Simplified Algorithm and Hardware Implementation for the (24, 12, 8) Extended Golay Soft Decoder Up to 4 Errors

Dongfu Xie
College of Mechanical and Electrical Engineering, Jiaxing University, China

Abstract: *The purpose of this paper is to present a soft decoding algorithm orienting to hardware implementaion for the (24, 12, 8) Golay code, and implement such an algorithm in Field Programming Gates Array (FPGA). The soft decoding algorithm devised by Lin's et al. for the (24, 12, 8) is not suitable for hardware implementation because of involving many arithmetic operations, such as multiplications and divisions. To remove the complexity arithmetic operations, the absolute value of the channel information instead of the bit-error probability is employed to indicate the channel confidence. Moreover, the architecture developed for realizing the proposed algorithm is verified in a FPGA prototype. The BER performance obtained by the proposed decoding algorithm equals to the one obtained by Lin's algorithm. At the same time, 25% of additions, 100% of multiplications, and 100% of exponents are reduced for computing the channel confidence. In addition, 1-db coding gain can be obtained by Lin's algorithm at the cost of the double of hardware complexity compared to Elia's algorithm.*

Keywords: *Soft decoding algorithm, golay code, FPGA.*

Received July 17, 2011; accepted May 22, 2012; published online January 29, 2013

1. Introduction

In 1949, a perfect linear error-correcting code named Golay code was first discovered by Golay [4]. Later, several improved algorithms decoding the Golay code efficiently are given such as the algebraic decoding algorithm [2], the standard array decoding [6], and so on. Moreover, the (24, 12, 8) extended Golay code is generated by added a parity check bit to Golay code for the deep space communication [9]. Nowadays, this code is also used for hashing as shown in [3].

To improve the decoding performance, Boyarinov *et al*. [1] proposed a decoding algorithm up to four errors for a special construction of the (24, 12, 8) extended Golay code. Unfortunately, the simulation results of this method show that few of the four-error patterns can be corrected. More recently, a decoding algorithm proposed by Lin *et al*. [5] was developed to facilitate further decoding of the (24, 12, 8) Golay code by using Reed's method [8] to estimate the individual bit-error probabilities on a received word. It is shown that nearly 1-dB code gain is obtained compared to hard decoding algorithm.

However, the research of the (24, 12, 8) Golay code soft decoding primarily focuses on the design of the algorithm. Few topics about hardware implementation are discussed. It is obvious that hardware implementation plays an important role in application. The difficulties of hardware implementation for the decoding algorithm proposed in [7] are that: The complexity of computing the Emblematic Probability Value (EPV) defined in (16) of [7], which requires many arithmetic operations, such as multiplications and divisions in the decoding algorithm.

In this paper, a simplified decoding algorithm which significantly reduces the complexity of computing the EPV is proposed. By utilizing the cyclical characteristics of the Golay (24, 12, 8) code, a decoding architecture based on the simplified decoding algorithm is investigated. Meanwhile, the Field Programmer Gate Array (FPGA) prototype of the decoder architecture is implemented and analysed in this paper.

2. Preliminaries

The soft decoding algorithm proposed in [8], is summarized as follows:

- *Step 1:* Obtaining the error pattern E_P from the received codeword r based on the algebraic Hard Decoding (HD) algorithm for the (23, 12, 7) Golay code proposed by Elia [2].
- *Step 2:* The decoder detects four errors using the algorithm proposed by Reed *et al*. [7].
- *Step 3:* After detecting four errors, the six possible error vectors, that is, $\tilde{e}_1$, $\tilde{e}_2$, $\tilde{e}_3$, $\tilde{e}_4$, $\tilde{e}_5$ and $\tilde{e}_6$, are determined by searching a lookup table of weight-7. C_1, C_2, C_3, C_4 and C_5 indicate five weight-7 code words. $\tilde{e}_i$ for $1 \le i \le 6$ are the possible four error pattern, and $Loc(w_i)=\{i|w_i \neq 0\} \subseteq \{0,1,\ldots,n-1\}$ is a set

of the locations of non-zeros components in a vector $w=\{w_0,w_1,\ldots,w_{n-1}\}$.

- *Step 4:* The individual bit-error probabilities, namely $p_0, p_1,\ldots, p_{22}$, are first determined by an extension of ideas given by Reed [8] from a received word of length 24, after the determination of all possible error patterns. The probability p_k' for $0\le k\le 23$ is given by:

$$p_k' = \frac{1}{1+\exp(2A|x_k|/\sigma^2)},\ -\infty < x_k < \infty \tag{1}$$

Where x_k, A, and σ^2 are the particular realizations of a discrete-time conditional Gaussian random sequence, the received bit amplitude estimated from the sequence x_k, and the channel noise power, respectively.

