A Fresh Look at the Berlekamp-Massey Algorithm with Application to Low Power BCH decoding

Ishai Ilani, *Western Digital*

Abstract— **BCH codes are gaining renewed attention lately as new applications based on Storage Class Memories (SCM) and fast NAND require codes with multiple error correction capability, and low latency decoding. BCH are a natural choice for an Error Correcting Code (ECC) for such applications. The main challenge for high speed decoding of such codes is computing the Error Locator Polynomial (ELP). Two main algorithms used for this challenge are the extended Euclidean (eE) and the Berlekamp-Massey (BM) algorithms. Efficient architectures for implementing the BM algorithm, (by simultaneous computation of discrepancies and updates), for decoding the wider class of Reed Solomon (RS) and BCH codes result in equivalence between BM algorithms and eE algorithms, and in some cases BM algorithms may have properties which will make them more attractive. In this paper we focus our attention to the class of Primitive Narrow Sense BCH codes. In this case the efficient architectures become even more efficient, and the area and power requirements for implementing the BM algorithm reduce by 33%. Combining these results with computation of the BM algorithm in a low latency mode result in a very efficient low power and low latency algorithm. Also a closer look at the parity check matrix of a RS or BCH code reveals a natural way for introducing the ELP into the algorithm, and a close similarity between the BM and eE algorithms.**

Index Terms—**BCH code, Primitive narrow-sense BCH code, Error Locator Polynomial, Berlekamp Massey algorithm.**

I. Introduction

The classic way of decoding BCH codes is a three step algorithm. During the first step the syndromes are computed. The second step computes an Error Locator Polynomial (ELP), and the third step finds the roots of the ELP from which the error locations may be deduced. The second step is considered the hardest step and it may be solved by an Euclidean (eE) algorithm or by the Berlekamp Massey (BM) algorithm (first presented in [1] and [2]). It was shown by [3] that it is possible to reformulate the BM algorithm to achieve extremely regular decoder architectures. In the original BM algorithm, [2], a discrepancy parameter, δ, is computed at each iteration. This parameter is used for updating a polynomial $C(D)$, (in the indeterminate D), which will eventually converge to the ELP. The discrepancy is computed as a scalar product of the coefficients of the current $C(D)$ and a vector of coefficients taken from the syndrome polynomial. The multiplications in such a scalar product may be done in parallel, but adding all the products together in a binary adder tree of depth $\log_2(t)$ requires a delay of at least $\log_2(t)$. The original BM algorithm defined an additional auxiliary polynomial $B(D)$, and both $B(D)$ and $C(D)$ were updated on each iteration as a function of one another. The main observation of [3] is that the discrepancy parameter δ is a coefficient in $SC(D)$, where $SC(D) = S(D)C(D)$. The polynomial $SC(D)$ together with an auxiliary polynomial $SB(D) = S(D)B(D)$ may be updated together in the exact same way as $B(D)$ and $C(D)$.

The updates of the polynomials involve one multiply and one add operation per coefficient of $C(D)$ and $SC(D)$, and one mux operation per coefficient of $B(D)$ and $SB(D)$, thus the latency of computing the discrepancy may be reduced significantly.

In this paper we take a closer look at the BM algorithm as introduced by [3] for the important special case of a *primitive narrow sense* BCH code. A BCH code is called *primitive narrow sense* if the roots of the generating polynomial include $\alpha, \alpha^2, \alpha^3, \ldots, \alpha^{2t}$, where α is a primitive element of the Galois field. In this case the discrepancy parameter is read from even powers of $SC(D)$. Moreover, all the polynomials $B(D)$, $C(D)$, $SB(D)$, and $SC(D)$ may be written as a sum of an *even* polynomial and an *odd* polynomial, and the *even* and *odd* polynomials are updated independently of each other. It follows that the *odd* portions of $SB(D)$ and $SC(D)$ may be discarded, and only the *even* portions of $SB(D)$ and $SC(D)$ need to be computed. Since the required length of $SB(D)$ and $SC(D)$ is twice the length of $B(D)$ and $C(D)$ this results in a reduction of 33% in memory and power requirements.

