Modelling and Simulation of Ethernet Based Networked Mechanical Systems

VAHID SHAKER

A Thesis

In

The Department

Of

Mechanical and Industrial Engineering

Presented in Partial Fulfillment of the Requirements

For the Degree of Master of Applied Science at

Concordia University

Montreal, Quebec, Canada

February 2004

Your file Votre référence
ISBN: 0-612-91111-X
Our file Notre référence
ISBN: 0-612-91111-X

ABSTRACT

Modelling and Simulation of Ethernet Based Networked Mechanical Systems

VAHID SHAKER

Distributed control and simulation based on Ethernet networks has become increasingly common in recent years due to its superior performance and cost over other networking technology. In order to systematically design such distributed systems, which combine Ethernet network components and mechanical systems, a modular modelling and simulation approach for this class of systems is necessary.

In this thesis, an innovative modular modelling and simulation approach is developed to predict the behaviour of distributed mechanical systems based on Ethernet (IEEE 802.3 protocol) networks. The main objective is to predict the overall time delay of transmitted packets and estimate the real-time performance of Ethernet / mechanical systems in different topologies for both normal and abnormal operating conditions. The approach is based on discrete finite state machine (FSM) models developed for the main types of Ethernet network components (NIC, bus, hub, and switch). The FSM models are implemented in C++ and encapsulated using Simulink S-functions. This allows the network models to be easily combined with Simulink mechanical system models that are common in control and simulation applications.

The new approach is tested for a number of case studies to test its capability to predict time delays that arise from network communication. Finally, combined network / mechanical simulations are performed to illustrate the application of the new approach to

distributed simulation problems. Together, these results provide a new approach for simulation of Ethernet based mechanical systems that can be used to design distributed control and simulation systems.

ACKNOWLEGEMENTS

I would like to express my sincere gratitude to Dr. Brandon W. Gordon for his encouragement, support and guidance during the course of this study. His ideas and suggestions were a source of guidance to me and without his timely help this research project would have taken a far longer time to finish. His support and encouragement is highly appreciated.

I want to thank all my family and friends in Montreal and colleagues at Concordia University, for their friendship, their help, and for many valuable discussions and suggestions.

I also wish to thank my lovely wife Raha, for her rich comments, constant encouragement, endless supports, patience and understanding.

This would be incomplete without a word for my family in Iran: without them, I would not have gone this far in my studies.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF VARIABLES

N	Total number of nodes,
N_{node}	Set of Nodes
$M(i)$	The number of message requested at node i
P_{eff}	Efficiency of Network
P_{util}	Utilization of Network
ΔT_1	Preparing the message
ΔT_2	Wait for MAC access
ΔT_3	Latency of transmission
ΔT_4	Post Processing Time
ΔT_{decode}	Decoding Time
ΔT_{code}	Encoding Time
ΔT_{scomp}	Computation Time while Sending
ΔT_{block}	Blocking Time
ΔT_{queue}	Queue Time
ΔT_{prop}	Propagation Delay of Network,
ΔT_{frame}	Total Transmission Time of a Message
ΔT_{dcomp}	Computation Time While Receiving
$T_{avg\text{-}delay}$	Average Time Delay

T_{delay}	Time Delay
$T_{sum\text{-}delay}$	Time used to send messages
T_{tx}	Transmission Time
T_{retx}	Retransmission Time
T_{max}	Total running time
T_{dclock}	Parallel port time delay of clock reading
T_{dtx_NIC}	NIC time delay of Tx
T_{drx_NIC}	NIC time delay of Rx
T_{dtx_p}	Parallel port time delay of Tx
T_{drx_p}	Parallel port time delay of Rx
T_{dround}	Round trip time delay
T_{dclock}	Round trip time delay
T_{hub}	Time delay of hub
T_{switch}	Time delay of switch
K	Spring Constant
b	Damping Constant
M	Mass
Δt	Sampling Period
X_0	Initial Displacement
V_0	Initial Translational Velocity

LIST OF ACRONYMS AND ABBREVIATIONS

AUI	Attachment Unit Interface
BEB	Binary Exponential Back off algorithm
BLAM	Binary Logarithmic Arbitration Method
BT	Bit Time
CAT5	Category 5 balanced cable
CIM	Computer Integrated Manufacturing
CRC	Cyclic Redundancy Check
CSMA/CD	Carrier Sense, Multiple Accesses with Collision Detection
DNS	Domain Name System
DTE	Data Terminal Equipment
FIFO	First IN First Out buffer
FOIRL	Fiber Optic Inter-Repeater Link
FSM	Finite State Machine
FTP	File Transfer Protocol
GUI	Graphic Unit Interface
HOL	Head Of Line blocking
IFG	Inter Frame Gap
IP	Internet Protocol
ISO	International Organization for Standardization
LAN	Local Area Network
LLC	Logical Link Control

MAC	Medium Access Control
MAU	Medium Attachment Unit
MDI	Medium Dependent Interface
MII	Media Independent Interface
NFS	Network File System
NIC	Network Interface Card
NRZ	Non Return to Zero
NTP	Network Time Protocol
OSI	Open System Interconnect model
PHY	Physical Layer entity sub layer
PLS	Physical Signaling sub Layer
PMA	Physical Medium Attachment
PMD	Physical Medium Dependent
PMI	Physical Medium Independent
SFD	Start-of-Frame Delimiter
SMTP	Simple Mail Transfer Protocol
TCP	Transmission Control Protocol
TDMA	Time Division Multiple Access
UDP	User Datagram Protocol
UTP	Unshielded Twisted Pair
VLAN	Virtual Bridged Local Area Network (see IEEE P802.1Q)
VOP	Velocity Of Propagation

CHAPTER 1: INTRODUCTION

1.1 Motivation

The increased use of automation systems that have standardized networking and multimedia functionality is leading to an increasing interest in extending the usage of common office networks like Ethernet for networked automation and simulation systems. Distributed control and simulation based on Ethernet networks has become increasingly common in recent years due to its superior performance and cost over other networking technology [8,9] (See Appendixes K and I for soft and / or hard real-time applications).

Ethernet is an active standard that continues to quickly evolve with greater performance levels, such as higher data rates and lower time response, which in turn improves the performance of associated control and simulation applications. It is therefore likely to be the best alternative for automation and simulation networks in the foreseeable future. This provides substantial motivation to predict the behaviour of complex Ethernet networks combined with mechanical systems. Such simulations can be used to systematically design, test, and optimize this class of systems for both normal and abnormal (fault) operating conditions.

Currently no simulation approach or package is available for simulating systems that combine discrete network components with continuous mechanical system components in a modular framework.

1.2 Problem Definition

In order to systematically design distributed systems, which combine Ethernet network components and mechanical systems, a modular modelling and simulation approach for this class of systems is necessary.

To simulate a combined Ethernet / mechanical system based on Ethernet protocol (IEEE 802.3), we need to model the main components of an interconnection network system, which are Network Interface Card (NIC), both in half and full duplex modes, cables, bus, switch, and hub. We can start with a small network and add or change the components of simulation models as our system grows. This modularity property is an important characteristic for studying design problems where different network topologies and mechanical components need to be evaluated and optimized.

The main objective is to predict the overall time delay of transmitted packets and estimate the real-time performance of Ethernet / mechanical systems in different topologies for both normal and abnormal operating conditions.

Modelling approximations are based on achieving this main goal, so we have tried to keep the essential parameters, which are concerned to this objective and ignore the others to simplify the model.

1.3 Our Approach

Our approach consists of the following steps:

1. Develop Finite State Machine (FSM) models of the main network components.
2. Implement the models using C++ functions.

3. Individually test, run and debug each subsystem model.
4. Encapsulate the codes using Simulink® S-functions.
5. Individually test, run and debug each S-function.
6. Interconnect the blocks in Simulink to arrange the appropriate model.
7. Test and run the interconnected model to finalize the debugging procedure for each part (block).
8. Validating the accuracy of the simulator by comparing the performance measurements obtained from our simulators with experiments and/or previous reported results.
9. Simulating networked mechanical models with the new modular simulator.

Together, these steps provide a new approach for simulation of Ethernet based mechanical systems that can be used to design distributed control and simulation systems. The following chapters will elaborate these steps in detail and provide numerous case studies to test and illustrate the new methodology.

CHAPTER 2: ETHERNET NETWORKS

2.1 Overview of Ethernet

2.1.1 History

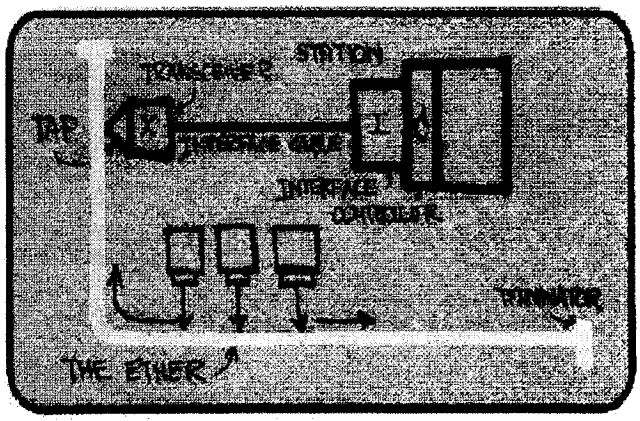


Figure 2-1: Drawing of the original Ethernet system [3]

In 1973, at Xerox Corporation's, researcher Bob (Robert) Metcalfe designed and tested the first Ethernet network, while he was working on a way to link Xerox's "Alto" computer to a printer. Metcalfe developed the physical method of cabling that connected devices on the Ethernet as well as the standards that governed communication on the cable. Xerox Corporation usually receives credit for inventing Ethernet. However, Xerox actually acquired the original technology (then known as Aloha Net) in the 1970s from the University of Hawaii. Xerox then joined with DEC and Intel to develop the earliest Ethernet standard, called Version 1, which was released in 1980. The three companies then released a follow-up standard, Ethernet Version 2, in 1982.

In the mid-1980s, the IEEE 802 committee adopted Ethernet as the 802.3 standard. This standard defines rules for configuring an Ethernet network as well as specifying how elements in an Ethernet network interact with one another. Ethernet has since become the most popular and most widely deployed physical layer LAN technology in the world.

2.1.2 Basics of Ethernet Technology

2.1.2.1 Channel Access Method

Various channel access methods are in use today depending on the network architecture. Ethernet uses a contention-based channel access method called CSMA/CD, which stands for Carrier Sense Multiple Access with Collision Detection. The CSMA/CD is an improvement of the CSMA access control technique. The difference is that in CSMA/CD the station continues to listen to the medium while transmitting. This leads to the following rules for CSMA/CD.

Channel access methods describe the rules used by devices that dictate how the communication medium is accessed, how frames are transmitted, and how the channel is released for use by other devices. Devices using the 1-persistent CSMA/CD channel access method do the following:

1. If the medium is idle, and the period of no carrier has continued for an amount of time that equals or exceeds the Inter Frame Gap (IFG), then transmit the frame immediately. If a station wishes to transmit multiple frames, it must wait for a period equal to the IFG between each frame. It starts to transmit if

the medium is busy, continues to listen until the channel is idle, then starts to transmit immediately.

Note that the IFG is provided to allow a very brief recovery time between frame receptions for the Ethernet interfaces. It is set to 96 Bit Time (BT). (Ethernet cards took this long at most to switch from listening mode to transmit mode.)

2. If there is carrier, then the node continues to listen until the carrier cease (deferring to the passing traffic). As soon as the channel becomes idle, the node repeats the process in step 1.
3. If two or more stations listen for the network traffic, hear none and transmit simultaneously, a collision occurs (Figure 2-2). By the time a collision detected during the transmission, the station will continue to transmit 32 bits of data called the collision enforcement jam signal. If the collision were detected very early, then the station continues sending until it completes the preamble of the frame then it sends the Jam signal. This guarantees that the signal stays on the channel long enough for the all nodes involved in the collision to recognize the collision and react accordingly.
4. After sending the Jam signal, the station waits a period of time chosen with the help of a random number generator and then proceeds to transmit again, starting over at step 1. This process is called back off. The result is collided stations choose different delay times so they will not be likely to collide with

one another again. We talk about this process in detail further in the following sections.

It is the responsibility of the transmitting device to detect and retransmit frames when collisions occur. Collisions are a normal occurrence with Ethernet, but excessive collisions or late collisions are cause for concern. Overloading a segment with too many devices cause excessive collisions as each one is contending for the channel.

Late collisions are those occurring any time after the 64th byte in a frame. Late collisions can be caused by exceeding maximum distance limitations of the media (known as propagation delay) or by hardware failure, such as a faulty transceiver (also referred to as a jabbering device).

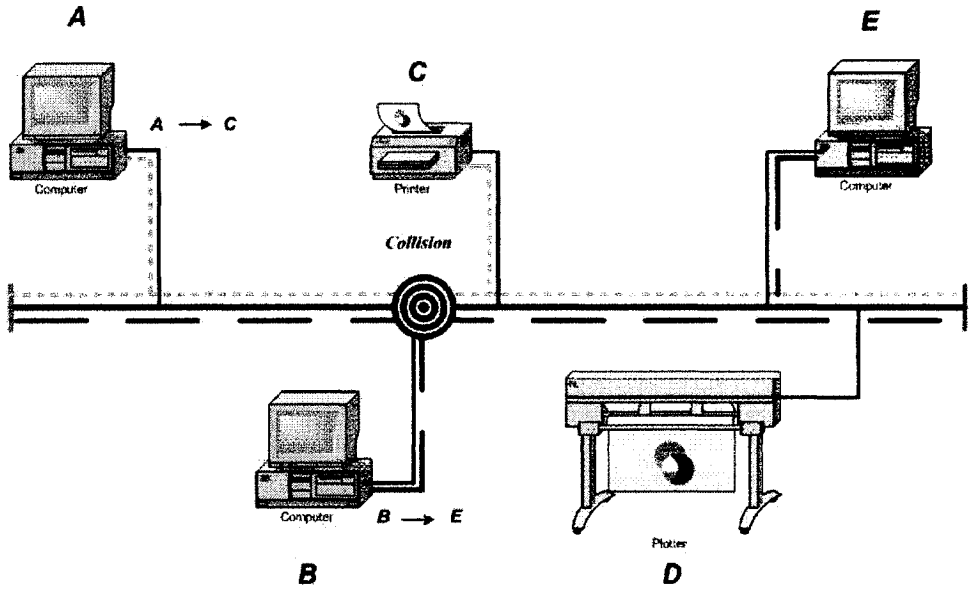


Figure 2-2: Collision when both station A and B transmitted simultaneously

Note that, on a properly functioning 10 and 100 Mbps Ethernet, once a station has transmitted 512 bits of a frame (not including the preamble, also called as the minimum Ethernet frame size and collision window), the station is said to have acquired the

channel and there should not be a collision after acquisition. The 512-bit time value is known as the slot time. In the half-duplex Gigabit Ethernet; this value has been extended to 512 bytes.

