2012

A study of the effects of fieldbus network inducted delays on control systems.

Joseph Mianoo
Indiana State University

VITA

Joseph Mainoo

EDUCATION

Master of Industrial Technology, Bowling Green State University, OH, 2006
Bachelor of Science in Technology, Electronics and Computer Technology, Bowling Green State University, OH, 2004
Diploma in Management Information Systems, The Institute for the Management of Information Systems, Orpington, Kent, UK, 1999

WORK EXPERIENCE

Adjunct Faculty Member:	Department of Engineering Technology, University of Toledo, Toledo, OH. January 2012 to Present
Adjunct Faculty Member:	Department of Engineering Technologies, Bowling Green State University, Bowling Green, OH. January 2012 to Present.
Adjunct Faculty Member:	Electrical/Electronic Technology, Owens Community College, Perrysburg, OH. January 2011 to Present.
Graduate Assistant:	Department of Technology Systems, Bowling Green State University, OH, August 2004 to August 2010
Coop Student & Hardware System Specialist:	Information Technology Service, Bowling Green State University, May 2002 to August 2004

PUBLICATIONS AND PRESENTATIONS

- Mainoo, J. & Kolla, S. (2011). *Simulation and statistical analysis of network delays for controller area network*. Proceedings of International Conference on Information, Computing and Telecommunication (ICICT-2011), Bangalore, India, December 2011.
- Mainoo, J. & Kolla, S. (2011). *A fieldbus simulator for controller area network*. Proceedings of 2011 ATMAE Annual Conference, Cleveland, OH, November 2011.
- Mainoo, J. & Kolla, S. (2008). *A Study to establish Masters Degree in Electronics and Computer Technology at Bowling Green State University*. Proceedings of ASEE Conference, Pittsburgh, PA, 2008.
- Kolla, S. & Mainoo, J. (2006). *Fieldbus Networks Topic in Instrumentation and Control Systems Courses*. Proceedings of ASEE Annual Conference, Chicago, IL, 2006.
- Mainoo, J. (2006). *Fieldbus Networks and Security Measurers*. Presentation at the 52nd International Instrumentation Symposium, Cleveland, OH, 2006.

HONORS AND AWARDS

- Doctoral Fellowship Recipient (3 years), Bowling Green State University, 2006-2010
- Graduate Assistantship (2 Years), Bowling Green State University, 2004-2006
- Electrical Manufacturing and Coil Winding Association (EMCWA) scholarship recipient, $2000 award, 2003
- The International Society of Automation (ISA) scholarship recipient, $500 award, 2004

A STUDY OF THE EFFECTS OF FIELDBUS NETWORK INDUCED DELAYS ON CONTROL SYSTEMS

A dissertation

Presented to

The College of Graduate and Professional Studies

Ph.D. Consortium Program

College of Technology

Indiana State University

Terre Haute, Indiana

In Partial Fulfillment

of the Requirements for the Degree

Doctor of Philosophy

---

by

Joseph Mainoo

May 2012

Keywords: fieldbus network, controller area network, network induced delay, PID controller, technology management

COMMITTEE MEMBERS

Committee Chair: Sri R. Kolla, Ph. D.

Professor, Electronics and Computer Technology, College of Technology,

Bowling Green State University

Committee Member: David A. Border, Ph. D.

Associate Professor, Electronics and Computer Technology, College of Technology,

Bowling Green State University

Committee Member: Gerald W. Cockrell, Ph. D.

Professor, Electronics and Computer Engineering Technology,

Indiana State University

Committee Member: Robert E. English, Ed. D.

Associate Dean & Professor, College of Technology,

Indiana State University

Committee Member: Todd C. Waggoner, Ph.D.

Professor, Engineering Technology, College of Technology,

Bowling Green State University

ABSTRACT

Fieldbus networks are all-digital, two-way, multi-drop communication systems that are used to connect field devices such as sensors and actuators, and controllers. These fieldbus network systems are also called networked control systems (NCS). Although, there are different varieties of fieldbus networks such as Foundation Field Bus, DeviceNet, and Profibus available in the automation industries, Controller Area Network (CAN) is more widely accepted in automotive applications. The growing popularity of, and demand for, fieldbus networks can be attributed to several advantages they have, such as: reduction in capital costs, interoperability, and greater system functionality. However, as the complexity of modern fieldbus systems continue to increase, the concern on performance, reliability, and security also increases. To better reflect on this concern, the fieldbus based control systems should be extensively studied using simulations before implementing them in hardware. Network induced delays that may result from the bus arbitration schemes of the messages is an issue that needs investigation for these fieldbus networks. The impact of these delays on control system performance measures such as peak overshoot and settling time needs investigation. The purpose of this research was to study the causes of fieldbus network induced delays and their effects on control systems.

The existence and causes for network induced delays were studied by other researchers. Previous delay analyses used analytical and stochastic methods to establish relationships for

delays. This dissertation, however, uses statistical analysis methods to study the effect of various CAN parameters on network delays. The data for the statistical analysis was obtained from simulations. Though the literature indicates use of general purpose simulation tools such as OPNET, OMNeT++, and Network II, there exist simulation tools that are designed specifically to address a particular type of fieldbus such as CAN. The research in this dissertation uses such a tool called CANoe for simulating an automobile system. The impact of these delays on control system performance was studied by other research on Proportional Integral (PI) controllers. This dissertation extends these studies to Proportional Integral and Derivative (PID) controllers.

In this dissertation, the causes of network delays and how these delays are affected by CAN parameters such as baud rate, bus load, and message length were investigated using CANoe simulations of an automobile system. The statistical techniques of descriptive statistics, and analysis of variance (ANOVA) were used to analyze data obtained for this part of the study. The findings of the ANOVA analysis revealed that CAN parameters have effect on CAN message delays. The effect of fieldbus network induced delays on control system performance such as stability and step-response for different PI and PID controllers were studied using a DC motor model. The delays considered were sensor-to-controller delay and controller-to-actuator delay. MATLAB/Simulink tools were used to analyze the effects of these delays. From this study, it was observed that fieldbus network induced delays have an effect on control systems stability and performance as described by the system step response. The results of this performance evaluation will be useful to design PID controller gains, and to verify how sensitive the control loops are under various time delays.

ACKNOWLEDGMENTS

It gives me great pleasure to thank all the people who in diverse ways have helped to make this dissertation possible. A debt of gratitude is owed to the chair of my committee, Dr. Sri Kolla, for demonstrating grace, patience, wisdom, thoughtful guidance and constructive criticism, well beyond the call of duty as I struggled to complete the requirements of this challenging journey. His enthusiasm, motivation, and great efforts to explain things more clearly have helped to make research more enjoyable for me. I also thank the other members of my dissertation committee: Dr. David Border, Dr. Gerald Cockrell, Dr. Robert English and Dr. Todd Waggoner for their tremendous guidance, great advice, and direction throughout my PhD study. Their support as well as helpful suggestions increased the readability of this report.

I would like to thank my Ph.D. planning committee chair Dr. Sri Kolla and committee members Dr. David Beach and Dr. Wilfred Roudebush for all their help and guidance in my preliminary exams. Dr. Sri Kolla provided warm encouragement, great advice, good company, and great ideas not only during my dissertation writing period but throughout my Ph.D. studies. I thank Dr. Wilfred Roudebush for his support during my internship and Dr. George Maughan for his support during my residency. I would like to thank Dr. Erik Mayer for his encouragement during the course of my Ph.D. study. I would also like to thank Mr. Bruce Emaus of Vector CANtech, Novi, MI, for providing support to BGSU through “CAN Goes to College” grant program, and for the use of CANoe material in this work. I am grateful to Dr. Donna Trautman and Ms. Kim Strickland at Bowling Green State University for their encouragement and their personal interest in my wellbeing and academic growth. I thank Ms. Mary Griffith at Indiana State University (ISU) and Ms. Heidi Hakel at Bowling Green State University (BGSU) for their help and support. I also wish to extend my appreciation to the faculty and staff in the College of Technology at both BGSU and ISU.

I would like to thank my family for all their love and encouragement, especially for my parents who bore me, raised me, and supported me in all my pursuits. I wish to express my thanks to Dr. Philip Twumasi-Ankrah for his brotherly love and support, and Dr. Alexander Y. Adusei, Jr. for his encouragement. For all my other siblings, extended family and friends who have been such great help and are not mentioned here, I would like to extend sincerest appreciations.

Finally, I thank my Lord Jesus Christ for His Grace, unfailing mercies and the privilege to complete this Ph.D. dissertation.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF EQUATIONS

CHAPTER 1

INTRODUCTION

Background and Context of the Study

Industrial process control technology has rapidly evolved over the last several years (Dunn, 2005). It progressed from electronic analog control (in 1950s), to direct digital control (in 1960s), to programmable logic controllers (in 1970s), to distributed control systems (in 1980s), and now to fieldbus control systems (in 1990s) (Goettsche, 2005). The integration of intelligent field devices, digital bus networks, and various open communication protocols has produced remarkable results at industrial process plants around the world. while ability to retrieve, share, and analyze data has significantly increased by the use of the Internet and PC network technologies in business organizations, the ability to control and manage industrial process plants have also been improved by the implementation of digital communication technologies. However, these technologies also bring additional issues on control system performance that needs further investigation. One such issue is the delays introduced by these networks.

These digital communication technologies are commonly referred to as “fieldbus networks.” Fieldbus is a bi-directional digital communication network that enables the connection of multiple field instruments and processes and operator stations (Mahalik, 2003). It interconnects “field” equipment such as sensors, actuators and controllers, and it functions as a Local Area Network (LAN) for instruments used in both process and manufacturing automation.

These networks are also called Networked Control Systems (NCS) (Zhang et al., 2001). For instance, in conventional communication, an instrument sends its measured value by varying the current it uses between 4 and 20mA. In a similar way, an actuator is controlled by varying the current signal sent to it. This standard is referred to as 4-20 mA. If devices with this standard are used in a large complex plant, large numbers of cables have to be installed. Fieldbus replaces this existing 4-20mA analog signal standard, and provides many advantages. Figure 1 shows a typical fieldbus network connection and 4-20 mA analog connection in an industry (Yu, 2007).

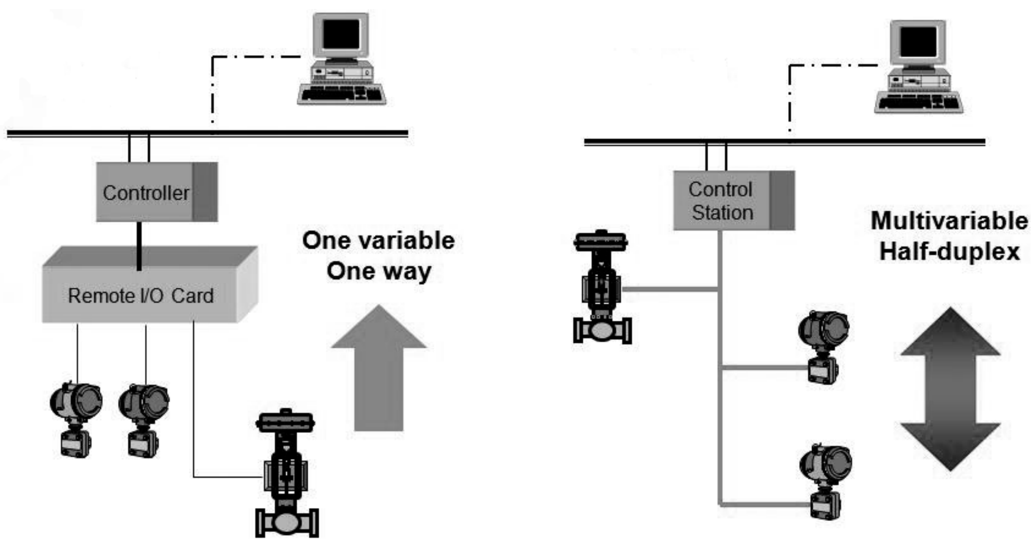
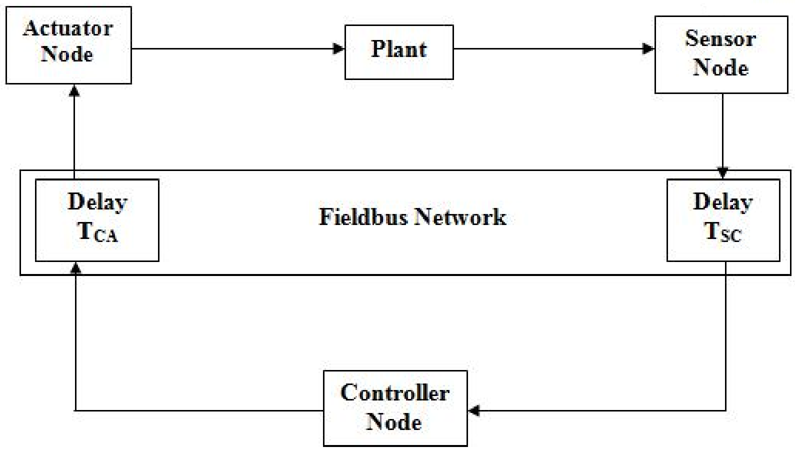


Figure 1. 4-20 mA Analog and Digital Fieldbus Standard Connections (courtesy of Yu, 2007).

There are different types of fieldbus networks in industrial application domain. These communications protocols can be classified as sensor networks, device networks, and process control networks (Kolla et al. 2003). The difference in these three network types can be attributed to the different areas of application the particular buses are used in. The sensor network is at the lowest level of process automation. AS-i (Actuator Sensor Interface) is the most common sensor bus. The device network is used for connectivity in areas with greater concentration of discrete devices, variable speed drives, and motor control centers. The most commonly used device bus networks include DeviceNet and Profibus-DP. DeviceNet is extensively used in factory automation and is also proving to be useful in process automation. It is built on the Controller Area Network (CAN) physical communications standard initially designed for the automotive industry. The process control network provides connectivity of sophisticated process measuring and control equipment. Examples of these networks are Foundation Fieldbus and Profibus PA (Networx, 2011). While the CAN protocol forms basis for some of the device bus networks such as DeviceNet as explained before, it is extensively used in automotive market to reduce the weight and cost of wiring harnesses as well as to provide additional communication capabilities (Emadi, 2005). The CAN bus is mostly utilized in embedded systems. Many variants of CAN protocol such as CANopen, CAN Kingdom, have also been developed and used in automobile and other transportation related areas (Bell, 2002). Because of this importance of CAN fieldbus, it will be utilized in this research study.

The fieldbus industry is continually improving its products down to the device and system levels in order to better meet end user specifications and to improve customer satisfaction. The process market indicates a rapid growth for fieldbus industry and manufacturing processes (Mahalik, 2003). The demand for new fieldbus based process control applications has further increased across the industry as plants move from outdated proprietary systems to a more advanced digital platforms. As indicated by O'Brien (2005), the ARC Advisory Group reported that the adoption of fieldbus network is growing among industrial end users worldwide. The ARC report was based on a survey of industrial end users, OEMs, and system integrators from around the world. It was established that fieldbus based control systems are being utilized on both large and small projects, and in an extensive range of mission critical applications. The growing dependence on fieldbus systems for better network performance has caused a shift to fieldbus automated systems and new strategic mergers have focused on pooling the best technological resources available and increasing market coverage, and penetration. Because of this, suppliers and engineers have formed different foundations in support of this technology (Mahalik, 2003). ABB and Aventis Pharmaceuticals have teamed up to work in the area of process automation. This is an indication that the user-industry and automation-manufacturers have taken a practical approach to develop more systems that are appropriate in the future (ABB News Center, 2002). The International Society of Automation (ISA), the International Electro-technical Commission (IEC), Profibus (German national standard) and the FIP (French national standard), further came together to form what is known as the IEC/ISA SP50 Fieldbus committee (Felser, 2002).

The fieldbus is implemented in the feedback control systems. The control loops are closed through a real-time network as shown in Figure 2. The significant aspect of this network is that information in the form of signals is communicated through the network and some control components. These components may include sensors, controllers, and actuators. Also, typical messages transmitted between and among devices can either be control data, which is time critical, or information data, which only needs guaranteed delivery. The implementation of fieldbus into control loop therefore provide some advantages such as easier installation and maintenance, easy diagnostics of system, and increasing flexibility of control system architecture. However, this network interconnection also has some fundamental concerns.



T_{CA}: Fieldbus Network Time Delay between Controller and Process Actuator.
T_{SC}: Fieldbus Network Time Delay between Process Sensor and Controller.

Figure 2. Network Control System.

One of the concerns is the issue of network-induced delay such as sensor to controller delay, and controller to actuator delay as indicated in Figure 2. This occurs during the exchange of data among devices connected to the shared medium. This delay can degrade the performance of control systems and can also affect the stability of the entire system (Jianyong et al., 2004). Other performance concerns are collisions in message transfers when multiple resources try to access the common bus. These collisions result in several issues which include information missing as packet dropout, and delays when the information is resent (Zhang et al., 2001). The packet dropout may be a concern in some network protocols. Also, the nature of the network induced delays depends on the chosen network protocol. Lian et al. (2005) compared some of the currently available control networks in regard to medium access mechanism (MAC). These control networks can be divided into the following three groups:

1. Control network based on a collision detection mechanism, CSMA/CD: Ethernet, Modbus/TCP, Ethernet/IP, European Installation Bus (EIB), and LonWorks.
2. Token passing control networks: Profibus, ControlNet, BACnet, MAP, P-Net, and WordFIP.
3. Control networks based on a mechanism of arbitration of messages priority CSMA/AMP: CAN, DeviceNet, and CANOpen.

CAN has been one of the popular protocols used for NCS (Bosch, 2006). CAN was initially developed to be used in automotive industrial applications to interconnect electronic control units (ECUs). The growing popularity of CAN is due to the low cost of its development and its greater acceptance in the industrial and academic areas. CAN-based networks are applied in distributed systems in many areas such as robotic control, automated manufacturing and process control environments (Voss, 2008). Despite the type of network used, the entire performance of NCS can be affected by network delays.

The primary objective of this dissertation work was to study the effects of fieldbus induced delays on control systems. The causes of message delays in an automobile system were first investigated. Message delays was defined as the time interval between the instances when a message was ready for transmission and queued at a transmitting node until the message was successfully arrived at a receiving node (Vector[CANoe], 2009). By this definition, message delay included the wait time at the transmission node, the time for the bus contention resolution, and the transmission time. CANoe, a software tool used for the development, testing, and analysis of CAN bus systems was used to study the effect of CAN parameters such as baud-rate, bus-load and message length on the message delay.

Second, an analysis of the effect of network induced delays on the performance of control systems was investigated. The simulation tool MATLAB/Simulink was used to perform this analysis. The performance criteria parameters used for the control system in this analysis was related to transient step response measures such as peak value, rise time, settling time (Dorf et al., 2010). Also System response with different Proportional, Integral, and Derivative (PID) controller gains with respect to delay times was analyzed. Electrical system such as DC Motor was used in this study. The system was modeled using Laplace Transform method. Systematic gains selection was developed and analyzed for PID type controllers for NCS. The results of this performance evaluation can be useful to design PID controller gains, and the verification of how sensitive the control loops are under various timing issues.

Statement of the Problem and Objectives

The problem of this research was to study the effects of fieldbus network induced delays on control systems. The causes of delays and how these delays are affected by CAN parameters such as baud rate, bus load, and message length was first investigated using CANoe simulations of an automobile system. The effect of fieldbus network induced delays on control system performance such as stability and step-response for different controllers was also studied using MATLAB/Simulink software for an electrical system. The step-response performance criteria studied was peak-value, settling-time and rise-time. Controllers such as Proportional, Integral, and Derivative (PID), was considered in this study. Controller designs to accommodate these delays were investigated.

Research Questions

In order to achieve the objective of the problem statement, the study was designed to help answer the following research questions:

1. Does the inclusion of fieldbus networks such as CAN in automobile systems induce network delays?
2. What are the effects of CAN parameter such as “bus loads”, “baud rate”, and “message length” on network delays?
3. Does fieldbus network induced delay have an effect on control systems stability and performance as described by system step response?
4. Do the PID type controller gains have effect on control systems with fieldbus network induced delays?

Research questions 1 and 2 were answered using statistical analysis methods such as ANOVA analysis using SPSS on data generated by CANoe simulations of an automobile system. The research hypotheses used for these statistical methods have been stated below. Research questions 3 and 4 were answered using mathematical analysis methods such as Laplace Transform (Dorf, 2010) equations for a DC Motor in MATLAB/Simulink simulations.

Research Hypothesis

The following were the hypotheses used to answer research questions 1 and 2 of the study:

Effect of CAN Baud Rates

1. H_{01}: $\mu_{50} = \mu_{100} = \mu_{200}$

There are no statistically significant differences in the mean values of network delays for signal transmission baud rates 50 kbps, 100 kbps and 200 kbps.

H_{A1}: $\mu_{50} \neq \mu_{100} \neq \mu_{200}$

There are statistically significant differences in the mean values of network delays for signals transmission baud rates 50 kbps, 100 kbps and 200 kbps.

Effect of CAN Message Length

2. H_{02}: $\mu_8 = \mu_6 = \mu_1$

   There are no statistically significant differences in the mean values of network delays for signals with message lengths 8 bytes, 6 bytes, and 1 byte.

   H_{A2}: $\mu_8 \neq \mu_6 \neq \mu_1$

   There are statistically significant differences in the mean values of network delays for signals with message lengths 8 bytes, 6 bytes, and 1 byte.

Effect of CAN Bus Loads

3. H_{03}: $\mu_{22\%} = \mu_{17\%}$

   There are no statistically significant differences in the mean values of network delays for busload of 22% and 17%.

   H_{A3}: $\mu_{22\%} \neq \mu_{17\%}$

   There are statistically significant differences in the mean values of network delays for busload of 22% and 17%.

Limitations of the Study

One of the limitations to the scope of this study was collecting sample data. The data collected focused on CAN message delay using CANoe software. As such, it was difficult to generalize the study findings across other fieldbus systems. The study was further limited by the hardware and software available at the Electronics and Computer Technology Laboratory at Bowling Green State University.

Significance of the Study

The National Institute for Standards and Technology (NIST) and the Abnormal Situation Management (ASM) Consortium estimate that U.S. process industries lose over $20 billion each year from abnormal situations (Walker et al., 2011). The abnormal situations refer to any process disruption where an automated process control system cannot maintain the process within acceptable limits (Rathinasabapathy, 2005). Several plants use distributed network control systems to concurrently control many process variables such as temperature and pressure. The growing demands for efficiency and productivity in these industries have resulted in significant increase in the complexity of these network control systems through the development of advanced sensor and control technologies. However, these sensor and control technologies have not completely eliminated abnormal situations.

Regardless of the type of network used, the general networked control system performance is always affected by network delays since the network is tied with the control system. These challenging problems, thus, network induced delay, packet dropout, and message collision can degrade the control performance of the overall real-time NCS which eventually can cause abnormal situations to the plants (Zhang et al., 2001). The network induced delays such as sensor to actuator and controller to actuator delays in the feedback loop influence the closed loop system performance and stability. A study of the performance evaluation with respect to the effect of fieldbus induced delay on control systems is, therefore, essential.

Currently, the CAN fieldbus is used in many industrial applications. These include space and aviation, maritime, medical, automobile, as well as household appliances such as washers, dryers and even coffee machines (Voss, 2008). This dissertation research, therefore, used CAN as the basis to investigate the causes of the delays and how CAN parameters such as baud rate, bus loads and message length affect these delays using CANoe simulation software. The impact of fieldbus induced delays on control systems stability and performance was also investigated using MATLAB/Simulink software.

Definition of Terms

- Application program interfaces (API): This is the specific method prescribed by a computer operating system or by an application by which a programmer writing an application program can make requests of the operating system or another application.
- AS-I (Actuator Sensor Interface): This is an industrial networking solution used in PLC, DCS and PC-based automation systems. It is designed for connecting simple field I/O devices such as binary ON/OFF devices like actuators, sensors, rotary encoders, analog inputs and outputs, push buttons, and valve position sensors in discrete manufacturing and process applications using a single 2-conductor cable.
- Baud rate: The baud rate is a measure of how many times per second a signal changes or could change and it is related to transmission speed.
- Busload: The busload is the ratio of transmitted bits to bus idle bits within a defined time unit. Where 100% means that bits are transmitted during the complete defined time unit and 0% means that the bus is in bus idle state during the complete defined time unit.
- Controller Area Network (CAN): This is a serial bus network of microcontrollers that connects devices, sensors and actuators in a system or sub-system for real-time control applications.

- CANdb++ editor: This supports developers in various phases of the development process such as creation of DBC files, adding messages and signals to existing DBC databases, definition of transmission and reception relations, defining environment variables for a CANoe simulation, adding special messages for test purposes, and adding messages for calibrating the ECUs
- CANLIB: This is a software product that provides support for different CAN interface boards. It also provides the application programmer with a quick and easy access to the CAN network.
- CAN Message: A message in CAN may be a data frame or remote frame.
- CANoe software: This is an all round tool for the development, testing and analysis of entire Electronic Control Unit (ECU) networks and individual ECUs. It supports the user during the entire development process, thus, from planning to the setting up of entire distributed systems and their individual ECUs.
- CAPL: This is Communication Application Programming Language. This makes it possible to program the CANoe for developer-specific applications that use the CAN protocol. CAPL may also be used in the CANalyzer tool for distributed product development.
- Carrier sense multiple access (CSMA): This is a Media Access Control protocol in which a node verifies the absence of other traffic before transmitting on a shared transmission medium, such as an electrical bus.
- Distributed architecture: This is an application architecture wherein application components are distributed across multiple computer systems.

- Fieldbus technology: This is the name of a family of industrial computer network protocols used for real time distributed control. It is currently standardized as IEC 61158.
- ISO11898: This specifies a serial communication technology called Controller Area Network that supports distributed real-time control and multiplexing for use within road vehicles.
- Message length: This may be the length of a data frame or remote frame.
- Message time (Delay of message): This is the time interval between generation of a message and correct reception of this message.
- Networked Control Systems: These are control systems that consist of the system to be controlled and of actuators, sensors, and controllers, the operation of which is coordinated via a communication network.
- Network Delay (t_{Tx}-t_{TxRq}): The difference between the times of transmits request and the time of transmission of messages.
- Peak-value: This is the maximum value of the output, reached after application of the unit step input after time t_p.
- PID: This is Proportional, integral, derivative control. It adjusts system outputs when there is a difference between the set-point and process variable.
- Real-time: An application in which information is received and immediately responded to without any time delay.
- Rise-time: This is the time required for the response to a unit step input rising from 10 to 90% of its final value. For under damped second-order system, the 0% to 100% rise time is normally used. For over damped systems, the 10% to 90% rise time is commonly used.

- Settling-time: This is the time required for the system output to settle within a certain percentage of the input amplitude.
- State feedback: This is a feedback control law in which the control inputs are explicit memory-less functions of the dynamical system state. This means that the control inputs at a given time are determined by the values of the state variables at that time and do not depend on the values of those variables at earlier times.
- τ_{ca} : Network control systems delay from controller to actuator.
- τ_{sc}: Network control system delay from sensor to controller.
- τ: Network Control Systems delay, $\tau = \tau_{sc} + \tau_{ca}$
- Time division multiple access (TDMA): This is a communication technique that uses a common channel such as multipoint or broadcast for communications among multiple nodes or users by allocating unique time slots to different users.
- Transmit (Tx): This is the actual time for the successful transmission of a message.
- Transmit Request (TxRq): This is set to indicate the CPU or remote node requested a message transmission, but not completed it. It is also internal event in the CAN controller to transmit a message.

Summary of Chapter

This chapter introduced the topic of this dissertation work by first giving the background and context of the study. The statement of the problem and objectives were also given. The research questions and the hypothesis to be tested in this work were stated. The significance of the study was explained. The chapter concluded with definition of important terms used in the study.

CHAPTER 2

LITERATURE REVIEW

Introduction

This chapter includes background on fieldbus networks and control systems. Previous research studies related to the performance evaluation of control systems with an emphasis on fieldbus network induced delays are discussed. The first section began with a discussion on the characteristics of fieldbus and industrial networks and how they are related to the International Organization for Standardization/Open Systems Interconnect (ISO/OSI) model. A brief introduction to some of the important fieldbus protocols such as Foundation Fieldbus, PROFIBUS, HART and others are also given. The second section discusses the impact and adaptation of fieldbus networks based on a study conducted by ARC. The third section gives background about control systems, system modeling, different controllers, and performance measures. The fourth section provides details of performance evaluation of control systems with fieldbus networks. The literature review presented in these sections provided a basis for the further studies on fieldbus network technology that were conducted in this dissertation.

Fieldbus and Industrial Networks

As explained before, a fieldbus is a digital communications network that is generally utilized for distributed instrumentation and control functions. According to Smith (2008), the fieldbus must have interconnectivity, interoperability and interchangeability. Interconnectivity is the ability of devices from different manufacturers to be securely connected to the same fieldbus thus, electrically compatible. Interoperability is also the ability to connect successfully, and operate elements from different suppliers. Interchangeability is the ability that devices from one manufacturer can be replaced with functionally equivalent devices from another manufacturer. According to Schuessler (2002) and Berge (2002), the increasing popularity of fieldbus network can be attributed to a host of advantages to the end user, listed below:

1. *Greater system functionality:* an unlimited number of parameters can be accessed from a device. This allows multiple measurements and a wide range of features to be crammed into advanced device firmware and accessed remotely by sophisticated software.
2. *Simplicity:* many features of fieldbus devices enhance their convenience and ease of use in comparison to traditional analog equipment.
3. *Accuracy:* Traditional 4-20 mA based systems require several stages of analog-to-digital (A/D) and digital-to-analog (D/A) conversions introducing quantization and other errors. Digital networking eliminates the need for these conversions.
4. *Less cost of purchase:* a system that is based on fieldbus technology requires significantly less hardware than a traditional system.
5. *Interoperability:* the ability of a device to work together with other devices. This enables easy and tighter integration of devices from different manufacturers.
6. *Savings:* fieldbus based systems will have a) engineering savings, b) construction savings, c) maintenance savings, and d) operation savings.
7. *Lower cost of expansion and change:* Since fieldbus systems are cheaper to buy and deploy, they are also cheaper to expand and modify.

LeBlanc (2000) indicated that, there are currently several different types of industrial networking or "fieldbus" technology on the market. Some of these are summarized in Table 1. These networks are designed based on the ISO/OSI seven-layer communication model.

Table 1

Existing Fieldbus Technologies.

Industrial Networks	**Type of Fieldbus**
ASI, Interbus-S	Sensorbus
WorldFIP	Fieldbus
CANOpen	Devicebus
ControlNet	Control
DeviceNet	Devicebus
Ethernet	Enterprise
FOUNDATION Fieldbus	Fieldbus
LonWorks	Devicebus
Profibus DP	Devicebus
Profibus PA	Fieldbus

The OSI Reference Model for Fieldbus Networks

The International Standard Organization (ISO) proposes a general model that can be used to look at the structure of network protocols and distributed applications. This is called International Standard Organization's Open System Interconnect (ISO/OSI) model (Berge, 2002). This section provides an introduction to this model and examines its applicability in the domain of fieldbus networks. The ISO/OSI model introduces a layered view of a network that distinguishes seven distinct layers. The OSI reference model is illustrated in Figure 3 and the different layers are explained below (Coulouris et al., 2005).

