

# Slotted Time Based Protocol and Polling Strategies for Low Speed, Narrow Band Applications

Ronald Ntakadzem, Ragadza



Thesis presented in partial fulfillment of the requirements for the  
Masters of Science in Engineering Sciences  
at the Stellenbosch University

Supervisor: Dr. Riaan Wolhuter

Department of Electrical and Electronic Engineering

October 2004

# Declaration

I, the undersigned, hereby declare that the work contained in this thesis is my own original work and has not previously in its entirety or in part been submitted at any university for a degree.

# Abstract

This thesis investigates and develops some aspects of low speed wireless telemetry applications. The system configuration is one of a single master, controlling and communicating with multiple slave stations over a radio network. The aim of this thesis is to develop a narrow band wireless communications network based on a protocol with slotted time base, utilizing standard, off the shelf hardware. The idea is to investigate different strategies for improvement of data throughput performance under conditions of both low- and high loading, using this basic protocol. Such strategies may, amongst others include collision detection, flow control and pre-scheduling algorithms.

This thesis will also show the how the non-adaptive system's data throughput performance can be improved using an adaptive system under low- and high loading. Half-duplex communication mode will be used for data communication.

# Opsomming

Hierdie tesis ondersoek en ontwikkel sommige aspekte van laespoed radiotelemetrie toepassings. Die stelselkonfigurasie is een van 'n enkel meesterstasie wat met 'n aantal buitestasies kommunikeer oor 'n radionetwerk. Die doel van die tesis is om 'n tydgegleufde protokol te ontwikkel vir nouband radiotoepassings, deur gebruik te maak van standaard, maklik beskikbare hardeware. Die doel is om verskillende benaderings te ondersoek vir die verbetering van data deurset onder toestande van beide hoë- en lae belading. Sulke strategieë mag botsingsdeteksie, vloeibeheer en voor-skedulering insluit.

Die tesis sal ook aandui hoe die data deurset van die nie-aanpasbare metode verbeter kan word, deur gebruik te maak van 'n aanpasbare metode vir beide beladingstipes. Die kommunikasiestelsel is half-dupleks gebaseer.

To my grand father, Joseph Ragadza

# Acknowledgements

I wish to thank the following people:

- Dr Riaan Wolhuter, my supervisor, for his guidance and advice. Without his advice, I would have gotten lost along the way.
- Sister Ntangedzeni Nematatani, for believing in my strength and for always encouraging me to work hard without putting pressure on myself.
- My parents, for giving me their undivided support and encouraging me to work hard during sleepless nights.
- Ramalata Phillip, for proof reading.
- Institute for Satellite and Software Applications (ISSA) for their financial support.
- I also want to thank everybody who contributed for moral support.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Introduction . . . . .	1
1.2	Objectives . . . . .	2
1.3	Application Area . . . . .	3
1.4	Thesis Outline . . . . .	4
<b>2</b>	<b>Related Research</b>	<b>6</b>
2.1	The OSI Model . . . . .	6
2.2	A Data Communication Model . . . . .	9
2.2.1	Data communication tasks . . . . .	10
2.2.2	Data communication modes . . . . .	11
2.2.3	Timing of digital data communication system . . . . .	12
2.3	Device Interfacing . . . . .	14
2.3.1	Physical connection . . . . .	14
2.3.2	Wireless connection . . . . .	16
2.4	Overview of Time Slotted and Non-slotted Protocols . . . . .	18

<i>CONTENTS</i>	iii
2.4.1 Introduction . . . . .	18
2.4.2 Aloha protocol . . . . .	18
2.4.3 Slotted aloha protocol . . . . .	19
2.4.4 Carrier Sense Multiple Access (CSMA) protocols . . . . .	20
2.5 Data Transmission Through Polling . . . . .	23
2.5.1 Round-Robin polling . . . . .	23
2.5.2 Priority polling . . . . .	24
2.6 Contention Protocols versus Fixed Assignment Protocols . . . . .	25
2.7 Summary . . . . .	26
<b>3 Error Management and Control</b>	<b>27</b>
3.1 Introduction . . . . .	27
3.2 Sources of Error . . . . .	27
3.2.1 Types of errors . . . . .	28
3.3 Error Detection . . . . .	29
3.3.1 Introduction . . . . .	29
3.3.2 What motivates the use of error detection techniques? . . . . .	29
3.3.3 Parity checking . . . . .	30
3.3.4 Block check . . . . .	31
3.4 Flow Control Techniques . . . . .	32
3.4.1 Stop-and-wait flow control . . . . .	32

<i>CONTENTS</i>	iv
3.4.2 Sliding window flow control . . . . .	35
3.5 Error Control Techniques . . . . .	35
3.5.1 Introduction . . . . .	35
3.5.2 Backward error control . . . . .	36
3.5.3 Forward Error Correction . . . . .	37
3.6 Summary . . . . .	37
<b>4 Standard IEC 870-5-101 protocol</b>	<b>38</b>
4.1 Introduction . . . . .	38
4.2 Protocol Communication Modes . . . . .	38
4.2.1 Balanced mode . . . . .	39
4.2.2 Unbalanced mode . . . . .	39
4.3 Network Configurations . . . . .	39
4.4 Framing . . . . .	40
4.4.1 Transmission frame format . . . . .	40
4.4.2 Meaning of the fields . . . . .	41
4.5 Transmission Rules . . . . .	44
4.6 Communication Process . . . . .	44
4.6.1 Initialization of slave station . . . . .	45
4.6.2 Synchronization . . . . .	45
4.6.3 Request/respond procedure . . . . .	46

<i>CONTENTS</i>	v
4.6.4 Send/Confirm procedure . . . . .	47
4.7 Summary . . . . .	47
<b>5 System Design and Design Considerations</b>	<b>48</b>
5.1 Master Configuration . . . . .	49
5.1.1 Master model . . . . .	49
5.2 Slave Configurations . . . . .	50
5.2.1 Slave station model . . . . .	50
5.3 Radio Modem Configurations . . . . .	51
5.4 Signal Processing Between a Master and Slave Stations . . . . .	51
5.5 Summary . . . . .	52
<b>6 Implementations</b>	<b>53</b>
6.1 Software Platform . . . . .	53
6.2 Implementation of Error Detection Algorithm . . . . .	54
6.3 Time-slots Implementation . . . . .	56
6.4 Non-adaptive Polling System . . . . .	58
6.4.1 RRP implementation . . . . .	58
6.4.2 Priority polling algorithm . . . . .	59
6.5 Adaptive Polling System . . . . .	62
6.6 Slave Station Implementation . . . . .	64
6.7 Summary . . . . .	66

<i>CONTENTS</i>	vi
<b>7 Results and System Reliability</b>	<b>67</b>
7.1 System Reliability . . . . .	67
7.2 Results . . . . .	70
7.2.1 Introduction . . . . .	70
7.2.2 Acceptable 3:1:1:1 priority polling results . . . . .	71
7.2.3 Performance comparison between adaptive and non-adaptive polling schemes . . . . .	75
7.3 Summary . . . . .	90
<b>8 Conclusions and Recommendations</b>	<b>91</b>
8.1 Conclusions . . . . .	91
8.2 Recommendations . . . . .	92
<b>Appendices</b>	<b>92</b>
<b>A RS232C Control Lines</b>	<b>93</b>
A.1 PC to modem connections . . . . .	94
<b>B RS232 specifications</b>	<b>95</b>
<b>C Commonly used Delphi functions and Source code</b>	<b>96</b>
<b>D Radio Modem and Its Specifications</b>	<b>99</b>

# List of Figures

2.1	Data transmission in accordance with the OSI Model . . . . .	9
2.2	Basic data communication model . . . . .	9
2.3	Simplex, Half-Duplex, and Full-Duplex communication . . . . .	12
2.4	Asynchronous serial transmission of ASCII 'M' . . . . .	13
2.5	Synchronous Frame Format . . . . .	13
2.6	DB-9 male connector . . . . .	16
2.7	Pure Aloha protocol . . . . .	18
2.8	Slotted Aloha protocol . . . . .	19
2.9	Comparison of performance between pure ALOHA and slotted ALOHA . . . . .	20
2.10	Throughput for various contention modes . . . . .	22
3.1	Four different scenarios for the stop-and-wait algorithm.(a) The ACK is received before the time expires; (b) The original frame is lost; (c) the ACK is lost; (d) The timeout is too short. . . . .	33
3.2	Stop-and-wait ARQ with 1-bit sequence number . . . . .	34
4.1	Network configurations supported by the IEC 870-5-101 protocol . . . . .	40

*LIST OF FIGURES*

viii

4.2	Frame formats defined by the standard IEC 870-5-101 protocol . . . . .	41
5.1	Designed system . . . . .	48
6.1	Implementation steps of CRC-16 . . . . .	55
6.2	Implementation of time slots . . . . .	58
6.3	Master station data flow when polling slaves using RRP . . . . .	59
6.4	Polling slaves with events generation ratio 8:4:2:1 . . . . .	60
6.5	Polling of slaves with event generation ratio of 3:1:1:1 . . . . .	61
6.6	Adaptive system procedure . . . . .	63
6.7	Process undergone by a slave station when receiving a poll message . . . . .	65
7.1	Serial port communication setting component . . . . .	68
7.2	Master station control command . . . . .	68
7.3	Polling type control page . . . . .	69
7.4	Initialization of slave using IEC 870-5-101 protocol . . . . .	69
7.5	High loading performance of slave 1 using 3:1:1:1 priority ratio . . . . .	71
7.6	High loading performance of slave 2 using 3:1:1:1 priority ratio . . . . .	72
7.7	High loading performance of slave 3 using 3:1:1:1 priority ratio . . . . .	72
7.8	High loading performance of slave 4 using 3:1:1:1 priority ratio . . . . .	73
7.9	Slave 1 high loading mean wait time . . . . .	73
7.10	Slave 2 high loading mean wait time . . . . .	74
7.11	Slave 3 high loading mean wait time . . . . .	74

*LIST OF FIGURES*

ix

7.12 Slave 4 high loading mean wait time . . . . .	75
7.13 Slave 1 data throughput comparison between adaptive and non-adaptive polling system . . . . .	76
7.14 Slave 2 data throughput comparison between adaptive and non-adaptive polling system . . . . .	77
7.15 Slave 3 data throughput comparison between adaptive and non-adaptive polling system . . . . .	77
7.16 Slave 4 data throughput comparison between adaptive and non-adaptive polling system . . . . .	78
7.17 Slave 1 delay comparison between adaptive and non-adaptive polling system	78
7.18 Slave 2 delay comparison between adaptive and non-adaptive polling system	79
7.19 Slave 3 delay comparison between adaptive and non-adaptive polling system	79
7.20 Slave 4 delay comparison between adaptive and non-adaptive polling system	80
7.21 Slave 1: Throughput comparison between adaptive and non-adaptive system	81
7.22 Slave 2: Throughput comparison between adaptive and non-adaptive system	82
7.23 Slave 3: Throughput comparison between adaptive and non-adaptive system	82
7.24 Slave 4: Throughput comparison between adaptive and non-adaptive system	83
7.25 Delays comparison between adaptive and non-adaptive system: Slave 1 . .	83
7.26 Delays comparison between adaptive and non-adaptive system: Slave 2 . .	84
7.27 Delay comparisons between adaptive and non-adaptive system: Slave 3 . .	84
7.28 Delay comparisons between adaptive and non-adaptive schemes: Slave 4 . .	85
7.29 Slave 1: Low to high loading comparison . . . . .	86

*LIST OF FIGURES*

x

7.30 Slave 2: Low to high loading comparison . . . . .	87
7.31 Slave 3: Low to high loading comparison . . . . .	87
7.32 Slave 4: Low to high loading comparison . . . . .	88
7.33 Slave 1: Low to high loading latency comparison . . . . .	88
7.34 Slave 2: Low to high loading latency comparison . . . . .	89
7.35 Slave 3: Low to high loading latency comparison . . . . .	89
7.36 Slave 4: Low to high loading latency comparison . . . . .	90
A.1 DB-25 to DB9 Adapter Cable . . . . .	94
D.1 EL705 OEM Series 450 Mhz Data Transceiver . . . . .	99

# List of Tables

2.1	The seven layers of the OSI Model . . . . .	7
3.1	Parity check operation . . . . .	30
4.1	Coding of C Field . . . . .	42
4.2	Function code of control fields in messages sent from master station: Unbalanced mode . . . . .	42
4.3	Function code of control fields in messages sent from slave station: unbalanced mode . . . . .	43
A.1	DB-9 and DB-25 pin connectors . . . . .	93

# Glossary

ACK	Acknowledgement
BEC	Backward Error Correction
BER	Bit Error Rate
CRC	Cyclic Redundancy Check
CSMA	Carrier Sense Multiple Access
CSV	Comma Separated files
CTS	Clear To Send
DCE	Data Communication Equipment
DTE	Data Terminal Equipment
DTR	Data Terminal Ready
FCS	Frame Check Sequence
FEC	Forward Error Correction
HDLC	High Level Data Link
ISO	International Organization of Standards
LSB	Least Significant Bit
MODEM	Modulator/Demodulator
MSB	Most Significant Bit
NACK	Negative Acknowledge
OSI	Open System Interconnection
PC	Personal Computer
REQ_UD	Request User Data
RF	Radio Frequency
RRP	Round-Robin Polling
RSP_UD	Respond User Data
RxD	Receive data
SCADA	Supervisory Control And Data Acquisition
TxD	Transmit data

# Chapter 1

## Introduction

### 1.1 Introduction

There are many forms of wireless communication in information superhighways nowadays. The need for fast and reliable wireless data transfer is exploding. The race of technology leads to new developments and improvements to the systems that have already existed. As wireless products have gained mainstream acceptance, developers are hard at work applying wireless technologies to free applications such as inventory-process and remote control from wiring. At low speed, the probability of error in the system is low due to lower number of bits in error. The ability to communicate with anyone on the planet from anywhere on the planet has been mankind's dream for a long time. Wireless is one of the mediums that can enable such untethered communication.

Wireless telemetry is defined as the transfer of data over a wide-area wireless network between two or more non-mobile (normally fixed but can be portable) machines for monitoring, recording or controlling purposes. Although "wire-free" telemetry has been around for some time, the applications for the technology are continuing to grow as more and more equipment and systems can be integrated to enhance the capabilities of the technology.

Different types of polling schemes have been used by the base station for monitoring the processes in the remote stations. Improvements have been made in the polling schemes so that they may be able to produce a better performance in today's technology. This thesis will be focused on wireless telemetry systems where the locations of the slave stations are non-mobile with respect to the master station's position. Reliability of a system depends mostly on the integrity of the communication process. For a communication process to

occur, devices must be interfaced correctly within the (master and slave) stations.

Radio offers superior characteristics as a wireless media because the omnidirectional radio transmitters can easily penetrate walls, floors, ceiling and the like. Electrically speaking, the waves that are classified as radio waves have certain frequencies that are grouped together for certain uses. Some are available for data transmission, but the bandwidth necessary to perform high speed data transfers is not found at any given slot on the radio spectrum (that is, communications over a radio link is limited to low speed). Radio, though limited by its speed, may be the wireless transmission of choice for many desktops because of its low cost and capabilities.

## 1.2 Objectives

The aim of this thesis is to develop a wireless communications network based on a time-slotted protocol using off the shelf narrow band hardware to utilize the improvement in performance of slotted protocols over non slotted versions. In practical, commercially available systems, this has not been done due to complexity and the normally substantial legacy of an installed base. This has been viewed as a shortcoming preventing definite possibilities of performance improvement. Due to widespread use of such installations, these enhancements offer good commercial benefits as well.

During the development of the time-slotted protocol, a further very worthwhile objective manifested itself. Too many non-standardized one-off protocols have been described, without wide acceptance and support. It was, therefore, decided to implement the protocol structure in line with a common international standard, such as IEC 870-5-101. This would enable the application of the development over commonly available hardware and with convenient interfacing to industry standard products. The merging of this standard and a time slotted protocol over narrow band communication links, has not been done previously and offers wide application.

The next step was to utilize this slotted protocol to achieve a greater improvement by developing an adaptive polling system and to compare the relevant performance with the non-adaptive polling system.

To achieve the goal of the project, interfacing of devices, framing, serial data transmission, error detection, error control, and flow control techniques as well as contention and static

protocols had to be well understood and implemented where necessary.

### 1.3 Application Area

With the use of narrow band wireless communication networks, a base station can be configured so that it will communicate with more than one remote station over a radio network. The use of radio communications helps to monitor and control remote stations from the base station. The base station must be updated on what is going on at each station and react immediately in case of any abnormal event. Wireless systems can be implemented in an office, building or over long range to free the area from wiring.

On site sensors collect data such as current, voltage, temperature and electrolyte levels. This is transmitted to a monitoring station and processed using specially written software that can be customized to suit individual user requirements. The availability of two-way communication opens up the possibility of remote control, safely switching off a station, which is under performing or malfunctioning for a certain number of polls.

The complete unit interfaces seamlessly with proprietary control systems and can be programmed to transmit data at regular intervals or can be interrogated remotely at any time. Like mobile phones, the demand for telemetry continues to increase as companies and individuals recognize the potential and the time / money saving applications in which telemetry can assist them in every day life. Such applications are so far reaching that they are literally limited only by an individuals imagination, for if you can monitor something, you can manage it using “wire-free” telemetry.

For example, if you owned a greenhouse full of tropical plants, or an outdoor pool with valuable KOI Carp that required a certain environment, instead of constantly checking, temperature, water levels or humidity levels and worrying about problems that may occur during the night, installing a telemetry system would be a perfect solution. By linking the unit into temperature or humidity sensors and using the software provided to program the desired thresholds into the unit, the slave can act as a remote monitoring device that would generate an instant alarm out to a PC or by a SMS text message to a mobile phone if any thresholds were exceeded, alerting the owner or manager to potential problems.

In a similar fashion wireless communication systems can be used to monitor other occurrences like how many times doors are opened, or logging analogue quantities such as flow

rates etc, storing information in its memory. This information could then be accessed and downloaded at required intervals giving valuable process data. This would be particularly important if for example a temperature threshold alarm was triggered in say a food area, or medical facility. The user could then see if the temperature was changing rapidly or was remaining steady, allowing them to make an informed decision to rush to the premises or wait until the morning.

Consider a case where slave stations have different event generation rates at different times. For example, during working hours, all slaves connected to the master station generate events to send to the master station at the same rate. In this case round-robin polling (RRP) is suitable. During the night, however, only a few stations may always have something to send. In this case, an adaptive system can be useful to enhance system performance by allocating more time slots to stations with more data to send.

This thesis will show how improvements can be made to improve system performance real-time, without user intervention.

## 1.4 Thesis Outline

This chapter forms the first part of this thesis. The remainder of the document is organized as follows:

**Chapter 2: Related Research.** The chapter explains the OSI model followed by different models of typical data communications protocols. It also gives the basic theory on what to consider when one needs to build a communications system. Device interfacing, communication modes and communication tasks are all discussed in this chapter. The chapter concludes by discussing non-slotted, slotted time base protocols, adaptive and non-adaptive polling systems.

**Chapter 3: Error Management and Control.** In this chapter, different types of errors and their origin will be explained. Different error detection techniques will be analyzed in detail to show why some error detection schemes are preferred more than others. Since error detection alone is not enough to make a system reliable, error control techniques will also be explained followed by the flow control techniques.

**Chapter 4: Standard IEC 870-5-101 protocol.** This chapter explains the industrial

communication protocols. It also explains in detail the IEC 870-5-101 protocol, starting from frame type, communication mode, and different fields of a frame. The IEC 870-5-101 protocol needs special attention, because, in this thesis it has been implemented using time slots.

**Chapter 5: System Design and Design Considerations.** This chapter focuses on system design and specifications for a required performance.

**Chapter 6: Implementations.** This chapter will address some implementation issues, which are worth to be mentioned. These include CRC implementation and polling algorithms.

**Chapter 7: Results and System Reliability.** This chapter explains the results obtained and how they were obtained. Graphs are also included to validate the claim.

**Chapter 8: Conclusion and Recommendations.** This chapter presents the conclusions that were drawn from this study and it also considers if the goals of the project have been met. It also presents some recommendations for future work.

# Chapter 2

## Related Research

### 2.1 The OSI Model

The ISO-OSI reference model provides a basis for the development of standards for Open Systems Interconnection (OSI). This model devised by the International Organization for Standardization (ISO) is intended to ensure that information from systems made by various manufacturers, and having different architecture, can be exchanged and interpreted in accordance with standardized procedures.

This model arranges the communications functions in seven layers, each of which has a virtual connection to the appropriate layer of the communicating partner. Only on the lowest layer (Layer 1) there is a physical connection for exchanging signals. Each layer, with the exception of Layer 1, obtains the necessary service from the layer below it. The OSI model merely defines the servicing and functions of the layers, but not the technical realization (the protocols) within the layers.

According to [18], two user programs can exchange information on Layer 7, if there is agreement between them (i.e. there are protocols) on the following points :

- the representation of information in Layer 6
- the flow of communications (contents and form) in Layer 5
- the completeness of the information and the security of transport in Layer 4

- the way information should be transferred through the network in Layer 3
- the security of transmission in Layer 2
- the physical medium in Layer 1

7	Application Layer	Application Oriented Layers
6	Presentation Layer	
5	Session Layer	
4	Transport Layer	
3	Network Layer	Transport Oriented Layers
2	Data Link Layer	
1	Physical Layer	

Table 2.1: The seven layers of the OSI Model

The functions of the individual layers shown in Table 2.1 will now be explained in more detail:

### Physical Layer

The basic physical connection between the communicating partners takes place in this lowest layer. The mechanical and electrical coupling to the transmission medium is determined here, by specifying (among other things) the cable, the distances involved, the pinning of connectors, and the way the bits are represented.

### Data Link Layer

This layer is responsible for assuring that a reliably operating connection is made between two participants. For this purpose the protocol of this layer determines the methods for protecting transmissions, the telegram structure, methods of accessing the transmission medium and for the synchronization and addressing of participants.

### Network Layer

The network layer undertakes the choice and implementation of the best transmission route in a network between the communicating parties, and provides this service (routing) to the transport layer. This function is of particular significance when different networks need to be connected by means of gateways.

### **Transport Layer**

The transport layer represents the boundary between the application oriented layers 5 to 7, and the transport oriented layers 1 to 4. Its job includes guiding the information through the network, controlling the flow of information and the grouping into individual packets.

### **Session Layer**

The session layer provides procedures for the opening, the orderly progressing, and the termination of a communication “session”. In this is included also the control of the dialogue between systems: that is, the determination of their respective transmission prerogatives.

### **Presentation Layer**

The data of the application are converted in the presentation layer into a data format which the receiving application can interpret. This layer thus implements the matching of data formats and the conversion of codes.

### **Application Layer**

This top layer represents the interface between the open system and the user. It offers the user or his program a service allowing him to work easily with the system. Application programs, which need to be developed can thus access the functions of the open system via the protocol of the application layer.

In Figure 2.1, the route to be followed by data from the transmitting to the receiving application can be seen, indicated by the continuous arrows. At the transmitting side information (overhead) which is necessary for transmission and processing is added to the actual data in each layer; at the receiving side this information is removed again in the reverse order after processing.

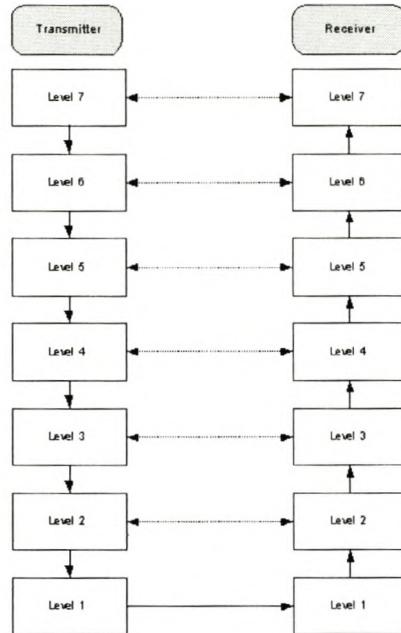


Figure 2.1: Data transmission in accordance with the OSI Model

## 2.2 A Data Communication Model

This section will show a model of communication which one needs to consider when designing a communication system[18]. Fig 2.2 shows a general block diagram of a typical communication model.

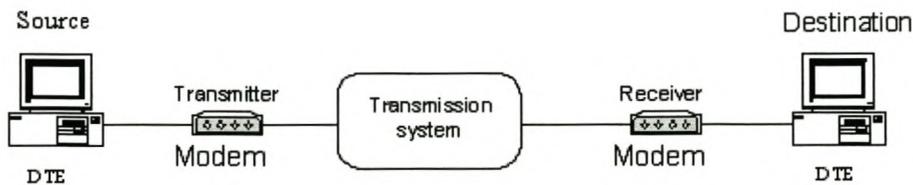


Figure 2.2: Basic data communication model

The key elements of the model are as follows:

- **Source.** This device generates data to be transmitted. A good example is a PC.
- **Transmitter.** Usually, data generated by the source device, which is frequently

part of the transmitter is not transmitted in the form it was generated. The modem transforms and encodes the information in such a way as to produce electromagnetic signals that can be transmitted across a transmission system.

- **Transmission system.** This is the link between the source and the destination. It can be a transmission line, radio, wireless or fibre optics.
- **Receiver.** The receiver accepts the signal from the transmission system and converts it into a form that can be recognized by the destination device.
- **Destination.** Takes the incoming data from the receiver.

The next section discussed various communication tasks that need to be taken into consideration to achieve communication process.

### 2.2.1 Data communication tasks

According to [18, 17], communication tasks are performed in a data communication system at different levels of the system. In this section, some of the key tasks will be discussed. To communicate, a device must **interface** with the transmission system. Once an interface is established, **signal generation** is required for communication. The properties of a signal, such as form and level must be such that the signal is:

- Capable of being propagated through the transmission system, and
- Interpretable as data at the receiver.

The transmission of a stream of bits from one device to another across a transmission link involves a great deal of cooperation and agreement between the two sides. One of the most fundamental requirements is **synchronization**. The receiver must know the rate at which bits are being received so that it can sample the line at regular intervals to determine the value of each received bit.

