Fall 2007

Application of hybrid ARQ to controller area networks

Krishna Chaitanya Emani

APPLICATION OF HYBRID ARQ TO CONTROLLER AREA NETWORKS

by

KRISHNA CHAITANYA SURYAVENKATA EMANI

A THESIS

Presented to the Faculty of the Graduate School of the

UNIVERSITY OF MISSOURI–ROLLA

in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

2007

Approved by

Dr. Y. Rosa Zheng, Advisor

Dr. Jagannathan Sarangapani

Dr. Steven L. Grant

ABSTRACT

This thesis proposes two types of Hybrid Automatic Repeat reQuest (HARQ) schemes for the Controller Area Network (CAN) to combat Electro-Magnetic Interference (EMI) and improve network efficiency. The proposed HARQ schemes encode the original CAN data frames by a Reed-Solomon (R-S) code so that burst errors due to EMI may be corrected at the receive nodes. Therefore, the probability of error frames is reduced, thereby reducing the probability of retransmission. Hence, the network efficiency of the system is improved. HARQ Type-I employs a 20-bit R-S code to encode the CAN frame and transmits the R-S code along with the CAN frame. A variant of HARQ Type-I is also studied, which replaces the error-detection code, Cyclic Redundancy Check (CRC) in the original CAN frame by an error-correction cyclic code. HARQ Type-II either uses a 20-bit R-S code alone or a combination of 20-bit and 40-bit R-S codes. This method transmits the original CAN frame in normal conditions. Only when the receiver detects an error frame, it sends a negative acknowledgement (NACK) and the transmitter will send the parity bits of the error frames.

Computer simulations show that the proposed HARQ schemes have several advantages over the conventional ARQ currently used in the CAN bus. First, the error-correction capability in HARQ schemes enables the receivers to correct random or burst errors when they are spread over 2 R-S symbols. The burst length can be of any length less than 11 bits. Second, the HARQ scheme reduces the probability of retransmission by 100% for burst lengths shorter than 7 bits. When the probability of burst error occurrence is 40%, the probability of retransmission is reduced by 30% for burst lengths of 8 bits and 11% for burst lengths of 10 bits. Third, the proposed HARQ schemes require minimal changes in the CAN frame structure or the communication protocol and are easy to implement in practical hardware.

ACKNOWLEDGMENTS

"There is no elevator to success. You have to take the stairs." - The Alchemist

I would like to thank everyone who helped me achieve the targets and goals that I have set for myself.

Firstly I would like to thank my father, Mr. Siva Rama Krishna Emani for creating the zeal in me to work toward my Masters degree to enhance my technical and professional skills. I would also like to thank my mother, Mrs. Uma Devi Emani and my brother Mr. Vivekanand Emani for their emotional support throughout my stay in United States.

Leaving one's hometown and coming to a completely different country to pursue graduate studies can be a daunting task. Many ambiguous questions arise during the course of the thesis and one needs guidance in the right direction. I would like to thank Dr. Rosa Zheng, my academic advisor, for her astute guidance throughout my Masters program. I would like to thank Dr. Jagannathan Sarangapani and Dr. Steven Grant for their comments and suggestions on the thesis.

I wish to thank Dr. MacIej Zawodniok for his help with NS2 simulations for CAN and in writing a paper titled "Improvement of CAN bus Performance by using Error-Correction Codes" for IEEE Region 5 Regional Conference, Arkansas, Fayetteville. Special thanks to my research mate Mr. Fei Ren for his valuable inputs and data which helped me emulate the CAN environment in MATLAB. I would like to convey my sincere thanks to Caterpillar Inc. for giving me the opportunity to work on the CAN bus project through the CAT University Challenge Program.

Apart from the professional and technical guidance provided by my professors, some very special people in my graduate life have supported me emotionally and professionally. I would specially like to thank Praveen, Mohana, Srikanth, Kiran, Karthik, Sidharth, Pavan, Susheel, Praneeth, and all my undergraduate friends for standing by my side in my difficult times and helping me come out with flying colors. I would like to thank the entire India Association in Rolla who made my stay comfortable and made me feel at home.

Finally, I would like to thank every person I met over the period of my graduate program and all those who helped me climb the stairs smoothly.

TABLE OF CONTENTS

LIST OF ILLUSTRATIONS

LIST OF TABLES

1. INTRODUCTION

1.1. INTRODUCTION TO THE PROBLEM

The Controller Area Network (CAN) bus has evolved as a de facto standard in automotive and industrial networks. It is a serial communication protocol which supports distributed real-time control systems with a high level of fidelity [1]. Current applications of CAN include data transfer between sensors and actuators on automobiles, off-road machinery, and industrial control systems [2]. CAN bus is the backbone network for communication between networking controllers for engine timing, transmission, chassis and brakes, networking components of chassis electronics, and electronics which makes vehicles more comfortable (lighting control, air-conditioning, central locking, and seat and mirror adjustment). In industrial control systems, manufacturers have chosen to use CAN in medical apparatus, textile machines, special-purpose machinery and elevator controls. The serial bus system of the CAN bus is particularly well suited for networking intelligent I/O devices as well as sensors and actuators within a machine or a plant [3]. An example of a typical automotive CAN network is shown in Fig. 1.1 [4].

Complexity of the CAN applications has been increasing due to recent trends in the automotive industry. The number of nodes connected to the CAN bus has been increasing dramatically as a result of these complex applications. Vehicle automation require large numbers of sensors communicating with each other in real-time. Hence, these applications

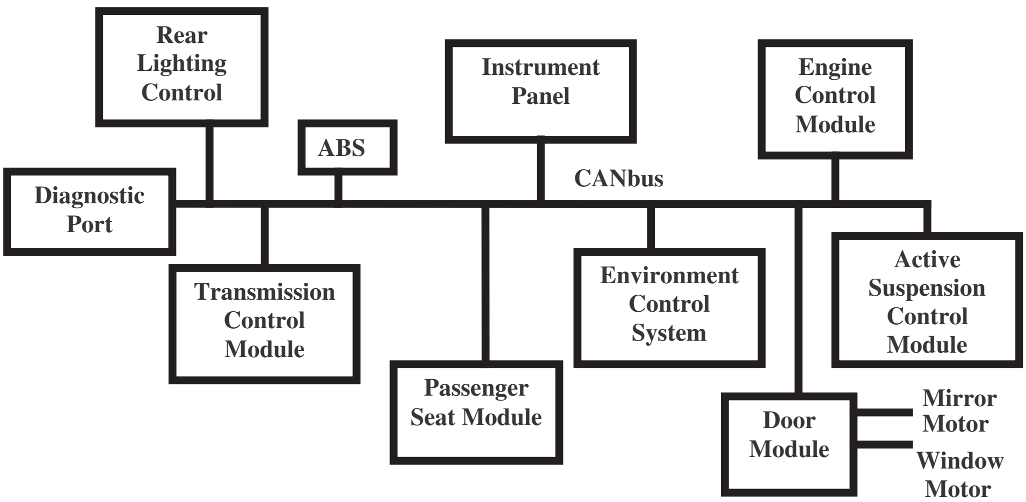

Figure 1.1 Example of Typical Automotive CAN Network

require high data rates to effectively communicate. Additionally, other parameters like the data throughput, latency demands, immunity to noise, and error detection capability are challenging the current capabilities of the CAN bus.

Standards and increased reliability in stringent Electro-Magnetic Interference (EMI) and Electrostatic Discharge (ESD) environments are challenging CAN system designers [4]. In general, a CAN transceiver must be able to survive high energy transients produced by a number of disturbances, including load dump, inductive load switching, relay contact chatter, and ignition system noise. In an automotive application the load dump is the severest transient. It occurs when the battery is inadvertently disconnected from the generator [4]. External interference, including cell phone signals, radio signals, etc., can also contribute to data corruption, and thereby inefficiency. The effects of EMI on the CAN bus can reduce efficiency by introducing bit errors into the frame. These bit errors can be single-bit or burst errors, depending on the severity of the EMI. Currently, the CAN bus uses differential bus twisted pair cables along with error-detection techniques to combat the EMI effects to a certain extent, if not completely.

The data throughput of the CAN network is a major concern in its current applications. The CAN network speed determines the kinds of application for which it be used. Small scale applications like general automobile applications use a low speed CAN bus with a data rate of 125 kbps. Large scale applications, like the off-road machinery, use a high speed CAN bus with a data rate of 1 Mbps. These applications utilize the CAN bus network not only because of the lower cost, but also because the network is robust [5]. Reliable transmission of the CAN frame is the priority of the CAN network. The CAN bus uses Carrier Sense Multiple Access with Bitwise Arbitration (CSMA/BA) MAC layer protocol to transmit the frames. The CAN transceivers check whether the frame is received correctly or not, but they do not concentrate on how many times the frame was retransmitted in order to reach the destination. In harsh environments, the effective bandwidth utilized by the high speed CAN bus is approximately 30%.

Five major factors affect the efficiency of the CAN bus. They are:

1. Electro-Magnetic Interference caused by the many magnetic relays and communication cables in a vehicle.

2. Inefficient stop-and-wait retransmission technique.

3. Broadcast systems which require retransmission when any node raises an error flag.

4. High bit overhead in each CAN frame.

5. Length of the CAN bus.

The types of errors created by interference could be single-bit errors or burst errors. This is one of the major factors contributing to the inefficiency of the CAN bus. As the number of inductive loads increase, the probability of a frame getting corrupted by EMI increases. In order to handle with the affect of EMI, the current CAN network implements three error-detection mechanisms at the message level: Cyclic Redundancy Check (CRC), Frame Check, and Acknowledgement Errors. It also implements two error-detection mechanisms at the bit level: Bit Monitoring and Bit Stuffing [3]. The corrupted frames are checked for errors at the receiver and if errors are detected, the receiver node raises an error flag and sends a request to the transmitter asking for retransmission [6]. Increase in the number of retransmissions has a huge impact on the throughput of the entire network.

The main aim of this work is to study the effect of Electro-Magnetic Interference (EMI) on the CAN bus and to propose new techniques to reduce its impact on the CAN bus performance.

1.2. MY APPROACH AND CONTRIBUTION

Current methods to combat EMI can detect errors, but not correct them. They ensure error free transmissions between the transmitter and the receiver at the expense of throughput. The Automatic Repeat reQuest (ARQ) protocol used in the CAN bus for retransmission of corrupted frames is very simple to implement, but inherently inefficient because of the idle time spent for acknowledgement and time spent for retransmission [7]. Hence, when the channel error rate is high, the number of retransmissions reduce the throughput of the network, which results in an efficiency close to 30% for a high speed CAN bus network. In applications like vehicle automation, this speed is insufficient to keep up with the current requirements. Hence, it is very important to improve the speed of the CAN bus without compromise to its error immunity. Implementing Forward Error-Correction (FEC) codes in conjunction with error-detection codes is a valuable solution to the problem. The function of the FEC system is to correct the possible errors and reduce the number of retransmissions, thereby increasing system performance. When error-correction alone is used to handle the errors, a decoding error is committed if the receiver either fails to detect or correct the errors. In both cases, a wrong word is delivered to the receiver [7]. Hence, use of FEC alone fails to maintain the reliability standards of the original bus. In order to overcome the drawbacks of the ARQ and FEC schemes, a hybrid technique of combining both the ARQ and FEC into one scheme is used. This technique is desicribed as Hybrid Automatic Repeat reQuest (HARQ) [7]. The FEC system corrects the possible errors in the frame and then the ARQ system detects if there are errors present in the decoded frame. In the case of error detection, retransmission of the frame is requested [8]. Hence, proper combination of the FEC and the ARQ schemes provide higher reliability than the FEC system alone and higher throughput than the ARQ system alone.

To design a FEC scheme to be used on the CAN bus, its important to find the impact of EMI on the CAN bus performance. Experimental tests were conducted by Fei Ren, an Electrical Engineering graduate student in one of the Electrical Engineering Laboratory G23 at the University of Missouri-Rolla. These tests were part of a systematic study of the CAN bus environment and the types of errors that can occur in such harsh environments. Fei programmed two 8051 micro-controllers to depict the CAN nodes and the connection between them to depict the CAN bus. One of these two boards was set as a transmitter and the other as a receiver. The connecting wire was an unshielded twisted pair copper cable wrapped around with a cable connecting the switch and the magnetic relay to emulate the environment of an inductive load on a CAN network. Communication was set up between the nodes and the magnetic relay was turned on and off manually using a switch. The CAN frame was observed at the receiver using digital oscilloscope to measure the EMI in terms of the number of bits corrupted. This provided valuable information for designing a new technique to combat the errors in the CAN environment. A similar environment has been emulated in simulation using MATLAB. Various simulations were performed to study the performance of the CAN bus using the proposed new techniques.

Five different methods of using error correction codes on the CAN bus have been discussed in this thesis. The first idea is to use cyclic codes to correct the bit errors instead of the Cyclic Redundancy Check (CRC). Two different versions of cyclic codes are proposed (Cyclic117, Cyclic60). Performance analysis in terms of error correcting capability and computational complexity are discussed in detail. Hybrid ARQ has been used as the basis for the other four methods proposed in this thesis. Two different versions of HARQ exist and they are termed as HARQ Type-I and HARQ Type-II. The second method proposed is based on HARQ Type-I in which Reed-Solomon codes have been used to correct the errors and CRC error-detection method has been used to detect the errors. This method is named as HARQ Type-I R-S20/CRC. The third method proposed is a slight modification to the HARQ Type-I R-S20/CRC method. The CRC bits in the CAN frame have been replaced by cyclic error-correction bits discussed in the first method to provide additional error-correction capability. This method is termed as HARQ Type-I R-S20/Cyclic. Performance analysis of these two variants of HARQ Type-I is provided in the results section. HARQ Type-II provides the same error-correction performance as HARQ Type-I but with a higher bandwidth efficiency. Two variants of HARQ Type-II are proposed in this thesis. Fourth method therefore, is to use the HARQ Type-II technique with the R-S codes and the CRC method for error-correction and detection respectively. This method is named as HARQ Type-II R-S20/CRC. The fifth method proposes to use a 40 bit parity R-S code along with the 20 bit parity method proposed earlier. This method is named as HARQ Type-II R-S40/CRC. In some cases when the R-S20 parity bits are not sufficient to correct the error frame, additional error-correction parity bits might correct the frame. Therefore, instead of transmitting the entire frame for the HARQ Type-I R-S20/CRC method or the HARQ Type-II R-S20/CRC method, these additional 40 R-S parity bits are transmitted. Simulation results to compare the bandwidth efficiency of each of these methods along with the conditions in which each of the method can be used are explained in the results and conclusion section.

Errors caused by EMI in the CAN bus are single-bit or burst errors depending on the environment in which it is being operated. For all the simulation results provided in this thesis, the burst lengths are assumed to vary from 0 bits to 10 bits, where 0 signifies no error condition and 10 bit burst represents worst conditions. The error-correction scheme has to be competent enough to correct these errors in most of the cases, if not in every situation. The first method (Cyclic60) of replacing the CRC bits with cyclic codes is capable of correcting single bit errors, but does not correct burst type errors. The second method (HARQ Type-I R-S20/CRC) corrects 100% of the frames for burst lengths shorter than 7 bits. When the probability of a CAN frame getting corrupted is 40%, the HARQ Type-I R-S20/CRC scheme requires 30% fewer frame retransmissions for burst lengths of 8 bits and 11% fewer for burst lengths of 10 bits. This is a significant contribution as the number of retransmissions will be reduced for all burst errors less than 7 bits and will be less than 50% for burst lengths greater than 7 bits. The HARQ Type-II methods improve the performance of the system in terms of bandwidth efficiency. Simulation results show that the percentage overhead for HARQ Type-II R-S20/CRC is always lower than that of the HARQ Type-I R-S20/CRC, when the probability of error is lower than 60%. At low channel error rate, the bit overhead of the HARQ Type-II is still less than its corresponding HARQ Type-I and also less than its corresponding ARQ scheme.

2. BACKGROUND

2.1. CONTROLLER AREA NETWORK (CAN)

The Controller Area Network (CAN) is a serial communication protocol supporting many distributed real-time applications with a high level of fidelity and security [1]. The applications of the CAN bus ranges from high speed to low speed communication networks. In applications like vehicle automation, sensors in the anti-skidding brake system require high speed communication to avoid the danger of delayed response. The bit rates for these kinds of applications are up to 1Mbps. For other applications, such as temperature control in a building, high bit rates are not required, so the bit rate of the CAN network used is 125 kbps. The CAN bus with data rates reaching the 1 Mbps mark is called a High Speed CAN bus. The other version of CAN bus, with speeds not exceeding 125 kbps, is called a Low Speed CAN bus.

