Journal of Network Intelligence
Taiwan Ubiquitous Information
 ISSN 2414-8105 (Online)
Volume **2**, Number **4**, November **2017**

Enhanced Error-Recovery CAN Bus System Using Reed-Solomon Codec

Shi-Huang Chen and Yu-You Chu

Department of Computer Science & Information Engineering
Shu-Te University, Taiwan
No.59, Hengshan Rd., Yanchao Dist., Kaohsiung City 82445, Taiwan
shchen@stu.edu.tw, s12639108@stu.edu.tw

Received July 2017; revised November 2017

Abstract. *This paper proposes an improved CAN bus system that makes use of Reed-Solomon (RS) codec to enhance the error resistibility of traditional CAN bus protocol defined in ISO 11898 standard. The basic principle of the proposed method is to replace the original BCH code by RS code, and the selection of RS code type is dependent on the message length of CAN bus payload. By the use of MATLAB SIMULINK platform, this paper builds a noise interference simulation system for CAN bus with 5 kinds of message lengths. Then this paper applies different types of RS codec to replace the original BCH codec and designs an enhanced CAN bus frame structure. Experimental results show that the enhanced CAN bus frame structure could improve the error recovery performance 5 times under 0dB white noise interference and just increase double computation time. The proposed enhanced version of CAN Bus system is suitable for the application of in internet of vehicle (IoV) system.*
Keywords: CAN bus; cyclic redundancy check (CRC); BCH code; Reed-Solomon (RS) code.

1. **Introduction.** The development of automobile industry is evolved from basic transportation equipment to high tech vehicle to meet the requirements of safety, comfort, and pollution prevention. This trend leads to more and more electronic devices, such as air bag system, anti-lock braking system (ABS), electronic dashboard, fuel injection system, and etc, are used in vehicle. The large application of these in-vehicle electronic devices will result in the massive and complex body wiring harness. It goes without saying that it will also shorten installation space, decrease operational reliability, and increase difficulty of maintain. Therefore, the need arose for reducing the number of wire harnesses and transferring large amounts of data at high speed. However, most of the traditional electrical systems use point-to-point communications, and are unable to completely overcome the problems mentioned above [3-5]. To meet this requirement, Robert Bosch GmbH developed CAN bus system in 1986 as a new communication protocol for automotives [2]. The word of CAN stands for Controller Area Network. CAN bus is one type of serial communication protocols and is proposed to replace the traditional complex wiring harness with a two-wire bus system. Its basic structure is a multi-master distributed real-time control and message broadcast system. Hence CAN bus is designed to allow microcontrollers and devices to communicate with each other within a vehicle without a host computer. It is designed to solve data exchange among numerous control and test equipments in modern vehicles. The maximum signaling rate of CAN bus could reach 1 megabit per second

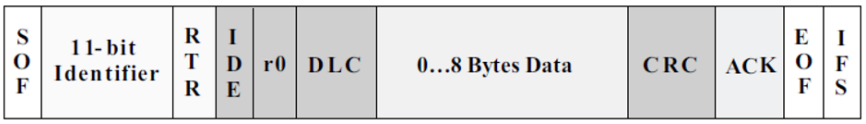


FIGURE 1. The format structure of a standard CAN 2.0A data frame [3]

(Mbps). Thereafter, CAN is proposed to be an international standard, i.e., ISO 11898, in 1993. The main advantages of CAN include good real-time feature, high reliability, quick communications rate, simple structure, perfect error handling mechanism of bus protocol, high flexibility, low price and so on. Today, CAN bus has been regarded as a vehicle bus standard and widely used in vehicle electric control system [1, 2]. Bosch published several versions of the CAN specification and the latest is CAN 2.0 released in 1991. The specification of CAN 2.0 has two parts called CAN 2.0A and CAN 2.0B, respectively. CAN 2.0A has an 11-bit identifier, whereas CAN 2.0B is the extended format with a 29-bit identifier [1]. Under CAN specification, messages are divided into frames with certain length and are sent in the format of frames. Fig. 1 shows the format structure of a standard CAN 2.0A data frame.