BCH codes are gaining renewed attention lately as new memory technologies, such as Storage Class Memory (SCM) require very low latency codes. Legacy LDPC codes may not be applicable for SCM, since long codes would result in high latency, and short LDPC codes do not exhibit performance advantages over BCH codes, and even short LDPC codes cannot guarantee short latency. The results of this paper may be implemented in SCM based products to reduce area and power of BCH decoders.

While studying the simplified algorithms for BM, and comparing to the eE algorithm we found a simple an elegant way to describe and develop the key equation, where the ELP arises naturally, and the BM algorithm may be explained in a more conceptual form, which is very similar to the Euclid

I. Ilani is with Western Digital Corp. Kfar Saba, Israel (e-mail: Ishai.Ilani@wdc.com).

algorithm. We shall study this similarity first, and then return to the simplified architectures mentioned above.

In tribute to Massey the notation in this paper follows the notations of [2], so the polynomials are polynomials in the indeterminate D.

II. Natural development of The key equation

Suppose a received codeword has an error vector $\boldsymbol{e} = (e_0, e_1, \dots, e_{n-1})^T$ with support at $i(1), i(2), \dots, i(L)$, i.e. the received codeword has L errors at coordinates $i(1), i(2), \dots, i(L)$, whose values are $Y_{i(1)}, Y_{i(2)}, \dots, Y_{i(L)}$. The syndrome vector $S = (s_1, s_2, \dots, s_{2t})^T$ defined as

$$S = H\boldsymbol{e} \ , \tag{1}$$

is a linear combination of the $i(1), i(2), \dots, i(L)$ columns of the matrix H, where each of the columns is a *geometric progression*. The decoding problem may be directly translated to the following statement

$$\begin{gathered}\textit{find } L \le t \textit{ geometric progressions such that}\\ S \textit{ is a linear combination of them.}\end{gathered} \tag{2}$$

However this problem is too difficult to solve directly, so we turn to look for an indirect approach which will combine properties of geometric progressions with a uniqueness theorem to generate a decoding algorithm. Transforming the columns of H and S to a polynomial representation and using the well-known formula of a sum of a geometric progression we get

$$\begin{gathered} S(D) = \sum_{l=1}^{L} Y_{i(l)} \left[\sum_{j=0}^{2t-1} \left(\alpha^{i(l)} D\right)^j\right] = \\ \sum_{l=1}^{L} Y_{i(l)} \frac{1-\left(\alpha^{i(l)}D\right)^{2t}}{1-\alpha^{i(l)}D} \end{gathered} . \tag{3}$$

The *Error Locator Polynomial*, (ELP), is defined naturally as the common denominator of (3) and given explicitly by

$$C(D) = \prod_{l=1}^{L} \left(1 - \alpha^{i(l)} D\right). \tag{4}$$

The sums of the geometric progressions contribute to the numerator polynomials zero-coefficients for all but 2 coefficients, the coefficients of D^0 and D^{2t}. Multiplying by the ELP the sequence of 0-s may be reduced, but still the coefficients of D^j for $L \le j < 2t$ are all 0. Formally

$$S(D)C(D) = \Omega(D) mod\ D^{2t}, \tag{5}$$

where $\Omega(D)$ given by

$$\Omega(D) = \sum_{l=1}^{L} \frac{Y_{i(l)}}{1-\alpha^{i(l)}D} C(D), \tag{6}$$

is called the *error evaluator polynomial*, and equation (5) is known as the *key equation*.

Note that (6) implies that $deg(\Omega) < deg(C)$.

III. Solving the key equation - theory

For a general polynomial S of degree $2t$-1 a solution of the *key equation* is a pair of polynomials $C(D)$ and $\Omega(D)$ with $c_0 \ne 0$, $deg(\Omega) < t$, $deg(C) \le t$ satisfying (5).

We slightly modify (5) to read

$$S(D)C(D) - \Omega(D) = 0\ mod\ D^{2t}, \tag{7}$$

and we shall solve (7) by an iterative algorithm comprising of $2t$ iterations, where at iteration T the algorithm computes a pair of polynomials $C^{(T)}$, and $\Omega^{(T)}$ with $c_0^{(T)} \ne 0$, $deg\left(\Omega^{(T)}\right) < T/2$, $deg\left(C^{(T)}\right) \le T/2$ such that

$$S(D)C^{(T)}(D) - \Omega^{(\mathrm{T})}(D) = 0\ mod\ D^T. \tag{8}$$

If the algorithm is successful a uniqueness theorem will come to the rescue and state that $C^{(2t)}(D) = C(D)$ is the desired ELP, and $\Omega^{(2t)}(D) = \Omega(D)$ is the error evaluator polynomial.