2.1.1. CAN-Protocol and Principles of Data Exchange. The CAN protocol is an international standard defined in ISO 11898 [6]. Communication in the CAN bus is based on a broadcast mechanism in which one node transmits the frames and all the other nodes receive them depending on their urgency. Transmission of the CAN frame is message oriented rather than address oriented. The content of the message is designated by an identifier that is unique throughout the network [3]. Every CAN frame also has a priority level assigned to its particular message. This priority assignment is an important process which prevents several nodes competing for bus arbitration [6].

The content based communication protocol used by the CAN bus provides a high degree of flexibility. It is very easy to add a node to the existing CAN network without making any hardware or software modifications if the added nodes are receive only nodes. The multicast communication mechanism and the concept of message filtering enable any number of nodes to simultaneously receive the messages and take necessary action [1].

In real-time data transmission some of the frequently or fast varying data (e.g., engine load), needs to be transmitted more often and with higher priority and less delay than other less important messages (e.g., engine temperature). The priority level of the message is determined by its identifier. These identifiers are defined during system design in the form of binary values with the least binary value having the highest priority. Bus access conflicts are handled by the Bit-wise Arbitration (BA) of the identifiers involved in the conflict. In Bit-wise Arbitration the dominant mode overwrites the recessive mode. Hence, all nodes in the dominant receive and recessive transmission modes lose the competition to access the bus. All the nodes in the dominant receive mode become active receivers of the message and do not

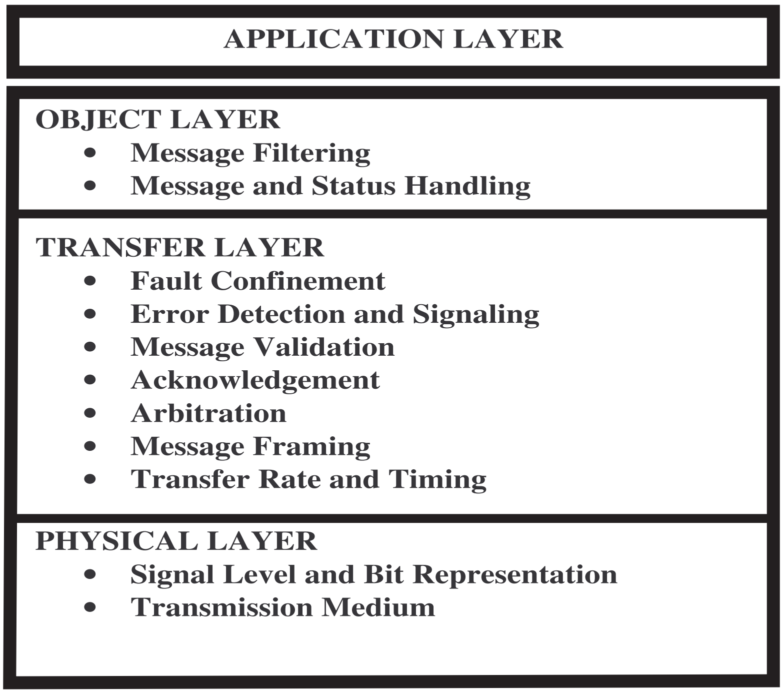

Figure 2.1 Layered Structure of a CAN node

attempt to transmit their message until the bus is available again. The system performance determines the order of importance of the transmitted messages [6].

To achieve design transparency and implementation flexibility the CAN node has been subdivided into three different layers [1]:

1. The CAN object layer
2. The CAN transfer layer
3. The physical layer

Layered structure of the CAN node is shown in Fig 2.1 [1]. All the major services and functions offered by the data link layer in the ISO/OSI architecture are handled by the object and the transfer layers in the CAN. The object layer provides an interface to the external hardware and deals with decisions regarding which messages are to be transmitted and which messages should be used. As the name implies, the transfer layer deals with the transfer protocol, i.e., control the framing, arbitration of the messages, error handling, etc. The physical layer deals with the transfer of the bits between the different nodes with respect to all electrical properties. These three layers make sure that the communication between the nodes is transparent to the end user or the external applications.

Each of the functions mentioned in the structured layout of the CAN node contribute to handling error free communication between the nodes. The functions handled by these three different layers are described in the next section to give an overview of the CAN operation and different constituents of the CAN protocol.

2.1.2. Information Routing. Content based message routing in the CAN bus leads to several other consequences which can be listed as follows:

1. System Flexibility
2. Message Routing
3. Multicast Communication protocol
4. Data Consistency
5. Bit Rate
6. Priorities Definition
7. Multimaster
8. Remote Data Request
9. Arbitration
10. Safety

Message routing in the CAN bus is content based with an identifier at the beginning of the message. This type of routing enables multiple receiver nodes to simultaneously receive data. Multicast communication demands that the data be consistent because messages should be compatible with all the nodes accessing the data. The bit rates of the CAN bus can be different for different systems. However, for a given system the bit rate is uniform and fixed [1].

The priorities have to be set when the network is initially designed. These priorities determine which unit has to gain control of the network for data transmission. A system in which any node can transmit data when the channel is available for transmission is called a multi-master system. When two or more systems try to access the network at the same time Bit-wise Arbitration resolves the conflict with the help of the identifier.

Safety is an important parameter in calculating the performance metrics of the CAN. In order to achieve utmost safety powerful error-detection techniques have been implemented on each of the CAN node. Different error-detection techniques on the CAN nodes include:

- Bit Monitoring
- Bit Stuffing
- Message Frame Check or Form Error
- Acknowledgement Error
- Cyclic Redundancy Check

2.1.3. Message Transfer in CAN. Messages are transferred in the form of frames. These frames are categorized into four different types depending on their function.

1. Data Frame
2. Remote Frame
3. Error Frame
4. Overload Frame

The data frame carries the data from the transmitter to the receiver, while the remote frame is transmitted by any unit to request for the frame with the same identifier. An error frame is transmitted by any node that detects an error on the bus. An overload frame is used to create an intentional delay between the transmissions of two frames. This overload frame is obsolete now and is hardly used in any application.

1. Data Frame: The CAN data frame is composed of seven different bit fields. They are: Start of Frame, Arbitration Field, Control Field, Data Field, Cyclic Redundancy Check Field, Acknowledge Field, End of Frame.

Two different versions of the CAN data frame, Standard Format and Extended Format are available depending on the number of bits in the Identifier field. Figure 2.2 and Fig. 2.3 show the Standard and Extended CAN frames respectively [1]. The Standard CAN frame contains 108 bits and the Extended CAN frame has 126 bits. The Identifier field has 11 bits in the Standard format and 29 bits in the Extended format. The greater number of bits in the identifier of the extended frame format provides access to a higher number of the CAN nodes. For all the simulations described in this thesis, the CAN frame length is assumed to be 128 bits. Certain applications use a CAN frame of 126 bits, neglecting two bits in the header. Two optional bits are present in the header of both versions of CAN frames: the Reserved bit (R0) and the IDentifier Extension bit (IDE) in the standard frame and the Substitute Remote Request (SRR) and the IDentifier Extension bit (IDE) in the extended frame structure. An

SOF 1bit	Arbitration 12 bits	Control 6 bits	Data Field 64 BITS	CRC 15 bits	ACK 2 bits	EOF 7 bits

Figure 2.2 Standard CAN Frame with Individual Fields Shown

SOF 1bit	Arbitration 30 bits	Control 6 bits	Data Field 64 BITS	CRC 15 bits	ACK 2 bits	EOF 7 bits

Figure 2.3 Extended CAN Frame with Individual Fields Shown

Table 2.1 Bit arrangement of the Data Length field

No: of Data Bytes	DLC3	DLC2	DLC1	DLC0
0	0	0	0	0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0

additional bit present in the tail called the CRC delimiter, is common to both versions of the CAN frame [9].

The data field length in the CAN frame is 64 bits. Four bits before the data field define its length. These four bits are either recessive or dominant, depending on the length of the data field. Table 2.1 gives all the possible combinations of the arrangements of the bits in these four bit positions [1].

Cyclic Redundancy Check (CRC) Field: This field consists of the CRC sequence followed by the CRC delimiter. The CRC sequence is derived from a cyclic redundancy code best suited for frames with fewer than 127 bits. All the fields including the Start of Frame, the Arbitration Field, the Control Field, and the Data Field are divided by the CRC polynomial. The remainder of this division is the CRC sequence, which is appended to the CAN frame and transmitted over the bus [1]. The expression for the CRC polynomial used by CAN to generate the CRC sequence is given by

$$X^{15} + X^{14} + X^{10} + X^{8} + X^{7} + X^{4} + X^{3} + 1 \tag{2.1}$$

2. Remote Frame: Any node can request transmission of a particular frame from the transmitter by sending a Remote Frame. The Remote Frame is same as the Data Frame, except that it has no data field. The Remote Transmission Request bit (RTR) is dominant for a Data Frame and recessive for a Remote Frame.

3. Error Frame: An Error Frame consists of two different fields. The first field is given by the superposition of error flags raised by different stations. The second field is the Error Delimiter. There are two kinds of error flags: active and passive error flags. The Active error flag consists of six consecutive dominant bits. The Passive error flag has six consecutive recessive bits, unless they have been overwritten by six dominant bits by another active node.

4. Overload Frame: This is an obsolete frame which is no longer used in the current CAN bus. This frame consists of two fields, the Overload Flag and the Overload Delimiter and is used to introduce an intentional delay between two successive frame transmissions [1].

2.2. ERROR CORRECTION TECHNIQUES

Errors can be handled in different ways. Simply neglect the errors and live with them, detect an error in the received data and inform the transmitter that an error was received and request the transmitter to re-transmit the data again, or a third option is to detect the error and use some technique to correct it, if possible.

The first of the methods is a crude way of saying live with the errors and get as much information as you can. The second method mentioned above is implemented in practice and is called Automatic Repeat reQuest (ARQ). This is the current standard used by the CAN bus. In this method, the receiver checks for a particular set of conditions that the received data has to meet. If the received data does not meet the set of rules, it detects an error and informs the transmitter. The transmitter then retransmits the frame and these retransmissions reduce bandwidth efficiency. Transmitters have to maintain a record of all the previous transmissions. The third method mentioned is receiving increasing attention. This method reduces the number of retransmissions and can also be used in applications where there is no reverse channel of communication. Additional bits are added to each frame, but the receiver is now capable of finding and correcting the errors in the received frame.

The Forward Error Correction (FEC) technique adds controlled redundancy to the data for the receiver to correct the errors. Hence, the data required to convey the same information is increased. For all real-time applications the data rate has to be increased which, in turn demands more bandwidth. At the cost of this increased bandwidth, the chance of receiving an erroneous frame is reduced. If the application is not real time then FEC can be used with certain allowable delay. Linear block codes are a major category in FEC techniques.

2.2.1. Linear Block Codes. Linear Block Codes are a class of parity check codes that can be characterized by the (n, k) notation [10]. The transmitter encodes a block of k bits into a block of n bits. The k bits can be arranged into 2^k messages called the k-tuples. Each of the 2^k messages has to be encoded as an n bit sequence. There are 2^n sequences possible and these sequences form what is called the n-tuples. Each of the 2^k k-tuples is mapped uniquely to one of the 2^k code words of the n-tuples.

Linear block codes can be executed with little complexity or calculation, an easy decoding technique as compared to other methods. Among the different types of linear codes available, cyclic codes are the most popular.

Cyclic Codes: As the name implies, Cyclic Codes are cyclic in nature, i.e., if C is a codeword for the given data set, then C with the bits rotated by k positions is also a codeword for that set of data.
If $C = C_1C_2C_3C_{n-1}C_n$ is a code word then,
$C = C_nC_1C_2C_3C_{n-1}$
and $C = C_{n-1}C_nC_1C_2C_3C_{n-2}$ are also valid code words.

The n and k values determine the amount of overhead that each one of these sequences has to carry. If the value of $k = 3$ and the value of $n = 7$, there is an overhead of 4 bits for every message frame. The ratio k/n is called the rate of the code.

For any number m greater than or equal to 3, the values of n and k are determined by

$$n = 2^m - 1; k = n - m \Rightarrow 2^m - 1 - m. \tag{2.2}$$

Example: For $m = 3$ we have $n = 7$ and $k = 3$. If the message is of length $k = 3$, the encoded message conveying the same information is of length 7 with the advantage that it can decode and correct all single bit errors in the received sequence. The rate of this code is $3/7$, which is less than $1/2$, so, this code might have better error handling performance, but the bandwidth utilization is not high. If $m = 4, n = 15$ and $k = 11$ with a code rate of $11/15$, the rate in this case is greater than $1/2$, but the error correction capability will not be as good as the $m = 3$ case. So, the values of n and k determine the efficiency of the cyclic codes in terms of error-correction capability and bandwidth efficiency.

The capacity of any linear code to correct an error is determined by a factor called the hamming distance. The weight of any vector is given the number of ones in that data vector.

Table 2.2 Number of Correctable and Detectable Errors

Correction(X)	Detection(Y)
0	4
1	3
2	2

The distance between any two vectors is given by the count of the number of bits by which they differ. The distance can be found by calculating the XOR sum between vectors A and B and then finding the weight of the sum [10]. This can be represented in the form of an equation as

$$D(A, B) = W(XOR(A, B)) \tag{2.3}$$

where $D(A, B)$ is the distance between the vectors A and B, $W(.)$ is the weight of the vector, and $XOR(A, B)$ is the XOR sum of the vectors A and B. So, in a given set of code words we have a minimum distance (D_{min}) that is always maintained in between any two vectors. This D_{min} determines the number of errors that can be corrected or the number of errors that can be detected by a linear code.

The relation between the number of errors that can be corrected (X) and the number of errors that can be detected (Y) and the minimum distance D_{min} is given by

$$D_{min} = 2 * X + 1. \tag{2.4}$$

$$D_{min} \geq X + Y + 1; X \geq Y \tag{2.5}$$

This implies that, if $D_{min} = 5$, the number of errors that can be corrected and number of errors that can be detected can be arranged in the combinations given in Table 2.2. The performance of cyclic codes in terms of error correction capability can be observed in Fig. 2.4, which compares the Bit Error Rate (BER) with and without coding plotted to the SNR values ranging from 0 to 10.

2.2.2. Combating Burst Errors. Simulations of the cyclic codes performance showed that they can correct random bit errors, but in the case of burst errors these cyclic codes do not offer good performance.

Two common approaches are used to combat burst errors. The first one is the interleaver approach. A linear interleaver feeds the data to be transmitted into rows of a matrix and then reads out from columns. At the receiver the received data is fed into the matrix column-wise

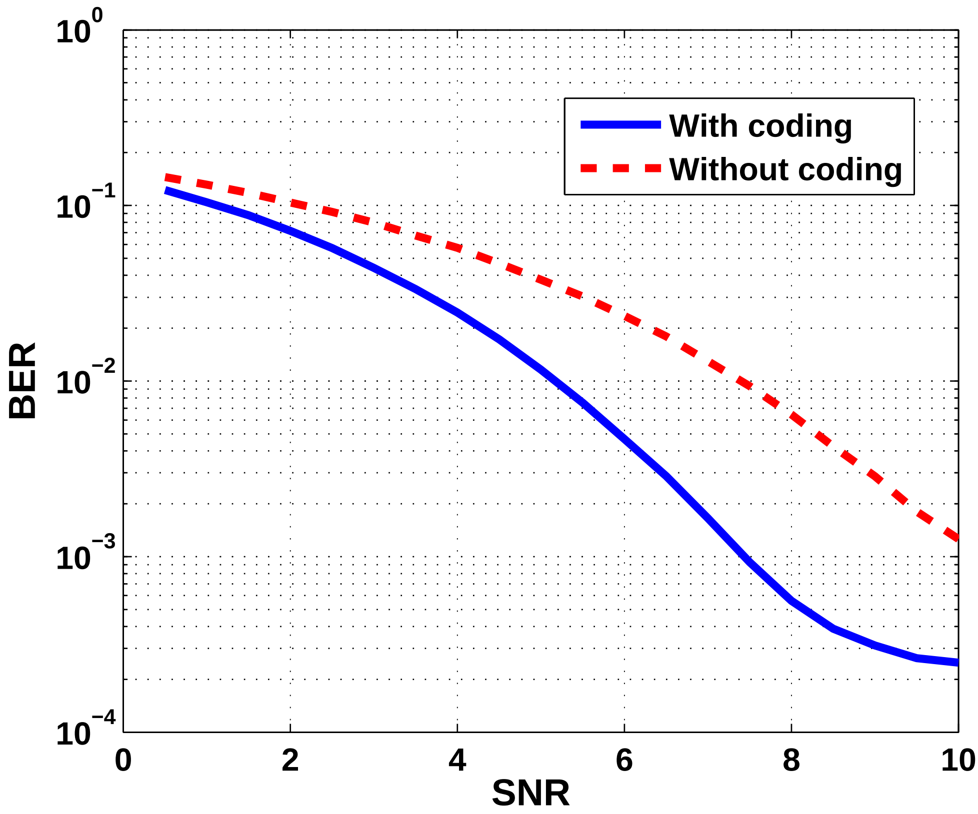

Figure 2.4 Comparison of BER curves without and with cyclic coding $n = 7; k = 3$

and read out row-wise. This de-interleaved data is decoded using normal decoding techniques. The process of interleaving shifts the burst error and converts it into a single bit error, i.e., separates adjacent bits by R positions, where R is the number of rows in the interleaver matrix. Figure 2.5 shows the performance improvement when the data is interleaved.

