An Efficient Interpolation-Based Chase BCH Decoder

D. Prabhakar[1] B. Rajesh[2] B. Shirisha[3]
[1]M.Tech Student [2,3]Assistant Professor
[1,2,3]Department of Electronics and Communication Engineering
[1,2,3] Vathsalya Institute of science and technology, R.R Dist,T.S, India

Abstract— Error correction codes are the codes used to correct the errors occurred during the transmission of the data in the unreliable communication mediums. The idea behind these codes is to add redundancy bits to the data being transmitted so that even if some errors occur due to noise in the channel, the data can be correctly received at the destination end. Bose,Ray Chaudhuri, Hocquenghem (BCH)codes are one of the error correcting codes. The BCH decoder consists of four blocks namely syndrome block, chien search block and error correction block. This paper describes a new method for error detection in syndrome and chien search block of BCH decoder. The proposed syndrome block is used to reduce the number of computation by calculating the even number syndromes from the corresponding odd number syndromes.

Keywords: BCH Codes, Syndrome Block, Chien search Block, Error detection

I. INTRODUCTION

BCH CODES can be found in manyapplications, including flash memory, optical transport network, and digital video broadcasting. For a BCH code with minimum distance $d_{\min}$, traditional hard-decision decoding (HDD) algorithms, such as the Berlekamp's algorithm, can correct $t = d_{\min}/2$ errors. Through flipping the η least reliable bits and trying 2^{η} test vectors,the soft-decision Chase algorithm can correct up to $t + \eta$ errors. It also has better error-correcting performance than the generalized minimum distance (GMD) decoder and the soft-decision decoder in [1], which assumes that all but one error are located in the $2t$ least reliable bits. To reduce the complexity of the Chase BCH decoding, one-pass schemes have been proposed to derive the error locators for all test vectors in one run based on the Berlekamp's algorithm [2], [3]. Comparatively, the scheme in [2] has lower complexity.

This brief proposes a novel interpolation-based Chase BCH decoder. It is inspired by the interpolation-based Chase decoder for Reed–Solomon (RS) codes. Nevertheless, by making use of the binary property of BCH codes, substantial modifications and simplifications are developed in this brief. In particular, instead of employing expensive parallel Chien search, the selec-tion of the interpolation output leading to successful decoding is achieved by simple evaluation value computation without any error-correcting performance or code rate loss. In addition, the recovery of each code word bit is done through testing the evaluation values of two low-degree polynomials. From architectural analysis, the proposed decoder with $\eta = 4$ for an example (4200, 4096) BCH code has 2.3 times higher efficiency in terms of throughput-over-area ratio than the Chase decoder based on the Berlekamp's algorithm [4], while achieving the same error-correcting performance.

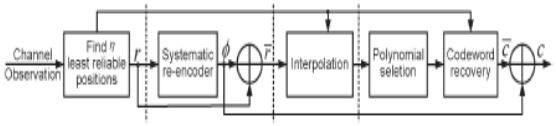
The structure of this brief is as follows. Section II introduces the Chase decoding. The interpolation-based Chase BCH decoding and its implementation architectures are proposed in Sections III and IV, respectively. Section V provides the hardware complexity analysis, and conclusions are drawn in Section VI

II. CHASE DECODING

The fundamental difference between a ST Chase decoder and previously published decoders is that the ST Chase decoder concentrates on correcting bit errors instead of symbol errors. In this section, we will combineideas from sphere decoding and binary block decoding to create a low complexity MIMO decoder.

III. INTERPOLATION-BASED CHASE BCH DECODER

An (n, k) t-bit error-correcting BCH code over $GF(2^p)$ is a subfield subcode of an $(n\ k)$ t-symbol error-correcting RS code over $GF(2^p)$. In another word, all the (n, k) BCH code words form a subset of the (n, k) RS code words. $n-k = 2t$, and $n - k$ is equal to or slightly less than pt. The interpolation-based decoding is developed based on the interpretation that the code word symbols are evaluation values of the message poly-nomial. BCH codes cannot be encoded this way since the eval-uation values of a binary message polynomial over finite field elements are usually not binary. Hence, BCH code words are considered as RS code words in order to apply the interpolation-based decoding. Applying RS systematic re-encoding to the last k code positions, $n - k$ points remain to be interpolated for each test vector. The same backward–forward interpolation scheme [7], [8] can be adopted to derive the interpolation results of all vectors in one run. Nevertheless, by making use of the property that r is binary in BCH decoding, substantial simplifications can be made on the polynomial selection and code recovery section.