IV. Backward polynomial division

For a polynomial $G(D)$ in the indeterminate D, let $ged(G)$ denote the lowest index of the non-zero coefficients of G, (mirroring $deg(G)$ which is the highest such index). Using this terminology any polynomial $G(D)$ may be written as

$$G(D) = \sum_{i=ged(G)}^{deg(G)} g_i D^i , \tag{9}$$

where both $g_{ged(G)}$ and $g_{\deg(G)}$ are non-zero elements. For shorthand we shall define $l(G) \overset{\text{def}}{=} g_{ged(G)}$, ($l$ for lower), and $u(G) \overset{\text{def}}{=} g_{deg(G)}$, ($u$ for upper).

For 2 polynomials $G(D)$ and $H(D)$ with $deg(G) \ge deg(H)$ the familiar ordinary long division defines a pair of quotient and remainder polynomials $(Q(D), R(D))$ satisfying

$$\begin{gathered} G = QH + R \\ deg(R) < deg(H) \end{gathered} . \tag{10}$$

The process of computing (Q, R) is summarized below

```
$R = G,\ Q = 0$
$Loop\ until\ deg(R) < deg(H)$
    $q = \frac{u(R)}{u(H)} D^{[deg(R) - deg(H)]}$          (11)
    $R = R - qH$
    $Q = Q + q$
```

The remainder R is known as $G\ mod\ H$.

We shall now define a mirror process, which we call *backward polynomial division* (BPD). Let G and H be 2 polynomials such that $ged(H) \le ged(G)$. Define a *backward polynomial division* process by the following sequence of polynomials

```
$R = G,\ Q = 0$
$Loop\ until$ condition $is\ satisfied$
    $q = \frac{l(R)}{l(H)} D^{[ged(R) - ged(H)]}$   .      (12)
    $R = R - qH$
    $Q = Q + q$
```

This process is the mirror of the ordinary process used for long polynomial division. The ordinary polynomial division has a natural stop which occurs when $deg(R) < deg(H)$. For BPD the stopping point will be defined per application. The general idea is to associate a function $L(G)$ with the polynomials such that L is monotonic non-decreasing in $ged(G)$, i.e. if $ged(G) > ged(H)$ then $L(G) \ge L(H)$, and the stopping condition for the BPD (12) is when $L(R) > L(G)$.

We denote the *"remainder"* R by a special notation

$$G\ dom\ H \overset{\text{def}}{=} R, \tag{13}$$

which mirrors $G\ mod\ H$ of the ordinary polynomial division. In addition we define a new polynomial Q_2 by

$$Q_2 \overset{\text{def}}{=} \sum_{j=ged(Q)}^{deg(Q)-1} q_j D^j. \tag{14}$$

Note that R, Q mirror their role in ordinary polynomial division. Q_2 is unique for BPD, and its role and significance will be revealed in the sequel. $deg(Q_2) \leq deg(Q) - 1$, and the inequality is strict if $q_{deg(Q)-1} = 0$.

V. Example

Consider the polynomials $S = \sum_{i=1}^{2t} s_i D^{i-1}$, $G = D{\cdot}S$, and $H = 1$. Performing a BPD on the pair G, H without a stopping condition, or with a poorly defined stopping condition, will terminate after $2t$ iterations with the result R=0, $Q = D{\cdot}S$, which is perfectly correct but totally useless.

However, repeating the division process with a properly defined stopping condition is a completely different story.

For any polynomial R which is a polynomial combination of G, H, i.e. $R = PG + QH$, the vector (P, Q) is called a Bezout pair, and denoted $bp(R)$. The degree of a Bezout pair may be defined by a natural extension of the notion of degree of a polynomial to the degree of a vector of polynomials $\boldsymbol{P} = (P_1, P_2, \dots, P_k)$ by

$$deg(\boldsymbol{P}) \overset{\text{def}}{=} max(deg(P_i)). \tag{15}$$

Thus, for $bp(R) = (P, Q)$,

$$\deg(bp(R)) = max(\deg(P), \deg(Q)). \tag{16}$$

NOTE: Bezout pairs are not unique, but if G, H are relatively prime and $\deg(P, Q) < \deg(G, H)$, then it is straight forward to prove that (P, Q) is the unique Bezout pair associated with $R = PG + QH$ and satisfying $\deg(P, Q) < \deg(G, H)$. In this case we say that (P, Q) is a minimal degree Bezout pair.