Use of an interleaver solves the problem of burst errors to an extent, but the disadvantage of this method is the delay in de-interleaving the received sequence. The receiver has to wait until it receives the frames that contribute to the data sequence and has to save all the received data until it receives all the interleaved frames to decode the original data sequence. Hence, a method in which the receiver can decode the data, depending only on the current frame needs to be designed. The second approach to combat burst errors is to use Reed-Solomon codes. Reed-Solomon codes offer good error-correction capability in the burst error environment.

Reed-Solomon(R-S) codes are also a kind of block codes. R-S codes are non-binary cyclic codes with symbols made up of m-bit sequences where m is any positive integer having a value greater than 2 [10]. R-S(n, k) codes for m-bit symbols exists for all values of n and k, satisfying the condition given by

$$0 < k < n < 2^m + 2 \tag{2.6}$$

where k is the number of symbols being encoded and n is the number of symbols in the encoded block. The values of n and k can be selected according to the requirement. The relation between the values of (n, k) can be given as

$$(n, k) = (2^m - 1, 2^m - 1 - 2t) \tag{2.7}$$

where t determines the number of symbols that the code can correct. Encoded block size n is determined by the expression given in Eq. 2.7 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

$$t = \frac{n - k}{2} \tag{2.8}$$

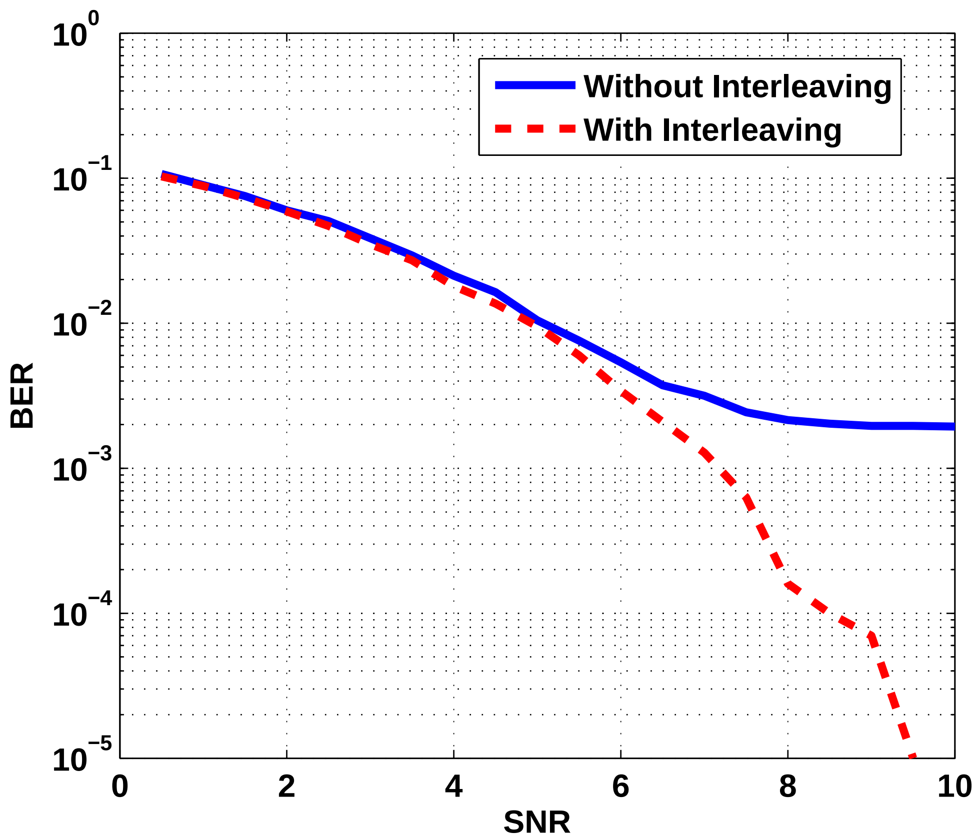

Figure 2.5 Comparison of BER curves for Cyclic codes with and without interleaving

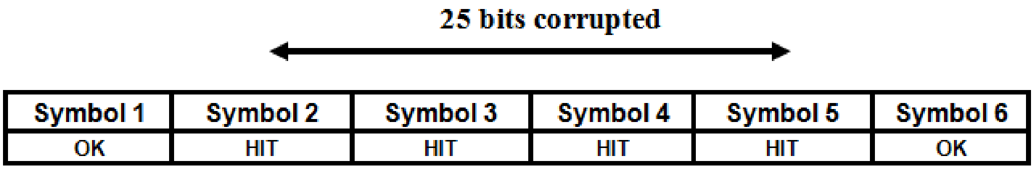

Figure 2.6 Frame in which 25 bits corrupt 4 symbols

Why R-S Codes Perform Well Against Burst Noise? Consider an example of $(n, k) = (255, 247)$ R-S code in which each symbol is made of $m = 8$ bits. Since $n - k = 8$, Eq. 2.8 indicates that this code can correct four symbols in a block of 255 symbols. Imagine a noise burst corrupts 25 bits of the frame in a transmission. This burst of 25 bits can corrupt exactly four symbols and, because the code has the capability of correcting all the four symbol errors without any regard to the damage suffered by each of the symbol, the original bit sequence can be recovered. Thus, if a symbol is wrong it might be corrupted in all the bit positions or it might be corrupted at only one bit position [10]. Figure 2.6 clearly shows how 25 bit errors can be distributed over four symbols. Eight bits are corrupted in each of the three symbols and only one bit is corrupted in the fourth symbol. All four symbol errors can be corrected using Reed-Solomon error correction method mentioned above.

Hence, Reed-Solomon codes have been used to counter the burst errors created by EMI in the CAN environment. The procedure followed and the design values of the R-S codes used for this particular application are discussed in detail in Section. 4.

2.3. ARQ AND HYBRID ARQ

This section introduces the three different types of Automatic Repeat reQuest (ARQ) protocols.

1. Automatic Repeat reQuest (ARQ)

2. Hybrid Automatic Repeat reQuest Type-I (HARQ Type-I)

3. Hybrid Automatic Repeat reQuest Type-II (HARQ Type-II)

2.3.1. Automatic Repeat Request (ARQ). Error detection is an important error control technique used by many communication systems these days. Automatic Repeat reQuest (ARQ) is a widely used error control technique because of its simplicity of use. ARQ relies on an error detection code such as the Cyclic Redundancy Check (CRC). Importance of the ARQ scheme is to turn an unreliable data link into a reliable one. In general, when a source

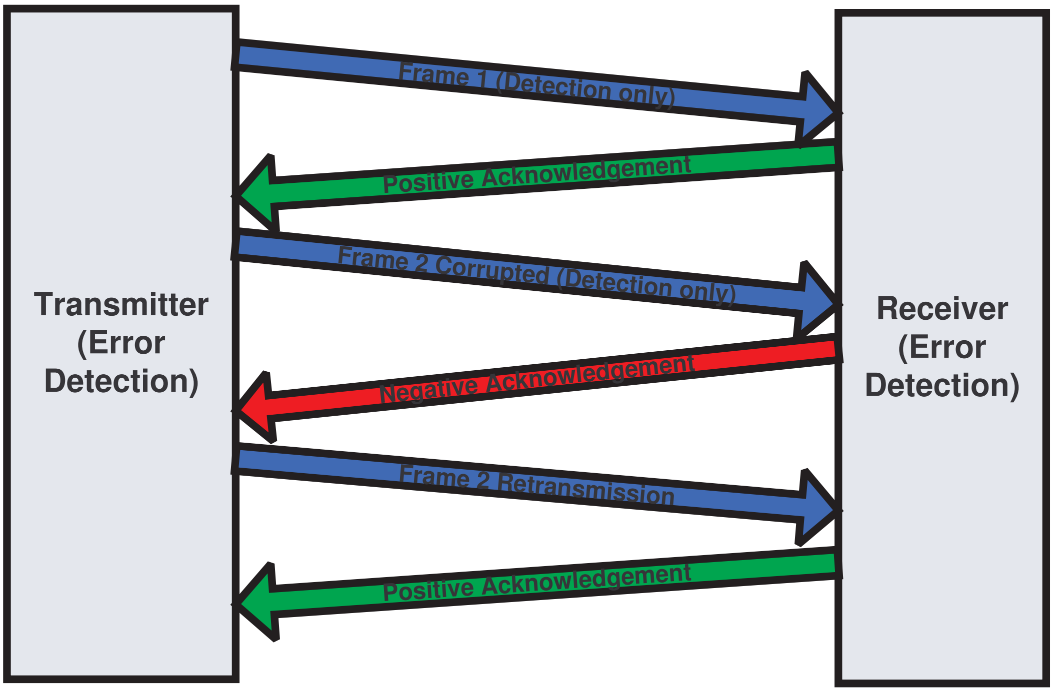

Figure 2.7 ARQ Protocol with the Data Flow and Acknowledgements

has a stream of data to transmit, it breaks up the stream of data into smaller packets and transmits the individual packets. Each of these packets are encoded using the error-detection coding technique and transmitted through the noisy channel [11]. When the packet is received, the receiver computes the syndrome of the error detection code used. If the syndrome for the received packet is zero, it implies that the packet is error free and the receiver accepts the packet, separating the parity bits. The receiver sends a positive acknowledgment to the transmitter to confirm the reception of the packet. If the acknowledgement is not received, the transmitter waits for a pre-determined amount of time and then retransmits the packet. If the syndrome is not equal to zero, receiver detects an error in the received packet and sends a negative acknowledgement to the transmitter requesting retransmission of the corrupted packet [7]. The receiver discards the corrupted packet and all further packets until the corrupted packet is correctly received. An erroneous packet is delivered to the destination node only when the error-detection scheme fails to detect the error. A proper error detecting code can greatly reduce the probability of receiving an erroneous frame. However, the throughput of the network is not constant as the increase in error rate reduces the throughput of the network drastically [7]. The flow of the data frames along with their acknowledgements for the ARQ scheme is shown in Fig. 2.7.

2.3.2. Hybrid Automatic Repeat Request(HARQ) Type-I. Retransmission of the data frames is a waste of bandwidth and reduces the throughput of the network. The

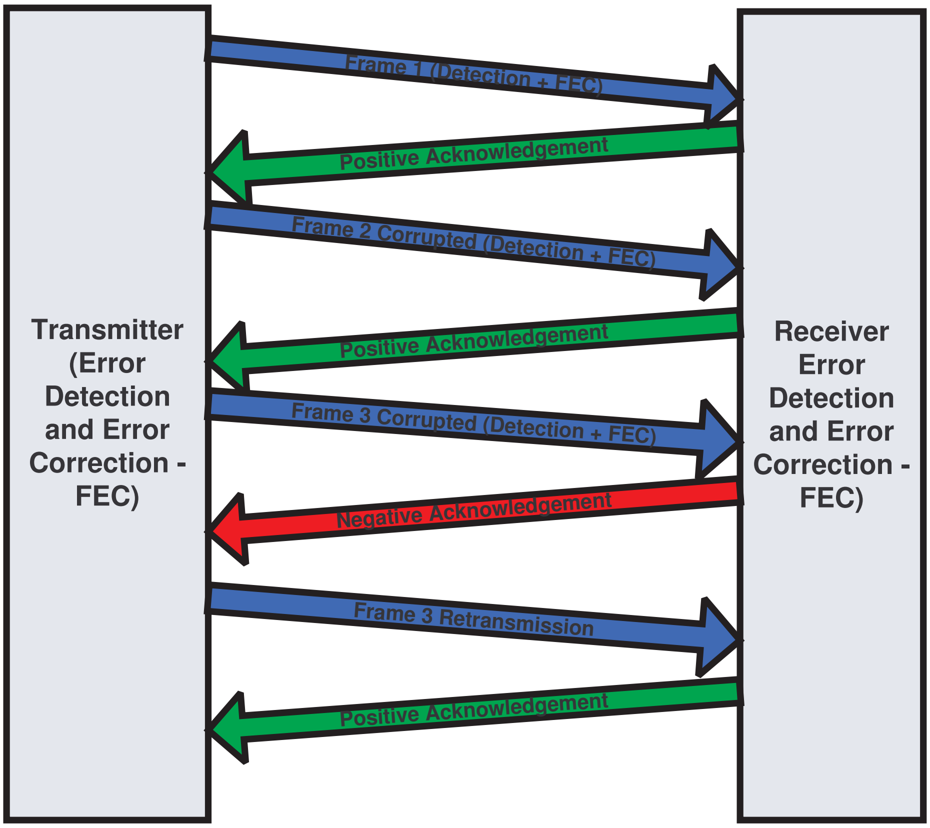

Figure 2.8 HARQ Type-I Protocol with the Data Flow and Acknowledgements

number of retransmissions should be reduced with the same level of reliability in order to improve the throughput of the network. Hybrid Automatic Repeat reQuest (HARQ) is a technique which implements Forward Error Correction (FEC), along with the error-detection used by the ARQ schemes. Hybrid ARQ offers the potential for better performance if the ARQ and the FEC schemes are properly designed. Either convolutional codes or block codes can be used in the FEC depending on the performance criteria. Due to combination of the two basic error control schemes, it is called Hybrid ARQ. In this system the purpose of the FEC is to reduce the frequency of retransmissions by correcting the frequently recurring error patterns. The purpose of the error-detection technique is to maintain the reliability of the network. When an error pattern that cannot be corrected is received, the receiver detects the error and requests for a retransmission. As a result of this combination, the throughput of the network increases [7]. HARQ schemes are classified into two categories: HARQ Type-I and HARQ Type-II. HARQ Type-I uses a codeword designed for simultaneous correction and detection. Hence, in this case the CAN frame contains the error-detection and correction parity bits. In this method the receiver attempts to correct the error upon reception of a frame and then checks for further error. If the packet does not have any errors after the FEC decoding, the

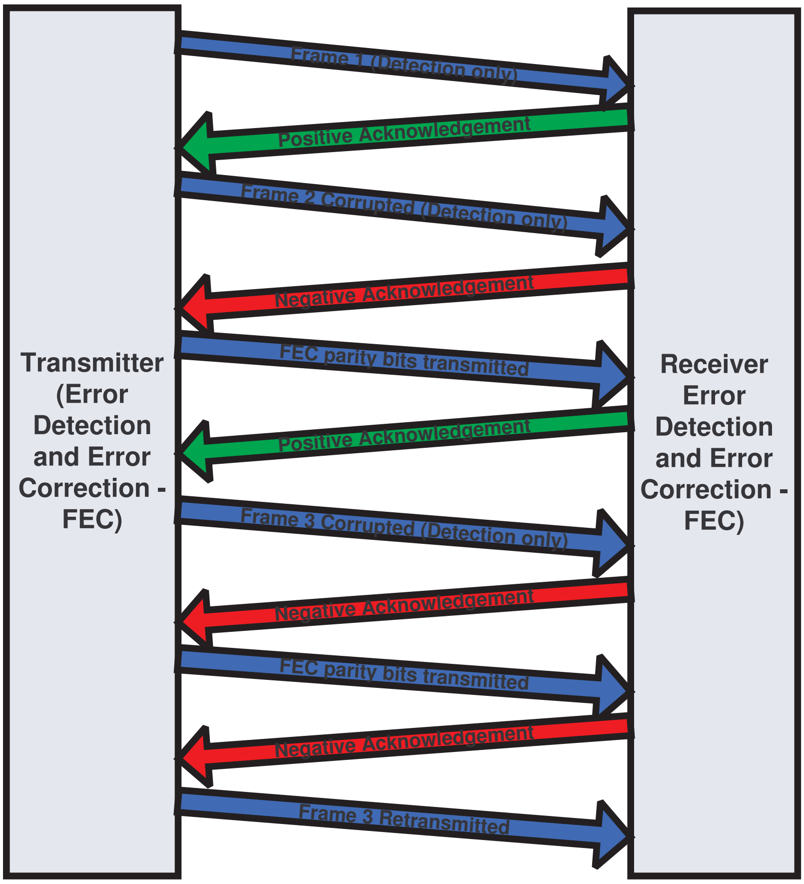

Figure 2.9 HARQ Type-II Protocol with the Data Flow and Acknowledgements

frame is accepted, removing the additional parity bits. If an error is detected in spite of the FEC decoding, then the receiver requests retransmission. In this method the bit overhead per packet is increased because the packet has to contain both the error-detection and the error-correction parity bits. Hence, when the channel error rate is low this method induces more overhead than it's corresponding ARQ scheme. But, when the error rate increases, the ARQ scheme performance reduces drastically thereby proving the importance of this method [7]. The flow of the data frames along with their acknowledgements for HARQ Type-I is shown in Fig. 2.8.