The **exchange management** plays a role when data has to be exchanged over a period of time. Both parties must cooperate. In all communication systems, there is a potential for errors, transmitted signal are distorted before reaching the destination. This leads to the requirements of **error detection and correction techniques**. It is simply not

acceptable for data to be accidentally altered in data processing systems. **Flow control** is required to make sure that the source does not send the data at the rate faster than it can be processed and absorbed. When more than one device share a transmission facility, a source system must assure that the destination system, and only that system receives the data. That's where **addressing** comes into play. **Message formatting** has to do with an agreement between two parties as to the form of the data to be exchanged. Each protocol uses its own message frame format.

Finally, it is important to provide some measure of **security** in a data communications system. The sender of the data may wish to be assured that only the intended receiver actually receives the data. And the receiver of data may wish to be assured that the data received is error-free and it comes from the intended sender. Some of these communication tasks will be discussed in the following chapters.

### 2.2.2 Data communication modes

There are three telecommunication modes namely: simplex, half duplex and full duplex which can be explained as follows:

- Simplex - Data flows in one direction only. An example can be a car radio. It can only receive radio signals (one directional transmission).
- Half-Duplex - Data flows in either direction, but in only one direction at a time. Example would be a CB radio (you can receive or talk but not both at the same time).
- Full-Duplex - Data flows in either direction at the same time. Think of it as a two-lane road. Cars can go in both directions at the same time. An example would be the telephone. Phone lines can carry signals in both directions.

Figure 2.3 shows the exchange of data between Device 1 and Device 2. Data flows in the direction of the arrow.

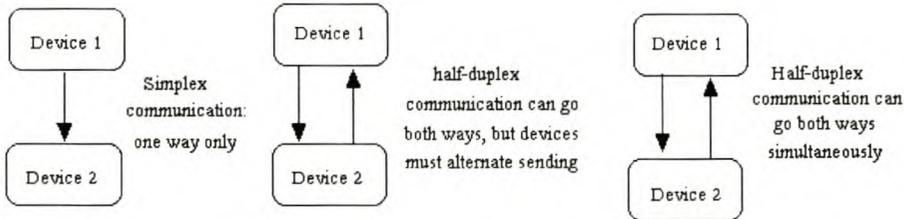


Figure 2.3: Simplex, Half-Duplex, and Full-Duplex communication

### 2.2.3 Timing of digital data communication system

In addition to providing the modulation and demodulation functions for analogue signals, digital communication also require a timing control. Telecommunications can only take place if the transmitting and receiving devices are timed correctly. Timing control is required to identify the rate at which bits are transmitted, and to identify the start and end of each bit. This permits the receiver to correctly identify each bit in the transmitted message. Data communications could be either synchronous or asynchronous.

#### 2.2.3 (a) Asynchronous transmission

**Basic Operation** - In asynchronous transmission, bits are sent in a small groups (usually a byte or character) independently. When no data is transmitted, the line will be in an idle state which is equivalent to binary 1. Each character begins with a start bit, binary 0, that alerts the receiver that a character is arriving and end with a stop bit, binary 1. The receiver samples each bit in the character and then looks for the beginning of the next character. This technique would not work well for long blocks of data because the receiver's clock might eventually drift out of sync with the transmitted signal. This type of transmission has found widespread application in both low- and high-speed applications.

**Character format** - A typical character consists of a start bit, data section, error control section, and one or more stop bits. Fig. 2.4 shows a waveform corresponding to a typical 7-bit character 'M'. In the figure, it can be seen that the even parity was used.

**Line sampling** - The clock rate of the receiving end must be the same as the sending

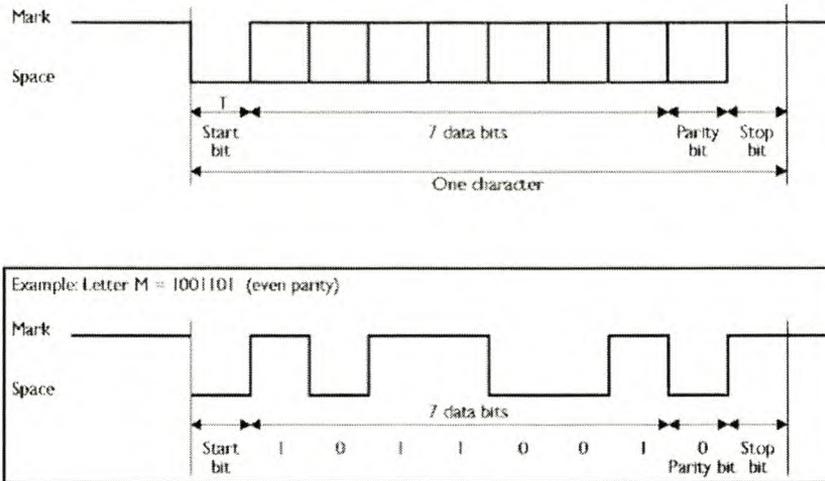


Figure 2.4: Asynchronous serial transmission of ASCII 'M'

end so that the receiving end will be able to recognize the data. If the sender is sending at 1200 baud, the receiver must sample the line fast enough to extract the information signal. Regardless of the type of transmission, the receiving end has to be set up for the same timing information included in the transmitted signal.

### 2.2.3 (b) Synchronous transmission

In synchronous communication, data are transmitted using timing devices. It used to be very popular for large data networks but, it is being replaced by asynchronous methods such as ATM. The accuracy of the message on the receiving end is dependent upon the sender and the receiver maintaining synchronization during decoding. The major issue with synchronous communication, is keeping the sending and the receiving end in synchronization. Extra bytes are added to allow the receiving end to synchronize with the sender when sampling the line does this. Communication within a computer is usually synchronous and governed by the microprocessor clock (CPU).

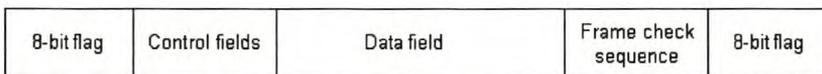


Figure 2.5: Synchronous Frame Format

**Operation**-Bits are never sent individually. They are grouped together in segments, called blocks. Figure 2.5 shows in general terms a typical frame format for synchronous

transmission. Typically the frame starts with a preamble called a flag, which is 8 bits long. The same flag is used as a postamble. The receiver looks for the occurrence of the flag pattern to signal the start of the frame. This is followed by some number of control fields, then the data field (variable length for most protocols), error control field, and finally the flag is repeated.

## 2.3 Device Interfacing

Interfacing is a very important aspect of data communications. In most cases, a PC's serial port uses a DB-9 while a modem's serial port uses a DB-25. To connect these two devices, a DB-9 to DB-25 adaptor cable is required. Some modem requires RJ-11 to DB-9 adaptor to interface with a PC. A PC-to-PC connection requires a DB-9 to DB-9 adaptor cable. Note that all these adaptors can be made using off the shelf hardware. When interfacing two or more devices that are far away from each other, a wireless connection is preferred. This section will explain wireless and physical interfacing.

### 2.3.1 Physical connection

There are two types of physical connection that are widely used in connecting devices namely: Serial and Parallel connections. In the next section, it will be shown why serial connections are widely used in fairly simple data transmission applications.

#### 2.3.1 (a) Serial vs Parallel connection

Let's consider the advantages and disadvantages of using the serial port when compared with parallel port.

The serial port has all the advantages of serial data transmission, i.e:

The serial port cable can be longer than a parallel port cable, as a typical serial port transmits '1' as voltage from -5 to -12V and '0' as voltage from +5 to +12 V, while the parallel port transmits '1' as voltage of 5 volts and '0' as voltage of 0 volts. At the same time the receiver of the serial port receives '1' as voltage from -3 to -25 V and '0' as voltage from +3 to +25 V. Thus serial port can have maximal swing up to 50 volts, while

parallel port has maximal swing of 5 volts. Thus the effect of losses in the cable when transmitting data using serial port are less substantial than losses when transmitting data using parallel port.

The number of wires needed when transmitting data serially is less than when the transmission is parallel. If the external device has to be installed at a great distance from the computer, a cable with three wires is much cheaper than a cable with 19 or 25 wires if the transmission is parallel. Still one should remember that there are interface creation expenses for every receiver/transmitter.

A further development of serial port communication is usage of infrared devices, which immediately proved popular. Many electronic diaries and palmtop computers have inbuilt infrared devices for connection with external devices.

Another example of serial port universality is micro-controllers. Many of them have built-in SCI (Serial Communications Interfaces), used for communication with other devices. In this case serial interface reduces the number of outputs pins on the chip. Usually only 2 outputs pins are used: Transmit Data (TXD) and Receive Data (RXD), compared with a minimum of 8 lines when using 8-bit parallel connection. Surely enough serial port communications has its drawbacks as well. The main one is, that when transmitting signal using serial connection, it is always necessary to convert the data into serial form, and vice versa.

### 2.3.1 (b) The RS232C interface

The serial port of the PC is compatible with the RS-232C standard. This standard was designed in the 1960s to communicate a DTE (the PC in this case) and a DCE (this in most cases being a modem).

The standard specifies 25 signal pins, but many of the pins are not needed and so in modern PCs, a DB-9 male connector is used. There should be one or more of these connectors in the rear panel of any modern PC. Figure 2.6 shows a DB-9 male connector. See Appendix A for more detail.

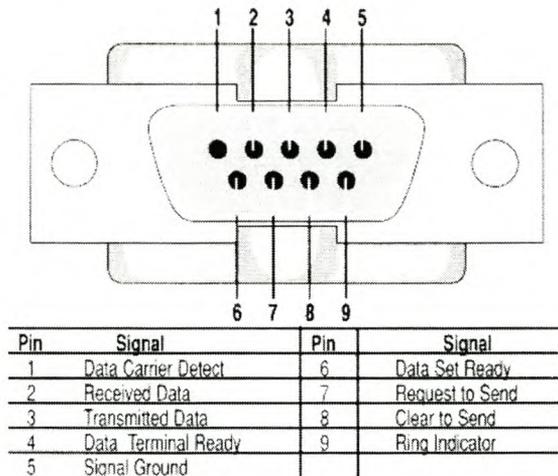


Figure 2.6: DB-9 male connector

### 2.3.2 Wireless connection

Wireless connection refers to a connection whereby more than one devices can communicate without the use of any physical connection. Wireless communication plays a major role in the world we are living in. Previously it was tough to communicate with someone who is out of town, but the daily improvement on wireless systems connect the world in terms of communication. Devices can be configured to communicate using infrared, microwave, or radio links.

#### 2.3.2 (a) Wireless data requirements

There are some aspects that need to be considered in the design of a wireless communication network. This section will discuss the requirements related to wireless data transmission.

##### Topology

The nature of wireless applications generally leads to a star configuration: a centrally located access point communicating with one (point-to-point) or more (Point-to-multi point) remote devices.

In determining the right topology for a wireless application, the following must be answered;

- Is the best configuration point-to-point or multi-point,
- Are peer-to-peer connections required,
- Will a polling scheme be used,
- Will remotes transmit data as needed, or
- Will the remote stations have to forward data from other units?

### **Speed and Latency**

Speed is important for straight-line racing, but the racing drivers know that many other factors such as handling and braking, are also critical. The same is true for wireless data communication. Speed is not the only issue, and faster is not always better. In fact, speed has trade offs with other important criteria; such as range and latency. The best design limits connection speed, according to data quantity and timing requirements.

Closely related to speed, latency measures the time required to transmit a data payload between devices. Latency and throughput are generally counterproductive; with decreased transmission overhead, longer packets increase throughput but increase latency when other devices have to delay as a result. This makes tolerance to latency an important application design factor.

### **Range**

Any wireless application must take into account how devices will be used and over what distances. How far apart will the devices be? Are they in line of sight or around obstacles? If needed, range and operability can be extended with multiple access points and repeaters.

### **Environment**

This is also an important aspect of wireless system design. The design must be suitable for the device's intended operating environment. Is the device going to be operated in a house, office or outside? The temperature range is also important.

## 2.4 Overview of Time Slotted and Non-slotted Protocols

### 2.4.1 Introduction

Time slotted protocol refers to protocols that transmits data within specific time slots. In these protocols, the time base is divided into slots with a size equal to frame transmission time plus propagation time between two stations. When a time slot protocol is implemented, data is only sent within the slot boundary. When using non-slotted protocols, data can be sent at any time after being generated. We will analyze the performance of this systems under both high- and low loading. A good example of a slotted protocol is slotted ALOHA and pure ALOHA for non-slotted protocols.

### 2.4.2 Aloha protocol

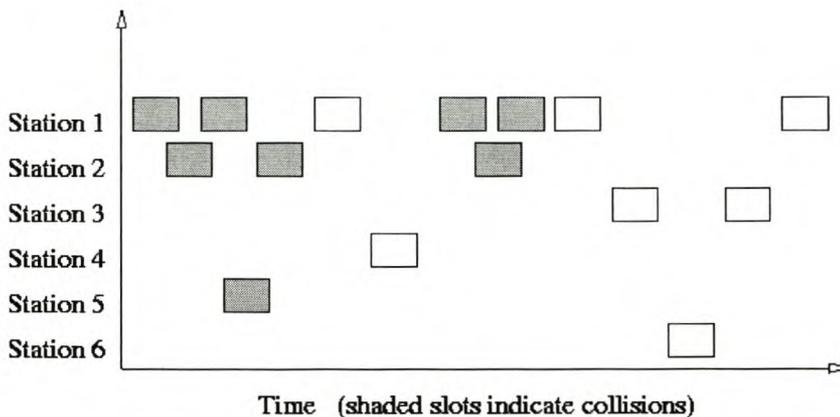


Figure 2.7: Pure Aloha protocol

This is the earliest technique of them all and was initially developed for packet radio networks[18]. However, it is applicable for any shared transmission medium. It is sometimes called Pure ALOHA. The operation is as follows:

The slave station can send at any time as shown in Figure 2.7. The station then listens for an amount of time equal to the maximum possible round-trip propagation delay on the network (twice the time it takes to send a frame between two mostly wide separated stations) plus a small fixed increment. If the station hears an acknowledgment during that time, fine; otherwise it resends the frame. If the station fails to receive an acknowledge

after a predetermined number of times, It gives up. A receiving station determines the correctness of frame by examining the frame check sequence (FCS) field as in the IEC 870-5-101 protocol discussed in the next chapter. If the frame is valid and the destination frame matches the receiver’s address, the station immediately sends an acknowledgment. The frame may be invalid due to noise in the channel, or another station might transmit at the about the same time. If a received frame is determined to be invalid, the receiving station simply ignores the frame.

This protocol is OK under low load but will start to degrade as the load increases (More stations wishing to send more and more data) which causes further collisions. Maximum throughput can be shown to be only 18% of attempted transmission[18][17]. See Fig. 2.9

### 2.4.3 Slotted aloha protocol

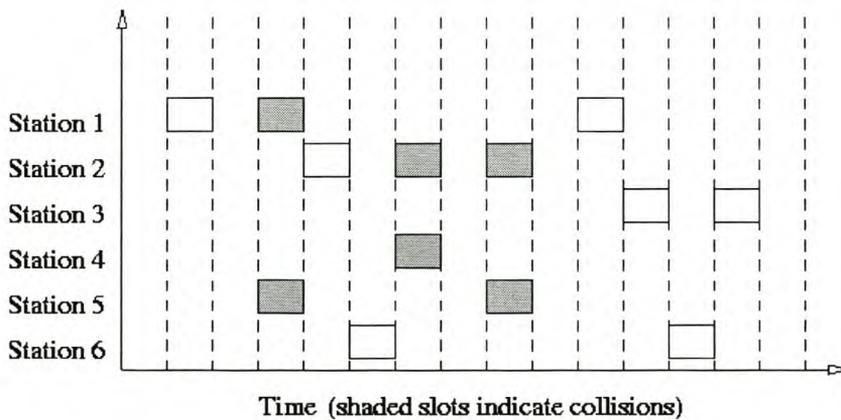


Figure 2.8: Slotted Aloha protocol

This protocol was developed to improve the channel efficiency of the system. It is a modification of Pure ALOHA protocol. It operates as follows:

Time on the channel is organized into uniform slots whose size equals the frame transmission time plus the ACK transmission time. Some form of synchronization is needed in order for each station to recognize the slot boundary. <sup>1</sup>Transmission is permitted to begin only at a slot boundary. This will avoid collisions that normally happen in pure ALOHA when any station can transmit while one frame is in transit. However, it does not avoid collisions that will happen when the waiting stations transmit at the start of the

<sup>1</sup>Stations that will be contending for the channel must be synchronized with the receiving station so that the slot boundary will be the same to all of the stations in the network.

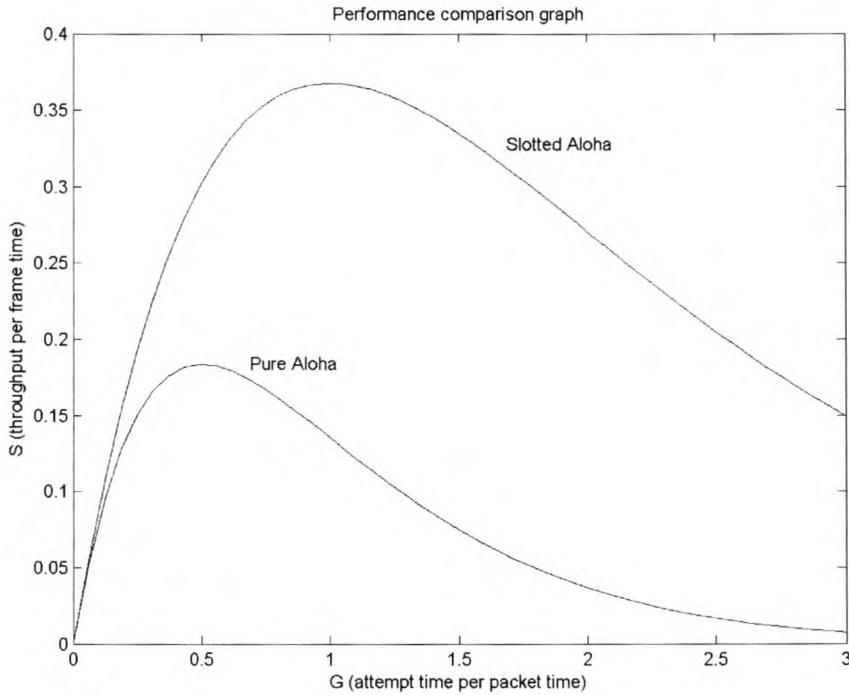


Figure 2.9: Comparison of performance between pure ALOHA and slotted ALOHA

next slot. Thus, frames that do overlap will do so totally. Figure 2.8 shows how collisions occur in slotted ALOHA protocol. According to [7] and [1], this increases the maximum utilization of the system to about 37% of attempted transmission as shown in Fig. 2.9. However, a small increase in load can drastically decrease performance.

By making a small restriction in the transmission freedom of the individual stations, the throughput of the ALOHA protocol can be doubled. Assuming constant length packets, transmission time is broken into slots equivalent to the transmission time of a single packet. Stations are only allowed to transmit at slot boundaries. When packets collide they will overlap completely instead of partially. This has the effect of doubling the efficiency of the ALOHA protocol.

#### 2.4.4 Carrier Sense Multiple Access (CSMA) protocols

With CSMA, a station wishing to transmit, first listens to the channel to determine if another transmission is in progress (carrier sense). If the medium is in use, the station must wait. If the medium is idle, the station may transmit. It may happen that two or

more stations attempt to transmit at about the same time. If this happens, there will be collision; the data from both transmissions will be garbled and not received correctly. To account for this, a station waits a reasonable amount of time for receipt of an acknowledgment, taking into account the maximum round-trip propagation delay and the fact that the acknowledging station must also contend for the channel to respond. If there is no acknowledgment, the station assumes that a collision has occurred and retransmits.

This strategy is effective for networks where the average frame transmission time is much longer than the propagation delay. Collisions can occur only when more than one user begins transmitting within a short time (the period of the propagation delay). If the station begins to transmit a frame and there is no collision during the time it takes the leading edge of the packet to propagate to the furthest station, then there will be no collision for this frame because all other stations are now aware of the transmission. Let's have a look at some approaches taken by the CSMA system when a medium is found busy.

#### 2.4.4 (a) Nonpersistent CSMA

In nonpersistent CSMA, a station wishing to transmit listens to the channel and obeys the following rules:

1. If the medium is idle, transmit; otherwise, go to step 2.
2. If the medium is busy, wait an amount of time drawn from a probability distribution (the transmission delay) and repeat step 1.

The use of random delays reduces probability of collisions. If two or more stations delay after collision is the same before trying again, they will both attempt to transmit at about the same time. Capacity will be wasted because the medium will remain idle following the end of a transmission, even if there are one or more stations waiting to transmit.

#### 2.4.4 (b) 1-persistent CSMA

To avoid idle time, the 1-persistent protocol can be used. A station wishing to transmit, listens to the medium and obey the following rules:

1. If the medium is idle, transmit; otherwise, go to step 2.
2. If the medium is busy, continue to listen until the channel is sensed idle; then transmit immediately.

#### 2.4.4 (c) p-persistent CSMA

This method is a compromise that attempts to reduce collisions, like nonpersistent, and reduce idle time like 1-persistent. The rules are as follows:

1. If the medium is idle, transmit with a probability  $p$ , and delay one time unit with probability  $(1 - p)$ . The time unit is typically equal to the maximum propagation delay.
2. If the medium is busy, continue to listen until the channel is idle and repeat step 1.
3. If transmission is delayed one time unit, repeat step 1.

Figure 2.10 shows throughput comparison between the ALOHA and CSMA protocols. It can be seen that the slotted protocol yields high throughput compared to the non-slotted protocols. These results encourage the use of time slotted protocols when one needs to design a telemetry systems.

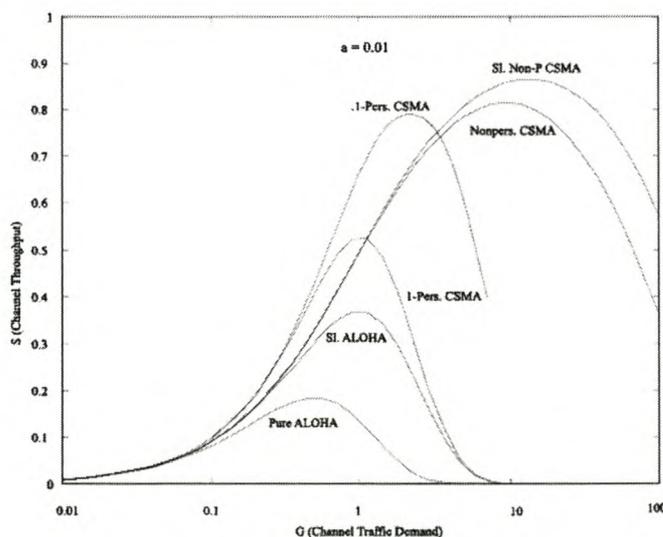


Figure 2.10: Throughput for various contention modes

## 2.5 Data Transmission Through Polling

Polling methods are implemented in a variety of different formats, of which a few will now be briefly discussed.

### 2.5.1 Round-Robin polling

In round-robin fashion, a master station polls the slave stations in a cyclical order. That is, the master station first polls the station with an address 1, followed by address 2 and so on. Each station is given a fair share to transmit data. The master station initiates the polling of data by sending the poll command to the first slave, then the slave will respond with a data frame if present. Upon receiving a poll message while the slave has no data to send, it replies with a NACK frame. In case where the master station detects an error on the received bits, the same station will be polled up to maybe three times.<sup>2</sup> If no success, it skips the station and poll the next station.

The total round-robin per station poll time, in an error free environment, can be stated as:

$$t_{ps} = t_{pr} + t_d \quad (2.1)$$

where  $t_{ps}$  = Total poll time/station

$t_{pr}$  = Poll request time/station

$t_d$  = Data transmit time/station

and:

$t_{pre}$  = Transmission pre-amble time

$t_{poa}$  = Transmission post-amble time

$t_b$  = Bit time

$t_r$  = Retry time delay

$n_h$  = No. of header bits

$n_{dmax}$  = Max no. of header bits

$n_{dmin}$  = Min no. of header bits

---

<sup>2</sup>If the master station does not receive the frame it requested and times out, it assumes that the frame is corrupted and poll the same slave station again.

$n_{dmax}$  = Min no. of data bits

$n_p$  = No.of poll request bits, additional to the header (if implemented)

Equation 2.1 can subsequently be written as:

$$t_{ps} = 2t_{pre} + 2t_{poa} + t_b(n_h + n_b) + t_b[n_h + (n_{dmax} - n_{dmin})/2] \quad (2.2)$$

The maximum cycle time for N stations is simply

$$t_{cyc-max} = N \cdot t_{ps} \quad (2.3)$$

When designing a system, we expect good performance all round. The drawback of using RRP is that it can happen that at some stage, some slave stations may experience a reduced event arrival rate. In that case, system's performance will drop and some of the slave stations will be delayed while the master polls slave stations with no data to send and the system will do nothing about it. In some cases, one will need to design a system with slave stations which has different event generation characteristics. This leads to the development of priority polling and similar schemes.

## 2.5.2 Priority polling

This scheme is used if the slave station's event generation rates differ noticeably and are known. A station, which generates events more often, is considered as a station of high priority and is allocated more channel bandwidth than other slaves. The polling pattern might either be 8:4:2:1, 3:1:1:1, or straight RRP, depending on the pattern of event generation. Stations can also be regarded as of higher priority depending on the information it supplies. <sup>3</sup>

If slave station 1 generates events more often than stations 2, 3 and 4, but station 2 also generates events more than station 3 and 4, then the designer must use a polling pattern which will give high throughput performance per station. In this case, the polling pattern might be 1→2→1→3→1→2→1→4. This will result in good system performance

---

<sup>3</sup>Note that polling pattern can be any format but here the focus are on those patterns that were implemented

but it also has some shortcomings. If the event generation pattern on some of the slaves changes dramatically, it is obvious that the system performance will drop until something is done in the system or until all slaves generate events as expected. This has led to the development of an adaptive polling system as part of the purpose of this thesis.

## 2.6 Contention Protocols versus Fixed Assignment Protocols

In Fixed Assignment Protocols, each slave station is allocated a time slot or frequency band to forward its data. Examples of this are Time Division Multiple Access (TDMA) and Frequency Division Multiple Access (FDMA). In contention systems, the master station does not poll slave stations for data communications, but each slave contend for the channel to forward its data. Good examples of contention protocols are ALOHA and CSMA protocols. The slave stations contend for the channel for data transmission.