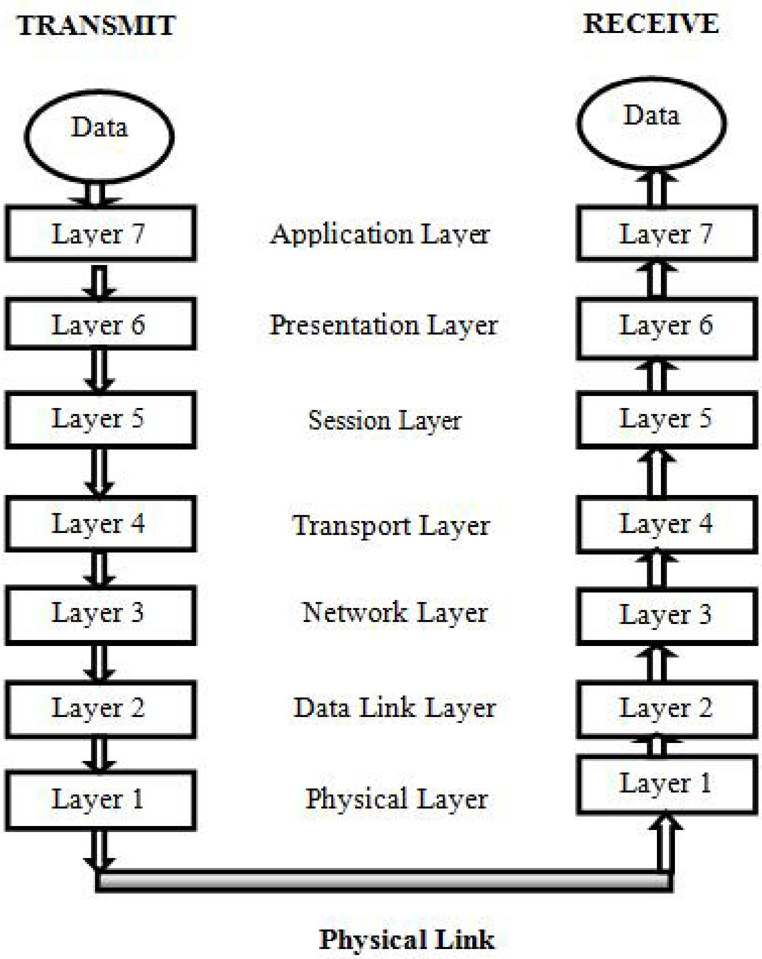


Figure 3. OSI Reference Model (courtesy of Coulouris et al., 2005).

Layer 1 - Physical

This layer contains the circuitry and hardware that drives the network. On this layer the bit-wise transmission of data is performed. The type of connectors, communication medium, electrical interfaces, electrical levels and procedures for sharing bandwidth (how long a bit can be present on a physical link) that are used in the network are also specified on this layer.

Layer 2 - Data Link

This layer is responsible for error-free transmission of packets between the connected communication devices. Raw (bit-wise) input data is converted into frames that are transmitted sequentially. This layer also deal with erroneous frames, retransmitted frames (due to lost acknowledgement messages), and flow control. The layer consists of two sub layers. These are Logical Link Control (LLC) and Medium Access Control (MAC). LLC defines how data is transferred over the cable and provides data link service to the higher layers. MAC controls media access by regulating the communicating nodes using pre-defined set of rules.

Layer 3 - Network

This layer deals with the transfer of data packets between computers in a specific network. The required routing for data that needs to be exchanged between multiple networks or subnets is also performed at this layer. For this reason, this layer is sometimes referred to as the communication subnet layer. Other tasks performed at this layer include monitoring (counting of packets) and treating network congestions.

Layer 4 - Transport

This is the first layer that deals with messages instead of packets (as in the three lower layers). The main function of this layer is to accept data from the layer above (the session layer), split the data into smaller parts, pass these parts to the network layer below and make sure that the data is successfully received by the other end. Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) are transport layer protocols. TCP ensures that a packet has reached its intended destination by using an acknowledgement. If not, it retransmits the lost messages.

TCP is called a connection oriented protocol. UDP basically transmits packets over the internet. It does not wait for an acknowledgement. It is the responsibility of upper layer protocols to ensure that the information had reached the intended entities. UDP is often called connectionless protocol.

Layer 5- Session

Communication between processes is established and error recovery is performed at this layer. This layer is no required in the case of connectionless services. A connection between two presentation layer processes is called a session, whereas the operation of setting up a connection is often called binding.

Layer 6 - Presentation

Data on this layer is transmitted in a network representation that is independent of the representations used in the individual accessing devices. These transformations include compression, conversion between different encoding schemes, and encryption.

Layer 7 - Application

At the application layer, protocols can be designed to meet the communication requirements of specific applications, thereby often defining the interface to a service. In general the content of the application layer is up to each individual user.

Some of the fieldbus protocols align their respective models to the ISO/OSI reference model but leave out some of the intermediate layers. On the other hand, some fieldbus protocols use an additional layer called user layer. Examples of these fieldbuses include Profibus, and Foundation Fieldbus. LonWorks follows a different approach in successfully implementing all layers of the reference model (Loy et al., 2001). The competing requirements of different areas of application led to different degree of relevance of the intermediate layers of the network model. For instance, in process automation, where very often, only a limited number of network nodes are connected to a single cable mainly for security reasons, the services of layer 3 (routing) is hardly required. This make the capabilities of layer 4 (transport) redundant.

Also, in the area of home automation, a network can consist of hundreds, or even thousands, of nodes. In order to handle such a large number of nodes the routing capabilities of a network become much more important. This kind of application leads to a model with at least five layers, compared to the variations with just three layers (physical, data link and application layers). Since LonWorks is applicable in many domains, the implementation of all seven layers could prove to be an advantage. Brief descriptions of some of the important fieldbus protocol standards are given below.

Foundation Fieldbus (FF)

This is an object-oriented protocol that uses multiple message formats and allows a controller to recognize a set of configuration and parameter information (device description) from devices that have been plugged into the bus. Foundation Fieldbus (ANSI/ISA-S50.02, Part 1, 1997) allows a device to transmit parameters that relate to the estimated reliability of a particular piece of data. FF uses a scheduler to guarantee the delivery of messages, so issues of determinism and repeatability are comprehensively addressed. Foundation Fieldbus communication layers consists of the physical layer, the communication stack, and the user layer as shown in Figure 4.

The communication stack performs the services required to interface the user layer to the physical layer. The communication stack consists of three layers: the Fieldbus Message Specification, the Fieldbus Access Sublayer, and the Data Link Layer (ANSI/ISA-S50.02, Part 4, 1997). The communication stack encodes and decodes user layer messages and ensures efficient and accurate message transfer. The Data Link Layer manages access to the Fieldbus through the Link Active Scheduler by splitting data into frames to send on the physical layer, receiving acknowledgment frames, and re-transmitting frames if they are not received correctly. It also performs error checking to maintain a sound virtual channel to the next layer. The Fieldbus Access Sublayer provides an interface between the Data Link Layer and the Fieldbus Message Specification layer.

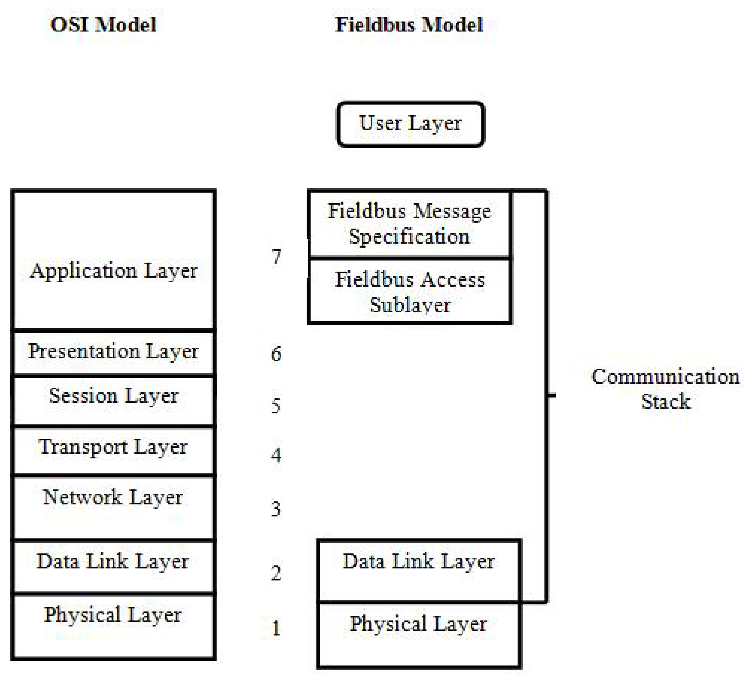


Figure 4. Detailed Fieldbus Network Protocol.

The Fieldbus Access Sublayer provides communication services such as client/server, publisher/subscriber, and event distribution. The Fieldbus Messaging Specification layer defines a model for applications to interact over the Fieldbus. The object dictionary and the virtual field device are important in this model. The object dictionary is a structure in a Fieldbus device that describes data that can be communicated on the Fieldbus. The services provided by the Fieldbus Messaging Specification allow user to read and write information about the object dictionary, read and write the data variables described in the object dictionary, and perform other activities such as uploading/downloading data and invoking programs inside a device. Within the Fieldbus Messaging Specification layer are two management layers called System Management and Network Management. System Management assigns addresses and physical device tags, maintains the function block schedule for the function blocks in that device, and distributes application time. Network Management contains objects that other layers of the communication stack use, such as data link, Fieldbus Access Sublayer, and Fieldbus Messaging Specification (National Instruments, 2003).

Profibus

This is a vendor-independent, open field bus standard used for various applications in manufacturing and process automation. The protocol architecture is oriented to the OSI reference model in accordance with the international standard ISO 7498. In this model, every transmission layer handles precisely defined tasks. Layer 1 (physical layer) defines the physical transmission characteristics. Layer 2 (data link layer) defines the bus access protocol. Layer 7 (application layer) defines the application functions (PROFIBUS PA, 2003).

Controller Area Network

CAN standard, popular in automotive applications, defines a simple broadcast serial network that works well for real-time short-range communications (CAN in Automation, 2009). Bosch developed the CAN protocol, which has since been standardized internationally as ISO11898 and has been “implemented in silicon” by several semiconductor manufacturers. CAN is the basis of several sensor buses such as DeviceNet, CAN Open, J1939, and Smart Distributed System. CAN uses a twisted pair cable to communicate up to 40m at speeds 1Mbit/s without repeaters, and up to 1 km at 20 kbps speed. It can support up to 40 devices. CAN utilized CSMA bus arbitration. The CAN protocol corresponds to the data link and physical layers in the ISO/OSI reference model. It also meets the real-time requirements of automotive applications (Voss, 2008). CAN data packets are 8 bytes long and use 11-bit packet identifier. A second version of CAN supports 29 bit identifier. A detail overview of CAN is given in chapter 3.

CAN Kingdom and CAN Open

CAN Kingdom is one of the higher Level protocols for the CAN protocol. Unlike other CAN high level protocols, it makes no attempt to follow the OSI model. The network is mostly distributed, as the nodes may run autonomously except that a “King” or master controller is needed to configure the network. A major design philosophy behind CAN Kingdom is that the system designer is fully aware of the capabilities of the nodes. A node designer, on the other hand, needs to know nothing about the other nodes: a node merely provides services to the network, and it is up to the system designer to activate a node’s services or not through runtime configuration (Voss, 2008). CAN Open was originally developed by Bosch as a high level protocol for CAN. In 1995 the specification was handed over to the CAN in Automation (CiA) International users and manufacturer group, where it has since been reviewed and expanded (CAN in Automation, 2009).

Local Interconnect Network

This is a single wire serial communications protocol based on the common SCI (UART) byte-word interface. A LIN cluster is controlled by a dedicated master, which holds all configuration information and controls the cluster. Communication is always initiated by the master. LIN was mainly developed as a low-level ultra low cost sub-bus for control networks in the automotive industry. It is mostly used for applications where CAN is considered to be expensive to be implemented (Rylander et al., 2003).

Impact and Adaptation of Fieldbus Networks

O’ Brian (2005) indicated that most process industries currently utilizes fieldbus network systems. According to the results from a survey research (Küster, 2005) there is a growing dependence in the implementation of fieldbus technology. This has led to an increasing number of installed fieldbus systems in most process and control industries. More than 60 end users, OEMs, and system integrators responded to the survey with representatives from many process industries. The integration of process fieldbus networks and device networks were rated highly as selection criteria for most of the industrial sectors. Frost and Sullivan (2009) also identified major industry challenges that affect the demand of fieldbus in their research study. Their research discussed the drivers and restraints that influenced fieldbus usage across the process industries in European countries. They broadly examined various types of fieldbus protocols used across the process industries. These include Highway Addressable Remote transducer (HART) fieldbus, Foundation Fieldbus, Profibus, Modbus and others. The application segments examined in this research study were oil and gas, chemical and petrochemical, power generation, food and beverages, pharmaceutical and other process industries. Bond (2010) further indicated that European market for fieldbus devices in the process industries will grow from $448 million in 2008 to more than $760 million in 2015. This positive trend is driven by the demand for real-time data and increased plant availability.

Control Systems

Figure 5 shows the block diagram of a closed-loop control system. The plant (system) block represents the process such as a DC motor to be controlled. The plant input is represented by the signal U and plant output is represented by the signal Y. The sensor block measure the plant output such as speed. The actuator block changes plant input such as DC motor voltage based on the control signal sent by the controller block. The objective of the controller is to make the output such as speed follow the desired set-point value of R such as 1000 rpm (Dorf, 2010).

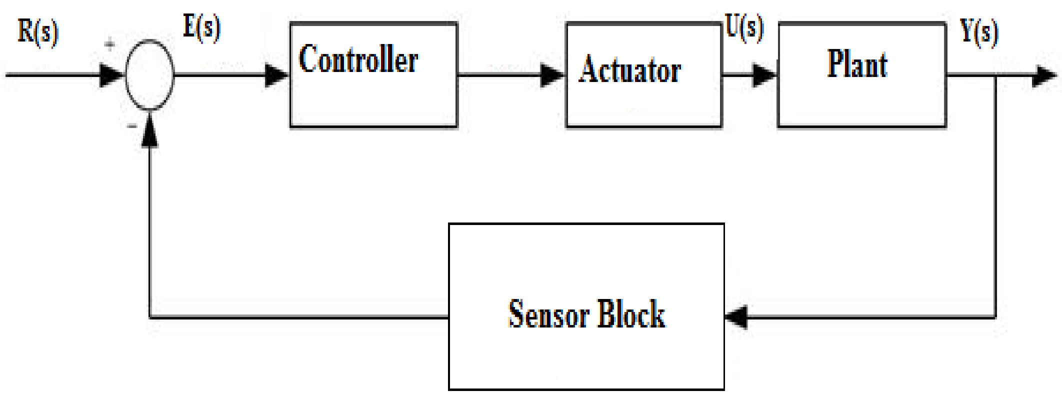


Figure 5. Closed-loop Control System.

Plants (System) can be mathematically modeled for analysis and design purposes. These modeling techniques for linear systems include Laplace Transform based transfer functions. A second-order system such as a dc motor transfer function is shown in the following equation:

$$G(s) = \frac{Y(s)}{U(s)} = \frac{as+b}{s^2+cs+d} \qquad (1)$$

where G is the transfer function, U is the input and Y is the output of the process. a, b, c, and d are process constants.

There are different types of controllers such as PID. A typical structure of a PID control system is mathematically described by

$$u = K_p e + K_I \int edt + K_D \frac{de}{dt} \qquad (2)$$

where Kp is Proportional gain, K_I is Integral gain, and K_D is Derivative gain, e is the error which is the difference between output (y) and set-point (r). The controller gains Kp, K_I, and K_D are adjusted to achieve desired system performance.

To measure the effectiveness of the controllers, several performance measures are used. One of them is the transient response of the system output for a step input. A typical step-response is shown in Figure 6 (Dorf, 2010), which would correspond to DC motor speed in rpm on y-axis. Various performance criteria used from this step-response are peak-time, peak-value, settling-time, and rise-time. The peak time (t_p) is the time required for the response to reach the first peak of the overshoot and the peak value is the maximum value of the output, reached after application of the unit step input after time (t_p). The overshoot is the amount by which the system output response proceeds beyond the desired response. The rise time is usually viewed as the time required for the response to a unit step input rising from 10 to 90% of its final value. For under damped second-order system, the 0% to 100% rise time is normally used. For over damped systems, the 10% to 90% rise time is commonly used. The settling time is the time required for the system output to settle within a certain percentage of the input amplitude. Overshoot is the amount by which the system output response, such as DC motor speed in rpm, proceeds beyond the desired response. This dissertation used these closed-loop control system concepts for studying the performance of the systems with fieldbus networks.

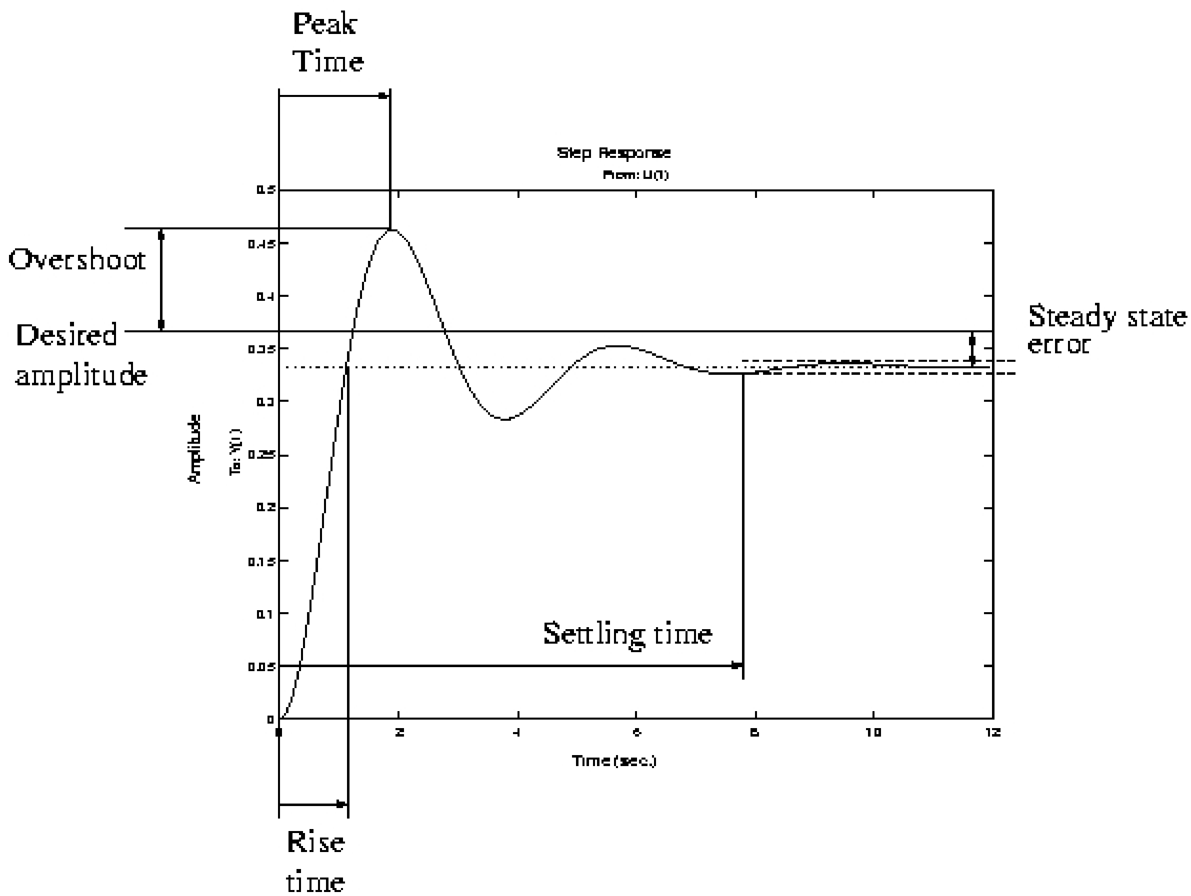


Figure 6. Performance Criteria in Control Systems (Step Response Graph).

Performance Evaluation of Control Systems with Fieldbus Networks

As explained before, the fieldbus networks provide several advantages such as reduced system wiring and improved flexibility and interoperability, among others. The introduction of fieldbus systems, however, introduces different forms of time delays among devices such as sensors, actuators, and controllers. According to Zhang et al. (2001), the existence of time delays degrades control performance in network control systems and makes closed-loop stabilization more difficult to achieve. This section contains an analysis of some of the basic problems in network control systems. These include network-induced delay, single-packet or multiple-packet transmission of plant inputs and outputs, and dropping of network packets.

Network-Induced Delay

Depending on the medium access control (MAC) protocol of the control network, the network delay can be constant, time varying, or random. MAC protocols generally fall into either random access or scheduling (Zhang et al., 2001). Carrier sense multiple access (CSMA) is used in random access networks. Scheduling networks use Token passing (TP) and time division multiple access (TDMA). The control networks such as DeviceNet based on CAN and Ethernet use CSMA protocols.

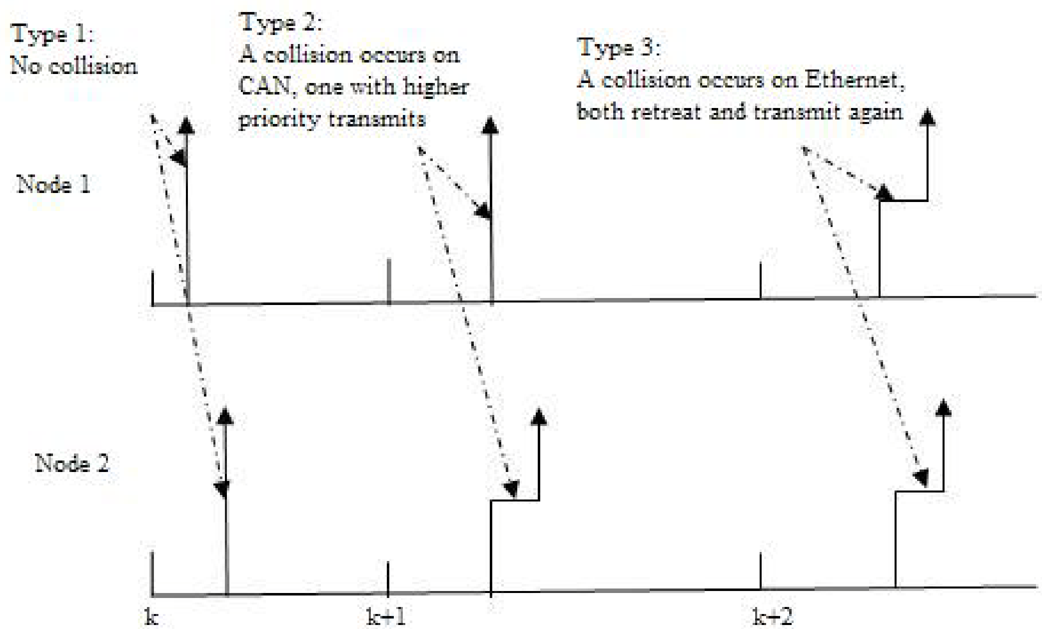


Figure 7. Timing Diagram for Two Nodes on a Network.

Figure 7 shows two nodes that are repeatedly transmitting messages with respect to a fixed time line for random access networks during different types of conditions (Zhang et al., 2001). A node on a CSMA network monitors the network before each transmission. As shown in Type 1 of Figure 7, a node begins transmission immediately when the network is idle; otherwise it waits until the network is not busy (Zhang et al., 2001). A collision occurs when two or more nodes try to transmit concurrently. The approach to resolve the collision depends on the protocol used. CAN utilizes CSMA with a bitwise arbitration (CSMA/BA) protocol. Since CAN messages are prioritized, the message with the highest priority is transmitted without interruption when a collision occurs, and transmission of the lower priority message is terminated and will be retried when the network is idle as shown in Type 2 of Figure 7. Ethernet also utilizes a CSMA with collision detection (CSMA/CD) protocol. All the affected nodes will retreat when there is a collision, wait a random time and retransmit as shown in Type 3 of Figure 7. CSMA networks are considered nondeterministic; but higher priority messages have a better chance of timely transmission when messages are prioritized as in CAN (Zhang et al., 2001).

Dropping Network Packets

One of the concerns raised in NCS is the unreliable transmission paths. This is because of the limited bandwidth and large amount of data packet transmitted over the single channel. This may result in network data packet dropout. Data packet dropout usually occurs while exchanging data among devices such as sensors, actuators, and controllers. The failure of a node to either send or receive data as well as the collision of message may further results in packet dropout. This can degrade the performance of the system. Although, some network protocols may be designed with retry mechanisms, and transmission, they can only retransmit for a limited amount of time. The packets may be dropped after the time expired. The un-transmitted message may be discarded and new packets may be transmitted when it becomes available. This enables the controller to constantly obtain fresh data for control calculation. While feedback control plants may usually still function after certain amount of data loss, it becomes very important to establish whether the system is stable when only transmitting the packets at a certain rate and to compute acceptable lower bounds on the packet transmission rate (Zhang et al., 2001).

NCS with Network-Induced Delay

The NCS model with network-induced delay is shown in Figure. 2. Two sources of delays occur in this model. This include sensor to controller (τ_{sc}) and controller to actuator (τ_{ca}). Any controller computational delay can be grouped to be either τ_{sc} or τ_{ca}. For fixed control law, thus, time-invariant controllers, the sensor to controller delay and controller to actuator delay can be put together as $\tau = \tau_{sc} + \tau_{ca}$ (Zhang et al., 2001). This is shown in Figure.8 for analysis purposes.

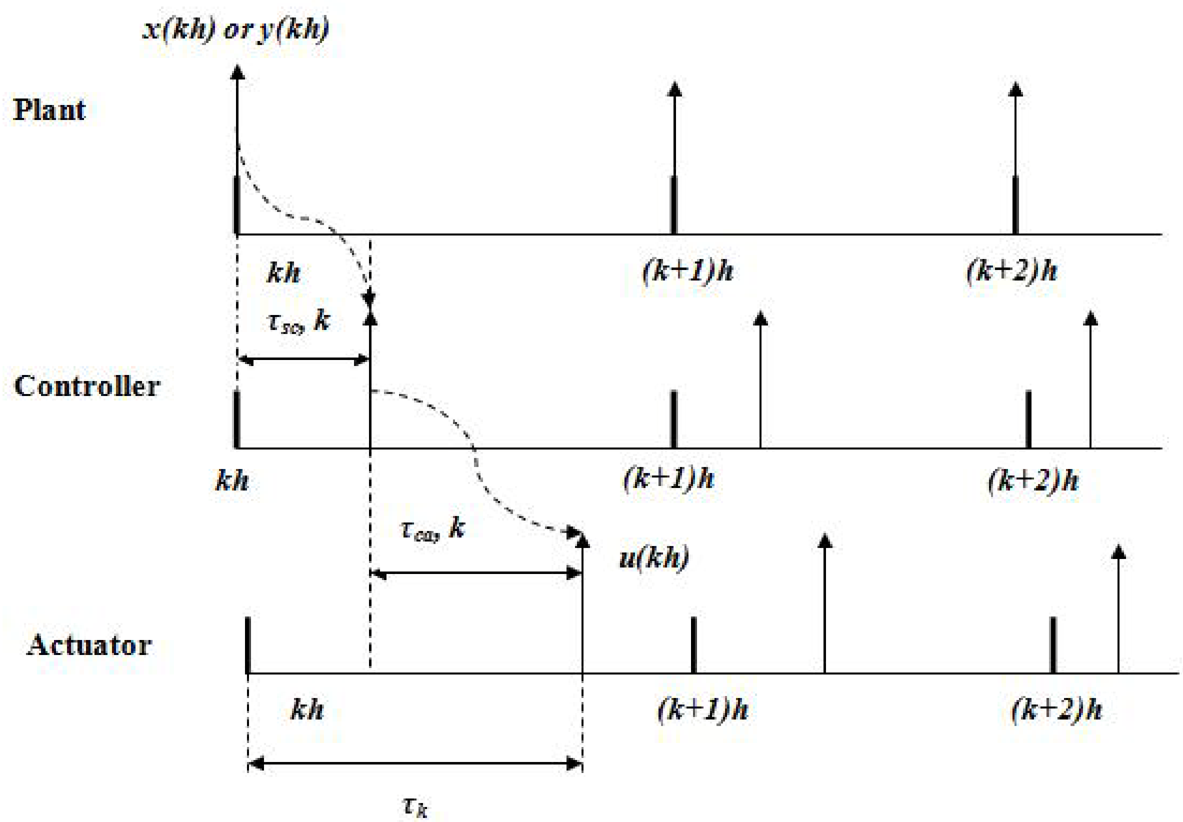


Figure 8. Network-Induced Delay.

Recently, there have been several theoretical results reported in the literature that address these time delays and other aspects of networked control systems (Zhang et al., 2001). For instance, Abdel-Ghaffar et al. (2008) utilized a pole-placement based control algorithm to analyze delay problems in ISA Fieldbus. Pang et al. (2006) analyzed the temporal characteristics of communication and computation tasks and the configuration of the function blocks to allow the control interval to be shortened in a Foundation Fieldbus based control systems. The delays associated with the use of foundation fieldbus (FF) H1 networks within control loops were investigated by Rankin et al. (2009). Analytical and experimental evaluations were performed with a test loop using hardwired analog channels as a benchmark. Three segments of FF-H1-network-induced delays were identified, their analytical models were developed, and suggestions to potentially reduce the delays were provided. As explained before, in these Foundation Fieldbus (ANSI/ISA-S50.02, 1997) based networks Link Active Scheduler plays an important role in the timing issues. The time delay aspects in the FF-based networks can be handled through proper scheduling as discussed in these papers.

Lee et al. (2002) presented the structure of network control system by using a Profibus-DP network and investigates the cause of network induced delay. They further compared the performance of the NCS to the traditional control system. Li et al. (2011) investigated the delays associated with the use of Profibus-PA networks within control loops.

A study of shipboard modular, networked based control system to evaluate their performance in the context of communication between controllers, distributed input/output and data acquisition systems were perform by Penera et al. (2011). Performance evaluation was analyzed by means of experimental measurements on a test bed that replicates a shipboard communication network and related timing analysis. Three common switched Ethernet topologies were also examined from the perspective of end-to-end packet delay, and network failover (reconfiguration) delay. Implementations of delay compensator approach using non-linear techniques were presented by Antunes et al., (2010). The delay compensator was implemented through a model that described the effect of the sampling to actuation delay in terms of the control signal. This model was then used to compensate the control signal according to the delay at the actuation moment.

Tipsuwan et al. (2003) surveyed recent articles in the NCS area. Special issues on NCS were edited by Antsaklis et al. (2007). Some researchers analyzed the effect of the time delays on traditional controllers such as Proportional, Integral, and Derivative (PID) controller (Tipsuwan et al, 2003), while others gave new control designs that take the delays into account. Some of the analysis results give bounds on the allowable delays to preserve stability of the networked control systems. Azimi-Sadjadi (2003) studied the stability of networked control systems in the presence of packet losses. Liu and Goldsmith (2004) studied the effects of wireless network medium access control protocols on the performance of the networked control systems.