- *Step 5:* The EPVs of the six possible error vectors are defined as:

$$\hat{P}_i' = \prod_{k\in Loc(\tilde{e}_i)} p_k',\ i=1,2,\ldots,6 \tag{2}$$

- *Step 6:* The most likely weight-4 error pattern corresponding to the maximal value of $\hat{P}_i'$ is chosen.

3. A Simplified Soft Decoding Algorithm for the (24, 12, 8) Extended Golay Code Up to Four Errors

Obviously, note that through obtaining the EPV of each error pattern, finding the most likely four error pattern is the key point to correcting four error patterns. The value EPV is defined as the product of four individual bit-error probabilities corresponding to the locations of the respective errors. Nevertheless, the process of obtaining the EPV results in highly computational complexity and consequently severely lowers the decoding speed for many arithmetic operations involved, such as multiplications and divisions.

Therefore, this paper proposed a simple scheme to estimate the EPV value which is redefined as the logarithm of the product of four individual precision measurement of the channel confidence rather than that used to be defined previously, and where the precision measurement is defined by the ratio of the bit-error probability and the bit-correct probability. Now, redefine the channel confidence p_k as $p_k=log(P_E/P_C).P_E$, called the bit-error probability, is given by the following formula:

$$\begin{aligned} P_E &= P\{A_K = -sgn(x_k)A \mid x_k\} \\ &= \frac{1}{1+\exp(\frac{2A\,sgn(x_k)x_k}{\sigma^2})} \\ &= \frac{1}{1+\exp(\frac{2A\mid x_k\mid}{\sigma^2})} \end{aligned} \tag{3}$$

P_C, called the bit-correct probability:

$$\begin{aligned} P_C &= P\{A_k = sgn(x_k)A \mid x_k\} \\ &= \frac{1}{1+\exp(\frac{-2A\,sgn(x_k)x_k}{\sigma^2})} \\ &= \frac{1}{1+\exp(\frac{-2A\mid x_k\mid}{\sigma^2})} \end{aligned} \tag{4}$$

Where the definition of σ^2, A, x_k, is the same as the ones mentioned above, and the sgn (x_k) denotes the sign function on x_k, respectively:

$$\begin{aligned} P_k &= log\,(P_E / P_C) \\ &= log\,(\frac{1+\exp(\frac{-2A\mid x_k\mid}{\sigma^2})}{1+\exp(\frac{2A\mid x_k\mid}{\sigma^2})}) \\ &= \frac{-2A\mid x_k\mid}{\sigma^2} \end{aligned} \tag{5}$$

Now, combined with equation 5, the EPV described in equation 2 can be modified as:

$$\begin{aligned} \hat{P}_i &= \sum_{k\in Loc(\tilde{e}_i)} p_k \\ &= \sum_{k\in Loc(\tilde{e}_i)} \frac{-2A}{\sigma^2}\mid x_k\mid \end{aligned} \tag{6}$$

Where $\hat{P}_i$ is called the modified EPV. To achieve the most likely weight-four error pattern, the maximum of the EPV, denoted as $max(\hat{p}_i)$, needs to be calculated as follows:

$$max(\hat{P}_i) = min(\sum_{k\in Loc(\tilde{e}_i)} \frac{2A}{\sigma^2}\mid x_k\mid),\ 1\le i\le 6 \tag{7}$$

If σ^2 and A are constants in a received word, the greatest EPV can be readily simplified as:

$$max(\hat{P}_i) = min(\sum_{k\in Loc(\tilde{e}_i)} \mid x_k\mid),\ 1\le i\le 6 \tag{8}$$

In fact, σ^2 and A are assumed to be constants among a received code word in a real communication systems. The number of operations required by using equations 2, 7 and 8 for computing the EPV and the maximum of the modified EPV are shown in Table 1, respectively. One observes from Table 1 that 25% of additions, 57% of multiplications, and 100% of exponents are reduced by using equation 7 when compared with the decoding algorithm of Lin *et al.* [5]. Specifically, if σ^2 and A are constants in a received word from equation 8, the number of additions needed to compute the maximum of the EPV is eighteen.