2.3.3. Hybrid Automatic Repeat Request(HARQ) Type-II. Transmitting all the frames with error-correction parity bits might lead to higher bit overhead in a low channel error rate case. Hence, a new method has to be used in order to reduce the bit overhead added by the extra parity bits. A solution to this problem is to transmit the error-correction parity bits once the error is detected. This method is called the HARQ Type-II [12]. Simply put, in this technique all the packets are encoded using error-correction codes such as the Reed-Solomon(R-S) or the Bose Chaudhuri Hocquenghem (BCH) codes. But, the packet with error-detection parity bits alone is sent across the channel without the error-correction parity bits. The receiver node computes the syndrome of the received frame and checks for errors using the CRC method. If no error is detected, the frame is accepted. If an error is detected, then the receiver asks the transmitter to transmit the error-correction parity bits. The receiver saves the corrupted frame until it receives the correction parity bits. Once it receives them, it decodes the frame using the error-correction technique. Upon decoding, the receiver checks for errors once again using the CRC technique and if the error still exists, then the receiver asks for the retransmission of the entire frame. Hence, when the probability of occurrence of error is low, the HARQ Type-II method reduces the bit overhead because every frame need not have the parity bits transmitted, proving that it is more efficient to reduce the bit overhead introduced by the HARQ Type-I method. The flow of the data frames along with their acknowledgements for HARQ Type-II is shown in Fig. 2.9.

3. ERROR HANDLING METHODS IN CAN

Error handling has already been built in the CAN bus and t is an important aspect of evaluating the the CAN bus performance as a standard communication protocol in many applications [13]. The error handling schemes used by the CAN bus aim at detecting the errors in the frames transmitted over the bus and requests the transmitter retransmit the erroneous frames. Any number of nodes can be connected to the CAN bus depending on the application. Every node connected to the CAN bus has the capability of detecting an error within the frame. The node that detected the error raises an error flag, halting the traffic on the bus and all the other nodes connected to the bus discard the frame [13].

At the transmitter end, the frame is considered valid if there is no error until the End of Frame. If a corruption is detected, retransmission follows automatically, according to priority. The frame is considered valid from the receiver point of view if the frame has no errors until the next to last bit of the End of Frame [1].

3.1. ERROR DETECTION MECHANISMS USED BY CAN

CAN protocol uses five different ways error detecting techniques defined in its protocol with two of them working at the bit level and other three working at the message level. The different mechanisms used by CAN are listed below [13].

1. Bit Monitoring
2. Bit Stuffing
3. Frame Check or Form Error
4. Acknowledgement Check
5. Cyclic Redundancy Check

3.1.1. Bit Monitoring. The transmitting unit that sends a bit on the bus also monitors the bus continuously. A bit error is detected when the bit value monitored on the bus is different from the bit value that is transmitted [1].

3.1.2. Bit Stuffing. Frame segments including the Start of Frame, the Arbitration Field, the Control Field, the Data Field, and the CRC Sequence are all coded using a method called Bit Stuffing. Whenever a CAN transmitter detects five consecutive identical bits in the bit stream to be transmitted it automatically inserts a complementary bit at the sixth bit position in the actual transmitted bit stream. Hence, a stuff error is raised whenever the 6th bit of equal level is detected in the message field [1].

3.1.3. Frame Check or Form Error. Several fields of the CAN frame have fixed formats. This implies that the CAN standard defines exactly what levels have to occur at what positions. Hence, when a CAN node detects a wrong bit level at one of these fixed fields, it raises a Form Error flag [13].

3.1.4. Acknowledgement Check. An acknowledgement error is detected by a transmitter whenever it does not monitor a 'dominant' bit during the Acknowledge Slot [1].

3.1.5. Cyclic Redundancy Check. Cyclic codes, which are widely used for error-detection, are typically called the Cyclic Redundancy Check (CRC) codes. These are very popular and most commonly used error-detection codes in many communication systems [14]. CRC codes generally use a generator polynomial to generate the sequence called the CRC sequence. A polynomial of degree equal to 15 is used by the CAN bus to detect errors that occur in data transmission.

With this CRC, the transmitter calculates a check sum for the CRC sequence from the Start of Frame bit until the end of Data Field [15]. A CRC error has to be detected if the result calculated by the receiver is not the same as that received in the CRC sequence. In this case the receiver discards the message and transmits negative acknowledgement to the transmitter. The CRC checksum used in the CAN bus is for error-detection, but not used for error-correction. The hamming distance for this particular CRC code is 6. With this hamming distance it is possible to detect up to 5 single bit errors that are randomly scattered about the message or burst errors up to a length of 15 bits [15].

Every CAN node will detect errors within each message using the methods defined above. The discovering node will raise an error flag, halting the bus traffic, and all the others nodes read the error flag and discard the current frame [13].

3.2. FAULT CONFINEMENT

With respect to fault confinement, the CAN bus nodes can be in any of the three states: Error Active, Error Passive or Bus Off. An Error Active unit can normally take part in bus communication and can send an active error flag when an error has been detected on the bus. This halts the communication on the bus and all nodes discard the message immediately when the active error flag is raised. An Error Passive unit takes part in the normal communication of the bus and can only send a Passive Error flag but not an Active Error flag. The Bus Off unit is not allowed to have any influence on the bus [1].

Two counters are used to determine the state of every CAN node: a Transmit Error Counter (TEC) and a Receive Error Counter (REC). Every node starts off in the Error Active mode. The node updates the counter every time an error is encountered. The Transmit Error Counter is increased by eight when the transmitter identifies an error in the transmitted frame.

The Receive Error Counter is incremented by one when any receiving node identifies an error in the frame. If either of these counters rises above 127, the node will move to the Error Passive state. If the Transmit Error Counter reaches a value of 255, the node moves to the Bus Off mode [13].

The rules for incrementing the error counters are complex and unique for both transmit and receive error flags. For example, imagine that a node 'A' connected to the CAN bus does not function properly. For every transmit error, the TEC of the node 'A' is incremented by 8 and the node goes into the Error Passive mode when the count reaches 127 (i.e., after 16 attempts). Once the node is in Error Passive mode it only transmits Passive Error flags, which do not interfere with the traffic on the bus. The counter on node 'A' keeps increasing and when the count reaches 255 the node goes into the Bus Off mode. This state is reached by node 'A' lot before the count of the other nodes reach 127. Henceforth, the count of the receive nodes is decreased by 1 every time a correct frame receive [13].

The CAN controller's method of automatically retransmitting messages when errors occur can be annoying at times [13]. The above fault confinement makes sure that the CAN bus communication is not affected by a malfunctioning node. Low speed CAN applications have high performance in maintaining the speeds required for the data transfer. For high speed CAN these errors have to be reduced because the number of retransmissions reduce the performance of the CAN bus.

4. PROPOSED HARQ METHODS AND SIMULATIONS

As discussed in the previous section, the methods used by the CAN bus to handle errors do not effectively maintain the data rates at the required level in severe interference environments or when data rates increase beyond 1Mbps. Hence, a novel protocol is proposed to improve the performance of the CAN bus for use in applications that demand high data rates. The proposed idea is to implement error-correction codes along to go with the error-detection codes that are currently being used.

4.1. SOURCES OF ERROR: EMI EFFECTS, NOISES, ETC

The first and the foremost step in designing the error-correction code for implementation in the CAN bus is to determine the kinds of errors and their sources. Once the environment in which the CAN bus operates is known, the job is cut down to select a particular type

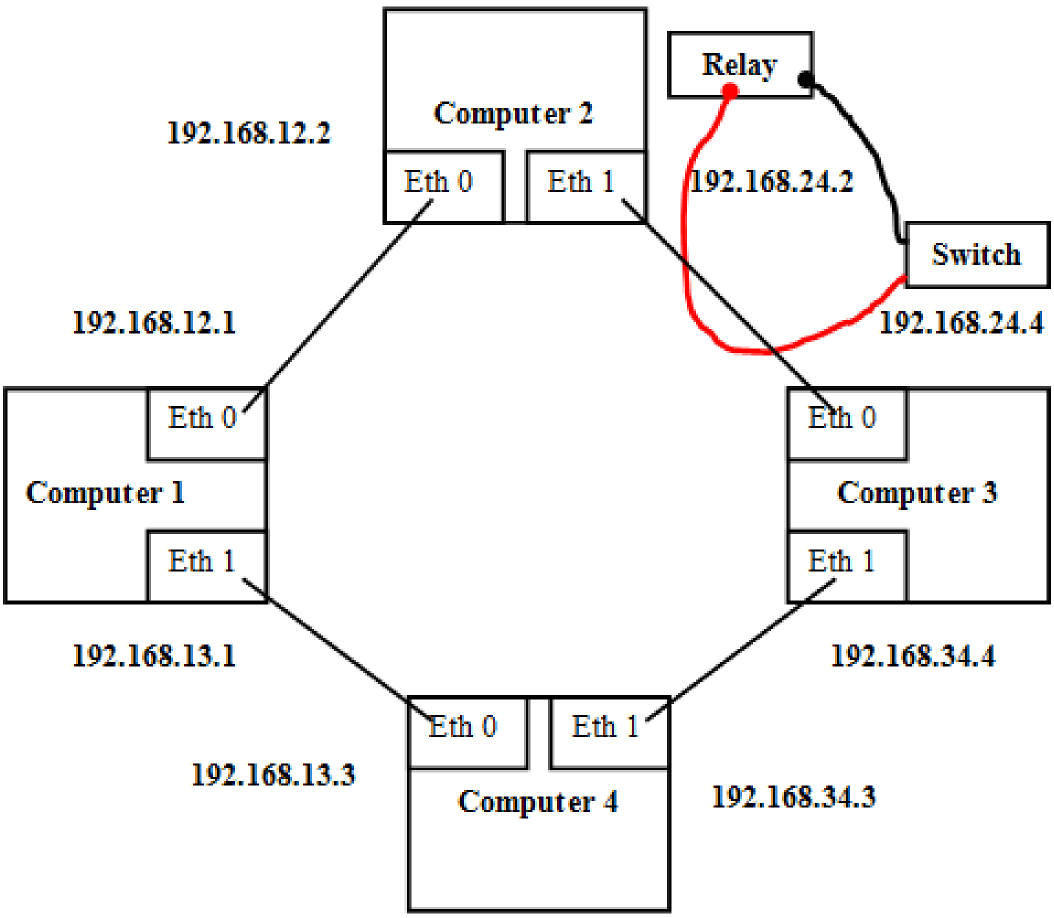

Figure 4.1 Lab Arrangement made in G23 with Four Computers

of error-correction coding technique to combat these errors. A small setup with four connected computers with LINUX operating systems was set up in the G23 laboratory, Electrical Engineering department, University of Missouri-Rolla, to check the impact of EMI on data transfer. This setup used Ethernet crossover cables to transfer data between the computers. Figure 4.1 shows the lab's setup, with the relay and the switch connected to each other and tangled with the connecting wire.

A command **ping -f** can be used to determine whether the packets are transmitted without error. Once the connections were made, as shown in the Fig. 4.1, IP addresses were assigned to each computer's network cards. Each computer was fitted with two Network Interface Cards (NIC). Then computer 4 was pinged from computer 2.

Ping -f 192.168.24.4-Pinging Back and Forth

This command induces a dot once the transmission is made and removes it once the packet is transmitted back without any error. Hence, an error can be seen when a dot is induced on the screen but not removed. The greater the number of dots at the end of the test, imply more the number of lost packets. Observations were made using this test bench in two cases:

1. When no EMI is introduced
2. When EMI is introduced by the switch and the magnetic relay to see its affect on data transfer.

The results of the experiment showed that when EMI was introduced by switching, loss of packets occured. A dot appears on the screen showing the transmission, but it is not removed because the received packet is not the same as the transmitted packet. Hence, the conclusion drawn from this experiment is that EMI is a major source of error. This information reveals that EMI is a major source of error in the CAN bus, but the severity of the errors that it creates is unknown. Some other means of studying how many bits are corrupted every time EMI is introduced needs to be developed. Are errors scattered randomly all over the frame or do they occur in a burst?

Another experiment was conducted to study the types of errors created by EMI. Fei Ren, a graduate student in Electrical Engineering at the University of Missouri-Rolla set up a test bench in the laboratory to study the CAN environment. Two 8051 micro-controllers, which were capable of emulating the CAN network between them, were programmed in such a way that one of them functioned as a transmitter and the other as a receiver. The switch and the magnetic relay provided by Caterpillar were connected and both these wires were matted together to create an Electro-Magnetic Interference (EMI) effect on the CAN bus.

The receiver was set to ‘receive ready’ state and the transmitter continuously transmitted frames of data, which were accepted by the receiver if they were error free. Whenever the receiver detects an error, it sends an Automatic Repeat reQuest (ARQ) to the transmitter and the transmitter then retransmits the frame. In a normal CAN (without EMI) errors rarely occur and almost all the frames are received without retransmission. When EMI was introduced the frames were corrupted, thereby introducing errors. Hence, it was important to measure the EMI to determine the kind of coding scheme to use.

A digital oscilloscope was connected to the CAN bus receiver, through which the CAN frames were observed. CAN frames were recorded with and without EMI and the data was analyzed in detail. Initial figures showed that the frames were corrupted in single instances. The errors were random and the EMI could only corrupt one or two bits in the entire frame. In some instances the errors were spread over more than 2 bits. Further investigation into the EMI and its effect on the CAN bus showed that the types of errors assumed in the first case were not correct. The method Fei used to view the waveforms on the oscilloscope in his initial study was proven to be wrong. In consultation with Dr. Thomas P. VanDoren, Emeritus