Wang et al. (1992) provided a model for computing the maximum and expected delays for CAN. Network induced time delay was stochastically modeled by Morales-Menendez et al. (2009) for CAN. A method to calculate CAN message response times was given by Tindell et al. (1995). A probabilistic approach to determine response time distribution for messages in CAN was given by Kumar et al (2009). Schedulability analysis for CAN was discussed by Davis et al. (2007). The authors discussed the possible impact on commercial CAN systems designed and developed using flawed schedulability analysis and made recommendations for the revision of CAN schedulability analysis tools. Li et al. (2011) investigated the delays associated with the use of Profibus-PA networks within control loops. The existing delay analyses used analytical and stochastic methods to establish relationships for delays. This dissertation used statistical analysis methods to study the effect of various CAN parameters on network delays.

Hedjar et al. (2006) discussed the analysis of a networked control system whose sensors and actuators exchange information with a remote controller over a shared network. The design of the controller was done in discrete domain without delay. The contribution of this work was the use of Jury test to provide the range of the delay that maintained the stability of the augmented system in closed loop. Double integrator system was then used in simulation to show the performances achieved in closed loop with delay. Tipsuwan (2003) developed a control methodology to handle network induced delay effects using optimal gain scheduling on existing controllers. The proposed gain scheduling technique adapts controller gains externally by modifying a controller output to enable the controller for uses over a data network.

Ji (2006) developed a framework for modeling, design, stability analysis, control, and bandwidth allocation of real-time control over networks. The author proposed a novel network control system model in which the effects of the network induced time delay, data-packet loss, and out-of-order data transmission were all considered. Simple algorithms based on model-estimator and predictor and timeout-scheme were proposed to compensate for the network-induced time delay and packet loss simultaneously. Zhang et al. (2001) presented network control system models with network-induced delay. The authors analyzed their stability using stability regions and a hybrid systems technique. They further discussed methods to compensate network-induced delay and presented experimental results over a physical network.

Lian et al. (2005) discussed the comparison among Ethernet, Token-based, and CAN protocols. The authors described the medium access control mechanisms of these protocols, which were responsible for satisfying both the time-critical/real-time response requirement over the network and the quality and reliability of communication between devices on the network. For each protocol, they studied the key performance parameters of the corresponding network when used in a control situation, including network utilization, magnitude of the expected time delay, and characteristics of time delays. Simulation results were presented for several different scenarios.

However, there are no generic performance analyses that have been applied on every NCS. Most of these analysis and techniques are subject to network configurations, network protocols, assumptions, and control techniques used. Statistical analysis methods such as ANOVA was used to determine the impact of baud rates, bus load, message length on transmission of CAN messages to ascertain the effect on message delays in this dissertation. The performance of network control system/fieldbus network was then analyzed with respect to delays using MATLAB/Simulink software to obtain design parameter values such as controller gains for PID.

Summary of Chapter

A review of relevant literature for this research study was presented in this chapter. The first section discussed the characteristics of fieldbus and industrial networks and how they relate to ISO/OSI model. The second section discussed the impact and adaptation of fieldbus networks based on a study conducted by O’Brian (2005). The third section gave background about control systems. The fourth section provided details of performance evaluation of control systems with fieldbus networks.

CHAPTER 3

METHODOLOGY AND FINDINGS – ANALYSIS OF NETWORK DELAYS

This chapter provides a description of the methodology for achieving the part of the objectives of this dissertation that relate to the analysis of the effects of different CAN parameters on the network delays. The research questions and hypothesis related to this part of the dissertation study are restated. The analysis was carried out using the data collected from the simulation of an automobile systems using CANoe software. Details of the simulation are presented. Descriptive statistics and ANOVA were used for the analysis.

Restatement of the Problem and Objectives

The problem of this research was to study the effects of fieldbus network induced delays on control systems. The causes of delays and how these delays are affected by CAN parameters such as baud rate, bus load, and message length was initially investigated using CANoe simulations of an automobile system. The effect of fieldbus network induced delays on control system performance such as step-response for different controllers was further studied using MATLAB/Simulink software for an electrical system. The step-response performance criteria studied was peak-value, settling-time and rise-time. Controllers such as Proportional, Integral, and Derivative (PID), State Feedback was considered. Controller designs to accommodate these delays were investigated.

Research Questions

In order to achieve the objective of the problem statement, this chapter was designed to help answer the following research questions:

1. Does inclusion of fieldbus networks such as CAN in automobile systems induce network delays?
2. What are the effects of CAN parameter such as "bus loads", "baud rate", and "message length" on network delays?

Research Hypotheses

The following were the hypotheses for the study:

Effect of CAN Baud Rates

4. H_{01}: $\mu_{50} = \mu_{100} = \mu_{200}$

   There are no statistically significant differences in the mean values of network delays for signal transmission baud rates 50 kbps, 100 kbps and 200 kbps.

   H_{A1}: $\mu_{50} \neq \mu_{100} \neq \mu_{200}$

   There are statistically significant differences in the mean values of network delays for signals transmission baud rates 50 kbps, 100 kbps and 200 kbps.

Effect of CAN Message Length

5. H_{02}: $\mu_{8} = \mu_{6} = \mu_{1}$

   There are no statistically significant differences in the mean values of network delays for signals with message lengths 8 bytes, 6 bytes, and 1 byte.

   H_{A2}: $\mu_{8} \neq \mu_{6} \neq \mu_{1}$

   There are statistically significant differences in the mean values of network delays for signals with message lengths 8 bytes, 6 bytes, and 1 byte.

Effect of CAN Bus Loads

6. H_{03}: $\mu_{22\%} = \mu_{17\%}$

   There are no statistically significant differences in the mean values of network delays for busload of 22% and 17%.

   H_{A3}: $\mu_{22\%} \neq \mu_{17\%}$

   There are statistically significant differences in the mean values of network delays for busload of 22% and 17%.

Controller Area Network (CAN)

CAN is a network protocol that permits Electronic Control Units (ECU) in a system to communicate efficiently with each other. The CAN protocol, however, is designed for short messages. It is generally used for sending signals to trigger events, such as to lock seat belts during forceful braking, and measurement values, such as temperature and pressure readings (CAN in Automation, 2009). It is also the standard for high-speed, mission critical, real-time control networks in many systems and machines. The basic structure of CAN based network is shown in Figure 9. It is very simple to add additional stations to an already existing CAN network without making any hardware or software modifications to the existing stations, provided that the new stations are purely receivers. The CAN based network structure supports the concept of modular electronics. This is because the data transmission protocol does not require physical destination addresses for the individual components. It also permits multiple receptions (broadcast, multicast) and the synchronization of distributed processes. For instance, measurements needed as information by some of the controllers can be transmitted via the network, in such a way that it becomes redundant for each controller to have its own sensor.

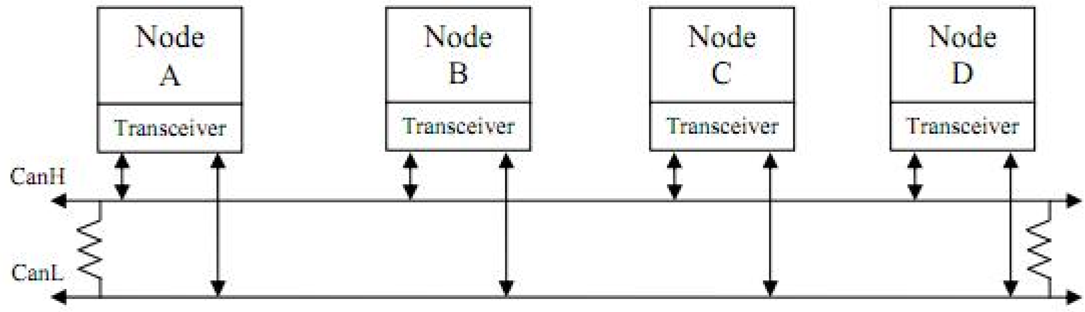


Figure 9. Basic Structure of CAN Based Network.

CAN is a serial, multi-master, and multicast protocol (CAN in Automation, 2009). This indicates that, when the bus is free, any node can send a message, thus, multi-master, and all nodes may receive and act on the message, thus, multicast. The node that initiates the message is called the transmitter. Any node not sending a message is called a receiver. Messages are assigned static priorities, and a transmitting node will remain a transmitter until the bus becomes idle or until it is superseded by a node with a higher priority message through a process called arbitration. A CAN message may contain up to 8 bytes of data. A message identifier describes the data content and is used by receiving nodes to determine the destination on the network. The nodes select those messages that are relevant and ignore the rest. A CAN system is able to transmit up to 7600 8-byte messages or 18,000 trigger signals per second. Bit rates up to 1 Mbit/s are possible in short networks distance of 40m without repeaters. Longer network distances reduce the available bit rate for instance up to 125 Kbit/s at 500 meters (CAN in Automation, 2009).

A CAN bus node usually create a Data Frame. The data frame is produced when the node wishes to transmit data. It is also produced when it is requested by another node. The Data Frame starts with a leading or dominant Start of Frame (SOF) bit for the synchronization of all nodes as shown in Figure 10. The SOF bit is followed by the Arbitration Field. This field contains information about message’s priority on the CAN bus. It is applied when two or more messages are sent or requested concurrently. The message with the highest priority is usually sent on the bus, while the less priority message waits for its turn to be transmitted. Besides, it works as an identifier. It is utilized by controllers to filter the messages.

The next field is the Control Field. This field primarily details the number of bytes of data that are contained in the message. The data field generally contains zero to eight bytes of data. This is up to 64 bits. The details of this field is set by the transmitting ECU, its inner setup, however, is predefined. Signals in the data field remain the same. This means that, the bit distribution over the data field is fixed, but its values may change. For instance, in a message, with a data field of eight bytes, sent by the instrument panel, one specific byte and two specific bits may contain information whether a button is pressed or not. For example, “00” could mean button not pressed while “01” could mean button pressed. “10” and “11” could indicate if the button was not accessible or the occurrence of an error. The other message fields are used for error check and acknowledgement of correctly received message. The Cyclic Redundancy Check (CRC) Field is employed to detect potential transmission errors. It contains a 15-bit CRC sequence that is completed by a recessive CRC delimiter bit. The transmitting node sends out a recessive bit during the Acknowledgement (ACK) Field. Any node that has received an error free frame acknowledges the correct reception of the frame by sending back a dominant bit. The recessive bits of the End of Frame end the Data Frame. Between two frames there must be a recessive 3-bit Intermission field. CAN data frame is shown in Figures 10 (Bosch, 2006).

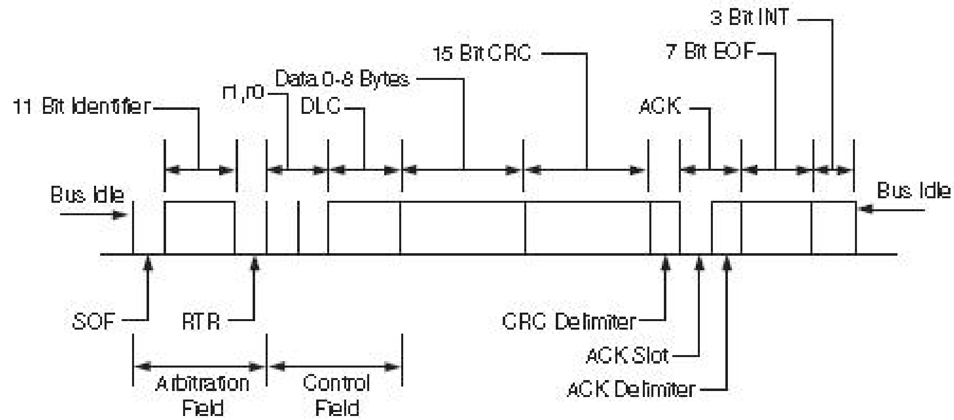


Figure 10. CAN Data Frame (courtesy of Bosch, 2006).

CANoe Software

CAN open environment (CANoe) software provides a universal development, testing, and analysis environment for CAN bus systems. It was created by Vector Informatik GmbH, and permits simultaneous network development of systems in the CAN, LIN, MOST, FlexRay, J1587 and many other CAN-based protocols. It was designed primarily to model both electronic control unit (ECU) nodes on a network as well as the network that integrates them (Vector[CANoe], 2009). CANoe provides two major windows. These are the “measurement setup” and “simulation setup” windows. The former is used to configure data and statistics that are needed to be monitored while the latter is used to configure and run the entire simulation.

The “measurement setup” has several sub windows shown in Figure 11. These include: “trace window” which allows the user to monitor messages with timestamp, channel they use, their id, name in the database, their DLC and the data bytes; “data window” which is used to display physical values and bar type representation of the messages/signals that are being monitored; “graphics window” which provide graphical representation for the signal values against time; “statistics window” which allows the user to view a histogram of how many messages of a particular ID have been sent and received; and the “write window” which provides the area where the user can output and print system messages and any other needed information.

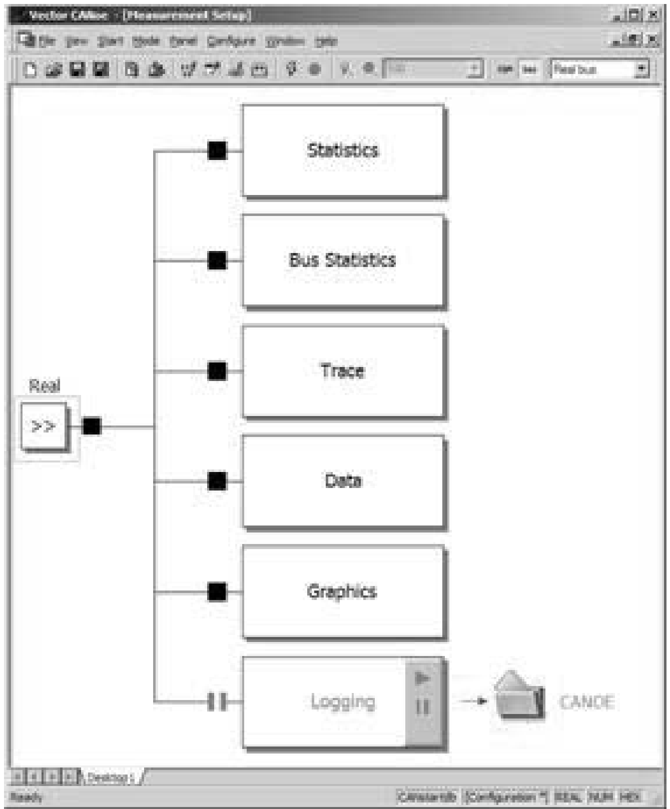


Figure 11. Measurement Setup in CANoe.

A typical CANoe system model can be created in the “simulation setup” using the following steps:

1. Create the database with messages, signals and environment variables. CANdb++ editor is used to create these databases.
2. Create the network node periphery, which includes control panels. Panel Editor is used to create these panels.
3. Create the network node model in C-like language called CAPL.

The database provides the environmental variables for describing the I/O interface among the nodes and their peripheries. Each periphery element is then “wired” to an environmental variable. It is also connected to the CAPL program for the network node. CANoe differentiates between discrete and continuous variables. Switch positions can be represented as discrete environment variables. With continuous environment variables, dimensions such as temperature or engine RPM can be described. The control panels provide a user-friendly interface to the environment variables. The user can therefore create the panels separately with the help of the Panel Editor. During the simulation run, values of environment variables can be displayed (lamps, counters) and interactively modified (switches, potentiometers). These three-steps of CANoe simulation creation and its running to display various measurements are described below through two systems.

Temperature Control Process

This process involves a simple CANoe configuration with two network node model and its associated peripherals (control panels). It is a simulation for a temperature control process with sensor, actuator and controller communicating on the CAN bus. The temperature sensor is simulated by a slider and its value is displayed. The controller compares the temperature with a set-point. If the temperature is above the set-point a fan actuator is turned-on, otherwise it is turned-off.

Step1: In this example there are exactly three peripheral elements: Temperature slider and a display unit at the first node and a fan as the second node. The three environmental variables are represented in the database as evDisplay, evFan and evTemperature shown in Figure 12.

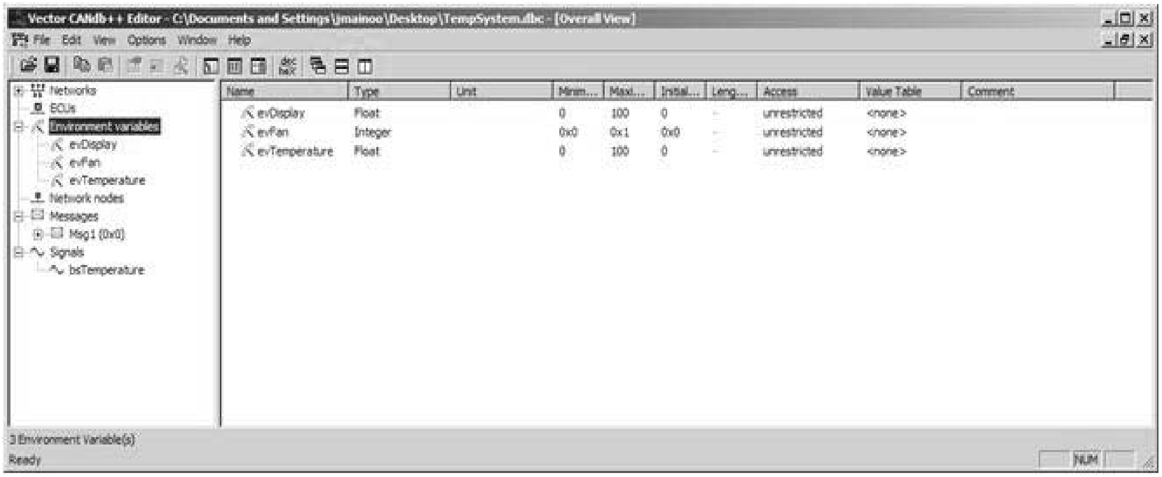


Figure 12. Environmental Variables in the Database.

Step 2: Using the panel editor provided in CANoe, panels representing various nodes' peripheries, shown in Figure 13, are created. The Temperature slider, Display unit and the Fan each represent the various environmental variables. The panels are then integrated into the CANoe configuration.

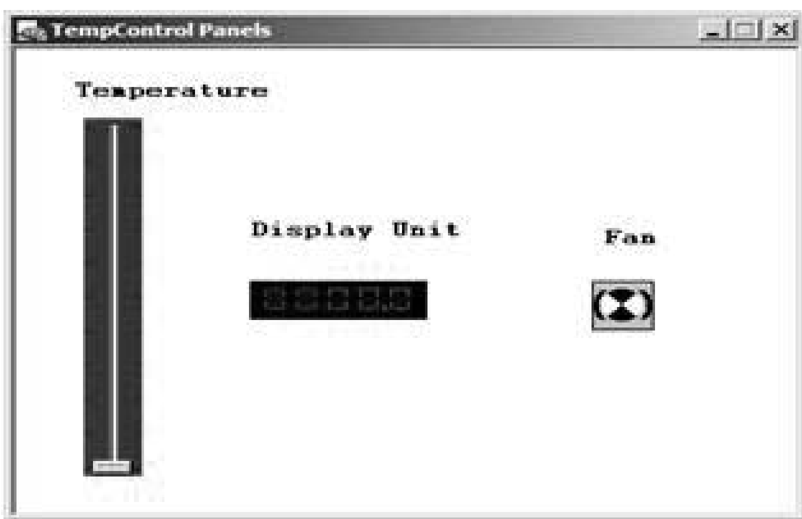


Figure 13. Panels in CANoe.

Step 3: The network node models are created in CANoe's simulation setup. There are two network nodes in this example as shown in Figure 14. The CAPL programs for each node, shown in Figure 15, simulate the functionalities of the various nodes. The first CAPL program belongs to the node corresponding to the temperature slider and the display unit. When the slider position changes, the program gets the new temperature value and immediately outputs it on the CAN bus. The display unit reacts to this message. Its CAPL program reads the value of the bus signal for the input temperature from the slider and then outputs the value on the indicator display at its peripheral. The second network node's CAPL program then compares the signal generated for the new temperature with the set point of 70 degrees. The program turns on a Fan, when the input temperature is greater than the set point as shown in Figure 16. The Fan goes to the off mode when the signal or temperature value received on the bus is less than the value of the set point.

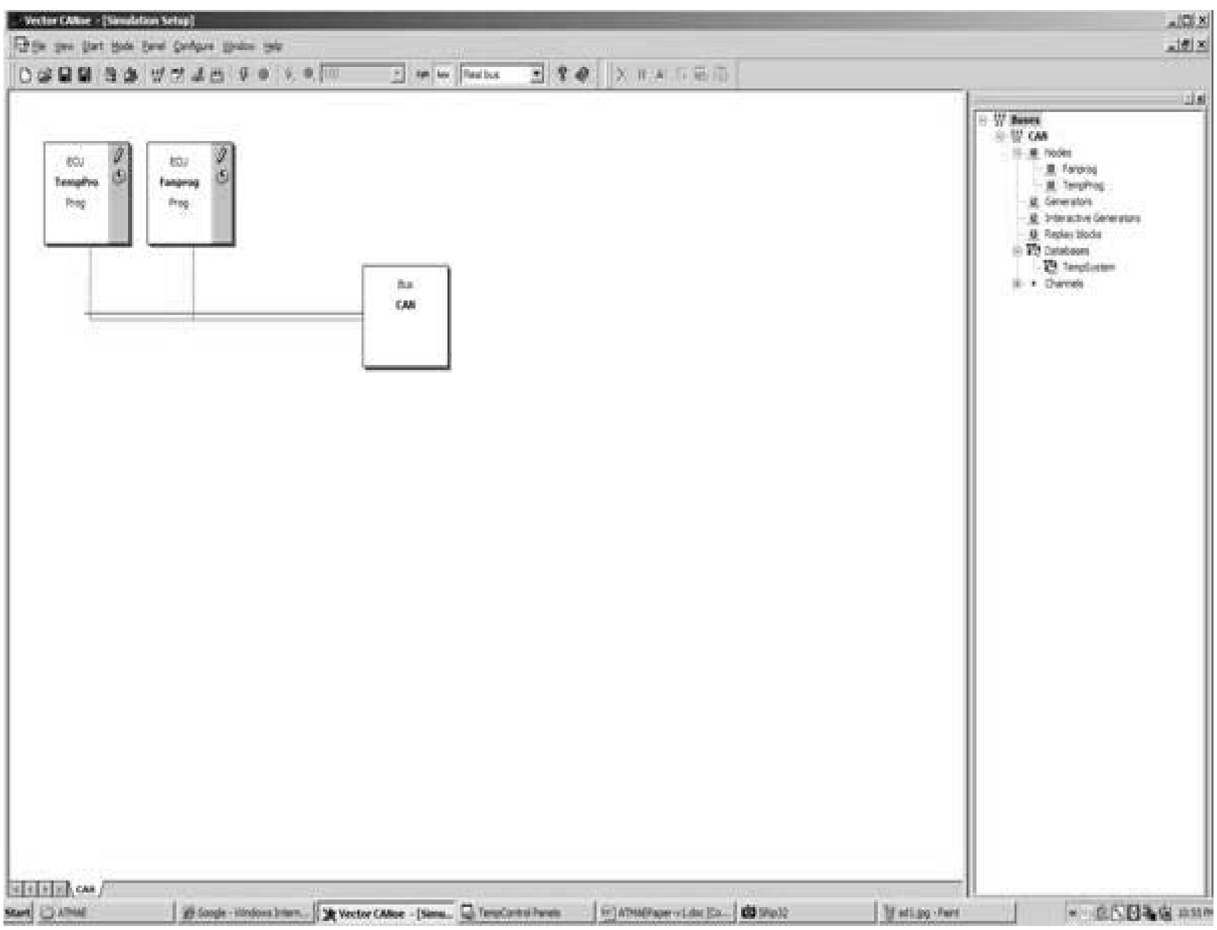


Figure 14. Network Nodes in the Simulation Setup.

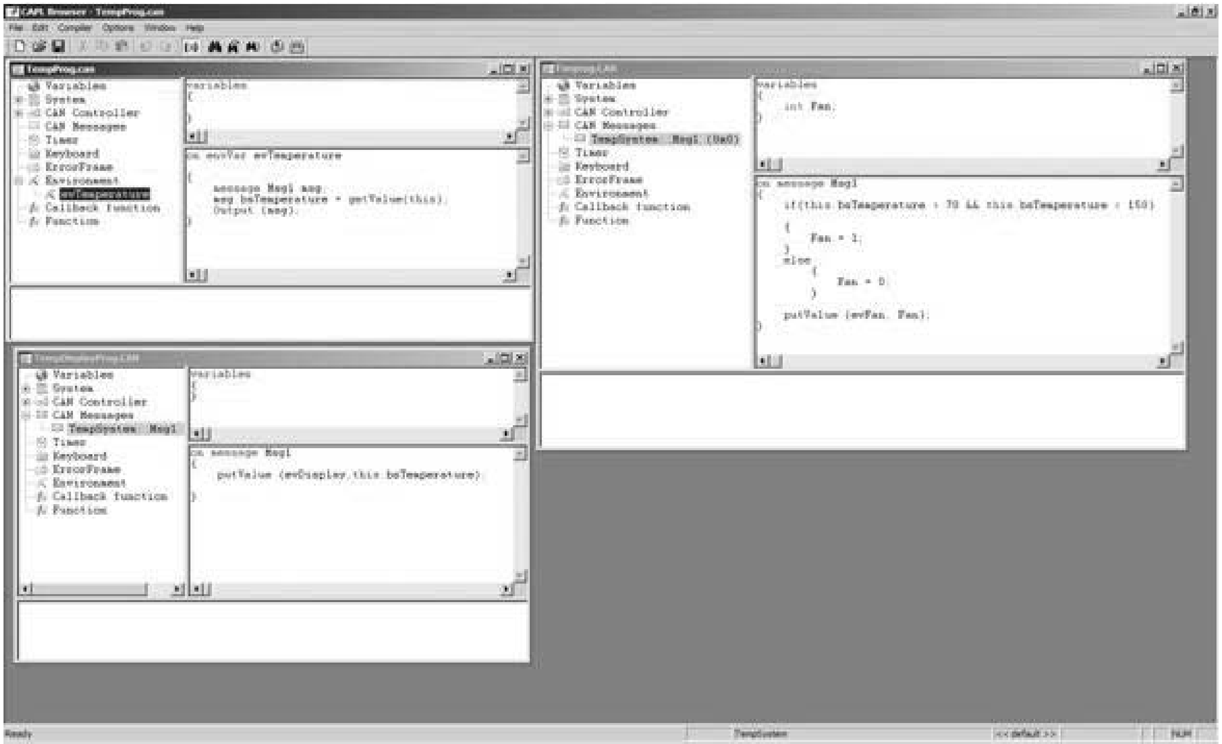


Figure 15. CAPL Program.

Once the system is created, the CANoe users have the ability to simulate the network behavior, observe the bus load, and determine the required performance of the hardware being developed. Evaluation windows are used to analyze data in CANoe environment. These windows are shown in Figure 16 for the temperature control process. The data that reached the trace block of the measurement setup are displayed on the trace window as CAN messages in bus oriented format. Similarly the “statistics window,” shown in Figure 16, offers bus related information that reached the statistics block. It gives the transmitted messages per second coded by identifiers. A statistics report is also generated for at least one acquisition range which is output to the “write window” at the end of each measurement. The write window shows the total number of messages for each identifier, the mean value, standard deviation, and the minimum and maximum for the recorded transmit interval. The “bus statistics window” also provides an overview of the bus data traffic. The total frequencies of data, remote, error as well as overload frames, bus loading and the status of CAN controller are displayed on this window.

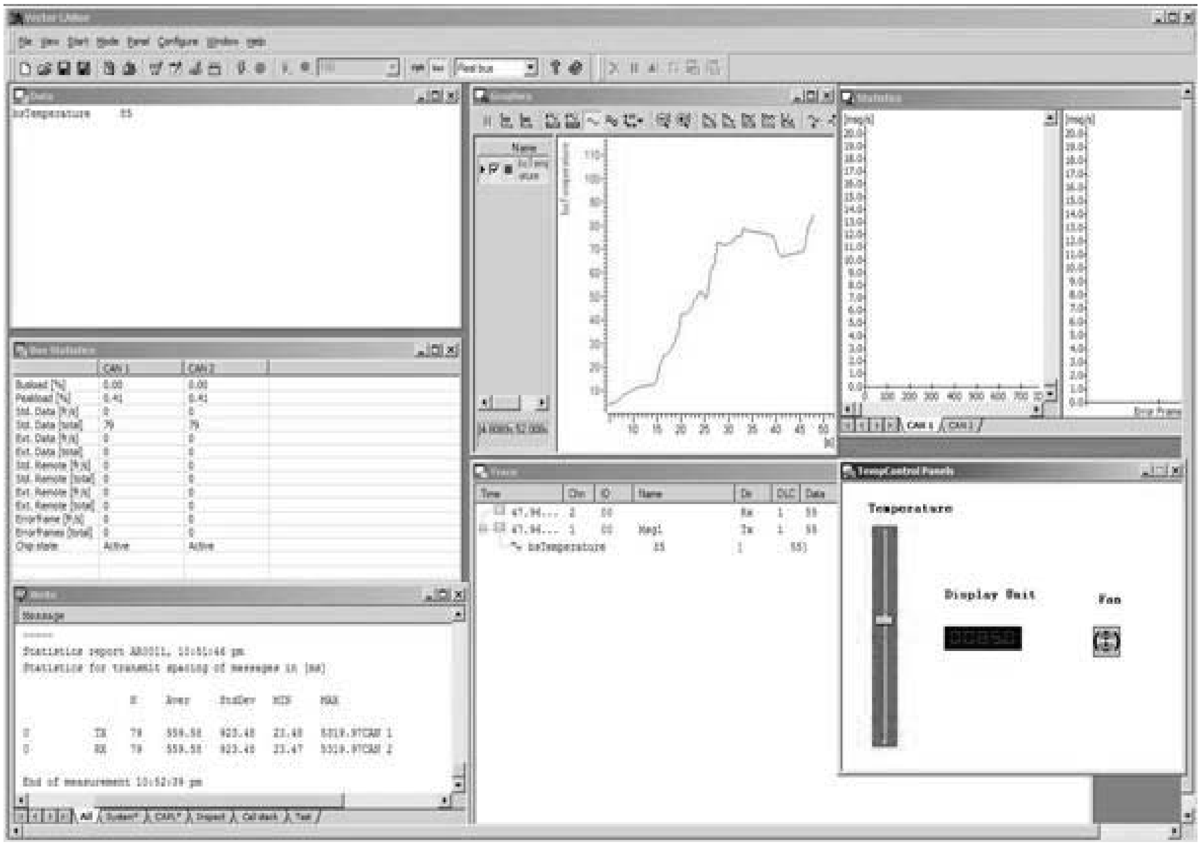


Figure 16. Various Panels and System Windows in CANoe (Temperature Control).

Automobile System

The above three-step process of CANoe was also followed in a more involved automobile system simulation (Vector[CANoe], 2009). The panels of the automobile dashboard and various other consoles are shown in Figure 17 and part of the network nodes in the simulation setup are shown in Figure 18. Figure 19 shows the CAPL program for Dashboard ECU. The partial list of messages in the database listed with their Name, ID, DLC, etc. is shown in Figure 20. These messages include ABS data with identifier c9 and length 6 bytes, Engine Data with identifier 64 and length 8 bytes, and Gear Box Info with identifier 3fc and length 1 byte. The Engine Data message consists of signals such as EngineSpeed with 16 bits length, EngineTemp with 7 bits length, IdleRunning with 1bit length, PetrolLevel with 8 bits length, EngForce with 16 bit length, and EngPower with 16 bit length. Similar signals are linked to the other messages c9 and 3fc. This automobile simulator is run with several conditions to generate the data required for the performance analysis of the system. The results of the analysis are presented in next section.