The meaning of the bit fields shown in Fig. 1 are [1, 3]:

- SOF-The single dominant start of frame (SOF) bit marks the start of a message, and is used to synchronize the nodes on a bus after being idle.
- Identifier-The Standard CAN 2.0A 11-bit identifier establishes the priority of the message. The lower the binary value, the higher its priority.
- RTR-The single remote transmission request (RTR) bit is dominant when information is required from another node. All nodes receive the request, but the identifier determines the specified node. The responding data is also received by all nodes and used by any node interested. In this way, all data being used in a system is uniform.
- IDE-A dominant single identifier extension (IDE) bit means that a standard CAN identifier with no extension is being transmitted.
- r0-Reserved bit (for possible use by future standard amendment).
- DLC-The 4-bit data length code (DLC) contains the number of bytes of data being transmitted.
- Data-Up to 64 bits of application data may be transmitted.
- CRC-The 16-bit (15 bits plus delimiter) cyclic redundancy check (CRC) contains the BCH error-correcting checksum (number of bits transmitted) of the preceding application data for error detection.
- ACK-Every node receiving an accurate message overwrites this recessive bit in the original message with a dominate bit, indicating an error-free message has been sent. Should a receiving node detect an error and leave this bit recessive, it discards the message and the sending node repeats the message after re-arbitration. In this way, each node acknowledges (ACK) the integrity of its data. ACK is 2 bits, one is the acknowledgment bit and the second is a delimiter.
- EOF-This end-of-frame (EOF), 7-bit field marks the end of a CAN frame (message) and disables bit-stuffing, indicating a stuffing error when dominant. When 5 bits of the same logic level occur in succession during normal operation, a bit of the opposite logic level is stuffed into the data.
- IFS-This 7-bit inter-frame space (IFS) contains the time required by the controller to move a correctly received frame to its proper position in a message buffer area.

Because of the execrable vehicle electromagnetic environment, the values of certain data bits may be altered due to the disturbance in the process of message transmission. This would cause false judgment at the receiving nodes. In this case, CRC specified in CAN standard is designed to detect such error codes. When an error is detected, the receiver sends an Automatic repeat ReQuest (ARQ) to the transmitter asking for retransmission. These methods enable error-free data transfer at the expense of the throughput. For high-speed CAN application, the efficiency is about 30%. It is obvious that the reduction in error frame detection failures would improve the application level of automobile network. It also could increase the reliability of communication, and furthermore ensure the safety of the automobile and the driver [6, 7]. This paper try to research the performance of using Reed-Solomon (RS) code to replace the original internal CRC-based BCH error-correcting checksum and proposed an enhanced version of the CAN bus system. Compared to the original version of CAN bus with standard BCH error-correcting checksum, experimental results show that the use of Reed-Solomon (RS) code could improve about 5 times of the anti-noise error correction capability. Therefore, the proposed enhanced version of the CAN bus system can improve the standard version of the anti-electromagnetic noise interference, and can be used in future electric vehicle control system. The rest of this paper is organized as follows: Section 2 describes the error-correction methods of Reed-Solomon (RS) code. Section 3 presents the proposed enhanced version of the CAN bus system. The experimental results are given in Section 4. Finally, the conclusions are provided in Section 5.

2. **Error Correction Schemes.** Error correction is performed by adding redundant information to the transmission message. Then the error correction scheme could detect and correct the error via the transmitted bits in the process of the receiving by the receiver. Correction codes are commonly divided into two types. They are convolutional codes and block codes. Convolutional codes are processed on a bit-by-bit basis. It uses the data of both current transmission and past transmission to form the redundant information. Block codes are processed on a block-by-block basis. It uses the data of the current transmission to form the redundant information. The error correction scheme used in this paper, i.e., Reed-Solomon code, and the BCH used in CAN bus standard, are belonged to block codes. In the block code structure, if a data block that contains k bits will be send, first the k bits data is mapped to a unique n bits code, and then the n bits codes are sent. One can use (n, k) to denote a block code with k bits data and n bit code. For a block code (n, k), the quantity of all the code is 2^n, and there are 2^klegal code. Therefore the encoded length n of a block code is $n = 2^k?1$, and the correctable error bit number t has a limitation, that is [8, 9]