Straightforward contention protocols do not support priority traffic and they have high variance in transmission delays[7]. At high loading, contention protocols suffer from delays because of high collision rate which is not the case with fixed assignment protocols. In lightly loaded networks, contention protocols provides good bandwidth utilization but with wide latency variations. Fixed assignment protocols provide fixed bandwidth, or throughput, per slave and therefore near constant latency. Bandwidth utilization under low loading, is not optimal however.

When multiple slaves frequently transmit data of similar amount, RRP can provide optimum bandwidth utilization. Because every slave is assigned a time slot and every remote transmit frequently, there are no collision and few unused slots. Priority and other variations of scheduled polling could improve channel utilization of fixed assignment protocols, when slaves transmit data at different rates (or when some slaves transmit more than others).

For the reason specified above, fixed assignment protocols were implemented because they can better support priority traffic over low speed links, which is one of the objectives of the project. It was clear that improvement of RRP should be investigated to lower transmission delay and variance.

## 2.7 Summary

After investigating different protocols, slotted and non-slotted, it was found that slotted protocols produce throughput which is twice that of a non-slotted protocols, particularly for contention based schemes. These results are well known and proved in different applications. This thesis will discuss the use of time slotted based protocols in round robin and adaptive polling techniques to show that significant improvement to the normal round-robin system is possible by modification of the scheme.

## Chapter 3

# Error Management and Control

### 3.1 Introduction

Today's data is generally reliably transferred. This is achieved by, amongst others, using error detection and control techniques. There are different communications media with different quality of service (QoS) characteristics. Consequently, there are several error detection and control schemes for different signal conditions. Some fundamental methods existed for decades. In addition, there are also new methods that can take advantage of particular signal format. These methods can be used to ensure reliable and adequate communications. This chapter presents an overview of error detection and the methods used for preventing errors that were used in this project.

### 3.2 Sources of Error

No errors can occur in the ideal transmission medium. However, none of the transmission medium is ideal [18, 17]. The signal representing the data is always subject to various error sources. As this signal propagates along the transmission media its amplitude decreases. This phenomenon is called signal attenuation. The signal cannot be detected if it is too weak. In addition, as the length of the medium increases, the waveform also changes during transmission. This phenomenon is called delay distortion and a signal cannot be recognized if it is too distorted. Furthermore, the transmission media can also be subject to interference resulting from other cables or signals caused by the electromagnetic radiation. The transmission process will be subject to thermal noise, proportionate to the

transmission bandwidth. All transmission errors increase as the length of the transmission medium increases.[17]

### 3.2.1 Types of errors

There are two general types of errors that can occur in digital data transmission, namely: random single-bit errors and burst errors

#### 3.2.1 (a) Random single-bit errors

Random single-bit errors occurs when a single bit is altered between the transmission and the reception; that is, a binary 0 is transmitted and the binary 1 is received or vice versa. This is an isolated error condition that alters one bit but does not affect nearby bits. It can occur in the presence of white noise, when a slight random deterioration of the signal-to-noise ratio is sufficient to confuse the receiver's decision of a single bit.

#### 3.2.1 (b) Error bursts

In practice, data communication systems are designed so that the transmission errors are within acceptable rate. Under normal circumstances, there are only few errors. However, it is possible that the signal conditions can sometimes be so weak that a signal cannot be received at all. It is also possible that sometimes the interference signal is stronger than the signal to be transmitted. Consequently, the data sent during the break is lost.

The contiguous block of data corrupted by the error signal are called **error bursts**. The length and frequency of the error bursts depend on the quality of the data link, which in turn depends on the transmission medium and signal conditions. Therefore, the error detection and control should be able to handle as many errors as possible. However, the particular error detection and control schemes will be dictated by the application. All applications will be subject to the efficiency of the scheme to be used. In addition, some applications may require short codeword lengths and low latency, whereas some of the applications might require extreme low error rates, for example.

## 3.3 Error Detection

### 3.3.1 Introduction

Error detection implies a method that allows some communication errors to be detected. The data is encoded by the sender so that the encoded data contains additional redundant information about the data. The data is decoded by the receiver so that the additional redundant information is used to match the original information to be received. This allow some errors to be detected. Unfortunately, some error bursts may cause incorrectly received blocks to pass the error detection test, if the latter is not sufficient.

An error detection technique must be chosen which suits the application it is being used for and so that the probability of such errors is acceptably small. In general, the more sophisticated the error detection technique, the more computation is needed to compute and transmit a large number of error detection and correction bits. The advantage is that the more sophisticated error detection techniques have a smaller probability of allowing undetected bit errors.

### 3.3.2 What motivates the use of error detection techniques?

Let's define these probabilities with respect to errors in transmitted frames:

- $P_b$ : Probability that a bit is received in error; also known as the bit error rate (BER);
- $P_1$ : Probability that a frame arrives with no bit errors;
- $P_2$ : Probability that, with an error-detection algorithm in use, a frame arrives with one or more undetected bit errors;
- $P_3$ : Probability that, with an error-detection algorithm in use, a frame arrives with one or more detected bit errors but no undetected bit errors.

First consider the case in which no means are taken to detect errors. The probability of detected errors ( $P_3$ ) is then zero. To express the remaining probabilities, assume that the probability that any bit is in error ( $P_b$ ) is constant and independent for each bit. Then we have

$$P_1 = (1 - P_b)^F \quad (3.1)$$

$$P_2 = 1 - P_1 \quad (3.2)$$

where  $F$  is the number of bits per frame. The probability that a frame will arrive with no bit errors decreases when the probability of a single bit error increases. Also, the probability that a frame arrives with no bit errors decreases with increasing frame length; the longer the frame, the more bits it has and the higher the probability that one of these bits is in error. This is the kind of result, that motivates the use of error detection techniques.

### 3.3.3 Parity checking

Parity checking is a primitive character-based error detection method. The characters are encoded so that an additional bit is added to each character. The resulting number is even or odd. The extra bit is set according to this result and according to parity setting, either even or odd, is being used.

If even parity is used, the extra bit is always set so that the codeword always contains an even number of bit set. In odd parity, the number of bits set in the codeword is always odd. The decoding is done simply by checking the codeword and removing the extra bit. The parity checking will only detect one bit error in each codeword. Parity checking has been used in character-based terminals but it is not suitable for today's reliable communications as shown in Table 3.1. However it is being used in memory chips to ensure correct operation.

Transmitted	Received	Results for odd parity
01010111	01010111	Odd number of 1s: Parity OK
01010111	01010101	Single-bit error: Even number of 1s: Error detected
01010111	01000101	Double-bit error: Odd number of 1s: Error not detected
01010111	01111101	Triple-bit error: Even number of 1s: Error detected

Table 3.1: Parity check operation

### 3.3.4 Block check

Block check is a block-based error detection method. The data is divided into blocks in the encoding process. An additional block check is added to each block of data. The check is calculated from the current block. The receiver also performs the same calculation on the block and compares the calculated result with the received result. If these checks are equal, the blocks are likely to be valid. Unfortunately, the problem with all block checks is that the block check is shorter than the block. Therefore there are several different blocks that all have the same checksum. It is possible that the data can be corrupted by a random error burst that modifies the contents so that the block check in the corrupted frame also matches the corrupted data. In this case, an error is not detected. Even the best block checks cannot detect all error bursts but they can minimize this probability. However, the reliability increases as the length of the block check increases.

#### 3.3.4 (a) Checksum

Checksum is a primitive block checksum that is the sum of all characters in the block. It uses a modulo 2 arithmetic, i.e binary addition with no carries, which is just the exclusive-OR (XOR) operation[18]. Unfortunately, it may allow relatively simple errors. In other words, it is easy to find different blocks that generate the same block checksum. Calculating a checksum is certainly fast and easy but the reliability of the checksum is not adequate for today's requirement for reliable communications. However, due to its speed, it is used in some applications, which require that the calculation is done by the software.

#### 3.3.4 (b) Cyclic Redundancy Check (CRC)

The CRC is an intelligent alternative for block checksum. It provides stronger error detection than simple checksum[7]. The CRC method operates on blocks of data called frames. Basically the sender appends a bit sequence to every frame, called the FCS (Frame Check Sequence). The resulting frame is exactly divisible by a predetermined number which is one bit longer than the CRC. Even though CRC method requires slightly more processing than the block checksum, it is widely used in today's reliable communication[2][18]. CRC can be easily implemented by using shift registers and xor-operations both in hardware and software implementations[7]. There are two options in which the receiver can validate

a received frame.

**Option 1:** When the receiver receive a frame, it separates the message (string of bits) and the checksum, and calculate the checksum of the message. If the result of a checksum equal to the checksum appended by the sender, then the frame transmission will be interpreted as successful. If not, the sender can be notified to resend the block of data.

**Option 2:** When the frame is received by the receiving entity, checksum is performed at the message bits and the received CRC. If the remainder is zero, then the data has been sent successfully. If not, the sender can be notified to resend the block of data.

*Note that the receiving entity can decide to either use option 1 or option 2 without even consulting the sender but the CRC used must be the same for both the sender and the receiver.*

A 16-bit cyclic redundancy code detects all single and double-bit errors, all errors with an odd number of bits, all bursts errors of length 16 or less, 99.997% of 17-bit error bursts, and ensures that 99.998% of all possible errors[7]. This level of detection assurance is considered sufficient for data transmission blocks of 4 kilobytes or less.

## 3.4 Flow Control Techniques

Flow control is a technique for assuring that a transmitting entity does not overwhelm a receiving entity with data. The receiving entity typically allocates a data buffer of some maximum length for a transfer. When data are received, the receiver must do a lot of processing before passing the data to the high-level software. In the absence of flow control data may fill up and overflow while the receiver is still busy processing the old data.

### 3.4.1 Stop-and-wait flow control

The idea of stop-and-wait is straightforward. After transmitting one frame, the sender waits for an acknowledgment before transmitting the next frame. If acknowledgment does not arrive after a certain period of time, the sender times out and retransmits the original frame.

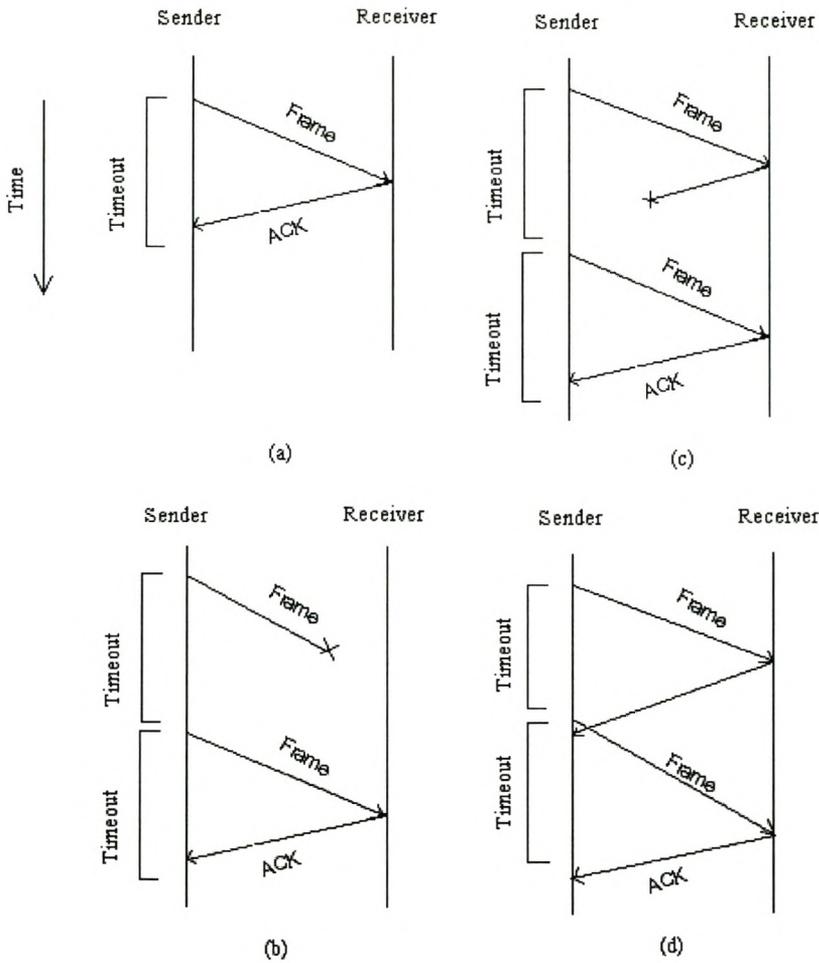


Figure 3.1: Four different scenarios for the stop-and-wait algorithm. (a) The ACK is received before the time expires; (b) The original frame is lost; (c) the ACK is lost; (d) The timeout is too short.

Figure 3.1 illustrate four different scenarios that result from this basic algorithm. This figure is a time line, a common way to depict a protocol's behavior and the time flows from the top to the bottom. Figure 3.1(a) shows the situation in which the ACK is received before the timer expires, (b) and (c) shows the situation in which the original frame and the ACK, respectively are lost, and (d) shows the situation in which the timeout fires too soon. Recall that by “lost” we mean that the frame was corrupted while in transit and this corruption was detected by an error code on the receiver, and the frame was discarded.

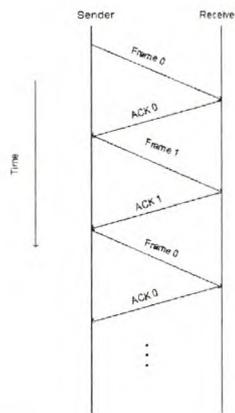


Figure 3.2: Stop-and-wait ARQ with 1-bit sequence number

There is one important aspect in stop-and-wait algorithm. Suppose that the sender sends a frame and the receiver acknowledges it, but the ACK is either lost or delayed in arriving. This situation is illustrated in time lines (c) and (d) of Figure 3.1. In both cases the sender times out and retransmits the original frame, but the receiver will think it is the next frame, since it is correctly received and acknowledge the previous frame. This will cause a receiver to have a duplicated frame. To avoid this problem, the stop-and-wait scheme must add a 1-bit sequence number; that is, the sequence number can either take the value 1 or 0— and the sequence number for each frame must alternate, as illustrated in Fig. 3.4.1. The receiver is then able to determine if the frame has been sent before. In case it obtain the duplicate frame, the receiver will discard the frame and send an ACK to the sender.

### 3.4.2 Sliding window flow control

In stop-and-wait flow control, frames are not sent one at a time. The receiver allocates buffer space for  $W$  frames. Thus, the receiver can accept  $W$  frames, and the sender is allowed to send  $W$  frames without waiting for an acknowledgment. To keep track of which frames have been acknowledged, each frame is labeled with a sequence number. The receiver acknowledges the frames by using an ACK that contains a sequence number of the next frame expected. This ACK also announces that the receiver is prepared to receive the next  $W$  frames, beginning with the number specified.

In case of an error in one of  $W$  frames, the receiver will acknowledge the correct ones only and the sender will decide either to send the frames that is not acknowledged or it will resend all the frame starting from the corrupted one. This makes this scheme less efficient if errors are always occurring. In a noisy channel, frames will always be retransmitted and this will have a negative impact because the longer the transmission, the more likely that there will be an error, necessitating retransmission of the entire frame.<sup>1</sup>

## 3.5 Error Control Techniques

### 3.5.1 Introduction

Error detection techniques are useless if no action has been taken after an error has been detected. Error control coding provides the means to protect data from errors. Data transferred from one place to the other has to be transferred reliably. Unfortunately, in many cases the physical link can not guarantee that all bits will be transferred without errors. It is then the responsibility of the error control algorithm to detect those errors and in some cases correct them so upper layers will see an error free link. Once an error has been detected, the system must take action to correct it, or inform the sender about the error so that the frame can be sent again (through some sort of error handling interface).

Data are sent as a sequence of frames; frames arrive in the same order in which they are sent; and each transmitted frame suffers an arbitrary and variable delay before reception. In addition there are possibility of two types of error:

---

<sup>1</sup>This reason leads to the use of stop-and-wait flow control in the project. In stop-and-wait flow control, frames were small and independent. This was done because the error in small frames are detected and eliminated sooner.

**Lost frame:** A frame fails to reach the destination. For example, a noise burst can damage a frame to an extent that the receiver is not aware that a frame has been transmitted.

**Damaged frame:** A recognizable frame does arrive, but some of the bits are in error (have been altered during transmission).

There are two basic types of error control which are backward error control and forward error control. These two error control strategies have been popular in practice. In this thesis, the focus will be on backward error control but forward error correction techniques worth to be mentioned.

### 3.5.2 Backward error control

This is the technique which uses error detection combined with retransmission of corrupted data. This strategy is generally preferred for several reasons. The main reason is that the number of overhead bits needed to implement an error detection scheme is much less than the number of bits needed to correct the same error, especially in a system where it is rare to find errors. Backward error control techniques are based on the following mechanism:

- **Error detection:** As discussed in Section 3.3
- **Positive acknowledgment:** The destination returns a positive acknowledge to successfully received, error free frames.
- **Retransmission after timeout:** The source retransmits a frame that has not been acknowledged after predetermined amount of time.
- **Negative acknowledgment and retransmission:** The destination returns a negative acknowledgment to frames in which error is detected. The source retransmit such frames.

Collectively these mechanism are all referred to as Automatic Repeat Request (ARQ); the effect of ARQ is to turn an unreliable data link into a reliable one. There are three versions of ARQ:

- Stop-and-wait ARQ

- Go-back-N ARQ
- Selective-reject ARQ

All these versions are based on the use of flow control techniques discussed in the Section 3.4. Stop-and-wait ARQ is based on the stop-and-wait flow control technique whereas Go-back-N and Selective-reject are based on sliding window flow control.

### 3.5.3 Forward Error Correction

This technique does not require retransmission of frames. When the FEC strategy is used, the transmitter sends redundant information along with the original bits and the receiver makes its best to find and correct errors.

The drawback is the overhead of additional data that are always sent, even when no error occurs. Depending on the level of FEC implemented, the overhead can approach 50% or more. An example of FEC technique, is the well known Hamming code.

## 3.6 Summary

This chapter gives an overview of different sources of error, types of error, error detection and control techniques as well as flow control techniques. Having understood these aspects, the idea was to choose techniques suitable to the particular application, in order to enhance system reliability, which is complementary to the goals of the project. Due to the throughput restriction of the channel, it was decided that the overhead imposed by FEC, would be too onerous and not justified.

## Chapter 4

# Standard IEC 870-5-101 protocol

### 4.1 Introduction

IEC 870-5-101 protocol is an industrial communication protocol which is defined as datagrams through which computer-based devices communicate with one another - the way they organize, and transmit the bits and bytes of electronic binary signals whose patterns encode data. A protocol is simply a set of rules that govern how the message containing data and control information are assembled at a source for their transmission across the network and then dissembled when they reach their destination.

IEC 870-5-101 is the data link layer protocol based on the International Standard IEC 870-5, which defines the transmission protocols for telecontrol equipment and systems. This chapter will discuss the IEC 870-5-101 standard derived from the above standard, but does not use all of the IEC functions. This protocol requires that there should be a master-slave structure, since slaves cannot communicate with each other in this definition.

### 4.2 Protocol Communication Modes

This protocol can be implemented in two different modes, namely: balanced and unbalanced mode. The focus will be mainly on the unbalanced mode but the balanced mode is worth to be mentioned.

### 4.2.1 Balanced mode

In this mode, each station may initiate message transfers. The balanced mode restrict procedures to “point-to-point” or “multiple point-to-point”.

### 4.2.2 Unbalanced mode

Unbalanced transmission procedures are used in supervisory control and data acquisition (SCADA) systems in which a master station controls the data traffic by polling outstations sequentially. In this case the master station (master) is the primary station that initiates all message transfers while outstations are secondary stations (slaves) that may transmit only when they receive a poll message.

## 4.3 Network Configurations

The following fixed network configurations are supported by the standard unbalanced mode:

- Point-to-point
- Multiple point-to-point
- Party line
- Redundant line

Figure 4.1 shows the network configurations mentioned above.

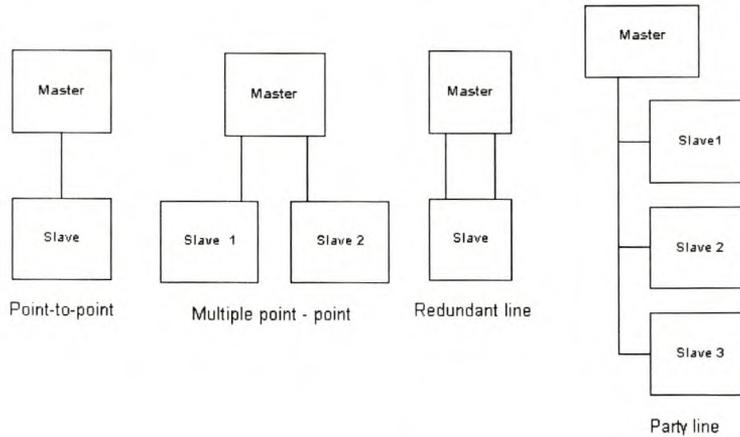


Figure 4.1: Network configurations supported by the IEC 870-5-101 protocol

## 4.4 Framing

IEC 870-5-101 standard is an asynchronous protocol with hamming distance of four and its character format is shown in Fig. 4.2. Framing is an important aspect which is used to encode data so that only the intended station will be able to decode it.

### 4.4.1 Transmission frame format

The IEC 870-5-101 protocol specifies frame format class FT 1.2 with three different frame formats for the transmission of remote control data, which can be recognized by means of special start characters as shown in Fig 4.2. These frames have specific tasks, which will be explained individually.

- **Single character:** This format consists of a single character, namely the E5h (decimal 229), and normally used to confirm user data on a point-to-point master-slave configuration but unsuitable when a master station is connected to more than one slave because it has no address.
- **Short frame:** This format with a fixed length begins with the start character 10h, and besides the C and A fields includes the checksum (this is made up from the two last mentioned characters), and the stop character 16h. Short frame is normally used for link layer services and in special cases, it can be used as a confirm receipt

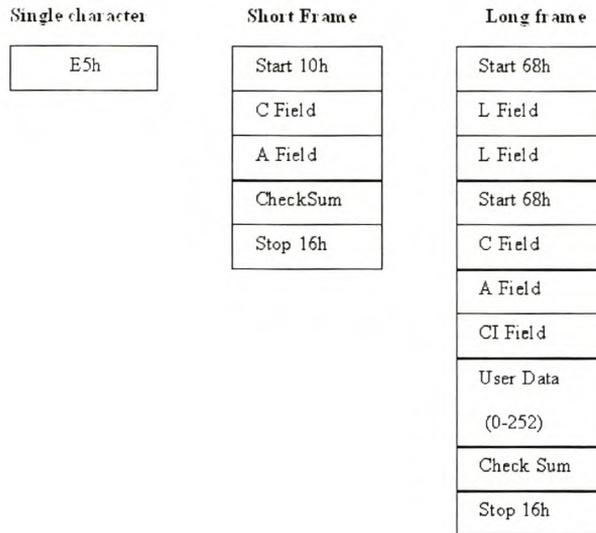


Figure 4.2: Frame formats defined by the standard IEC 870-5-101 protocol

of transmission instead of a single character.

- Long frame:** This format is used for data transmission of user data between a master and a slave. After the start character 68h, the length field (L field) is first transmitted twice, followed by the start character once again. After this, there follow the function field (C field), the address field (A field) and the control information field (CI field) follows. The L field gives the quantity of user data inputs plus 2 (for C and A fields)<sup>1</sup>. After the user data inputs, a checksum is transmitted, which is built up over the same area as the length field, and in conclusion, the stop character 16h is transmitted.

#### 4.4.2 Meaning of the fields

In this section, fields used for framing formats will be explained. Some need not to be explained since they are clear already. All fields have a length of 1 byte, corresponding to 8 bits.

<sup>1</sup>The CI field was not used in this project and it was omitted.

### C Field (Control Field, Function Field)

Besides labeling the functions and the actions caused by them, the C field specifies the direction of data flow, and is responsible for various additional tasks in both calling and replying direction. Table 4.1 shows the coding of the individual bits of the C field:

Bit Number	7	6	5	4	3	2	1	0
Calling Direction	Res	PRM	FCB	FCV	F3	F2	F1	F0
Replying Direction	Res	PRM	ACD	DFC	F3	F2	F1	F0

Table 4.1: Coding of C Field

The most significant bit (MSB), Res is reserved for future functions, and at present is allocated the value 0; bit number 6, PRM is used to specify the direction of data flow. A bit 1 represent a master station calling message while a bit 0 represent an slave reply. The frame count bit (FCB) indicates successful transmission, in order to avoid multiplication or loss. If the expected reply is missing, or reception is faulty, the master sends the same frame with an identical FCB, and the slave replies with frame as previously stated. The master indicates with a “1” in the FCV bit (frame count bit valid), that the FCB is used. When the FCV contains a “0”, the slave should ignore the FCB.

In the replying direction, the DFC (Data Flow Control) serves to control the flow of data. The slave with a DFC = 1 indicates that it can accept no further data. With an ACD bit (Access demand) with a value of 1, the slave shows that it wants to transmit Class 1 data, which has to be transmitted as soon as possible because it is of higher priority. The support of Class 1 data and the bits DFC and ADC is not required by the standard and won't be discussed further.

Function code	Frame type	Service function	FCV
0	Send/Confirm expected	Reset remote link	0
3	Send/Confirm expected	User data	1
4	Send/No reply expected	User data	0
9	Request/Response expected	Request status of link	0
11	Request/Response expected	Request user data class 2	1

Table 4.2: Function code of control fields in messages sent from master station: Unbalanced mode

Four bits; F0, F1, F2 and F3 of the control field is the function code. This means that the maximum number of function codes per station is  $2^4 = 16$ . Not all of these 16 function codes will be used. Table 4.2 shows the function code used by the master station to

achieve the goal of the project.<sup>2</sup>

Function code no	Frame type	Description
0	Confirm	ACK: Positive acknowledgment
1	Confirm	NACK: message not accepted, link busy
8	Respond	User data
9	Respond	NACK: Data not available
11	Respond	Status of link or access demand

Table 4.3: Function code of control fields in messages sent from slave station: unbalanced mode

### A Field (Address Field)

The address field serves to address outstation in the calling direction, and to identify the sender of information in the receiving end. The size of this field is one byte, and can therefore take the values 0 to 255. The addresses 1 to 255 can be allocated to individual slaves while the addresses 254 (FEh) and 255 (FFh) are used to transmit information to all participants (Broadcast). With address 255, none of the slaves reply and with address 254 all slaves reply with their own address. The latter case results in collision when more than one station are connected, and should only be used for test purposes. The address 253 (FDh) indicates that the addressing has been performed in the network layer. Addresses 251 and 252 are kept for future applications.

