0], the column-parity vector c[y (Y 1) : 0] and the diagonal parity vector d[z max(X; Y) 1 : 0]. According to the 'little endian' mode, the diagonal parity can be calculated from the most significant bit of the most significant register to the least significant bit of the least significant register. The combined check of row- column- and diagonal-parity is the so called cross-parity check.

$$rX = M_{X,Y-1} \oplus M_{X,Y-2} \oplus M_{X,Y-3} \oplus \ldots \oplus M_{X,0} \quad (7)$$

$$cY = M_{X-1,Y} \oplus M_{X-2,Y} \oplus M_{X-3,Y} \oplus \ldots \oplus M_{0,Y} \quad (8)$$

IV. BLOCK DIAGRAM

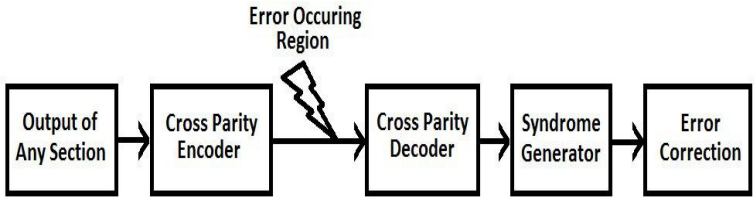

Fig. 1. Generic block diagram of cross parity based error correction architecture

The block diagram of the error correction technique using cross parity decoder is shown here. Which consists of three blocks? They all as a whole gives finest outcome from the project.a novel cross parity algorithm for multiple error correction using the error detection features of the BoseChoudhury Hocquenghem (BCH) codes cross coupled with output parity prediction. The proposed technique corrects a wide range of multiple errors with acceptable area overhead, and comprises a novel decoding architecture for multiple error correction. The design technique has been tested with bitparallel circuits of various complexities and with FIPS/NIST standard 163-bit digit-serial cryptographic multipliers. The other blocks are,

1. Functional Block
2. Cross Code Parity Decoder
3. Correction Block

A. Cross parity decoder

A novel methodology for multiple bit error correction in logic circuits, which relies only on the error detection features of the BCH codes and simple parity prediction, is proposed, thus achieving a tradeoff between the area overhead and error tolerance by simply avoiding the complex decoding in the existing techniques.

Here calculates the row parity, column parity and diagonal parity and using these values we can locate the error position. From that we can obtain the output corrected by performing the modulo 2 addition, of the parity of the remaining bits except the erroneous row or column or diagonal and the parity of output. Thus can correct the output from errors.

B. Correction logic

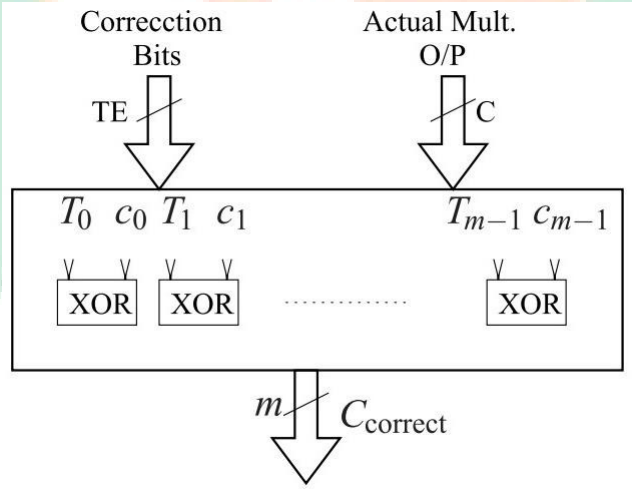

Fig. 2. Internal details of correction logic

Correction logic corrects the output errors by the values obtained from the Functional block and the cross code parity predictor. Here which checks if there is any error , and if identifies any error ,which is corrected by performing the modulo 2 addition of the both input terms to this block.

Registers are used to save data in the systems. Parity is obtained from a matrix formed values, which is obtained by making the values of registers are given to a column and the next to below it. Thus the length of the column determines the size of the register. And the number of registers gives the idea about the row size.

There are delay flip flops which gives the proper delay and makes the whole values of parity obtained in a particular manner. The input message given are processed and will give the proper way of the output takings, and gives the column parity.

The GF(2^m) multiplication can be performed by a straightforward polynomial multiplication followed by modular reduction. Considered irreducible trinomials as reduction polynomials and showed that a modular multiplication operation

in GF(2^m) can be performed with (! 1)(m 1) bit additions, where ! is the Hamming weight of the irreducible polynomial.

Block IP(m) corresponds to an inner product unit which has two input vectors of m elements each. Assuming that only two input logic gates are used, IP(k) for k 0, requires k AND gates and k - 1 XOR gates and has a gate delay of TA + $d\log_2 k$dTX, where TA and TX correspond to the delays due to an AND and an XOR gate respectively.