$$t < \frac{2^k - 1}{2} \tag{1}$$

One can be found out from the block codes principle that if the number of error bit exceeds the limitation given in (1), then block code cannot detect and correct the error. As mentioned in Section 1, the standard CAN bus protocol makes use of a 15-bit cyclic redundancy check (CRC) BCH code as an error-detecting method. It is designed for detecting up to 5 randomly distributed errors or burst errors of length less than 15 in a message. However, such an original BCH code could not recovery message errors from serious radio jamming situations and result in resending of message. It will lead to reduction of CAN bus transmission rate. Furthermore, when the encoded message contains more than 6 errors, the undetected error frame will occur and the error data will be received as the correct data. If the data relate to vehicle driving safety, it would lead to unpredictable and serious consequences. To overcome this problem, this paper applies Reed-Solomon

(RS) code to replace the original BCH code and proposes an enhanced CAN bus frame structure. The description of Reed-Solomon code is as follow [8, 9]. Reed-Solomon (RS) code is non-binary cyclic codes with symbols made up of m-bit sequences, where m is any positive integer having a value greater than 2. RS (n, k) codes for $m-$bit symbols exists for all values of n and $k$satisfying the condition

$$0 < k < n < 2^m + 2 \tag{2}$$

where k is the number of symbols being encoded and n is the number of symbols in the encoded block. The relation between n and k of RS code can be given as

$$(n, k) = (2^m - l, 2^m - l - 2t) \tag{3}$$

where t is the number of symbols that the code can correct. Encoded block size n is determined by the expression given in (3) and k can be any value less than n with the only condition that $n?k$ should be even. The value of t is given by $t = (n - k)/2$. Let original message polynomial and encoded message polynomial be

$$M(x) = m_0 + m_1 x + m_2 x^2 + ... + m_{k-1} x^{k-1} \tag{4}$$

$$C(x) = M(x)G(x) = x^{n-k} M(X) + Q(x) \tag{5}$$

where $c_i, x_i \in GF(2^m), 0 < i < n - 1$. Then the corresponding RS generator polynomial is by the equation

$$G(x) = \prod_{i=d}^{i=d+2t-1} (x - a^i) \tag{6}$$

The encoded message polynomial could be calculated via the equation

$$C(x) = M(x)G(x) = x^{n-k} M(X) + Q(x) \tag{7}$$

. where $Q(x)$ is the remaining polynomial. Depending on the error-correction performance needed, the values of (n, k) can be selected and the number of symbols that can be corrected will always be t irrespective of the number of bits corrupted in each symbol. This is one of the major advantages of using RS codes. They can be used to counter the burst errors in CAN bus that have an average length of 3 to 5 bits. For more information about RS code, please refer to [Matlab-Communications System Toolbox-Error Detection and Correction].

3. **The Proposed Enhanced CAN Bus System.** Fig. 2 is the simulation flowchart of the proposed enhanced CAN bus system. The CAN system will first construct a completed frame format from input bit data. These bit data will be encoded via CRC (Cyclic Redundancy Check) with RS encoder. Before adding AWGN (Additive White Gaussian Noise) as interference, the PSK (Phase-Shift Keying) will first transfer these encoded binary bit data into analog signal. At the receiver, the RS decoded frame is then decoded using CRC to obtain the final decoded bit stream. Finally, the simulation error rate could be detected by comparing the bit value between original bit data and decoded output data. Comparisons have also been made for different bit value to evaluate the robustness performance.