### User data

This field is only available on the long frame format. It is able to carry data of size 0 to 252.

### Checksum

The checksum serves to recognize transmission and synchronization faults, and is configured from specific parts of the long frame. These parts are mentioned when presenting the individual frame in Section 4.4.1. Checksum was used for a short frame but it was replaced by CRC-16 for increased data security.

---

<sup>2</sup>Note that both master station and the outstation has 16 function code with different service function.

## 4.5 Transmission Rules

1. Line idle is binary 1.
2. Each character has one start bit (binary = 0), 8 information bits, one even parity bit and one stop bit (binary = 1).
3. No line idle intervals are admitted between characters of a frame.
4. Upon detecting an error according to Rule 6, a minimum interval of 33 bits (3 characters) is required between frames.
5. The sequence of user data characters is terminated by a 8 bits checksum (CS). The checksum is the arithmetic sum over all user data octets.
6. The receiver checks:
  - the start bit, the stop bit and the even parity bit.
  - the start character, the length (2 bytes in frames with variable lengths), the frame checksum and the end character.

When a fault has been detected as a result of the above checks, the frame will not be accepted, and the reply or acknowledge will not be sent.

## 4.6 Communication Process

The data link layer uses two kinds of transmission services:

- Send/Confirm: SND/CON
- Request/ Respond: REQ/RSP

Before any of this service is performed, the master station must initialize and synchronize with all the slaves as discussed in the next section.

### 4.6.1 Initialization of slave station

The initialization of the master station starts e.g. with power on/off. Any data requested right before initialization cannot be received by the slave station because it is no longer available. The link of the master station then establishes connection with the link of the slave station by transmitting a “Request status of link” that is answered by the “Status of link” from the slave station. The master station then transmits a “Reset of remote link” that is answered by an “ACK”, which confirms that the station is ready for data transmission. The process occur as follows:

- Master → Slave: Request status of link
- Master ← Slave: Status of link
- Master → Slave: Reset of remote link
- Master ← Slave: ACK

If initialization fails, the master will again try to initiate the process mentioned above for a configured number of retries. If it fails after the configured number of retries, it report back to the master station that the link is currently unavailable. When the connection is established, the clock of two stations is then synchronized by a clock synchronization command. The synchronization processes will be explained in the next Section.

### 4.6.2 Synchronization

After initialization, clocks of slave stations are initially synchronized by the master station and then resynchronized periodically.

#### 4.6.2 (a) Synchronization process

The synchronization process occur as follows:

- Master → Slave: Sync frame
- Master ← Slave: ACK

After synchronization, the drift between the master and a slave will be tolerable and the slave will be ready for data transmission.

The master station sends a clock synchronization command containing the master station's clock immediately after the initialization process is complete. Stations can be synchronized one by one, or by using a broadcast synchronization message, which synchronizes all stations configured on the communications medium. Upon arrival of the sync frame, slave station immediately correct its clock according to the sum of the propagation delay, and frame transmission time, and the received master's clock, and send an ACK to master station. It must be noted that when this procedure is used for time synchronization the accuracy of the synchronization will always depend on the characteristics of the transmission line.

#### 4.6.2 (b) Why is synchronization so important?

The importance of synchronization depends on the protocol used. Slotted protocols require synchronization so as to prevent packet overlap and to organize station's media access. If time slots are not synchronized, it might happen that, by the time a master station polls a slave, it will be clearing its buffer. This will disrupt communications between two stations. Data will never get through to the intended stations if stations are not synchronized correctly.

#### 4.6.3 Request/respond procedure

The master requests data from the slave according to class 2. The slave can either transfer its data with RSP\_UD, or give no response indicating that the REQ\_UD2 frame has not been received correctly or that the address contained in the REQ\_UD2 frame does not match. This kind of procedure is known as polling. The operation is as follows:

- Master → Slave: REQ\_UD2
- Master ← Slave: SND\_UD

#### 4.6.4 Send/Confirm procedure

With this procedure, the master transfers user data to the slave. The slave can either confirm with a single character (E5h), or by omitting a confirmation signal that it did not receive the transmitted frame correctly. The operation is as follows:

- Master → Slave: SND\_UD
- Master ← Slave: ACK

### 4.7 Summary

This chapter discussed the implementation of the industrial IEC 870-5-101 protocol which has found widespread use in telecontrol and automation processes. This protocol is one of the communication protocols, which is used most in Europe. This protocol is suitable for this research, because it supports remote control and monitoring processes. It was then decided to implement a standard, which uses CRC-16 as an error detection technique, instead of a checksum. The change in error detection technique enables the IEC 870-5-101 protocol to be used over radio networks, which are normally more noise prone than cable based transmission media.

## Chapter 5

# System Design and Design Considerations

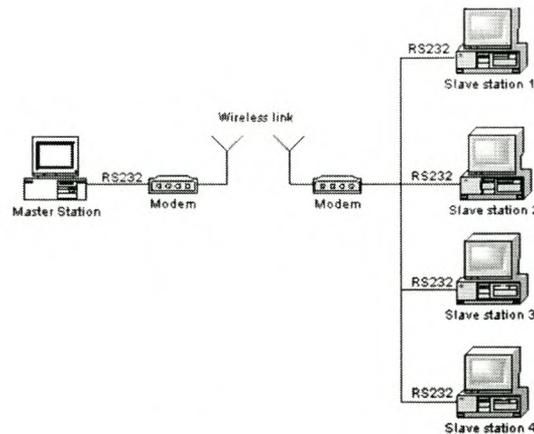


Figure 5.1: Designed system

In order to develop software for system simulation, it should be understood how the system is structured and the way it operates. The system was built as shown in Figure 5.1<sup>1</sup>. The channel is shared using a time division duplex (TDD) protocol, and managed by the master. The operation can be discussed in two ways: In the first case, the master station allocates the time slots to each of the slaves in a cyclic order and poll for data at the beginning of each slot. This case provides fairness among the stations.

In the second case, a master station allocate time slots dynamically, to allow stations with

---

<sup>1</sup>In real applications, slave stations can be placed at different places but in this case they were placed at one point for simulation purposes

more data to send more often. Stations which generate events more often get more slots depending on the priority of the information they provide.

EL705 OEM series radio modems were used as data transmitting and receiving device. The system was built and tested in the lab to investigate various schemes and to validate performance. Checksum, CRC and stop-and-wait mechanisms were implemented. The IEC 870-5-101 protocol was implemented on a time slotted base and the master station obtain data from slaves through polling. Stations were configured as follows:

## 5.1 Master Configuration

The Master station used was a PC which was then connected to a radio MODEM via RS-232 cable. The radio modem was then connected with an omni-directional antenna. If there is no repeater in between stations, RF signals will be transmitted and received at the same frequency, in half duplex mode.

### 5.1.1 Master model

The master station is designed as the center of all processes. The master station initiate and control all data communication processes.

The master station was modeled as follows:

- The station must monitor and control all the slave stations.
- The master station must synchronize all slaves before data transmission starts.
- Time is divided into discrete intervals (slots). Frame transmissions always begin at the start of a slot. A frame from the slave station may occupy more than 1 slot. The master allocates the time slots to each slave with respect to the protocol used.
- The master station polls the slave using a short frame of the IEC 870-5-101 protocol.
- The master station does not know the status of the queues at the slaves, because there are no provisions for exchange of such information in the IEC 870-5-101 packet structure.

- The master station must store the data it receives from the slave on a CSV file.
- The master station must calculate the mean wait time of frame by subtracting the event generation time (discussed in slave model) from the time it receives the frame.
- Master must be able to report immediately, to the user in case of an unexpected results.
- Propagation delay is small compared to the transmission time.

## 5.2 Slave Configurations

Four independent slave stations were connected to the modem using RS-232 cable as shown in Fig. 5.1. The modem antenna could be placed anywhere within a range covered by the master station antenna. This will obviously depend on the terrain.

### 5.2.1 Slave station model

The model was based on the following assumptions:

- The model consists of four independent stations, each with a program that generates frames for transmission. The probability of a frame being generated in an interval of length  $\Delta t$  is  $\lambda\Delta t$ , where  $\lambda$  is not a constant (the arrival rate of new frame).
- When an event is generated, a slave station attached the time tag to a frame which contain the time in which an event was generated.
- Once a frame has been generated, the station is blocked and does nothing until the frame has been successfully transmitted.
- The contents of an event generated are random characters and they are not of great importance in this case.
- A slave station may not transmit and receive simultaneously.
- Slave stations do not communicate with each other. That is, communication occurs in a master-slave configurations as supported by the IEC 870-5-101 protocol.

- All stations must be synchronized with the master station before polling for data starts.
- All slave stations receive a poll message from the master (point to multi-point configurations), but only the intended slave will respond with a long frame specified in IEC 870-5-101 protocol and other slaves discard the poll frame not directed to them to keep their buffers empty.

### 5.3 Radio Modem Configurations

The radio modem was modeled as follows:

- A radio modem and a DTE must be interfaced correctly to avoid data misinterpretation between the master and a slave. See Appendix A.1 for PC to modem interfacing.
- The modem's antenna of a receiving entity must be at a range covered by the antenna of the transmitting entity.
- Over-the-air propagation delay must be small compared to transmission time.
- Radio modems must transmit and receive data at the same frequency since there is no repeater between them. In this case, data was transmitted and received at a licensed narrow band radio frequency of 455 MHz.

### 5.4 Signal Processing Between a Master and Slave Stations

The master station sends an initialization command in a form of digital data bits through a radio modem which modulates the digital data bits to analog RF signal and transmits the RF signal to the slave through an antenna.

The slave receives the data bits in the form of an RF signal over the antenna and passes it through to the radio modem, which will demodulate them into digital data. On the data link layer, the digital data will be distributed to all the built-in slaves and the one

for which the data has been addressed to, will respond. If the message was a broadcast, all the slaves will be able to perform the required applications requested by the master station. The reverse process happens when the slave replies.

## 5.5 Summary

The system was designed to check whether the results foreseen are obtainable in practice. Slave stations were constructed on one PC only and operate independently to each other for simulation purposes, the reason being that slave stations do not communicate with each other. A master station would normally be communicating with slave stations in different locations. The system design and design considerations were in line with the goals of the project.

# Chapter 6

## Implementations

### 6.1 Software Platform

All system software was developed under the Delphi environment. Delphi is a powerful windows development tool, which allows you to produce self-contained windows applications quickly and effectively. Delphi contains a virtual development environment that generates the necessary code for the user interface. You can modify this code or add functions or procedures that will extend the effects of user's actions. All code is developed using Pascal programming language, which has been extended to handle the additional requirements of windows and Linux applications.

One big fact about Delphi is that you can really do it all “easily” reaching everything your machine and operating system has to offer.

The following are just a few Delphi's advantages

- It has a beautiful object model. Delphi fully supports polymorphism, encapsulation, and true inheritance.
- It has full support for interfaces.
- It produces small, fast executables.
- It has sophisticated, low level support for COM. COM development in Delphi is easy, but when you want to dig down into the hard stuff, it lets you open up the hood and get your hands dirty.

- Delphi has no black boxes. With a few minor exceptions, you get the source to the entire VCL. You can see exactly how Delphi does everything.
- Delphi is written in itself. We didn't have to resort to using some other language when creating Delphi. Delphi is truly a complete development solution.
- Delphi runs under both windows and Linux platforms.
- Object Pascal is an easy to use and strongly typed language. Delphi provides a solid syntactical framework on which you can build your applications.
- Delphi fully supports the operating systems on which it runs. If you can do something in Windows, then you can do it in Delphi.

For the reasons specified above Borland Delphi 5 was used as a simulation tool to determine the necessary results. The IEC 870-5-101 protocol was implemented and data were transmitted in the binary form, meaning ASCII characters were converted to binary strings before transmission and encoded with CRC or checksum depending on the frame type. At the receiving end, data was then be decoded and converted back to the ASCII characters. This requires Delphi functions, which can convert data from ASCII to integer, integer to binary string and vice versa. The coding for these functions is included in Appendix C

## 6.2 Implementation of Error Detection Algorithm

In this section, the steps taken to implement CRC-16 will be shown and presented in a flow chart. To see how parity check and Checksum where implemented, Appendix C can be consulted.

To calculate the CRC string, the message (address, function code and data without the start, stop and parity bits) is considered as a single continuous binary number which most significant bit (MSB) is sent first.

The step by step procedure for the CRC-16 calculation is as follows:

- 1) Load a 16-bit register with 0000h (all bits set to 0).
- 2) Execute the exclusive OR of the first character with the high order byte in the register and place the result in the register.

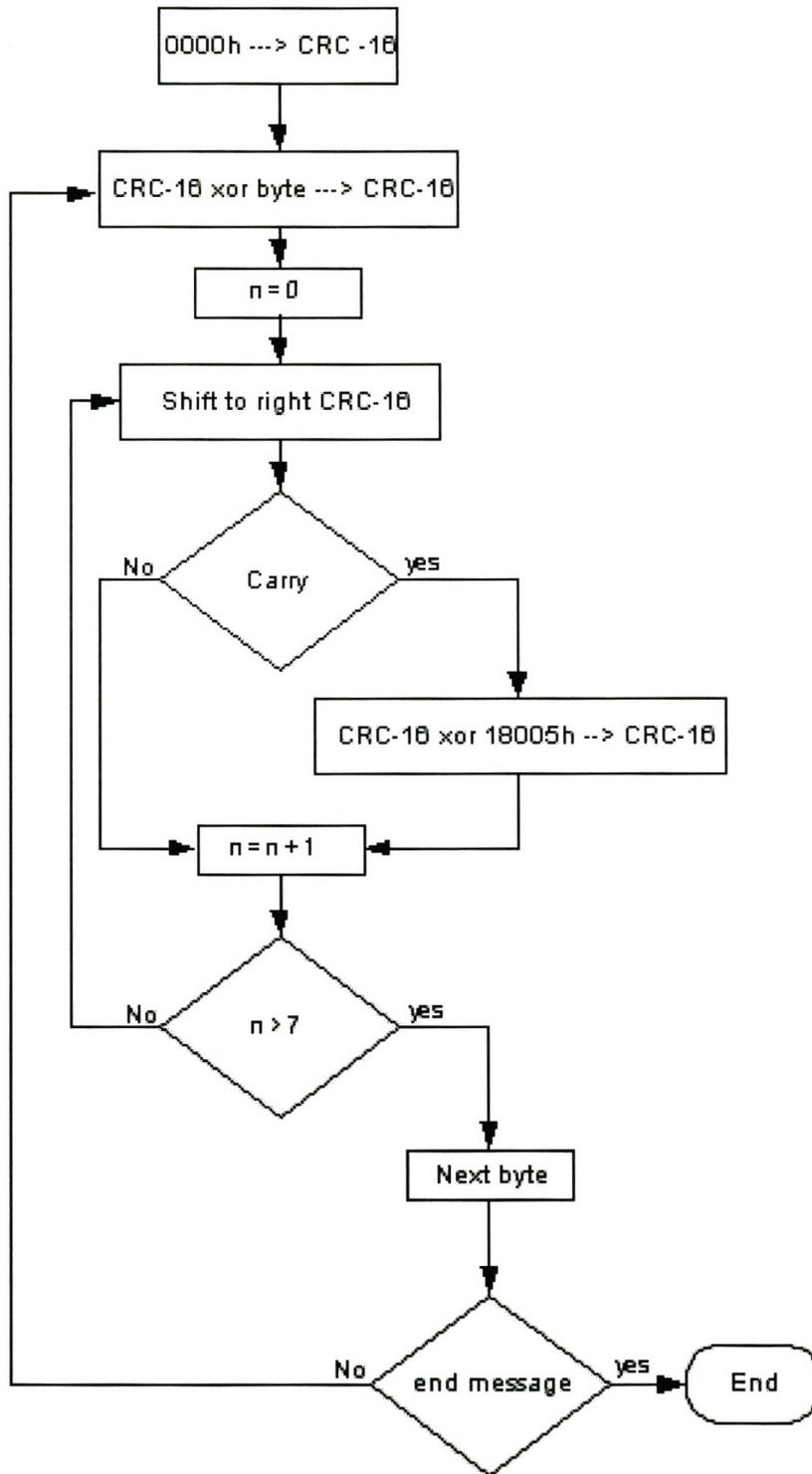


Figure 6.1: Implementation steps of CRC-16

- 3) Shift the register to the right by one bit.
- 4) If the bit that left the register on the right (flag) is a 1, execute the exclusive OR of the polynomial 1100000000000101 (18005h) with the register.
- 5) Repeat steps 3 and 4 eight times.
- 6) Execute the exclusive OR of the next character with the high order byte in the register and place the result in the register.
- 7) Repeat steps 3 to 6 for all the characters in the message.
- 8) The contents of the 16 bit register are the CRC code that is to be added to the message.

The CRC obtained in this manner has the MSB on the right so, if the source and the destination uses CRC obtained using steps followed above, option 2 discussed in Sec 3.3.4 (b) of error detection will never validate a frame. If data is transmitted with CRC obtained in this manner, the receiving device must first separate the received CRC from the message and recalculate the CRC, because the CRC calculated from the message plus received CRC will not be zero and the receiving device will always report error message received, even if the message is correct.

In order to validate a frame using option 1 discussed in Sec 3.3.4 (b), the CRC obtained in this manner must be first inverted before data transmission. The CRC-16 code was only used on the long frame defined by the IEC 870-5-101 standard, for high security. The standard itself supports checksum for error checking, but not CRC.

### 6.3 Time-slots Implementation

Time slots were implemented as shown in Figure 6.2. Slot size is wide enough to accommodate the poll time, data transmission time and two times process time. The calculations which helps to select a time-slot of 5 seconds where done as follows:

$$t_{ps} = t_{pr} + t_d + t_{procs} + t_{procm} + 2t_{prop} \quad (6.1)$$

where  $t_{ps}$  = Total poll time/station

$t_{pr}$  = Poll request time/station

$t_d$  = Data transmit time/station

$t_{procs}$  = Slave process time

$t_{procm}$  = Master process time

$t_{prop}$  = Propagation delay

The frame size = 240 bytes , Poll frame = 40 bytes and Transmission speed of 1200 bps. Asynchronous transmission with a byte consisting of 1 start bit, 8 data bits, 1 parity bit and 1 stop bit makes 1 character to carry 11 bits. Therefore,

$$t_{pr} = \frac{\text{Pollframe}}{\text{Transmissionspeed}} = \frac{40 \times 11}{1200} = 0.36 \text{ sec}$$

$$t_d = \frac{\text{Framesize}}{\text{Transmissionspeed}} = \frac{240 \times 11}{1200} = 2.2 \text{ sec}$$

The master station takes about 1.5 seconds to process a data frame, and each slave station takes about 300 milliseconds = 0.3 sec to process a poll frame. From 6.1, the total poll time per station is approximately:

$$t_{ps} = 0.36 \text{ sec} + 2.2 \text{ sec} + 0.3 \text{ sec} + 1.5 \text{ sec} = 4.36 \text{ seconds.}$$

The 5 seconds time slot was chosen to make the system stable. The allocation depends on the protocol used. For example, if RRP is used, time slot 1 is allocated to slave 1 and slot 2 to slave 2 and so on. If an error occurs, say at time slot 2 while polling slave 2, the master station allocates slot 3 to slave 2 and all the slots will be reallocated dynamically to adapt to any changes in the system. The time slot of 5 seconds was chosen to accommodate a poll frame and a data frame while considering the propagation delay, process time the systems.

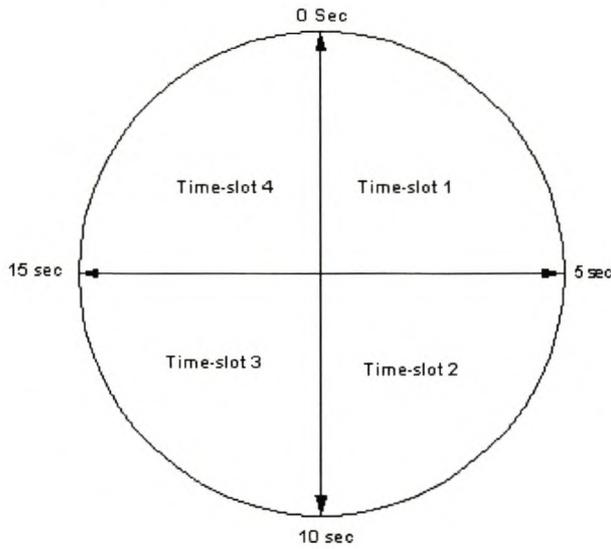


Figure 6.2: Implementation of time slots

## 6.4 Non-adaptive Polling System

In non-adaptive system, master station control and monitor the process of remote stations by polling. A system can either use a round-robin polling or priority polling scheme.

### 6.4.1 RRP implementation

After the master station has initializes and synchronizes all slave stations, polling for data starts at the start of the slot as shown in Figure 6.3. The master station sends a poll message to the first slave station and waits for data or NACK and starts a timer. If the timer expires without receiving data or NACK, the master station repeats the poll to the same station three times. If it fails to get a response, the master station gives up and poll the next station.<sup>1</sup> In case where the master station receives data or a NACK, it stores the information and polls the next slave station.

---

<sup>1</sup>The timer expires after a station wait for maximum round-trip propagation time is exceeded.

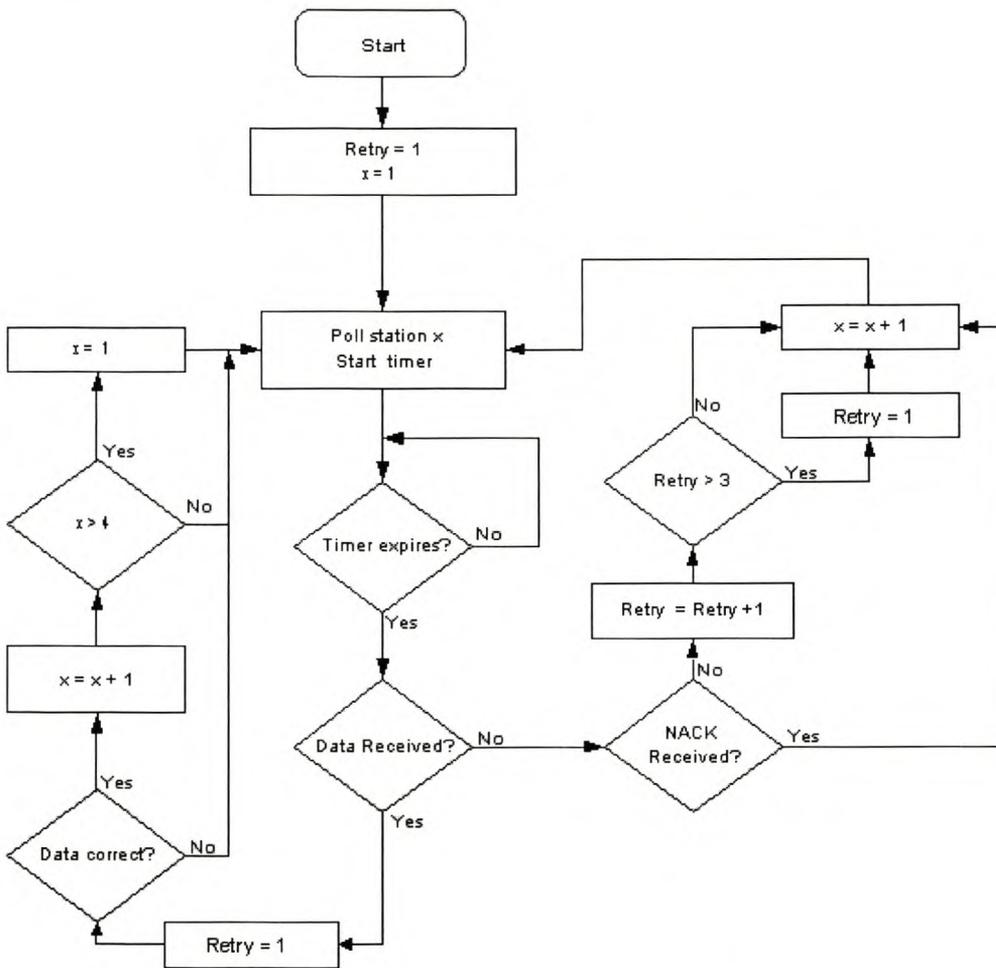


Figure 6.3: Master station data flow when polling slaves using RRP

### 6.4.2 Priority polling algorithm

It was decided to investigate possible improvement of the performance of straight RRP, by implementing the priority based RRP. Figure 6.4 shows the algorithm used by the master station to poll data from four slaves with a ratio of 8:4:2:1. According to the algorithm, slave station 1 receives more polls followed by station 2 then followed by station 3. This algorithm is suitable if slave 1 is generating events two times faster than slave 2, and four times faster than slave 3 and eight times faster than slave 4. The actual ratios are obviously, selected to suit the particular event generation rates.