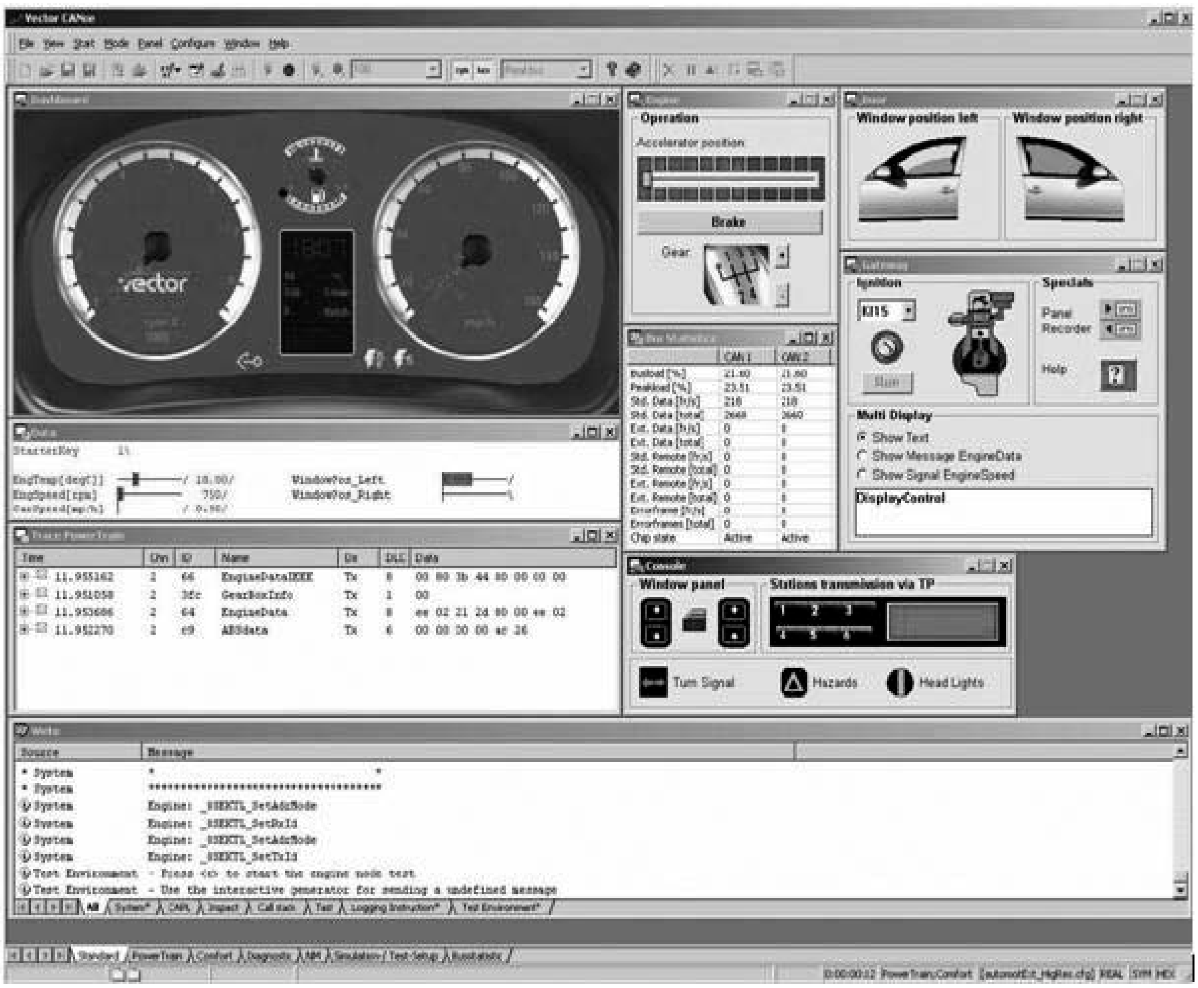


Figure 17. CANoe Simulation of an Automobile Application (courtesy of Vector [CANoe], 2009).

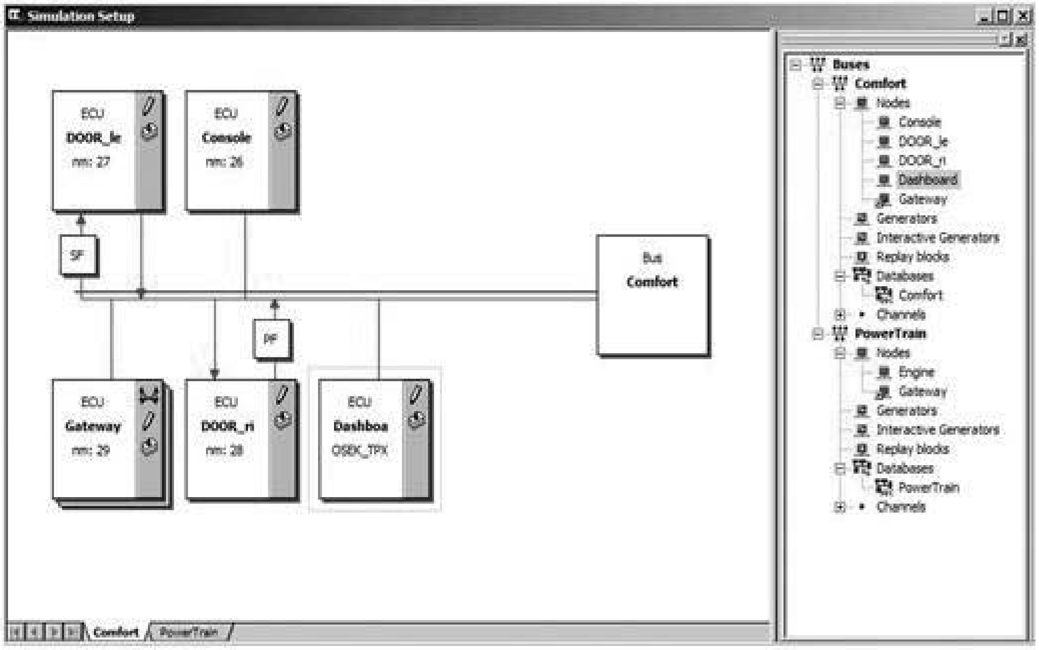


Figure 18. Simulation Setup of an Automobile Application "Comfort Nodes" (courtesy of Vector[CANoe], 2009).

.

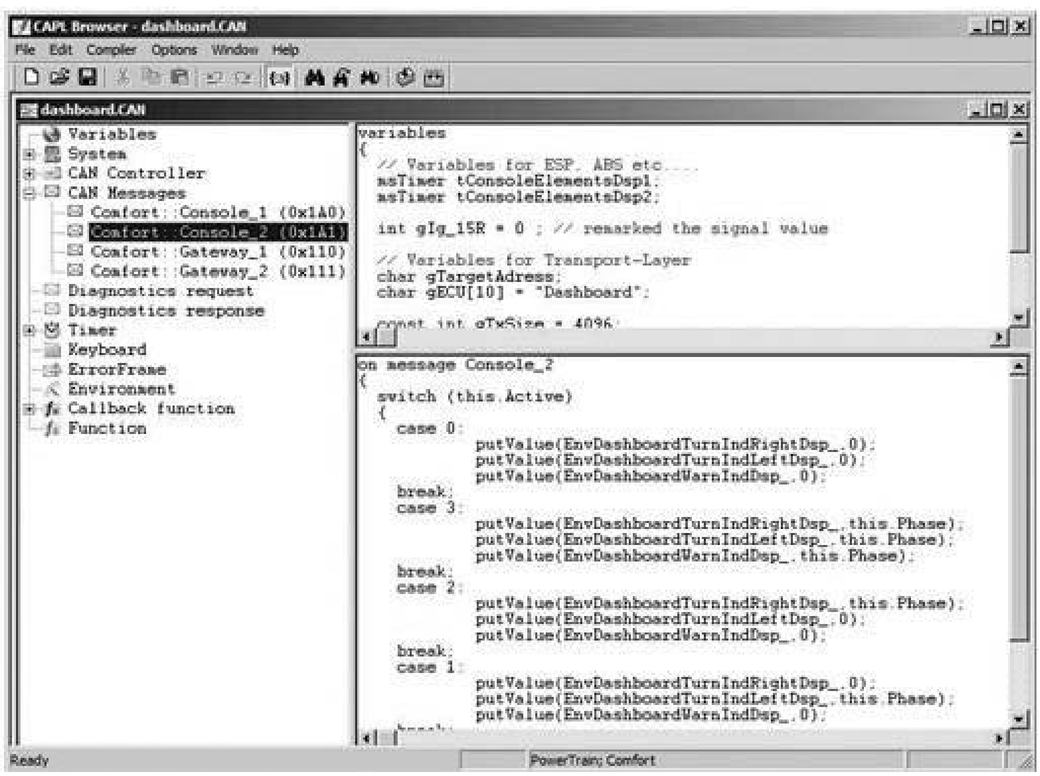


Figure 19. CAPL Program for Dashboard ECU (courtesy of Vector[CANoe], 2009).

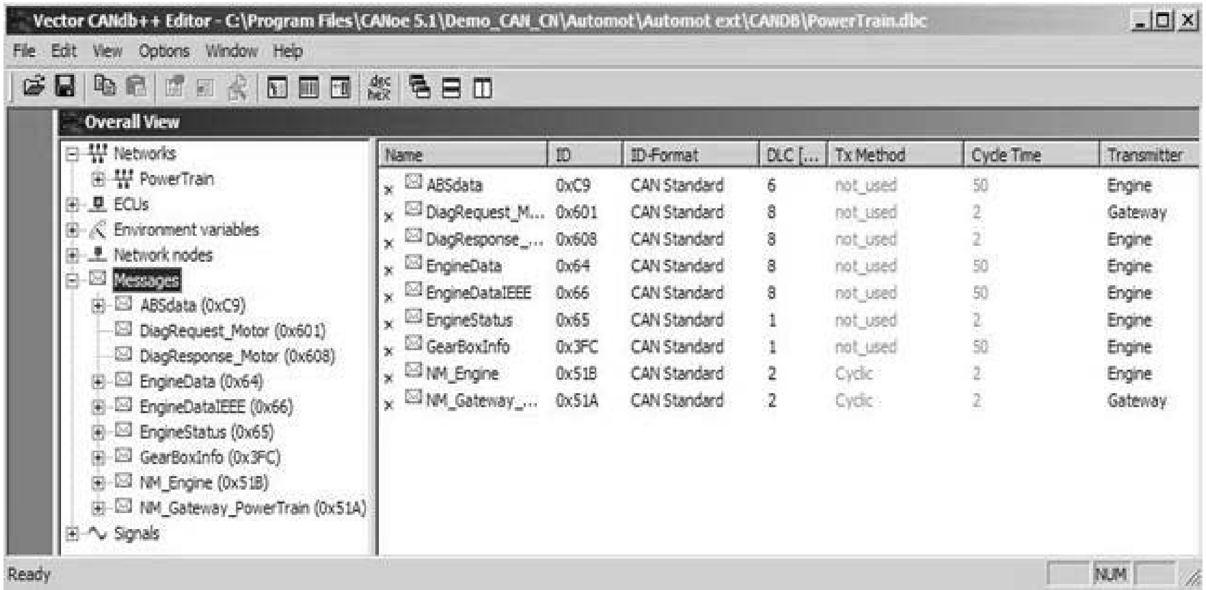


Figure 20. Partial Database of Automobile Simulation Network (courtesy of Vector[CANoe], 2009).

Technique Used for Data Collection

The variable of interest for this part of the study was the delay time which was measured in millisecond. The factors (also referred as CAN parameters) considered were baud-rate (transmission speed), bus-load (% of bus activity) and message-length (0 to 8 bytes). Equipments and applications that were used for data collection include

1. Dell desktop computer with windows XP operating systems
2. CAN Controller (PC CANcard XL)
3. CANoe Software

CANoe was configured as a data source to transmit data to the CAN controllers. It assumes the role of both sender and receiver. The data was gathered from simulation of automobile system discussed in previous Section. This simulation allowed various data sources in CANoe to place information or messages on the bus “cyclically” and by “event driven”. Each individual message had a unique identifier. Messages received by the controller gets the attribute R_x and a time stamp from the card's clock when they were received. The driver returns the time of the transmit request assigned to the CAN microcontroller. The message to be transmitted was assigned an attribute T_xR_q. After the successful transmission, the message was returned with the actual time of transmission and the attribute T_x, so that the transmit messages were displayed and logged in the Trace windows. The time between the message's T_x attribute and T_xR_q attribute was the network delay time (Vector[CANoe], 2009). This was the time that the CAN controller placed a message completely on the CAN bus. The network delay times were observed for all the messages at different baud rates, bus loads, and message lengths.

Figure 21 shows CAN bus activity for the automobile system executed at 100 kbps. For message identifier 64 (Hex) and c9 (Hex) the data has been highlighted for one sample. The delay time (T_x-T_xR_q) for message with identifiers 64 and c9 were observed to be 0.001141s (1.141 ms) and 0.00097s (0.97 ms), respectively. Samples of delay times were collected for all the messages at different CAN parameters values such as baud rates, bus loads and message lengths during simulation runs. The data was analyzed using statistical methods (Norusis, 2009) to study the effects of various CAN parameters on the delays and to test various hypotheses state in the beginning of this chapter.

Time	Chn	ID	Name	Dir	DLC	Data
18.280309	1	65	EngineStatus	TxRq	1	01
18.280869	2	65		Rx	1	01
18.280879	1	65	EngineStatus	Tx	1	01
18.273700	1	66	EngineDataIEEE	TxRq	8	4e 63 9f 45 00 80 3b 45
18.274819	2	66		Rx	8	4e 63 9f 45 00 80 3b 45
18.274829	1	66	EngineDataIEEE	Tx	8	4e 63 9f 45 00 80 3b 45
18.278465	1	41b	NM_DOORleft	TxRq	4	1d 12 01 ff
18.279269	2	41b		Rx	4	1d 12 01 ff
18.279279	1	41b	NM_DOORleft	Tx	4	1d 12 01 ff
18.279309	1	c9	ABSdata	TxRq	6	10 01 00 00 49 27
18.280269	2	c9		Rx	6	10 01 00 00 49 27
18.280279	1	c9	ABSdata	Tx	6	10 01 00 00 49 27
18.275458	1	64	EngineData	TxRq	8	ec 13 3f 24 b8 0b 14 37
18.276589	2	64		Rx	8	ec 13 3f 24 b8 0b 14 37
18.276599	1	64	EngineData	Tx	8	ec 13 3f 24 b8 0b 14 37
18.270957	1	1a0	Console_1	TxRq	4	41 87 35 01
18.271749	2	1a0		Rx	4	41 87 35 01
18.271759	1	1a0	Console_1	Tx	4	41 87 35 01

Figure 21. Trace Window Showing Various Bus Activities Executed at 100 kbps.

Results and Data Analysis of Network Delays

Descriptive statistics was used to provide explanations on the characteristics of the sample data (Keller, 2008) and (Norusis, 2006). This involves arranging, summarizing, and processing sets of data to be generated from CANoe software. The effects of baud rate, message length and bus loads on message delay were studied to answer research questions 1 and 2 and to test the hypothesis 1, 2, and 3. Therefore, the dependent variable for this study was the network delay time, and the independent variables were baud-rate, bus-loads, and message length. One-way ANOVA was used to analyze the effect of the respective factors on the time delays. The data was re-coded into SPSS before the analysis. Alpha was defined as the probability that, according to null hypothesis, a statistical test will generate a false-positive error, thus, affirming a non-null pattern by chance. Conventional methodology for statistical testing is, in advance of undertaking the test, to set a nominal alpha criterion level (often 0.05). The outcome was classified as showing statistical significance if the actual alpha (probability of the outcome under the null hypothesis) was no greater than this nominal alpha criterion level (Norusis, 2009). The alpha level was specified in this analysis to be 0.05. The usual assumptions (Norusis, 2009) of statistical test procedure related to ANOVA are followed in this work.

Effect of Baud Rates on Network Delays

Descriptive Analysis of Delay Time for Different Baud Rates

In this study, 208 samples of delays for engine data signal with identifier 64 were collected at three different baud rates (50 kbps, 100 kbps, 200kbps) and analyzed using statistical package SPSS. Descriptive statistics of the dependent variables were analyzed to examine their distributions. The data in Table 2 provided descriptive statistics including the mean, standard deviation and 95% confidence intervals for the dependent variable (network delay coded as BaudRate_Delays for SPSS purpose) for each separate group (50 kbps, 100 kbps and 200 kbps). The mean time delays for 50 kbps, 100 kbps and 200 kbps baud rates were 2.3054, 1.1628 and 0.7626 respectively. The standard deviation for 50 kbps was 0.02689; that of 100 kbps was 0.0571; and 0.20078 for 200 kbps. The skewness value for 50 kbps baud rate delay time was 0.000. The mean of the data was almost equal to the median and the distribution of this data was symmetrical around the mean. The kurtosis value of -0.143 made the distribution flatter than a normal distribution with a wider peak. This data distribution is shown in Figure 22. The skewness value for 100 kbps was 3.575. This had a right skewed distribution. Most of the data values were concentrated on the left of the mean, with extreme values to the right. The kurtosis value of 26.224 makes the distribution sharper than a normal distribution, with values concentrated around the mean and thicker tails. This means high probability for extreme values. This was shown in the histogram in Figure 23. Data distribution for 200 kbps had skewness value of 0.516. This was skewed to the right. Also, most values were concentrated on the left of the mean, with extreme values to the right. The value of its Kurtosis was -1.201. The distribution was flatter than a normal distribution with a wider peak. The probability for extreme values was less than for a normal distribution, and the values were wider spread around the mean. The histogram for this distribution is shown in Figure 24.

Table 2

Descriptive Statistics of Delay Time for Different Baud Rates.

	N	Mean		Std. Deviation	Variance	Skewness		Kurtosis	
	Statistic	Statistic	Std. Error	Statistic	Statistic	Statistic	Std. Error	Statistic	Std. Error
BaudRate_50	208	2.305	0.001	0.026	0.001	0.000	0.169	-0.143	.336
BaudRate_100	208	1.162	0.003	0.057	0.003	3.575	0.169	26.224	.336
BaudRate_200	208	.762	0.013	0.200	0.040	0.516	0.169	-1.201	.336
Valid N (listwise)	208								

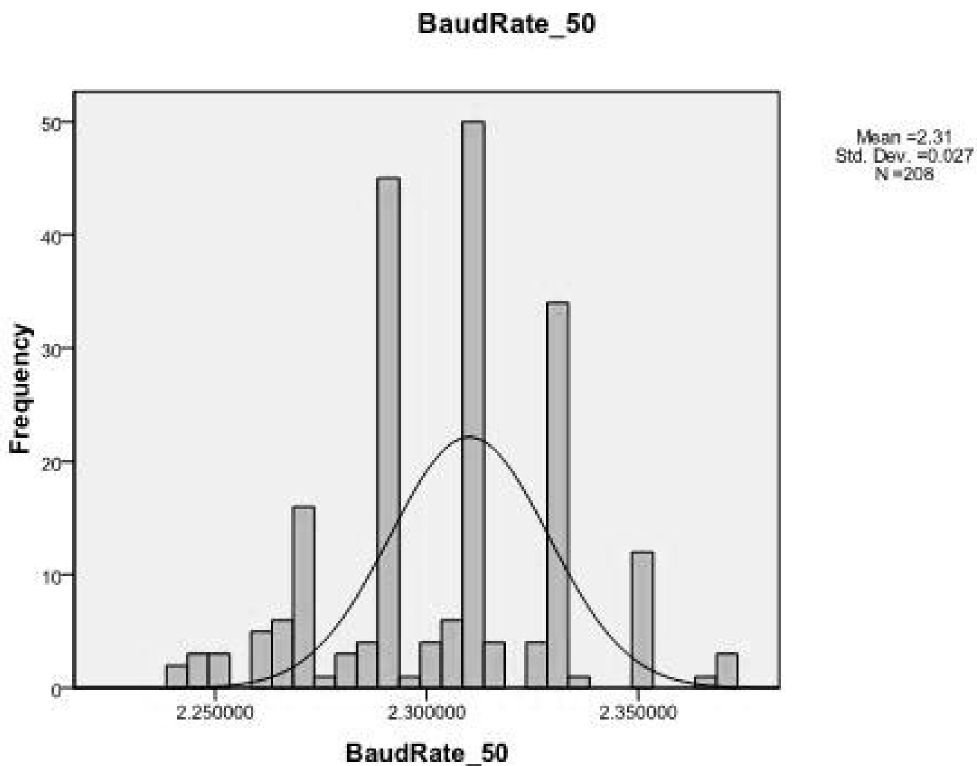


Figure 22. Histogram for 50 kbps Baud Rate Data.

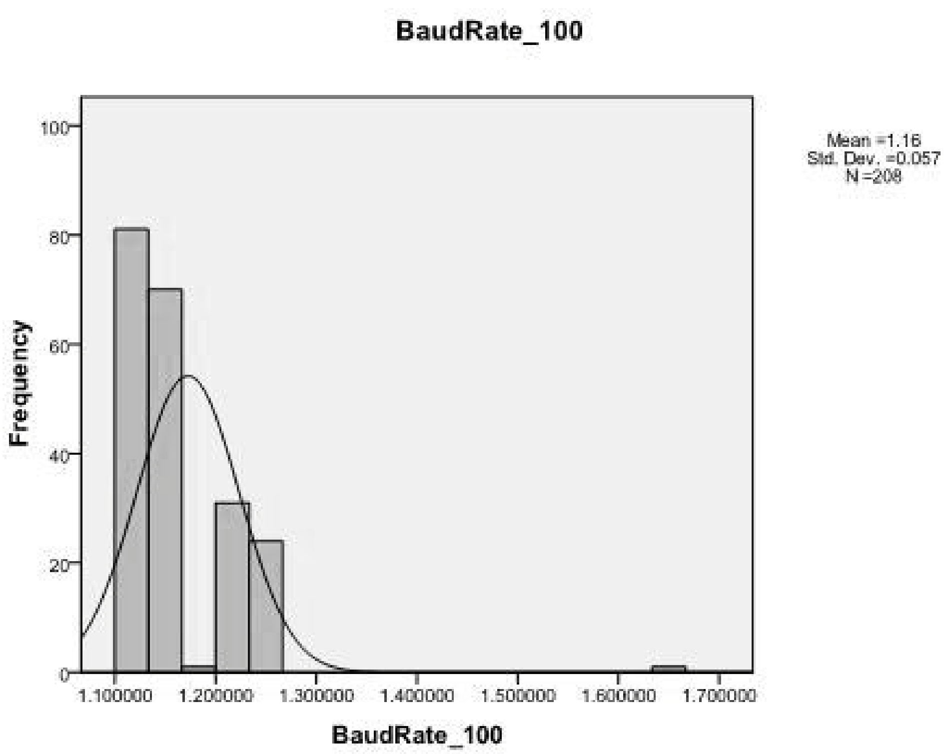


Figure 23. Histogram for 100 kbps Baud Rate Data.

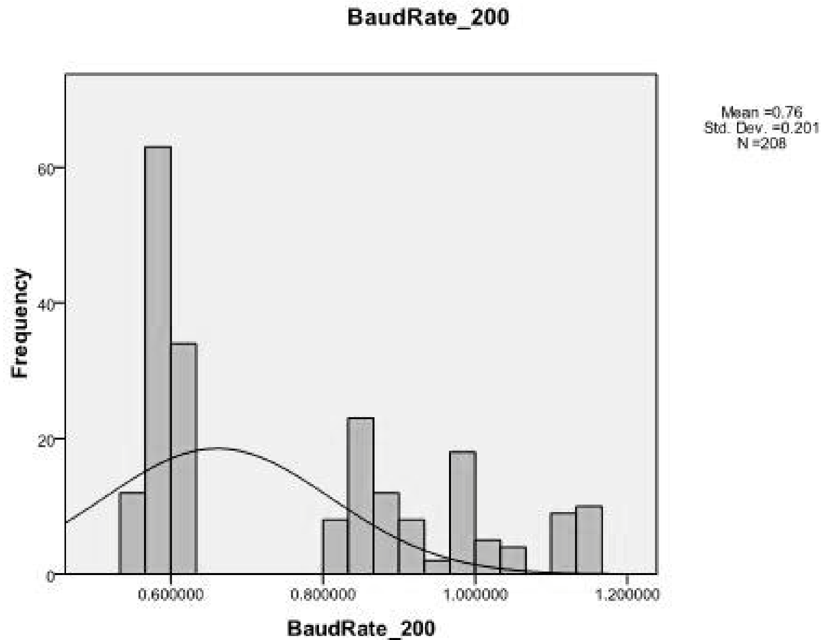


Figure 24. Histogram for 200 kbps Baud Rate Data.

Normality Test of Delay Time for Different Baud Rates

Table 3 presents the results of the normality test. Shapiro-Wilk Test is more appropriate for small sample sizes but can also handle sample sizes as large as 2000. For this reason, the Shapiro-Wilk test was used as the numerical means of assessing normality.

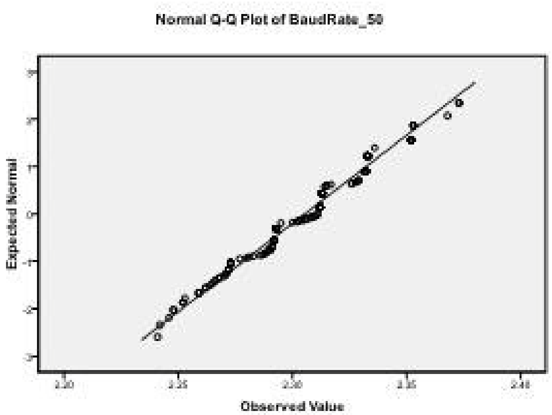
Table 3

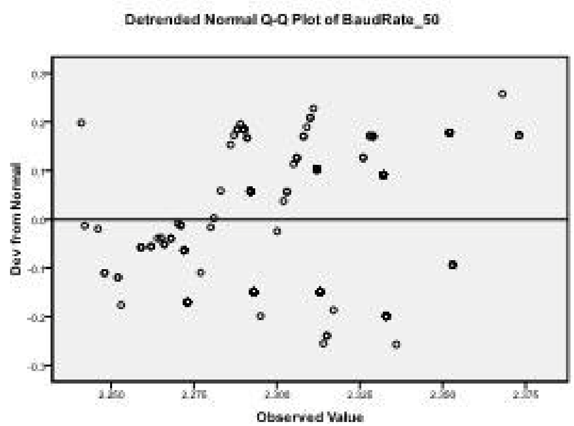
Tests of Normality – Baud Rates.

	BaudRate_ Groupings	Kolmogorov-Smirnov[a]			Shapiro-Wilk		
		Statistic	df	Sig.	Statistic	df	Sig.
BaudRate_Delays	50 Kbits/s	.104	208	.000	.975	208	.001
	100 Kbits/s	.250	208	.000	.642	208	.000
	200 Kbits/s	.287	208	.000	.831	208	.000

a. Lilliefors Significance Correction

If the significant value of the Shapiro-Wilk Test is greater the 0.05 then the data is normal. If it is below α = 0.05, then the data significantly deviate from a normal distribution. Given that *p* = .001 for the 50Kbits/s, *p* = .000 for the 100Kbits/s, and *p* = .000 for the 200Kbits/s and using α = 0.05, it was concluded that each of the levels of the independent variable (BaudRate_Groupings) were not normally distributed. Therefore the assumption of normality was not met. Also, in Figure 25a, the expected normal distribution is the straight line and the line of little boxes is the observed values from the data. The plot in Figure 25a shows that the data distribution of delay time for 50 kbps was not normally distributed.



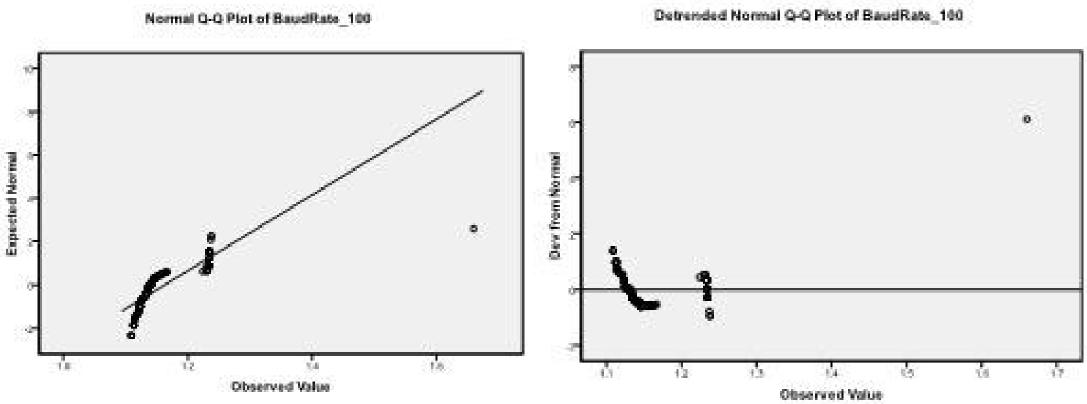


a. Normal Q-Q Plot b. Detrended Normal Q-Q Plot

Figure 25. Q-Q Plot of Delay Time for 50 kbps (a and b).

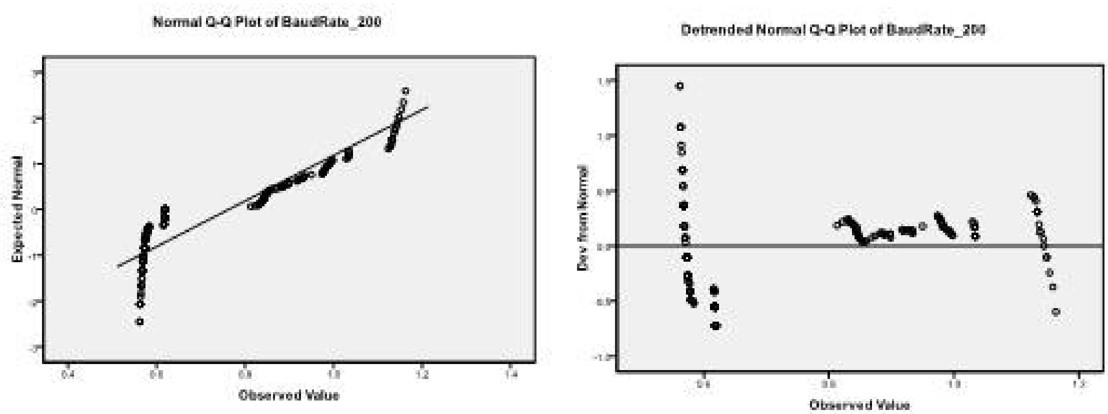
The Detrended Normal Q-Q plot in Figure 25b shows that, the distribution somewhat deviates from normal distribution. The Normal Q-Q plot of Delay Time for 100 kbps in Figure 26a shows that, the distribution deviates from normality. The Detrended Normal Q-Q plot in Figure 26b shows the distribution deviates somewhat from normality at the lower end and slightly at the upper end. Also, the Normal Q-Q plot of Delay Time for 200 kbps in Figure 27a shows the distribution deviates slightly from normality at the lower as well as the upper end.