Back to the BPD algorithm and stopping condition. Perform the algorithm (12) on G, H, R and in parallel on their minimal degree Bezout pairs, and stop when

$$deg\big(bp(R)\big) > deg(bp(G)). \tag{17}$$

In our example, let j_0 denote the first index which $s_i \neq 0$. Applying (12) we have $Q = s_{j_0} D^{j_0}$, $R = \sum_{i=j_0+1}^{2t} s_i D^i$.

$$\begin{gathered} bp(G) = (1,0), \quad bp(H) = (0,1) \\ R = G - s_{j_0} D^{j_0} H, \quad bp(R) = (1, -s_{j_0} D^{j_0}) \end{gathered} . \tag{18}$$

At this point (17) is satisfied and the BPD is complete.

Spoiler: The ELP can be computed by extending the BPD to a *Backward Euclid Algorithm* (BEA) beginning with the polynomials G, H defined above. However, both the original algorithm of [2] and the presentation of the algorithm in [3] may be presented as a BEA applied to the polynomials

$$G = D{\cdot}S, \quad H = 1{+}D{\cdot}S, \tag{19}$$

but the Bezout pairs are computed relative to the polynomials

$$\mathfrak{G} \overset{\text{def}}{=} D{\cdot}S, \qquad \mathfrak{H} \overset{\text{def}}{=} 1, \tag{20}$$

so the initial Bezout pairs are

$$bp(G) = (1,0), \quad bp(H) = (1,1). \tag{21}$$

VI. Backward Euclid Algorithm

Given 2 polynomials G and H with $ged(H) \leq ged(G)$) $deg\big(bp(H)\big) \leq deg\big(bp(G)\big)$, where the Bezout pairs are computed relative to 2 forefather polynomials $\mathfrak{G}, \mathfrak{H}$ define a *backward Euclid algorithm* (BEA) according to the following:

$$\begin{gathered} G^{(0)} = G, \ H^{(0)} = H \\ G^{(i)} = G^{(i-1)} \ dom \ H^{(i-1)} \\ H^{(i)} = G^{(i-1)} - Q_2^{(i-1)} H^{(i-1)} \end{gathered} . \tag{22}$$

This is a mirror process with a slight variation on the ordinary Euclid algorithm which operates according to

$$\begin{gathered} G^{(0)} = G, \qquad H^{(0)} = H \\ G^{(i)} = H^{(i-1)}, \quad H^{(i)} = G^{(i-1)} \ mod \ H^{(i-1)} \end{gathered} . \tag{23}$$

The following table summarizes the BPD and BEA concepts and terminology in comparison with the ordinary polynomial division and Euclid algorithm.

TABLE I

Ordinary Division and Euclid	**Backward Division and Euclid**
Stopping Condition $deg(H) > deg(R)$	Stopping Condition $deg(bp(G)) < deg(bp(R))$
Euclid Remainder $R = G \ mod \ H$	Euclid Remainder $R = G \ dom \ H$
Euclid Update $G \leftarrow H \leftarrow R$	Euclid Update $R \rightarrow G,\ G\text{-}Q_2 H \rightarrow H$
$deg(G) \geq deg(H) > deg(R)$	$ged(R) > ged(G) \geq ged(H)$

VII. Uncovering the BM disguise

The (ordinary) eE algorithm for computing the ELP of an RS code applies the Euclid algorithm to the pair

$$G = D^{2t}, \ H = S, \tag{24}$$

and it terminates at the first iteration i for which $deg(G^{(i)} \ mod \ H^{(i)}) < t$. At this point denote

$$\begin{gathered} \Omega = G^{(i)} \ mod \ H^{(i)} \\ (A, C) = bp(\Omega) \end{gathered} , \tag{25}$$

C is the ELP and Ω is the evaluator polynomial.