Here consider the time delay and gate count of the proposed multiplier as a function of degree m and the reduction matrix Q.Using the Q matrix, the complexities of multipliers based on special reduction polynomials, namely: 1) trinomials, 2) ESPs, and 3) two classes of pentanomials are obtained. We also present explicit formulas for multiplication for the above three special classes. These formulas maximize the number of intermediate signals that are reused. These formulas can be easily coded using hardware description languages such as VHDL or Verilog to implement an optimized multiplier.

Thus the multiplier circuit forms the functional block of the circuit, which uses in the cross code parity checking also. Since the same block is duplicated there to perform the proper functioning of the entire system. The input of the block are the message bit and the (n-k) bit and the product formation there occurs .

Functional block may be a cryptographic system unit, the output of the functional block may occure errors . Which can be corrected using the correction logic which is implementing here. The given input value is multiplied and then which undergo some cyclic shift and then again the values are multiplied. The whole ideas are shown in such a way that the cyclic shift and the multiplication both can done using the circuit.

V. BASIC CONCEPT

The three blocks are the major parts of this project. The first block gives the message and can obtain the cross code parity from the other block, and they both are given to the correction logic. The block correction logic will check the both inputs of it and determines if they are wrong or not. Based on that which is processed to get the correct data by performing modulo 2 addition.

In hardware implementation, its multiplication operations can be realized with m^2 AND and $(m\ 1)^2$ + (! 1)(m 1) XOR gates. Also present explicit formulas for multiplication for the above three special classes. These formulas maximize the number of intermediate signals that are reused. These formulas can be easily coded using hardware description languages such as VHDL or Verilog to implement an optimized multiplier. These codings can be done by a hardware designer without running an algorithm for precomputation or even having any knowledge of finite field arithmetic.

No. of Bits	Error Correction Capability in Hamming Code	Error Correction Capability in BCH Code
10	142%	160%
15	123%	152%
20	121%	140%
48	105%	116%
64	104%	114%
90	101%	106%

TABLE I.AREA OVERHEAD COMPARISON OF VARIOUS MULTIPLIER SIZES

The area overhead of the proposed technique is shown in Table. Clearly, the area overhead for both BCH and Hamming-based cross parity schemes are close. This is because we are using only the error detection part of the BCH codes. The area overhead for a simple 10-bit multiplier is 142%. As the multiplier size grows, the percentage area overhead due to the parity generation circuit and the correction logic grow much slowly. For example, in contrast, the area overhead of an 80-bit multiplier with multiple error

VI. CONCLUSION

The system needs a spectacular way of approach to get the correct output. When we consider the multipliers and other blocks , they are easy to implement. But the multiplier will vary in accordance with the selected primitive polynomial.

Primitive polynomial has a major role in the total system performance. The blocks can be implemented with suitable way and are simulated.

By proposing a particular system which reuses the block which comes more than once. Then can reduce the effective area and also the time delay of the circuit. Thus the circuit gets reduced , and they will produce less delay in the way of output and the propagation too.

Made the functional block,register single block, correction logic of the simulation. Functional block gives the output and the same data is required to make the syndrome values. Register block are implemented and works as a storage and the source as giving input. Finally the correction is done at the final point of view.
The authors would like to thank...

REFERENCES

1) Mahesh Poolakkaparambil,Jimson Mathew,Abusaleh M. Jabir and Dhiraj K. Pradhan ”A Low-Complexity Multiple Error Correcting Architecture Using Novel Cross Parity Codes Over GF(2m)” VERY LARGE SCALE INTEGRATION (VLSI) SYSTEMS,VOL. 23, NO. 8, AUGUST 2015

2) D. Boneh, R. A. DeMillo, and R. J. Lipton, On the importance of eliminating errors in cryptographic com-putations, J. Cryptol., vol. 14, no. 2, pp. 101119, 2001.

3) M. Pflanz, K. Walther, C. Galke, and H. Vierhaus, On-line error detection and correction in storage elements with cross-parity check, in Proc. 8th IEEE Int. On-Line Test. Workshop, 2002, pp. 6973.

4) J. Mathew, A. M. Jabir, H. Rahaman, and D. K. Pradhan, Single error correctable bit parallel multipliers over GF(2m), IET Comput. Digital Techn., vol. 3, no. 3, pp. 281288, May 2009.

5) M. Poolakkaparambil, J. Mathew, A. M. Jabir, D. K. Pradhan, and S. P. Mohanty, BCH code based multiple bit error correction in finite field multiplier circuits, in Proc. 12th IEEE Int. Symp. Qual. Electron. Des., Mar. 2011, pp. 615620.

W. T. Huang, C. Chang, C. W. Chiou, and F. H. Chou, Concurrent error detection and correction in a polynomial basis multiplier over GF(2n), IET Inf. Security, vol. 4, no. 3, pp. 111124, Sep. 2010.

6) A. Reyhani-Masoleh and M. A. Hasan, Low complexity bit parallel architectures for polynomial basis multipli-cation over GF(2m), IEEE Trans. Comput., vol. 53, no. 8, pp. 945959, Aug. 2004.