However, the middle part of the distribution is pretty much normal. It's Detrended Normal Q-Q plot shown in Figure 27b indicates some deviation from normal distribution.



a. Normal Q-Q Plot b. Detrended Normal Q-Q Plot

Figure 26. Q-Q Plot of Delay Time for 100 kbps (a and b).



a. Normal Q-Q Plot b. Detrended Normal Q-Q Plot

Figure 27. Q-Q Plot of Delay Time for 200 kbps (a and b).

ANOVA Analysis of the Effect of Baud Rates on Network Delays

One of the assumptions of the one-way ANOVA is that the variances of the groups being compared should be similar. The Levene's *F* Test for Equality of Variances, which is the most commonly used statistic, was used to test the assumption that the variances of the three baud rates groups are equal; thus, not significantly different. Levene's test uses the level of significance set a priori for the ANOVA ($\alpha = .05$) to test the assumption of homogeneity of variance. Table 4 shows the result of Levene's Test of Homogeneity of Variance. If the significance value is greater than alpha of 0.05 ($p > 0.05$), then there is homogeneity of variances. It was observed from Table 4 that the F value for Levene's test was 634.842 with a Significant (p) value of .000 ($p < .05$). Because the Significant value was less than alpha of .05 ($p < .05$), the null hypothesis (no difference) was rejected for the assumption of homogeneity of variance and concluded that there was significant difference between the three group's variances for 50 kbps, 100 kbps and 200 kbps. The groups do not have similar variances and that the assumption of homogeneity of variance was not met.

Table 4

Test of Homogeneity of Variances – Baud Rates.

BaudRate_Delays

Levene Statistic	df1	df2	Sig.
634.842	2	621	.000

Using the Welch statistic results in Figure 5 "Robust Tests of Equality of Means," it was observed that $F(2, 327.975) = 38500.703$, $p < 0.05$.

Table 5.

Robust Tests of Equality of Means- Baud Rates.

BaudRate_Delays

	Statistic[a]	df1	df2	Sig.
Welch	38500.703	2	327.975	.000

a. Asymptotically F distributed.

Since the alpha level was set at α = 0.05, it was observed that the adjusted *F* ratio was significant. Since the *p* value (*p*= 0.000) was smaller than α = .05, the null hypothesis in hypotheses question 1“There are no differences in mean values of network delays for signal transmission baud rates 50 kbps, 100 kbps and 200 kbps” was rejected. The results obtained indicated that there are significant differences between the groups as a whole.

The Multiple Comparisons given in Table 6 shows which groups differed from each other. The Tukey post-hoc test is generally the preferred test for conducting post-hoc tests on a one-way ANOVA. Table 6 shows that there was a significant difference in the delay time between 50 kbps and 100 kbps baud rates (*P* = 0.000), between 50 kbps and 200 kbps baud rates (*P* = 0.000), as well as between the 100 kbps and 200 kbps baud rates (*p* = 0.000). Figure 28 shows the graph of the Means for the different baud rates. The Mean for the 100 kbps baud rate falls between that of 50 kbps and 200 kbps. The data distributions for the 50 kbps and that of 200 kbps have the highest and lowest Means respectively.

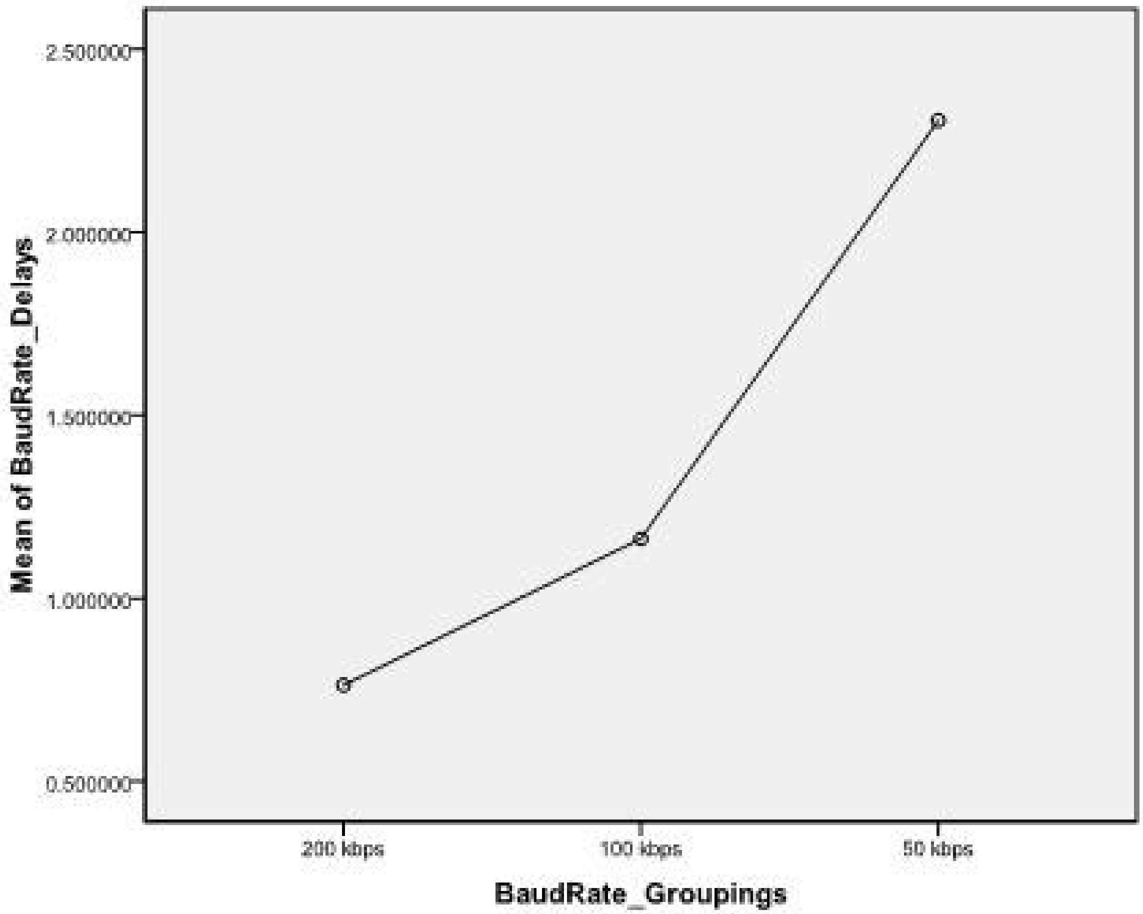


Figure 28. Graph of Means of Different Baud Rate.

Table 6.

Multiple Comparisons – Baud rates.

Dependent Variable:BaudRate_Delays

	(I) BaudRate_ Groupings	(J) BaudRate_ Groupings	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval	
						Lower Bound	Upper Bound
Tukey HSD	50 Kbits/s	100 Kbits/s	1.142*	0.011	.000	1.114	1.170
		200 Kbits/s	1.542*	0.011	.000	1.514	1.570
	100 Kbits/s	50 Kbits/s	-1.142	0.011	.000	-1.170	-1.114
		200 Kbits/s	0.400*	0.011	.000	.372	0.428
	200 Kbits/s	50 Kbits/s	-1.542	0.011	.000	-1.570	-1.514
		100 Kbits/s	-0.400*	0.011	.000	-0.428	-0.372
Bonferroni	50 Kbits/s	100 Kbits/s	1.142*	0.011	.000	1.113	1.171
		200 Kbits/s	1.542*	0.011	.000	1.514	1.571
	100 Kbits/s	50 Kbits/s	-1.142	0.011	.000	-1.171	-1.113
		200 Kbits/s	0.400*	0.011	.000	0.371	0.428
	200 Kbits/s	50 Kbits/s	-1.542	0.011	.000	-1.571	-1.514
		100 Kbits/s	-0.400*	0.011	.000	-0.428	-0.371
Games-Howell	50 Kbits/s	100 Kbits/s	1.142*	0.004	.000	1.132	1.152
		200 Kbits/s	1.542*	0.014	.000	1.509	1.575
	100 Kbits/s	50 Kbits/s	-1.142	0.004	.000	-1.152	-1.132
		200 Kbits/s	0.400*	0.014	.000	0.366	0.434
	200 Kbits/s	50 Kbits/s	-1.542	0.014	.000	-1.575	-1.509
		100 Kbits/s	-0.400*	0.014	.000	-0.434	-0.366

*. The mean difference is significant at the 0.05 level.

Effect of Message Lengths on Network Delays

Descriptive Analysis of Delay Time for Different Message Lengths

For this study also 208 samples of delays for three different signals that have different message lengths were collected. The delays were in milliseconds (ms). The signals were engine data with identifier 64 (Hex) which had a message length of 8 bytes, ABS data with identifier c9 (Hex) which had a message length of 6 bytes, and gearbox info with identifier 3fc (Hex) which had a message length of 1 byte. The baud rate selected for this study was 200 kbps. The data was analyzed using statistical package SPSS.

Table 7 provides descriptive statistics such as the mean, standard deviation, variance, Skewness, and Kurtosis at 95% confidence intervals for the dependent variable (DelayTime) for each separate group (DataLength_64, DataLength_C9 and DataLength_3fc).

Table 7

Descriptive Statistics of Delay Time for Different Message Data Length.

		DataLength_64	DataLength_C9	DataLength_3fc
N	Valid	208	208	208
	Missing	1065	1065	1065
Mean		.762	0.705	0.520
Std. Error of Mean		.0139	0.015	0.015
Std. Deviation		0.200	0.220	0.223
Variance		0.040	0.049	0.050
Skewness		0.516	0.409	0.159
Std. Error of Skewness		0.169	0.169	0.169
Kurtosis		-1.201	-1.393	-1.487
Std. Error of Kurtosis		0.336	0.336	0.336

The mean value of time delays for data length of 8 bytes (signal 64-Engine data) was 0.7626. It had a standard deviation of 0.20078. Signal with data length of 6 bytes (signal C9-

ABS Data) had mean time delay of 0.7055 with a standard deviation of 0.22023. The mean time delay for the signal with data length of 1 byte (signal 3fc- Gearbox Info) was observed to be 0.5207. It had a standard deviation of 0.22309. The skewness value for DataLength_64, DataLength_C9 and DataLength-3FC time delays were 0.516, 0.409 and 0.159 in that order. The time delay distributions for the various signals were skewed to the right. Most of the data values were concentrated on the left of the mean, with extreme values to the right. This can be observed in Figure 29, 30 and 31 respectively. All the signals have negative Kurtosis value. This means that their individual distribution is flatter than a normal distribution with a wider peak. The probability for extreme values was less than for a normal distribution, and their values are wider spread around their various means.

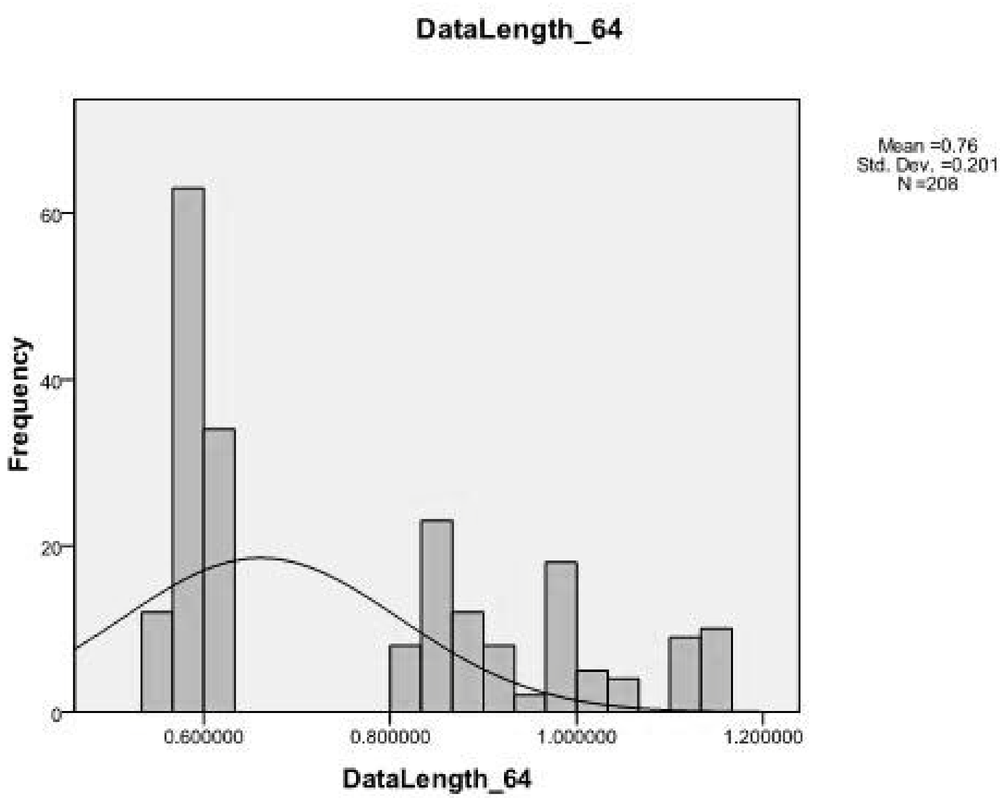


Figure 29. Histogram for Data Length_64 Delay Time.

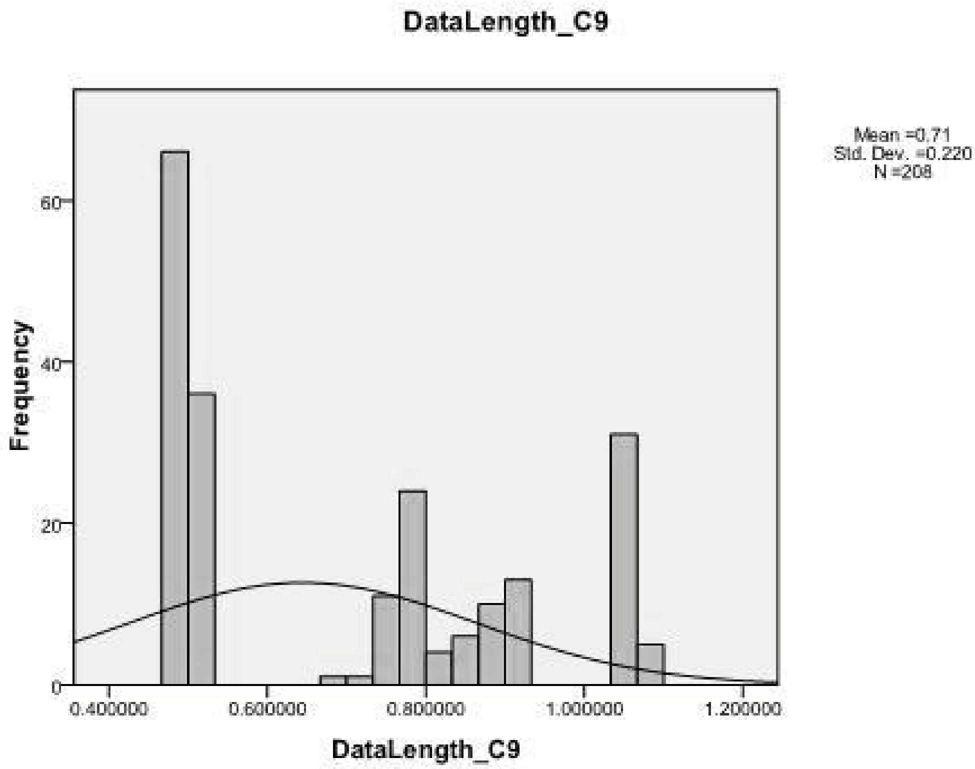


Figure 30. Histogram for Data Length_C9 Delay Time.

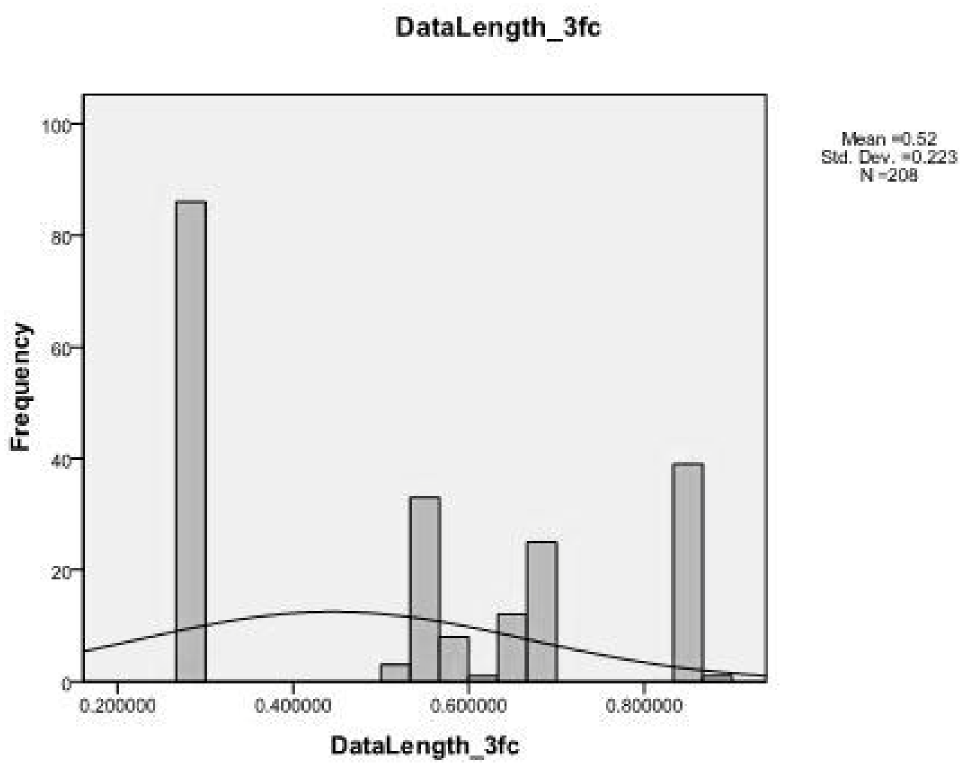


Figure 31. Histogram for Data Length_3fc Delay Time.

Test of Normality – Messages Lengths

Table 8 shows the results of the Shapiro-Wilk Test. This was used as the numerical means of assessing normality. As shown in this table, computations using $\alpha = 0.05$, the "High Data Length of 8 (Signal 64-Engine Data)", "Data Length of 6 (Signal C9-ABS Data)" and "Low Data Length of 1 (Signal 3fc- GearBox Info)" signal group dependent variable, "TimeDelays", was not normally distributed. This was because the significant values of the various data length (p=0.000) was less than $\alpha = 0.05$.

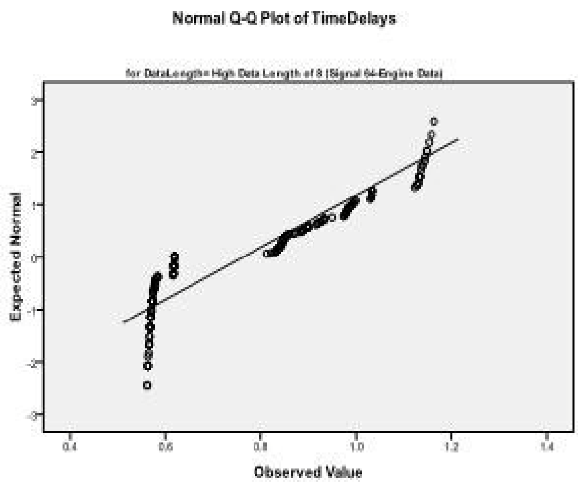
Table 8.

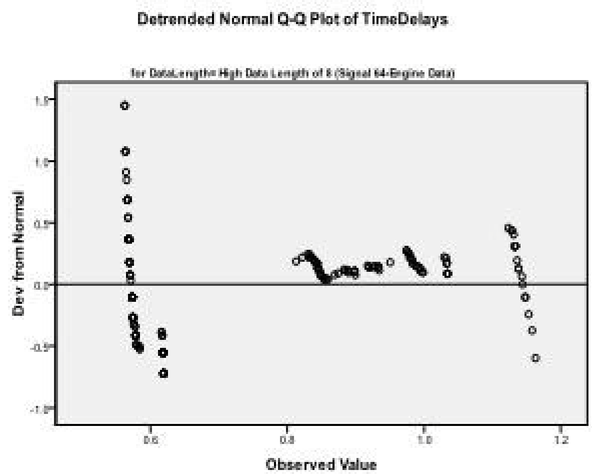
Tests of Normality – Data Lengths.

		Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	DataLength	Statistic	df	Sig.	Statistic	df	Sig.
TimeDelays	High Data Length of 8 (Signal 64-Engine Data)	.287	208	.000	.831	208	.000
	Data Length of 6 (Signal C9-ABS Data)	.297	208	.000	.811	208	.000
	Low Data Length of 1(Signal 3FC-GearBox Info)	.268	208	.000	.828	208	.000

a. Lilliefors Significance Correction

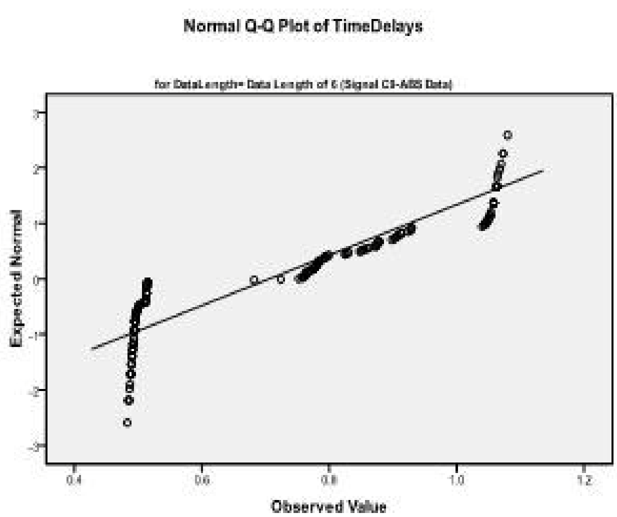
The output of the Normal Q-Q Plots in Figure 32a, 33a, and 34a shows that the data distribution for High Data Length of 8 bytes (Signal 64-Engine Data)", "Data Length of 6 bytes (Signal C9-ABS Data)" and "Low Data Length of 1 byte (Signal 3FC- GearBox Info)" are normally distributed in the middle but deviate from normality at the lower and upper end of the graphs respectively. The same trend was observed for the Detrended Normal Q-Q plot in Figures 32b, 33b and 34b.

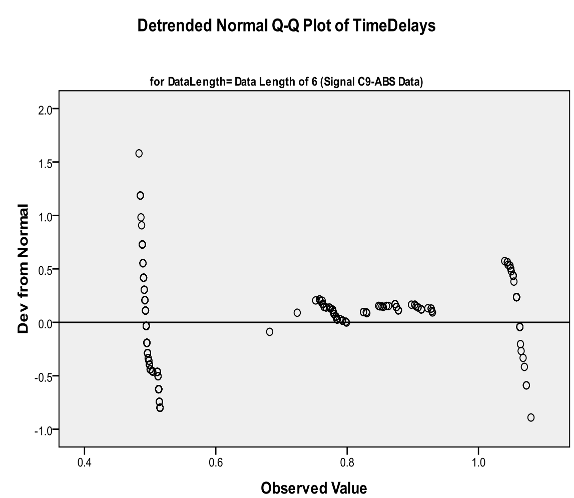




a. Normall Q-Q Plot b. Detrended Normal Q-Q Plot

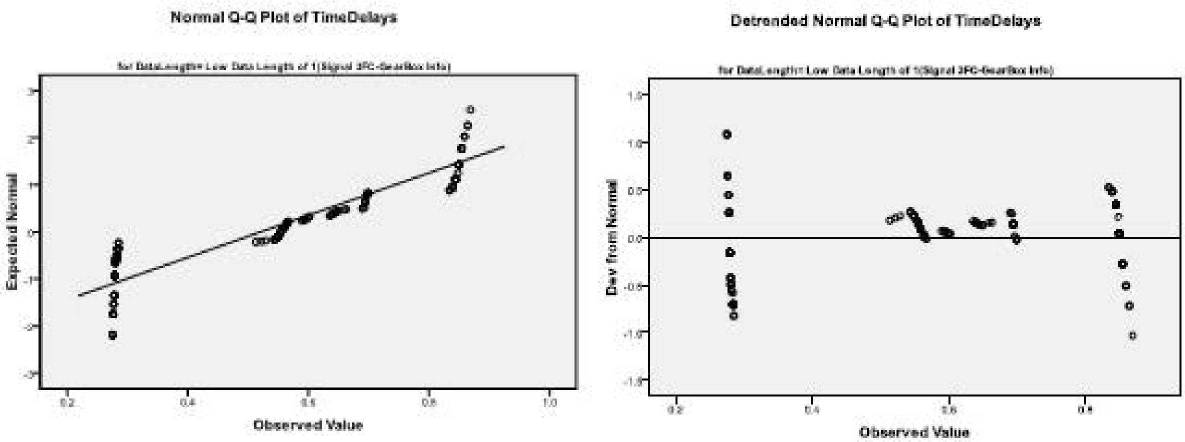
Figure 32. Q-Q Plot of Delay Time Data for Data Length 64 (a and b).





a. Normall Q-Q Plot b. Detrended Normal Q-Q Plot

Figure 33. Q-Q Plot of Delay Time for Data Length C9 (a and b).



a. Normall Q-Q Plot b. Detrended Normal Q-Q Plot

Figure 34. Q-Q Plot of Delay Time for Data Length 3FC (a and b).

Table 9 shows the result for the Test of Homogeneity of Variances. An alpha level of .05 was used for all analyses. The test for homogeneity of variance was not significant [Levene F (2, 621) = 2.28, $p > 0.05$] indicating that this assumption underlying the application of ANOVA was met. This means that the delay times for the various signals have similar variances and there was no need for further test using the Robust Tests of Equality of Means.

Table 9.

Test of Homogeneity of Variances – Data Lengths.

Delay Time

Levene Statistic	df1	df2	Sig.
2.280	2	621	.103

The one-way ANOVA results in Table 10 revealed a statistically significant main effect [F (2, 621) = 71.94, $p < .05$] indicating that not all three groups of the delay time for signals with different data length resulted in the same means. The null hypothesis H_{02}: $\mu_8 = \mu_6 = \mu_1$, “there are no statistically significant difference in the mean values of network delays for signals with message lengths 8 bytes, 6 bytes, and 1 byte,” was therefore rejected.

Table 10

ANOVA Table – Data Lengths.

TimeDelays

	Sum of Squares	Df	Mean Square	F	Sig.
Between Groups	6.647	2	3.323	71.938	.000
Within Groups	28.688	621	.046		
Total	35.334	623			

The results obtained indicated that there were statistically significant differences between the groups as a whole. However, Post hoc comparisons Test using Tukey procedures shown in Table 11 was used to determine which pairs of the three group means differed. The results obtained shows that significant differences exist in the delay time between the signals with a high data length of 8 bytes (signal 64) and the signal with data length of 6 bytes (signal c9) ($p = 0.019$) as well as between the signal with a high data length of 8 bytes (signal 64) and the signal with low data length of 1 byte (signal 3fc-GearBox Info) ($p = 0.000$). Again, there was statistically significant difference between mean groups as determined by one-way ANOVA results (Table 10) ($F (2, 621) = 71.938$, $p = 0.000$). A Tukey post-hoc test revealed that the time delay for signal (data) transmission were statistically significantly lower for low data length of 1 “GearBox Info” (0.5207± 0.223 min, $P = 0.000$), and data length of 6 “ABS Data” (0.7055± 0.220 min, $P =.019$) signals compared to the high data length of 8 “Engine Data” (0.763± 0.2 min). The graph of the means plot of delay time for various data length is shown in Figure 35.

Table 11

Multiple Comparisons – Data Lengths.

Dependent Variable:TimeDelays							
	(I) DataLength	(J) DataLength	Mean Difference (I-J)	Std. Error	Sig.	95% Confidence Interval Lower Bound	Upper Bound
Tukey HSD	High Data Length of 8 (Signal 64-Engine Data)	Data Length of 6 (Signal C9-ABS Data)	0.05707*	0.021	.019	0.007	0.106
		Low Data Length of 1(Signal 3FC-GearBox Info)	0.2418*	0.021	.000	0.192	0.291
	Data Length of 6 (Signal C9-ABS Data)	High Data Length of 8 (Signal 64-Engine Data)	-0.0570*	0.021	.019	-0.106	-0.007
		Low Data Length of 1(Signal 3FC-GearBox Info)	0.1847*	0.021	.000	0.135	0.234
	Low Data Length of 1(Signal 3FC-GearBox Info)	High Data Length of 8 (Signal 64-Engine Data)	-00.2418*	0.021	.000	-0.291	-0.192
		Data Length of 6 (Signal C9-ABS Data)	-0.1847*	0.021	.000	-0.234	-0.135
Games-Howell	High Data Length of 8 (Signal 64-Engine Data)	Data Length of 6 (Signal C9-ABS Data)	0.0570*	0.020	.017	0.008	0.105
		Low Data Length of 1(Signal 3FC-GearBox Info)	0.2418*	0.020	.000	0.192	0.290
	Data Length of 6 (Signal C9-ABS Data)	High Data Length of 8 (Signal 64-Engine Data)	-0.0570*	0.020	.017	-0.105	-0.008
		Low Data Length of 1(Signal 3FC-GearBox Info)	0.1847*	0.021	.000	0.133	0.235
	Low Data Length of 1(Signal 3FC-GearBox Info)	High Data Length of 8 (Signal 64-Engine Data)	-0.2418*	0.020	.000	-0.290	-0.192
		Data Length of 6 (Signal C9-ABS Data)	-0.1847*	0.021	.000	-0.235	-0.133

*The mean difference is significant at the 0.05 level.