4. **Simulation Results.** This paper makes use of SIMULINK simulation tool [11] and communication system toolbox [10] build-in MATLAB 8.0 (Release 2012b) to implement the original CAN bus 2.0A and the proposed enhanced CAN bus frame format. According to the CAN specification, the input bit data include SOF (Start of Frame), ID (Identifier), DLC (Data Length Code), and Data Field are provided by users, whereas the bit data in other fields are generated depended on CAN specification. This paper selects 0, 2, 4, 6, and 8 bytes as 5 types of input data. In the simulations, random data is generated as the bit value in the data field. Header and tail in other fields are added to make the frame

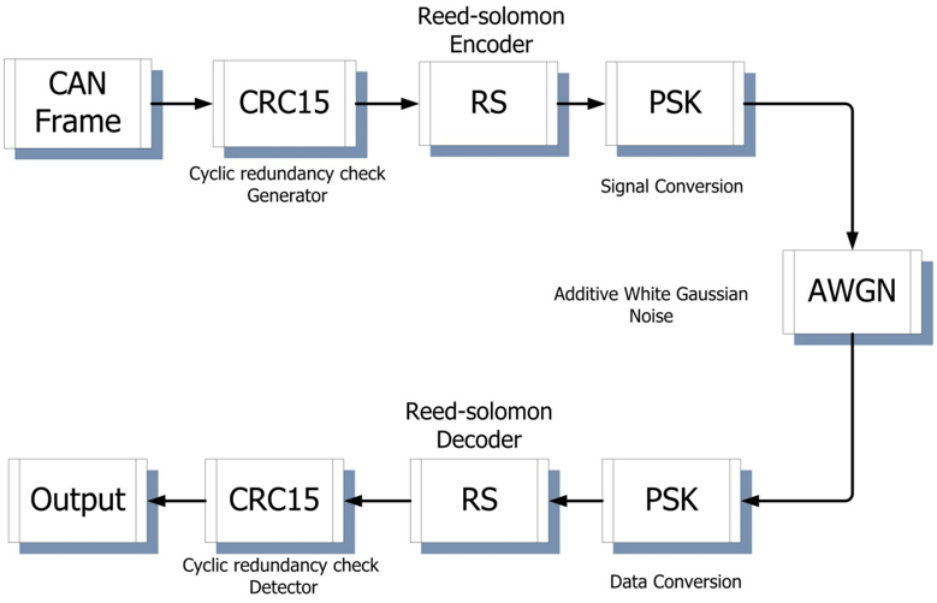


FIGURE 2. The simulation flowchart of the proposed enhanced CAN bus system

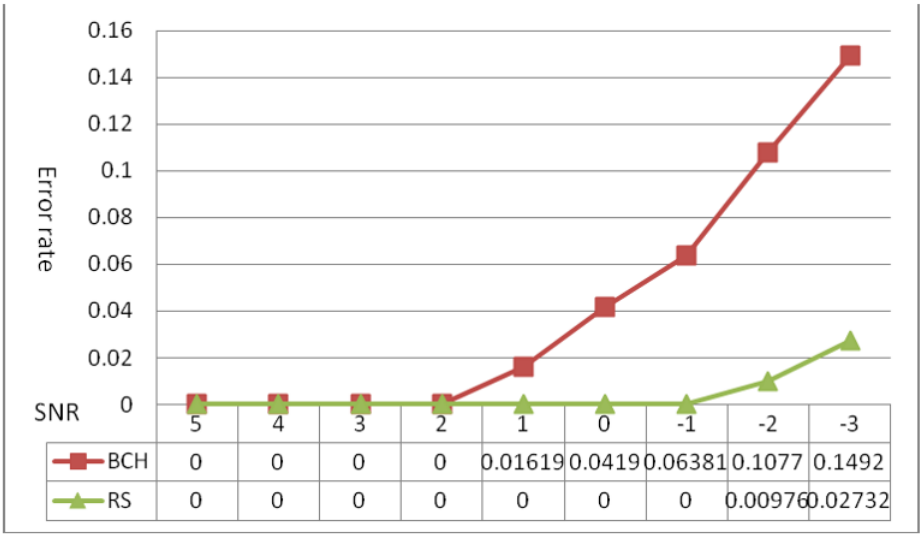


FIGURE 3. The error rate comparison results of BCH(63.36) and RS (63.39)