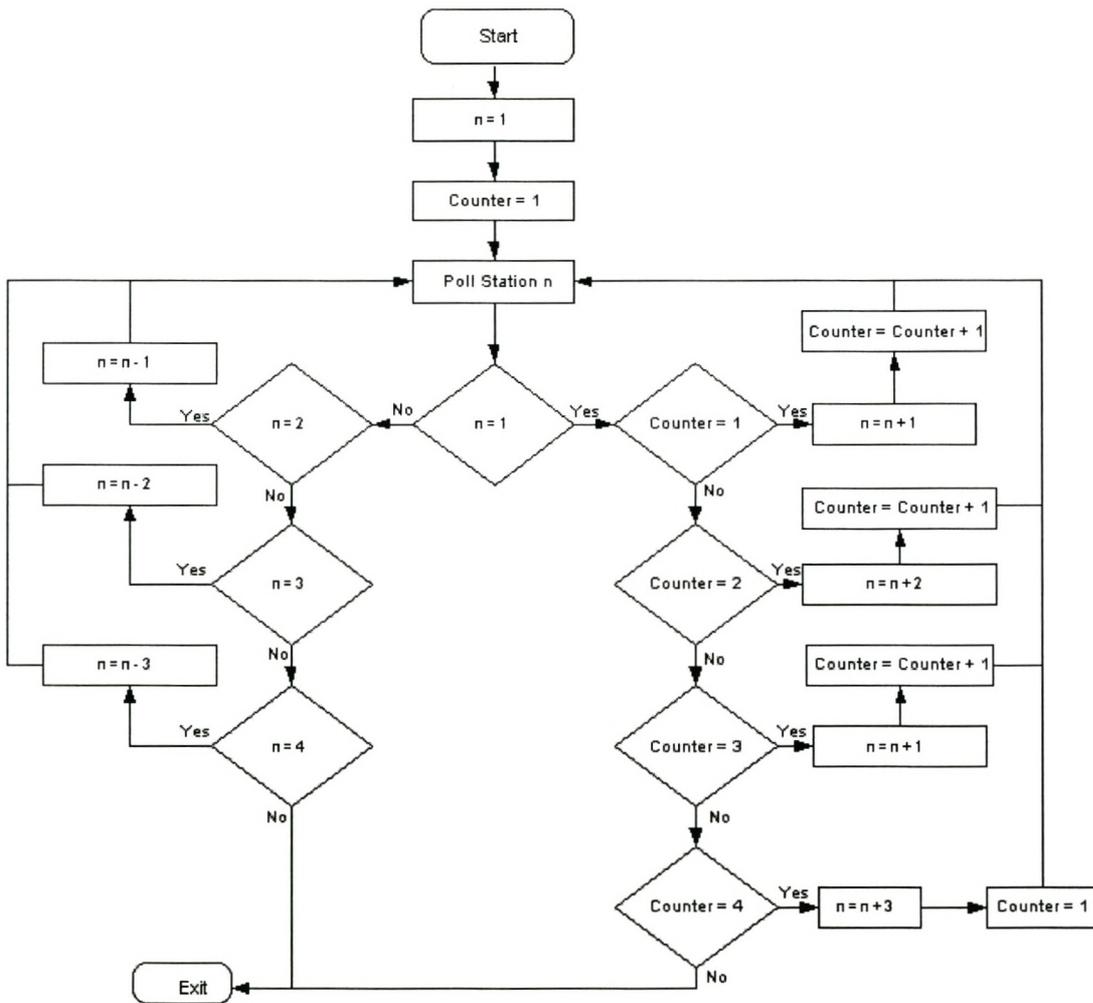


Figure 6.4: Polling slaves with events generation ratio 8:4:2:1

Figure 6.5 shows a polling type which is suitable for slave stations which generate events using a ratio 3:1:1:1. Using this technique, Slave 1 is polled more than slave 2, 3 and 4, to reduce unnecessary transaction wait time and improve system's throughput while polling slaves with no transaction to send.

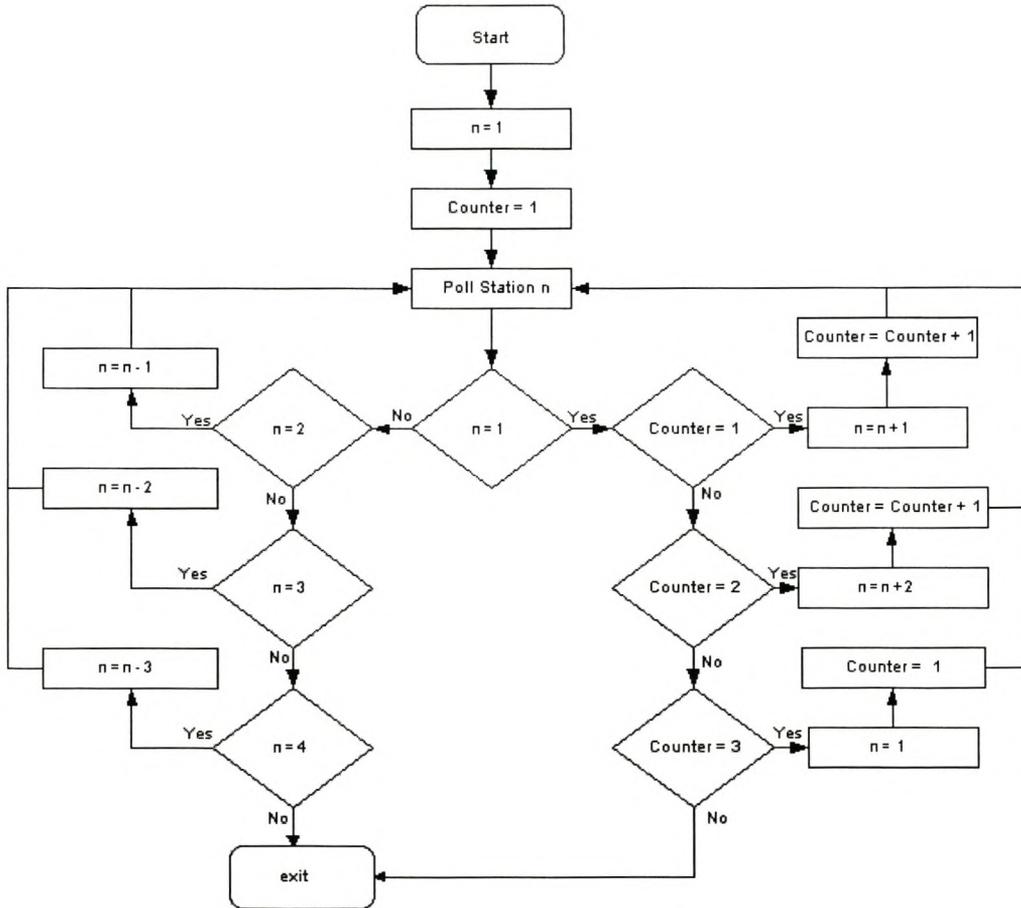


Figure 6.5: Polling of slaves with event generation ratio of 3:1:1:1

## 6.5 Adaptive Polling System

An adaptive allocation algorithm is one in which the algorithms and parameters used to allocate time-slots change dynamically according to the previous and current system behavior. At the start, the master station polls slaves using straight RRP for data communications. This was done, because RRP gives each slave station a fair chance to send data to the master station. The data received were placed on a CSV file for analysis. This system was designed as an improvement of both the RRP and priority polling. The master station continuously evaluates system performance and decides which polling algorithm will produce high throughput and low delays.

Figure 6.6 shows the steps taken by an adaptive system to increase system performance. Initially, the master station polls the slaves in a round robin fashion and adapt to a new poll strategy depending on the results obtained. After a predetermined number of polls, the system will analyze positive responses per number of polls. If the performance is satisfactory, it will continue to poll slaves using a RRP technique. In case where the system performance drops, or the mean wait time per station increases, the system will adjust itself to a polling scheme that will increase the system performance and decrease the mean wait time as with normal priority polling. This protocol is useful when the patterns of events on the slave stations are not predeterminable. The adjustment does not require any user intervention, it occurs dynamically.

Figure 6.6 shows steps taken by the master station to enhance system performance. After 20 polls per slave, the master station changes the polling method if the results are not satisfactory. The reason for choosing 20 polls per slave was, because, we need the system to detect changes in a system as early as possible so that the system performance will only be down for a short period. If the results are satisfactory, the master station continue to poll with the same technique until continuing analysis shows a performance decrease. The adaptive polling system greatly enhances system performance, without any change in the physical operating environment.

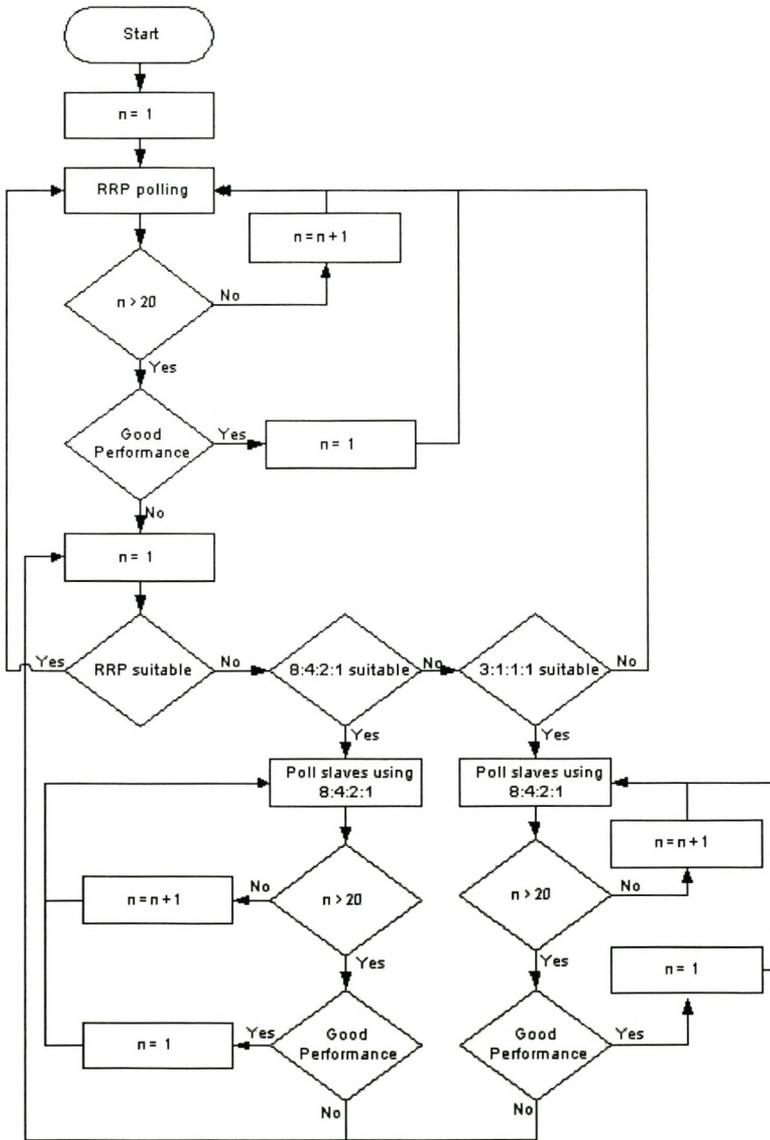


Figure 6.6: Adaptive system procedure

## 6.6 Slave Station Implementation

Figure 6.7 shows the way in which each slave station process an incoming poll frame. When a master station polls a slave station, a slave station must first check the validity of the frame. If the slave address matches the received address, it performs the necessary function dictated by the function code. For example, if function code equal to binary number “1101” is attached on a received frame, the slave must reply with class 2 user data. In case where the station does not have anything to send, it reply with a short frame defined by IEC 870-5-101 protocol with a function code equal to 9 (binary “1001”) to inform the sender that data requested is not available. Note that slave stations do not always expect a poll frame. The master station can also send a “direct command”, “synchronization command” or “initialization command” to the slave.

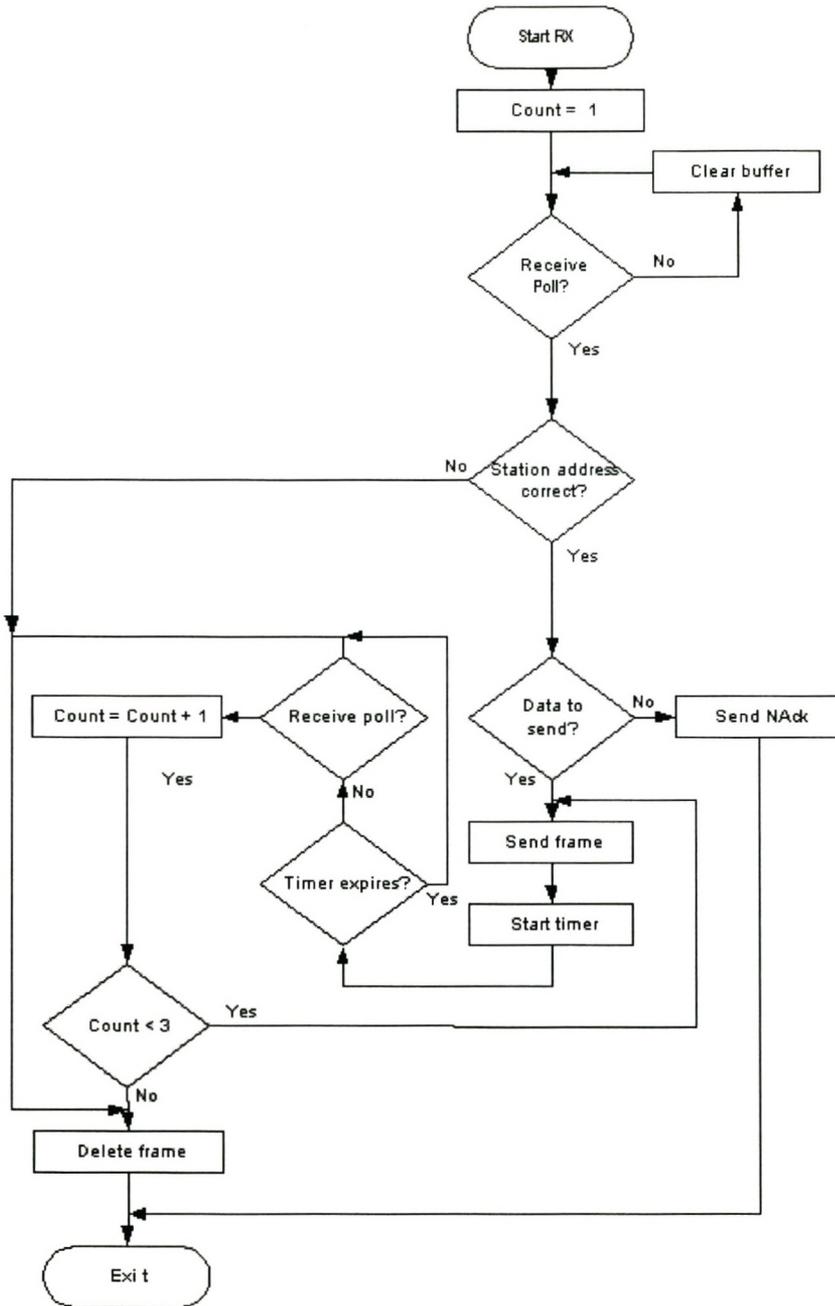


Figure 6.7: Process undergone by a slave station when receiving a poll message

## 6.7 Summary

This chapter shows some implementation procedures followed in order to secure the required outcome of the project. After showing of these implementation steps have been completed, the subsequent results may then be evaluated.

# Chapter 7

## Results and System Reliability

### 7.1 System Reliability

The system performs as expected. The system supports both automatic and manual operation. It can run for hours without losing any information. If one needs to get information at the present time, there are some function codes, which can be used to satisfy the user's needs. The system has the ability to handle errors and to notify the user if one of the slave stations is off line after a predetermined number of polls. The transmission occurs at low speed (1200 baud) and the serial communication component shown in Fig 7.1 was set to the conditions supported by the IEC 870-5-1-101 protocol (asynchronous transmission with even parity). The component also specifies the serial port to be used, which in this case, is COM 1. Using this component, one can also choose whether to include hardware, or software handshake, or none.

Furthermore, the master station can control and monitor real time processes at the remote stations using commands or function codes specified by the data link layer protocol IEC 870-5-101. Figure 7.2 shows the function codes defined by the IEC 870-5-101 protocol and 4 slave stations that are controlled by the master station. The master station selects one function code and one slave station it wants to send a command to and specifies the type of command by clicking a "direct command to slave button" or "request data button".

Figure 7.3 shows a window which is used to select the polling scheme manually. Only one polling scheme can be selected at a time and if a radio button corresponding to priority polling is selected, the priority type must be selected as well, or else the system will use the default polling scheme, which is RRP. If the radio button corresponding to RRP

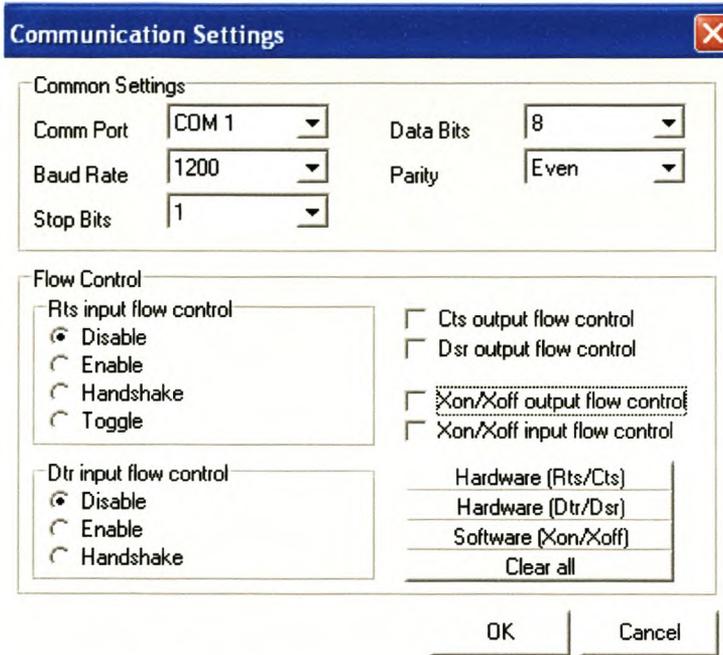


Figure 7.1: Serial port communication setting component

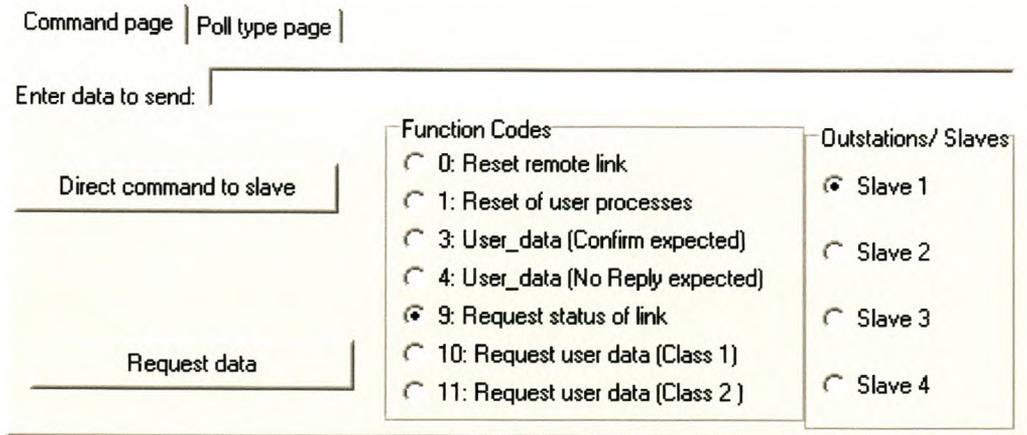


Figure 7.2: Master station control command

is selected, the master station poll slaves using RRP. If the adaptive polling button is selected, the system starts with RRP and it can adapt to any priority type depending on the results obtained.

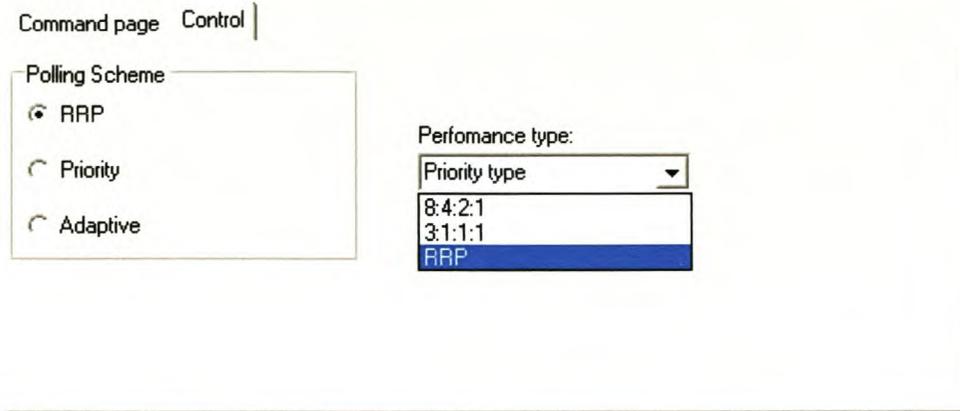


Figure 7.3: Polling type control page

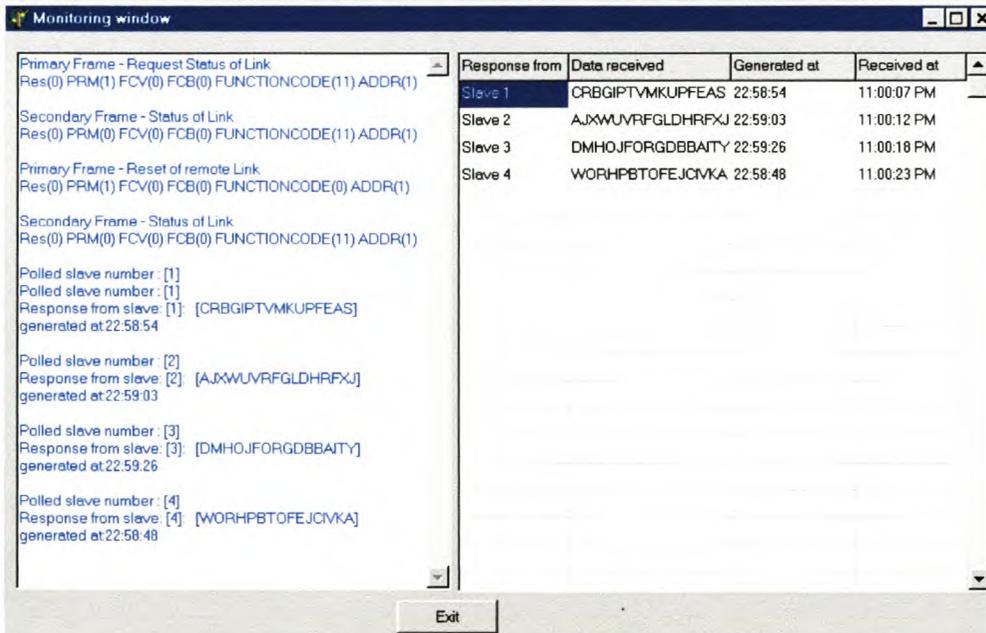


Figure 7.4: Initialization of slave using IEC 870-5-101 protocol

After initialization and synchronization of slaves, polling for data starts automatically using the poll type selected in Figure 7.3. Figure 7.4 shows the initialization of Slave 1 and the results obtained for RRP. The master station records the time of event generation and the time in which the event, is received. Out of those results, the master station, calculates the wait time per event for all the slaves per poll. From Figure 7.4, it can be

seen that slave 1 was polled two times. For the first time, the master station did not receive either data or an ACK frame and decided to poll the same slave station again. This system has the ability to handle errors with the use of CRC-16 and to control the flow of data without collisions. Whenever a slave responds with data, master station put a '1' in a table, and a transaction wait time per event and a '0' if an ACK is received. After 20 polls per station the master calculates the number of times a station replies with data and the mean wait time. These results were used to compare the adaptive and the non-adaptive system.

The reliability of the system has been proved and it is time to have a look at the results obtained.

## 7.2 Results

### 7.2.1 Introduction

It is clearly necessary that results be obtained and evaluated in terms of the initial goals set for the project. Results were obtained and compared, for non-adaptive and adaptive systems to prove the claim that adaptive polling schemes could produce high throughput and low latency with better fairness guarantees, compared to the non-adaptive schemes. In order for the comparison to be valid, both systems must collect data over the same period, i.e data samples can be taken for two hours in both systems.

In this adaptive system, the master station records a positive response and mean wait time per 10 polls for each slave station, using different polling techniques. Out of every 20 polls made to each slave station, the master station takes two samples of 10 polls each and records average positive responses and mean wait times. If the performance is not satisfactory, the adaptive system adapts dynamically to a polling scheme that will result in high data throughput and low latency, whereas the non-adaptive system will continue to poll using the same polling technique. In case where there is no suitable polling scheme to adapt, the adaptive system continues to use the same poll type while analyzing the performance. The results will be compared under both high- and low-loading. Fairness is also taken into considerations for stations which generate events at the same rate.

As a start, a case where good results were obtained will be discussed.

### 7.2.2 Acceptable 3:1:1:1 priority polling results

Figure 7.5 shows the results for slave 1 when the system is polling slaves at a ratio of 3:1:1:1 slave station. It can be seen that out of every 10 polls made to this station, it manages to reply more than 6 times (80% average). Figure 7.9 shows the mean transaction wait time per 10 polls of Slave 1. Figure 7.6, Figure 7.7 and Figure 7.8 shows throughput performance of Slave 2, Slave 3 and Slave 4 respectively. All of these stations' performance is greater than 70%. All these stations were polled every 30 seconds because the rate of event generation is three times less compared to Slave 1 event generation rate. Figure 7.10, Figure 7.11 and Figure 7.12 shows the delay suffered by Slave 2, Slave 3 and Slave 4 respectively. The delays suffered by a slave station to send events to the master is at acceptable range. A system which yields this result is regarded as a stable system, and adaptation of polling scheme is not necessary.