The slot time is based on the maximum round trip signal propagation time of an Ethernet system. This time is a combination of two elements:

- **Physical layer round trip propagation time:** This is the time which takes a signal goes from one end of a maximum sized system to the other end and return.
- **Maximum time required by collision enforcement:** This is the required time for collision detection and sending the collision enforcement jam signal.

Adding a few extra bits for a hidden factor to these two elements results in a slot time equal 512 BT. It is a little longer than the actual amount of time to get to one end of a maximum size Ethernet and back. The maximum network cable length (which determines the maximum diameter of the system) and the slot time are tightly coupled.

Length and type of the cabling in use plus the number of devices in the path affect the actual round trip propagation time. This consequently has an influence on the network diameter.

2.1.2.2 Half Duplex Versus Full Duplex

- **Half Duplex:** The term half-duplex communication refers to a mode of transmission in which only one device transmits at a time. Examples of network

topologies that have shared medium are the bus topology, and the star topology interconnected with a hub.

- **Full Duplex:** The Full-duplex mode enables devices to transmit and receive simultaneously without listening across a dedicated link that is collision free. This provides the benefit of increased bandwidth capacity and throughput. Full-duplex capability requires point-to-point connections between devices, such as a switch-to-switch connection. Both devices must support and be configured for full-duplex mode.

2.1.2.3 Ethernet Frames

Four different frame types exist within the realm of Ethernet standards, each designed with a different purpose by a different entity. The four frame types are as follows:

- Ethernet II (DIX)(Figure 2-4)
- Ethernet 802.3 (Novell proprietary)
- IEEE 802.3(Figure 2-3)
- IEEE 802.3 SNAP (Sub Network Access Protocol)

These four frame types can be used in an Ethernet network. The original Ethernet frame known as Ethernet II was developed by DEC, Intel, and Xerox, which is why this frame is sometimes also referred to as DIX. Novell developed its own proprietary frame (Ethernet 802.3) to be used exclusively for IPX/SPX traffic, and the last two frames are developed and named by IEEE.

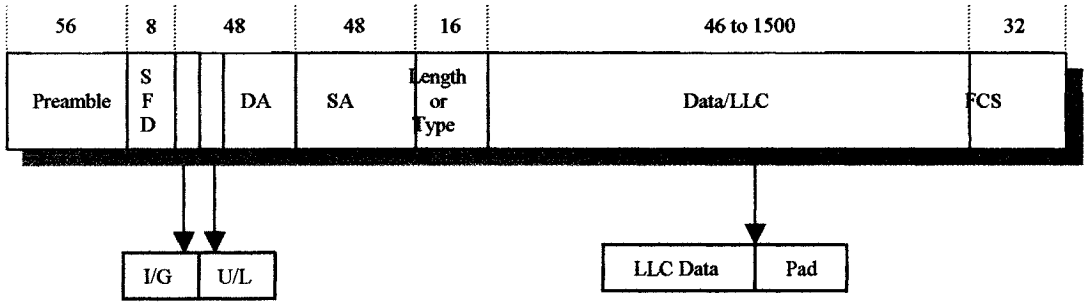


Figure 2-3: Frame IEEE 802.3

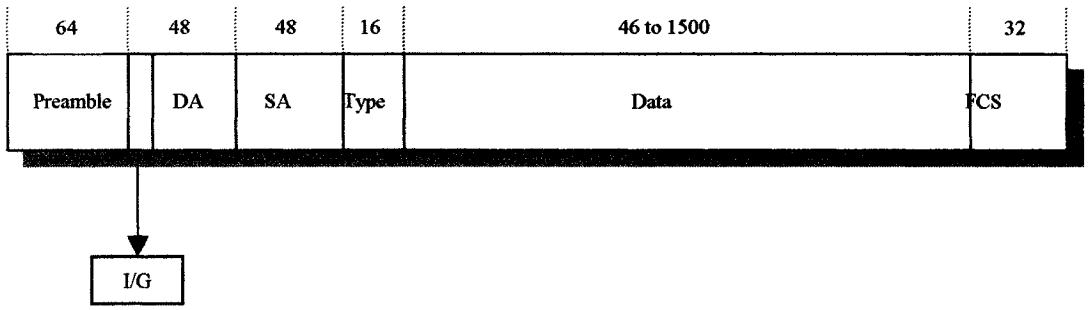


Figure 2-4: Frame DIX

Frames consist of the following fields:

Preamble: A seven-octet pattern of alternating 1s and 0s used by the receiver to establish bit synchronization. It exits to allow the beginning of the frame to lose a few bits due to the signal start up delays as it travels through the 10 Mbps system and protects the rest of the frame from these effects. Both Fast and Gigabit Ethernet use more complex mechanism but the preamble is maintained in them only to provide compatibility with the original Ethernet frame.

Start frame delimiter (SFD): The bit sequence 10101011, which indicates the actual start of the frame and enables the receiver to distinguish the first bit from the rest of the frame.

Destination address (DA): Specifies the station(s) for which the frame is intended. It may be an individual physical address (I), group address (G), a universal (global) address (U) or a local address (L).

Source address (SA): Specifies the station that sent the frame. It is a unique 48-bit physical or hardware number. It is created by the manufacturer of Ethernet interface and is called MAC (Media Access Control) address.

Length: Length of LLC (Logical Link control) header and data field in bytes. (Only for IEEE 802.3 frame)

Type: Ethernet type field for identifying the contents of the data field. (This field is included in the LLC header for IEEE 802.3.)

LLC header: Logical link control header, i.e. IEEE 802.2 protocol.

Data: The data to send (usually an IP datagram). This field has a minimum size and has to be padded if it is shorter.

Frame check sequence (FCS): A 32bit Cyclic Redundancy Check (CRC), based on all fields except preamble, SFD and FCS.

Note that the upper limit for the Data field (or consequently maximum frame size) is used to prevent one transmission from monopolizing the channel for too long. Meanwhile, a four byte long virtual LAN (VLAN) tag header may optionally be inserted in frame between the source address and the Length/Type field to identify the VLAN to which the frame belongs. VLAN are used in switching hubs as a way to direct Ethernet traffic to those ports of the hub, which are defined to be members of a given VLAN. This

extends the maximum size of frame from 1518 bytes (not including the preamble) to 1522 bytes.

2.2 Networking Protocol Models

2.2.1 OSI Model

A set of protocols that would allow any two different systems to communicate regardless of their underlying architecture is called an open system. In order to achieve this goal, ISO has developed its Open Systems Interconnect (OSI) model, which has seven layers, and each layer performs specific functions and communicates with the layers above and below it directly. The following list describes the layers briefly:

1. **Physical Layer:** Transmits physical data, it has two primary aspects: transmission media and connection strategies. The physical protocol can be divided into three specifications:
    I. **Mechanical:** Specifies the pluggable connectors, signal conductors and wiring scheme.
    II. **Electrical:** Specifies the representation of bit values and transmission rates.
    III. **Procedural:** Specifies the sequence of events by which bit streams are exchanged across the physical medium.
2. **Data link Layer:** Provides the reliable transfer of information across the physical link, define frames, detect or correct errors, and flow control.

    The data link layer is divided into two sub-layers, the Media Access Control (MAC) which implements the access of the stations to the network through protocols and Logical Link control (LLC) which is responsible for all the services

from/to the upper layer (Application). So this layer is the one, which receives the most attention in this approach as it describes the Ethernet frame format and MAC protocol.

3. **Network Layer:** Provides the upper layers with independence from the data transmission and switching technologies used to connect the systems; responsible for establishing, maintaining and terminating connections.
4. **Transport Layer:** Determines network, may assemble and reassemble packets.
5. **Session Layer:** Provides the control structure for communication between applications; establishes, manages and terminates connections between cooperating applications.
6. **Presentation Layer:** Translates data formats, encrypts and decrypts data.
7. **Application Layer:** Provides electronic mail, file transfers and other user services.

Every layer encapsulates the data in a protocol. The encapsulation is usually done by adding a header to the data, but one layer, the data link layer, can also add a tail to the data. The physical layer is a little bit different; the protocol instead specifies a set of rules and the physical interface.

Figure 2-5 shows how each layer adds and removes their protocol when application A sends data to application B.

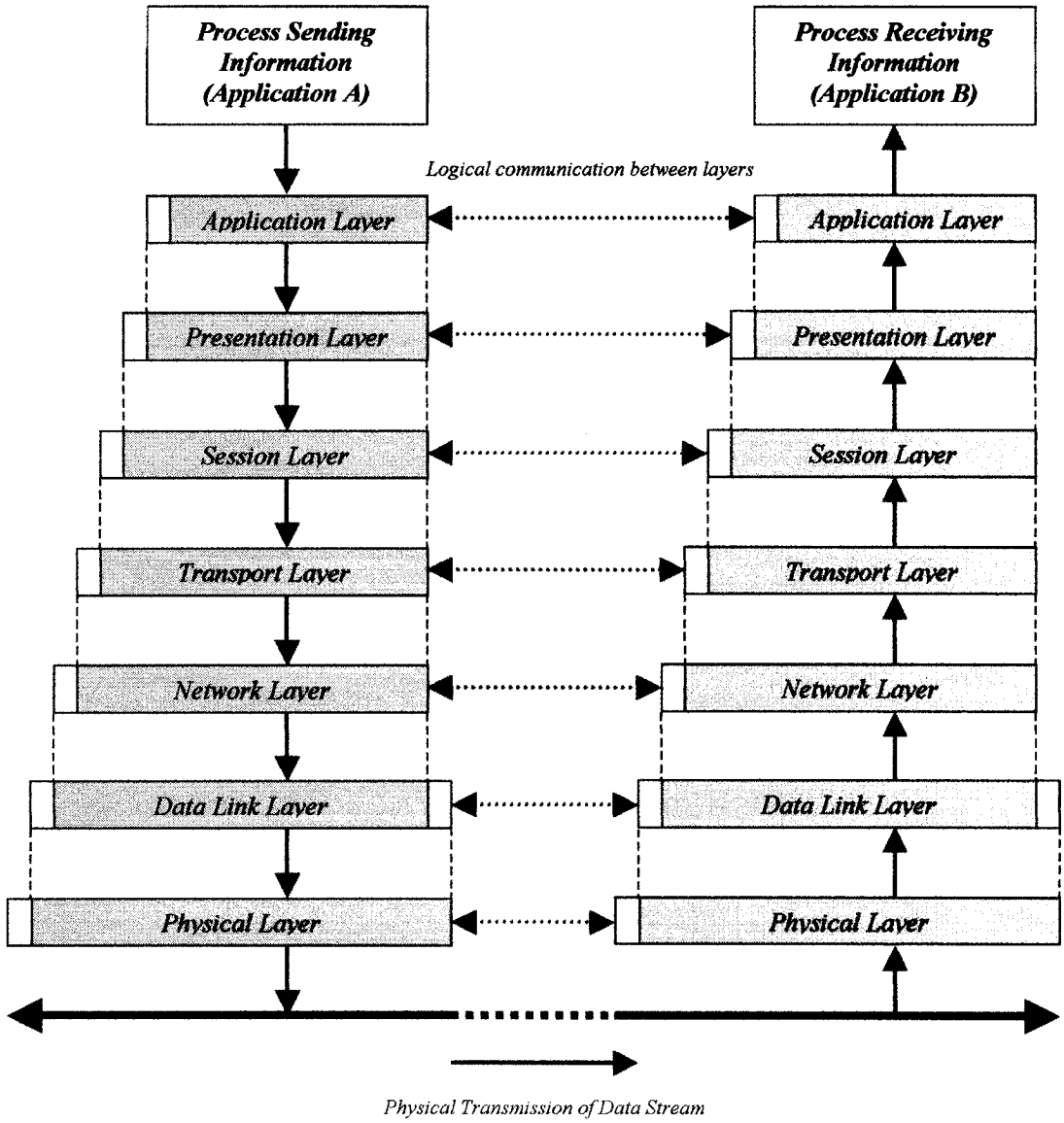


Figure 2-5: Communication using the OSI seven-layer protocol

We can also see in Figure 2-6 that the Ethernet standard concerns itself with elements described in layer 2 and 1 (Appendix F, for more details). LLC control fields are used in all LAN systems and not just in Ethernet and it is not formally part of IEEE 802.3 system specification.

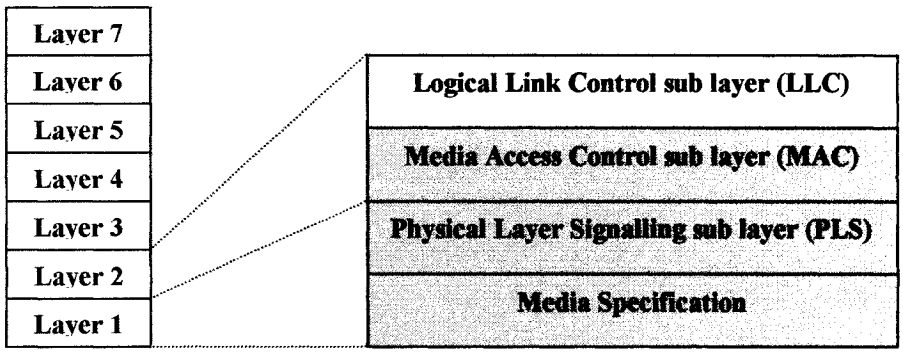


Figure 2-6: OSI layers and the major IEEE sub layers

For the other networking protocol models see Appendix N.

2.3 Network Interconnection

This section briefly describes how local area networks (LAN); wide area networks (WAN) and data terminal equipment (DTE) can be interconnected.

2.3.1 Common Network Topologies

Topology is the way that each node is physically connected to the network. Figures 2-7, 2-8 and 2-9 show how Data Terminal Equipment (DTE) can be connected to each other in a network. Networks of different topologies can be connected to each other using a bridge, a hub, a switch or a router. Common topologies are:

2.3.1.1 Bus

Each node is connected one right after the other along the same backbone (Figure 2-7). Information sent from a node travels along the backbone. All the nodes receive this information but only the destination node process the received data. Each end of a bus network must be terminated with a resistor to keep the signal that is sent by a node across the network from bouncing back when it reaches the end of the cable.

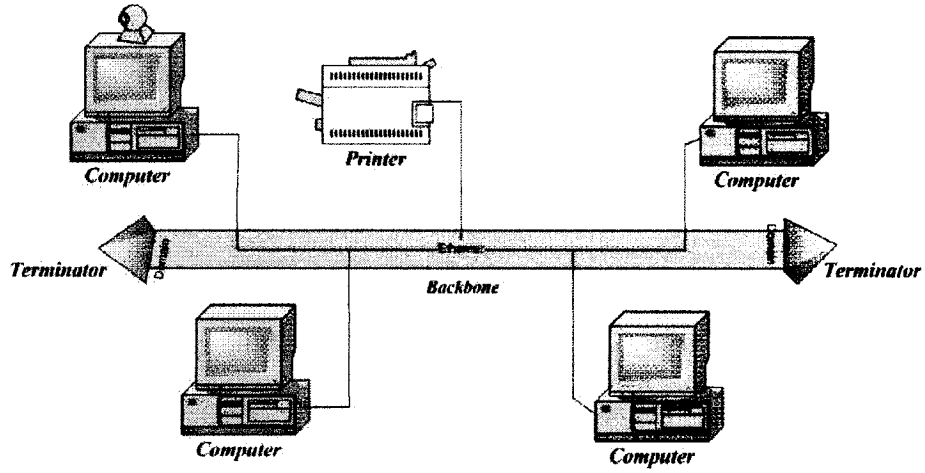


Figure 2-7: Bus topology

2.3.1.2 Star

The star topology is usually interconnected with a central device called a hub or a switch (Figure 2-8). Hub takes a signal that comes from any node and passes it along to all the other nodes in the network. A hub does not perform any type of filtering or routing of the data. It is simply a junction that joins all the different nodes together. You will see more about Hub and Switches in section 2.3.2.5 and 2.3.2.7.