length 64, 80, 96, 112, and 128 bits, respectively. Hence, the BCH and the Reed-Solomon (RS) coding parameters are selected as BCH(63.36), RS (63.39), BCH(63.51), RS (63.55), BCH(127.64), RS (127.65), BCH(127.85), RS (127.85), BCH(127.99), and RS (127.99), respectively. All of the simulations are performed using a PC comes with Intel® Core? i5-3210M 2.5GHz with turbo Boost up to 3.1GHz, NVIDIA® GeForce® GT640M with 2GB Dedicated VRAM, 8 GB DDR3 Memory, and Windows 7 /64-bit OS. The simulation results are described as follow. When zero data, i.e., 0 byte, contained in the data field, the CAN bus frame length is 64 bits. This paper selects BCH(63.36) and RS (63.39) to encode the CAN data frame. Fig. 3 shows the comparison results of these two error correction schemes. It follows from Fig. 3 that RS (63.39) could recover the error up to -1 dB AWGN interference, whereas BCH(63.36) will lose error correction capability at 2 dB AWGN interference.

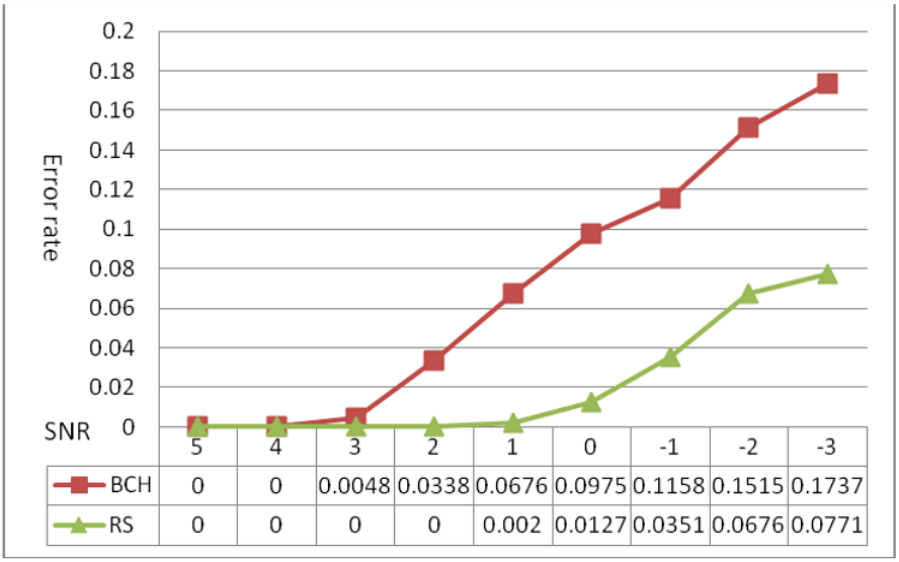


FIGURE 4. The error rate comparison results of BCH(63.51) and RS (63.55).