A similar computation of minimal degree Bezout coefficients may be done for the BEA such that

$$A^{(i)} G + B^{(i)} H = G^{(i)} \ dom \ H^{(i)} = G^{(i+1)}. \tag{26}$$

Our main proposition is:

Main Proposition: The BM algorithm of [2], [3] is actually a BEA applied to $G = D{\cdot}S$, $H = 1 + D{\cdot}S$ with Bezout pairs computed relative to $\mathfrak{G} = G = D{\cdot}S$, $\mathfrak{H} = 1$.

The BEA is applied until we reach an i with $ged\big(G^{(i)}\big) > 2t$. At this stage if $ged\big(H^{(i)}\big) \leq 2t$, then the desired ELP and evaluator polynomial are derived from $bp\big(G^{(i)}\big)$. If $ged\big(H^{(i)}\big) > 2t$, then the desired ELP and evaluator polynomial are derived from $bp\big(H^{(i)}\big)$. Specifically, if the Bezout pair of the relevant polynomial is (C, W) then C is the ELP, and $\Omega \overset{\text{def}}{=} -W/D$ is the evaluator polynomial.

Sketch of Proof: A Bezout pair for R at a typical step of the algorithm, is given by (C, W) such that $R = C\mathfrak{G} + W\mathfrak{H}$. It is straight forward to see that at any iteration $ged(C) = 0$, and after the first iteration also $ged(W) > 0$, so W may be divided by D. If

$$R = C\mathfrak{G} + W\mathfrak{H} = D{\cdot}CS + W = 0 \ mod \ 2t{+}1 \tag{27}$$

then

$$CS + W/D = 0 \ mod \ 2t. \tag{28}$$

If S originated from an RS code with $\leq t$ errors, then the proof will be complete once we prove that (28) viewed as an equation in (C, W) has a unique solution (up to scalar multiplication) with $\deg(C, W) \leq t$, and $ged(C) = 0$, and that the BEA converges to this solution. At this point we rely on [2] for the proofs, and leave to the reader to verify that the BEA actually follows the footsteps of the algorithm as defined in [2]. Note that [2] only updates the first coordinate of the Bezout pair, denoted by the polynomial C, with the aid of the auxiliary polynomial B, (which corresponds to the first coordinate of the Bezout pair of H). The discrepancy parameter of [2] is the lowest coefficient of G to be considered at the specific iteration. In [3] both the polynomials G, H are considered together with the first coordinate of their Bezout pairs. In the next section we will reconstruct all of Massey's propositions in the terminology of this paper. This will create a simple coherent and complete treatment of the ELP subject where the two alternative solutions via ordinary Euclid or backward Euclid have a nice symmetry.

VIII. Reconstructing the BM algorithm

In this section we will reconstruct the original paper of Massey [2] and formulate it according to the language developed in this paper.

Theorem 1: Let (C, Ω) be a polynomial pair with $ged(C) = 0$. Suppose $ged(C \cdot S - \Omega) = N$. Let $(C2, \Omega 2)$ be another polynomial pair with $ged(C2) = 0$ satisfying $ged(C2 \cdot S - \Omega 2) > N$.
Then: $deg(C2, D \cdot \Omega 2) + deg(C, D \cdot \Omega) \geq N + 1$.

Proof: If $deg(C, D \cdot \Omega) \geq N + 1$ the statement is trivial. So assume $deg(C, D \cdot \Omega) \leq N$, and assume $deg(C, D \cdot \Omega) + deg(C2, D \cdot \Omega 2) \leq N$. It is straight forward to see that

$$ged(C2[C \cdot S - \Omega)]) = N\,. \tag{29}$$

Similarly

$$ged(C[C2 \cdot S - \Omega 2]) > N\,, \tag{30}$$

and therefore subtracting (29) from (30) we get

$$ged(C \cdot \Omega 2 - C2 \cdot \Omega) = N\,. \tag{31}$$

But according to the assumptions

$$\begin{gathered} deg(C \cdot \Omega 2 - C2 \cdot \Omega) \leq \\ max\{deg(C \cdot \Omega 2), deg(\, C2 \cdot \Omega)\} < N \end{gathered} \tag{32}$$

which clearly contradicts (31).

Next we redefine $L_N(S)$ of [2] in the terminology of this paper.