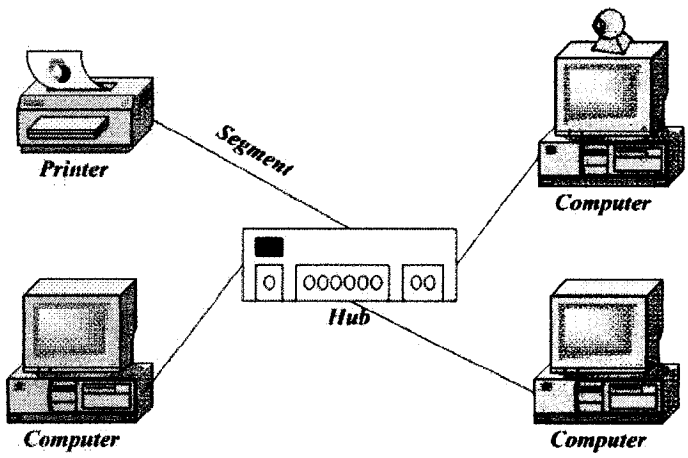


Figure 2-8: Star topology

2.3.1.3 Star Bus

Probably the most common network topology in use today; star bus combines elements of the star and bus topologies to create a versatile network environment. Nodes in particular areas are connected to hubs (creating stars) and the hubs are connected together along the network backbone (like a bus network). Quite often, stars are nested within stars, as it seen in the Figure 2-9:

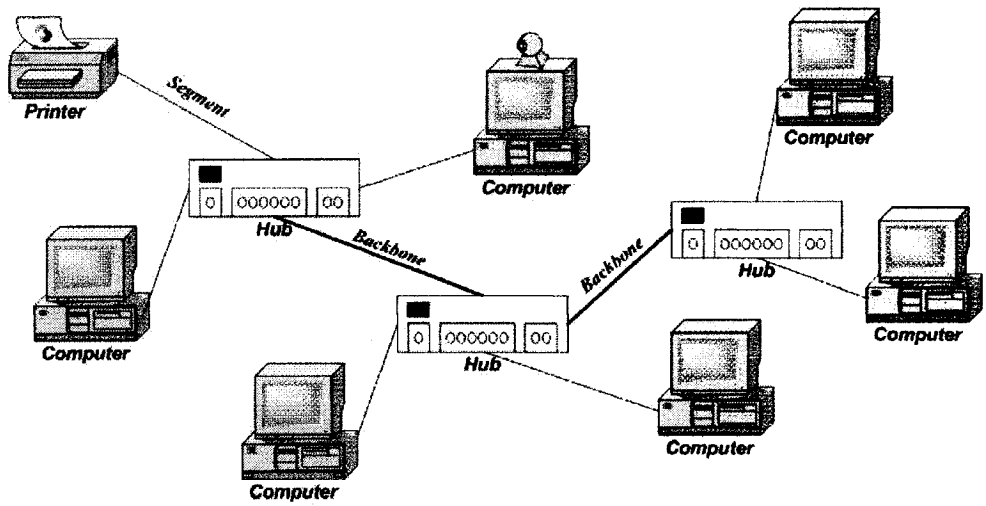


Figure 2-9: Star-Bus topology

See Appendix O for Ring topology.

2.3.2 Ethernet Hardware Components

Although Ethernet is classified as a bus topology, as we discussed before, there are actually several ways to connect devices. So, first the primary possible connection between a PC and an Ethernet segment which is representative of a thick Ethernet (Figure 2-10), more formally known as a 10Base5 media is introduced. We discuss this and other

media specifications shortly. Note that, may other devices also be connected to the cable as well as the PCs.

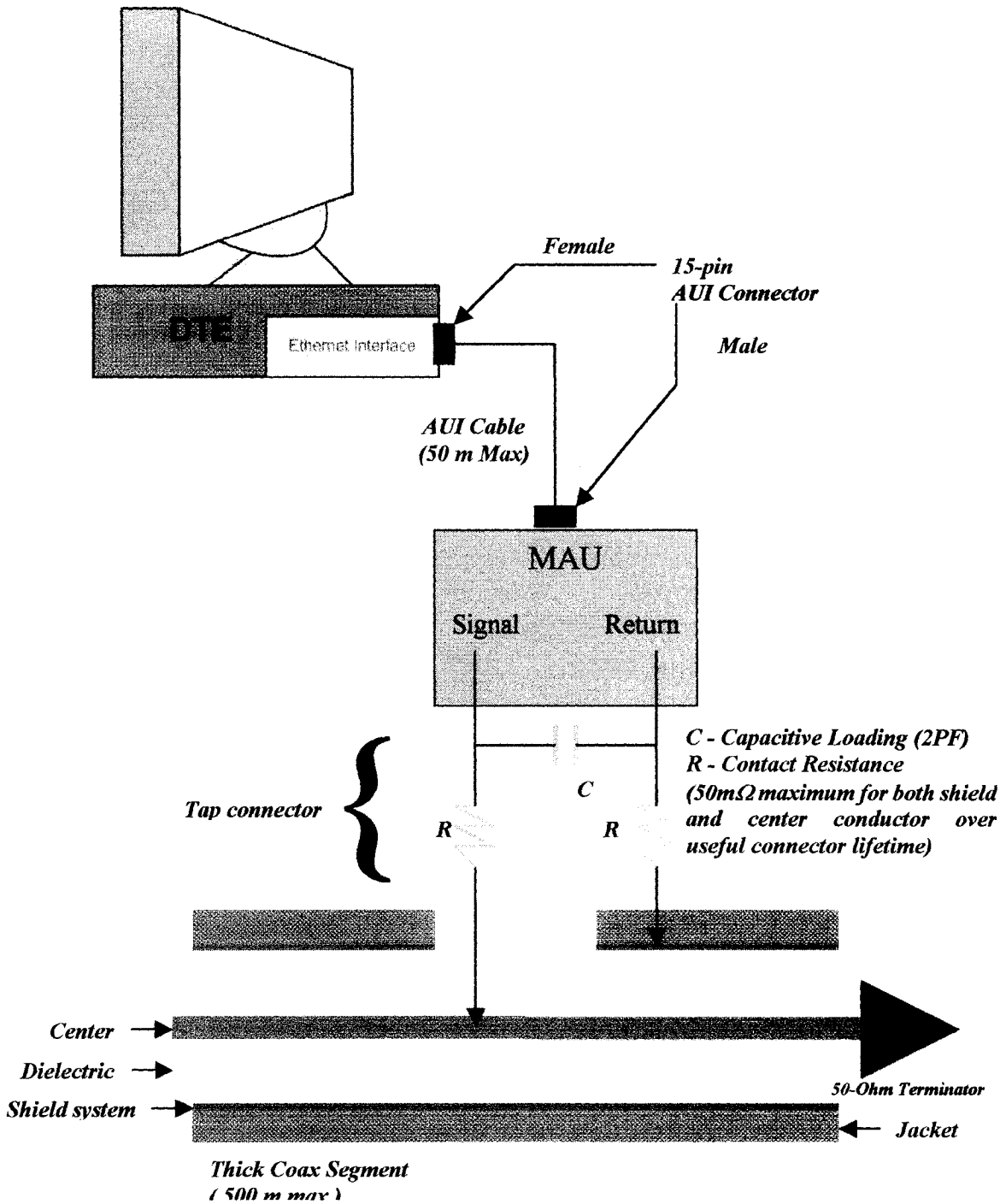


Figure 2-10: Thick Ethernet and typical coaxial tap connection circuit

2.3.2.1 Terminators (Only in Coaxial Cables)

Electronic terminators are placed at the both ends of the cable. They prevent electronic echoing of signals back and forth through the cable, which creates false signals and causes confusion.

2.3.2.2 Transceiver (MAU)

A PC can connect to the cable, but it requires some hardware named "Transceiver" (transmitter/receiver). They are either internal or external. Nowadays most transceivers are internal and embedded on the NIC cards. In the external case (Figure 2-10), when we use coaxial cables it clamps onto the cable using a vampire clamp; a device with a pin that pierces the outer covering of the cable and makes contact with the cable's core.

The transceivers primary function is to create an interface between the PC and the cable. One of its functions is to transmit bits onto the cable using CSMA/CD contention, which allows to determine when there is information moving along the cable and to detect the collisions when they occur. The transceivers communicate with the PC using a transceiver cable. Some call it an Attachment Unit Interface (AUI) cable. The cable consists of five twisted pairs (Figure 2-11). Two are used to send data and control information to the PC. Two more are used for receiving data and control information. The fifth pair can be used to connect to a power source and ground.

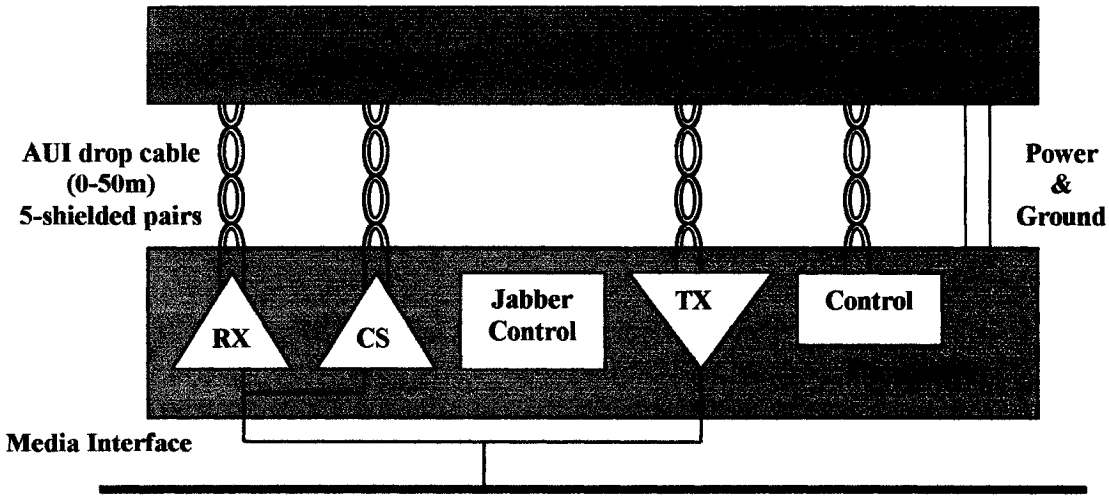


Figure 2-11: AUI and transceiver connection

During transmissions, a transceiver is responsible for encoding the signal on the medium and listening for collisions. If one occurs, the transceiver's internal collision detection circuitry, notifies the network adapter card by sending a signal, causing the adapter to abort its transmission.

A link segment medium, such as twisted pair or fiber optic Ethernet has independent transmitting and receiving data paths. A collision is detected in a link segment transceiver by the simultaneous occurrence of activation on both transmit and receive data paths. On a coaxial cable medium, the transceivers detect a collision by monitoring the average DC signal level on the coax. When the average reaches to a level it triggers the collision detect circuit in the coax transceiver. (For the circuit see Appendix B.)

2.3.2.3 Network Interface Card (NIC)

The transceiver cable connects to the PC through a Network Interface Card (NIC) installed in the PC. The NIC contains the necessary logic to buffer data and move it between the transceiver cable and the PC's memory. It also does error checking, creates frames, determines when to retransmit after collisions occur and recognize frames destined for its PC. In short it performs those functions appropriate for the MAC layer protocol. It also relieves the PC processor from these tasks and allows it to attend to typical PC activities.

Nowadays, an Ethernet interface is contained in a single chip that incorporates the required functions, including the MAC protocol. Ethernet interface chips are designed to keep up with the full rate of the Ethernet system. Various elements have an effect on how many Ethernet frames a given computer can send and/or receive within a specified period of time. These include the speed with which the computer system can respond to signals from the Ethernet interface chip and the amount of available buffer memory for storing frames. The efficiency of the software of interface driver also has an effect. As mentioned before these days, most of the NIC cards have a built in transceivers and appropriate connection depends on the supported data rate by card:

- 10 Mbps: RJ-45 style jack that connects to a twisted pair cable (Figure 2-12 and Table 2-1).
- 100 Mbps (Fast Ethernet): RJ-45 style that connects to a twisted pair cable and/or Duplex SC connector for the fiber optic cables.

- 1000 Mbps (Gigabit Ethernet): RJ-45 twisted pair and/or Duplex SC fibre optic or MT-RJ fibre optic or etc.

Table 2-1: Assignment of signals to connector contacts in RJ-45

Contact	MDI signal
1	TD+
2	TD-
3	RD+
4	Not used by 10BaseT
5	Not used by 10BaseT
6	RD-
7	Not used by 10BaseT
8	Not used by 10BaseT

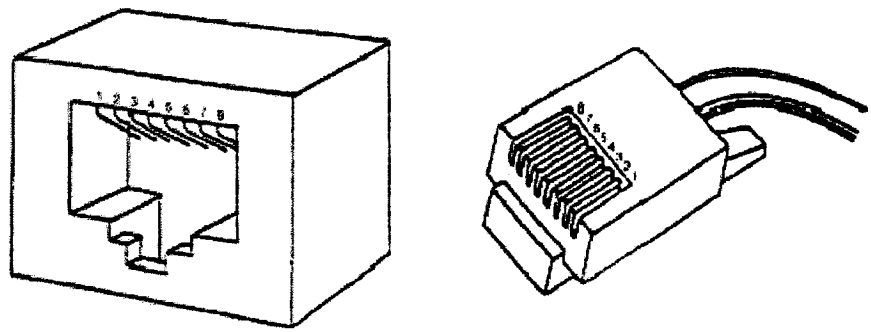


Figure 2-12: MAU-MDI connect (left) and twisted-pair link segment connector (right)[2]

2.3.2.4 Medium

There are three types of transmission media, each with many variations. The first is a conductive metal such as copper or iron (Coaxial cables and twisted pairs). The second medium is a transparent glass or plastic strand that transmits data using light

waves (Fiber optic cables) and the third requires no physical connection at all and relies on electromagnetic waves such as those found in non-cable television and radio broadcasts. The two latter cases are beyond the scope of our discussion and we only stick to the first, which is a conductive metal.

One factor to consider in the selection of media is cost (both the cables and the devices to which they attach). Another is the number of bits each medium can transmit per unit of time. There are two important measures: bit rate and bandwidth. The bit rate is a measure of the number of bits that can be transmitted per unit of time. The typical unit of measure is bits per second (bps).

