High-Throughput Low-Complexity Four-parallel Reed-Solomon Decoder Architecture for High-Rate WPAN Systems

Chang-Seok CHO†, Hyo-Jin AHN†, Nonmembers, and Hanho LEE†(a), Member

SUMMARY This paper presents a high-throughput low-complexity four-parallel Reed-Solomon (RS) decoder for high-rate WPAN systems. Four-parallel processing is used to achieve 12-Gbps data throughput and low hardware complexity. Also, the proposed pipelined folded Degree-Computationless Modified Euclidean (fDCME) algorithm is used to implement the key equation solver (KES) block, which provides low hardware complexity for the RS decoder. The proposed four-parallel RS decoder is implemented 90-nm CMOS technology optimized for a 1.2 V supply voltage. The implementation result shows that the proposed RS decoder can be operated at a clock frequency of 400 MHz and has a data throughput 12.8-Gbps. The proposed four-parallel RS decoder architecture has high data processing rate and low hardware complexity. Therefore it can be applied in the FEC devices for next-generation high-rate WPAN systems with data rate of 10-Gbps and beyond.

key words: forward error correction (FEC), Reed-Solomon (RS), decoder, mmWAVE, WPAN

1. Introduction

The emergence of a multitude of “bandwidth hungry” multimedia applications has definitely exacerbated the need for multi-gigabit wireless solutions, which are beyond reach of conventional WLAN technology (802.11a, b and g). Uncompressed high-definition video distribution and massive data synchronization are driving data-throughput requirements well beyond gigabits/s (Gbps), and already demanding up to 10-Gbps with introduction of, for example, the HDMI 1.3 video standard [1]. Such a strong commercial interest in using the 57–66 GHz band known as the millimeter wave band for indoor wireless communications is evidenced by the recent industrial and standard development efforts in several international standard groups including ECMA TC-387, IEEE 802.15.3e and the 802.11 VHT60 [2].

These task groups are developing a millimeter-wave (mmWave) based alternative physical layer (PHY) for high-rate Wireless Personal Area Network (WPAN) standard [3], [4]. This mmWave WPAN system will allow high coexistence with all other microwave systems in the 802.15 family of WPANs. In addition, the mmWave WPAN will support high data rate applications such as high speed internet access and streaming content download (video on demand, home theater, 3D TV etc.). Very high data rates in excess of 10-Gbps beyond will be provided for simultaneous time-dependent applications such as real time multiple HDTV video stream and wireless data bus for cable replacement.

These reasons, such a demand for ever higher data rates, makes it necessary to devise very high-speed Forward Error Correction (FEC) architectures. Reed-Solomon (RS) codes have been adopted WPAN systems as a FEC scheme [3], [4], and also several multi-giga bit RS decoders have been reported. To get a high throughput, parallel processing method can be a best solution for the hardware design. The one-shot Reed-Solomon encoder/decoder scheme [5], [6], which is based on parallel combinational circuit, can be a representative example for high throughput RS decoder.

In this paper, we present the four-parallel RS (240,224) encoder/decoder architecture for mmWAVE WPAN systems especially ECMA standard. Four-parallel processing is used to achieve 12-Gbps data throughput rates. Also, folded Degree-Computationless Modified Euclidean (fDCME) architecture is applied for key equation solver (KES) block to reduce a hardware complexity.

This paper is organized as follows. Section 2 shows the proposed four-parallel RS encoder architecture. In Sect. 3, we will describe the key ideas applied to four-parallel RS decoder design, especially those for achieving high throughput and reduced hardware complexity. Four-parallel syndrome computation, Chien search & error correction block and pipelined fDCME architecture are proposed. Section 4 gives implementation results and performance comparison. Finally, conclusions are provided in Sect. 5.

2. Four-Parallel Reed-Solomon Encoder

The systematic RS encoding produces codeword polynomial in Eq. (1), which is comprised of message symbols followed by parity symbols. The message polynomial \( M(x) \) is multiplied by \( x^{k-\ell} \) after then added the parity polynomial \( P(x) \). If generator polynomial \( G(x) \) was given as Eq. (2), the following parity polynomial \( P(x) \) can be written as Eq. (3). To apply four-parallel structure, the Eq. (3) should be reformulated. The \( M(x) \) consists of 224 symbols, which are multiple of four. As a result the four-parallel based \( P(x) \) can be rewritten to Eq. (4) and we can derive the following partial generator polynomial as shown in Eq. (5).