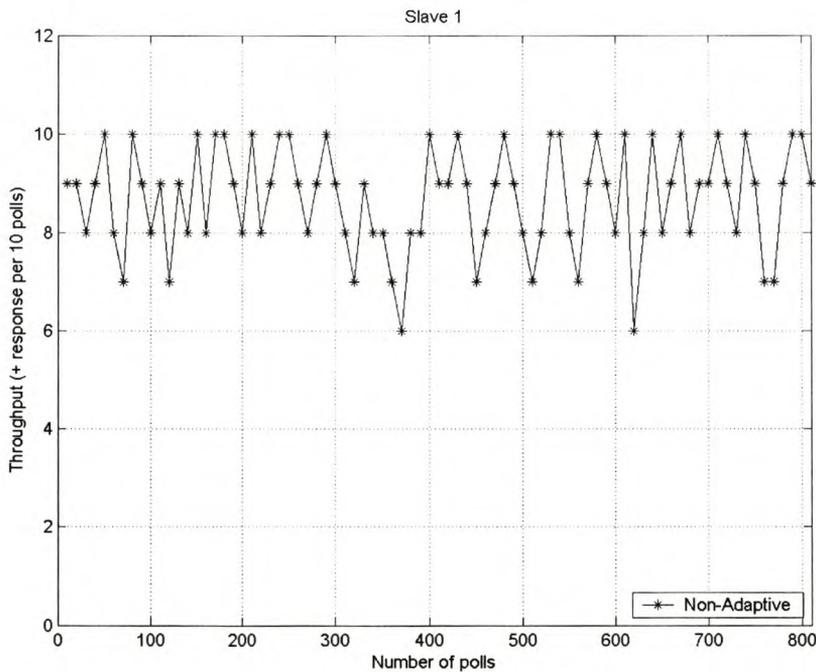


Figure 7.5: High loading performance of slave 1 using 3:1:1:1 priority ratio

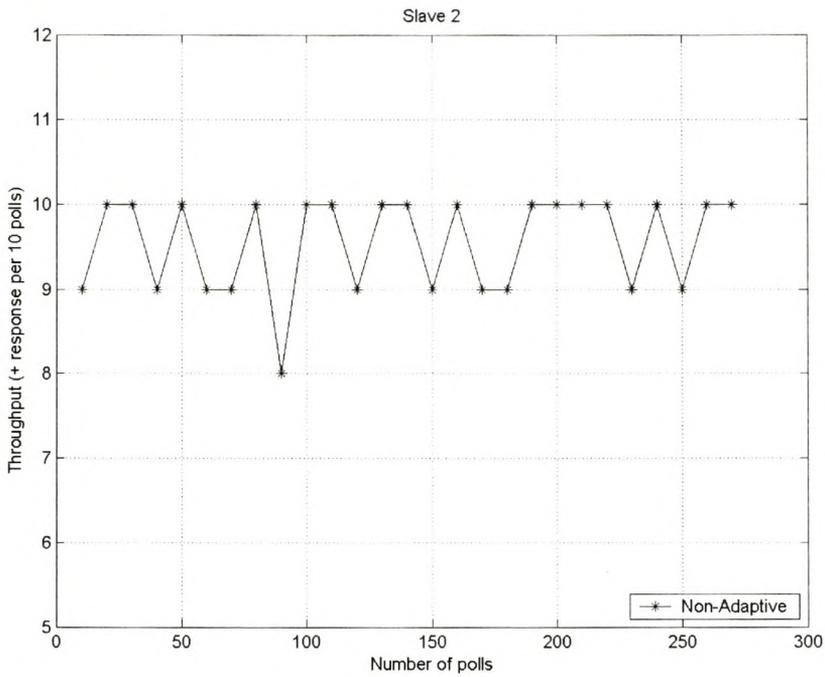


Figure 7.6: High loading performance of slave 2 using 3:1:1:1 priority ratio

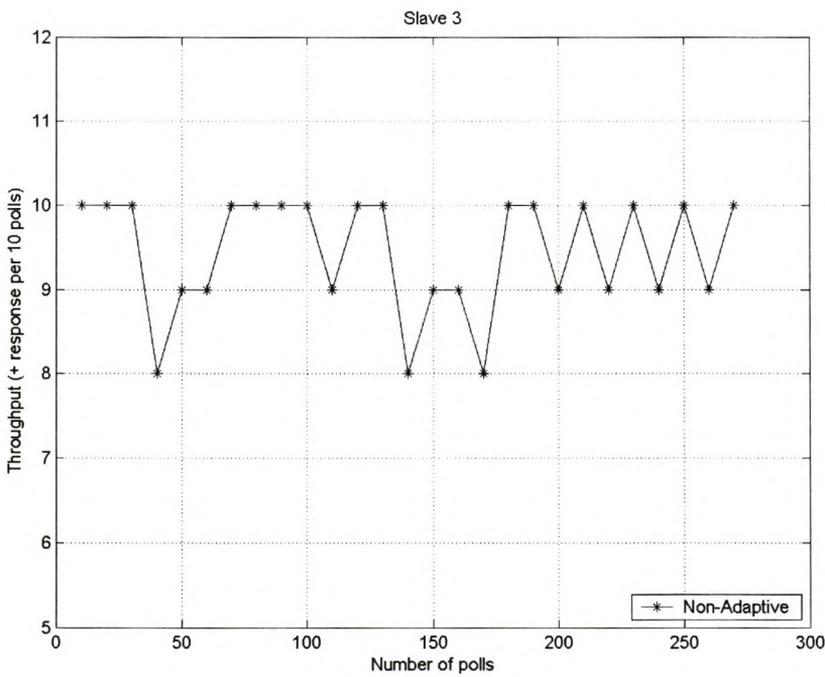


Figure 7.7: High loading performance of slave 3 using 3:1:1:1 priority ratio

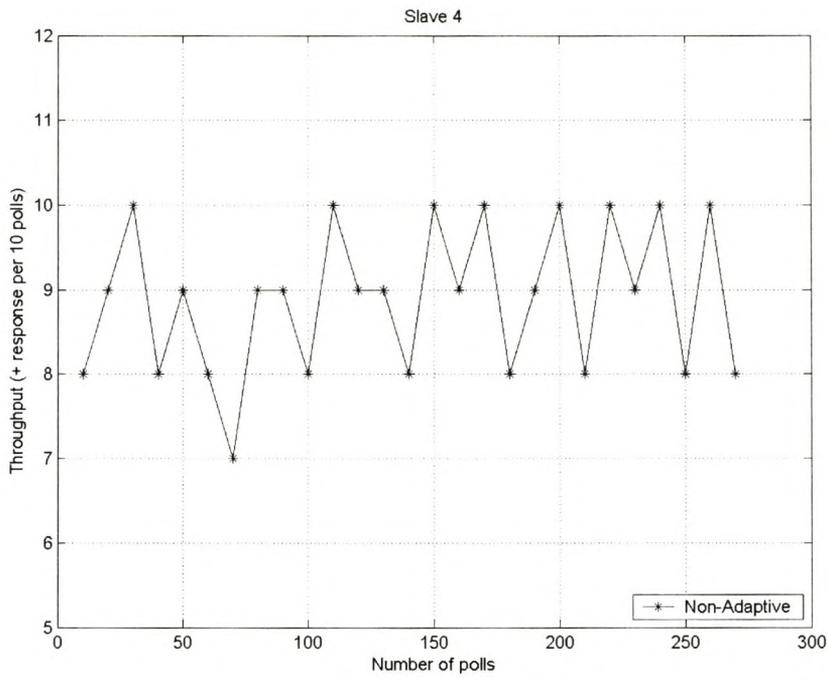


Figure 7.8: High loading performance of slave 4 using 3:1:1:1 priority ratio

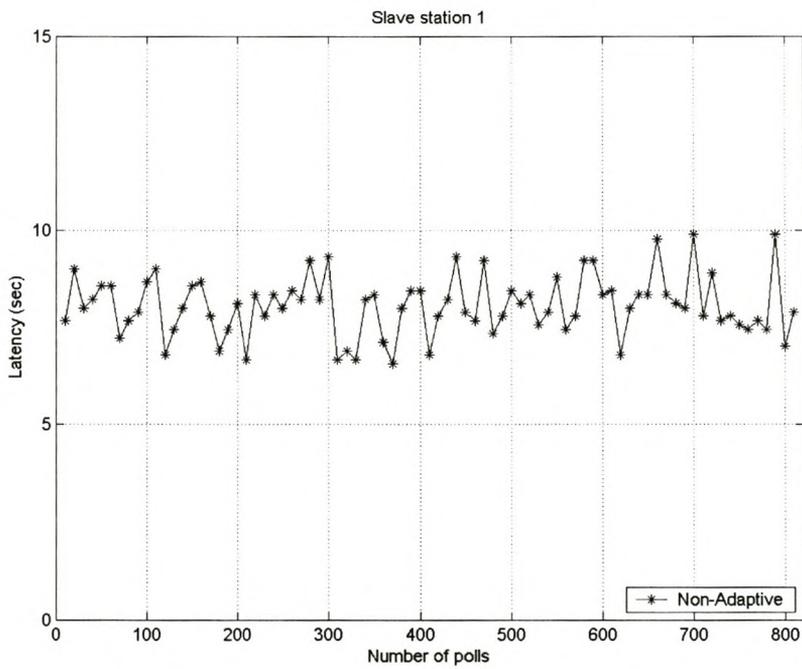


Figure 7.9: Slave 1 high loading mean wait time

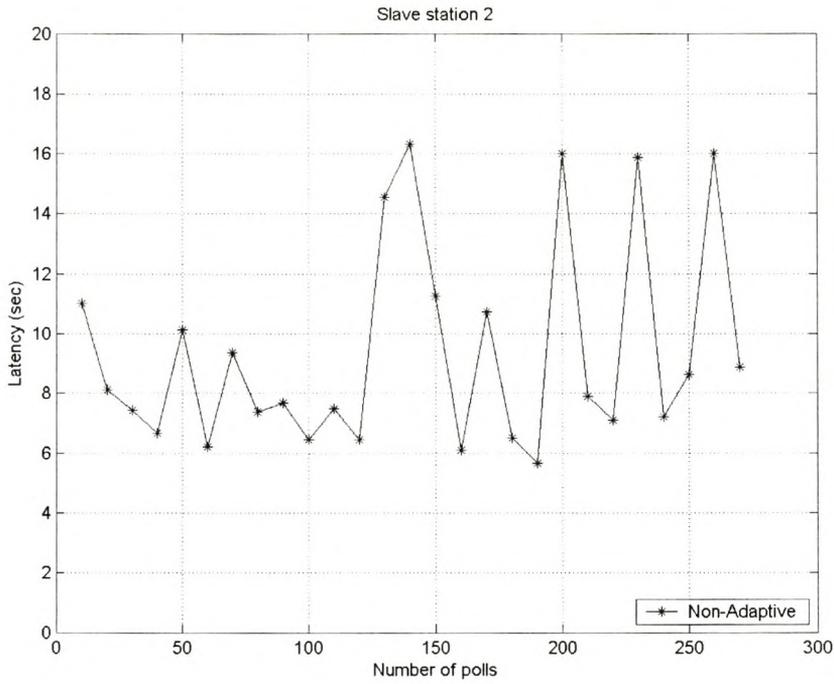


Figure 7.10: Slave 2 high loading mean wait time

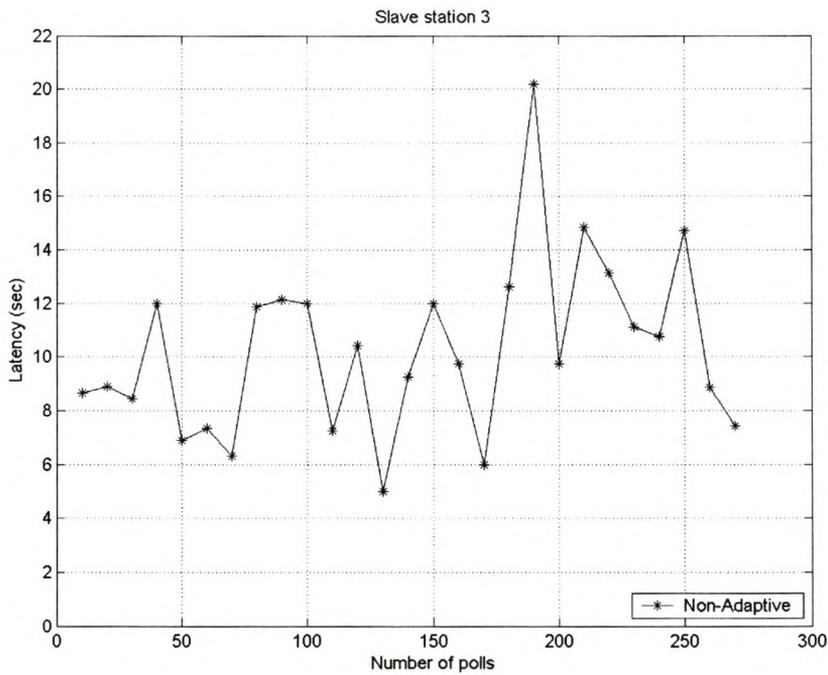


Figure 7.11: Slave 3 high loading mean wait time

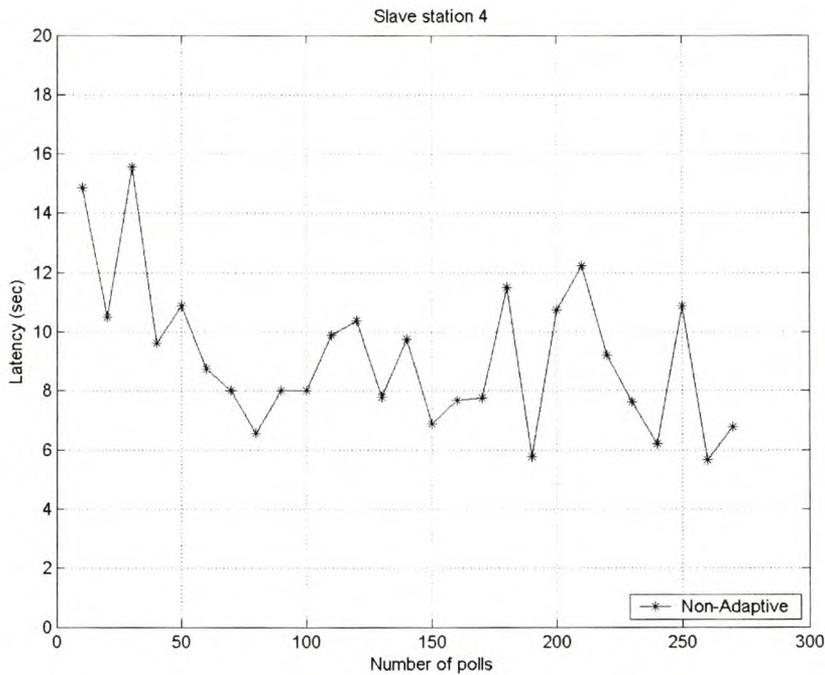


Figure 7.12: Slave 4 high loading mean wait time

### 7.2.3 Performance comparison between adaptive and non-adaptive polling schemes

In this section, the results obtained for the Round Robin scheme and adaptive scheme at different loadings, will be investigated. Case 1 will compare results obtained for low loading RRP with the results of an adaptive system. Case 2 will show results obtained for a system which drops its performance from high to low loading and Case 3 will show a system which starts at low loading and increases the load with time.

#### 7.2.3 (a) Case 1: Low-loading RRP vs adaptive polling system

Initially, both adaptive and non-adaptive polling systems were using normal RRP to poll slave stations. After 20 polls, the adaptive polling system realizes that the performance of slaves 2, 3 and 4 is not satisfactory. The non-adaptive polling system continues to poll slave stations using the RRP protocol whereas the adaptive polling system detects that some of the slave stations' performance is low and slave 1 is delayed while generating events faster. The adaptive polling system checks if there is any built-in poll type which

can improve throughput and decrease latency per station. After analyzing the results, the adaptive polling system realizes that a suitable polling scheme is a priority polling with a polling ratio 3:1:1:1 and change dynamically to this poll type. It can be seen that throughput performance of slave 2, 3 and 4 increases as shown in Figure 7.14, Figure 7.15 and Figure 7.16 while the latency of all slaves decreases for adaptive polling system as shown in Figure 7.17, Figure 7.18, Figure 7.19 and Figure 7.20. Slave 1's throughput was always at maximum as shown in Fig 7.13 because it generates events faster (every 5 seconds). Using RRP, station 1 was polled every 20 seconds whereas using 3:1:1:1 priority polling, it was polled every 10 seconds. Slave 2, 3 and 4 where all generating events at a rate of one event per 30 seconds and they were all polled once in 30 seconds using 3:1:1:1 polling ratio by the adaptive polling system and once in 20 seconds by the non-adaptive polling system. In Figure 7.14, Figure 7.15 and Figure 7.16, it can be seen that the adaptive polling system polls slave fewer times than the non-adaptive polling system during the same polling period with higher throuput and low mean transaction wait time.

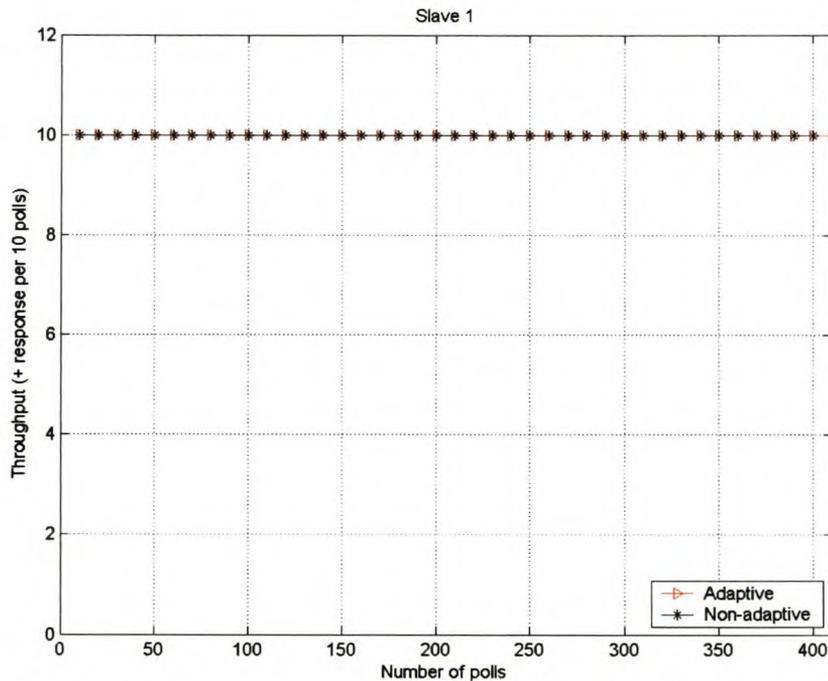


Figure 7.13: Slave 1 data throughput comparison between adaptive and non-adaptive polling system

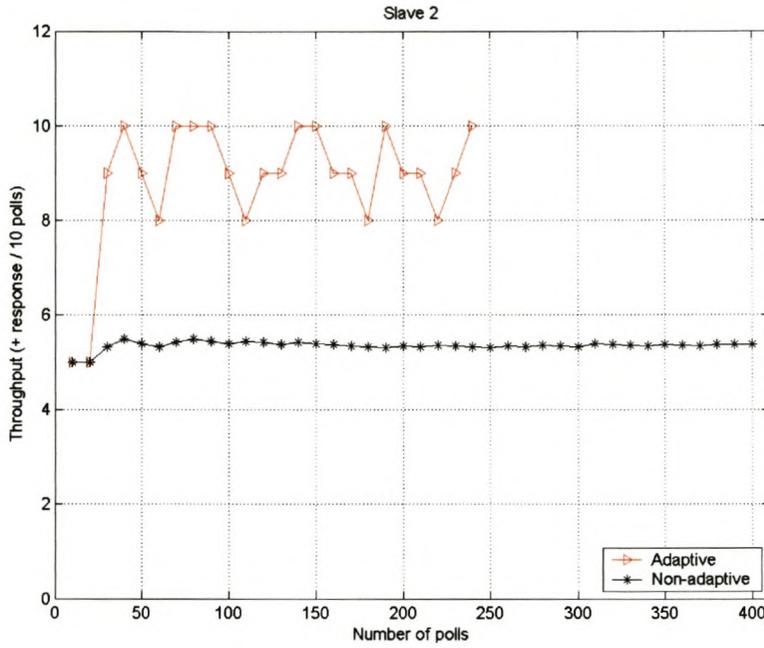


Figure 7.14: Slave 2 data throughput comparison between adaptive and non-adaptive polling system

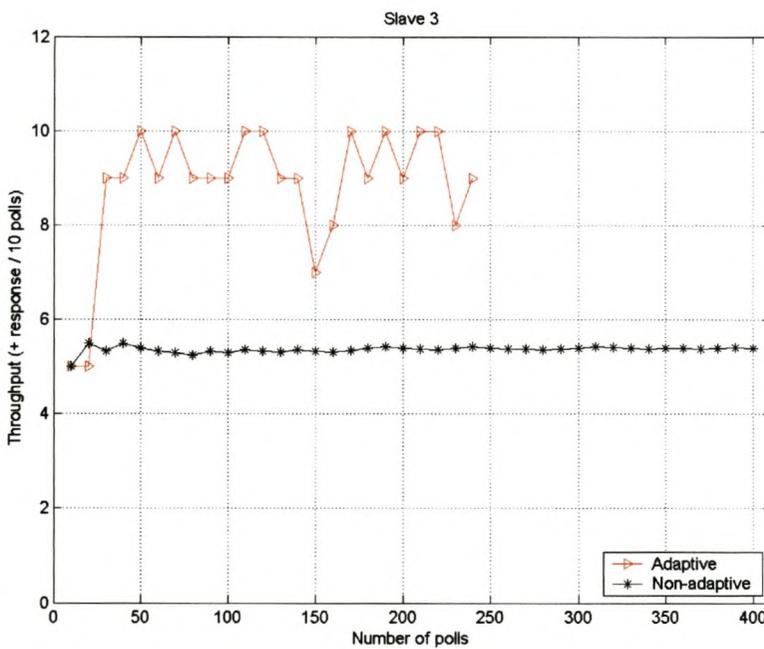


Figure 7.15: Slave 3 data throughput comparison between adaptive and non-adaptive polling system

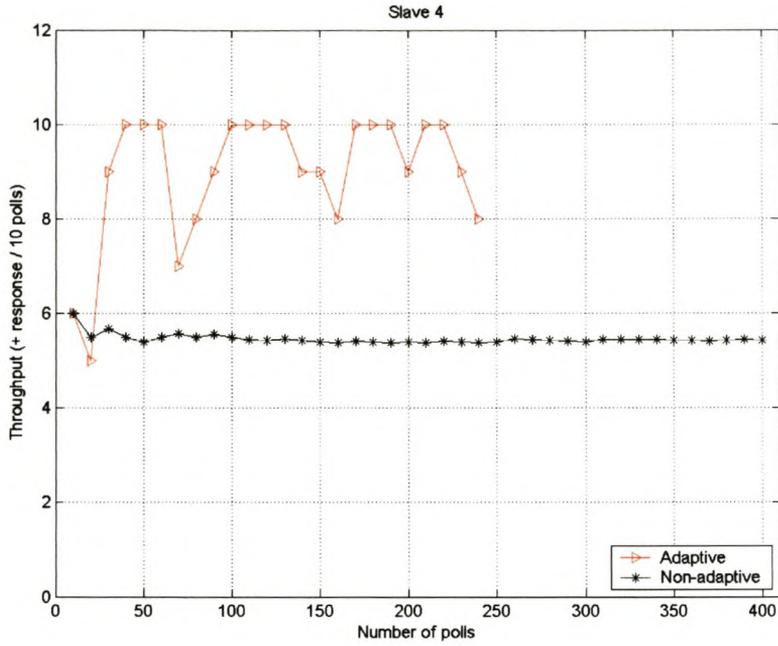


Figure 7.16: Slave 4 data throughput comparison between adaptive and non-adaptive polling system

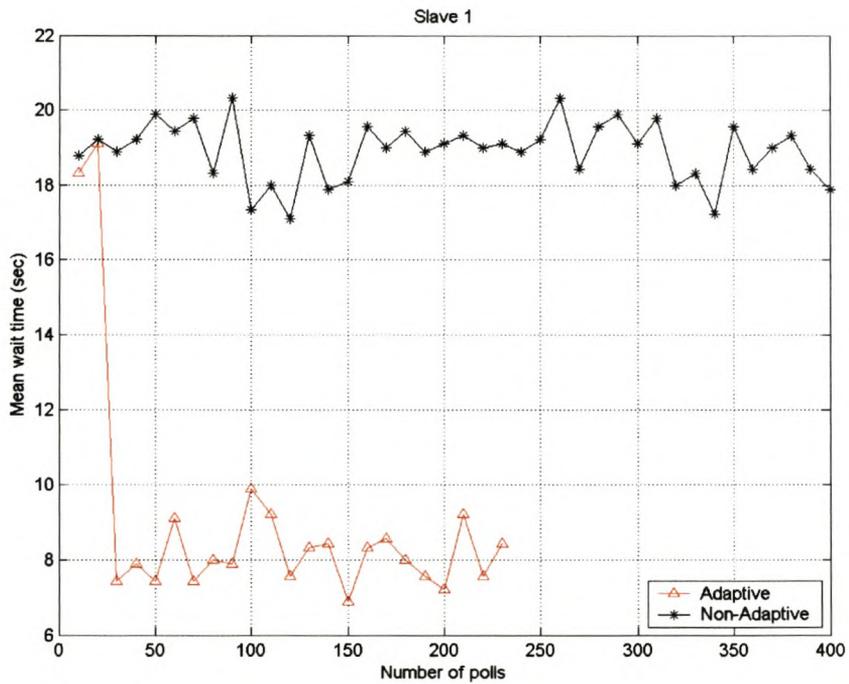


Figure 7.17: Slave 1 delay comparison between adaptive and non-adaptive polling system

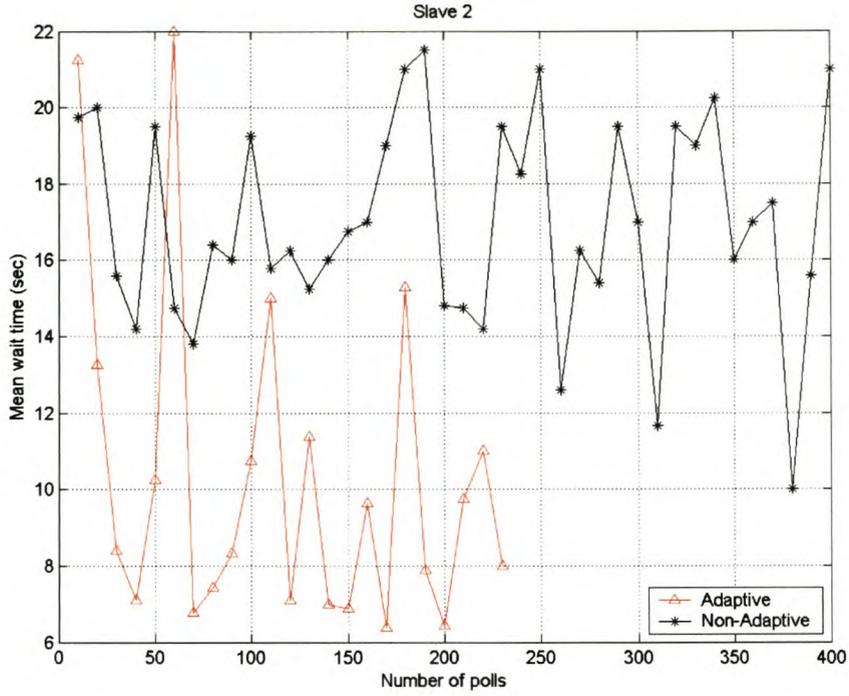


Figure 7.18: Slave 2 delay comparison between adaptive and non-adaptive polling system