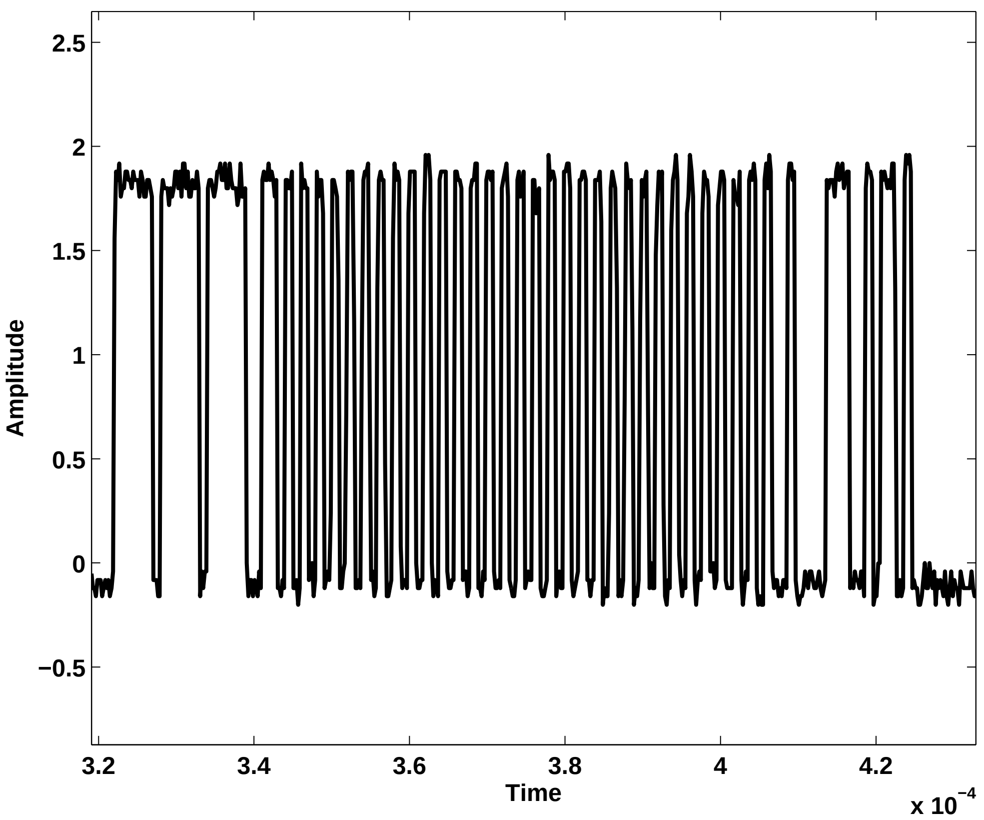

Figure 4.2 CAN Frame without EMI

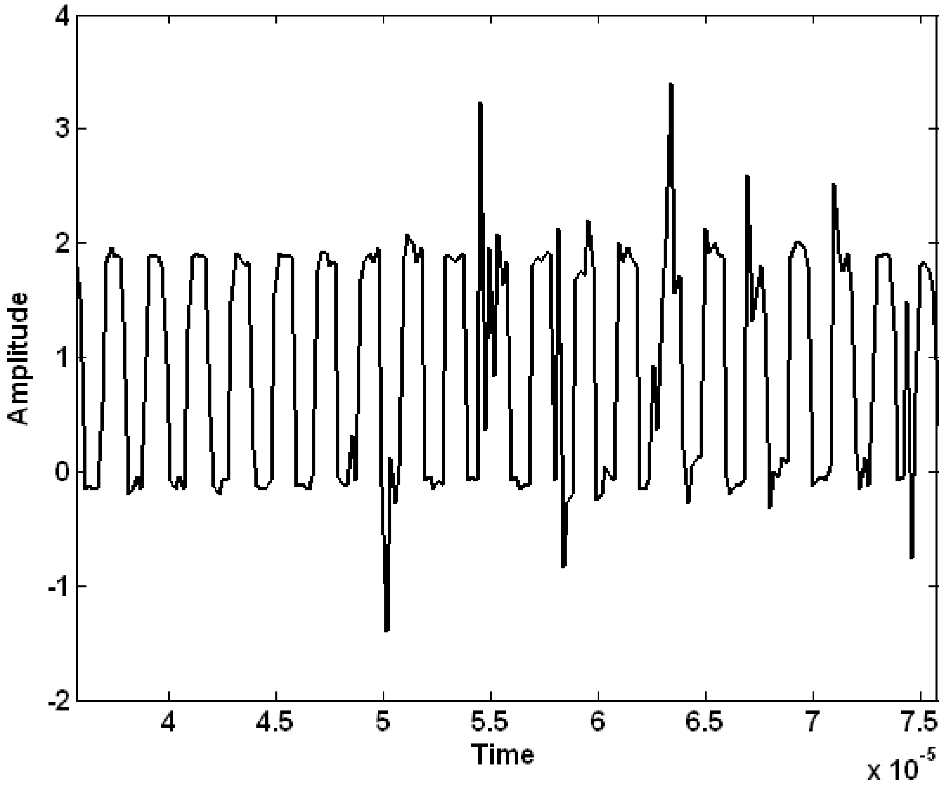

Figure 4.3 Part of a CAN Frame with EMI Corruption

Professor at the University of Missouri-Rolla, the waveforms were properly observed. The new figures from the oscilloscope showed that the errors due to EMI in the CAN bus are not random and single-bit, but occurred in bursts. The length of the burst varied from 3 to 8 bits.

Figures 4.2 and 4.3 shows the CAN frame without and with EMI respectively. The spikes shown in Fig. 4.3 are the result of EMI acting on the CAN bus. Careful observation of the signal shows that EMI adversely affects about $3-5$ bits of the CAN data frame. The CAN bus receiver or any node on the CAN bus will discard frames with even a single bit error. However, with error-correction schemes these issues can be resolved.

4.2. DESIGN AND IMPLEMENTATION OF CYCLIC CODES ON CAN

When looking at the CAN frame affected by EMI from the initial study by Fei, it first appears that any error-correction technique capable of correcting random errors would be sufficient to combat and correct errors. Cyclic codes were used to correct the errors because they are simple to implement. Two different cyclic code designs (Cyclic117 and Cyclic60) were applied and the performances were compared with respect to the error correction capability and the computational complexity. Depending on the size of the CAN frame, the values of n

and k were selected such that every bit in the given frame is encoded. A compromise had to be made in selecting the values of n and k, because the generator polynomial does not exist for all combinations of n and k. Hence, the values were selected as close to the frame length as possible to include all the bits to be encoded. Two different pairs of (n, k) were selected for the simulations and their performance is presented in detail. As mentioned earlier, the number of bits in the CAN frame is assumed to be 128 in all simulations. The performance analysis presented in this report holds good in the case of the 126 bit frame length. Hereafter, when the phrase CAN frame length is used, it implies 128 bits.

4.2.1. Case 1: $n = 117, k = 102$. The values of $(n, k) = (117, 102)$ were used for the first simulation (Cyclic117). The number of bits to be encoded was 103 and the length of the codeword encoded should be 118. These two values of (n, k) do not have a generator polynomial. Hence, when $k = 102$ is used, one bit is neglected in the encoding process and appended after the encoding is complete. In other words, the first bit was not encoded and all the bits starting from second to the 103rd were encoded to a codeword of 117 bits.

Depending on these values of n and k, the minimum distance between the codewords was calculated to be 6. The minimum distance between the codewords is a measure of the number of errors that can be corrected using this error-correction technique. The number of errors that can be corrected by any error-correction technique is given by

$$t = \lfloor \frac{d_{min} - 1}{2} \rfloor \tag{4.1}$$

where d_{min} is the minimum distance between the code words.

Equation 4.1 shows that this code is capable of correcting 2 bit errors occurring anywhere in the received frame. The CAN frame with EMI from the initial study showed only two errors in some cases, so this error-correction scheme seemed to be sufficient to combat the affect of EMI on the CAN bus. This scheme had 100% probability of correcting all the single bit and double bit errors. The only drawback of this scheme is its computational complexity. Computational complexity is directly proportional to the value of n and the difference between the values of n and k because the size of the syndrome table is $n * 2^{(n-k)}$. Hence, calculating the syndrome table and storing it is a huge task in terms of computational complexity and memory space. The number of elements in the syndrome table for this coding scheme is $117 * 2^{15} = 3,833,856$ elements.

Every time an error frame was received the scheme compared the received frame with all the rows of possible codewords and the codeword closest to the received frame was finalized as the original transmitted frame. Hence, this method provided good performance in terms of error correction capability for the kind of errors that the EMI created but the computational

Table 4.1 Comparison of the Two Cyclic Code Methods $(n, k) = (117, 102)$ and $(n, k) = (60, 53)$

Cyclic117 $(n, k) = (117, 102)$	Cyclic60 $(n, k) = (60, 53)$
1. The values of (n, k) are very close to the required value but do not encode the whole CAN frame.	1. The values of (n, k) encode the whole CAN frame with slight modification in the encoding process.
2. The minimum distance is 6 and hence the number of errors that can be corrected is 2.	2. The minimum distance is 4 and hence the number of errors that can be corrected is 1.
3. The computational complexity of this method is very high and the syndrome table is very huge	3. The computational complexity is less than the first method and the syndrome table is 1/500 times of the first case.

complexity is a disadvantage of this method. So, a simpler method with fewer computations was proposed by breaking the CAN frame into two sub-frames during the encoding process.

4.2.2. Case 2: $n = 60, k = 53$. The second approach had smaller n and k values equal to 60 and 53 (Cyclic60). In this approach the given frame is divided into two different sub-frames, each of which is encoded individually. The encoded frames are appended together and transmitted to get the complete CAN frame. Three bits were appended to the 103 bits to be encoded, then the frame was broken down into two sub-frames of 53 bits each and encoded to produce a codeword of 120 bit length. The three extra bits appended for the sake of encoding were removed and the frame is transmitted over the CAN channel. At the receiver the three bits were appended back to decode the frames. The final decoded sequence is compared with the transmitted sequence to check for the number of errors.

The minimum distance in this case is equal to 4. Hence, the number of bits that can be corrected is equal to 1. Each sub-frame can correct 1 error, so 2 errors can be corrected in the whole CAN frame, provided only one error occurs in each of the sub-frames. The performance of the latter method is not as good as the former, but in terms of computational complexity the number of elements in the syndrome table in the latter case is almost 1/500 of the former. The number of elements in the syndrome table in this case is $7,680$. The two methods are compared in Table 4.1.

One major disadvantage of both the methods is that there is no guarantee of an error free reception. When the frames have a burst of 2 bits or greater, the original frames can never be recovered from the corrupted frames. In this case the two methods discussed would fail badly because they are only capable of correcting single bit errors (not burst errors). In this situation there is no means by which the receiver can request retransmission of the frames. Hence, a new method had to be proposed to tackle these burst errors, ensuring error free communication while reducing the number of retransmissions. This situation motivates the introduction of Hybrid Automatic Repeat reQuest (HARQ).

4.3. DESIGN AND IMPLEMENTATION OF PROPOSED HARQ

Two variants of HARQ Type-I and two variants of HARQ Type-II are proposed to improve the performance of the CAN bus. As mentioned in the previous section, error-correction coding is the technique used to correct the errors in the received data. It introduces systematic redundancy in the transmitting data in order to combat the errors caused by various noise sources such as EMI. Block codes are a popular category of error-correction schemes. Block codes are defined by (n, k) where k is the number of input data bits and n is the number of bits in the encoded frame. These codes are simple to implement and have lower computational complexity than the other competing coding schemes. Reed-Solomon (R-S) codes are non-binary cyclic codes with symbols made up of m-bit sequences where m is any positive integer having a value greater than 2 [10]. Reed-Solomon codes are a special type of block codes well suited to combat burst type errors with less computation time as the computations are done at the symbol level. The errors induced by EMI being burst type can be easily corrected by the proposed scheme. Depending on the error-correction performance needed, values of (n, k) are selected. The number of symbols that can be corrected will always be t given by Equation 2.8, irrespective of the number of bits corrupted in each symbol. This is a major advantage of using R-S codes.

Figure 4.4 shows the flowchart of the algorithm used to simulate the two HARQ Type-I methods (HARQ Type-I R-S20/CRC and R-S20/Cyclic) on the CAN bus. Both the methods use R-S codes to combat the burst errors, but the R-S20/CRC method uses the CRC error-detection technique while the R-S20/Cyclic replaces the CRC bits with cyclic codes. Both the methods are shown in the flowchart. The arrows with solid lines show the flow of the HARQ Type-I R-S20/CRC method. The arrows shown by dashed lines represent the flow of the HARQ Type-I R-S20/Cyclic method. Random data is generated and each time 64 bits are selected and a header is added to the frame. In the HARQ Type-I R-S20/CRC the frame is now encoded with CRC error-detection technique, while for the HARQ Type-I R-S20/Cyclic the frame is encoded with cyclic codes. The cyclic code used in this method is Cyclic60 discussed previously. After the parity bits are appended the tail is added to bring the frame length to 128 bits. This frame is then encoded using the R-S20 encoder. At the receiver the frame is first decoded using the R-S20 decoder. In case of the R-S20/CRC method the

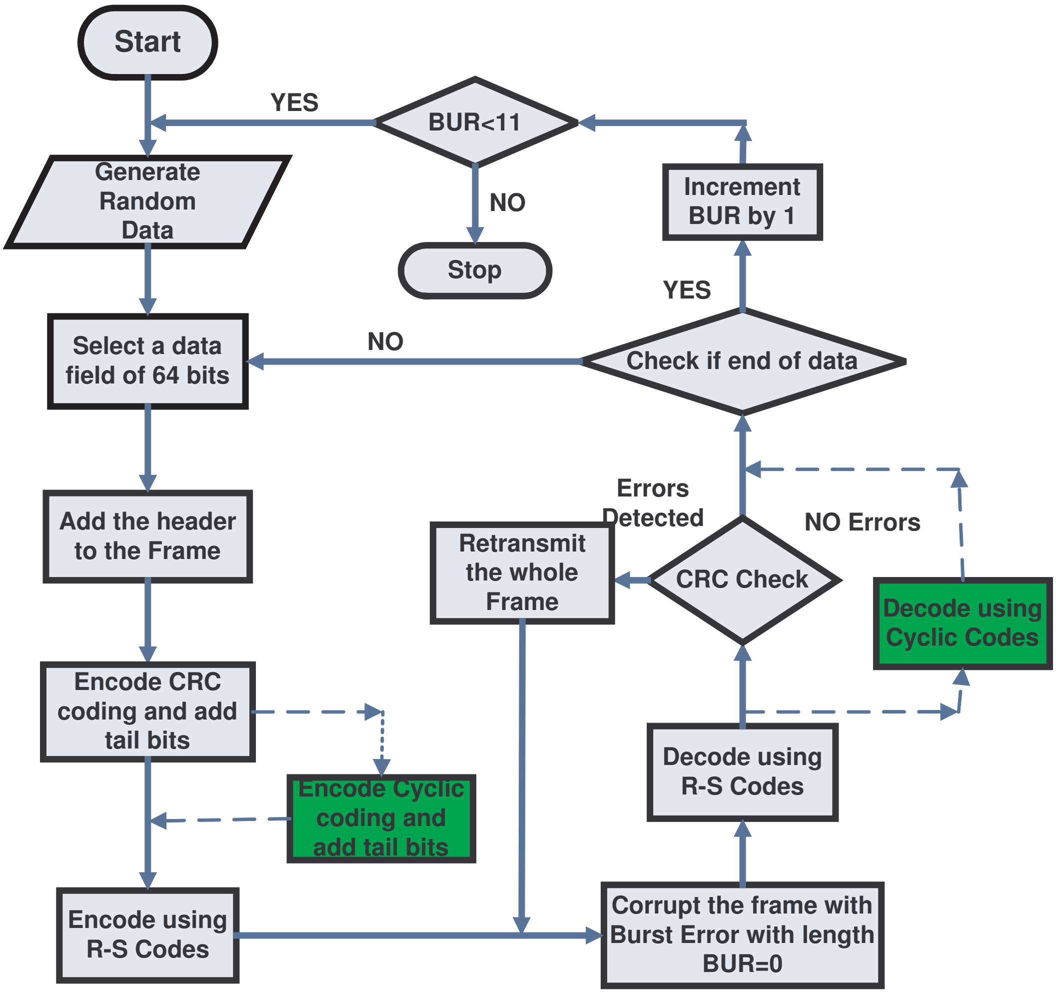

Figure 4.4 Flowchart Depicting Implementation of HARQ Type-I R-S20/CRC and HARQ Type-I R-S20/Cyclic for Randomly Generated Data and Burst Length Increasing from 0 to 10

frame is checked for errors and a retransmission is requested if errors are detected. In the R-S20/Cyclic method the R-S20 decoded frame is then decoded using cyclic codes to obtain the final decoded bit sequence. In this case the final bit sequence has to be accepted even if errors exist because this method has no mechanism for error-detection. The parameters and the methods mentioned in this section were simulated and compared to find out the advantages and disadvantages of each of the methods. These comparisons are shown clearly in the results sections with necessary plots to support the observations. The parameters for the R-S encoder used in these methods were determined using the predefined rules given by

$$m = 5; n = 2^m - 1 \Rightarrow n = 31; k = 27 \Rightarrow t = \frac{n-k}{2} \Rightarrow t = 2 \tag{4.2}$$

The CAN frame length is equal to 128 bits, so the coding scheme has to encode all 128 bits. The value chosen of $m = 5$ gives the value of $n = 31$. Choosing $k = 27$ enables the encoding of all the bits in the CAN frame. Zeros are appended at the end of the CAN frame to make the length equal to 135 bits. Once the frame is encoded the appended zeros are removed, the CAN frame is transmitted along with the error-correction parity bits.

The CAN frame length increases from 128 to 148 bits with the 20 additional error-correction bits. In order to check the performance of the proposed HARQ scheme, 8000 frames are encoded with R-S codes and are transmitted over a channel having SNR=20dB. The SNR value for the CAN bus is expected to be high because its a wired network. The only interference in the network is the Electro-Magnetic Interference (EMI) created by the external sources. The analysis and study of the affect of EMI on the CAN bus showed that the errors are burst type. Hence, different length bursts were created to corrupt the CAN frame and study the performance of the HARQ scheme. One other parameter considered was the probability of the occurrence of the burst errors in the CAN communication. The probability of a burst error depends on the environment in which the CAN network is being operated. If the inductive load present in that particular environment is high, then the probability of occurrence of burst error is high. Hence, all the simulations are presented with the different probabilities of burst error occurrence to give a clear picture of the HARQ scheme's performance in various environments. The length of the burst has been varied from 0 to 10 bits. The first 8000 frames were transmitted with a 0 bit burst error, then the burst length was increased in steps of 1 for every 8000 frames to reach a maximum burst length equal to 10 bits. The position of the burst in the frame is random and can occur at any position.

The erroneous frames reach the receiver, zeros are appended back to make the length equal 155 bits. The corrupted frames are then decoded using the R-S decoder. The zeros are removed from the decoder output to obtain 128 bit frame back. The decoded CAN frame is then checked for errors using the CRC bits commonly used by the CAN. If the receiver detects an error on the CAN frame, it asks for a retransmission and the transmitter retransmits the entire frame. If no errors are detected then the receiver accepts the frame. Simulation graphs in the results section give a picture about the CAN performance improvement via R-S codes. Comparison between the performance of HARQ Type-I R-S20/CRC and HARQ Type-I R-S20/Cyclic codes is also given in the results section.