The proposed four-parallel RS encoder is shown in Fig. 1. Four-parallel message symbols are inputted from ports \([M3,M2,M1,M0]\) during 56 clocks and multiplied by each partial generator polynomials \( g_0(x) \sim g_3(x) \) in Eq. (5). Finally, parity symbols are generated through Linear Feed-

---

Manuscript received September 4, 2010.
Manuscript revised January 12, 2011.
†The authors are with School of Information and Communication Engineering, Inha University, Incheon, 402-751, Republic of Korea.
a) E-mail: hhlee@inha.ac.kr
DOI: 10.1587/transcom.E94.B.1332

Copyright © 2011 The Institute of Electronics, Information and Communication Engineers
back Shift Register (LFSR).

\[
U(x) = x^{m-k}M(x) + P(x) \quad (1)
\]

\[
G(x) = (x - a^0)(x - a^1) \cdots (x - a^{14})(x - a^{15}) \quad (2)
\]

\[
P(x) = x^{n-k}M(x) mod G(x) \quad (3)
\]

\[
U(x) = [((m_{223} \cdot x^{19} + m_{222} \cdot x^{18} + m_{221} \cdot x^{17} + m_{220} 
\cdot x^{16}) mod G(x)] \cdot x^4 + [((m_{219} \cdot x^{19} + m_{218} \cdot x^{18} + m_{217} \cdot x^{17} + m_{216} \cdot x^{16}) mod G(x)] 
\cdot x^4 + \cdots + (m_1 \cdot x^{19} + m_2 \cdot x^{18} + m_1 \cdot x^{17} + m_0 \cdot x^{16}) mod G(x) \quad (4)
\]

\[
g_0(x) = x^{16} mod G(x), \quad g_1(x) = x^{17} mod G(x) \quad (5)
\]

3. Four-Parallel Reed-Solomon Decoder

Generally, the RS decoder consists of following three blocks, which are syndrome computation block, KES block, Chien search and error correction block. This iterative process

The proposed four-parallel RS decoder architecture is shown in Fig. 2. The proposed architecture includes four-parallel syndrome computation block, fDCME block, and four-parallel Chien search and error correction block. This section gives full explanation about sub-blocks.

3.1 Four Parallel Syndrome Computation Block

The syndrome computation block calculates all syndromes \( S_i \) (0 < \( i \) < 15) by putting the roots of generator polynomial \( G(x) \) into the received codeword polynomial \( R(x) \) in Eq. (6). As shown in Fig. 3, proposed four-parallel syndrome computation block is implemented by following Eq. (7).

\[
R(x) = r_{239}x^{239} + r_{238}x^{238} + \cdots + r_1x + r_0 \quad (6)
\]

\[
S_i = R(a^i) = ((r_{239} \cdot a^i)^3 + r_{238} \cdot a^i)^3 + r_{237} \cdot a^i)^3 + r_{236} \cdot a^i)^3 + \cdots + (a^i)^3 + r_{235} \cdot (a^i)^2 + r_{234} \cdot (a^i)^2 + r_{233} \cdot (a^i) + r_{232} + \cdots + (a^i)^3 + r_{231} \cdot (a^i)^2 + r_{230} \cdot (a^i) + r_{229} + \cdots + (a^i)^2 + r_{228} \cdot (a^i) + r_{227} + \cdots + (a^i) + r_{226}) \quad (7)
\]