Definition: Suppose $S(D) = \Sigma_{i=0}^{\infty} s_i D^i$ is a power series. Define $L_N(S)$ to be the minimal integer L for which there exist a polynomial pair (C, Ω) with $deg(C, D \cdot \Omega) = L$ and $ged(C) = 0$, such that $ged(C \cdot S - \Omega) \geq N$.

At this point we return to the BEA, and prove that the algorithm applied to the polynomials of (19) with Bezout pairs computed relative to (20) will exhaust all the possible values of $L_N(S)$ $N = 1, \ldots, 2t$. If $\deg(ELP) \leq t$ then $L_{2t}(S) \leq t$, and the BEA will find a pair of polynomials (C, Ω) such that $ged(C)$=0, $deg(C, D \cdot \Omega) \leq t$, and $ged(C \cdot S - \Omega) \geq 2t$. The proof will be complete once we prove that under these conditions the pair (C, Ω) is unique (up to scalar multiplication).

At iteration i of the BEA, we have the polynomials $G^{(i)}, H^{(i)}$. Assume by induction that

$$\begin{gathered} ged(H^{(i)}) = deg(bp(G^{(i)})) + deg(bp(H^{(i)})) \\ L_{ged(H^{(i)})-1}(S) = deg(bp(H^{(i)})) \\ L_{ged(H^{(i)})}(S) = L_{ged(G^{(i)})-1}(S) = deg(bp(G^{(i)})) \end{gathered} \quad , \tag{33}$$

and prove that this will be true for iteration i+1 as well.
First lets look at $H^{(i+1)} = G^{(i)} - Q_2 H^{(i)}$. By definition

$$ged(H^{(i+1)}) \geq ged(G^{(i)}), \tag{34}$$

and

$$deg(bp(H^{(i+1)})) \leq deg(bp(G^{(i)})). \tag{35}$$

The induction assumption in (33) that $L_{ged(G^{(i)})-1}(S) = deg(bp(G^{(i)}))$ implies that (35) is actually an equality.
The polynomial $G^{(i+1)}$ is expressed in terms of $H^{(i)}, H^{(i+1)}$ as

$$G^{(i+1)} = H^{(i+1)} + xD^{\Delta}H^{(i)}, \tag{36}$$

where Δ is set such that

$$ged(H^{(i+1)}) = ged(D^{\Delta}H^{(i)}) = \Delta + ged(H^{(i)}), \tag{37}$$

and x is set such that

$$ged(G^{(i+1)}) > ged(H^{(i+1)}). \tag{38}$$

By definition $deg(bp(G^{(i+1)})) > deg(bp(H^{(i+1)}))$, and therefore

$$\begin{gathered} deg(bp(G^{(i+1)})) = deg\left(bp(D^{\Delta}H^{(i)})\right) = \\ \Delta + deg\left(bp(H^{(i)})\right) \end{gathered} \quad . \tag{39}$$

Regarding $ged(H^{(i+1)})$, according to the induction hypothesis

$$\begin{gathered} ged(H^{(i+1)}) = \Delta + ged(H^{(i)}) = \\ \left(\Delta + deg(bp(H^{(i)}))\right) + deg(bp(G^{(i)})) = \\ deg(bp(G^{(i+1)})) + deg(bp(H^{(i+1)})) > \\ 2 \cdot deg(bp(H^{(i+1)})) \end{gathered} \quad , \tag{40}$$

so if the Bezout pair associated with $H^{(i+1)}$ is $bp(H^{(i+1)}) = (C_H, W_H)$, and setting $L = deg(bp(H^{(i+1)}))$, $\Omega_H = W_H / D$, we see that

$$ged(C_H \cdot S - \Omega_H) = ged(H^{(i+1)}) - 1 \geq 2L. \tag{41}$$

Therefore the Bezout pair associated with $G^{(i+1)}$, $bp(G^{(i+1)}) = (C_G, W_G)$, satisfies

$$ged(C_G \cdot S - W_G/D) \geq ged(H^{(i+1)}), \tag{42}$$

thus according to Theorem 1, $bp(G^{(i+1)})$ must satisfy

$$L + deg\left(bp(G^{(i+1)})\right) \geq ged(H^{(i+1)}). \tag{43}$$

But according to (40) equality is achieved in (43), and therefore

$$L_{ged(H^{(i+1)})}(S) = deg\left(bp(G^{(i+1)})\right), \tag{44}$$

and this proves that if (33) is true for iteration i, then it is true for iteration $i + 1$, so the proof of the induction is complete.