HARQ Type-II is another protocol that can be used for CAN bus communication. The error-correction capabilities of HARQ Type-I and HARQ Type-II are similar since the same error-correction scheme is used. The difference in performance is in terms of the bit overhead added to the network. Figure 4.5 shows the flowchart of the two variants of the HARQ Type-II method. HARQ Type-II R-S20/CRC uses the CRC for error-detection and 20 R-S parity

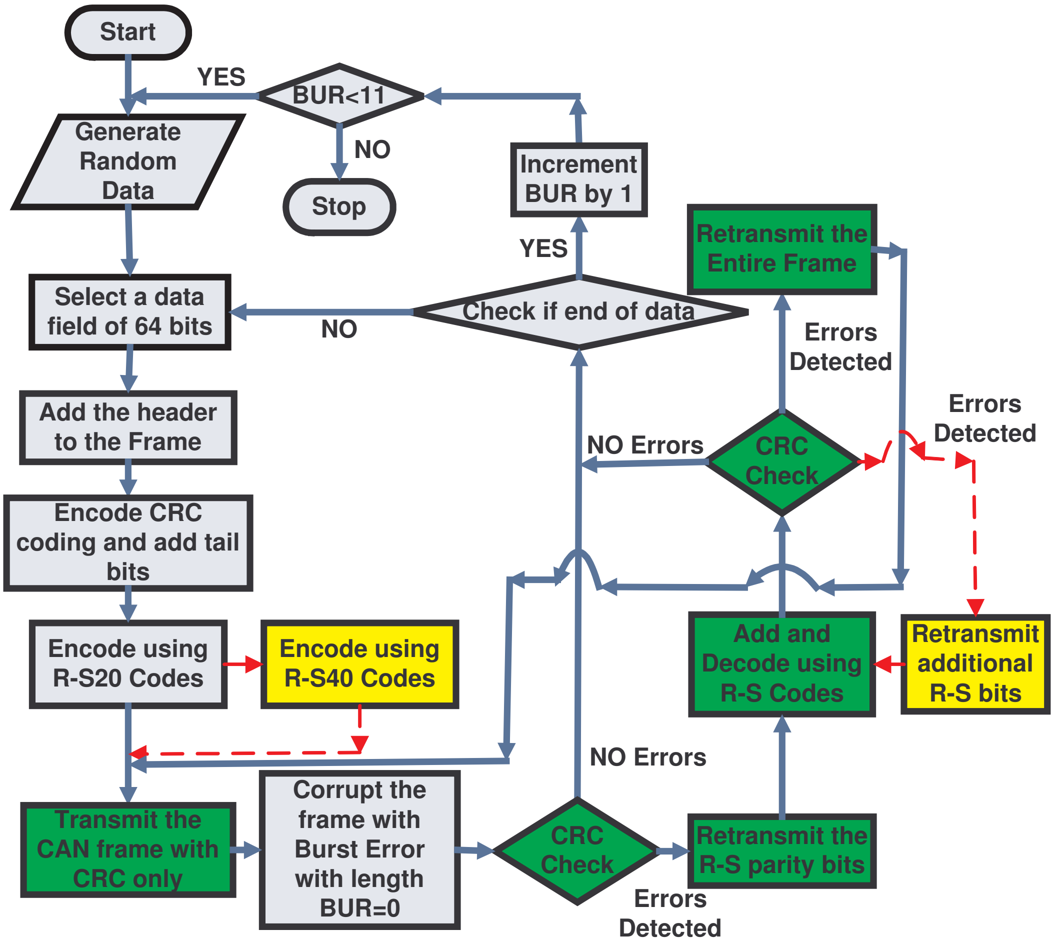

Figure 4.5 Flowchart Depicting Implementation of HARQ Type-II R-S20/CRC for Randomly Generated Data and Burst Length Increasing from 0 to 10

bits but in this method sends the original CAN frame with ARQ parity bits only without the R-S20 parity bits. In case of error-detection, the receiver requests for error-correction parity bits and the transmitter transmits the parity bits. The bits are appended and checked if the frame can be corrected. In case the receiver cannot decode the uncorrupted frame then the whole frame is requested. Here, as the retransmission involves only the parity bits instead of the whole frame, the bit overhead is reduced.

In some cases, when either the HARQ Type-I R-S20/CRC or the HARQ Type-II R-S20/CRC method might be unable to correct the errors in the corrupted frame. In these cases, the whole frame along with the error-correction parity bits have to be retransmitted in the former method and the frame with detection parity bits alone have to be retransmitted in the latter method. A new method of transmitting additional error-correction parity bits can be used instead of transmitting the entire frame. This method is termed as HARQ Type-II R-S40/CRC. Hence, instead of retransmitting the entire frame, in the HARQ Type-II

SOF 1bit	Arbitration 12 bits	Control 6 bits	Data Field 64 BITS	CRC 15 bits	ACK 2 bits	EOF 7 bits

Figure 4.6 Standard CAN Frame with Individual Fields Shown

R-S40/CRC the frame can be encoded with additional error-correction bits by encoding the CAN frame twice. Once with the R-S20 encoder and then with R-S40 encoder. When the first retransmission parity bits (R-S20) fail to correct the erroneous frame, the second set of parity bits (R-S40) can be transmitted to correct the errors. The R-S40 codes have twice the error-correction capability of the R-S20 codes. Hence, most of the errors that occur in the CAN bus communication could be corrected with this scheme. This reduces the bit overhead added by the earlier schemes providing additional error-correction capability.

The comparison of the percentage bit overhead for ARQ, HARQ Type-I R-S20/CRC, HARQ Type-II R-S20/CRC and HARQ Type-II R-S40/CRC is discussed in detail in the results section with necessary graphs to support the results.

4.4. COMPARISON OF THE ORIGINAL AND PROPOSED CAN FRAMES

The current CAN data frame structure is briefly repeated here to illustrate the modification of the frame structure for implementing the HARQ scheme in the CAN bus. Two versions of the CAN frames are currently in use: the standard version and the extended version [1], shown in Fig. 2.2 and Fig. 2.3. These figures are redrawn here to provide a clear comparison. The number of bits in each of the fields for both versions is listed below:

- Standard Frame Structure: Header: 19 bits; Data field: 64 bits; CRC field: 15 bits; Tail:10 bits.
- Extended Frame Structure: Header: 39 bits; Data field: 64 bits; CRC field: 15 bits; Tail:10 bits.

SOF 1bit	Arbitration 30 bits	Control 6 bits	Data Field 64 BITS	CRC 15 bits	ACK 2 bits	EOF 7 bits

Figure 4.7 Extended CAN Frame with Individual Fields Shown

Figure 4.8 Extended CAN Frame with Individual Fields and RS-Bits Shown. Frame used for HARQ Type-I R-S20/CRC

Figure 4.9 Extended CAN Frame with Individual Fields, Cyclic Parity Bits and R-S Bits Shown. Frame Used for HARQ Type-I R-S20/Cyclic

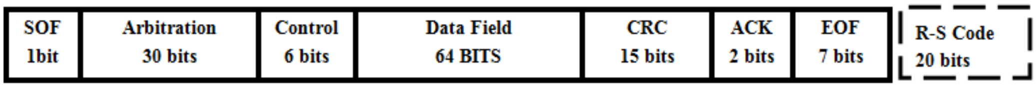

Figure 4.10 Extended CAN Frame with Individual Fields and RS-Bits Shown. Frame used for HARQ Type-II R-S20/CRC, dotted part is the R-S parity bits retransmitted upon request

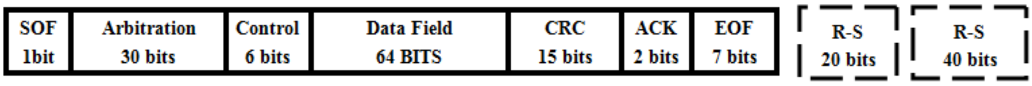

Figure 4.11 Extended CAN Frame with Individual Fields and RS-Bits Shown. Frame used for HARQ Type-II R-S40/CRC, dotted part is the R-S parity bits retransmitted upon request

In the HARQ Type-I R-S20/CRC the CRC method is supplemented by the R-S encoder. In this method the CAN frame is appended by error-correction parity bits at the end of the frame with the rest of the frame left unchanged. Figure 4.8 shows the extended CAN frame encoded with R-S codes for HARQ Type-I R-S20/CRC method. For the HARQ Type-I R-S20/Cyclic the CRC bits are replaced with cyclic parity bits and the rest of the frame is same as the HARQ Type-I R-S20/CRC. Figure 4.9 shows the frame structure of the HARQ Type-I R-S20/Cyclic. For the HARQ Type-II R-S20/CRC the 20 R-S parity bits part of the frame can be detached from the frame. Figure 4.10 shows it as a dotted line. This method makes the

HARQ scheme more compatible with current CAN bus as it requires no change to the current CAN frame but an additional frame has to be introduced to transmit the R-S parity bits and all the CAN nodes should be incorporated with error-correction techniques. For HARQ Type-II R-S40/CRC the 20 R-S bits and 40 R-S bits are detached from the frame and shown as dotted line in Fig. 4.11. Each one of the two R-S parity sub-frames are transmitted upon error-detection and receiver request.

5. RESULTS AND OBSERVATIONS

MATLAB was used to check the performance of the HARQ schemes presented in this thesis, in terms of error-correction capability, percentage reduction in the number of retransmissions and the percentage bit overhead added by each of the methods. Results and observations are divided into two categories with the first one giving a thorough analysis of the each method's performance in terms of error-correction capability and the second one comparing the methods in terms of the bit overhead added to the network.

5.1. ERROR HANDLING COMPARISON OF PROPOSED METHODS AND CONVENTIONAL CAN

The first method proposed replaces the CRC bits in the CAN frame with the cyclic error-correction bits (Cyclic60). The performance of the proposed method as compared to the conventional CAN is illustrated in Fig. 5.1. This figure shows the performance of the Cyclic60 method in terms of the percentage of correct frames received. For this simulation the $SNR = 25dB$ and the probability of the CAN frame getting corrupted varies from 0.1 to 0.5. All the single bit errors are corrected in this method, but burst errors ranging from 2 to 10 bits are not. If the receiver fails to correct the errors there is no way the receiver can get back the uncorrupted frame because there is no feedback mechanism in this method. Hence, this method can only be used in environments where the impact of EMI is limited to single bit error corruptions in single instances. In such an environment this method corrects all the frames without any need for frame retransmissions.

The second method proposed in this thesis is to use the HARQ scheme with Reed-Solomon Codes in conjunction with the CRC error-detection technique used by the CAN bus (HARQ Type-I R-S20/CRC). The SNR value for this simulation is assumed to be 20dB. As mentioned in the previous section, the SNR value is assumed to be high and the burst errors are added over the SNR value. The probability of burst error occurrence signifies the different environments in which CAN bus can be operated. Figure 5.2 compares the performance of the CAN but without FEC and with the HARQ Type-I R-S20/CRC method in terms of the percentage of frames correctly received. The R-S codes were able to correct all errors in the case of burst lengths smaller than or equal to 6 bits. A burst of length of 6 bits always fits into two R-S symbols in the current design and, hence, may be corrected without any need for frame retransmissions. As the burst length exceeds 6, the ability of the R-S codes to correct these errors reduces. Performance of the R-S codes is reduced for burst lengths of 7 bits or higher because the burst in this case can span over more than two R-S symbols. For example,

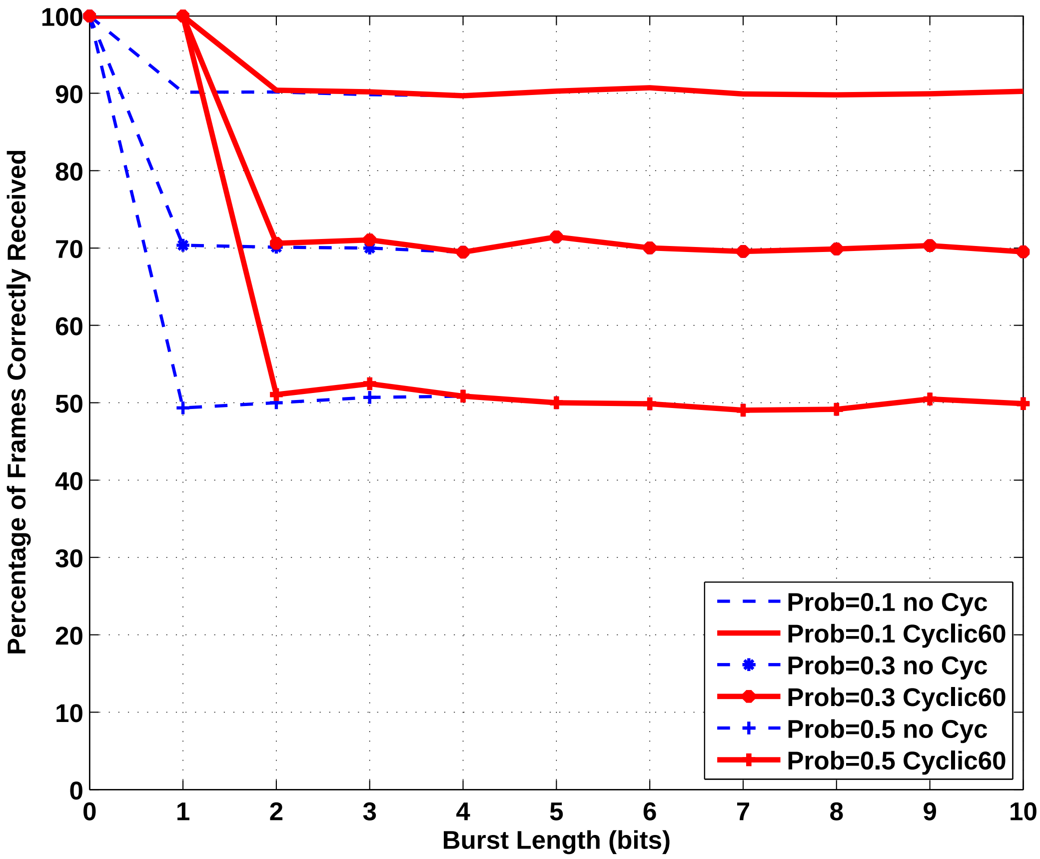

Figure 5.1 Comparison of the Percentage of Correct Frames Received with and without Cyclic60 codes. $SNR = 25dB$, $n = 60$, $k = 53$.

when the burst length is equal to 7, the burst can be distributed over three symbols (1, 5, and 1). In such cases, the R-S codes cannot correct the frame. The probability of correcting errors decreases with the burst length because in more and more cases the burst corrupts three symbols as the burst length increases. The R-S codes will be unable to correct the frame in such cases. Hence, for burst lengths greater than or equal to 7 bits, the percentage of correct frames received decreases. When the probability of the CAN frame getting corrupted is 40%, the HARQ Type-I R-S20/CRC scheme requires 30% fewer frame retransmissions for burst lengths of 8 bits and 11% fewer for burst lengths of 10 bits. For burst lengths smaller than 7 bits all the frame retransmissions can be avoided with the percentage of correct frames received being at 100%. The probability of burst error occurrence do not impact the R-S codes performance for burst lengths of less than 7 bits. Hence, this method can be implemented in different environments with different intensities of EMI impacts.