The received codeword consists of 240 symbols which are multiple of 4, so that the proposed syndrome computation block should calculate syndromes during 60 clock cycles. At the first clock, the received codeword \((r_{239}, r_{238}, r_{237}, r_{236})\) are inputted by parallel, and then partial syndromes \( r_{239}(a^i)^3 + r_{238}(a^i)^2 + r_{237}(a^i) + r_{236} \) are computed following stored in the flip-flop (1). At the next clock cycle, the flip-flop (1) is multiplied by \((a^i)^4\) and then added with \( r_{235}(a^i)^3 + r_{234}(a^i)^2 + r_{233}(a^i) + r_{232} \) are computed following stored in the flip-flop (1). At the next clock cycle, the flip-flop (2) is multiplied by \((a^i)^4\) and then added with \( r_{235}(a^i)^3 + r_{234}(a^i)^2 + r_{233}(a^i) + r_{232} \) are computed following stored in the flip-flop (2). Finally, the syndromes \( S_0, S_1, \ldots, S_{15} \) are outputted serially to the KES block, and new syndromes can be computed in the syndrome cells.

3.2 Key Equation Solver Block

The KES block is used to obtain the error locator polynomial \( \sigma(x) \) and the error value polynomial \( \omega(x) \) by solving the key equation \( \omega(x) = S(x)\sigma(x) \mod m^2 \). The KES block is the most critical part in the design of RS decoders. The KES architectures based on the modified Euclidean (ME) algorithm [7]–[9], [17], [18] or Berlekamp-Massey (BM) algorithm [11], [12] are regular structure, but the hardware cost
is very high, because their architectures are required both systolic-array structure and degree computation units. So, the pDCME algorithm was suggested alternatively in [15], but the pDCME architecture still has high hardware complexity. While pDCME architecture can be implemented by 2t processing element (PE), the proposed fDCME algorithm, which is employed folding technique, consists of only 2 PEs with shift-registers.

The proposed fDCME algorithm is described by the pseudo-code shown in below. Two array of PE performs the DCME algorithm continuously and then the error locator polynomial $\sigma(x)$ and error value polynomial $\omega(x)$ can be computed. Until when the index ‘stage’ is reached at $t$ times, $a_{t-1}$ and $b_{t-1}$ are the leading coefficients of polynomial $F_{t-1}(x)$ and $G_{t-1}(x)$ respectively. Either Step2 (swap operation) or Step3 (delaying previous coefficients) is executed until when the index ‘loop’ of Step1 reaches 2, repeatedly. The Step2 is controlled by stop-signal (stop), swap-signal (sw) and Shift-signal (sht).

\begin{equation}
G_0(x) = x^{2^t}, F_0(x) = \sum_{i=0}^{2t-1} S_i x^i \begin{cases} S_i \neq 0, & 0 \leq i \leq 2t-1 \end{cases} \tag{8}
\end{equation}

\begin{equation}
G_0(x) = x^{2^t}, F_0(x) = \sum_{i=0}^{2t-1} S_i x^i \begin{cases} S_i = 0, & 0 \leq i \leq 2t-1, k \geq 0 \end{cases} \tag{9}
\end{equation}

\begin{equation}
G_0(x) = x^{2^t}, F_0(x) = \sum_{i=0}^{2t-1} S_i x^i \begin{cases} S_i \neq 0, & 2t-1-m \leq i \leq 2t-1 \end{cases} \tag{10}
\end{equation}

Inputs of PE (1) and (2) have several patterns, which correspond to Eqs. (8)~(10). These patterns are used to generate two control signals which are ‘sw’ and ‘sht’. The ‘sw’ signal determines whether two polynomials pair $F_{t-1}(x)$, $G_{t-1}(x)$ and $H_{t-1}(x)$, $I_{t-1}(x)$ should be swapped or not. The ‘sht’ signal determines either polynomial arithmetic operation or shift operation. In Eq. (8), $G_0(x)$ is $x^{2^t}$ and $F_0(x)$ is $S(x)$ multiplied $x$. And the coefficient $S_{2t-1}$ is non zero. Since the degree of two polynomials are same as 16, the PE (1) executes the arithmetic operation. After the operation of PE (1), $G_1(x)$, $F_1(x)$ have same degree as 15. In Eq. (9) the coefficient $S_{2t-1}$ is zero. That means the degree of $F_0$ input is 15. So PE (1) executes only delay operation for $G_0$’s output to make the same degree of two inputs. And then PE (2) executes the arithmetic operation since degree of two inputs is same. In Eq. (10), the coefficient $S_{2t-1}$ is non zero but $S_{2t-2}$ is zero. In case of this, the PE (1) has same operation as Eq. (10). But degree of $F_0$’s output is 14. Thus, the PE (2) executes only delay operation for output of $G_1$. Since the degree of $F_1(x)$ is less than the degree of $G_1(x)$, two inputs were swapped before the PE (1) operation. When the index ‘stage’ is reach at $t$ times, the fDCME algorithm stops. The output $F_{16}(x)$ of PE (2) becomes the error value polynomial $\omega(x)$ and the output $H_{16}(x)$ becomes the error locator polynomial $\sigma(x)$.