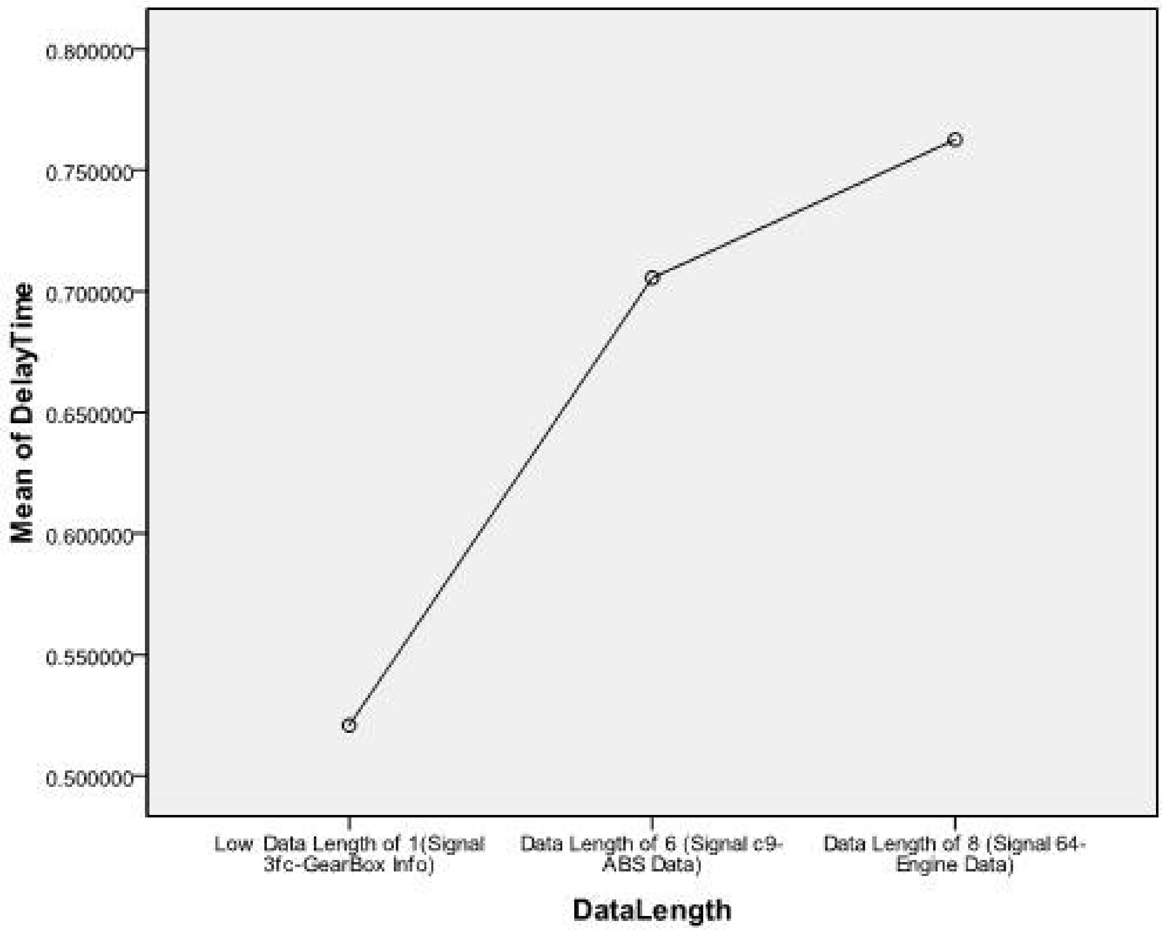


Figure 35. Graph of Means of Different Data Length.

Effect of Busload on Network Delays

Descriptive Statistics of Delay Time for Different Busloads

For this study 120 samples of delays for the signal 3fc (Hex) at two different bus loads were collected. The delays were measured in milliseconds (ms). The bus loads considered were “Low Busload” 17% and “High Busload” 22%. The baud rate selected for this study was 100 kbps. The data was analyzed using statistical package SPSS. As shown in Table 12, the delay time for the “High Busload” have a mean of 2.1369 and a standard deviation of 1.4922. The mean delay time for the “Low Busload” was 1.4458 and the standard deviation was 0.9354. The skewness values for the "High Busload" and "Low Busload" delay time were 0.401, and 0.507 in that order. The time delay distributions for the data were skewed to the right. This can be observed in Figure 36, and 37 respectively. The data for both the "High Busload and Low Busload" have negative Kurtosis value. This means that their individual distribution is flatter than a normal distribution with a wider peak. The probability for extreme values was less than for a normal distribution, and their values are widely spread around their means.

Table 12

Descriptive Statistics - Busloads.

		High_Busloads	Low_Busloads
N	Valid	120	120
	Missing	1153	1153
Mean		2.136	1.445
Std. Error of Mean		0.136	0.085
Std. Deviation		1.492	0.935
Variance		2.227	0.875
Skewness		0.401	0.507
Std. Error of Skewness		0.221	0.221
Kurtosis		-1.157	-0.976
Std. Error of Kurtosis		0.438	0.438

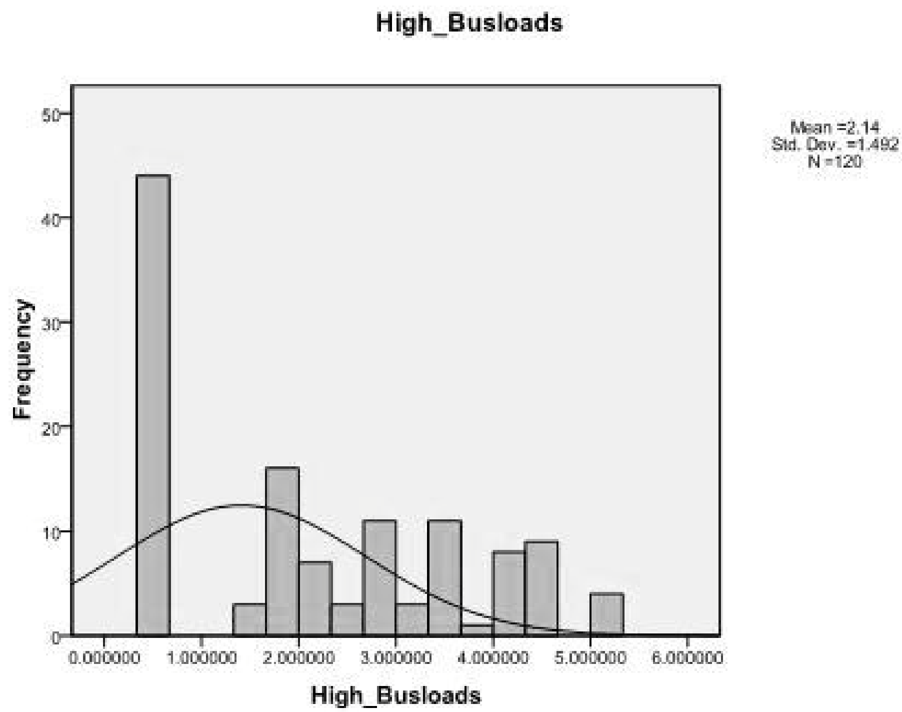


Figure 36. Histogram for High Busload Data.

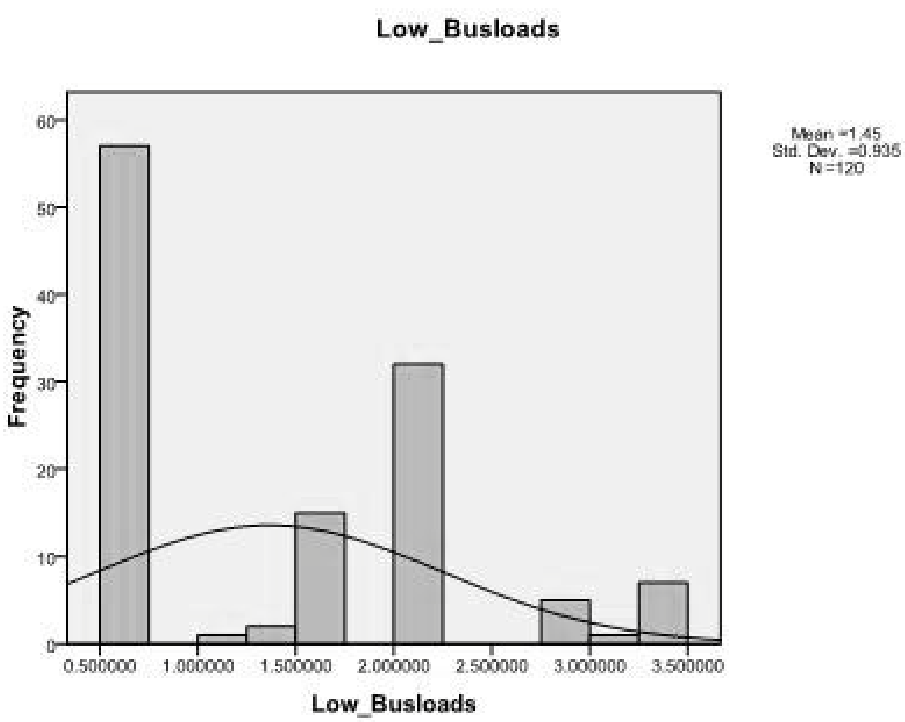


Figure 37. Histogram for Low Busload Data.

Test of Normality

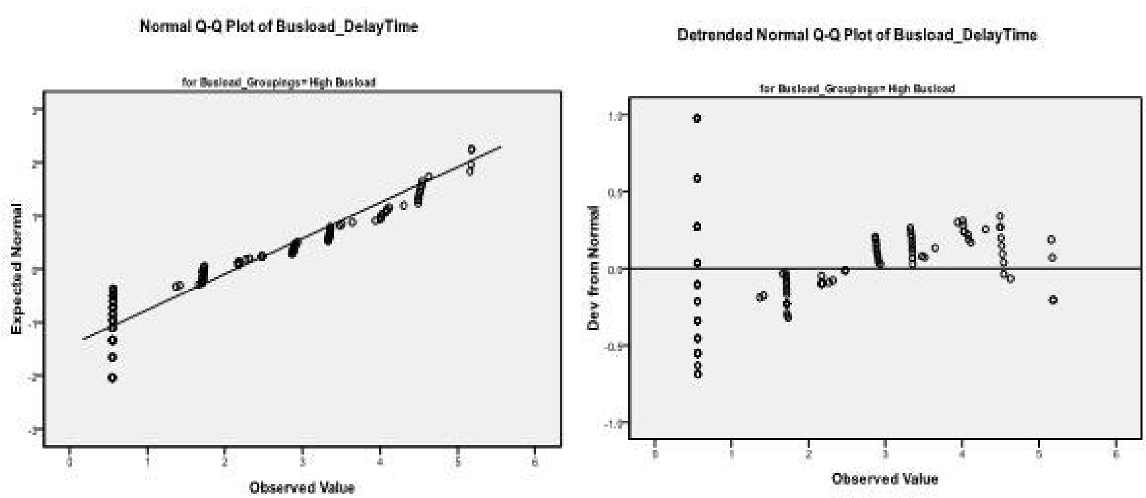
The result of the normality test is shown in Table 13. The Table revealed that the “High Busload” and “Low Busload” groupings dependent variable, “TimeDelays”, was not normally distributed. This was because the significant values for “High Busload and Low Busload” (p = 0.000) was less than $\alpha = 0.05$. The output of the normal Q-Q Plots in Figures 38a and 39a shows that the data distribution for the “High Busload” and “Low Busload” somewhat deviate from normality. The same trend was observed for the Detrended Normal Q-Q plot in Figures 38b and 39b in that order.

Table 13

Tests of Normality - Busloads.

		Kolmogorov-Smirnov[a]			Shapiro-Wilk		
	Busload_Groupings	Statistic	df	Sig.	Statistic	df	Sig.
Busload_ DelayTime	High Busload	.222	120	.000	.873	120	.000
	Low Busload	.301	120	.000	.806	120	.000

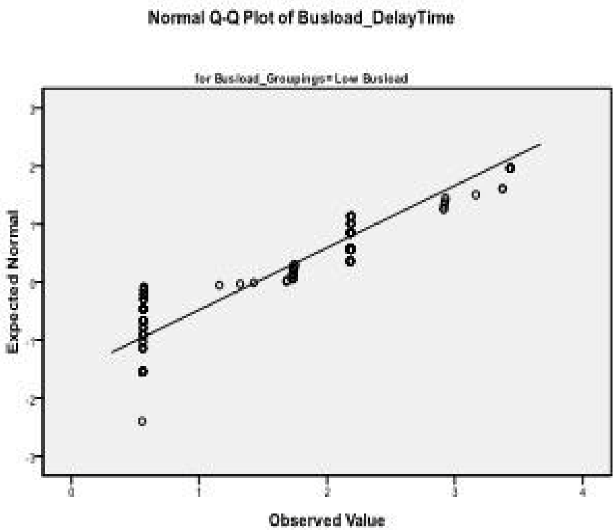
a. Lilliefors Significance Correction

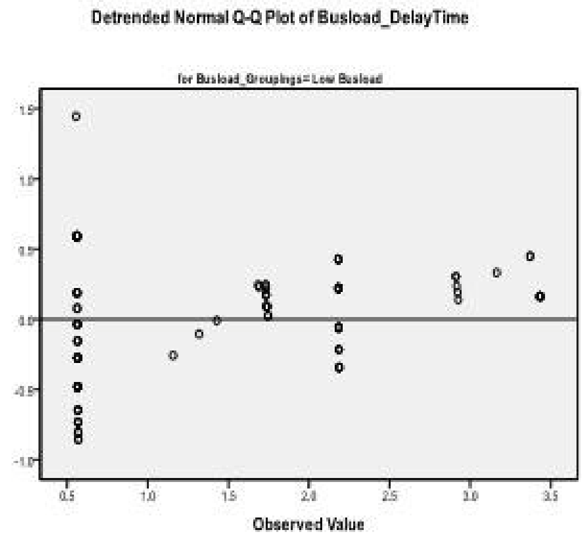


a. Normall Q-Q Plot

b. Detrended Normal Q-Q Plot

Figure 38. Q-Q Plot of Delay Time for High Busload (22%) *(a and b).*





a. Normall Q-Q Plot b. Detrended Normal Q-Q Plot

Figure 39. Q-Q Plot of Delay Time for Low Busload (17%) *(a and b).*

Table 14 shows the result for the Test of Homogeneity of Variances. The test for homogeneity of variance was not significant [Levene F (1, 238) = 38.44, $p < 0.05$] indicating that the assumption underlying the application of ANOVA was not met. This means that the delay time for "High Busload and Low Busload" data does not have similar variances and therefore, was the need for further test using the Robust Tests of Equality of Means.

Table 14

Test of Homogeneity of Variances - Busloads.

Busload_DelayTime

Levene Statistic	df1	df2	Sig.
38.440	1	238	.000

Table 15 shows the result of the Robust Tests of Equality of Means. Using the Welch statistic, it was revealed that F (1, 200.021) = 18.48, $p < 0.05$. The significance value was shown to be 0.000. This was less than 0.05. This means that, there are statistically significant differences between the delay time for the High and Low Busloads data. The null hypothesis, H_{03}: $\mu_{22\%} = \mu_{17\%}$, there are no statistically significant difference in the mean values of network delays for busload of 22% and 17%, was therefore rejected.

Table 15

Robust Tests of Equality of Means - Busloads.

Busload_DelayTime

	Statistic[a]	df1	df2	Sig.
Welch	18.483	1	200.021	.000

a. Asymptotically F distributed.

The means plot of delay times for the High and Low Busloads is shown in Figure 40. It was observed that the mean values of the delays increased with the increase in busload.

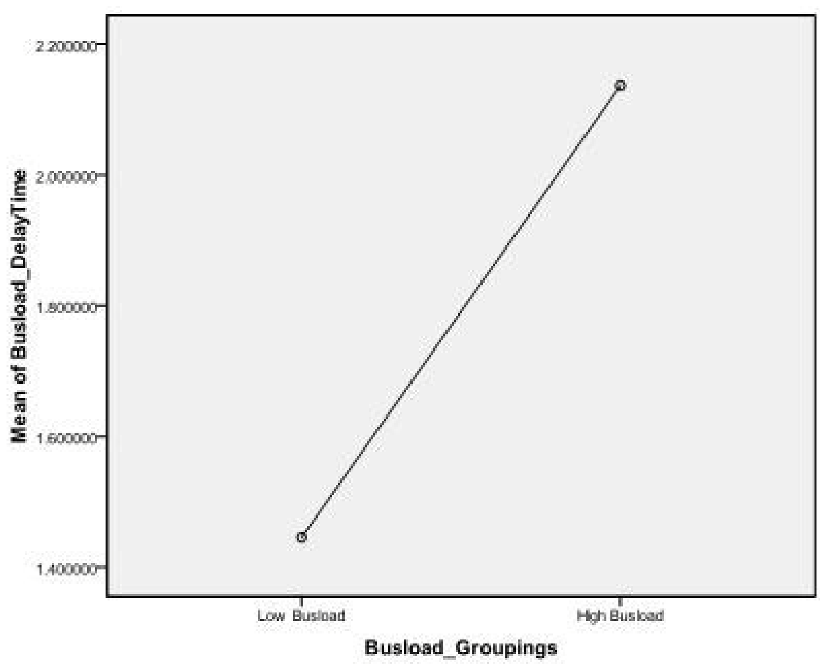


Figure 40. Graph of Means for High and Low Busloads.

Summary of Chapter

The methodology to study the effect of various CAN parameters on network delays was presented in this chapter. The research questions and the various hypotheses related to the analysis of network delays part of this dissertation study were restated. The analysis was done using the data collected from the simulation of an automobile system via CANoe software. Details of the simulation were presented. Descriptive statistics and ANOVA were used for the analysis. It was observed from the analyzed data that there are statistically significant differences in the mean values of network delays for different CAN parameters, baud rates, message lengths and busloads, thus rejecting the null hypotheses.

CHAPTER 4

METHODOLOGY AND FINDINGS - EFFECTS OF NETWORK DELAYS ON CLOSED-LOOP SYSTEMS

This chapter provides a description of the methodology for achieving the objectives of this dissertation that relate to the effects of network induced delays on closed loop control systems. The research questions that were related to this part of the dissertation study are restated. The analyses of the effects were carried out using a continuous-time system with PID controller. A DC motor model in transfer function form was used for the continuous-time system. The simulations were performed using MATLAB/Simulink and details of the simulation are presented.

Research Questions

In order to achieve the objectives of the dissertation, this chapter was designed to help answer the following research questions:

3. Does fieldbus network induced delay have an effect on control systems stability and performance as described by system step response?
4. Do the PID type controller gains have effect on control systems with fieldbus network induced delays?

Mathematical analysis tools (Dorf, 2010) are used to answer these research questions, in contrast to statistical analysis methods used in the previous chapter to answer research questions

1 and 2. MATLAB/Simulink program was used for this analysis to contrast the SPSS program in the previous chapter.

Effect of Delays in Control Systems with PID Controllers

The closed loop control system was shown in Figure 5, and its operation was described in Chapter 2. As discussed in Chapter 1, when fieldbus networks are used in the implementation of this closed loop system, as a result of the limited network bandwidth, two sources of delays may occur. These are

1. Sensor-to-controller delay (τ_{sc})
2. Controller to actuator delay (τ_{ca})

Any controller computational delay can be absorbed into either τ_{sc} or τ_{ca} without loss of generality. This is shown in Figure 2 in Chapter 1. For fixed control law, thus, time-invariant controllers, the sensor to controller delay and controller to actuator delay can be put together as $\tau = \tau_{sc} + \tau_{ca}$ as for analysis purposes. The delays can affect the transient response behaviors shown in Figure 6 in a control system. The general system performance as described by the criteria for step response may be degraded. This effect may include the increase of both overshoot and settling time of the system step response. The delays can also affect the stability of a system and cause the system to become unstable.

In this section the effects of network induced delays was studied using a DC Motor model. Two different control schemes were used for this research analysis to control the speed of the DC motor. The first one was a Proportional and Integral (PI) controller studied by Tipsuwan et al., (2003). In the second controller, the PI concepts were extended to include a Derivative mode, resulting in a Proportional, Integral and Derivative (PID) controller. These were done to study the effect of the network-induced time delay on PID controllers.

In order to compare the performance of these controllers under different network delays, the evaluation criteria used was: rise time, settling time, and percentage overshoot of the step response. As described in Chapter 2, rise time is the time required for the DC motor speed to rise from 10% to 90% of its final steady-state value. The settling time is the time required for the DC motor speed to reach and stay within 5% range about the final value. The maximum overshoot is the maximum peak value of the DC motor speed measured from the unit step input. The DC motor model used for this study was the same as the one used by Tipsuwan (2003).

DC Motor Model

DC motor is commonly used for producing continuous movement. Its speed of rotation can easily be controlled, thus, making them ideal for use in applications where speed control, servo type control, and or positioning is required. Many control applications use DC motors as a drive force. When the DC motor turns on, the controller quickly adjust the input voltage for the output speed to track the required reference speed with an acceptable rise time, settling time and overshoot criteria. But as the output speed approaches the required reference speed, the controller will adjust the input voltage since the error signal between the reference speed and DC motor speed is reduced. When a fieldbus network is used in the network-based closed-loop DC motor control, the input voltage command from the controller to the DC motor and the speed feedback signal from the sensor to the controller may be delayed due to network-induced time delays. If these delays are long enough, the rise time, settling time and overshoot may affect the overall acceptable control of the DC motor, and it may even degrade the system performance and make the system unstable.

The DC motor consists of two parts: the stator which is the stationary part provides fixed magnetic field and a rotor which is the rotating part (armature) to which mechanical load is connected and input voltage is supplied (Dorf et al., 2010). The schematic diagram of the DC motor is shown in Figure 41.

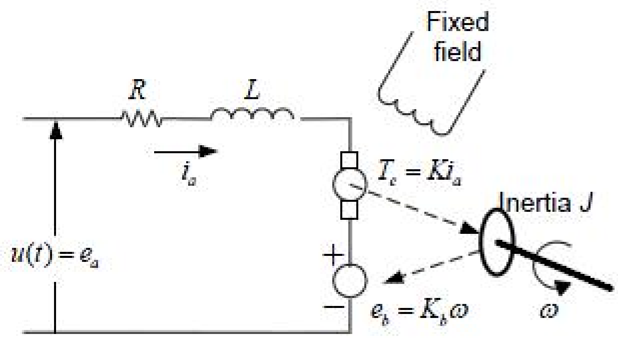


Figure 41. Diagram of DC Motor Circuit (courtesy of Dorf et al., 2010).

The Kirchhoff's voltage law equation for the armature electrical circuit of DC motor gives

$$u(t) = e_a = L\frac{di_a}{dt} + Ri_a + K_b\omega \quad (3)$$

where $u(t) = e_a$ is the armature input voltage, L is the armature inductance, *R* is the armature resistance, i_a is the armature current, K_b is the back emf constant, and ω is the rotor angular speed. Using Newtonian mechanics, the mechanical torque balance equation gives

$$J\frac{d\omega}{dt} + B\omega + T_l = T_e = Ki_a \quad (4)$$

Where *J* is the motor moment of inertia, *B* is the motor damping coefficient; *K* is the torque constant and T_l is the load torque. The parameters of the DC motor used in this research are shown in table 16.

Table 16

Parameters of DC Motor.

J	Inertia	42.6 e-6 Kg-m^2
L	Inductance	170 e-3 H
R	Terminal Resistance	4.67 Ω
B	Damping Coefficient	47.3 e-3 N-m-sec/rad
K	Torque Constant	14.7 e-3 N-m/A
K_b	Back emf Constant	14.7 e-3 V-sec/rad

By taking the Laplace Transformation on both sides of the above two equations, rearranging them and substituting the above numerical values, the resultant DC motor system (plant) dynamics was obtained and described by the transfer function below (Tipsuwan et al., 2003).

$$G_p = \frac{2029.826}{(s+26.29)(s+2.296)} \tag{5}$$

This plant transfer function between speed and armature voltage was used in the MATLAB/ Simulink program to evaluate the performance of the networked control system.

PI and PID Controllers

The transfer function of the PI controller is given by

$$G_c\,(s) = \frac{K_p\left(s+\frac{K_I}{K_p}\right)}{s} \tag{6}$$

Where K_p is the Proportional gain, and K_I is the Integral gain. The transfer function of the PID controller is also given by

$$G_c(s) = K_P + \frac{K_I}{S} + K_D S = \frac{K_D S^2 + K_P S + K_I}{S} \tag{7}$$

where K_D = Derivative gain. It was assumed that the PI and PID speed controllers for the dc motor were first designed without network delays, and then with network delays were included. The effects of these network delays on the controllers were studied.

Results and Analysis of the Effect of Network Delays on DC Motor Speed with PI/PID Controller

As discussed before, network delays affect the magnitude of the transient behaviors in a control system. These delays deteriorate or degrade the performance of the system. This effect can result in the increasing of both overshoot and settling time. In this section these effects are studied for the DC motor control system using MATLAB/Simulink simulations. The motor transfer function equation (5) described was used as the plant model, and the PI/PID controller transfer functions (6) and (7) given before were used with different values for K_p, K_I and K_D in the simulations. The Simulink software determines the closed loop system transfer function for the entire system based on the motor and controller transfer functions. Network delays between sensor to controller and controller to actuator were then included in the simulations. These delays were varied to study their effect on system performance. The Simulink diagram of the setup of the NCS is shown in Figure 42.

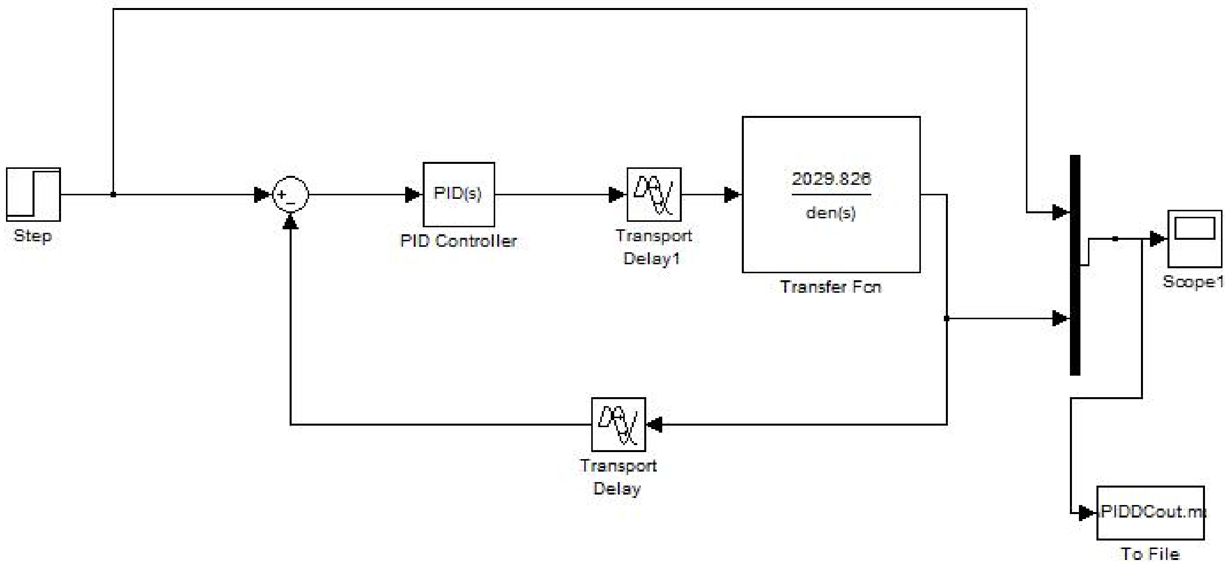


Figure 42. Networked Based DC Motor Control System Simulation Diagram.

PI Controller

First, a PI controller with gains K_P = 0.1701 and K_I =0.378 was used (Tipsuwan et al., 2003) in the study. For this controller, the graph of the step input, various outputs when delays (τ_{sc} and τ_{ca}) have equal values of 0, 0.02s, and 0.04s is shown in Figure 43. It can be observed that the system performance gradually degraded as the delays were increase. The system became unstable at values of delays beyond 0.05s. For instance, the system became unstable at delays of 0.06s, as shown in Figure 44.

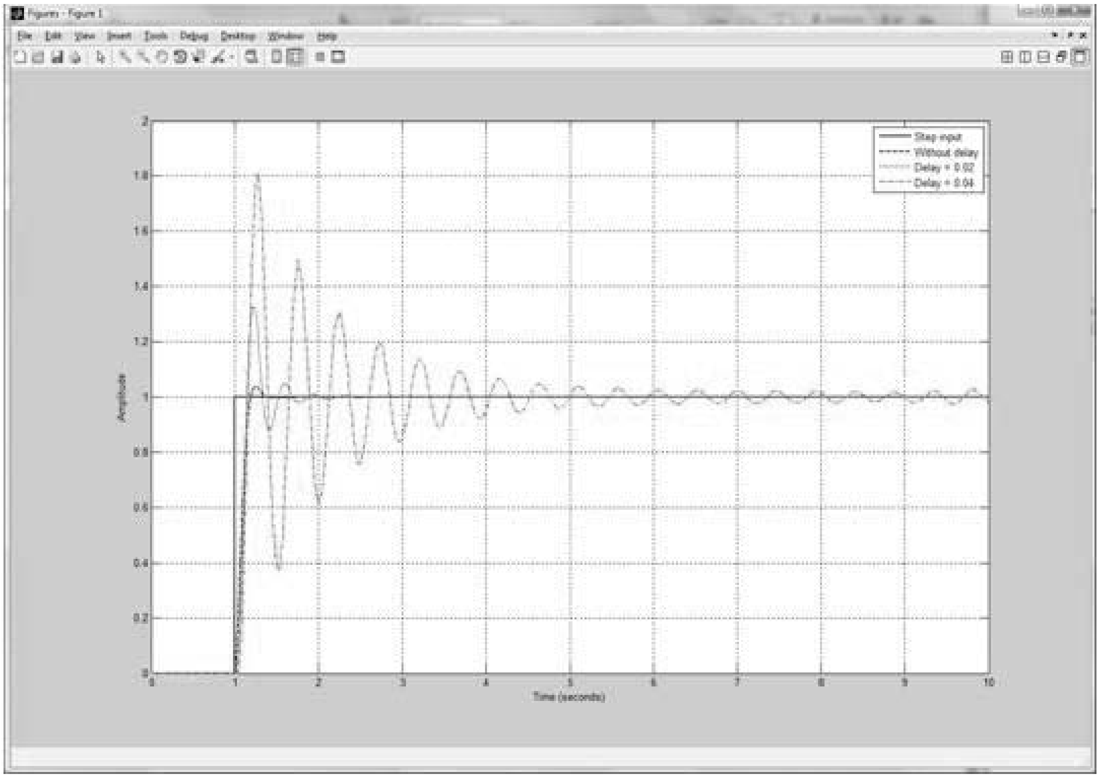


Figure 43. PI Controller Performance Measures at Different Network Delays

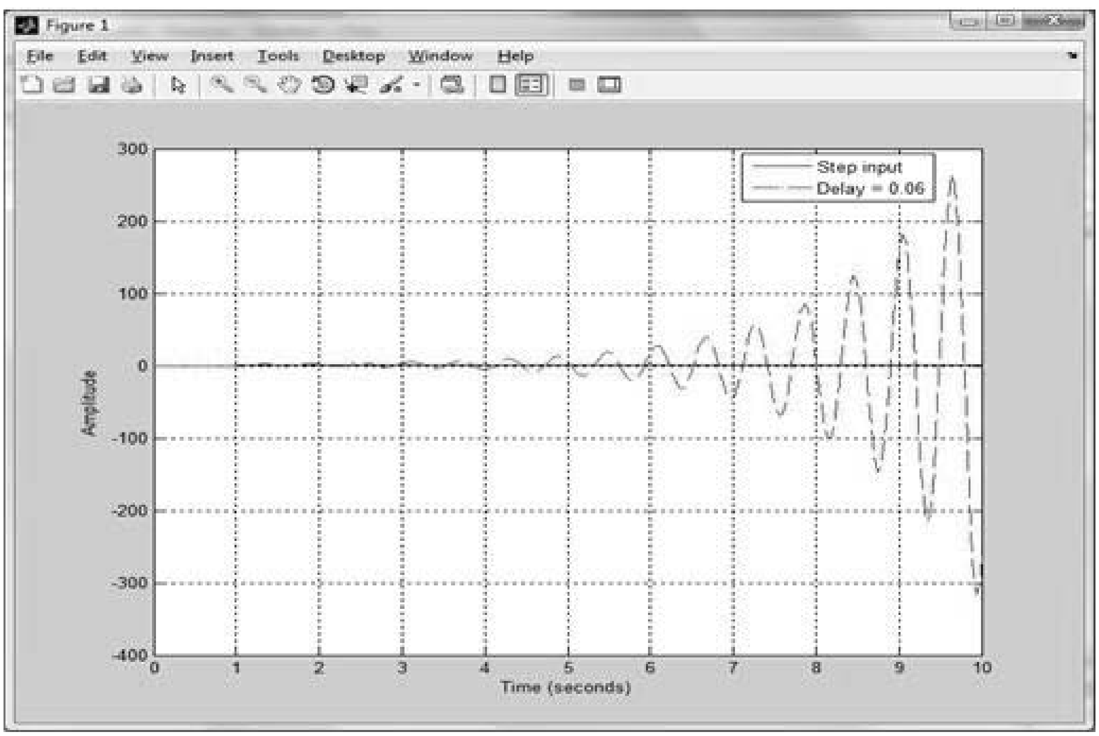


Figure 44. PI Controller Performance when Delays (τ_{sc} and τ_{ca}) = 0.06s.