A given transmission medium can accommodate signals within a given frequency range. The bandwidth is equal to the difference between the highest and the lowest frequencies that maybe transmitted. Sometimes the term bandwidth is used when referring to the number of bits that can be transmitted. Technically, bandwidth and bit rates are different, but there is an important relationship between them. (See Appendix C)

The thick coaxial media system was the first media system specified in the original Ethernet standard of 1980. Coaxial Cables consist of four components. (See Figure 2-10 for details). First is the inter-most conductors, a copper or aluminium wire core, which carries the signal. An insulation layer surrounds the core and prevents the conductor from making contacts with the third layer, typically tightly wound wire mesh (sometime a solid metal). The wire mesh acts as a shield, protecting the core from electromagnetic interferences. The last layer is a plastic protective cover called jacket. The distance between the outer conductor (shield) and inner conductor plus the type of

material used for insulating the inner conductor determine the cable properties or impedance.

These cables have a bandwidth of about 500MHZ and can achieve data rates of up to 500 Mbps. Typical impedances for connections with coaxial cables are 75 ohms for Cable TV, 50 ohms for Ethernet Thinnet (10Base2) and Thicknet (10Base5). Coaxial cable typically transmits information in either a baseband mode or a broadband mode. In baseband mode, the cable's bandwidth is devoted to a single stream of data. Thus the high bandwidth capability allows high data rates over a cable. With Broadband, the bandwidth is divided into ranges. Each range typically carries separate coded information, which allows the transmission of multiple data streams over the same cable simultaneously. The excellent control of the impedance characteristics of the coaxial cable allow higher data rates to be transferred comparing with twisted pair cable.

Today most sites use twisted-pair media for connections to the desktop (Figure 2-13). One of the most common uses of copper is in the twisted pair in which two insulated copper wires are twisted around each other. The number of twists per unit length of the wire can rate such cables. A larger frequency of twists, causes a greater reduction in capacitance and also helps reduce crosstalk; the electromagnetic interference between adjacent pairs. Twisted pairs often are bundled together and wrapped in a protective coating that allows the bundled cable to be buried. The insulation helps to prevent short-circuiting. A common use of twisted pair wire is the transmission of a balanced signal. This means that each wire carries current, but the signals are 180 degree out of phase with each other. The effects on each current from outside electromagnetic sources nearly

cancel each other; resulting in a signal that degrades less rapidly. The twisting helps counteract any electrical capacitance that may build up as the current travels the length of the wire.

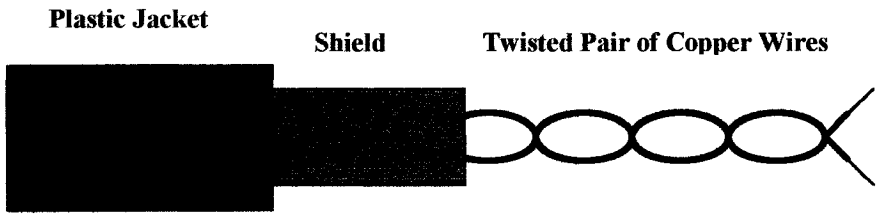


Figure 2-13: Twisted pair wires detail

To further improve noise rejection, a foil or wire braid "shield" is woven around the twisted pairs. This shield can be woven around individual pairs or around a multi-pair conductor (several pairs). Cables with a shield are called shielded twisted pair and are commonly abbreviated STP. Cables without a shield are called unshielded twisted pair or UTP and are used on Ethernet 10BaseT. They can also be used with Token Ring. STP is used with the traditional Token Ring cabling and has a characteristic impedance of 150 ohms. Typical impedance for UTP is 100 ohm for Ethernet 10BaseT media. Both use the RJ line of connectors (RJ45, RJ11, etc.).

Figure 2-14 shows that the most important characteristics of medias can be identified by their abbreviated names.

Thick coaxial segments are still sometimes installed as a backbone segment for interconnecting Ethernet hubs, since thick coaxial media provides a low-cost cable with good electrical shielding that can carry signals relatively long distances between hubs.

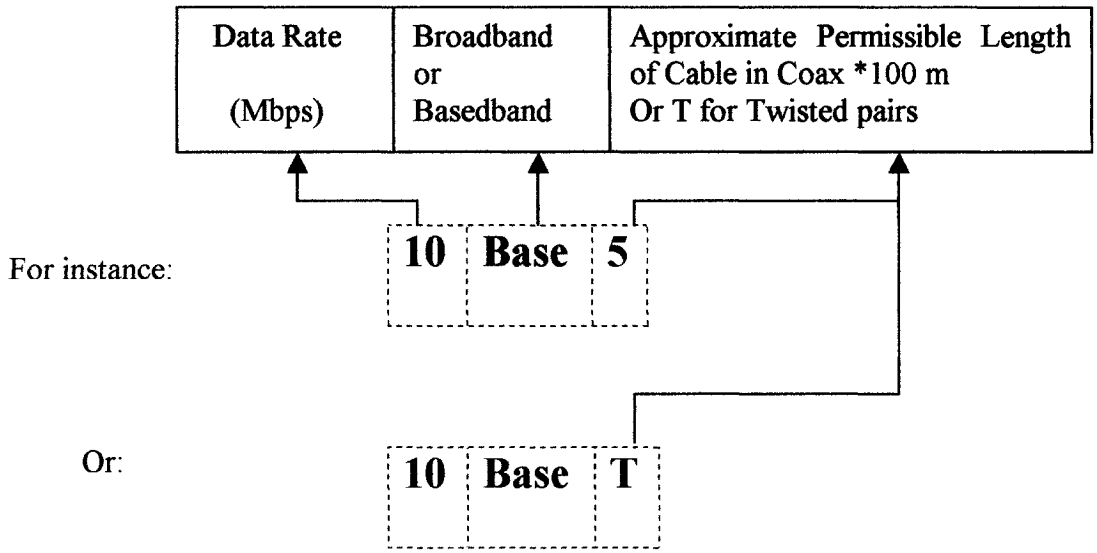


Figure 2-14: Definition of abbreviation of media

On the other hand, thick coaxial cable is limited to carrying 10Mbps signals only, which means we must replace the cable if we wish to link hubs together at higher speeds. High quality twisted-pair cable or fiber optic cable can carry either 10, 100 or 1000Mbps signals, therefore many sites prefer to use these cables as a way of linking hubs together.

Refer to Appendix L to see characteristics of the most popular media.

2.3.2.5 Hub (Concentrator)

Usually, each file server has only one NIC. Therefore, it would be impossible to connect every workstation directly to the file server. To solve this problem LAN may use hubs, which are very common networking devices. Generally speaking, the term hub is used instead of repeater when referring to the device that serves as the center of a network (Figure 2-16).

Ethernet hubs and repeaters (Figure 2-15) operate at the Physical Layer of the OSI model and are defined IEEE 802.3c/d. They amplify and regenerate signals to restore a good level before sending it from one cable segment to another. The distance and the number of attached systems to the segment that a LAN can cover is limited due to attenuation; i.e. the signal weakens as it travels through the network. It is caused by the resistance in the cable, or medium and ultimately would lead to errors in data. By allowing two or more LAN segments to be connected, hub allows the network to span a larger distance. They also provide electrical isolation from failures in the cable or attached systems, protecting equipment on other LAN segments from the effect of the fault.

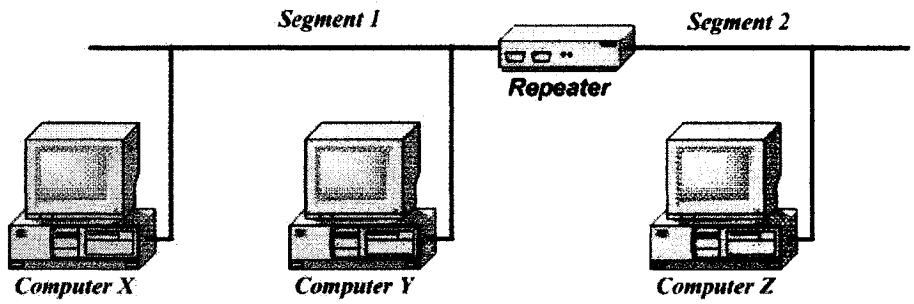


Figure 2-15: Repeater

Hubs or multi port repeaters propagate signal through the networks and are used as network concentration points. They are sometimes also known as "concentrators" or "active star networks". They split the networking media or provide multiple connections. The disadvantage of using a hub is that it cannot filter network traffic. Filtering generally refers to a process or device that screens network traffic for certain characteristics, such as source address, destination address, or protocol, and determines whether to forward or discard that traffic based on the established criteria. The hub is not able to recognize the

addresses in the header of a frame, and therefore is unable to identify which port to send to. Therefore, data arriving at one port gets sent out on all other ports regardless of whether the data needs to go there or not, which is in contrast to a bridge or switch, that only forwards a packet if the destination address of the packet corresponds to a system reachable via the output interface.

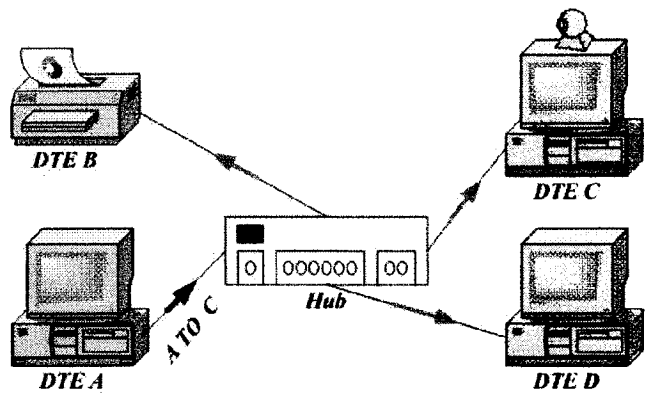


Figure 2-16: Hub or Concentrator

Each port of a hub has an RJ-45 connector able to accept a patch cable to connect to a wall outlet. The other end of the cable is normally connected to a network interface card with a 10BaseT. Many hubs also have an AUI connector to connect an external transceiver to which other types of Ethernet media may be attached or other types of connections like 10Base2. Whatever the type of connector, a single hub is only able to connect a group of equipments operating at the same speed (i.e. all equipment connected to a 10BaseT hub must operate at 10 Mbps).

2.3.2.6 Bridge

A bridge is primarily used for interconnecting two LANs, with the same physical layer and data link layer (Figure 2-17). It makes forwarding decisions on the OSI data

link layer. The entire transmitted frame would be read on LAN A and those whom are addressed to any station on LAN B would be accepted and retransmitted to LAN B and vice versa.

The only problem is to know where the stations are located which can be done by a fixed routing table or using automatic address learning [18].

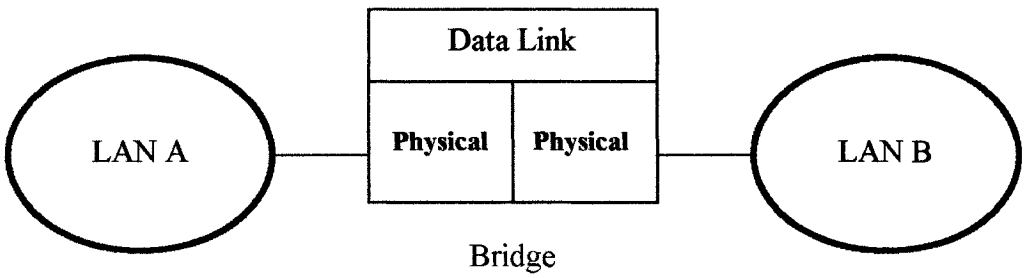


Figure 2-17: Connection of two LAN with a bridge

2.3.2.7 Switch

A bridge with more than two ports is also known as a switch (Figures 2-18a,b). There are important differences between switches and hubs. In particular, in the way they forward frames.

Switched networks replace the shared medium of legacy Ethernet with a dedicated segment for each station. These segments connect to a switch, which acts much like an Ethernet bridge, but can connect many of these single station segments. Some switches today can support hundreds of dedicated segments. Since the only devices on the segments are the switch and the end station, switch picks up every transmission before it reaches another node. It then forwards the frame over the appropriate segment, just like a bridge, but since any segment contains only a single node, the frame only

reaches the intended recipient. This allows many conversations to occur simultaneously on a switched network. Evidently it can effectively double the apparent speed of the network when two nodes are exchanging information. For instance, if the speed of the network is 10 Mbps, then each node can transmit simultaneously at 10 Mbps.

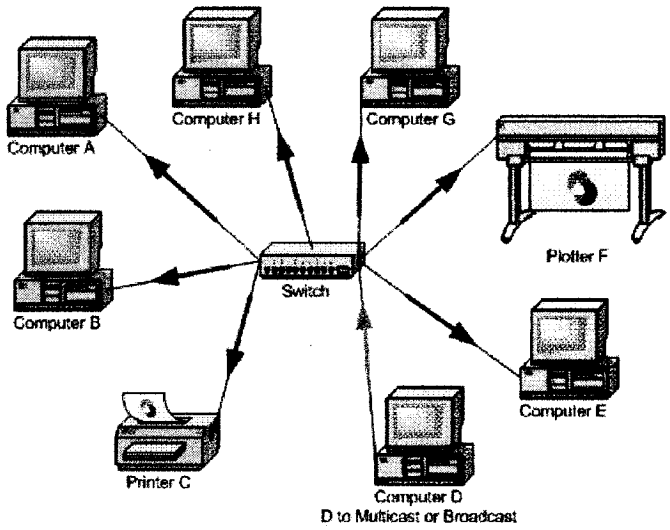


Figure 2-18a: Switch functionality (Multicast or Broadcast)

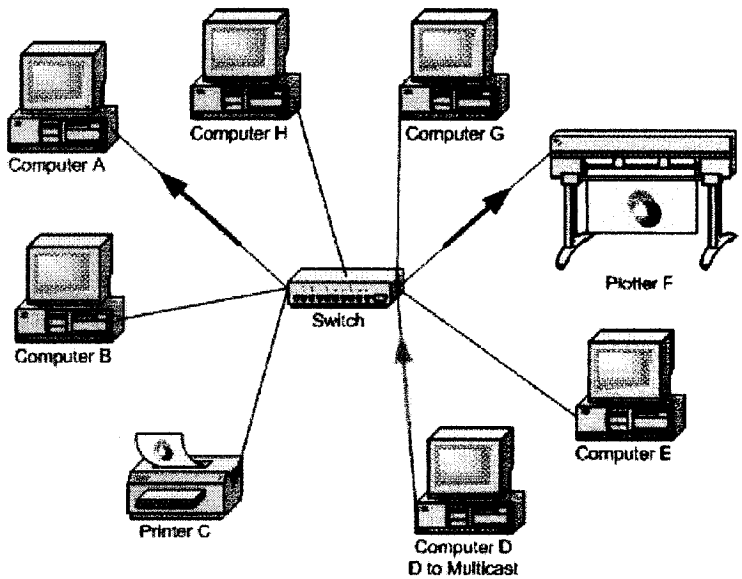


Figure 2-18b: Switch functionality (Multicast)

2.3.2.7.1 Connecting Bridges and Switches Together

There is a special rule controlling the interconnection of bridges and switches (as there is for Ethernet Hubs). The rule says simply that a bridge / switch / hub must form a tree, and not a ring. That is, there must be only one path between any two computers. If more than one parallel path exist, a loop would be formed, resulting in endless circulation of frames over the loop. This, results in overload of the network in a short time. To prevent this happening, the IEEE 802.1D has defined the Spanning Tree Algorithm (STA), which automatically detects loops and disables one of the parallel paths. The Spanning Tree Algorithm may also be used to build fault-tolerant networks, since if the chosen path becomes invalid (e.g. due to a cable / bridge / switch fault) and an alternate path exists, the alternate path is enabled automatically.