Now, the soft decoding algorithm proposed in [5] can be modified as follow:

- *Step 1:* The same operations are performed till the determination of all possible error patterns in the soft decoding algorithm described in [5]. Then, in view of σ^2 and A are assumed to be constants among receiving a code word in a real

communication systems, let the channel measurement be:

$$\vec{p}_k = -|x_k| \tag{9}$$

The EPVs of the six possible error vectors are computed as:

$$\begin{aligned} \hat{P}_i &= \sum_{k \in Loc(\tilde{e}_i)} \vec{p}_k \\ &= \sum_{k \in Loc(\tilde{e}_i)} -|x_k| \end{aligned} \tag{10}$$

- *Step 2:* The most likely weight-4 error pattern corresponding to the maximal value of $\hat{P}_i$ is chosen:

$$max(\hat{P}_i) = min(\sum_{k \in Loc(\tilde{e})} |x_k|),\ 1 \le i \le 6 \tag{11}$$

Table 1. The number of operations required by using equation 2 and equations 7 and 8 proposed in this paper for computing EPV.

Operation	Addition	Multiplication	Division	Exponent
Formula (2)	24	42	24	24
Formula (7)	18	18	24	0
Formula (8)	18	0	0	0

The Bit-Error Rate (BER) performances in AWGN are illustrated in Figure 1. It is seen obviously that the BER performance obtained by the Simplified Decoding Algorithm (SDA) equals to the one obtained by Lin's algorithm proposed in [5].

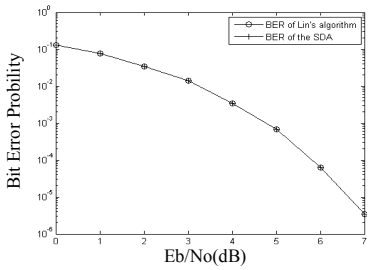

Figure 1. BER performance of the (24, 12, 8) extended Golay code in AWGN.

4. Hardware Implementation for the (24, 12, 8) Extended Golay Code Up to Four Errors

4.1. The Top Level of Soft Decoder

In this section, a decoder hardware implementation with a pipeline architecture for the (24, 12, 8) extended Golay code up to four errors is presented. This decoder can accept a new received message every twelve clocks. In other words, the time of each pipeline stage is twelve clocks. Concerning the decoding algorithm described in section 3, the functional block diagram of soft decoder is shown in Figure 2. The block named calculating $\vec{p}_k$ computes the channel measurement of each bit. The block of hard decoder obtains E_P based on the hard decoding algorithm proposed by Elia [2]. A detailed implementation for the hard decoder is described in [10]. The block of searching the most possible error pattern is used for finding the most possible four errors vector. The block called detecting four errors monitors an event that the receive code word is occurred by a four errors pattern. The architecture for detect four errors is devised in [7]. Is First In First Out (FIFO) buffer in Figure 2.

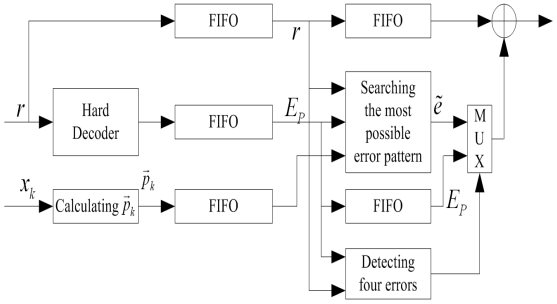

Figure 2. The functional block diagram of soft decoder.

4.2. Calculating $\vec{p}_k$

The block of calculating $\vec{p}_k$ computes the channel measurement $\vec{p}_k$ described in Equation 9, before searching the most possible error pattern.

4.3. Searching the Most Possible Error Pattern

If *r* occurred by four errors is detected by the block of detecting four errors, the block of searching the most possible error pattern starts to work. Once receiving E_P from the hard decoder, the block of searching the most possible error pattern reads $\vec{p}_k$ corresponded to E_P from FIFO. As shown in Figure 3, twenty-two search engines produce $\hat{P}_i^{tmp}$ and $\tilde{e}_i^{tmp}$ through searching the 253 valid code words in twelve clocks. 253 valid code words can be obtained by 11 valid code words $v=\{v_i | 1 \le i \le 11\}$ according to the characteristics of the cyclic codes. Each v_i generates twenty-two valid code words without overlap. The detailed searching of each search engine shown in Figure 4 is described as:

The vector $\tilde{e}_i^a$ is obtained from the XOR of E_P and v_i. Each element of $\tilde{e}_i^a$ controls its corresponding MUX, whose output is 0 for the control line equals to 0, and output is $\vec{p}_k$ for the control line equals to 1. The results of twenty-three MUX are added together to get the value $\hat{P}_i^a$, which then is compared to the $\hat{P}_i^{tmp}$ of the initial value of 0:

$$\begin{aligned} \hat{P}_i^{tmp} &= \begin{cases} \hat{P}_i^a & if\ \ \hat{P}_i^a > \hat{P}_i^{tmp} \\ \hat{P}_i^{tmp} & otherwise \end{cases} \\ \tilde{e}_i^{tmp} &= \begin{cases} \tilde{e}_i^a & if\ \ \hat{P}_i^a > \hat{P}_i^{tmp} \\ \tilde{e}_i^{tmp} & otherwise \end{cases} \end{aligned} \tag{12}$$