As noted above we still have to prove that the pair (C, Ω) which is found by the BEA is unique. Uniqueness is a consequence of the following corollary to Theorem 1.

Corollary: Suppose (C, Ω), and $(C2, \Omega 2)$ are two polynomial pairs satisfying

$$ged(C \cdot S - \Omega) \geq 2L,\ ged(C2 \cdot S - \Omega 2) \geq 2L, \tag{45}$$

with $c_0 = c2_0 = 1$, and $deg(C, D \cdot \Omega, C2, D \cdot \Omega 2) \leq L$.
Then $(C, \Omega) = (C2, \Omega 2)$.

Proof: Without loss of generality assume

$$2L \leq ged(CS - \Omega) \leq ged(C2S - \Omega 2)\,. \tag{46}$$

If $ged(CS - \Omega) < ged(C2S - \Omega2)$, then according to theorem 1 we must have

$$deg(C, D \cdot \Omega) + deg(C2, D \cdot \Omega2) > 2L, \tag{47}$$

which contradicts the assumption that $deg(C, D{\cdot}\Omega, C2, D{\cdot}\Omega2) \leq L$.
So assume $ged(CS - \Omega) = ged(C2S - \Omega2)$. In this case we may find a polynomial $\bar{S} \overset{\text{def}}{=} S + D^{2L}T$, (for some polynomial T), such that $ged(C{\cdot}\bar{S} - \Omega) = N$ and $ged(C2{\cdot}\bar{S} - \Omega2) > N$ for some $N > 2L$, which again contradicts theorem 1.

IX. Low Power and Low Latency BCH Decoding

Consider a narrow sense primitive BCH code with a design distance of $2t$+1. The BM algorithm of [2] may be modified in accordance with [3] to a low latency algorithm according to the following flow:

```
Initialize:  $C(D) = 1$;  $B(D) = D$;
     $L = 0; \beta = 1;\ T = 0;$
     $S(D) = \sum_{i=1}^{2t} s_i D^{i-1}$;  $G(D) = S(D)$;
                    $H(D) = 1 + D \cdot S(D)$;
while  T < t,
    $\delta = g_{2T}$;
    $tmpC\ =\ C(D)$;   $C(D) = \beta C(D) + \delta B(D)$;
    $tmpG = G(D)$;    $G(D) = (\beta G(D) + \delta H(D))/D^2$;
    if  $\delta == 0 \mid L > T$,
       $B(D) = D^2 B(D)$;
    else,
       $B(D)\ =\ D^2 tmpC$;
       $H(D)\ =\ tmpG$;  $L = 2T + 1 - L; \beta = \delta$;
    end (if)
    $T = T + 1$;
end (while)
```

As noted in section VII, G and $bp(G)$ should have been initialized as G=$D{\cdot}S$, $bp(G)$=(1,0), and H, $bp(H)$ should have been initialized as H=1+$D{\cdot}S$, $bp(H) = (1,1)$. The first step of a BPD algorithm applied to G, H is to multiply H by D, but then $bp(H)$ must also be multiplied by D, so after this first step H is transformed to $D + D^2{\cdot}S$, and $bp(H)$ is transformed to (D, D). After completion of this step both G and H may be shifted downward (divided by D), which has no effect on the Bezout pairs in the following steps of the algorithm. The above algorithm flow initialized at this point. It considers only the first coordinate of the Bezout pairs, thus $C(D) = 1$ is the first coordinate of the Bezout pair of G, and $B(D) = D$ is the first coordinate of the Bezout pair of H.

After each of the following steps of the algorithm both G and H are shifted downward by 2 positions, (i.e. divided by D^2), which has advantages from a HW perspective but again does not affect the computation of the Bezout pairs.