Figure 4 shows a block diagram of proposed fDCME architecture, which consists of two PEs and shift-registers connected by means of a recursive loop. $F_{t-1}(x)$, $G_{t-1}(x)$, $H_{t-1}(x)$ and $I_{t-1}(x)$ generates the updated coefficients of each polynomial serially. The output of PE (2) is fed back into the PE (1) in descending order. The PE (1) and (2) consist of a polynomial arithmetic structure, control-signal generate block and stop-signal generate block. One PE consists of four Galois-field ($GF$) multipliers, two $GF$ adders and ten multiplexers.

The PE unit has three pipelining stages to provide significant improvement for the clock frequency. The twelve stage shift-registers are used to store the output of PE (2) at each recursive iteration step. Therefore, the fDCME block has eighteen pipelining stages. The PE (1) and (2) use pipelined fully-parallel $GF$ multiplier to reduce the critical path delay and to provide significant gains for the clock frequency. Therefore, the critical path delay of PE is $T_{\text{inst}} + T_{\text{and}2} + 3T_{\text{mux2}} + T_{g}$, where $T_{\text{inst}}, T_{\text{and}2},$ and $T_{\text{mux2}}$ are delays of the inverter, 2-input AND gate, and 2×1 multiplexer.

### fDCME Algorithm:

- **Input**: $xS(x), x^{2^n}$
- **Initialization**:
  - $F_0(x) = xS(x)$, $G_0(x) = x^{2^n}$, $H_0(x) = x$, $I_0(x) = 0$;
  - Index ‘i’ is initialized to 0;
- **Start algorithm**:
  - for stage=1 step 1 until t
    - **Step1**: for loop=1 step 1 until 2
      - begin
        - $i = i+1$
      - end
    - **Step2**: if (stop=1) begin
      - **Step2.1**: if (sw=1)
        - begin
          - swap($F_{i-1}(x)$, $G_{i-1}(x)$);
          - swap($H_{i-1}(x)$, $I_{i-1}(x)$);
          - swap($a_{i-1}$, $b_{i-1}$);
        - end
      - **Step2.2**: else if (sht=1)
        - begin
          - $a_{i-1} = 0$; $b_{i-1} = 1$;
        - end
    - **Step3**: else begin
      - **Step3.1**: $F_i(x) = F_{i-1}(x) - a_{i-1}G_{i-1}(x)$;
      - **Step3.2**: $H_i(x) = H_{i-1}(x) - a_{i-1}I_{i-1}(x)$;
      - $I_i(x) = I_{i-1}(x)$;
    - end
  - end
- **Output**: $\sigma(x)$, $\omega(x)$;
Fig. 4  Block diagram of folded degree-computationless modified Euclidean (fDCME) architecture.

Fig. 5  Block diagram of Chien search and error correction block, (a) Chien search block, and (b) Chien search cell.
3.3 Four-Parallel Chien Search and Error Correction Block

After the KES block operation, the error locator polynomial \( \sigma(x) \) and the error value polynomial \( \omega(x) \) are obtained. Let \( X_l = e_{ml} \) and \( Y_l = e_{ml} \), the Eq. (11) can be transformed to the Eq. (12), where \( X_l \) and \( Y_l \) are the possible error location and the possible error value, respectively. Chien search algorithm can be implemented using the Eq. (13). The roots of \( \sigma(x) \) are the inversion of error location. In case of RS(240,224) code, \( \sigma(\alpha^{16}) = 0 \) means that \( r_{239} \) was corrupted by an error. At first, \( \alpha^{16} \) is putted into \( \sigma(x) \) because the first symbol of received codeword is \( r_{239} \) in the RS(240,224) codes.