The performance of the third method HARQ Type-I R-S20/Cyclic where the CRC bits were replaced with the cyclic codes has been simulated. Figure 5.3 shows the graph comparing

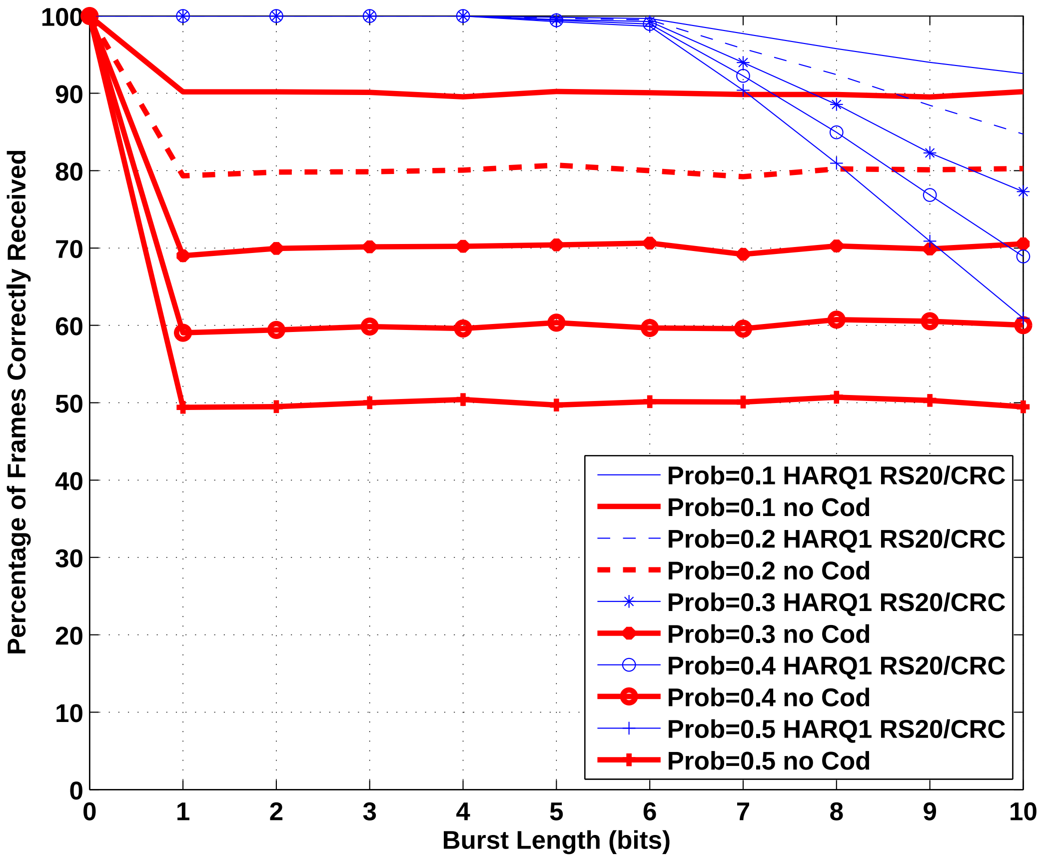

Figure 5.2 Comparison of the Percentage of Correct Frames Received with and without R-S codes. $SNR = 20dB$, Probability of Burst Error= 0.1 to 0.5, Burst Length= 0 to 10 bits.

the percentage of correct frames received for the HARQ Type-I R-S20/CRC method and the HARQ Type-I R-S20/Cyclic method. The HARQ Type-I R-S20/Cyclic method performs exactly the same as the HARQ Type-I R-S20/CRC method because, when the R-S codes cannot correct the errors in the CAN frame, the cyclic codes can never correct them. The error-correction capability of the cyclic codes is lower when compared to the R-S codes. Another major disadvantage of the HARQ Type-I R-S20/Cyclic method is the fact that the receiver cannot detect the errors in the received frame and can never receive the corrected frame. This method did not provide any improvement over the HARQ Type-I R-S20/CRC method, but surely helped in observing that use of error-correction codes alone without any error detection can lead to erroneous frames at the receiver. Therefore, in an environment where the type of error cannot be determined, both error-correction and error-detection have to be used together to get the best throughput performance with high reliability. This method also

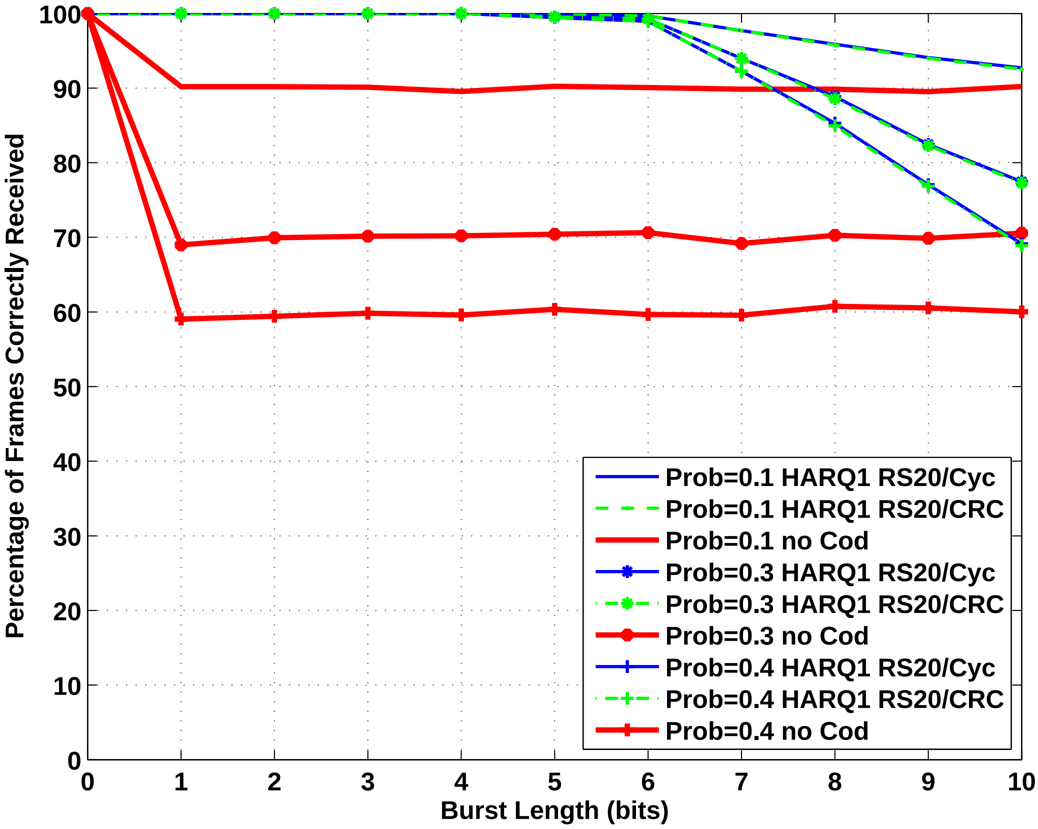

Figure 5.3 Comparison Percentage of Correct Frames Received with HARQ Type-I R-S20/Cyclic, HARQ Type-I R-S20/CRC and without Coding. $SNR = 20dB$, Probability of Burst Error= 0.1 to 0.5, Burst Length= 0 to 10 bits.

laid the foundation for another idea of increasing the error-correction capability of the HARQ Type-II R-S20/CRC scheme by increasing the R-S parity bits to correct more burst errors.

Figure 5.4 shows a bar graph for the percentage of frames corrupted with and without R-S codes. The probability of burst error occurrence was varied from 10% to 50% and a total of 10^6 bits (8000 frames) were transmitted as the probability of burst error increased linearly. As the probability of error increased from 10% to 50% the percentage of corrupted frames without coding increased from 10% to 47.18%. The implementation of the R-S coding scheme in the form of HARQ Type-I R-S20/CRC reduced the percentage corrupted frames to 1.6513% for a probability of 10% and 7.82% for a probability of 50%. This proves that the the percentage of corrupted frames for HARQ Type-I R-S20/CRC scheme is 1/8 times the conventional CAN without coding. This is a significant contribution as the number of retransmissions will be reduced.

The SNR value has been kept constant for all the previous simulations, while the other parameters such as the burst length and the probability of burst error occurrence were varied.

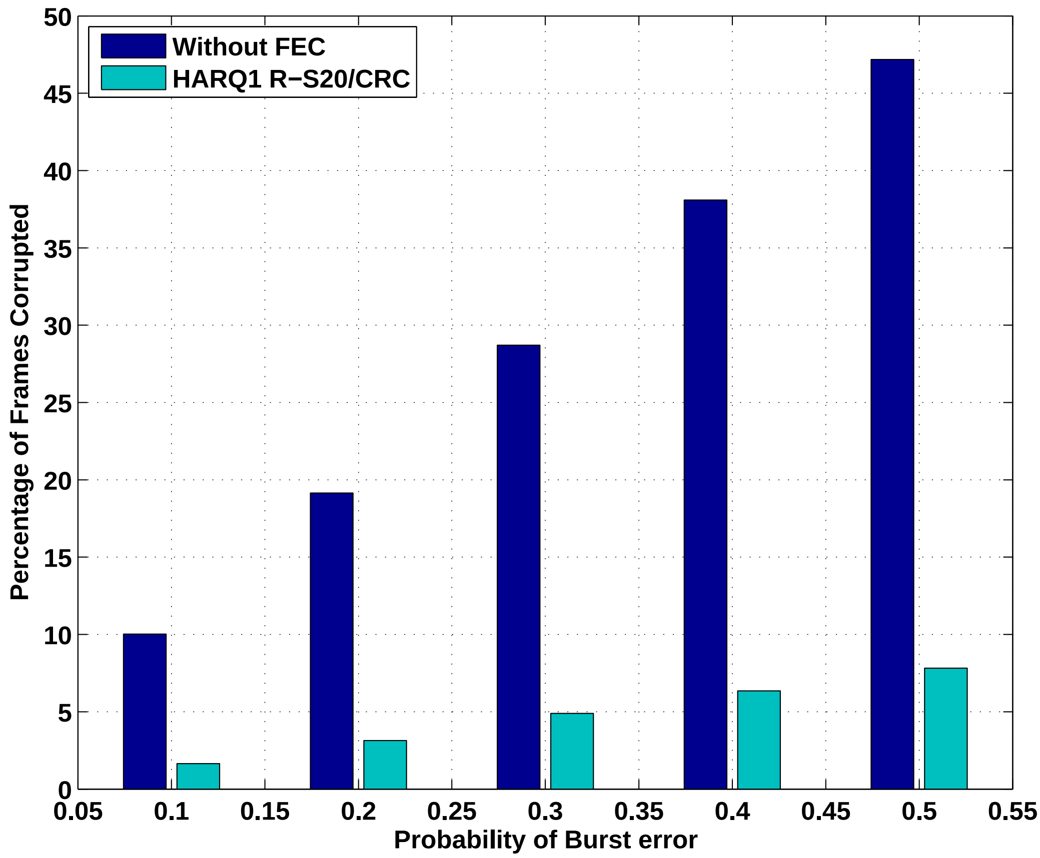

Figure 5.4 Comparison of Percentage of Frames Corrupted with and without R-S coding for different Probability of Burst Error Occurrence

The performance of the HARQ Type-I R-S20/CRC method with varying SNR values would provide important information about the performance of the scheme for other wired CAN bus applications or in entirely different applications like the wireless CAN. Figure 5.5 shows the curves with the R-S coding and without the R-S coding for varying SNR, different probabilities of burst error and random length of the burst error. The thick lines represent the percentage of correctly received frames without error-correction codes and the thin lines represent the percentage of correctly received frames with error-correction codes. The SNR of the channel has been varied from 5dB to 25dB and the length of the burst for this simulation is random, varying from 1 to 10 bits. The probability of burst error occurrence is varied from 10% to 50%. Even in the worst case of SNR = 5dB and the probability of burst error occurrence equal to 50%, the percentage of frames correctly received by the HARQ Type-I R-S20/CRC method is 7% higher than the method without coding.

The simulation results show that the best way to improve the CAN bus performance is to use the Hybrid ARQ technique, which ensures correct transmission of complete data.

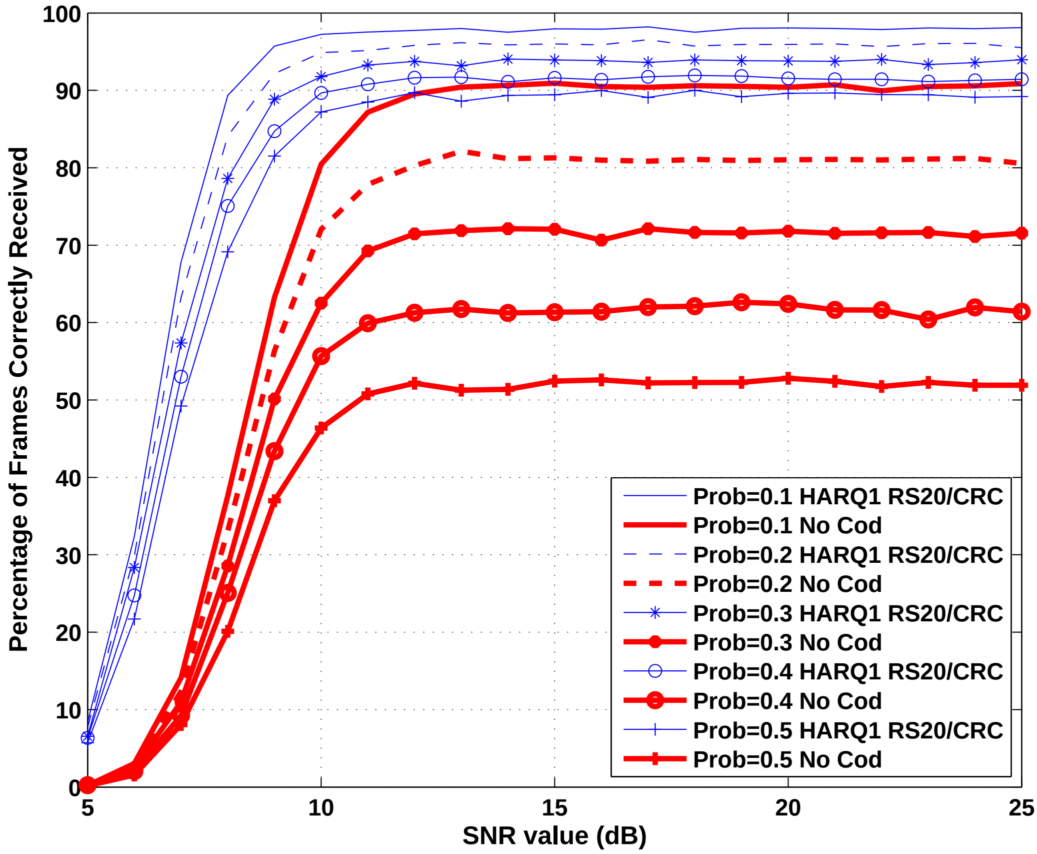

Figure 5.5 Plot for Percentage of Frames Correctly Received for Different Probability of Burst Errors, Varying SNR and Random Burst Length for HARQ Type-I R-S20/CRC and no error-correction

Moreover, AWGN does not seem to affect the CAN frame because in the CAN bus the SNR values are so large that random noise cannot play a vital role in inducing errors [16]. EMI causes most of the errors in CAN communications.

5.2. BIT OVERHEAD COMPARISON OF PROPOSED METHODS AND CONVENTIONAL CAN

The HARQ Type-II R-S20/CRC uses the same Reed-Solomon(R-S) codes, but the parity bits are only sent once the error has been detected by the receiver. A comparison of the conventional CAN bus ARQ, the HARQ Type-I R-S20/CRC, and the HARQ Type-II R-S20/CRC in terms of the percentage of bit overhead due to the retransmission process is shown in Fig. 5.6. In the following simulation, 100000 frames were transmitted and each frame has been encoded with R-S codes. HARQ Type-I R-S20/CRC method is assumed to be able to correct all errors with the R-S20 parity bits added to the frame i.e. no retransmission

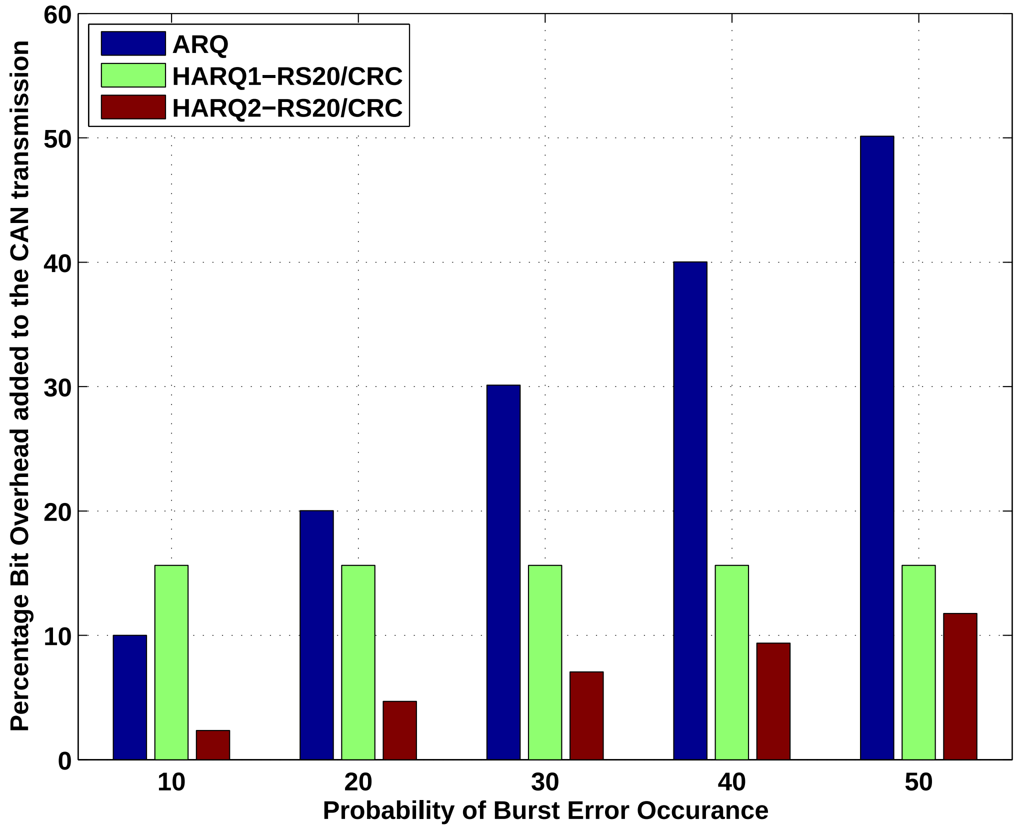

Figure 5.6 Bar Plot Comparison of Percentage overhead for ARQ, HARQ Type-I R-S20/CRC and HARQ Type-II R-S20/CRC methods. Probability of Burst Error Occurrence Varies from 10% to 50%. Percentage of uncorrected frames = 10%.

are necessary for the HARQ Type-I R-S20/CRC method. In a conventional ARQ system, the number of bits retransmitted will be equal to bits per frame times the number of frames retransmitted. In case the of HARQ Type-I R-S20/CRC, the number of overhead bits will be equal to the number of parity bits in each of the HARQ Type-I R-S20/CRC CAN frame times the number of frames transmitted.