2.3.2.7.2 Advantages and Disadvantages

In summary, comparing to shared-media Ethernet, switched Ethernet has following advantages. The Bandwidth of the whole system is equal to the sum of all output ports and network system is also deterministic, as we have no BEB algorithm, which results in a less delay in a high load system. The efficiency of the network is higher. Every port on the switch has it own collision domain. On the other hand, the main disadvantage is, the complexity of administration of the switch and its high built costs. Because most switches works in layer 2, they cannot do congestion control, so they need the protocol in higher layer like TCP to do this task.

2.4 Advantages and Disadvantages of Ethernet

Advantages: Because of the low medium access overhead, Ethernet uses a simple algorithm for operation of the network and has almost no delay at low network loads [19]. No communication bandwidth is used to gain access to the network compared with the token bus or token ring protocol. Ethernet used as a control network commonly uses the 10 Mbps standard and high-speed (100 Mbps or even 1 Gbps) [21,24].

Disadvantages: Ethernet is a non-deterministic protocol and does not support any message prioritization. At high network loads, message collisions are a major problem because they greatly affect data throughput and time delay, which may be unbounded [19], [14]. The Ethernet capture effect existing in the standard Binary Exponential Back off (BEB) algorithm, in which a node transmits packets exclusively for a prolonged time despite other nodes waiting for medium access and causes unfairness, and results in substantial degradation of performance [22]. Based on the BEB algorithm, a message may be discarded after a series of collisions; therefore, end-to-end communication is not guaranteed. Because of the required minimum valid frame size, Ethernet uses a large message size to transmit a small amount of data. (Packets transporting such data have to be extended to 64 Bytes for transmission, so in comparison to many traditional field busses, Ethernet needs a higher raw data rate to achieve a similar effective user data rate [8]).

Several solutions have been proposed for using Ethernet in control applications. For example, every message could be time-stamped before it is sent. This requires clock synchronization, however, which is not easy to accomplish, especially with a network of

this type [24]. As mentioned above, various schemes based on deterministic retransmission delays for the collided packets of a CSMA/CD protocol result in an upper-bounded delay for all the transmitted packets. However, this is achieved at the expense of inferior performance to CSMA/CD at low to moderate channel utilization in terms of delay throughput [20]. Other solutions also try to prioritize CSMA/CD (e.g., Lon Works) to improve the response time of critical packets [25]. Using switched Ethernet by subdividing the network architecture is another way to increase its efficiency [21][14].

The main interest to use Ethernet in the industrial context is to have only one network type, at each level of Computer Integrated Manufacturing (CIM) architecture, and then to facilitate the information system management inside and outside a company, by using the internet (TCP/IP) protocol stack. In addition, nowadays the industrial applications need to exchange information that is more and more complex (such as images, sound, video). Consequently, the size of the technical data flow supported by the network increases and requires more bandwidth. In opposition to the field buses (whose transmission rates are around 1 Mbps), Ethernet networks enable transmission rates up to 10 Gbps. Thus, the non–deterministic characteristic of their protocol becomes secondary. The idea is to consider that the data flow is handled quickly enough, compared with the time-cycle of the industrial applications [15]. (For the real time applications of Ethernet in control systems, see Appendix I.)

CHAPTER 3: MODELLING AND SIMULATION OF ETHERNET NETWORKS

3.1 Overview of Modelling and Simulation Approach

Simulink® is a software package for modelling, simulating, and analyzing dynamical systems. It supports linear and nonlinear systems, continuous, discrete and combination of them. Systems can also be multi rate, for instance, have different parts that are sampled or updated at different rates. For modelling, Simulink provides a Graphical User Interface (GUI) for building models as block diagrams, using click-and-drag mouse operations. Simulink includes a comprehensive block library. We can also customize and create our own blocks.

S-Functions (System-Functions) provide a powerful mechanism for extending the capabilities of Simulink. S-functions allow us to add our own blocks to Simulink models. We can create our blocks in MATLAB®, C, C++, Fortran, or Ada. C, C++, Ada, and Fortran S-functions are compiled as MEX-files using the mex utility. [11][12]

We need to know how S-Functions work if we want to create them and this in return needs understanding of mathematics of a block. In this approach, discrete time simulation is the basis, so in the following we only focus on this mode. Discrete systems can be modeled by the following set of equations:

Output: $y = f_0(t, x_d, u)$

Update: $X_{d+1} = f_u(t, X_d, u)$

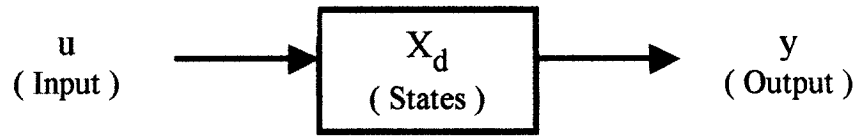


Figure 3-1: Modelling of discrete system

The following flowchart (Figure 3-2) visualizes how the S-function simulation works:

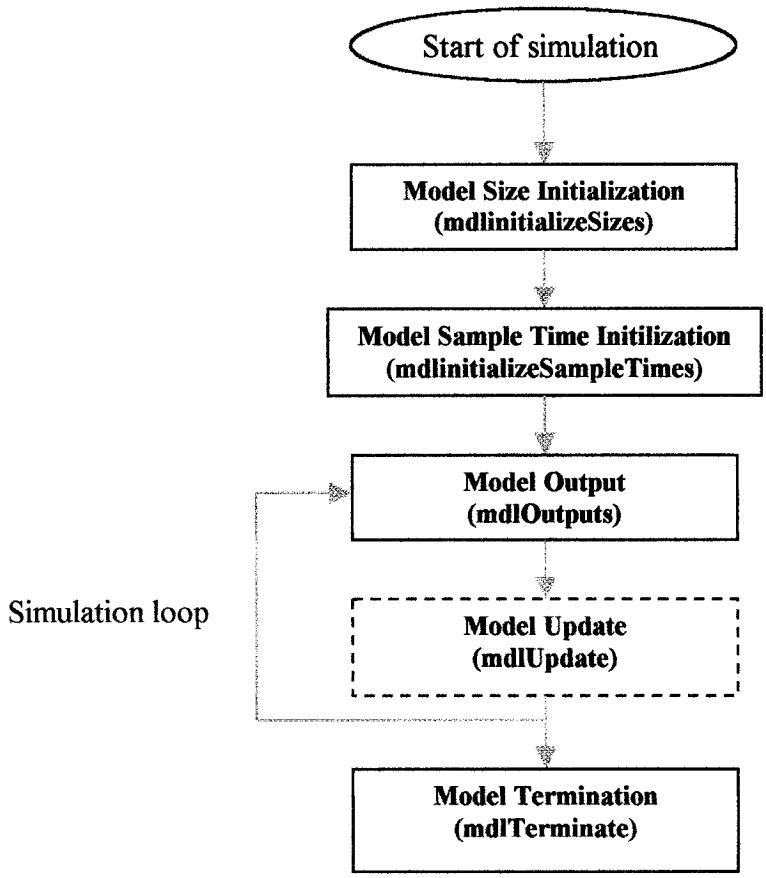


Figure 3-2: How S-Function simulation works

The output portion is placed in mdlOutputs and the update portion in mdlUpdate. Note that, mdlUpdate is an optional function and can be eliminated. To know more about how Simulink interacts with C S-Functions and more detailed flowchart, refer to Appendix D. In that figure, both the above output and update equations correspond to the mdlOutputs and mdlUpdate in the major time step. As our model does not contain

continuous elements, the integration phase is skipped and time is moved forward to the next discrete sample hit.

In order to call a block in Simulink, the sample time should be defined. We can also define an offset that delays each sample hit, which cannot exceed the corresponding sample time. Sample time hit can be determined by the following formula:

Time Hit = (n * Period) + offset Eqn. (3-1)

Where *n* is an integer and it is the current simulation step. The first value for the *n* is always zero. Simulink calls the S-function mdlOutput and mdlUpdate routines at each sample time hit.

S-function comprises a set of S-function call back methods that perform tasks required at each simulation stage. During simulation of model, at each simulation stage, Simulink calls the appropriate methods for each S-function block in the model. Tasks performed by S-function methods include:

- **Initialization:** Prior to the first simulation loop, Simulink initializes the S-function. During this stage, Simulink:
  - ✓ Initializes the SimStruct; a simulation structure that contains information about the S-function.
  - ✓ Sets the number and dimensions of input and output ports.
  - ✓ Sets the block sample time(s).
  - ✓ Allocates storage areas and the sizes of arrays.
- **Calculation of next sample hit:** If we have created a variable sample time block, this stage calculates the time of the next sample hit; it calculates the next step size.

- **Calculation of outputs in the major time step:** After this call is complete; all the output ports of the blocks are valid for the current time step.
- **Update discrete states in the major time step:** In this call, all blocks should perform once-per-time-step activities such as updating discrete states for next time around the simulation loop.
- **Termination:** Performing any necessary actions at the termination of a simulation. For instance, if memory was allocated in the start, this is the place to free it.

3.1.1 Previous Approaches

Ethernet refers to a family of LAN multiple access protocols that vary in details such as bandwidth, collision detection mechanism etc. In this research, we use Ethernet with an un-slotted, 1-persistent, carrier-sense multiple access method with collision detection and binary exponential backoff. Many researchers have worked on Ethernet performance for the past two decades. Many analytical models have been formulated. Some detailed simulation models are presented in [1] [14] [29] [30] [31] [32] [33] [36], which can be used to model Ethernet with different size, transmission rate, Ethernet length and station distribution, etc.

As the CSMA/CD retransmission algorithm is so complex, different assumptions have been seen in different studies. These analytical approaches employ a number of simplifying assumptions. Some have considered small packet size, some constant packet lengths. Some researches are based on finite populations and no buffering to obtain tractable results of Ethernet performance. However, it is not clear how relevant these are

to the actual performance of Ethernet. To have some examples, what Smith and Hain [34] have presented as results by monitoring the stations, shows a big difference comparing to predictions from analytical models. Besides, the maximum achievable throughput with CSMA/CD is 60% [35] while the actual throughput with CSMA/CD can reach to 90% when we have less than 5 hosts and large packet size [27]. Seems that there is no existing analytical model, which is applicable, and sufficient to estimate the real Ethernet performance, hence it is not easy to have an accurate evaluation of performance. Referring to our previous explanations in chapter 1 and above discussion, simulation and/or measurement are necessary to obtain accurate and adequate information on the Ethernet performance.

The affect of packet size and offered load on the link throughput and packet delay have been studied by Gonsalves [28] on operational 10 Mbps Ethernet. Behavior of an Ethernet under varying combinations of packet lengths, network lengths, and number of hosts to show the good capability of Ethernet for high-bandwidth applications, especially when response time is not closely constrained presented by Boggs, Mogul and Kent [27].

Lian, Moyne and Tilbury [14] focused on comparing the performance of three different control systems: 10Mbps Ethernet bus (CSMA/CD), Token passing bus (ControlNet) and Controller area network (CAN) bus (DeviceNet).

Wang and Keshav [1] presented another efficient alternative approach, that they call distributed simulation. In this approach, the medium is passive, not active. Instead, each station on the Ethernet acts as a router, forwarding packets from an incoming link to an outgoing link. An idle station that receives a packet changes its state to busy. If a

packet arrives at a busy station, a collision is detected and the station broadcasts a jam indication to the other stations. In their approach, therefore, the stations cooperate to jointly simulate the medium. This makes the simulation both easy to program and easy to validate. They also simulate a fast Ethernet model only in full-duplex mode.

3.1.2 Our Approach

By using simulation, performance can be easily measured for various Ethernet topologies and system configurations. An event driven simulation model is the standard approach. In such a model, the movement of packets in the model is expressed in terms of events. A global table is maintained to record each event that takes place at a specified time. We distinguish between two types of event-driven simulation models. Most existing simulations of CSMA/CD model the transmission medium as a centralized active entity. In a centralized approach, the medium is simulated by an active entity that keeps track of packets sent by each station, and informs each station about the current state of the medium. The centralized medium also detects and computes the exact time a collision occurs and sends out jam signals. Each station need only model packet transmission and / or retransmission due to collision and packet drop due to buffer overflow. Although these detailed simulation models may achieve accurate performance results, they are too complex.

Determining these times, are non-trivial, because multiple packets can be placed on different parts of an Ethernet nearly simultaneously. Multiple collisions may happen at different places on the Ethernet link simultaneously, from which multiple jam signals are sent out. Indeed, it turns out that to accurately determine these times, the simulation has

to correctly model the electromagnetic propagation of data signals on the medium. This makes it algorithmically complex and hard to correctly implement.

In our approach we accepted the complexity of modelling the electromagnetic propagation of data signals on the medium to achieve a modular and completely similar to what happens in a real Ethernet in the real world. In fact we have accepted parts of both approaches (centralized and distributed) when we designed a half and full duplex NIC based on Wang and Keshav [1] approach but in this case the bus, hub and switch are active entities. The next subsections describe this approach in more details.

Scalability and modularity are the advantages of our approach. We can start small with add the components to the simulators and then step by step grow up the model to reach the ultimate complex goal and monitor the behaviours in this interval of transition. Then we extend the diameter, test the combination of parts in different topologies and configurations, while we change the parameters and having variety of speeds or data rate, sample times and priority in a single model. Using the vast capability of Simulink and particularly its S-functions with the inserted flexibility in the written (developed) codes provide the computational power required to integrate and run complex mechanical models in real-time. In the future work we can also use the external mode and real-time workshop for hard real-time distributed processing.

3.2 Network Interface Card (NIC)

Before we turn to the modelling and simulation, it is necessary to talk about the sequence of activities required for a PC to send data to another PC.

The sending PC executes the networks software that puts a packet of information in the PC's memory. It then signals the NIC via its internal bus that packet is waiting to be sent. The NIC gets the packet and creates the correct frame format, storing the packet in the frame's data field. It then waits for a signal from the transceiver, which is monitoring the segment waiting for a chance to send. When the transceiver detects a quiet cable, it signals the NIC, which then sends the frame to the transceiver. The transceiver transmits the bits onto the cable listening for any collision. If none occurs, it assumes the transmission was successful. If a collision does occur, the transceiver notifies the NIC. The NIC executes the binary exponential back off algorithm to determine when it should try again. If collision continues to occur it will signal to the network software, which will provide the user with an error message or execute some algorithm in response to the error. The transceiver at the receiving end, monitors cable traffic. It copies frames from the cable and routes them to the NIC at the end. The NIC then does a CRC error checking. If there is no error, the NIC checks the destination address in the frame. If it is destined to its PC, the NIC buffers the frame's data (packet) in memory and generates an interrupt, thus inform the PC that a packet has arrived. The PC executes network software and determines whether the packet can be accepted according to the flow control algorithms. If it is true, the PC gets the packet from the memory for further processing. If not, the network software responds according to the protocols at the next higher layer.