The error value polynomial can be derived as the Eq. (14). Finally the error value can be computed using the Eq. (15), where \( \sigma'(x) \) is the derivative of \( \sigma(x) \). Rewriting \( \sigma(x) \) as the sum of the even terms \( \sigma_{even}(x) \) and the odd terms \( \sigma_{odd}(x) \), we have \( \sigma_{odd}(x) = x \cdot \sigma'(x) \). Therefore, the Chien search and error correction block is implemented as shown in Fig. 5.

\[
S_i = r(a^i) = e(a^i) = \sum_{l=1}^{e} e_{ml} a^{ml} \quad (11)
\]
\[
S(x) = \sum_{i=0}^{15} S_i x^i = \sum_{i=0}^{v} \sum_{l=1}^{e} Y_l X_l x^i \quad (12)
\]
\[
\sigma(x) = (1 - xX_1)(1 - xX_2) \cdots (1 - xX_v) \quad (13)
\]
\[
\omega(x) = S(x) \cdot \sigma(x) \mod x^8 \quad (t = 8)
\]
\[
Y_l = \omega(X_l) / ((-X_l^{-1}) \cdot \sigma'(X_l^{-1})) \quad (15)
\]

The dividing operation is implemented by 256 × 8 ROM in which the inverse of field elements are stored. As shown in Fig. 5(b), serial Chien search cell was expanded into four-parallel Chien search cell, because the following Chien search and Forney algorithm block should calculate four locations of error at each clock cycle. Because the RS(240,224) code is shortened version of the RS(255,239), first 16 symbols don’t have to be computed. Thus, at the first clock cycle, \( \sigma(\alpha^{16}), \sigma(\alpha^{17}), \sigma(\alpha^{18}), \sigma(\alpha^{19}) \) are calculated, and at the last clock cycle, \( \sigma(\alpha^{252}), \sigma(\alpha^{253}), \sigma(\alpha^{254}), \)

Fig. 6 Timing chart of (a) four-parallel RS encoder, and (b) four-parallel RS decoder.
Table 1  Comparison results of various RS decoder architectures.

<table>
<thead>
<tr>
<th>Design</th>
<th>Tech. (μm)</th>
<th>Clock (MHz)</th>
<th>Latency (clocks)</th>
<th>KES block (% of Gates)</th>
<th>Total Area (% of Gates)</th>
<th>Throughput (Gbps)</th>
<th>TSNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>Proposed fDCME</td>
<td>0.09</td>
<td>400</td>
<td>209</td>
<td>7,700</td>
<td>23,920</td>
<td>12.8</td>
<td>370.46</td>
</tr>
<tr>
<td>JIT-FMEA[13]</td>
<td>0.18</td>
<td>400</td>
<td>512</td>
<td>-</td>
<td>20,614</td>
<td>3.2</td>
<td>214.93</td>
</tr>
<tr>
<td>pDCME[16]</td>
<td>0.13</td>
<td>660</td>
<td>355</td>
<td>29,000</td>
<td>53,200</td>
<td>5.3</td>
<td>99.62</td>
</tr>
<tr>
<td>DCME [14]</td>
<td>0.25</td>
<td>-</td>
<td>288</td>
<td>18,000</td>
<td>42,123</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Pipelined ME[8]</td>
<td>0.16</td>
<td>83</td>
<td>120</td>
<td>80,000</td>
<td>166,000</td>
<td>2.5</td>
<td>18.53</td>
</tr>
</tbody>
</table>

σ(α255) are calculated, consecutively.

4. Timing Chart and Performance Comparison

4.1 Timing Chart

The timing charts of proposed RS encoder and decoder are shown in Figs. 6(a) and (b), respectively. The proposed RS encoder has only one clock latency, and generates an encoded codeword (ECWD_A~ECWD_D) continuously. The encoder start signal (RSEST) is needed during 1 clock cycle, after then effective codeword symbols are entered (UCWD_A~UCWD_D). As shown in Fig. 6(b), when the start signal of RS decoder (RSDST) is inputted, the decoder accepts a received codeword (RCWD_A~RCWD_D) at the same time.