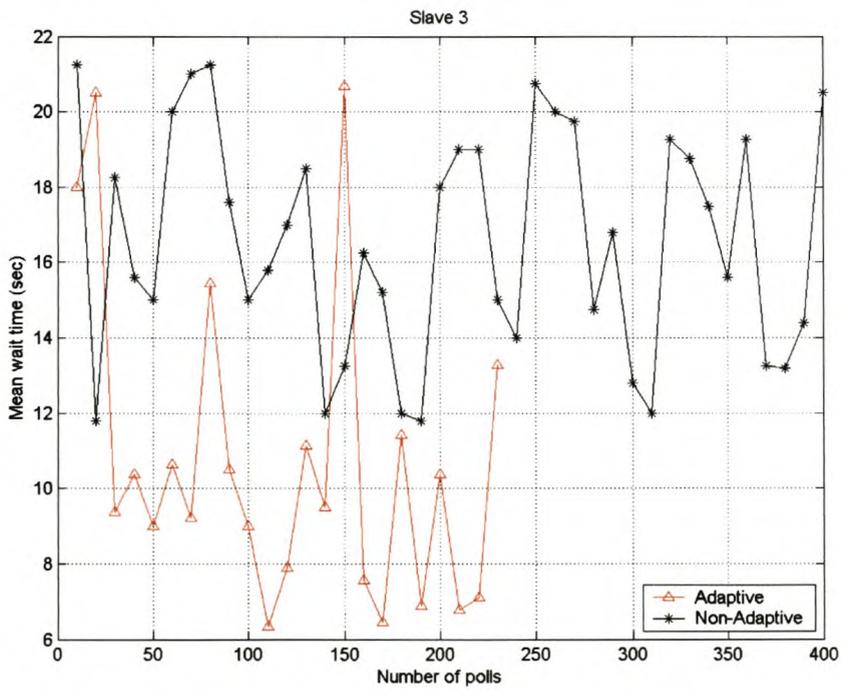


Figure 7.19: Slave 3 delay comparison between adaptive and non-adaptive polling system

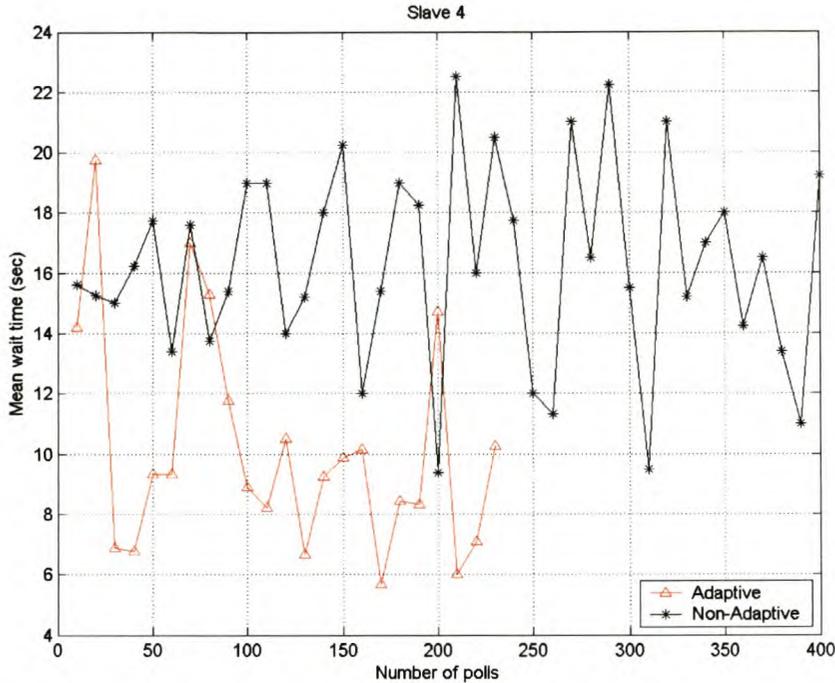


Figure 7.20: Slave 4 delay comparison between adaptive and non-adaptive polling system

### 7.2.3 (b) Case 2: High- to low-loading RRP vs adaptive polling system

In this case, the system was producing high throughput at the early stages. After about 140 polls, the event generation rates of slave 3 and 4 drops from 5 to 40 seconds per event and the transaction wait times in both systems (adaptive and non-adaptive) decreases as shown in Figure 7.27 and Figure 7.28, while the throughput decreases as well as shown in Figure 7.23 and Figure 7.24. The event generation rate of slave 1 remains the same and the throughput performance was not affected when other system drops their performances. Figure 7.21 shows the throughput of slave 1 in both systems. Slave 2 event generation rate drops from 5 to 10 seconds per event and that result in no change in throughput and reduction in transaction wait times on both systems as shown in Figure 7.22 and Figure 7.26 respectively.

After 220, the adaptive polling system compares the previous and current results of throughput and transaction wait time, at each slave and realize that there is a drop in performance in slave 3 as shown in Fig 7.23 and slave 4 as shown in Fig 7.24 and there is a change in event generation rate in slave 2. The result causes the adaptive polling system to dynamically change its polling scheme and use the scheme, which will enhance

system performance (in this case the adaptive polling system adapt dynamically to polling ratio 8:4:2:1). The adaptive polling system also try to minimize mean wait transaction times which was low at that point, to suit its polling type.

The non-adaptive polling system has no answer to this problem, it continues to poll slaves using RRP and system's throughput never improves. The non-adaptive polling system waste time slots by polling stations, which have nothing to send and cause higher delays to the stations, that generate events faster. Figure 7.25 shows the decrease in transaction mean wait times at slave 1 in adaptive polling system compared to the delays caused by the non-adaptive polling system .

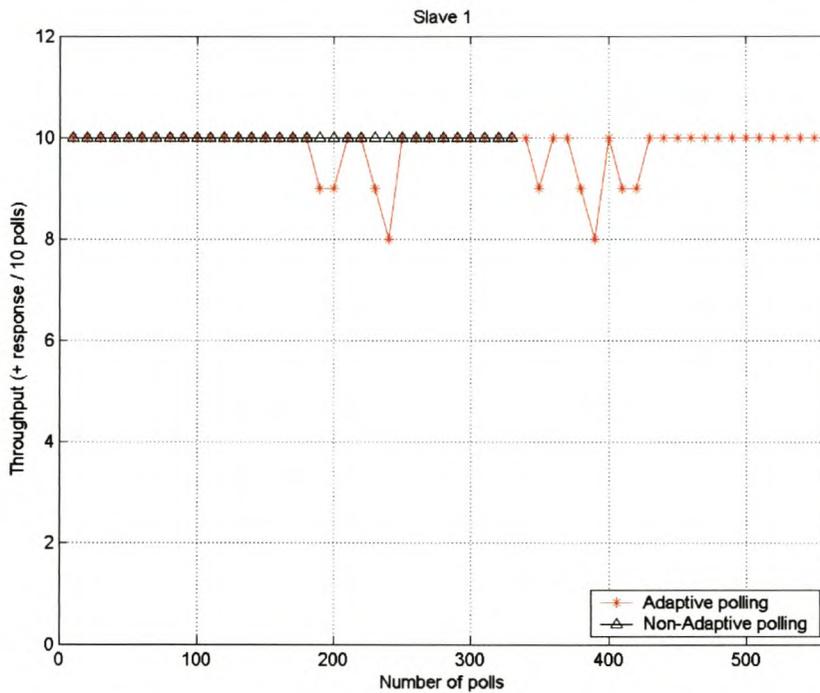


Figure 7.21: Slave 1: Throughput comparison between adaptive and non-adaptive system

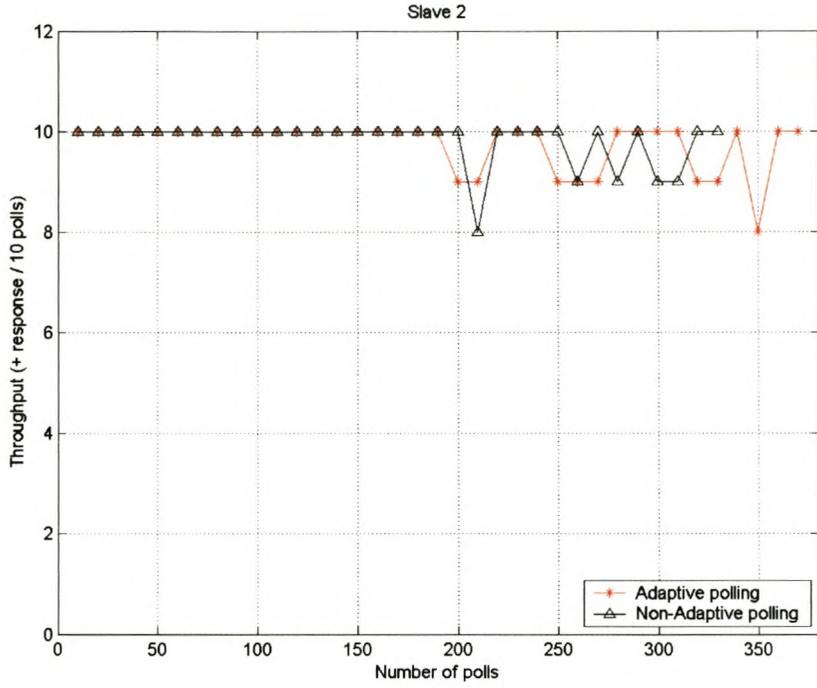


Figure 7.22: Slave 2: Throughput comparison between adaptive and non-adaptive system

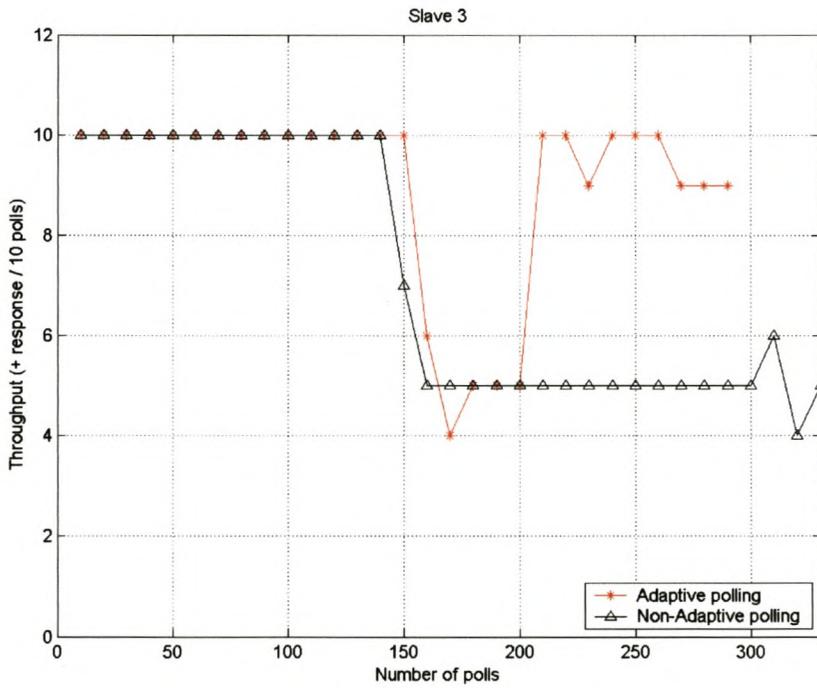


Figure 7.23: Slave 3: Throughput comparison between adaptive and non-adaptive system

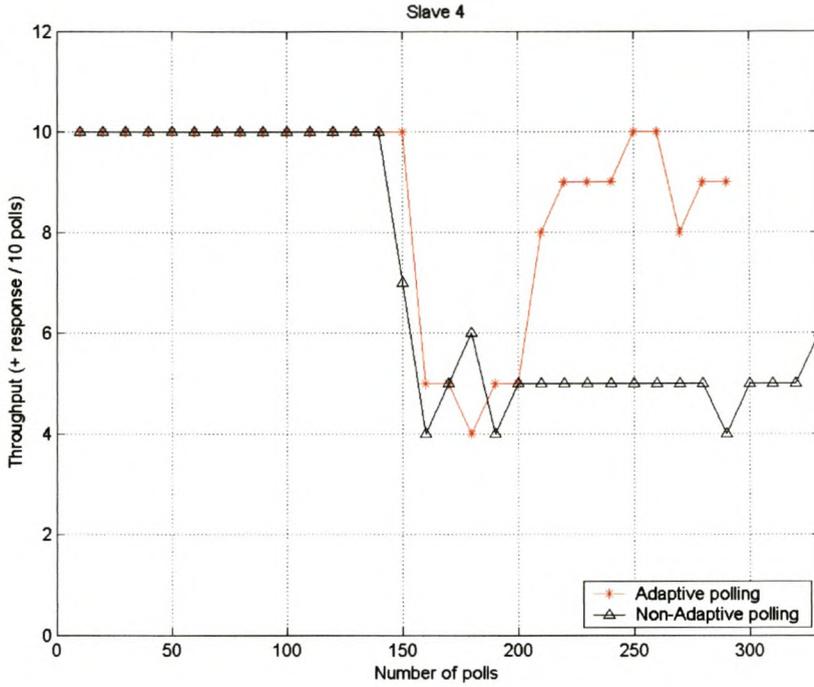


Figure 7.24: Slave 4: Throughput comparison between adaptive and non-adaptive system

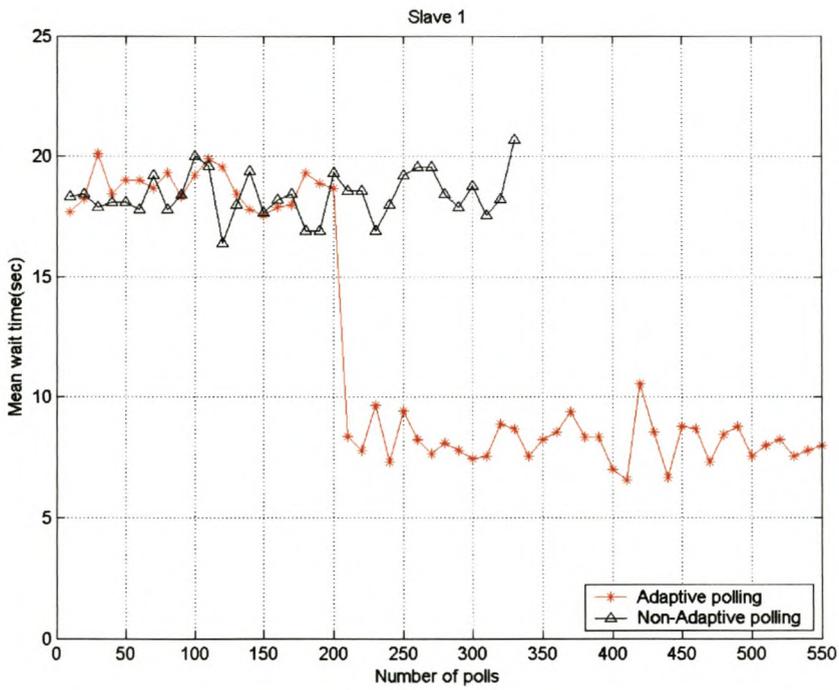


Figure 7.25: Delays comparison between adaptive and non-adaptive system: Slave 1

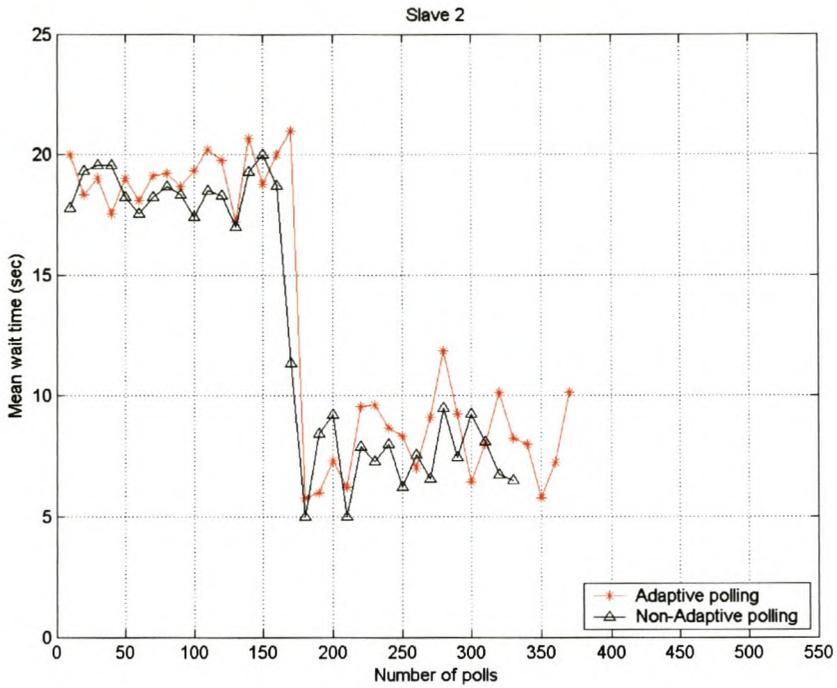


Figure 7.26: Delays comparison between adaptive and non-adaptive system: Slave 2

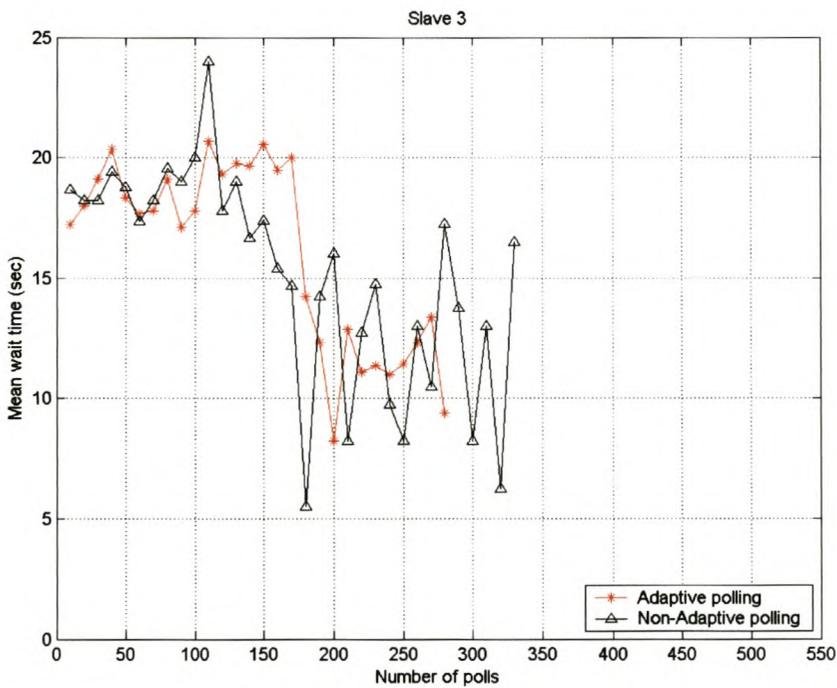


Figure 7.27: Delay comparisons between adaptive and non-adaptive system: Slave 3

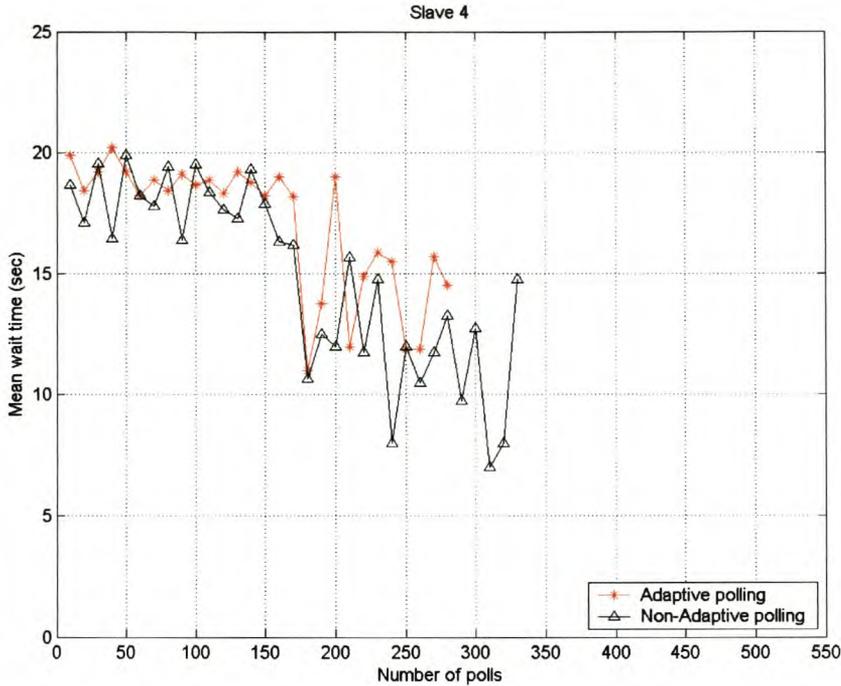


Figure 7.28: Delay comparisons between adaptive and non-adaptive schemes: Slave 4

### 7.2.3 (c) Case 3: Low- to high-loading RRP

This section presents results obtained when both systems (adaptive and non-adaptive), where initially polling slaves using a priority type 3:1:1:1 because slave 1 where generating events three times faster than other slaves. If RRP were used in this case, results were going to be like those discussed in case 1. For the first 220 polls, the system was yielding high throughput and low mean transaction wait times while using a polling ratio 3:1:1:1. As polling continues, slave stations 2, 3, and 4 start to generate events at the same rate as slave 1 and their transaction mean wait time increases as well. Both system's slaves stations 2, 3 and 4 suffer a higher delays as shown in Figure 7.33, Figure 7.34, Figure 7.35, Figure 7.36. The adaptive polling system detects the increases in transaction mean wait times at slave 2, 3, and 4 and dynamically changes its polling type to RRP scheme. The latter causes a delay on slave 1 but causes the decrease in delay on slave 2, 3 and 4. It was fair to give stations, which are generating events at the same rate equal chance of transmitting data.

The non-adaptive polling system continues to poll using the polling ratio 3:1:1:1 while producing higher throughput in all slave stations as shown in Figure 7.29, Figure 7.30,

Figure 7.31 and Figure 7.32. The mean transaction wait time for Slave 1 in the non-adaptive polling system remains the same and for the adaptive polling system increases as shown in Figure 7.33 to promote fairness among the slaves. From these results, it can be seen that the adaptive system not only improves system performance, but it also reduce delays and provide fairness to stations, which generate events at the same rate.