In the HARQ Type-II R-S20/CRC all the frames are encoded using R-S codes, but the frame is not appended with the parity bits. In this case the number of overhead bits will be equal to the number of parity bits times the number of frame retransmissions. When the percentage of bit overhead for the three methods is compared, at a low probability of burst error occurrence the percentage bit overhead for the HARQ Type-I R-S20/CRC is the maximum. At a probability of error equal to 10%, HARQ Type-I R-S20/CRC is 5.65% higher than the conventional ARQ system and 13.28% higher than the HARQ Type-II R-S20/CRC method. Figure 5.7 shows that the percentage overhead for HARQ Type-I R-S20/CRC is better than that of the conventional ARQ for a probability of error higher than 15%. The

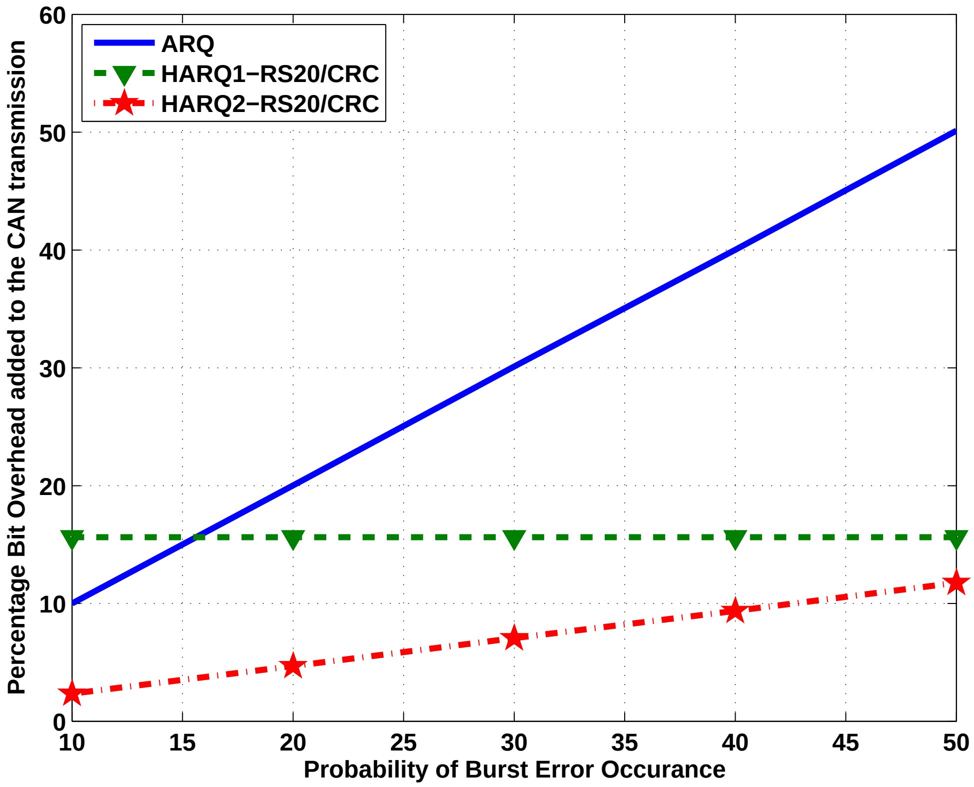

Figure 5.7 Comparison of Percentage overhead for ARQ, HARQ Type-I R-S20/CRC and HARQ Type-II R-S20/CRC methods. Probability of Burst Error Occurrence Varies from 10% to 50%. Percentage of uncorrected frames = 10%.

percentage overhead for HARQ Type-II R-S20/CRC is always lower than the HARQ Type-I R-S20/CRC for probabilities lower than 60%. Above 60% the HARQ Type-I R-S20/CRC has lesser bit overhead compared to the HARQ Type-II R-S20/CRC. Though the bit overhead of is lower for the HARQ Type-II R-S20/CRC, the receiver has to wait for the parity bits. This is a tradeoff that should be taken care while designing the right method for a particular environment.

In some cases the HARQ Type-I R-S20/CRC or HARQ Type-II R-S20/CRC method might not be able to correct all the corrupted frames. In these cases additional error-correction parity bits would correct the errors and hence improve the performance of the system in terms of bandwidth efficiency. This method is called HARQ Type-II R-S40/CRC as mentioned in Section 4. Figure 5.8 shows the percentage bit overhead comparison of the three HARQ R-S/CRC methods and the conventional ARQ scheme. In this simulation the probability of burst error occurrence increases from 10% to 50%. At the receiver, 10% of the corrupted frames are assumed to be uncorrectable. In this case the receiver with the HARQ Type-I R-S20/CRC or

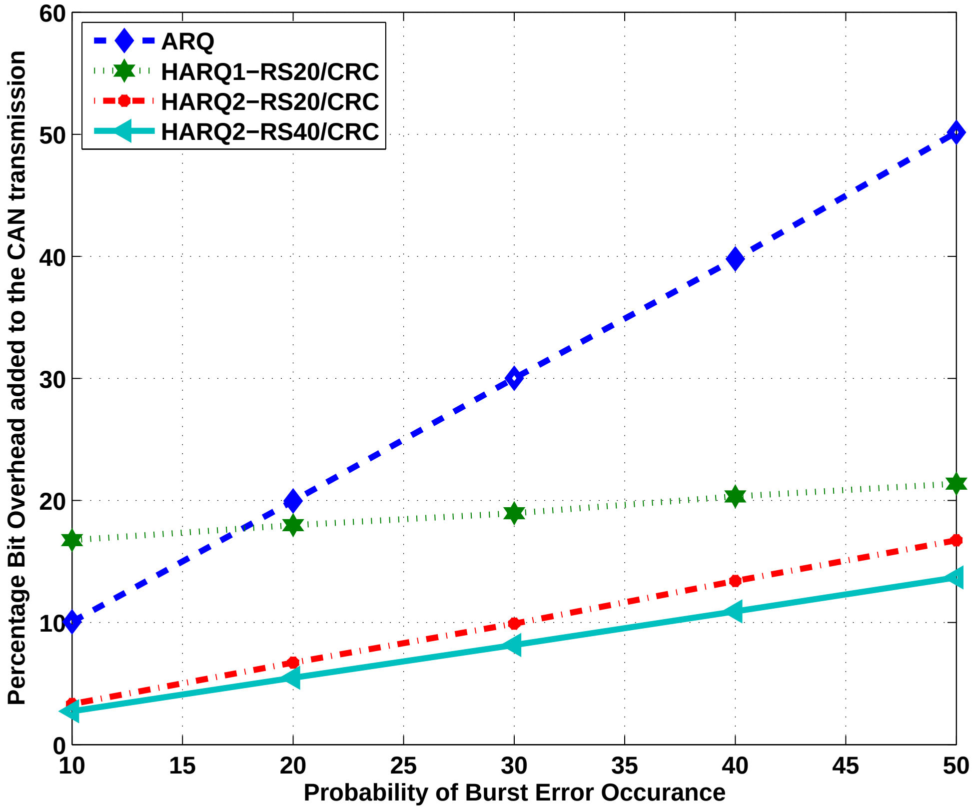

Figure 5.8 Comparison of Percentage overhead for ARQ, HARQ Type-I R-S20/CRC, HARQ Type-II R-S20/CRC and HARQ Type-II R-S40/CRC methods. Probability of Burst Error Occurrence Varies from 10% to 50%. Percentage of uncorrected frames = 10%.

the HARQ Type-II R-S20/CRC scheme requests for the whole frame retransmission, while the receiver with the HARQ Type-II R-S40/CRC requests for the 40 additional error-correction parity bits. In most of the cases the HARQ Type-II R-S40/CRC scheme will correct all the possible errors in CAN bus communications. When the probability of burst error occurrence is 40%, and when 20% of the corrupted frames are assumed not correctable by R-S20 codes, HARQ Type-II R-S40/CRC has 31.4% lower bit overhead than the conventional ARQ scheme, 16% lower than HARQ Type-I R-S20/CRC and 6% lower than the HARQ Type-II R-S20/CRC. Hence, the HARQ Type-II has the least bit overhead compared with any other method discussed in this thesis. This method is also an adaptive method which adapts to the environment and the performance of this method is always better than the other methods in terms of less retransmissions and less bit overhead at an expense of computational complexity. Another tradeoff that needs to be considered is the wait time for the retransmission of the parity bits.

6. CONCLUSION

The first method (Cyclic60) can be used for the CAN environments where the severity of the burst created by the EMI is limited to 1 bit error. In this environment, all the frame errors can be corrected without frame retransmissions because the method can correct all single bit errors. In an environment with a high intensity of EMI such that a burst error spreads over 5 to 6 bits, the proposed HARQ Type-I R-S20/CRC method improves the performance of the CAN system. The R-S codes used in HARQ Type-I R-S20/CRC correct 100% of corrupted frames for burst lengths shorter than 7 bits. When the burst length is greater than 7 bits and when the probability of the CAN frame getting corrupted is 40% the HARQ Type-I R-S20/CRC scheme requires 30% fewer frame retransmissions for burst lengths of 8 bits and 11% fewer for burst lengths of 10 bits. The HARQ Type-II R-S20/CRC performs exactly the same as the HARQ Type-I R-S20/CRC in terms of error-correction capacity, but the former has less bit overhead than the latter. When the probability of error is equal to 10%, the bit overhead of the HARQ Type-I R-S20/CRC is 5.65% higher than the conventional ARQ system and 13.28% higher than the HARQ Type-II R-S20/CRC method. The HARQ Type-I R-S20/CRC has higher percentage bit overhead at low channel error rates because every frame is appended with the 20 R-S parity bits.

The replacement of the Cyclic Redundancy Check bits (CRC) by the cyclic codes (HARQ Type-I R-S20/Cyclic) did not improve the performance of the the system in terms of receiving the corrected frames but this simulation laid the foundation for using a new method where higher error-correction capacity is provided to the receiver (HARQ Type-II R-S40/CRC). It also confirmed that using error-correction alone in a varying environment is not a convincing way of handling the frame errors. Hence, it is always suggested to use both error-correction and error-detection codes together in order to maintain the same level of reliability and also improve the performance in terms of bandwidth efficiency.

Depending on the environment in which the CAN bus operates, the appropriate protocol must be chosen from the ones suggested in this thesis and the one originally used by the CAN bus. ARQ (Conventional CAN) can be used when the error rate of the channel is low (less than 10%) because, the system is simple to use and requires no change to the current system being used. The HARQ Type-I R-S20/CRC can be used when the error probability is higher than 15% because, in this environment the bit overhead added to each frame for error-correction overtakes the bit overhead due to retransmissions when compared to the conventional ARQ scheme. Another advantage with the HARQ Type-I R-S20/CRC method is that the error-correction parity bits are transmitted along with the CAN frame. Hence the receiver does not have to wait for the retransmission of R-S20 bits. If the receiver does not require the correct frames immediately and if it can afford to wait for the retransmission of R-S parity bits, then HARQ Type-II R-S20/CRC can be used from very low probabilities of burst error occurrence up to very high values of the probability (close to 60%) because the percentage bit overhead for the HARQ Type-II R-S20/CRC is always lower than the HARQ Type-I R-S20/CRC for probability of error values lower than 60%.

When the HARQ Type-I R-S20/CRC or the HARQ Type-II R-S20/CRC method is not able to correct the erroneous frames, retransmission of the entire frame is requested. It is a good option to use the HARQ Type-II R-S40/CRC method in this case because the bandwidth efficiency will definitely be better than the other two methods. When the HARQ Type-I R-S20/CRC or the HARQ Type-II R-S20/CRC is not being able to correct 20% of the corrupted frames, and when the probability of burst error occurrence is 40%, HARQ Type-II R-S40/CRC has 31.4% lower bit overhead than the conventional ARQ scheme, 16% lower than the HARQ Type-I R-S20/CRC and 6% lower than the HARQ Type-II R-S20/CRC.

Finally, in a highly inductive environment with the probability of error occurrence around 20% to 30%, the best method that can be used is a combination of HARQ Type-I R-S20/CRC and the R-S40 bits idea to correct the errors that are not correctable by the former method. Using this combination most of the errors can be corrected with immediate action because the HARQ Type-I R-S20/CRC method has the error-correction parity bits along with the mail. And in cases when it fails to correct the errors, the R-S 40 scheme can correct the additional errors.

BIBLIOGRAPHY

[1] "Can specification version 2.0," 1991. The Bosch IC Design Center, Reutlingen, Germany. Available Online: http://www.semiconductors.bosch.de/pdf/can2spec.pdf.

[2] X. Wang, C. Huiyan, and D. Huarong, "The application of controller area network on vehicle," in *Vehicle Electronics Conference, 1999. (IVEC '99) Proceedings of the IEEE International*, pp. 455–458 vol.1, 1999.

[3] "Controller area network - a serial bus system - not just for vehicles," Available Online: http://www.esd-electronics.com/german/PDF- file/CAN/Englisch/intro-e.pdf.

[4] J. Lepkowski, "Emi/esd protection solutions for the can bus," July, 2004.

[5] B. C. Baker, "Ease into the flexible canbus network," 2003 Microchip Technology, Inc. Available Online: http://www.elforum.ro//files/adn004_can_117.pdf.

[6] CiA, "Controller area network (can) - protocol," Available Online: http://www.can-cia.org/can/protocol/.

[7] L. Shu, D. Costello, and M. Miller, "Automatic-repeat-request error-control schemes," *Communications Magazine, IEEE*, vol. 22, no. 12, pp. 5–17, 1984. 0163-6804.

[8] V. Tripathi, E. Visotsky, R. Peterson, and M. Honig, "Reliability-based type ii hybrid arq schemes," in *Communications, 2003. ICC '03. IEEE International Conference on*, vol. 4, pp. 2899–2903 vol.4, 2003.

[9] Wikipedia, "Controller area network," 26 September 2007 Available Online: http://en.wikipedia.org/wiki/Controller_Area_Network.

[10] B. Sklar, *Digital Communications: Fundamentals and Applications.* Prentice Hall Professional Technical Reference, second edition ed., January 2001.

[11] W. Stallings, *Wireless Communications and Networking.* New Delhi: Asoke K. Ghosh, Prentice Hall of India PVT LTD., 2003.

[12] E. Soljanin, R. Liu, and P. Spasojevic, "Hybrid arq with random transmission assignments," MAY 2003.

[13] KVASER, "The can protocol tour," 07-14-2007 Available Online: http://www.kvaser.com/can/intro/index.htm.

[14] R. E. Blahut, *Algebraic Codes for Data Transmission.* Cambridge University Press, 1st edition ed., July 15, 2002.

[15] Softing, "Can bus (controller area network), an overview," Available Online: http://www.softing.com/home/en/industrial- automation/products/can-bus/more-can-bus/index.php?navanchor=3010320.

[16] D. M. Davenport and R. T. Hoctor, "A physical layer for the can bus using modulated plc," in *Power Line Communications and Its Applications, 2005 International Symposium on*, pp. 176–180, 2005.

VITA

KRISHNA CHAITANYA SURYAVENKATA EMANI was born on November, 16th, 1983 in the southern part of India. He received his Bachelor's degree in Electronics and Communications Engineering from Aurora's Engineering College, Hyderabad, India in 2005. As a part of Curricular Practical Training he worked at Sprint-Nextel as a Summer Intern from May to August 2007 on the Voice Over IP team. He received his Master's degree from the University of Missouri-Rolla in December 2007. His topics of interest include digital communications, wireless communications, signal processing, and networking with the specialization being error correction codes and digital communications.