An Efficient Interpolation-Based Chase BCH Decoder

P. V. Raja Reddy¹, K. Bala Krishna²

¹M. Tech Student, Department of Electronics and Communication Engineering, VRS & YRN College of Engineering Technology, Chirala, A.P., India
²Assistant Professor, Department of Electronics and Communication Engineering, VRS & YRN College of Engineering Technology, Chirala, A.P., 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

1. Introduction

BCH CODES can be found in many applications, 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 \(\eta\) least reliable bits and trying \(2^\eta\) 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 \(2^\eta\) 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 selection 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 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 combine ideas from

Figure 1: High level decoder design

The BCH decoder has four modules as mentioned below:

- Syndrome Calculator
- Solving the key equation
- Error Location
- Error Correction
3. 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

\[
c(x) = c_0 + c_1x + c_2x^2 + \ldots + c_nx^{n-1}\]

Received bits equation
\[
r(x) = r_0 + r_1x + r_2x^2 + \ldots + r_nx^n\]

Error bits equation
\[
e(x) = e_0 + e_1x + e_2x^2 + \ldots + e_nx^n\]

Thus, the final transmitted data polynomial equation is given as below:
\[
r(x) = r(x) + e(x)
\]
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 equations of the syndromes are given as follows [4]:

Define the syndromes \( S_j \) as
\[
S_j = \sum_{i=0}^{n-1} r_i \alpha^{i-j} \quad \text{for} \quad (1 \leq j < 2t).
\]

Rewrite the syndrome equation as:
\[
S_j = \sum_{i=0}^{n-1} (c_i + e_i) \alpha^{i-j} = \sum_{i=0}^{n-1} c_i \alpha^{i-j} + \sum_{i=0}^{n-1} e_i \alpha^{i-j}
\]

By the definition of BCH codes
\[
S_j = \sum_{i=0}^{n-1} e_i \alpha^{i-j} = 0 \quad \text{for} \quad (1 \leq j < 2t)
\]

Thus,
\[
S_j = \sum_{i=0}^{n-1} e_i \alpha^{i-j} = 0
\]
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.

4. Key Equation Solver

The second stage in the decoding process is to find the coefficient of the error location polynomial using the generated syndromes in the previous stage. The error location polynomial is given as: \( \sigma(x) = \sigma_0 + \sigma_1x + \ldots + \sigma_x^x \) The relation between the syndromes and the error location polynomial is given as below [4]:
\[
\sum_{i=0}^{t} S_{j-i} \sigma_j = 0 \quad (i = 1, \ldots, t)
\]

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 \( S_j \).
2. Initialize the \( k = 0 \), \( L = 0 \) and \( T(x) = x \).
3. Assign \( k = k + 1 \) and then the discrepancy \( \Delta^k \) is calculated as follows:
\[
\Delta^k = S_k - \sum_{i=1}^{t} \Lambda^{k-i} S_{k-i}
\]
(4) If the value of \( \Delta^k \), \( 2L \geq \text{equals} \ 0 \), then go-to step 7.
5. Calculate the \( \Lambda^k(x) = \Lambda^{k-1}(x) / \Lambda^0(x) \).
6. Set the value of \( L = k - L \) and \( T(x) \) is calculated as
\[
T(x) = \Lambda^{k-1}(x) / \Lambda^k(x)
\]
(7) Set \( T(x) = x, T(x) \).
(8) If the value of \( k \leq 2t \), 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 Berlekamp algorithm (iBM) for Key Equation Calculation. The architecture for iBM algorithm is explained in detail in the next chapter.

5. 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 \( \alpha \) for \( j = 0 \) to \( n - 1 \), \( \alpha^j \) is taken as the test root
2. Calculate the polynomial coefficients, of the current root using coefficients of the past iteration, using, \( \Lambda_{i-1}^{(j)} = \Lambda_{i}^{(j-1)} \alpha^j \) 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
4. Continue to Step 1 till \( j = n-1 \)

Figure 2: Architecture for Proposed Polynomial Selection

Figure 3: Conventional Syndrome Calculator
6. 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.

7. Simulation Results

The lfsr encoder is shown below

Error detected shown below

Corrected shown below

Syndrom cal shown below

8. Conclusion

This brief developed an efficient interpolation-based one pass 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 interpolation based BCH decoder without incurring large area overhead.
References