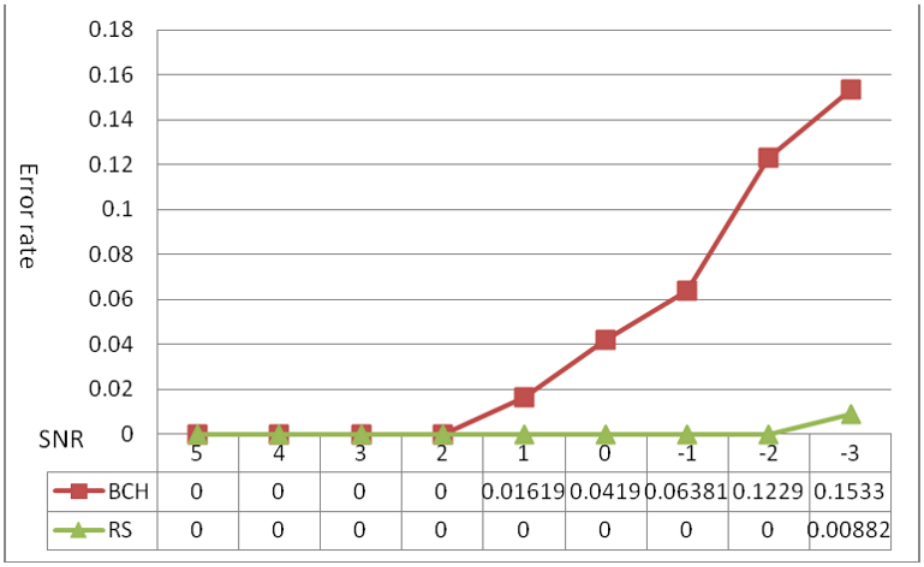
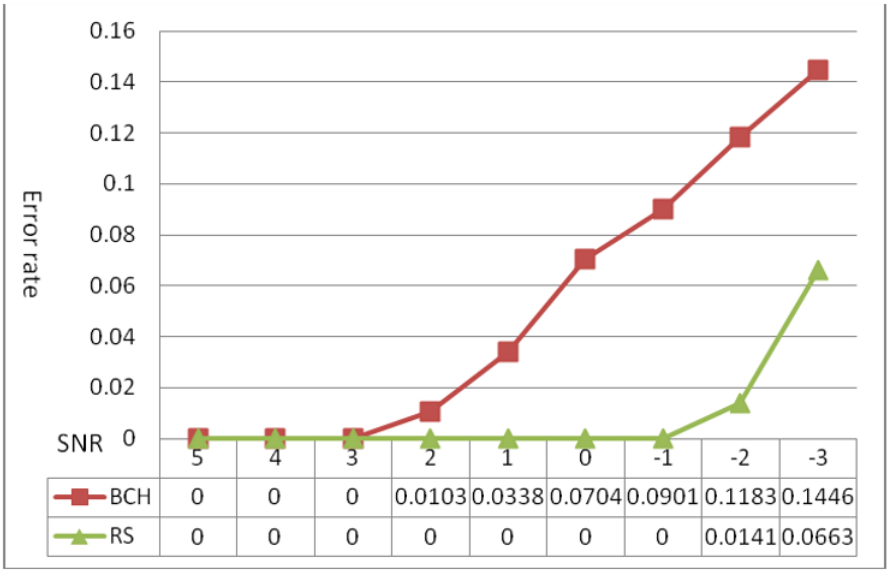


FIGURE 5. The error rate comparison results of BCH(127.64) and RS (127.65).

When the data field contains 2 bytes, the CAN bus frame length is 80 bits. This paper selects BCH(63.51) and RS (63.55) to encode the CAN data frame. Fig. 4 shows the comparison results of these two error correction schemes. It follows from Fig. 4 that RS (63.55) could recover the error up to 2 dB AWGN interference, whereas BCH(63.51) will lose error correction capability at 4 dB AWGN interference.