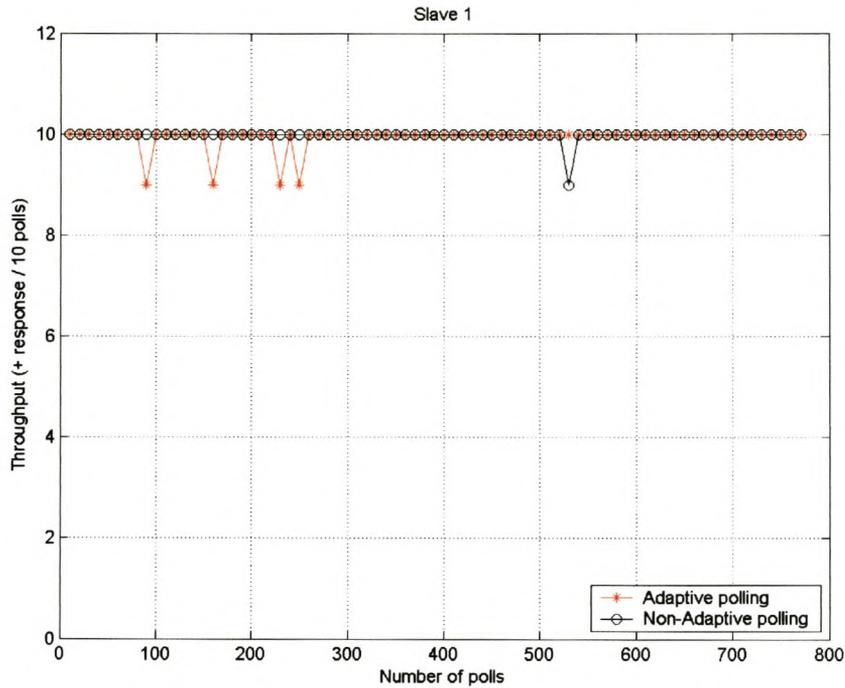


Figure 7.29: Slave 1: Low to high loading comparison

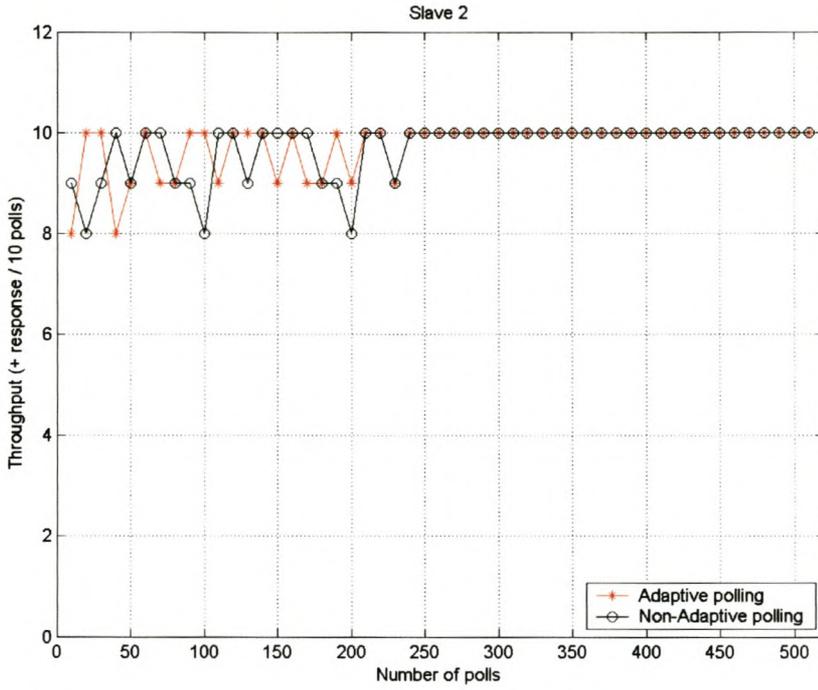


Figure 7.30: Slave 2: Low to high loading comparison

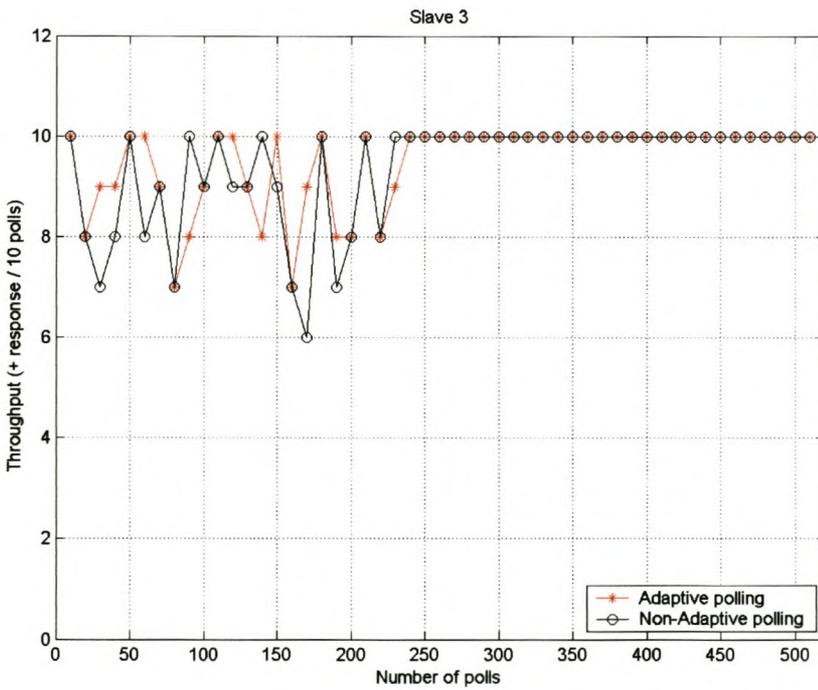


Figure 7.31: Slave 3: Low to high loading comparison

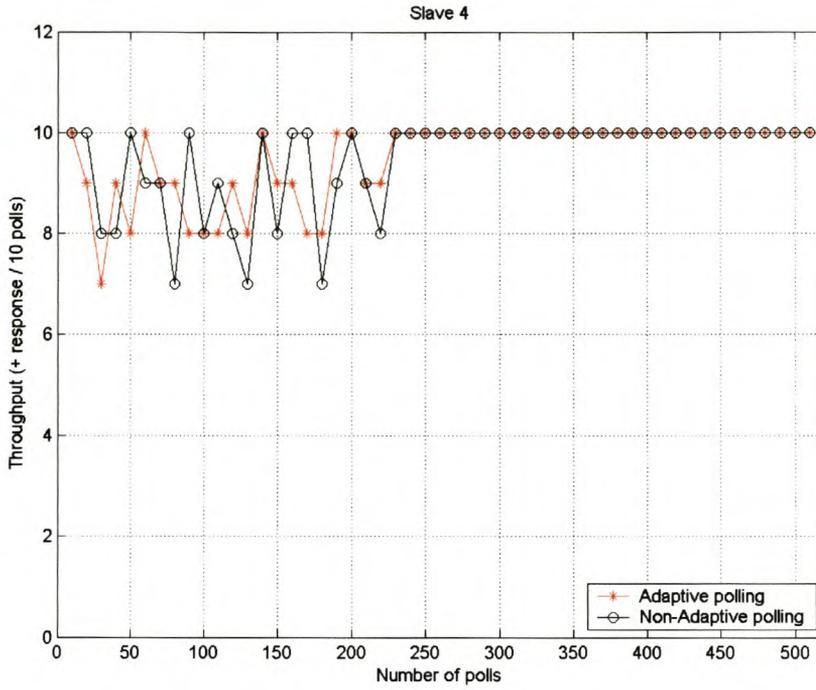


Figure 7.32: Slave 4: Low to high loading comparison

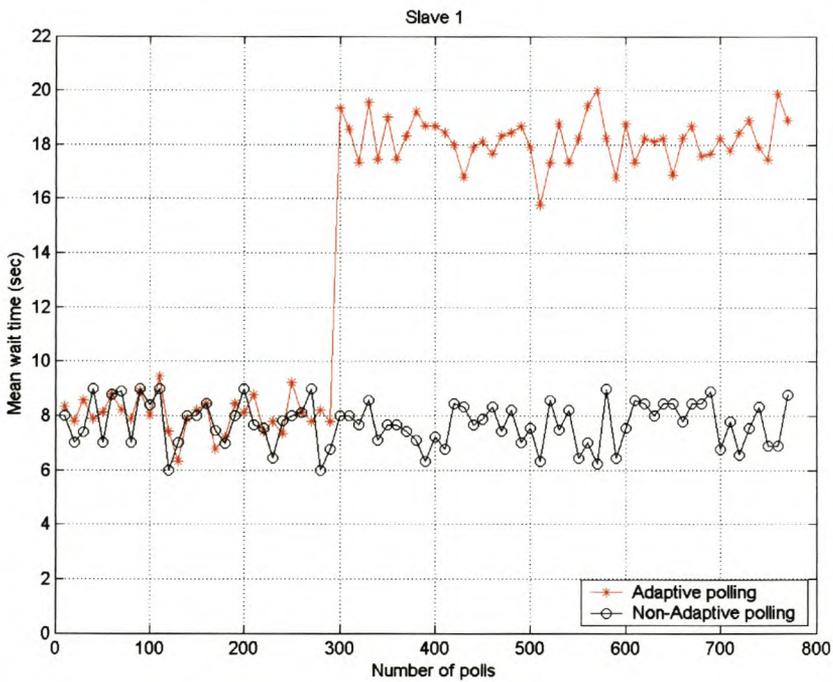


Figure 7.33: Slave 1: Low to high loading latency comparison

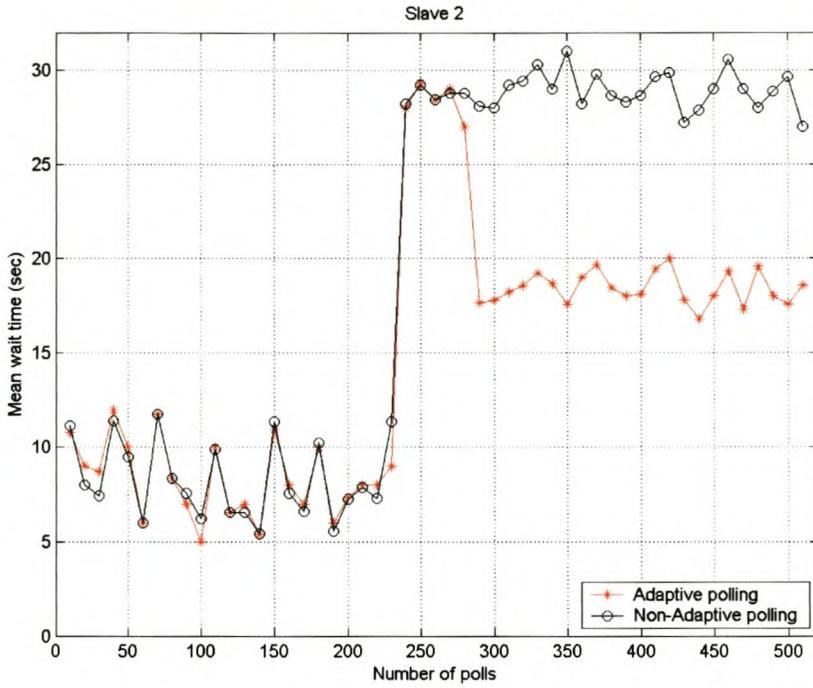


Figure 7.34: Slave 2: Low to high loading latency comparison

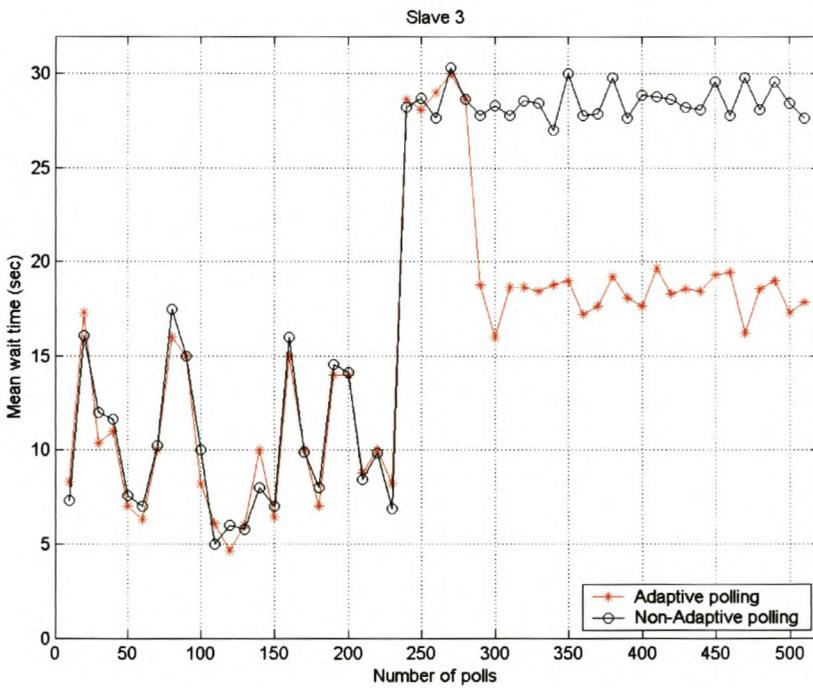


Figure 7.35: Slave 3: Low to high loading latency comparison

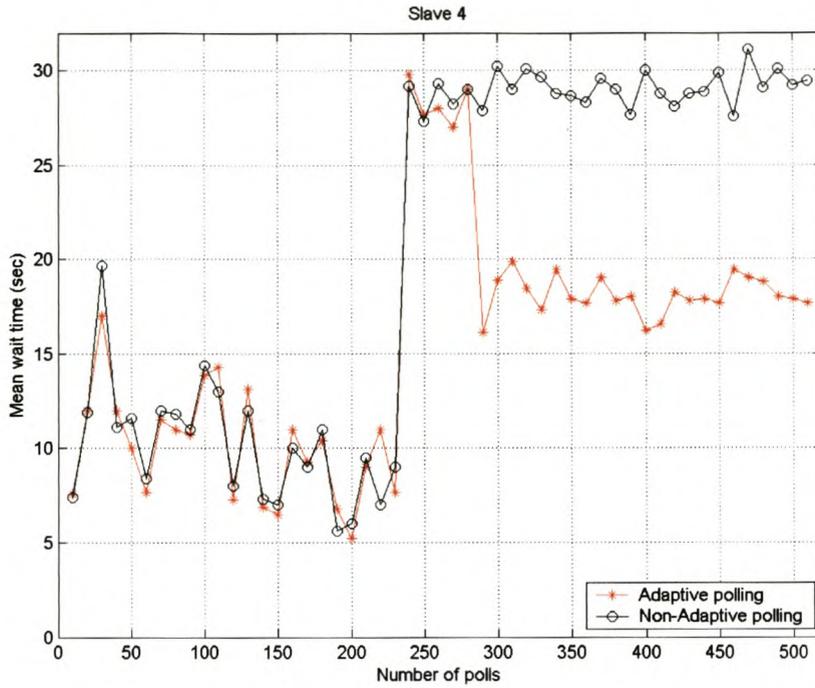


Figure 7.36: Slave 4: Low to high loading latency comparison

### 7.3 Summary

It is clear that the intention to possibly improve system performance by a time slotted based self-adaptive polling scheme, was achieved. The ability of the system to dynamically adjust to varying operational input conditions result in a drastic reduction in transaction wait times and corresponding increases in throughput. The adaptive system, not only increase the throughput and mean wait transaction times, but also gives a fairness guarantee to all the slave stations.

## Chapter 8

# Conclusions and Recommendations

### 8.1 Conclusions

The stated aim of this research was to investigate means to improve the data throughput performance of networks using standard narrow band based hardware. These aims were met by implementing the following:

1. The development of an air-based time slotted protocol, to utilize the known performance benefits of such schemes, for the applications as intended.
2. The implementation of a standard, internationally accepted format, such as IEC 870-5-101 over the developed protocol. This would enable the application of the development using a variety of hardware platforms, without significant driver adaptation.
3. The development of several non-adaptive and adaptively scheduled polling schemes, to utilize the protocol as above mentioned and to realize a significant performance improvement over conventional approaches.

Based on the above, it can be stated that:

1. Slotted timebase protocols present an improvement over non-slotted time base versions, but is normally not implemented over narrow band radio links. In this case it's merit was proved.

2. After analyzing the adaptive and non-adaptive polling systems, it was found that:
  - The adaptive system yields high throughput guarantees, low mean wait-time guarantees and provides more fairness among the slaves under low- and variable loading, than the non-adaptive polling system.
  - Under high-loading, adaptive and non-adaptive polling systems yield the same results, as could be expected.
3. Based on the points mentioned in 2, it can be concluded that the adaptive polling system would generally give an improvement over the non-adaptive polling system.
4. The goals of the project have been met but still leaves room for a large amount of potentially interesting work.

## 8.2 Recommendations

It has been proved that an adaptive polling system is an improvement over non-adaptive polling system, yielding high data throughput and low latency. For future projects, forward error correction can be used so that the system can be reliable on a noisy environment. For an adaptive system to be absolutely robust, it will require more built-in poll types. The system must be able to handle virtually any system change. This improvement will have to be accompanied by much additional statistical analysis and innovative adaptation algorithms.

# Appendix A

## RS232C Control Lines

The RS232 standard describes the functions carried out by several control signals between the DTE and the DCE. The following control signals implement most of the important functions of an R232 DTE to DCE link.

DB-9 Pin No.	Full Name	DB-25 Pin No.
Pin 1	Carrier Detect)	Pin 8
Pin 2	Received Data	Pin 3
Pin 3	Transmit Data	Pin 2
Pin 4	Data Terminal Ready	Pin 20
Pin 5	Signal Ground	Pin 7
Pin 6	Data Set Ready	Pin 6
Pin 7	Request To Send	Pin 4
Pin 8	Clear To Send	Pin 5
Pin 9	Ring Indicator	Pin 22

Table A.1: DB-9 and DB-25 pin connectors

Data terminal ready (DTR) - This is a signal from the DTE to the DCE. When asserted, DTR indicates that the DTE is ready to accept data from the DCE. In systems with a modem, it maintains the connection and keeps the channel open. If DTR is negated, the communication path is broken. In everyday terms, negating DTR is the same as hanging up a phone.

Request to send (RTS) - This is a signal from the DTE to the DCE. When asserted, RTS indicates to the DCE that the DTE wishes to transmit data to it.

Data set ready (DSR) - This is a signal from the DCE to the DTE which indicates the

readiness of the DCE. When this signal is asserted, the DCE is able to receive from the DTE. DSR indicates that the DCE (usually a modem) is switched on and is in its normal functioning mode (as opposed to its self-test mode).

Clear to send (CTS) - This is a signal from the DCE to the DTE and, when asserted, indicates that the DCE is ready to receive data from the DTE.

Transmit Data (TxD) - Data Line for transmission

Receive Data (RxD) - Data Line for reception

## A.1 PC to modem connections

Depending on the type a modem, connections between the PC and a radio's data interface port may be different. Some modems uses rj-11 to DB-9 adapter to interface with a PC whereas some modems DB-9 to DB-25 adapter cable. This promotes a construction of an adapter cable from scratch. Figure A.1 shows a connection between EL705 OEM series radio modem and a PC.<sup>1</sup>

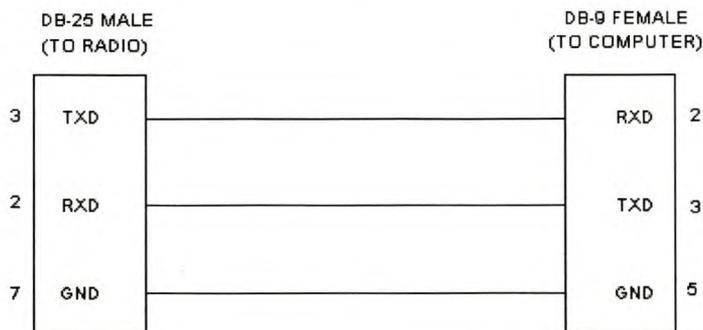


Figure A.1: DB-25 to DB9 Adapter Cable

<sup>1</sup>Note that a radio modem uses a DB-25 port for communication whereas a PC uses a DB-9 serial port and it was used to achieve the goal of the project.

## Appendix B

### RS232 specifications

#### RS-232 Specs.

SPECIFICATIONS		RS232	RS423
Mode of Operation		SINGLE -ENDED	SINGLE -ENDED
Total Number of Drivers and Receivers on One Line		1 DRIVER 1 RECVR	1 DRIVER 10 RECVR
Maximum Cable Length		50 FT.	4000 FT.
Maximum Data Rate		20kb/s	100kb/s
Maximum Driver Output Voltage		+/-25V	+/-6V
Driver Output Signal Level (Loaded Min.)	Loaded	+/-5V to +/-15V	+/-3.6V
Driver Output Signal Level (Unloaded Max)	Unloaded	+/-25V	+/-6V
Driver Load Impedance (Ohms)		3k to 7k	>=450
Max. Driver Current in High Z State	Power On	N/A	N/A
Max. Driver Current in High Z State	Power Off	+/-6mA @ +/-2v	+/-100uA
Slew Rate (Max.)		30V/uS	Adjustable
Receiver Input Voltage Range		+/-15V	+/-12V
Receiver Input Sensitivity		+/-3V	+/-200mV
Receiver Input Resistance (Ohms)		3k to 7k	4k min.

## Appendix C

# Commonly used Delphi functions and Source code

Check the disk supplied to see a complete source code.

Below are some of the mostly used Delphi functions. A function shown below converts an integer Value into a binary string of size specified by Digits plus 1. For example a function `IntToBin(65,7)` produce a binary "01000001"

```
Function IntToBin(Value: LongInt; Digits : Integer): String;
Var
  i : Integer;
Begin
  Result := '';
  For i := Digits downto 0 do
    if Value and (1 shl i) <> 0 then
      Result := Result + '1'
    else
      Result := Result + '0';
  end;
```

*APPENDIX C. COMMONLY USED DELPHI FUNCTIONS AND SOURCE CODE*<sup>97</sup>

A function below convert a binary string into a decimal number(integer value). For example, If one need to convert a binary strings A = "01000001" to decimal value, a function BinToDec(A) will yield an integer value 65.

```
Function BinToDec(M : String):integer;
  var
    i,j : Integer;
begin
  J := 0;
  For i := 1 to Length(M) do
    Begin
      if M[i] = '0' then
        Begin
          j := 2*j + 0;
        end
      else
        Begin
          j := 2*j + 1
        end
      end;
    Result := j;
  end;
```

*APPENDIX C. COMMONLY USED DELPHI FUNCTIONS AND SOURCE CODE*98

The CRC function was coded as shown below. The input to the function can either be a string of ASCII characters or binary numbers and the output will be an integer value. This function has been explained in the implementation chapters.

```
Function CRC16(S: String): Word;
Const
  Gpoly = $18005;           // generator polynomial: x16 + x15 + x2 + 1
var
  CRC : Word;
  Index1, Index2 : Byte;
Begin
  CRC := 0;
For Index1 := 1 to length(S) do
  Begin
    CRC := (CRC xor (ord(S[Index1]) shl 8));
    for Index2 := 1 to 8 do
      if ((CRC and $8000) <> 0) then
        CRC := ((CRC shl 1) xor Gpoly )
      else
        CRC := (CRC shl 1)
    end;
    CRC16 := (CRC and $FFFF)
  end;
```

To see how this function used check the disk attached

## Appendix D

# Radio Modem and Its Specifications



Figure D.1: EL705 OEM Series 450 Mhz Data Transceiver

<b>General</b>	
Type	: 450 MHz Licensed OEM Board Level EL7054 - 330-512 Radio Type Synthesized, Half Duplex, Channel 12.5 kHz Channel Spacing, Split Freq or Simplex
<b>Transmitter</b>	
Frequency Ranges (450 MHz)	: 330 to 355 MHz 355 to 380 MHz 380 to 400 MHz 400 to 420 MHz 420 to 450 MHz 450 to 480 MHz 480 to 512 MHz 406 to 430 MHz (Canadian Plan)
Frequency Increments	: 6.25 kHz or 5 kHz (Factory Configurable)
Modulation Type	: 4 Level CPFSK
Carrier Power	: 100 mw, 1 Watt, 2 Watt Programmable (+20dBm, + 30dBm, +33 dBm)
Duty Cycle	: 50% (100% with additional heatsinking)
Output Impedance	: 50 Ohms
Frequency Stability	: 1.5 ppm, -30 to + 60 C
Channel Spacing	: 12.5 kHz
Spurious and Harmonics	: -65 dBc
Transmitter Keying	: On Data
Time-out Timer	: 1 to 255 Seconds
Key-up time	: 2 ms

<b>Receiver</b>	
Bandwidth	: 12.5 kHz
Data Performance	: 1x10 <sup>-6</sup> @ -108 dBm
Frequency Ranges (450 MHz)	: 330 to 355 MHz 355 to 380 MHz 380 to 400 MHz 400 to 420 MHz 420 to 450 MHz 450 to 480 MHz 480 to 512 MHz 406 to 430 MHz (Canadian Plan)
Intermodulation Rejection	: -70 dB Minimum
Selectivity	: 55 dB typical at Adjacent Channel (EIA)
Spurious and Image Rejection	: -70 dB
Type	: Double Conversion Superhetrodyne (84 MHz and 450 kHz IF)
Sensitivity	: 12 dB Sinad @ -116 dBm
Frequency Stability	: 1.5 ppm, -30 to +60 C
<b>Interfaces</b>	
Baud Rates Supported at Interface Port	: 1200, 2400, 4800, 9600, 19200, 38400 bps
Data Latency	: < 15 ms typical
Interface	: RS-232 through DB-25 Connector
Over-the-Air Data Rate	: 9600 bps
<b>Power</b>	
Circuit Protector	: 2 Amp board mounted Fuse, Internal Reverse Polarity
Protection	: Diode across primary input
Voltage	: 10 to 30 Vdc through 5.5 mm pin plug or DB-25
TX Supply Current	: 480 ma typical @ 13.8 Vdc with output power set to 2 watts
RX Supply Current	: 75 ma typical @ 13.8 Vdc
<b>Environmental</b>	
Humidity	: 0 to 95% @ 40 C
Temperature Range	: -30 to + 60 C

# Bibliography

- [1] A.S TANEBAUM: *Computer Networks*, Second Edition, Prentice-Hall, Inc., Englewood Cliffs, New Jersey 07632, 1989.
- [2] A.S Tanenbaum: *Computer Networks*, Fourth Edition, Prentice Hall PTR, Upper Saddle River, New Jersey 07458, 2003.
- [3] Doll Dixon R, *Data Communication: facilities, network, and system design*, New York, Wiley, 1978.
- [4] Gruenberg, Elliot Lewis, *Handbook of Telemetry and remote control*, New York, McGraw-Hill, 1967.
- [5] J FitzGerald and Tom S. Eason, *Fundamentals of Data Communications*, John Wiley & Sons, Inc., 1978.
- [6] Kleinrock and Takagi, *Throughput Analysis for Persistent CSMA Systems*, IEEE Transaction on Communications, Vol. Com-33, NO. 7, JULY 1985.
- [7] L.L Peterson & Bruce S. Davie, *Computer Networks*, Second Edition, Morgan Kaufmann, San francisco, CA 94104-3205, 2000.
- [8] M.M Mano and C.R Kime, *Logic and Computer design fundamentals*, Second edition updated, Prentice Hall, Inc., Upper Saddle River, New Jersey 07458, 2001.
- [9] Marco Cantu, *Mastering Delphi 7*, Sybex, Inc., 2002
- [10] Mary, E.S Looms, *Data Communications*, Prentice Hall, Inc., Englewood cliffs, NJ 07632, 1983.
- [11] Paul H. Young, *Electronic Communication Techniques*, Fourth Edition, Prentice Hall, Inc., Upper Saddle River, New Jersey 07458, 1999.

- [12] Ross N. Williams, *A Painless Guide to CRC Error Detection Algorithm*, Version 3.00, 19 August 1993.
- [13] Statnett SF, *Norwegian IEC 870-5-101 User conventions*, Approved version, Revision no. 2.0, [www.IEC.ch](http://www.IEC.ch), March 2000.
- [14] Stephen Morris, *Delphi 5 Made Simple*, Butterworth-Heinemann, 2000
- [15] *Unlicensed Wireless Data Communications, PartI: Defining requirements* [http:// www.ce-mag.com](http://www.ce-mag.com), world wide web,
- [16] William M. Shovdian, *Multiple Priority Distributed Round Robin MAC Protocol for Sattelite ATM*, IEEE, 1998.
- [17] William Stallings: *Data and Computer Networks*, Seventh Edition, Pearson Prentice Hall, inc., Upper Saddle River, New Jersey 07458, 2004.
- [18] William Stallings, *Data and Computer Communications*, Sixth Edition, Prentice Hall, Inc., Upper Saddle River, New Jersey 07458, 2000.
- [19] Wushow Chow, *Computer Communications*, Volume II, Prentice Hall, Inc., Englewood cliffs, NJ 07632, 1985.