Table 17 presents a summary of the step response information obtained after the simulations of the DC motor with NCS using these PI controller gains. It was observed that, the percentage overshoot as well as settling time increased with an increase in delay. For example, the percentage overshoot and settling time for 0.01s delay were 14.10%, and 0.270s respectively. The percentage overshoot was increased to 81.0% and the settling time also increased to 2.987s when the delays became 0.04s.

Table 17

PI Controller Parameters and Performance Measures.

	Controller Parameters			**Performance Measures**			
Delay (s)	**K_P**	**K_I**	**K_D**	**% Overshoot**	**Peak Time (s)**	**Rise Time (s)**	**Settling Time (s)**
0	0.1701	0.378	0	3.80%	0.239	0.117	0.162
0.01	0.1701	0.378	0	14.10%	0.212	0.0255	0.270
0.02	0.1701	0.378	0	34.20%	0.212	0.083	0.494
0.03	0.1701	0.378	0	55.50%	0.226	0.079	1.154
0.04	0.1701	0.378	0	81.0%	0.267	0.078	2.987
0.05	0.1701	0.378	0	106.10%	0.297	0.078	> 200

A PID controller was designed using the same PI controller gains of the previous section and a derivative gain term was added. The gains used for K_P, K_I and K_D were 0.1701, 0.378 and 0.0075. For this controller, the graph of the step input, various outputs when network delays (τ_{sc} and τ_{ca}) have equal values of 0.00s, 0.03s, and 0.06s are shown in Figure 45. The PID controller response was better than the PI controller without delays with respect to percentage overshoot, though the settling time slightly increased. It can also be observed that the system performance gradually degraded as the network delays increased in values, and especially became unstable at values of delays beyond 0.06s. For instance, the system became unstable at delays of 0.07s, as shown in Figure 46. However, this PID controller improved the performance of the system as it accommodated more network delay values before it became unstable than that of the PI controller.

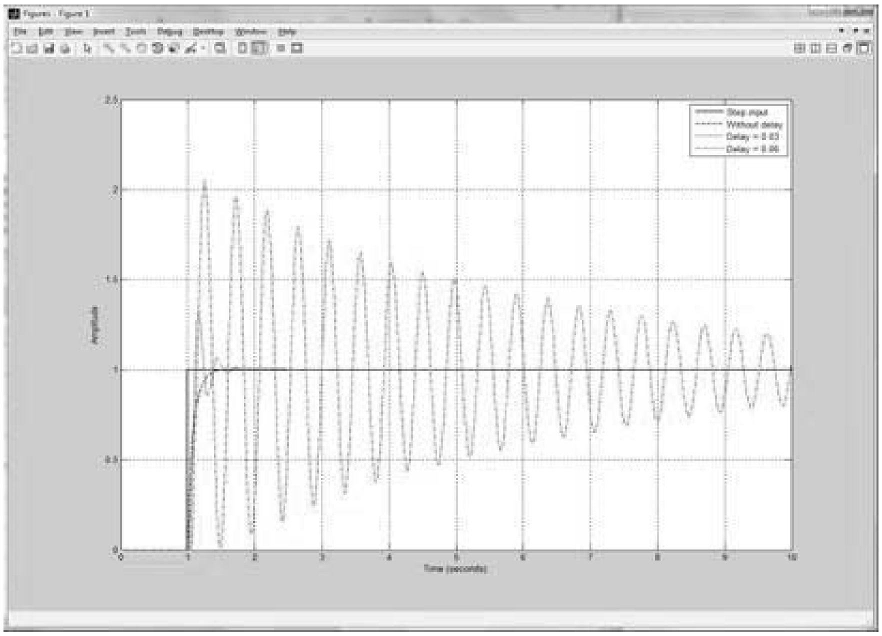


Figure 45. Performance Measures for PID Controller (1) at Different Delays.

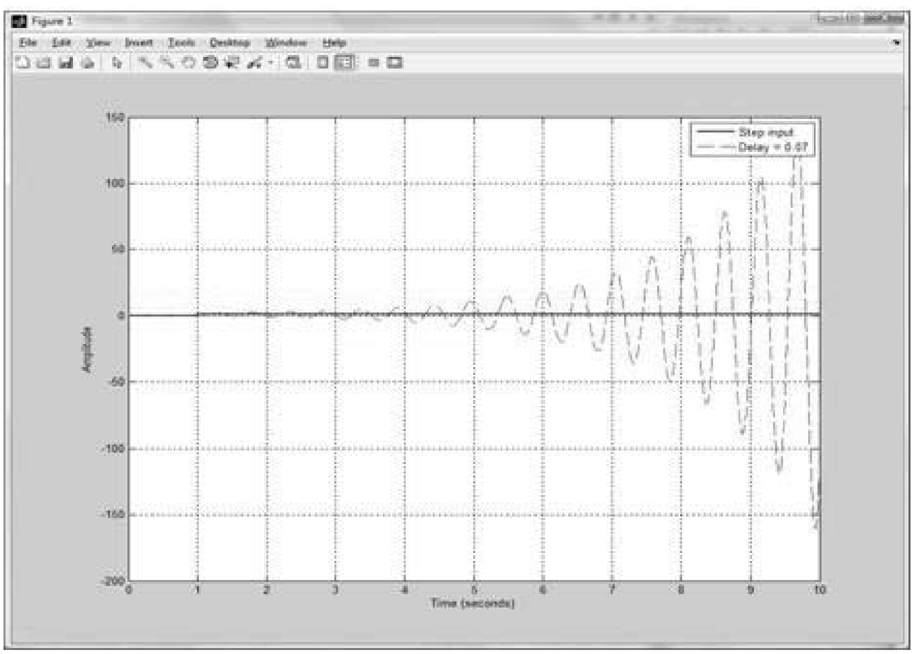


Figure 46. PID Controller (1) Performance when Delays (τ_{sc} and τ_{ca}) = 0.07 s.

Table 18 presents a summary of the step response information obtained after the simulations of the DC Motor with NCS using these PID controller gains. The PID controller response was better than the PI controller without delays with respect to percentage overshoot though the settling time slightly increased. As with the PI controller the percentage overshoot increased with an increase in delay for PID controller also. The settling time increased with an increase in delays for higher delay values but the change was not significant at lower delay values. For example, the percentage overshoot and settling time for 0.02s delay was 8.0% and 0.247s respectively. The percentage overshoot was increased to 81.2% and the settling time also increased to 2.269s when the delays became 0.05s.

Table 18

PID Controller (1) Parameters and Performance Measures.

	Controller Parameters			**Performance Measures**			
Delay (s)	**K_P**	**K_I**	**K_D**	**% Overshoot**	**Peak Time (s)**	**Rise Time (s)**	**Settling Time (s)**
0	0.1701	0.378	0.0075	0.40%	0.615	0.186	0.267
0.01	0.1701	0.378	0.0075	0.50%	0.605	0.146	0.229
0.02	0.1701	0.378	0.0075	8.00%	0.137	0.060	0.247
0.03	0.1701	0.378	0.0075	32.80%	0.162	0.056	0.470
0.04	0.1701	0.378	0.0075	57.00%	0.192	0.056	0.910
0.05	0.1701	0.378	0.0075	81.20%	0.222	0.055	2.269
0.06	0.1701	0.378	0.0075	105.30%	0.254	0.056	14.92

Another PID controller was designed using the same PI controller gains as before but the derivative gain term was further increased from the previous case. The gains used for K_P, K_I and K_D were 0.1701, 0.378 and 0.01. For this controller, the graph of the step input, various outputs when network delays (τ_{sc} and τ_{ca}) have equal values of 0.00s, 0.03s, and 0.05s are shown in Figure 47. This PID controller response was also better than the PI controller without delays with respect to percentage overshoot though the settling time slightly increased. It was also observed that the system performance gradually degraded as the delays increased in values, and became unstable at values of delays beyond 0.05s. For instance, the system became unstable at delays of 0.06s, as shown in Figure 48. However, this PID controller did not improve the performance of the system as it did not accommodate more network delay values than that of the previous PID controller. Therefore, increasing the derivative gain may improve the delay performance only up to a certain derivative gain value after which the performance may degrade.

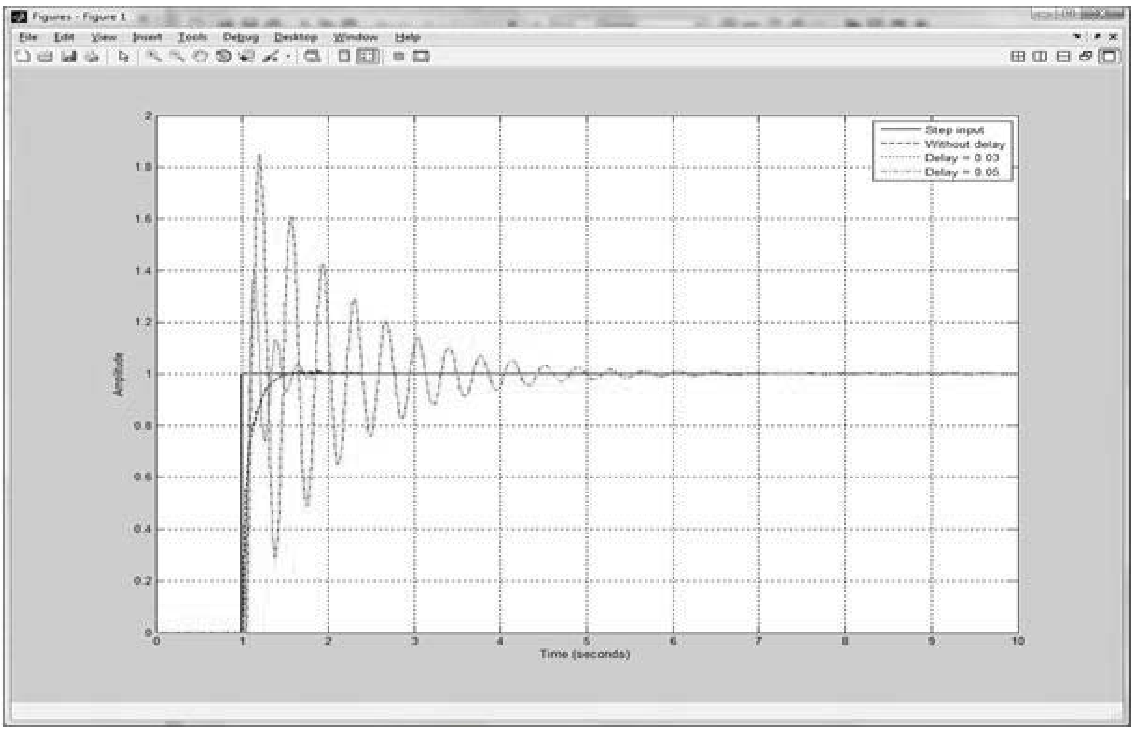


Figure 47. Performance Measures for PID Controller (2) at Different Delays.

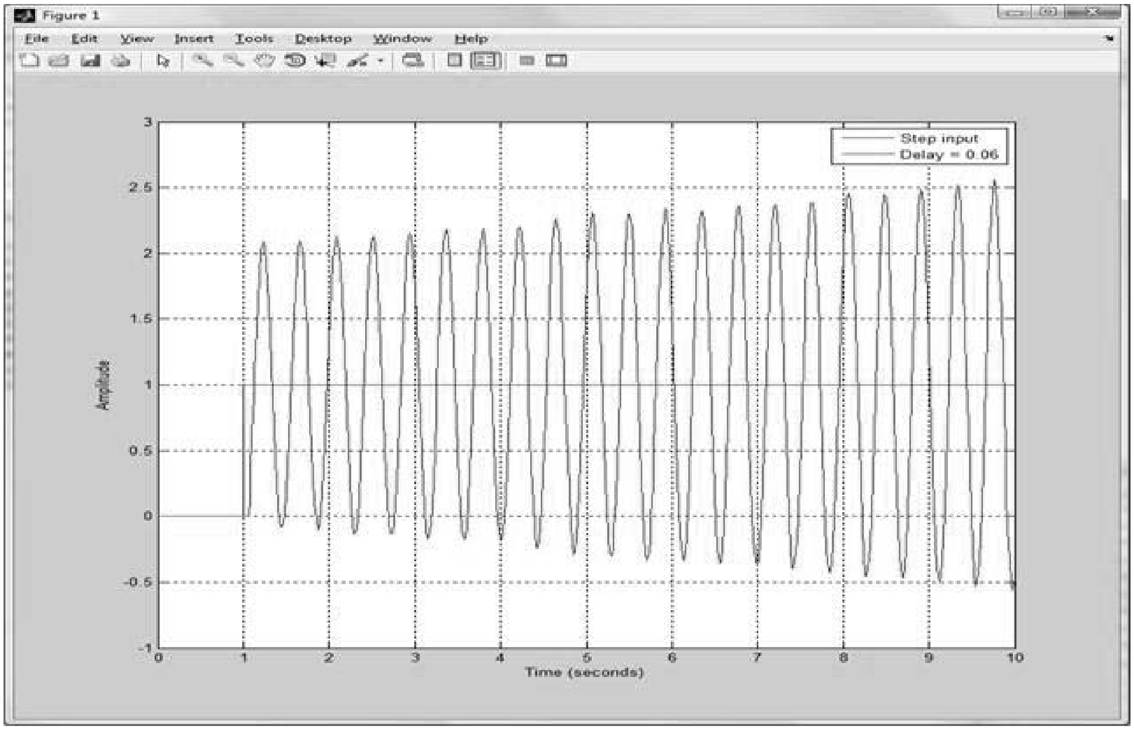


Figure 48. PID Controller (2) Performance when Delays (τ_{sc} and τ_{ca}) = 0.06 s.

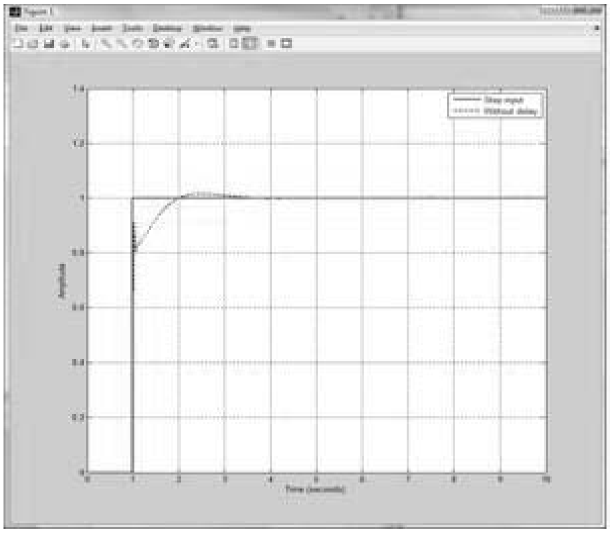
Table 19 presents a summary of the step response information obtained after the simulations of the DC motor with NCS using these PID controller gains. This PID controller response was also better than the PI controller without delays with respect to percentage overshoot though the settling time slightly increased. Similar to the previous PID controller, the percentage overshoot increased with an increase in delay. The settling time increased with an increase in delays for higher delay values but the change was not significant at lower delay values. For example, the percentage overshoot and settling time for 0.02s delay was 11.3% and 0.258s respectively. The percentage overshoot was increased to 85.6% and the settling time also increased to 3.155s when the delays became 0.05s.

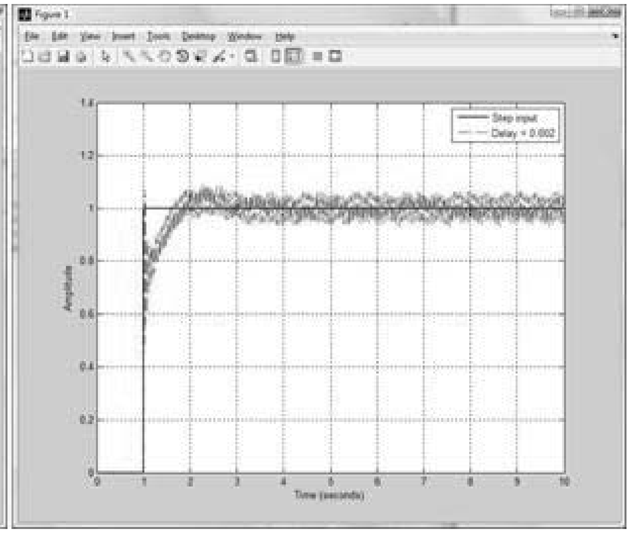
Table 19

PID Controller (2) Parameters and Performance Measures.

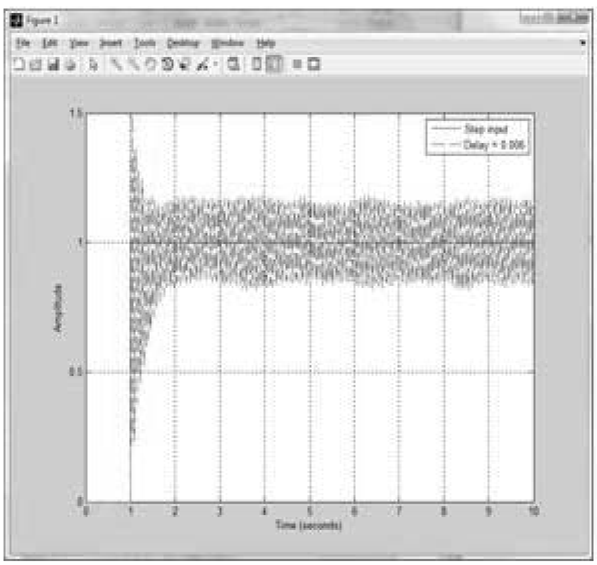
	Controller Parameters			**Performance Measures**			
Delay (s)	**K_P**	**K_I**	**K_D**	**% Overshoot**	**Peak Time (s)**	**Rise Time (s)**	**Settling Time (s)**
0	0.1701	0.378	0.01	0.50%	0.667	0.210	0.301
0.01	0.1701	0.378	0.01	0.70%	0.689	0.175	0.264
0.02	0.1701	0.378	0.01	11.30%	0.112	0.049	0.258
0.03	0.1701	0.378	0.01	38.10%	0.143	0.046	0.543
0.04	0.1701	0.378	0.01	61.60%	0.17	0.047	1.058
0.05	0.1701	0.378	0.01	85.60%	0.204	0.047	3.155

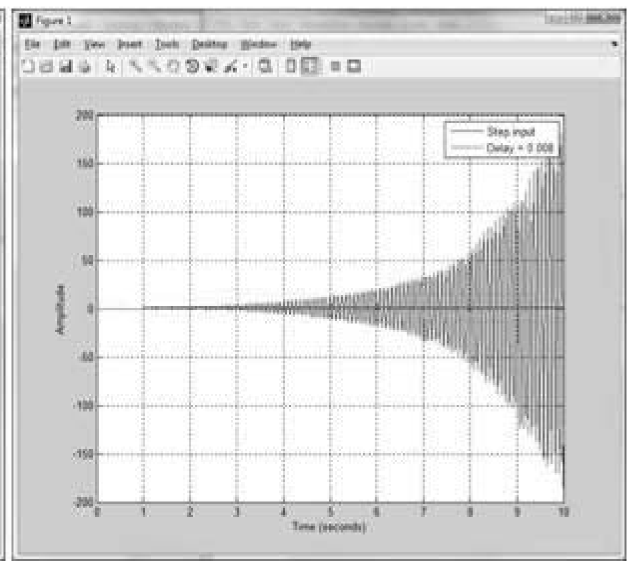
The simulation results for PID controller (3) having the gains for K_P = 0.1701, K_I = 0.378 and that of K_D = 0.05 are presented in Figure 49. This controller has the same gains as the PI controller but the derivative gain was further increased. Four graphs are shown: the system performance when the network delays (τ_{sc} and τ_{ca}) are 0.000s, 0.002 s, 0.006s, and 0.008s. From the figure it can be concluded that the performance of the system degraded when the delay increased. The system however, became unstable when the network delay (τ_{sc} and τ_{ca}) was 0.008s.





a. Network Delays (τ_{sc} and τ_{ca}) = 0. b. Network Delays (τ_{sc} and τ_{ca}) = 0.002s.

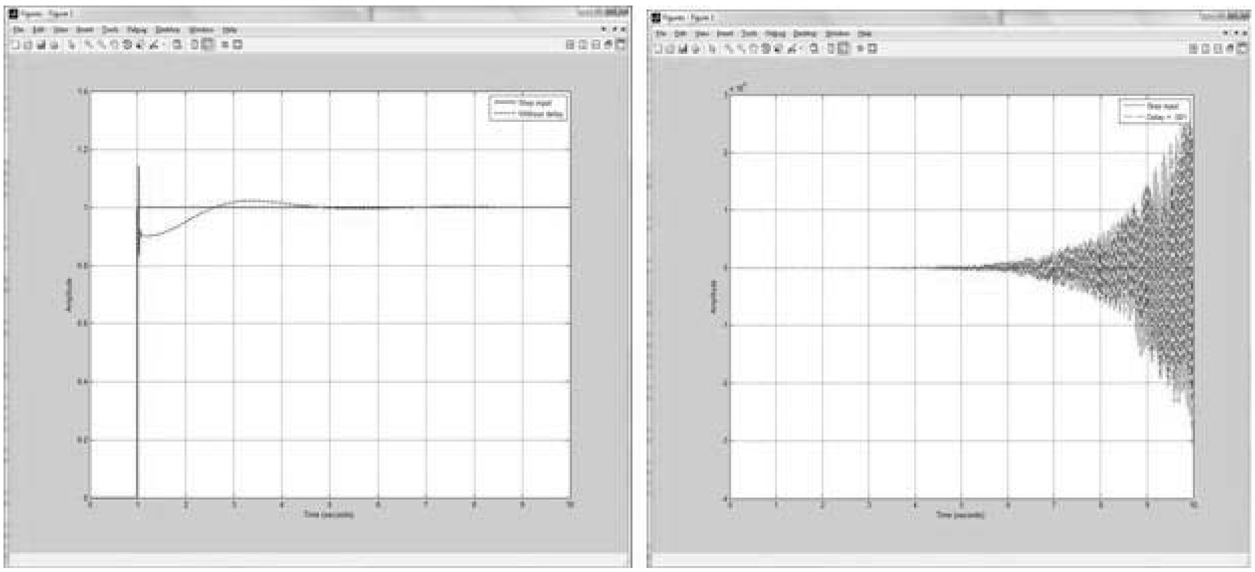




c. Network Delays (τ_{sc} and τ_{ca}) = 0.006s. d. Network Delays (τ_{sc} and τ_{ca}) = 0.008s.

Figure 49. PID Controller (3) Performance Measures under various Network Delay.

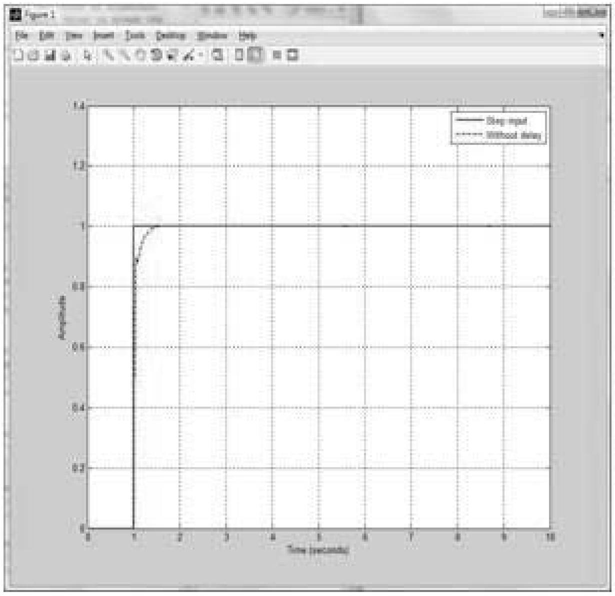
The simulation results for PID controller (4) having the gains for K_P = 0.1701, K_I = 0.378 and that of K_D = 0.13 are presented in Figure 50. This controller has the same gains as the PI controller but the derivative gain was further increased. Two graphs are shown: the system performance when the network delays (τ_{sc} and τ_{ca}) are 0.00s, 0.001s. From the figure it can be concluded that the performance of the system degraded when the network delay was increased. The system became unstable when the network delays (τ_{sc} and τ_{ca}) were 0.001s. This 1ms delay was comparable to the delays that were observed in the automobile CAN system simulation in the previous chapter.

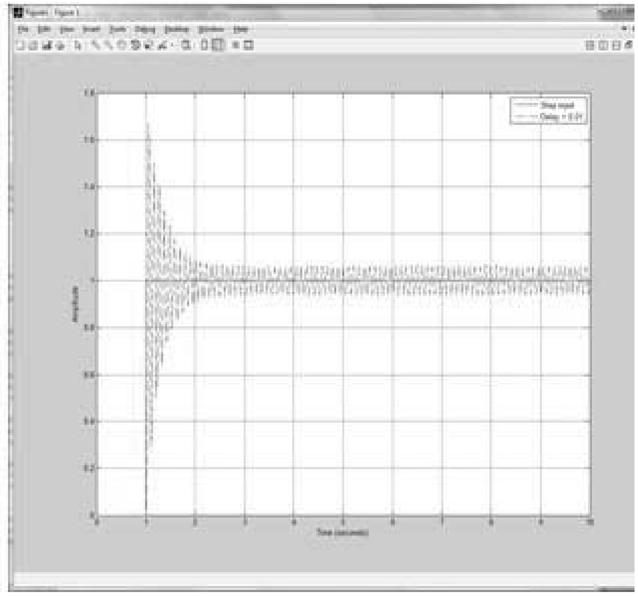


a. Network Delays (τ_{sc} and τ_{ca}) = 0s. b. Network Delays (τ_{sc} and τ_{ca}) = 0.001s.

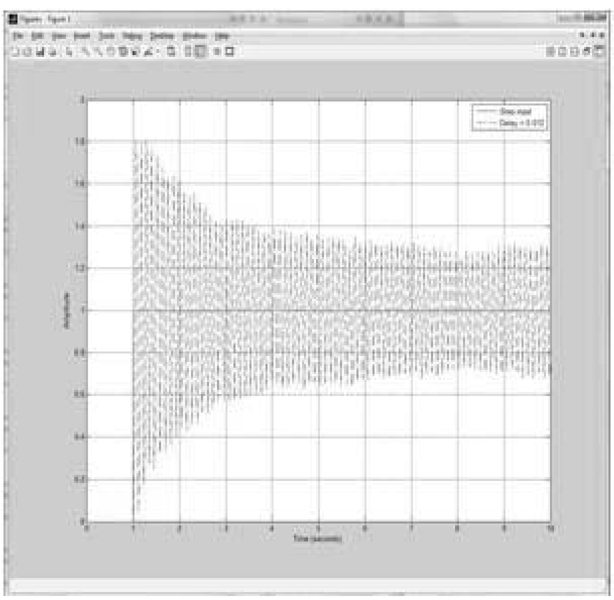
Figure 50. PID Controller (4) Performance Measures Under Various Network Delays.

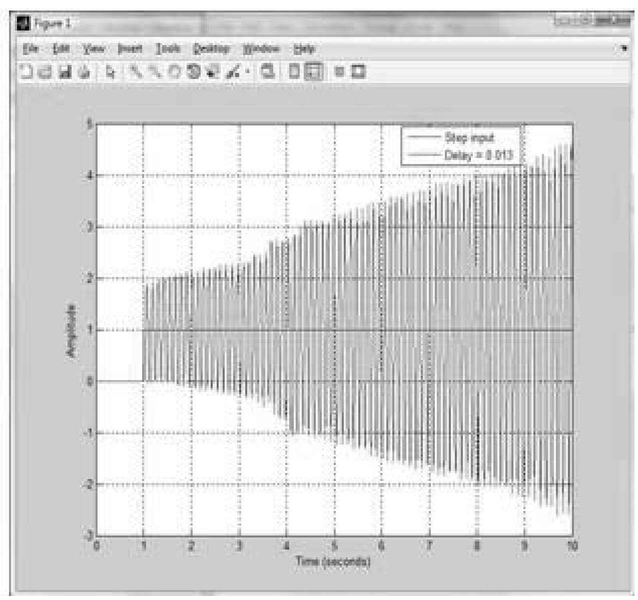
The simulation results for PID controller (5) having the gains for K_P = 0.4, K_I = 0.8 and that of K_D = 0.03 are presented in Figure 51. This controller used different PID gains than the previous cases including the P and I gain terms. Four graphs are shown: the system performance when the network delays (τ_{sc} and τ_{ca}) are 0.0s, 0.01s, 0.012s, and 0.013s. The performance of the system degraded when the delay increased. The system became unstable when the network delays were 0.013s.





a. Network Delays (τ_{sc} and τ_{ca}) = 0. b. Network Delays (τ_{sc} and τ_{ca}) = 0.01s.





c. Network Delays (τ_{sc} and τ_{ca}) = 0.012s. d. Network Delays (τ_{sc} and τ_{ca}) = 0.013s.

Figure 51. PID Controller (5) Performance Measures Under Various Network Delay.

From these simulations, it was observed that fieldbus network induced delay has an effect on control systems stability and performance as described by the system step responses. For instance, comparing the step response results of the DC motor system without network delays and the step response results of the DC motor system with network delays, the system with the delays had higher overshoot and the longer settling time for network delays with higher delay values. From some of the cases observed, the DC motor system became unstable even with a 1ms delay time, which was comparable to the delay times obtained in the automobile CAN system simulation of the previous chapter. The presence of network delays between the sensors, actuators and controllers of the control system did in fact degraded the performance and destabilized the DC motor system.

Also, the PID type controller gains did have effect on control systems with the presence of fieldbus network induced delays. From these MATLAB/Simulink simulations of the DC motor speed control system, it was observed that, as K_D increased, the system performance to accommodate network delays initially improved from PI till $K_D = 0.075$, then it became worse.

Summary of Chapter

This chapter began with the description of the methodology for achieving the objectives of the dissertation that was related to the effects of network induced delays on closed loop control systems. The research questions that were related to this part of the dissertation were restated. The analysis of the effects was carried out using a continuous-time system with PID controller. A DC motor model in transfer function form was used for the system. The various simulations were performed using MATLAB/Simulink and the details of the simulation were explained and presented.

CHAPTER 5

SUMMARY, CONCLUSIONS AND RECOMMENDATIONS

Summary and Conclusions

The introduction of fieldbus networks in industrial systems can improve the efficiency of the control system. The increasing popularity and demand of this technology can be attributed to several advantages such as reduction of time and costs of installation, and maintenance. However, as identified through this dissertation research, one of the challenges related to the deployment of NCS is the effects caused by the inclusion of the communication network in the closed loop control system. The presence of network delays between the sensors, actuators and controllers of the control system can degrade the performance and destabilize the system. The impact of these delays on control system performance measures such as peak overshoot and settling time were investigated. In this dissertation, the causes of network delays and how these delays were affected by CAN parameters such as baud rate, bus load, and message length were also investigated using CANoe simulations of an automobile system.