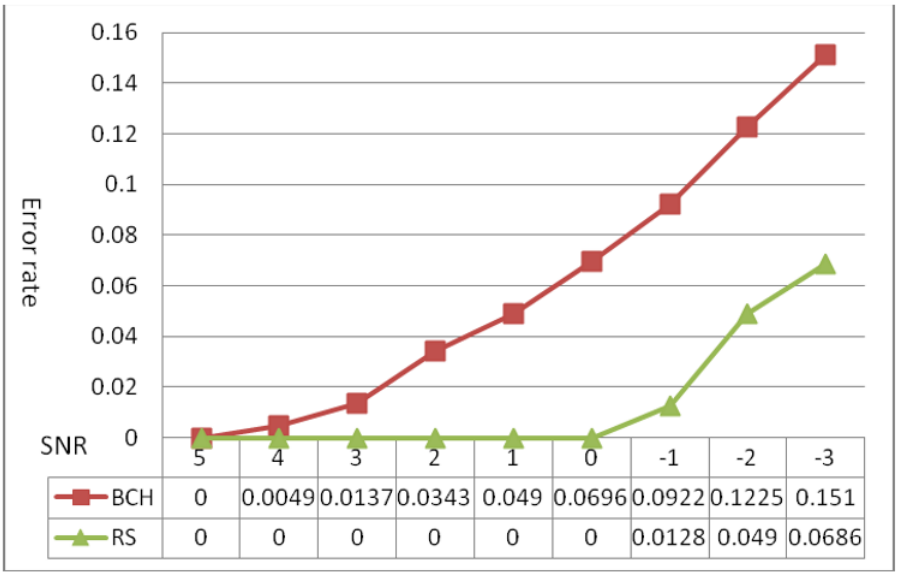
When the data field contains 4 bytes, the CAN bus frame length is 96 bits. This paper selects BCH(127.64) and RS (127.65) to encode the CAN data frame. Fig. 5 shows the comparison results of these two error correction schemes. It follows from Fig. 5 that RS (127.65) could recover the error up to -2 dB AWGN interference, whereas BCH(127.64) will lose error correction capability at 4 dB AWGN interference.

When the data field contains 6 bytes, the CAN bus frame length is 112 bits. This paper selects BCH(127.85) and RS (127.85) to encode the CAN data frame. Fig. 6 shows the



SNR	5	4	3	2	1	0	-1	-2	-3
BCH	0	0	0	0.0103	0.0338	0.0704	0.0901	0.1183	0.1446
RS	0	0	0	0	0	0	0	0.0141	0.0663

FIGURE 6. The error rate comparison results of BCH(127.85) and RS (127.85)



SNR	5	4	3	2	1	0	-1	-2	-3
BCH	0	0.0049	0.0137	0.0343	0.049	0.0696	0.0922	0.1225	0.151
RS	0	0	0	0	0	0	0.0128	0.049	0.0686

FIGURE 7. The error rate comparison results of BCH(127.99) and RS (127.99).

comparison results of these two error correction schemes. It follows from Fig. 6 that RS (127.85) could recover the error up to -1 dB AWGN interference, whereas BCH(127.85) will lose error correction capability at 3 dB AWGN interference.

When the data field contains 8 bytes, the CAN bus frame length is 128 bits. This paper selects BCH(127.99) and RS (127.99) to encode the CAN data frame. Fig. 7 shows the comparison results of these two error correction schemes. It follows from Fig. 7 that RS (127.99) could recover the error up to 0 dB AWGN interference, whereas BCH(127.99) will lose error correction capability at 5 dB AWGN interference. It is observed from Figs. 3 7 that the error correction performance of RS codes is better than that of BCH codes. The enhanced CAN bus frame structure with RS code could improve the error recovery performance 5 times under 0dB white noise interference. However, it is also well-known that computational loading of RS code is more complex than the BCH code.