3.2.1 Assumptions

A real NIC like the RTL8139C (L)+ incorporates two independent FIFOs for transferring data to/from the system interface and from/to the network. The FIFOs,

providing temporary storage of data freeing the host system from the real-time demands of the network. The way in which the FIFOs are emptied and filled is controlled by the FIFO threshold values in the Transmit Configuration and Receive Configuration registers. These values determine how full or empty the FIFOs must be before the device requests the bus. Additionally, there is a threshold value that determines how full the transmit FIFO must be before beginning transmission. Once the NIC requests the bus, it will attempt to empty or fill the FIFOs as allowed by the respective of some settings in the Transmit Configuration and Receive Configuration registers. [4]

This process has been simplified and two arrays are considered as buffers in this approach. This transmission and receive buffers are discharged at the initialization moment of the simulation. They would be gradually filled upon the data availability to transmit or receive without any threshold.

Due to a fault condition, a station may transmit for a longer period of time than the permissible packet length. We call this "Jabber" and there is a mechanism called "Jabber function" to control this. In this type of modelling, Jabber never happens so we have ignored the control function either. In a non-controlled destination mode, each node chooses its destination randomly within a range.

Digitally encoded data are represented by a sequence of 0s and 1s. There are two popular encoding schemes named "NRZ (Non Return to Zero)" and "Manchester" encoding. In order to synchronize the clocks in both transmitters and receivers, Manchester encoding is invented. Differential Manchester encoding is a variation of this method, which needs even less bandwidth than the first one.

In this model, we simply associates 0 volt for the idle signal, 1 volt for transmit / receive mode and more than one volt is either Jam signal or a collision. Frame size is set to an appropriate and permissible size at the beginning of the simulation so we don't need to pad the data within the process. We also send the frame as a chain stream of bits and each has the following format to carry the minimum necessary information for transmitting and receiving (Figure 3-3):

Figure 3-3: Regular frame format in the proposed model

Regular transmitted bit array has the following elements:

SA: Source address or the sender station number, which is assigned in the model for the each node and works like a MAC address in this approach.

DA: Destination address (Note that, we do not have a group address or multi cast addresses).

Length: The total frame size (Min: 512 bit and Max: 4096 bit) + Preamble.

ID: Each frame has an ID number, starts from one in each node; both ID and SA together, clearly identify a frame.

Voltage: Self describe.

Time: Depends on how statistic acquisition is done this can vary, normally pick the start time that a frame has waited to send.

This pattern could vary for the Jam, generated in NICs as follows:

SA: Jam sender node number.

DA: One more than number of stations in the model.

Length: The length of the jam pattern or 32 bits.

ID: Null or zero.

Voltage: Comes from the following equation:

$$V\_JAM = V\_XMIT + 1.0/(MAX\_NODE\_ETHERNET+1) \quad \text{Eqn. (3-2)}$$

Where: V_XMIT is the transmission voltage.

Time: Time stamping each bit with the simulation time.

This pattern is different in Hubs for the first, third and the last elements (they are all zero) and would be discussed soon. Meanwhile, in an idle case all the elements of a bit array would be 0.

In case of simulation of mechanical systems and data transmission between nodes, there are another field named "Data" which will be attached to the tale of a frame and depends on the type of data, can be more than one element (Figure 3-4):

SA	DA	Length	ID	Voltage	Time	Data

Figure 3-4: Frame format in the proposed mechanical model

There is an optional capability called frame bursting to improve performance and efficiency for the short frames in the half duplex gigabit Ethernet. This makes possible sending more than one frame for a limited duration of time by pretending the channel busy with a special signal while we are at the IFG process. This duration is equal to 65,536 BT plus the final frame, which sets the limit maximum burst transmission time. So far we have ignored this option but it is simply possible to modify the model and to apply it as well [3].

Binary logarithmic Arbitration Method (BLAM) mechanism could be a cure for "channel capture" phenomenon but it is still non-standardize so we have decided not to model it in this research [26].

A station at one end of a full duplex link can request the station at the other end of the link to stop transmission for a period of time by using MAC Control frames to send PAUSE requests [3]. We did not apply this in our approach and we discuss the reason more in detail in section 3.5.1. Though we have warning messages for the filled out receiving buffers in NICs, but we always choose the buffer size parameter somehow to never face it in simulation.

3.2.2 Model Description

As mentioned in section 2.1.2.2, we have two mode in Ethernet operation; half and full duplex which have their own model to describe:

3.2.2.1 Half Duplex

We model half duplex CSMA/CD using the NIC state diagram shown in Figure 3-5 for the Bus topology and Figure 3-6 for the Star topology which are inspired from [1] (see Appendix A). Each simulated station is responsible for the tasks such as packet transmission, retransmission, collision detection and signalling.

A simulated station can be in one of the seven states and/or two possible actions:

- STATES:

  State 1: Idle

  State 2: Sending

  State 3: Receiving

State 4: Wait for back off end and Jam End

State 5: Wait for Jam End

State 6: Wait for Back off End

State 7: Receiving and Waiting for Back off End

- ACTIONS:

  Action 1: Send Jam

  Action 2: Back off

- EVENTS:

  Each listed events below switches one state to another:

  Event 1: Data available to send

  Event 2: End of transmission (No data available to send)

  Event 3: End of Jam Signal

  Event 4: Receiving Jam Signal

  Event 5: Receiving data on the input port

  Event 6: End of receiving data (By sensing the idle signal on the input port)

  Event 7: End of Back off

  Event 8: No receiving buffer available, which causes the error signal

  Event 9: Receiving data from another source in twisted pair cables or detect a collision by sensing higher voltage (U (4) >= V_CD) in the coaxial cables.

There are some minor differences in events between a model with Bus topology (Using Coaxial Cables; in this model: 10Base5 media)(Figure 3-5), and Star topology (Using hub and twisted pair cables; in this model 10BaseT media or higher)(Figure 3-6).

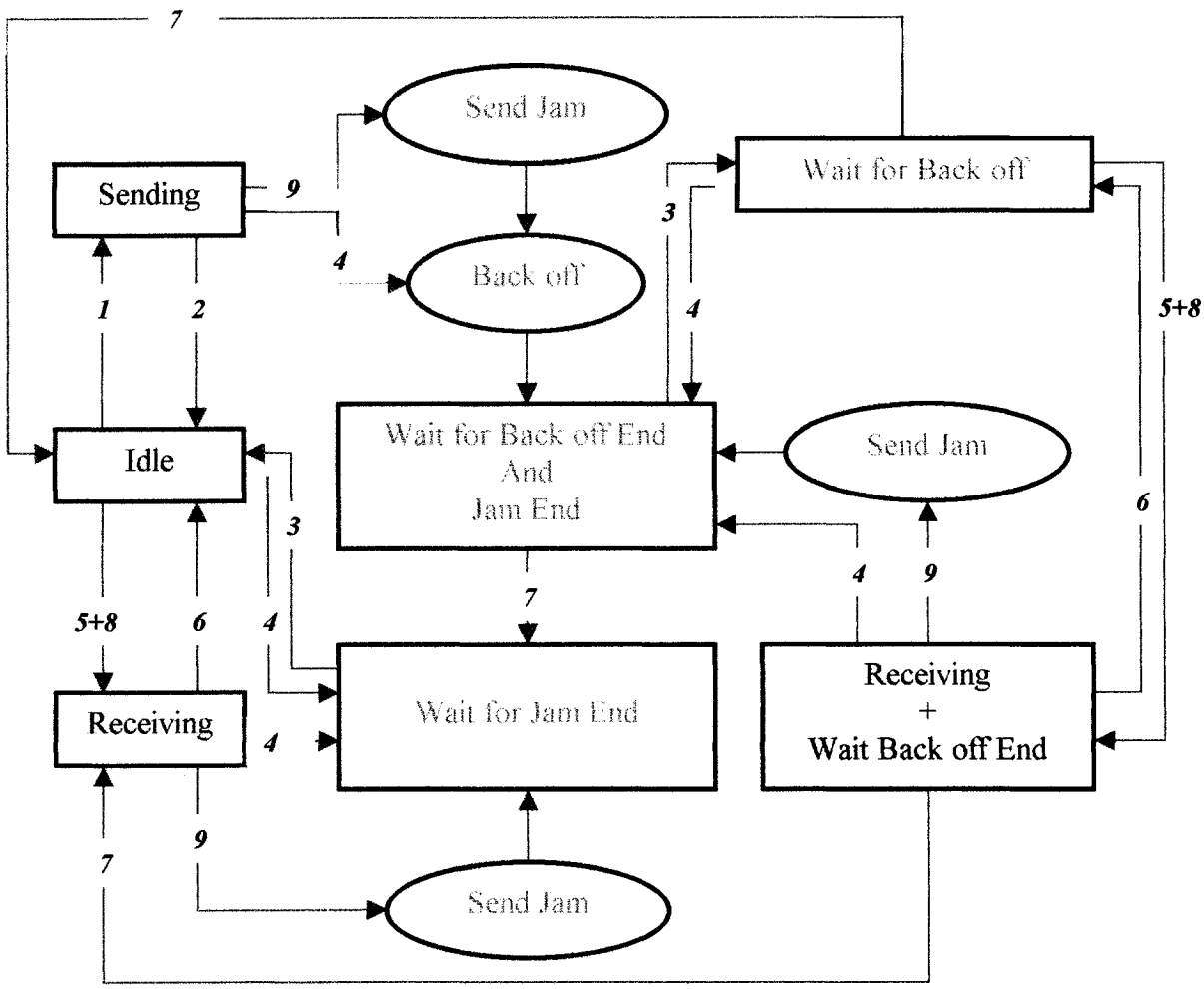


Figure 3-5: State diagram for the model of Bus topology with coaxial cable

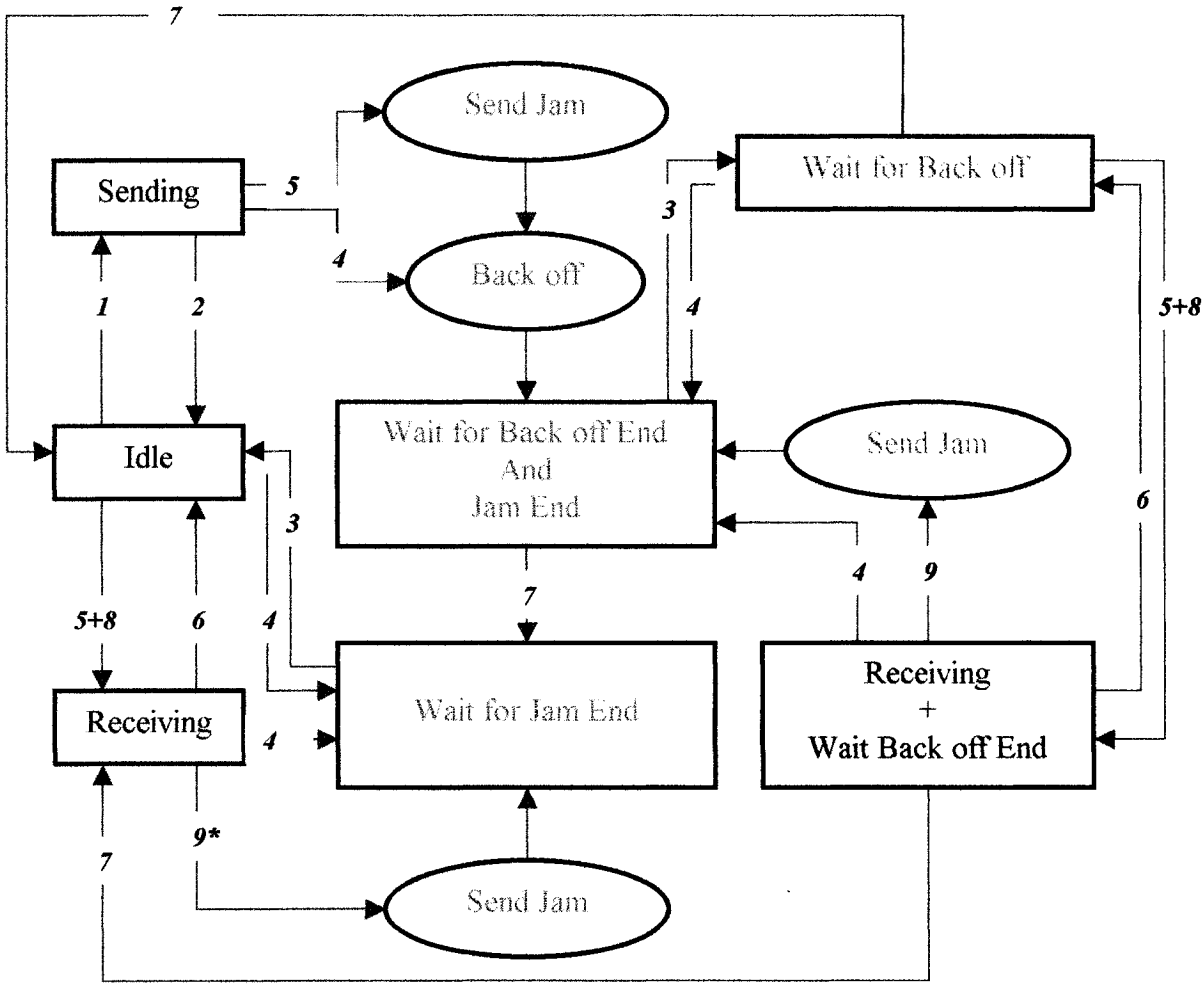


Figure 3-6: State diagram for the model of Star topology with twisted pair cable

While we use HUB (in case of twisted pair as medium) we trap the collision in HUB, thus we never need to have the send Jam action but in case of considering flexibility for the future modification we still keep it in the model.