Proposed RS decoder has 209 clock latencies. When the proposed RS decoder outputs the corrected codeword (CCWD_A~CCWD_D), error count signal (ERRCNT) for the first codeword is outputted at 266th clock if there are errors, otherwise (FAIL) signal is outputted.

4.2 Performance Comparison

The proposed four-parallel RS encoder/decoder architecture was modeled in Verilog HDL and simulated to verify its functionality. After complete verification of the design functionality, it was then synthesized using appropriate time and area constraints. Both simulation and synthesis steps were carried out using SYNOPSIS synthesis tool and 90nm CMOS technology optimized for a 1.2 V supply voltage. The total number of gates for proposed four-parallel RS decoder is 23,920 gates from the synthesized results including memory block. From the post-layout simulation, the proposed four-parallel RS decoder architecture can operate at a clock frequency of 400 MHz and has a data processing rate of 12.8-Gbps.

Table 1 shows the comparison results of various RS decoder architectures. In case of KES block, proposed fDCME architecture provides much lower hardware complexity than other KES architectures based on ME algorithm. For the purpose of comparison, we used Technology-Scaled Normalized Throughput (TSNT) in [13]. The TSNT is the silicon area normalized to a 0.13 μm technology, as shown in below. We can see that the throughput rate and the TSNT index of our design is the highest among all other architectures.

\[
TSNT = \frac{\text{Throughput Rate}}{\#\text{of Total Gates}} \times \frac{\text{Tech.}}{0.13 \mu m}
\]

The implementation result shows that the proposed four-parallel RS decoder architecture has much higher data processing rate and low hardware complexity compared with the conventional ME algorithm based RS decoder architectures.

5. Conclusions

This paper presented the design and implementation of four-parallel RS encoder/decoder for high-rate WPAN systems. Four-parallel processing is used to achieve 12-Gbps high data throughput. A high-speed low-complexity fDCME block is applied in the KES block. Four-way parallelizing for syndrome computation and Chien search blocks allow the inputs to be received at very high data rates and the outputs to be delivered at correspondingly high rates with a minimum delay. As a result, the proposed four-parallel RS decoder architecture has a much higher data processing rate and low hardware complexity compared with the conventional RS decoder architectures. The proposed RS decoder can be applied in the FEC devices for next-generation high-rate WPAN systems.

Acknowledgments

This work was supported by Inha University.

References


Chang-Seok Choi received the B.S. degree in information & communication engineering from Hanshin University in 2005 and M.S. degree in information & communication engineering from Inha University, Incheon, Korea, in 2007, respectively. He is currently working toward the Ph.D. degree in Inha University. His research interests VLSI design and implementation for communication systems, especially forward error correction architectures.

Hyo-Jin Ahn received the B.S. degree in IT electronic engineering from Daejeon University in 2005 and M.S. degree in information & communication engineering from Inha University, Incheon, Korea, in 2010. His research interests VLSI design and implementation for communication systems, especially forward error correction architectures.

Hanho Lee received the Ph.D. and M.S. degrees, both in Electrical & Computer Engineering, from the University of Minnesota, Minneapolis, in 2000 and 1996 respectively. In 1999, he was a Member of Technical Staff-1 at Lucent Technologies, Bell Labs, Holmdel, NJ. From April 2000 to August 2002, he was a Member of Technical Staff at the Lucent Technologies (Bell Labs Innovations), Allentown, where he was responsible for the development of VLSI architectures and implementation of high-performance DSP multiprocessor SoC for wireless infrastructure systems. From August 2002 to August 2004, he was an assistant professor at the Department of Electrical & Computer Engineering, University of Connecticut, Storrs. Since August 2004, he has been with the School of Information and Communication Engineering, Inha University, Incheon, Korea, where he is presently a Professor. His research interests include design of VLSI circuits and systems for communications, System-on-a-Chip (SoC) design, reconfigurable architecture, and forward error correction coding.

Chang-Seok Choi

Hyo-Jin Ahn

Hanho Lee