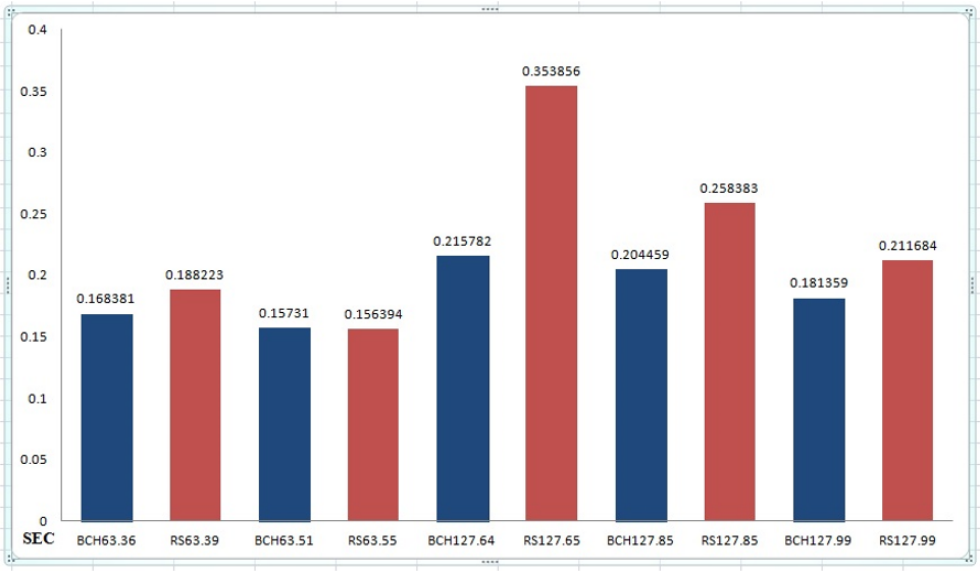


FIGURE 8. The average execution time of the five kinds of BCH codes and RS codes

According the simulations, the execution time of RS code is generally larger than BCH code about 0.02 sec. Fig. 8 is the average execution time of the five kinds of BCH codes and RS codes, e.g., BCH(63.36), BCH(63.51), BCH(127.64), BCH(127.85), BCH(127.99), and RS(63.39), RS(63.55), RS(127.65), RS(128.85), RS(127.99).

5. **Conclusion.** This paper applies Reed-Solomon (RS) codec to replace the original BCH codec and proposes an enhanced CAN bus frame structure. Several types of RS code including RS (63.39), RS (63.55), RS (127.65), RS (127.85), and RS (127.99) are tested in this paper. Experimental results show that the enhanced CAN bus frame structure could improve the error recovery performance 5 times under 0dB white noise interference even though RS code will increase computational time about 0.02 sec. The proposed enhanced version of the CAN Bus system is suitable for the application of in internet of vehicle (IoV) system.

REFERENCES

[1] CAN specification Version 2.0. Robert Bosch GmbH. 1991.
[2] Kvaser. A tour on the CAN Protocol. http://www.kvaser.com
[3] Steve Corrigan, Introduction to the Controller Area Network (CAN), *Texas Instruments Application Report*, 2008.
[4] Renesas Electronics Corporation. Renesas application note: Introduction to CAN, 2010.
[5] Wolfhard Lawrenz. CAN System Engineering: From Theory to Practical Applications, *Springer; 2nd ed.* 2013 edition.
[6] Emani, K.C., Rolla, Keong Kam, Zawodniok, M. Improvement of CAN BUS Performance by Using Error-Correction Codes, *Proceedings of the 2007 IEEE Region 5 Technical Conference, April* pp. 20-22, 2007.
[7] I.-An Chen, C. H. Cheng, H. Y. Jheng, C-K. Liu, F.-C. Cheng, S. J. Ruan, and C. H. Lin, An error-correction scheme with Reed-Solomon codec for CAN bus transmission, *Proceedings of the 2011 International Symposium on Intelligent Signal Processing and Communications Systems (ISPACS), Dec.*, pp. 7-9, 2011.
[8] Y. Wu, New List Decoding Algorithms for Reed?Solomon and BCH Codes, *IEEE Transactions on Information Theory,* pp. 3611 ? 3630, Aug. 2008

[9] Zhen Zhang, Some Recent Progresses in Network Error Correction Coding Theory, *Proceedings of the 2008 Fourth Workshop on Network Coding, Theory and Applications (NetCod 2008)*, Jan. pp. 3-4, 2008.

[10] Matlab - Communications System Toolbox - Error Detection and Correction. http://www.mathworks.com/help/comm/ug/error-detection-and-correction.html

[11] F. Li, L. F. Wang, C. L. Liao. CAN (Controller Area Network) Bus Communication *System Based on Matlab/Simulink, Proceedings of the 2008 4th International Conference on Wireless Communications, Networking and Mobile Computing (WiCOM08)*, Oct., pp. 12-14, 2008.