High level decoder design

The BCH decoder has four modules as mentioned below:

- Syndrome Calculator
- Solving the key equation
- Error Location
- Error Correction

IV. SYNDROME CALCULATOR

The syndrome calculator is the first module at the decoder also, the design of this module is almost same for all the BCH code decoder architecture. The input to this module is

corrupted codeword. The equations for the codeword, received bits and the error bits are given in equations (5.1), (5.2), (5.3)[4].

Codeword equation

$c(x) = c_0 + c_1x + c_2x^2 + ... + c_{n-1}x^{n-1}$

Received bits equation

$r(x) = r_0 + r_1x + r_2x^2 + ... + r_{n-1}x^{n-1}$

Error bits equation

$e(x) = e_0 + e_1x + e_2x^2 + ... + e_{n-1}x^{n-1}$

Thus, the final transmitted data polynomial equation is given as below:

$r(x) = c(x) + e(x)$

The 1st step at the decoding process is to store the transmitted data polynomial in the buffer register and then to calculate the syndromes sj. The important characteristic of the syndromes is that depends on only error location not on transmitted information. The equation of the syndromes are given as follows [4]:

Define the syndromes S_j as

$$S_j = \sum_{i=0}^{n-1} r_i \alpha^{i \cdot j}$$ for $(1 \le j \le 2t)$. Since $r_j = c_j + e_j$

(j = 0, 1, ..., n-1)

Rewrite the syndrome equation as:

$$S_j = \sum_{i=0}^{n-1} (c_i + e_i)\alpha^{i \cdot j} = \sum_{i=0}^{n-1} c_i \alpha^{i \cdot j} + \sum_{i=0}^{n-1} e_i \alpha^{i \cdot j}$$

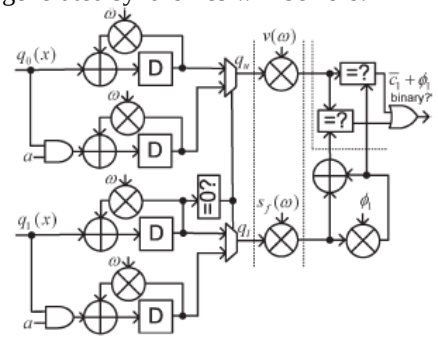
By the definition of BCH codes

$$\sum_{i=0}^{n-1} c_i \alpha^{i \cdot j} = 0 \text{ for } (1 \le j \le 2t)$$

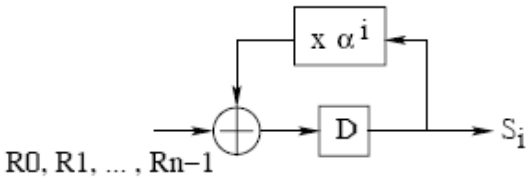
Thus,

$$S_j = \sum_{i=0}^{n-1} e_i \alpha^{i \cdot j}$$

The above equation is indicates the output of the syndrome calculator. From the equation it can be observed that the syndromes are depends on only error polynomial e(x), so if there is no error occurs during the transmission then all the generated syndromes will be zero.



Architecture For Proposed Polynomial Selection



Conventional Syndrome Calculator

V. KEY EQUATION SOLVER

The second stage in the decoding process is to find the co-efficient of the error location polynomial using the generated syndromes in the previous stage. The error location polynomial is given as: $\sigma(x) = \sigma_0 + \sigma_1x + ... + \sigma_tx^t$. The relation between the syndromes and the error location polynomial is given as below [4]:

$$\sum_{j=0}^{t} S_{t+i-j}\sigma_j = 0 \text{ (i= 1, ..., t)}$$

There are various algorithms used to solve the key equation solver. This project is using the Inversion less Berlekamp Massey algorithm to solve the key equation. Berlekamp Massey Algorithm.

The steps of berlekamp Massey algorithm is given as below:

(1) First step is to calculate error syndromes Sj.
(2) Initialize the k = 0, $\Lambda^{(0)}(x) = 1$, L = 0 and T(x) = x
(3) Assign k = k + 1 and then the discrepancy $\Delta^{(k)}$ is then calculated as follows:

$$\Delta^{(k)} = S_k - \sum_{i=1}^{L} \Lambda_i^{(k-1)} S_{k-i}$$

(4) If the value of $\Delta^{(k)}$, 2L ≥ equals 0, then go-to step 7.
(5) Calculate the $\Lambda^{(k)}(x) = \Lambda^{(k-1)}(x) - \Delta^{(k)} T(x)$
(6) Set the value of L = k – L and T(x) is calculated as $T(x) = \Lambda^{(k-1)}(x) / \Delta^{(k)}$
(7) Set T(x) = x.T(x).
(8) If the value of k< 2t , then go-to step 3
(9) Continue for i = 2t – 1 and then End.