Both topologies use the same BEB algorithm to retransmit. The following flow chart (Figure 3-7) illustrates how it works in our modeling:

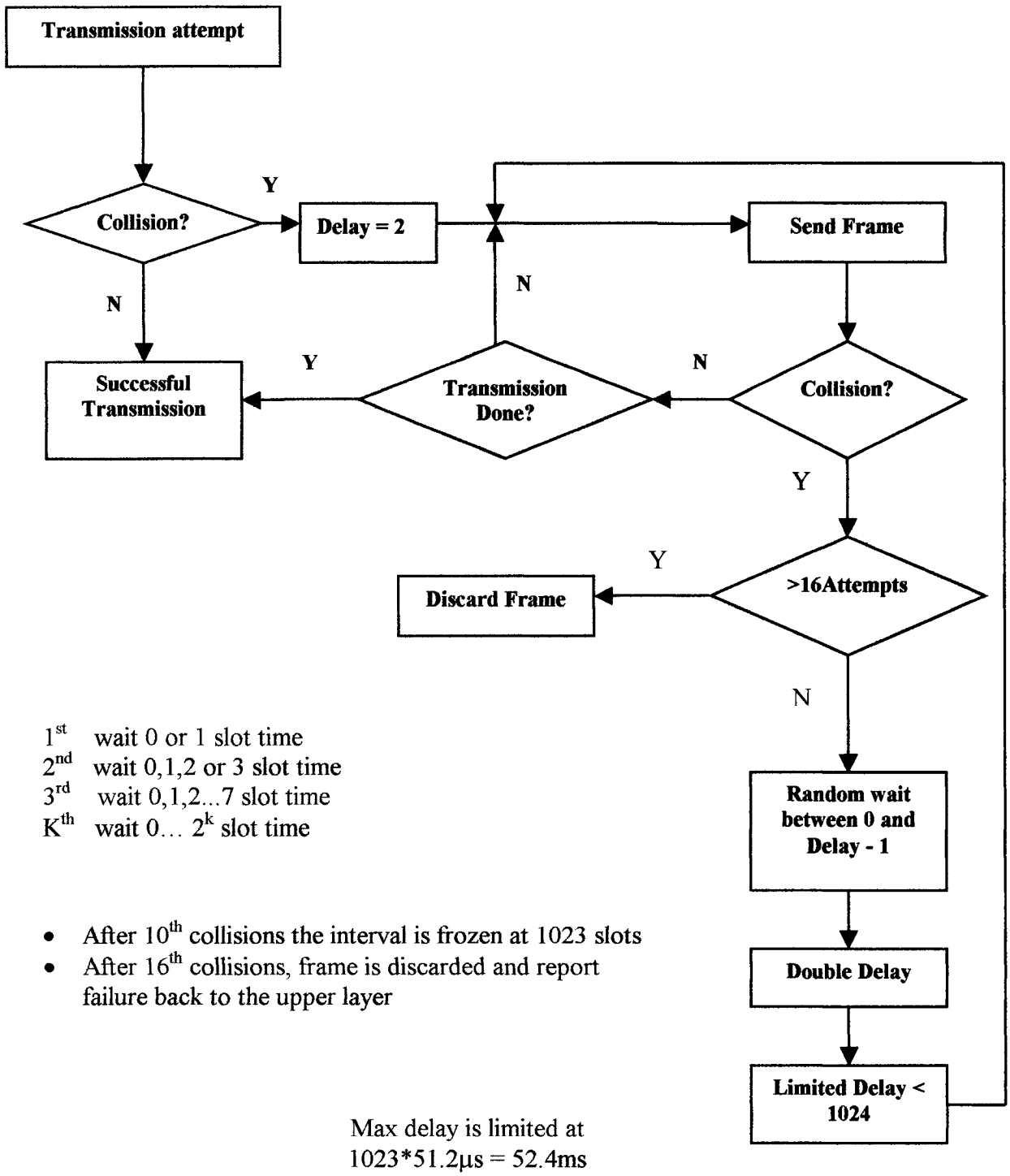


Figure 3-7: Truncated Binary Exponential Back-off (BEB) algorithm

3.2.2.2 Full Duplex

Figure 3-8 shows the FSM, designed in this approach, which has the following states and events:

States:

State 1: Idle

State 2: Sending

State 3: Receiving

State 4: Sending and Receiving

Events:

Event 1: Data available to send

Event 2: End of transmission (no data available to send)

Event 3: Data available to send while we are receiving data on the input port

Event 4: End of transmission and receiving data

Event 5: Receiving data on the input port

Event 6: End of receiving data (sensing the idle signal on the input port)

Event 7 = Event 5

Event 8 = Event 6

Event 9 = Event 2

Event 10 = Event 1

(Note that in the code: event 7 stands for the no receiving buffer available and causes the error signal)

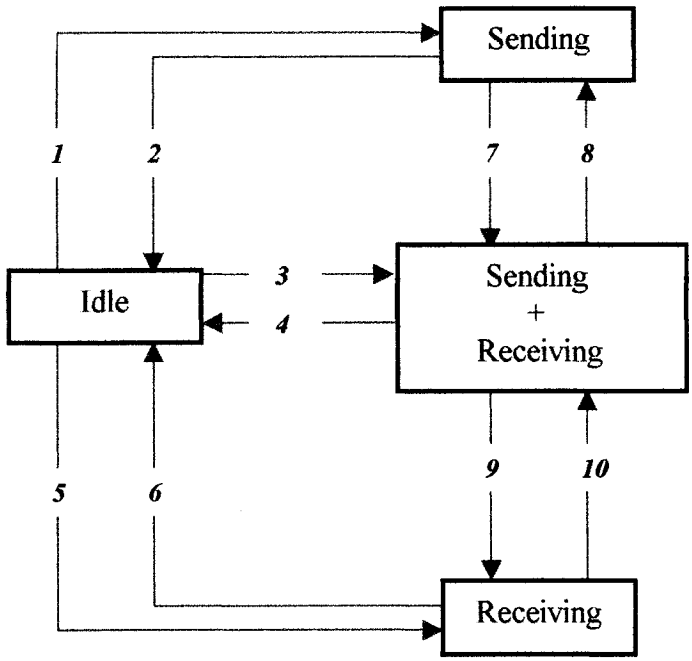


Figure 3-8: State diagram for the full duplex NIC in the model

3.2.3 Simulation Description

3.2.3.1 Half Duplex

Assume the simulated LAN has a topology as shown in Figures 3-5 and / or 3-6. Simulation starts in State 1 or idle mode of the station. By the time a station check the states in the update states function, it moves to the appropriate state, which is State 1 at the initialization point. At this time, we check the events and the following scenarios could happen:

- If there is data available in transmit buffer to send and the medium voltage is zero, it switches to State 2 for sending.
- Receiving any jam signal, switches the state from idle to wait for the end of this signal or State 5.

- Receiving any data with the matched destination address, switches the state to the receiving mode or State 3.

In each cycle of simulation, we update the states of each station and check the voltage (listen to the channel) to react on any changes.

While a station is sending data, two cases could cause changing the situation; if jam signal detected on the input port, NIC takes the back off action and then it waits for the end of receiving jam and the current action. On the other hand, detecting either a collision in case of using coaxial cable as medium or receiving data in case of twisted pair cables as medium, causes the send jam action and consequently go to the back off and wait for both to end (State 4). Two options are still available to face:

- By the end of back off duration, it goes to the State 5 and waits for the end of Jam signal. Then, there is no way but moving to idle state again.
- By the end of either receiving or sending jam signals, NIC has to wait for the end of back off (State 6). End of Back off, causes an idle state, and in case of receiving any data (in twisted pairs) or facing with another collision (in Coaxial cables) switches back to State 4; the above-mentioned process would be repeated again.

While the NIC is waiting for the Back off to end (State 6), it can also receive data simultaneously (State 7). In this case either having a true Event 9 or receiving jam signals move the state to 4. It can also change the state from 7 to 3 as soon as the back off ends.

3.2.3.2 Full Duplex

It is not as complicated as the half duplex since we do not have any actions in process. So, as it is illustrated in Figure 3-8, whenever we have data in buffer to send, there is a transmission and having enough buffer to receive is a guarantee to get data if there are any. Two separate lines for transmission and/or receiving enable us to have the State 4.

3.2.4 Simulation and Testing

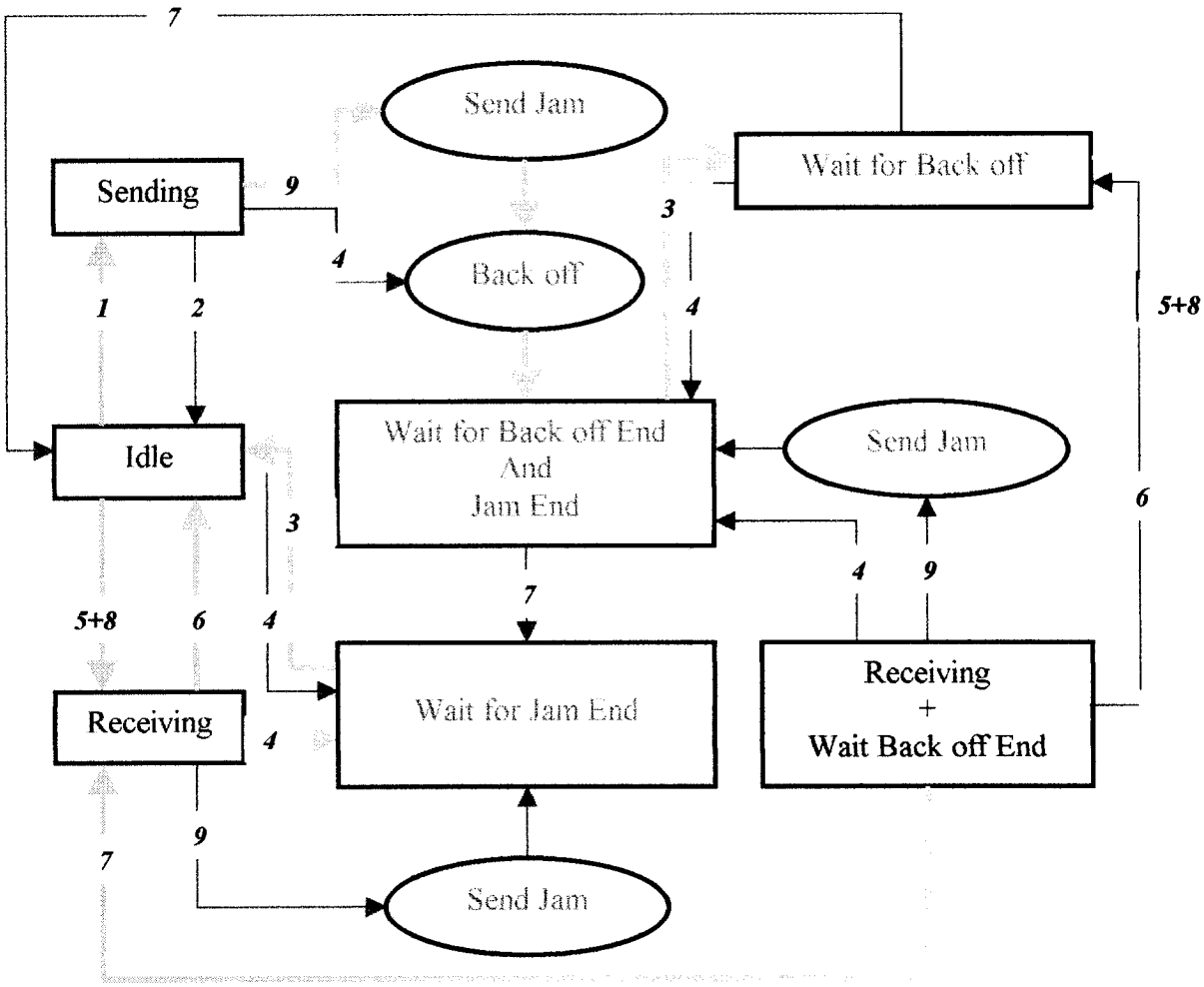


Figure 3-9: Illustration of paths followed in the simulation test

It was too complicated to test and confirm the functionality of NIC model and after several times tests and run the codes under different conditions, using log files and output graphs, it is now obvious that the systems works properly. There are some ways to see if the NIC has the whole demanded functionality. The following method would be able to demonstrate functionality of a NIC with simulation of the input and show how the NIC, switches from one state to another (Figure 3-12). Instead of having the whole system, like number of NICs and the BUS, we have only feeder block connected to the target NIC that works like the rest of system and generate the necessary signals (Figure 3-11) for the test. The feeder conducts the target NIC through the above-illustrated path (Figure 3-9). In this simulation, we have a NIC works in half duplex mode where 10Base5 (coaxial cable) is the media (Figure 3-10). It can easily be seen how the NIC responds to signals coming through the BUS (in this test, feeder) accordingly (Figure 3-11).

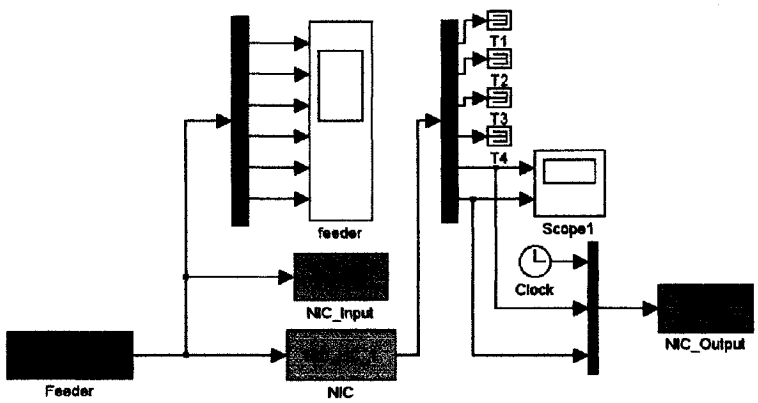


Figure 3-10: Simulation configuration used to test the NIC

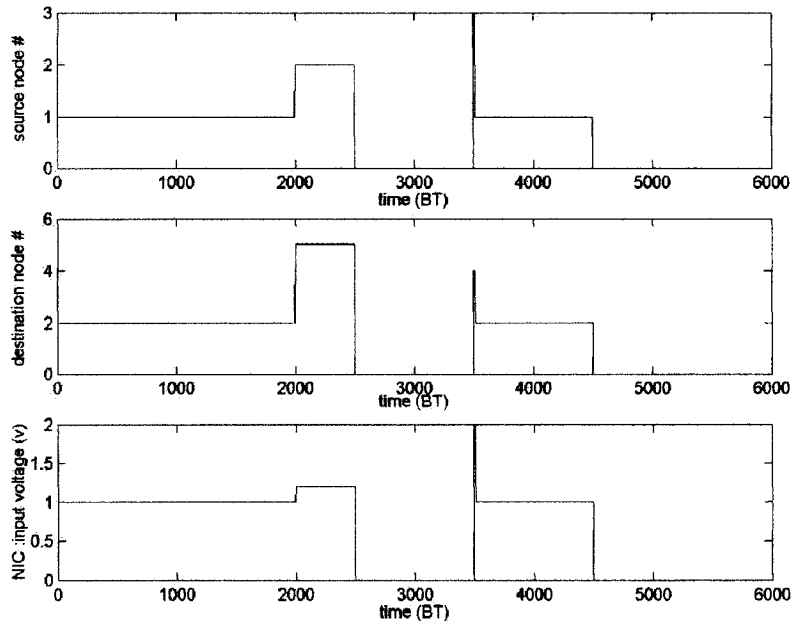


Figure 3-11: Feeder input used to test the NIC

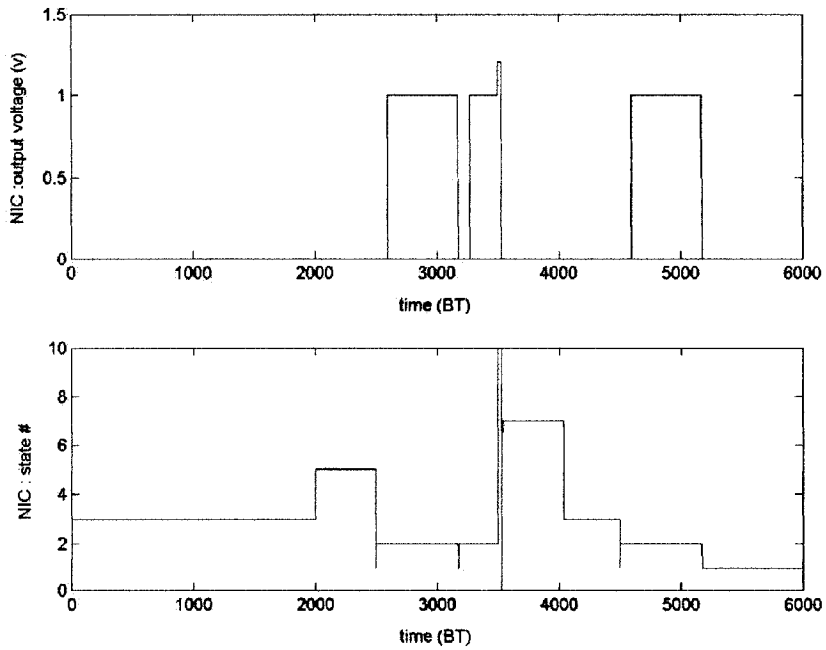


Figure 3-12: NIC output voltage and states

Note that, state 10 in Figure 3-12, means that we are in Jam process action. We do not have this state in our state diagrams and it is only for test purpose.