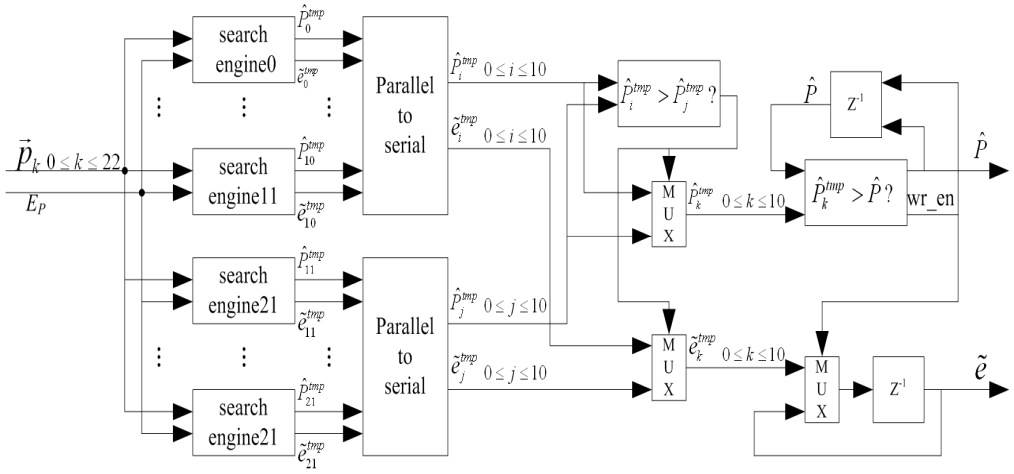

Figure 3. The architecture of searching the most possible error pattern.

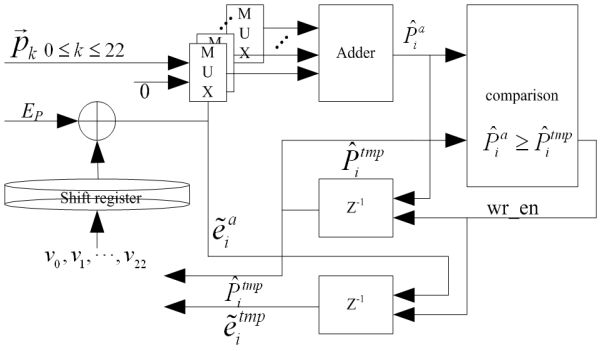

Figure 4. The architecture of search engine.

Then $\hat{P}_i^{tmp}$ and its corresponding $\tilde{e}_i^{tmp}$ are stored in the registers z-1, respectively. It is noted that during the period of generatin g one $\hat{P}_i^{tmp}$, where one new value v is generated from the previous v by left shifting one bit concurrently to obtained the next $\hat{P}_i^{tmp}$. After twelve clocks, the final maximum $\hat{P}_i^{tmp}$ corresponding to the EP is generated. The outputs of twenty-two search engines are divided into two groups, which are parallel-to-serialized into $\hat{P}_i^{tmp}$ $\{0 \le i \le 10\}$. $\hat{P}_j^{tmp}$ $\{0 \le j \le 10\}$ and $\tilde{e}_j^{tmp}$ $\{0 \le j \le 10\}$. Then the values $\hat{P}_k^{tmp}$ and $\tilde{e}_k^{tmp}$ $\{0 \le k \le 11\}$ are obtained by comparing the $\hat{P}_i^{tmp}$ and the $\hat{P}_j^{tmp}$ $\{i=j=0,1,...,10\}$:

$$\hat{P}_k^{tmp} = \begin{cases} \hat{P}_i^{tmp} & if\ \hat{P}_i^{tmp} > \hat{P}_j^{tmp} \\ \hat{P}_j^{tmp} & otherwise \end{cases}$$
$$\tilde{e}_k^{tmp} = \begin{cases} \tilde{e}_i^{tmp} & if\ \hat{P}_i^{tmp} > \hat{P}_j^{tmp} \\ \tilde{e}_j^{tmp} & otherwise \end{cases} \quad (13)$$

Then $\hat{P}_k^{tmp}$ is compared with $\hat{P}$ of initial value $\sum_{k \in Loc(E_P)} |x_k|$, if $\hat{P}_k^{tmp}$ is greater than $\hat{P}$, then $\hat{P} = \hat{P}_k^{tmp}$, else $\hat{P}$ keeps unchanged. The maximum element of $\hat{P}_k^{tmp}$ which is equals to $max(\hat{P}_i)$ is obtained after the twelve clocks.