A careful look at the algorithm flow above shows that only the even powers of G and H need to be considered. Therefore the algorithm may initialize G and H as

$$\begin{aligned} G(D) &= S_{even}(D) \\ H(D) &= 1 + D{\cdot}S_{odd}(D) \end{aligned} \tag{48}$$

Originally the polynomials $C(D), B(D)$ require t memory units each, and $G(D), H(D)$ require $2t$ memory units each for a total of $6t$ memory units. Considering only the even powers reduces the number of required memory units for G and H to t units for each. Thus the total amount of memory required is reduced from $6t$ to $4t$. The area and power of a HW implementation of the BM algorithm is reduced accordingly.
Block diagrams for updating the parameters of the BM algorithm are given below:

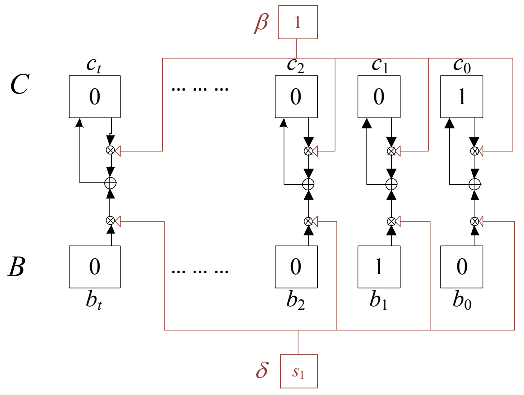

Figure 1: Updating C

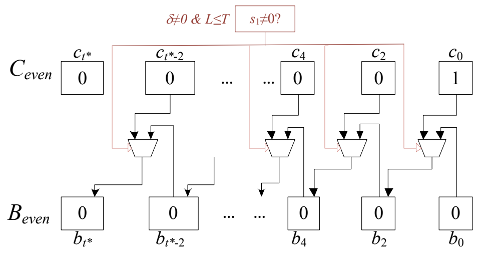

Figure 2: Updating B_{even}

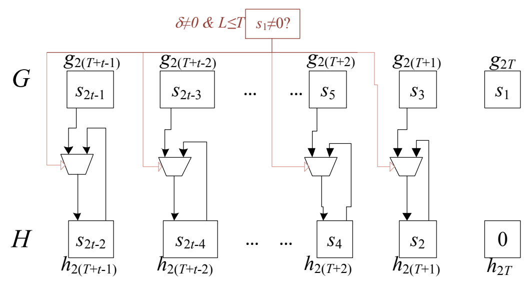

Figure 3: Updating H (only H_{even})

References

[1] E.R. Berlekamp, *Algebraic Coding Theory*, McGraw-Hill, New York, chapter 7 (1968).

[2] J.L. Massey, "Shift register synthesis and BCH decoding", IEEE Trans. Inform. Theory, vol. IT-15, pp. 122–127 (1969).

[3] D. V. Sarwate and N. R. Shanbhag, "High-speed architectures for Reed-Solomon decoders," in *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 9, no. 5, pp. 641-655, Oct. 2001.

[4] F.J. MacWilliams, N.J.A. Sloane, *The Theory of Error-Correcting Codes*, Volume 16, North-Holland Mathematical Library.

[5] Wikipedia, https://en.wikipedia.org/wiki/Berlekamp%E2%80%93 Massey_algorithm

[6] K. Imamura and W. Yoshida, "A simple derivation of the Berlekamp-Massey algorithm and some applications", in *IEEE Transactions on Information Theory*, vol. 33, no. 1, pp. 146-150, Jan 1987.

[7] M. Bras-Amoros M. E. O'Sullivan, "The Berlekamp-Massey Algorithm and the Euclidean Algorithm: a Closer Link", [Online.] Available: https://arxiv.org/abs/0908.2198

[8] J.L. Dornstetter. "On the equivalence between Berlekamp's and Euclid's Algorithms", *IEEE Trans. Inform. Theory*, 33(3):428–431, 1987.

[9] A. E. Heydtmann and J. M. Jensen. "On the equivalence of the Berlekamp-Massey and the Euclidean algorithms for decoding", I*EEE Trans. Inform. Theory*, 46(7):2614–2624, 2000.

[10] T. D. Mateer, "On the equivalence of the Berlekamp-Massey and the euclidean algorithms for algebraic decoding", *12th Canadian Workshop on Information Theory*, Kelowna, BC, 2011, pp. 139-142.

[11] Y. Sugiyama, M. Kasahara, S. Hirasawa, and T. Namekawa. "A method for solving key equation for decoding Goppa codes", *Information and Control*, 27:87–99, 1975.