The decoder of this project is based on the Inversion-less Berkelamp algorithm (iBM) for Key Equation Calculation. The architecture for iBM algorithm is explained in detail in the next chapter.

VI. ERROR LOCATION – CHAIN'S SEARCH

To calculate the error location is the next step of decoding process, which can be done using chain search block.

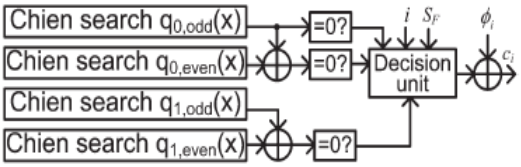
A. Chain Search Algorithm

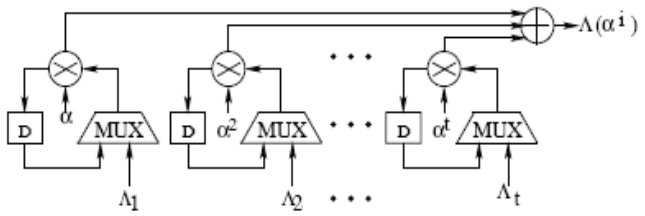
The roots are calculated as follows [12] [4]:

(1) For each power of α for ($j = 0$ *to* $n - 1$), α^j is taken as the test root
(2) Calculate the polynomial coefficients, of the current root using, coefficients of the past iteration, using, $\mathbf{\Lambda_i^{(j)} = \Lambda_i^{(j-1)} \alpha^i}$ during the j^{th} iteration
(3) Calculate the sum of the polynomial coefficients

$$\sum_{i=1}^{t} \Lambda_i^j = 1$$

(4) The sum is equal to
(5) Continue to Step 1 till j = n-1





Chain's Search architecture – Error Location

VII. Error Correction

The output of the chain search block is called roots of equation. The reciprocal of the roots of equations are added with the corresponding location of the corrupted codeword received by decoder. The result of this addition is the original codeword that was encoded by the encoder before transmission.

Conclusion

This brief developed an efficient interpolation-based onepass Chase BCH decoder. By making use of the binary property of BCH codes, novel polynomial selection and code word recovery schemes were proposed. In particular, the proposed polynomial selection led to significant complexity reduction without sacrificing the error-correcting performance. Future work will be directed to further speeding up the interpolationbased BCH decoder without incurring large area overhead.

References

[1] Y.-M. Lin, H.-C. Chang, and C.-Y. Lee, "An improved soft BCH decoder with one extra error compensation," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2010, pp. 3941–3944.

[2] Y. Wu, "Fast Chase decoding algorithms and architectures for Reed–Solomon codes," *IEEE Trans. Inf. Theory*, vol. 58, no. 1, pp. 109– 129, Jan. 2012.

[3] N. Kamiya, "On algebraic soft-decision decoding algorithms for BCH codes," *IEEE Trans. Inf. Theory*, vol. 47, no. 1, pp. 45–58, Jan. 2001.

[4] X. Zhang, J. Zhu, and Y. Wu, "Efficient one-pass Chase soft-decision BCH decoder for multi-level cell NAND flash memory," in *Proc. IEEE Int. Midwest Symp. Circuits Syst.*, Aug. 2011, pp. 1–4.

[5] W. J. Gross, F. R. Kschischang, R. Koetter, and P. Gulak, "A VLSI architecture for interpolation in soft-decision decoding of Reed- Solomon codes," in *Proc. IEEE Workshop Signal Process. Syst.*, Oct. 2002, pp. 39–44.

[6] R. Kötter, "On algebraic decoding of algebraic-geometric and cyclic codes," Ph.D. dissertation, Dept. Elect. Eng., Linköping Univ., Linköping, Sweden, 1996.

[7] X. Zhang and J. Zhu, "Algebraic soft-decision decoder architectures for long Reed-Solomon codes," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 57, no. 10, pp. 787–792, Oct. 2010.

[8] X. Zhang and Y. Zheng, "Systematically re-encoded algebraic softdecision Reed-Solomon decoder," *IEEE Trans. Circuits Syst. II, Exp. Briefs*, vol. 59, no. 6, pp. 376–380, Jun. 2012.

[9] C.-S. Choi and H. Lee, "High throughput four-parallel RS decoder architecture for 60 GHz mmWAVE WPAN systems," in *Proc. Int. Conf. Consum. Electron.*, 2010, pp. 225–226.

[10] K. Lee, H.-G. Kang, J.-I. Park, and H. Lee, "A high-speed lowcomplexity concatenated BCH decoder architecture for 100 Gb/s optical communications," *J. Signal Process. Syst.*, vol. 66, no. 1, pp. 43–55, Jan. 2012.