5. FPGA Prototype

To verify the decoding architecture proposed in section 5, two FPGA prototypes are implemented in xc6slx150. These prototypes are designed for the SDA and Elia's algorithm respectively. Lin's algorithm is not implemented in the FPGA platform, because its some arithmetic operations are not suitable for hardware implementation. The working frequency of each FPGA prototype is 100MHz. The hardware cost and the corresponding thoughput of each decoder are given in Table 2. The BER performance comparisons of hardware simulations between the SDA and Elia's algorithm are described in Figure 5. It is illustrated clearly that SDA obtains a performance gain about 1-dB over Elia's algorithm although its hardware complexity doubles that of the latter.

Table 2. The hardware cost and the corresponding thoughput of decoders based on the SDA and Chase's algorithm.

	LUTs	Registers	Throughput
The SDA	5515	8491	240Mbps
Elia's algorithm	3737	3779	240Mbps
Ratio	1.47	2.25	1

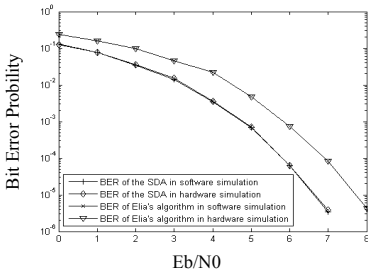

Figure 5. The BER performance comparison of software simulation and hardware simulation between the SDA and Elia's algorithm.

6. Conclusions

A simplified soft decoding algorithm up to four errors for the (24, 12, 8) extended Golay code is proposed in this paper. Simulation results show that the coding gain obtained by the simplified decoding algorithm

over elia's algorithm is the same as the one achieved by Lin's algorithm. At the same time, 25% of additions, reduced for computing the channel confidence. In line with this simplified algorithm, this paper investigates an efficient hardware implementation on FPGA platform, and presents the detailed architecture of the soft decoder. Compared to Elia's algorithm, 1-db coding gain can be obtained by Lin's algorithm at the cost of the double of hardware complexity.

Acknowledgements

The author would like to thank Dr. Lin wang and Dr. Hsin-Chiu Chang for their constructive suggestions on improving the quality and the presentation of this paper.

References

[1] Boyarinov I., Martin I., and H-onary B., "High-Speed Decoding of Extended Golay Code," *IEE Proceeding of Communication*, vol. 147, no. 6, pp. 333-336, 2000.

[2] Elia M., "Algebraic Decoding of the (23, 12, 7) Golay Code," *IEEE Transactions on Information Theory*, vol. 33, no. 1, pp. 150-151, 1987.

[3] El-Qawasmeh E., Safar M., and Kanan T., "Investigation of Golay Code (24, 12, 8) Structure in Improving Search Techniques," *the International Arab Journal of Information Technology*, vol. 8, no. 3, pp. 265-271, 2011.

[4] Golay M., "Notes on Digital Coding," *in Proceedings of IRE*, vol. 37, pp. 657, 1949.

[5] Lin T., Truong T., Su W., Shih P., and Dubney G., "Decoding of the (24, 12, 8) Extended Golay Code Up to Four Errors," *Institution of Engineering and Technology Communications*, vol. 3, no. 2, pp. 232-238, 2009.

[6] Mceliece R., *Theory of Information and Coding*, Addison Wesley, USA, 1977.

[7] Reed I., Yin X., Truong T., and Holmes J., "Decoding the (24, 12, 8) Golay Code," *in Proceedings of IEEE Computers and Digital Techniques*, vol. 137, no. 3, pp. 202-206, 1990.

[8] Reed I., "Statistical Error Control of a Realizable Binary Symmetric Channel," *Technical Report*, Massachusetts Institute of Technology, USA, 1959.

[9] Wicker S., *Error Control Systems for Digital Communication and Storage*, Prentice-Hall, USA, 1995.

[10] Wei S. and Wei C., "On High-Speed Decoding of the (23, 12, 7) Golay Code," *IEEE Transactions Information Theory*, vol. 36, no. 3, pp. 692-695, 1990.

Dongfu Xie received his BS degree in electronic engineering from Hangzhou Dianzi University, China, in 2006, and PhD in circuit and system from Xiamen University, China, in 2011. He is currently an assistant professor in the College of Mechanical and Electrical Engineering, Jiaxing University, China. His research interests includes error control coding, s ignal processing, and VLSI design.