Next, the research methodologies were discussed. The research hypothesis was developed to statistically test the hypotheses that were related to the first methodology, thus, the analysis of network delays. The analysis was carried out and data was collected from the simulation. Details of the simulation were presented. The data were carefully analyzed to statistically test the hypotheses using descriptive statistics and statistical analysis methods such as one-way ANOVA.

The overall analysis revealed that CAN parameters have effect on CAN message delays. For instance, several samples of delays for engine data signal were collected at three different baud rates (50 kbps, 100 kbps, and 200 kbps). Based on the ANOVA results obtained, there were statistically significant differences between the mean delay times for the different baud rates. It was clear that the mean values of the delays decreased with an increased in baud rates.

Samples of delays for three different signals with different message lengths were collected. The signals were engine data which had a message length of 8 bytes, ABS data which had a message length of 6 bytes, and gearbox info which had a message length of 1 byte. Baud rate selected for this study was 200 kbps. Based on the ANOVA results obtained, there was statistically significant differences between the mean delay times for the different message lengths considered. It was observed that the mean values of the delays increased with the increase in message length.

In addition, several samples of delays for the data signal 3fc (Hex) at two different bus loads were collected. The bus loads considered were 17% and 22%. The baud rate selected for this study was 100 kbps. Based on the ANOVA results, there were statistically significant differences between the mean delay times for the different bus loads considered. It was revealed that the mean values of the delays increased with an increased in busload.

The second methodology for achieving the objectives of this dissertation was related to the effects of network induced delays on closed loop control systems. The delays considered were sensor-to-controller delay (τ_{sc}) and controller to actuator delay (τ_{ca}). The research questions that related to this part of the dissertation study were presented. The analyses of the effects were carried out using a DC motor model in transfer function form with PID controller. The simulations were performed using MATLAB/Simulink and details of the simulation were further presented. From this study, it was observed that fieldbus network induced delays have an effect on control systems stability and performance as described by the system step responses. Comparing the step response results of the DC motor system without network delays and the step response results of the DC motor system with network delays, the system with the delays had higher overshoot and longer settling time for network delays with higher delay values. Also, the PID type controller gains did have effect on control systems with the presence of fieldbus network induced delays. From these MATLAB/Simulink simulations of the DC motor speed control system, it was observed that, as the derivative gain K_D increases, the system performance to accommodate network delays initially improved from PI controller gains till K_D= 0.075, then it became worse. The results of this performance evaluation will be useful to design PID controller gains, and to verify how sensitive the control loops are under various time delays.

Recommendations for Further Studies

This research focused on investigating the causes of network delays and how these delays are affected by CAN parameters such as baud rate, bus load, and message length using CANoe software simulation of an automobile system. Some of these recommendations are listed below.

1. The causes of these delays determined through simulations in this study can be tested using hardware implementation of an automobile CAN system.
2. The causes of delays were investigated for CAN fieldbus based system using software in this dissertation. These causes of delays can further be investigated for other fieldbus based systems such as Profibus, Foundation Fieldbus.
3. While this study concentrated on the impact of baud rates, message length and busloads on CAN message delays, there may be other CAN parameters that might have impact on

the message delays. Identification of these other parameters and a study on their impact on message delays is yet another area of future research that can be pursued.

4. There are mathematical relationships available in the literature for delays in CAN based systems. The statistical analysis methods of delays from this study can be extended to include the verification of the mathematical relationships of delays for CAN based systems.
5. Future research related to this study could derive a mathematical relationship of PID controller gains and network delays for control systems performance and stability.
6. Finally, this research utilized simulations to investigate delays on control system stability and performance measures. Future research related to this study could implement the control system in hardware and test the stability and performance.

REFERENCES

ABB News Center. (2002). ABB and Aventis Sign Global Partnership Agreement. Retrieved May 25, 2011, from http://www.abb.com/cawp/seitp202/c1256c290031524bc1256bd50027ab74.aspx.

Abdel-Ghaffar, H., Hammad, S. and Zaki, A. (2008). H1 Fieldbus Network Delay; A Digital Pole Placement Control Design. *Proceedings of the 17th World Congress the International Federation of Automatic Control*, 9433-9436, Seoul, Korea.

ANSI/ISA-S50.02. (1997). *Part 1: Fieldbus Standard for Use in Industrial Control Systems – Introductory Guide*. USA: ISA.

ANSI/ISA-S50.02. (1997). *Part 4: Fieldbus Standard for Use in Industrial Control Systems – Data Link Protocol Specification*. USA: ISA.

Antsaklis, P. and Baillieul, J. (eds.). (2007). Special Issue on Technology of Networked Control Systems. *Proceedings of IEEE*, 95, 5-8.

Antunes, A., Dias, F., Vieira, J. and Mota, A. (2010). Implementation of the Delay Compensator Approach, Factory Automation, Javier Silvestre-Blanes (Ed.), ISBN: 978-953-307-024-7, InTech, Available from: http://www.intechopen.com/articles/show/title/implementation-of-the-delay-compensator-approach.

Azimi-Sadjadi, B. (2003). Stability of Networked Control Systems in the Presence of Pocket Losses. *Proceedings of the 2003 IEEE Conference on Decision and Control.*

Berge, J. (2002). *Fieldbuses for Process Control: Engineering, Operation, and Maintenance*. USA: ISA.

Bond, A. (2010). Fieldbus Markets Poised to Resume Growth. Retrieved May 20, 2011, from http://www.controlglobal.com/industrynews/2010/008.html.

Bosch. (2006). CAN Specification Version 2.0. Retrieved May 25, 2011, from *esd.cs.ucr.edu/webres/can20.pdf.*

CAN in Automation. (2009). Controller Area Network (CAN). Retrieved May 20, 2011, from http://www.can-cia.org/.

Coulouris, G., Dollimore, J., and Kindberg, T. (2005). *Distributed Systems: Concepts and Design* (3rd ed.). London: Addison-Wesley.

Davis, R. I., Burns, A., Bril, R. J. and Lukkien, J. J. (2007). Controller Area Network (CAN) Schedulability Analysis: Refuted, Revisited and Revised. Real-Time Systems, 35(3), 239-257.

Dorf, R. C. and Bishop, R. H. (12th ed.). (2010). *Modern Control Systems*. USA: Prentice Hall.

Dunn, William. (2005). *Fundamentals of Industrial Instrumentation and Process Control*. USA: McGraw-Hill.

Emadi, A. (2005). *Handbook of Automotive Power Electronics and Motor Drives*. USA: Taylor and Francis Group.

Felser, M. (2002). The Fieldbus Standards: History and Structures. Retrieved May 20, 2011, from http://felser.ch/download/FE-TR-0205.pdf.

Frost and Sullivan. (2009). European Fieldbus Market: Real-Time Data Demand Boosts Fieldbus. Retrieved May 25, 2011, from http://www.researchandmarkets.com/research/95c30f/european_fieldbus_market.

Goettsche, L. D. (2005). *Maintenance of Instruments and Systems*. North Carolina: ISA - International Society of Automation.

Hedjar, R., Al-Qahtani, S., and Al-Goblan, A. (2006). Assymptotic Behavior of Linear Networked Control Systems. *National Computer Conference*, Riyadh, Saudi-Arabia, 26 – 29.

Ji, K. (2006). *Real-Time Control Over Networks*, (Doctoral Dissertation). Available from ProQuest Dissertations and Theses Database. AAT 3270760.

Jianyong, Y., Shimin, Y., and Haiqing, W. (2004). Survey on the Performance Analysis of Networked Control Systems. *IEEE international Conference on Systems, Man and Cybernetics*, 5068-5073.

Keller, G. (2008). *Statistics for Management and Economics* (8th Ed.). USA: South-Western College.

Kolla S., Border, D. and Mayer E., (2003). Fieldbus Networks for Control System Implementations. *Proceedings of Electrical Insulation and Electrical Manufacturing and Coil Winding Technology Conference,* 493-498.

Kumar, M., Verma, A. K. and Srividya, A. (2009). Response-Time Modelling of Controller Area Network (CAN) in Distributed Computing and Networking, V. G. Wattenhofer, and K. Kothapalli, (Eds). Berlin: Springer-Verlag,163-174.

Küster, E. (2005). ARC Fieldbus Survey. Retrieved May 20, 2011, from http://www.industrialnetworks.ch/dateien/referate/ref06-sind2005.pdf.

LeBlanc, C. (2000). The Future of Industrial Networking and Connectivity. Retrieved May 20, 2011, from http://www.dedicated-systems.com/magazine/00q1/2000q1_p009.pdf.

Lee, K. C. and Lee, S. (2002). Implementation of Networked Control System Using a Profibus-DP Network. *International Journal of the KSPE*, 3(3), 12-20.

Li, Q., Jiang, J. and Rankin, D. J. (2011). Evaluation of Delays Induced by Profibus PA Networks. *IEEE Transaction on Instrumentation and Measurements,* 60 (8), 2910-2917.

Lian, F.-L., Moyne, J. R., and Tilbury, D. M. (2005). Network Protocols for Networked Control Systems. *Retrieved from Handbook of Networked and Embedded Control Systems, Control Engineering*, 651-675.

Liu, X. and Goldsmith, A. (2004). Wireless Medium Access Control in Networked Control Systems. *Proceedings of the American Control Conference*.

Loy, D., Dietrich, D., and Schweinzer, H.-J. (Eds.). (2001). *Open Control Networks*. Boston: Kluwer Academic Publishing.

Mahalik, N. P. (2003). *Fieldbus Technology: Industrial Network Standards for Real-Time Distributed Control*. Germany: Springer

Morales-Menendez, R., Garza-Castanon, L., Vargas-Rodriguez R. and Ramirez-Mendoza, R. (2009). Time Delay in Controller Area Network (CAN)-Based Networked Control Systems. *NAMRI/SME Transaction*., 37, 221-228.

National Instruments. (2003). Fieldbus: Foundation Filedbus Overview. Retrieved May 25, 2011, from http://www.ni.com/pdf/manuals/370729a.pdf.

Networx. (2011). Field Networking: Types of Fieldbus Networks. Retrieved May 20, 2011, from http://www.topworx.com/networx/net_process.html.

Norusis, M. J. (2006). *Guide to Data Analysis*. New Jersey: Prentice Hall.

Norusis, M. J. (2009). *SPSS 17.0 Statistical Procedures Companion*. New Jersey: Prentice Hall.

O'Brien, L. (2005). HPI Integration Strategies. Retrieved May 20, 2011, from http://www2.emersonprocess.com/siteadmincenter/PM%20DeltaV%20Documents/Articles/HP_Fieldbus.pdf.

Pang, Y., Yang, S.-H. and Nishitani, H. (2006). Analysis of Control Interval for Foundation Fieldbus-Based Control Systems. *ISA Transactions*, 45 (3), 447 – 458.

Penera, E. A. and Kulkarni, S. S. (2011). Network Performance Considerations for Shipboard Control System. Retrieved May 26, 2011, from http://www.navalengineers.org/SiteCollectionDocuments/2011%20Proceedings%20Documents/ISSIX/Papers/Penera_38_ISS2011Final.pdf.

PROFIBUS PA. (2003). PROFIBUS PA User and Installation Guide. Retrieved May 25, 2011, from http://www.sci-eng.mmu.ac.uk/ascent/literature/documents/A06_Installation_Guide_PA_2092_V22_Feb03.pdf.

Rankin, Q. L., and Jiang, D. J. J. (2009). Evaluation of Delays Induced by Foundation Fieldbus H1 Networks. *IEEE Transactions on Instrumentation and Measurement*. 58(10), 3684-3692.

Rathinasabapathy, S. (2005). *Diagnostic Localization in Chemical Process Industries using Plant Wide causal Patterns*. Ph.D. Dissertation, University of California, Los Angeles, United States, California. Retrieved August 1, 2011, from Dissertations and Theses: AandI. (Publication No. AAT 3196308).

Rylander, A. and Walli, E. (2003). Local Interconnect Network- for Use as Sub-bus in Volvo Trucks. Retrieved May 25, 2011, from http://www.mrtc.mdh.se/rtis/contrib/Exjobb/03.PDF.

Schuessler, B. (2002). Using Smart Valves in a Fieldbus Network: Slashing Costs and Downtime. Retrieved May 20, 2011, from http://www.am.pepperl-fuchs.com/resources/articles/PA/SmartValves-ChemProces1-02.PDF.

Smith, J. (2008). Fieldbus in Focus: Understanding Network Similarities, Differences. Retrieved May 20, 2011, from http://www.plantengineering.com/article/187927-Fieldbus_in_focus_Understanding_network_similarities_differences.php.

Tindell, K., Burns, A. and Wellings, A. J. (1995). Calculating Controller Area Network (CAN) Message Response Times. *Control Engineering Practice*, 3(8), 1163.

Tipsuwan, Y. and Chow, M. Y. (2003). Control Methodologies in Networked Control Systems. *Control Engineering Practice,* 11, 1099-1111.

Vector[CANoe]. (2009). The Professional Development and Test Tool for CAN, LIN, MOST, FlexRay, Ethernet and J1708. Retrieved May 20, 2011, from http://www.vector.com/vi_canoe_en.html.

Voss, W. (2008). *A Comprehensible Guide to Controller Area Network* . Greenfield, MA: Copperhill Media Corporation.

Walker, P. D., Cammy, N. E., Ellis, B. J., and Seiber, K. D. (2011). Operations Skills for the 21st Century. Retrieved May 25, 2011, from http://www.uop.com/wp-content/uploads/2011/04/UOP-Operations-Skills-for-the-21st-Century-Paper.pdf.

Wang, Z., Lu, H., Hedrick, G. E. and Stone, M. (1992). Message Delay Analysis for CAN Based Networks. *Proceedings of ACM/SIGAPP Symp. Applied Computing*, 89-94.

Yu, L. (2007). The Application of Fieldbus and Other Networking Technologies in NPPs: An Overview. Retrieved May 20, 2011, from http://entrac.iaea.org/I-and-C/TM_TORONTO_2007/Start.htm.

Zhang, W., Branicky, M. S., and Phillips, S. M. (2001). Stability of Networked Control Systems. *IEEE Control Systems Magazine*, 21 (1), 84-99.

APPENDIX A: BAUDRATE (TRANSMISSION SPEED) DELAY TIMES

Table A. *Baud-rates Delay Times*

Baud Rate (Transmission Speed) Delay Times (ms)		
50 kbps	**100 kbps**	**200 kbps**
2.241	1.109	0.562
2.242	1.109	0.562
2.246	1.109	0.563
2.248	1.113	0.563
2.248	1.113	0.563
2.252	1.113	0.564
2.252	1.113	0.565
2.253	1.113	0.566
2.259	1.113	0.566
2.259	1.114	0.566
2.259	1.114	0.566
2.262	1.115	0.566
2.262	1.115	0.567
2.264	1.115	0.567
2.265	1.115	0.568
2.266	1.116	0.568
2.266	1.119	0.568
2.268	1.119	0.568
2.268	1.119	0.568
2.27	1.119	0.568
2.271	1.12	0.568
2.271	1.122	0.568
2.272	1.123	0.568
2.272	1.123	0.569
2.272	1.123	0.569
2.272	1.123	0.569
2.272	1.123	0.569
2.273	1.123	0.569
2.273	1.123	0.569
2.273	1.123	0.57
2.273	1.123	0.57
2.273	1.123	0.57
2.273	1.123	0.57

Table A. (*Continued*).

Baud Rate (Transmission Speed) Delay Times (ms)		
50 kbps	**100 kbps**	**200 kbps**
2.273	1.123	0.571
2.273	1.123	0.573
2.277	1.123	0.573
2.28	1.123	0.573
2.281	1.123	0.573
2.283	1.123	0.573
2.286	1.123	0.573
2.287	1.123	0.573
2.288	1.123	0.573
2.288	1.124	0.573
2.289	1.124	0.573
2.29	1.125	0.573
2.29	1.125	0.573
2.29	1.125	0.573
2.29	1.125	0.573
2.29	1.125	0.573
2.291	1.125	0.574
2.291	1.126	0.574
2.292	1.127	0.574
2.292	1.128	0.574
2.292	1.129	0.574
2.292	1.129	0.574
2.292	1.129	0.575
2.292	1.129	0.576
2.292	1.129	0.577
2.292	1.132	0.577
2.292	1.133	0.578
2.292	1.133	0.578
2.292	1.133	0.578
2.292	1.133	0.578
2.292	1.133	0.578
2.292	1.133	0.578
2.292	1.133	0.578
2.292	1.133	0.578
2.292	1.133	0.578

Table A. (*Continued*).

Baud Rate (Transmission Speed) Delay Times (ms)		
50 kbps	**100 kbps**	**200 kbps**
2.292	1.133	0.579
2.293	1.133	0.579
2.293	1.133	0.579
2.293	1.133	0.583
2.293	1.133	0.583
2.293	1.133	0.584
2.293	1.133	0.584
2.293	1.133	0.616
2.293	1.133	0.616
2.293	1.133	0.617
2.293	1.133	0.617
2.293	1.133	0.617
2.293	1.133	0.618
2.293	1.135	0.618
2.293	1.135	0.618
2.293	1.135	0.618
2.293	1.135	0.618
2.293	1.135	0.618
2.293	1.135	0.618
2.293	1.135	0.618
2.295	1.135	0.618
2.3	1.135	0.618
2.302	1.135	0.618
2.303	1.135	0.618
2.303	1.135	0.618
2.305	1.135	0.618
2.306	1.135	0.618
2.306	1.136	0.618
2.306	1.136	0.618
2.308	1.137	0.618
2.308	1.137	0.618
2.309	1.139	0.618
2.31	1.139	0.619
2.31	1.139	0.619
2.311	1.139	0.619

Table A. (*Continued*).

Baud Rate (Transmission Speed) Delay Times (ms)		
50 kbps	**100 kbps**	**200 kbps**
2.312	1.139	0.619
2.312	1.141	0.619
2.312	1.141	0.619
2.312	1.142	0.619
2.312	1.143	0.619
2.312	1.143	0.619
2.312	1.143	0.813
2.312	1.143	0.822
2.312	1.143	0.829
2.312	1.143	0.832
2.312	1.143	0.833
2.312	1.143	0.833
2.312	1.143	0.833
2.312	1.143	0.833
2.312	1.143	0.837
2.312	1.143	0.838
2.312	1.143	0.839
2.312	1.143	0.839
2.312	1.145	0.843
2.312	1.145	0.843
2.312	1.145	0.843
2.312	1.145	0.843
2.312	1.145	0.844
2.312	1.145	0.844
2.312	1.145	0.844
2.312	1.146	0.845
2.313	1.146	0.847
2.313	1.146	0.848
2.313	1.146	0.848
2.313	1.149	0.848
2.313	1.149	0.848
2.313	1.151	0.848
2.313	1.151	0.852
2.313	1.153	0.853
2.313	1.153	0.853

Table A. (*Continued*).

Baud Rate (Transmission Speed) Delay Times (ms)		
50 kbps	**100 kbps**	**200 kbps**
2.313	1.153	0.857
2.313	1.153	0.859
2.313	1.155	0.869
2.313	1.155	0.874
2.313	1.158	0.883
2.313	1.159	0.884
2.313	1.159	0.888
2.313	1.159	0.888
2.313	1.159	0.889
2.313	1.159	0.898
2.313	1.163	0.898
2.314	1.163	0.898
2.315	1.165	0.898
2.315	1.167	0.899
2.317	1.224	0.917
2.326	1.23	0.919
2.326	1.231	0.919
2.328	1.232	0.924
2.328	1.233	0.928
2.329	1.233	0.933
2.329	1.233	0.933
2.329	1.233	0.933
2.332	1.233	0.934
2.332	1.233	0.95
2.332	1.233	0.974
2.332	1.233	0.974
2.332	1.233	0.976
2.332	1.233	0.978
2.332	1.233	0.978
2.332	1.233	0.979
2.332	1.233	0.979
2.332	1.233	0.982
2.332	1.233	0.983
2.332	1.233	0.983
2.332	1.233	0.983

Table A. (*Continued*).

Baud Rate (Transmission Speed) Delay Times (ms)		
50 kbps	**100 kbps**	**200 kbps**
2.332	1.233	0.988
2.332	1.233	0.988
2.332	1.233	0.993
2.332	1.233	0.993
2.332	1.233	0.994
2.332	1.233	0.998
2.332	1.233	0.998
2.333	1.233	1.029
2.333	1.233	1.032
2.333	1.233	1.033
2.333	1.234	1.033
2.333	1.234	1.033
2.333	1.234	1.034
2.333	1.234	1.034
2.333	1.235	1.034
2.333	1.235	1.034
2.333	1.235	1.123
2.333	1.235	1.128
2.336	1.235	1.128
2.352	1.235	1.131
2.352	1.235	1.133
2.352	1.235	1.133
2.352	1.235	1.133
2.352	1.235	1.133
2.352	1.235	1.133
2.352	1.235	1.136
2.352	1.235	1.138
2.353	1.235	1.138
2.353	1.235	1.143
2.353	1.235	1.144
2.353	1.235	1.148
2.368	1.237	1.148
2.373	1.238	1.153
2.373	1.238	1.158
2.373	1.66	1.163

APPENDIX B: MESSAGE LENGTH (0 TO 8 BYTES) DELAY TIMES

Table B. *Message Length Delay Times*

Message Length (0 to 8 bytes) Delays Time (ms)		
Data Length_64	**Data Length_C9**	**Data Length _3FC**
0.562	0.483	0.275
0.562	0.485	0.275
0.563	0.485	0.275
0.563	0.485	0.275
0.563	0.486	0.275
0.564	0.487	0.276
0.565	0.488	0.276
0.566	0.488	0.276
0.566	0.488	0.276
0.566	0.488	0.276
0.566	0.488	0.276
0.566	0.489	0.277
0.567	0.489	0.277
0.567	0.489	0.277
0.568	0.49	0.278
0.568	0.49	0.278
0.568	0.49	0.278
0.568	0.49	0.278
0.568	0.49	0.278
0.568	0.491	0.278
0.568	0.491	0.278
0.568	0.491	0.278
0.568	0.492	0.279
0.569	0.492	0.279
0.569	0.492	0.279
0.569	0.492	0.279
0.569	0.492	0.279
0.569	0.493	0.279
0.569	0.493	0.279
0.57	0.493	0.279
0.57	0.493	0.279
0.57	0.494	0.279
0.57	0.494	0.279

Table B. *(Continued).*

Message Length (0 to 8 bytes) Delays Time (ms)		
Data Length_64	**Data Length_C9**	**Data Length _3FC**
0.571	0.494	0.279
0.573	0.494	0.279
0.573	0.494	0.279
0.573	0.494	0.279
0.573	0.494	0.279
0.573	0.494	0.279
0.573	0.494	0.279
0.573	0.494	0.279
0.573	0.494	0.279
0.573	0.495	0.279
0.573	0.495	0.279
0.573	0.495	0.279
0.573	0.495	0.279
0.573	0.495	0.279
0.573	0.495	0.279
0.573	0.495	0.279
0.574	0.495	0.279
0.574	0.496	0.279
0.574	0.496	0.28
0.574	0.496	0.28
0.574	0.496	0.28
0.574	0.496	0.28
0.575	0.497	0.281
0.576	0.497	0.281
0.577	0.498	0.281
0.577	0.498	0.281
0.578	0.499	0.281
0.578	0.499	0.281
0.578	0.499	0.282
0.578	0.499	0.282
0.578	0.5	0.282
0.578	0.5	0.282
0.578	0.5	0.283
0.578	0.503	0.284
0.578	0.504	0.284

Table B. *(Continued).*

Message Length (0 to 8 bytes) Delays Time (ms)		
Data Length_64	**Data Length_C9**	**Data Length _3FC**
0.579	0.511	0.284
0.579	0.511	0.284
0.579	0.511	0.284
0.583	0.511	0.284
0.583	0.512	0.284
0.584	0.512	0.284
0.584	0.512	0.284
0.616	0.513	0.284
0.616	0.513	0.284
0.617	0.513	0.284
0.617	0.513	0.284
0.617	0.513	0.284
0.618	0.513	0.284
0.618	0.513	0.284
0.618	0.513	0.284
0.618	0.513	0.284
0.618	0.513	0.285
0.618	0.513	0.285
0.618	0.513	0.513
0.618	0.513	0.522
0.618	0.513	0.529
0.618	0.513	0.544
0.618	0.513	0.544
0.618	0.513	0.544
0.618	0.514	0.549
0.618	0.514	0.549
0.618	0.514	0.549
0.618	0.515	0.549
0.618	0.515	0.549
0.618	0.515	0.549
0.618	0.515	0.549
0.618	0.515	0.554
0.619	0.515	0.554
0.619	0.515	0.554
0.619	0.682	0.554

Table B. *(Continued).*

Message Length (0 to 8 bytes) Delays Time (ms)		
Data Length_64	**Data Length_C9**	**Data Length _3FC**
0.619	0.724	0.554
0.619	0.752	0.554
0.619	0.758	0.554
0.619	0.758	0.556
0.619	0.759	0.559
0.619	0.762	0.559
0.813	0.763	0.559
0.822	0.763	0.559
0.829	0.763	0.559
0.832	0.763	0.559
0.833	0.763	0.559
0.833	0.765	0.559
0.833	0.768	0.563
0.833	0.768	0.564
0.837	0.773	0.564
0.838	0.773	0.564
0.839	0.775	0.564
0.839	0.778	0.564
0.843	0.778	0.564
0.843	0.778	0.567
0.843	0.779	0.589
0.843	0.78	0.594
0.844	0.78	0.594
0.844	0.78	0.594
0.844	0.783	0.594
0.845	0.783	0.599
0.847	0.783	0.599
0.848	0.784	0.602
0.848	0.785	0.635
0.848	0.789	0.639
0.848	0.793	0.639
0.848	0.793	0.639
0.852	0.793	0.639
0.853	0.793	0.639
0.853	0.798	0.644

Table B. *(Continued).*

Message Length (0 to 8 bytes) Delays Time (ms)		
Data Length_64	**Data Length_C9**	**Data Length _3FC**
0.857	0.799	0.644
0.859	0.825	0.649
0.869	0.825	0.649
0.874	0.829	0.659
0.883	0.83	0.663
0.884	0.848	0.689
0.888	0.85	0.69
0.888	0.853	0.693
0.889	0.855	0.694
0.898	0.86	0.694
0.898	0.863	0.694
0.898	0.873	0.694
0.898	0.873	0.694
0.899	0.873	0.694
0.917	0.875	0.694
0.919	0.875	0.694
0.919	0.878	0.694
0.924	0.878	0.694
0.928	0.878	0.694
0.933	0.878	0.694
0.933	0.898	0.694
0.933	0.903	0.694
0.934	0.903	0.694
0.95	0.905	0.694
0.974	0.908	0.696
0.974	0.908	0.699
0.976	0.913	0.699
0.978	0.913	0.699
0.978	0.913	0.699
0.979	0.923	0.699
0.979	0.928	0.834
0.982	0.928	0.834
0.983	0.929	0.839
0.983	0.93	0.839
0.983	1.04	0.839

Table B. *(Continued).*

Message Length (0 to 8 bytes) Delays Time (ms)		
Data Length_64	**Data Length_C9**	**Data Length _3FC**
0.988	1.044	0.839
0.988	1.044	0.839
0.993	1.045	0.844
0.993	1.048	0.844
0.994	1.049	0.844
0.998	1.049	0.844
0.998	1.05	0.844
1.029	1.053	0.844
1.032	1.053	0.844
1.033	1.053	0.844
1.033	1.053	0.844
1.033	1.054	0.844
1.034	1.058	0.844
1.034	1.058	0.848
1.034	1.058	0.849
1.034	1.058	0.849
1.123	1.058	0.849
1.128	1.058	0.849
1.128	1.058	0.849
1.131	1.058	0.849
1.133	1.058	0.849
1.133	1.058	0.849
1.133	1.058	0.849
1.133	1.063	0.849
1.133	1.063	0.849
1.136	1.063	0.854
1.138	1.063	0.854
1.138	1.063	0.854
1.143	1.064	0.854
1.144	1.065	0.854
1.148	1.068	0.859
1.148	1.07	0.859
1.153	1.073	0.864
1.158	1.073	0.864
1.163	1.08	0.869

APPENDIX C: BUSLOAD (% OF BUS ACTIVITY) DELAY TIMES

Table C. *Busload Delay Times*

Busload (% of bus activity) Delays Time (ms)	
Busloads (22%)	**Busloads (17%)**
0.549	0.554
0.549	0.559
0.549	0.559
0.549	0.559
0.55	0.559
0.55	0.559
0.55	0.559
0.551	0.559
0.551	0.559
0.551	0.559
0.551	0.559
0.551	0.559
0.551	0.559
0.551	0.56
0.552	0.56
0.552	0.56
0.552	0.56
0.552	0.561
0.553	0.561
0.553	0.562
0.553	0.562
0.553	0.562
0.554	0.562
0.554	0.562
0.554	0.563
0.555	0.563
0.555	0.563
0.555	0.564
0.555	0.564
0.555	0.564
0.555	0.564
0.556	0.564
0.556	0.564

Table C. *(Continued).*

Busload (% of bus activity) Delays Time (ms)	
Busloads (22%)	**Busloads (17%)**
0.556	0.565
0.557	0.565
0.557	0.565
0.557	0.565
0.557	0.565
0.557	0.565
0.558	0.565
0.558	0.565
0.559	0.565
0.559	0.565
0.559	0.565
1.369	0.566
1.418	0.566
1.664	0.566
1.705	0.566
1.708	0.567
1.709	0.567
1.714	0.567
1.714	0.567
1.715	0.568
1.716	0.568
1.717	0.568
1.718	0.569
1.718	0.569
1.718	1.157
1.718	1.318
1.718	1.428
1.719	1.684
1.725	1.69
1.735	1.732
2.174	1.732
2.177	1.733
2.177	1.734
2.177	1.734
2.177	1.734

Table C. *(Continued).*

Busload (% of bus activity) Delays Time (ms)	
Busloads (22%)	**Busloads (17%)**
2.267	1.737
2.317	1.737
2.477	1.737
2.477	1.737
2.477	1.737
2.866	1.745
2.876	1.745
2.882	2.182
2.883	2.182
2.884	2.182
2.887	2.182
2.897	2.183
2.905	2.183
2.907	2.183
2.913	2.183
2.937	2.183
3.325	2.183
3.332	2.183
3.333	2.183
3.342	2.183
3.343	2.183
3.345	2.183
3.346	2.183
3.352	2.183
3.353	2.183
3.353	2.184
3.356	2.184
3.476	2.184
3.51	2.184
3.646	2.184
3.941	2.184
4.004	2.184
4.01	2.186
4.023	2.186
4.023	2.186

Table C. *(Continued).*

Busload (% of bus activity) Delays Time (ms)	
Busloads (22%)	**Busloads (17%)**
4.075	2.187
4.085	2.187
4.114	2.187
4.303	2.187
4.494	2.912
4.495	2.912
4.495	2.916
4.504	2.922
4.514	2.926
4.524	3.164
4.541	3.372
4.543	3.372
4.628	3.434
5.159	3.434
5.173	3.434
5.182	3.434
5.182	3.434