3.3 Bus

3.3.1 Assumptions

In this research, 10Base5 (Coaxial Cable) characteristics are the basis of modelling the propagation delay in medium. We also assumed that the Bus length is so short to ignore the different distance between the connected nodes to the Bus. So they all use the same distance and propagation delay to reach each other. This approach is close to the star topology idea. For the future job, by developing sort of two-dimension table addressing, each node would be able to look for the appropriate propagation delay and set up a variable buffer size to get its destination.

3.3.2 Model Description

Absence of carrier in Bus keeps the state as idle and presence of more than one signal from different nodes causes the collision as it is seen in designed FSM in this approach (Figure 3-13).

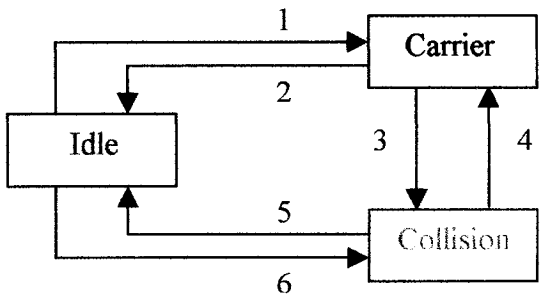


Figure 3-13: Bus state diagram

FSM (Figure 3-13) for Bus has three states and six events as follow:

States:

State 1: Idle

State 2: Carrier

State 3: Collision

Events:

Event 1: One NIC transmit data

Event 2: No transmission

Event 3: More than one NIC transmit data

Event 4: Event 1

Event 5: Event 2

Event 6: Event 3

3.3.3 Simulation Description

Assume a simulation LAN illustrated in the schematic in Figure 3-14 (bus fabric designed in this approach) with four nodes connected in a bus topology. Consider the Bus as a block in simulation with multiple input port and only one output port. Each station sends in its data, then all the fifth elements of each bit (Voltage) will be replaced with the grand total of all voltages. Afterwards, the new bit and/or bits (in case of collision) go through the FIFOs buffer with the new value of their fifth elements.

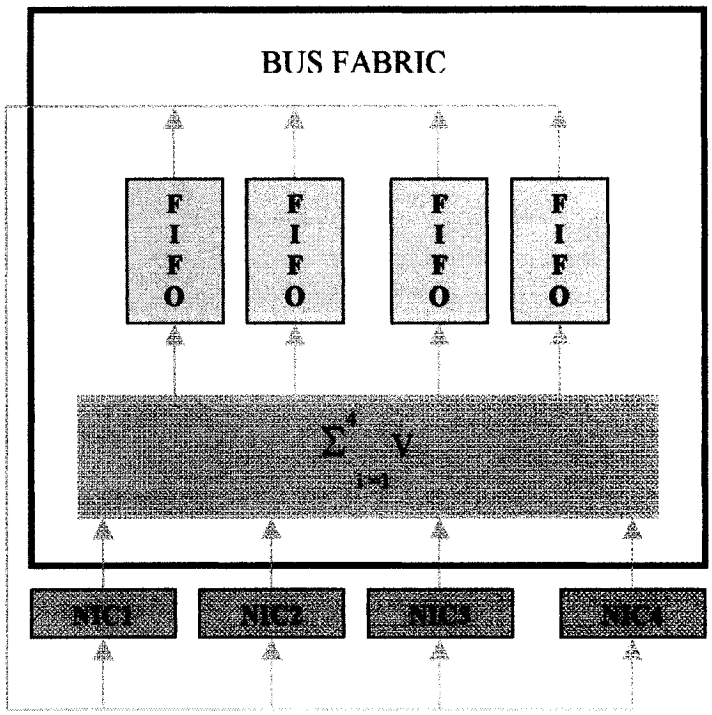


Figure 3-14: Bus fabric of the model

3.3.4 Simulation and Testing

Feeders (developed codes to generate signals) are the appropriate leverage to test the S-Functions. Figures 3-15, 16, 17, clearly show the process.

It can be seen that collision happens when both, feeder and feeder1, are active for the 1040-1140 time- intervals and the voltage raise in this case. (Note that in this simulation, feeder2 works as a NULL generator.)

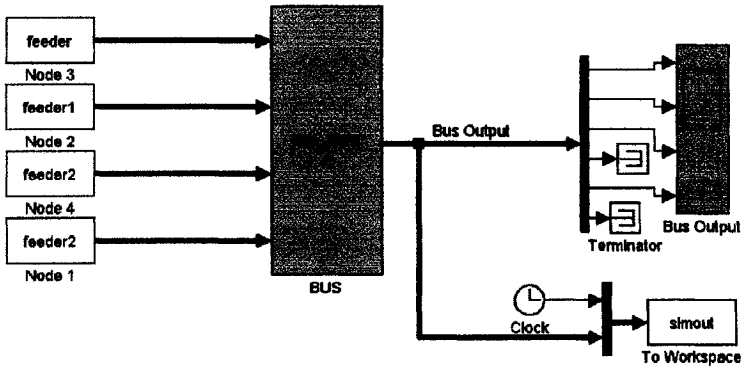


Figure 3-15: Simulation configuration used to test the BUS

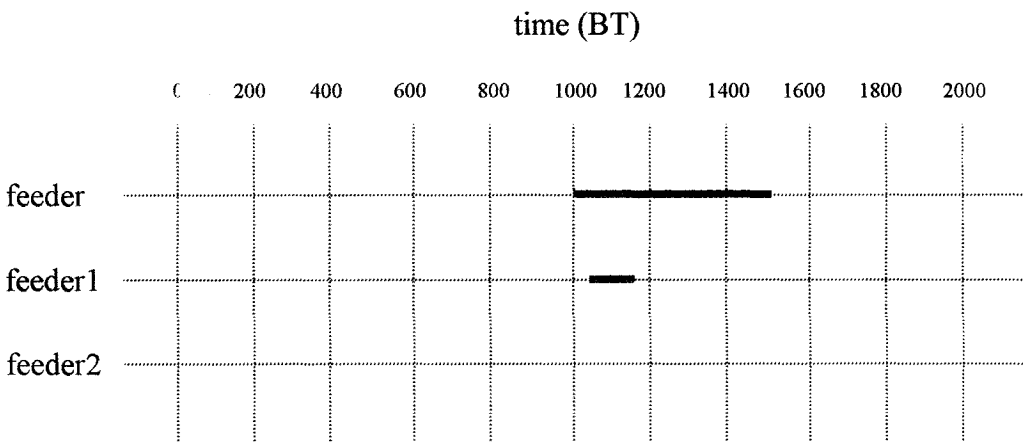


Figure 3-16: Comparison of feeder activity

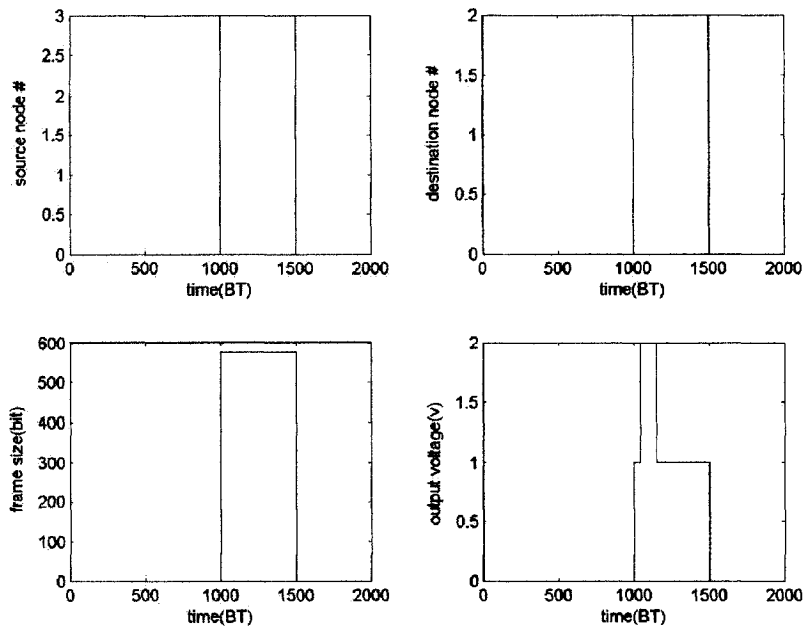


Figure 3-17: Output of Bus activity

3.4 Hub

If there is no filtering device such as Hub, connected into the segments of a network, more than one user may try to send data at the same time, which causes the collision. Hence, the data from each device hits each other and is damaged. The network area within which data packets originate and collide is called a collision domain. If a repeater detects a collision, it generates and sends a jam signal to all connected output ports. This ensures that every computer connected to the LAN is aware of the collision, and does not try to transmit during the collision period.

Any repeater or hub may optionally monitor the number of collisions, which it observes from each port. If the number rises above a threshold, the repeater / hub may then disable the port (sometimes called a "partition").

An important job of a hub (repeater) is to reconstruct the preamble before the frame is sent out of the hub. It happens as the transceiver cannot always lock on to the start of a frame's preamble and usually this takes a number of bits to completely synchronize with the Manchester Encoded data.

In addition, as a result of a collision an Ethernet LAN may transport parts of a frame. Any under-sized frame is known as a "Runt". A hub must extend any Runts (less than 96 bits) to ensure that all computers on the LAN see them. This ensures proper operation.

Note that, since a hub is responsible for changing the bit timing, and may introduce additional bits, as mentioned above, it can not guarantee that the IFG would be kept safe in all cases, however IFG should not be reduced below 47 BT.

A hub may also monitor the state of the transceiver at each port. On these interfaces, nodes send a unique signal every 16 +/- 8 ms. This signal is called an "idle pulse". Receivers monitor the medium to check that idle pulses are received. If neither an Ethernet frame, nor an idle pulse is received within 50-150 ms, the receiver may assume that the cable is not functioning correctly. A hub may then disable the port. A disabled port is made active again when 2-10 valid idle pulses are received. This can protect other correctly operating ports from the disruption caused by a faulty NIC, transceiver or hub.

The idle pulse is also used in some pieces of equipment to automatically detect the speed of transmission (10 or 100 Mbps) and availability of the full duplex mode.

3.4.1 Assumptions

In this approach, we assume a semi smart hub, which is not only a data distributor but also works as a collision detector and Jam generator. All the connected stations have equal and constant length of the cable. Two separated sets of buffers model the connected twisted pair wires to the hub, one for the input port like the transmit line of the cable, named "Trans-Buffer" and the other for the output port like the receiver line of the cable, named "Rec-Buffer". Though we already designed the cable block but for the ease of use and faster simulation reasons, connected cable delays and port delays (Appendix E) together, constitute the buffers sizes for the both input and output ports.

We do not need to amplify or regenerate the signal, as we do not simulate the degradation or attenuation of the signals. We simply, use the permissible parameters in simulation to avoid these stuffs. Meanwhile, this hub could be connected only to twisted pair cables. There is a small difference of delay on ports while we face the collision in hub (Appendix E), which is ignorable. There is no monitoring of over threshold collision numbers or cable disconnection on the ports; therefore partition mechanism is not necessary.

There is also no need to detect either the speed, as we set this parameter before start of simulation, or the half or full duplex as we work on half duplex in our hub.

3.4.2 Model Description

Basically, it has the similar state diagram as Bus (Figure 3-18), consists of three states: idle, carrier and wait for jam end and one action for send jam. The two latter cases play the collision rule in Bus. The designed FSM in this approach for Hub has also six events which occurrence of one, switch one state to another:

Event 1: One NIC transmit data

Event 2: No transmission

Event 3: More than one NIC transmit data

Event 4: Transmission jam signal

Event 5: End of jam signal and one NIC start to transmit data

Event 6: End of jam signal

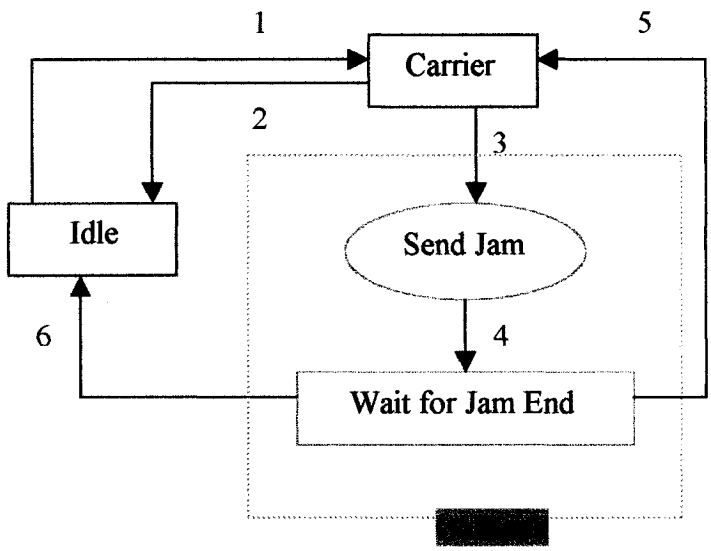


Figure 3-18: Hub state diagram

3.4.3 Simulation Description

At each cycle of simulation, all the input ports would be checked. In case of detecting only one active port, this signal after passing through the input FIFO buffer would be distributed to the all output FIFO buffers (ports) but itself. On the other hand detecting two or more active ports means collision. In this approach, as the cable and input port delays all gather in only one buffer, after detection, we wait for time that the signals take to reach and rise on the hub and then hub apply for the transmission of jam

signal pattern to all ports. During this period, all the input ports would be suspended. The following Figures (3-19,3-20) show the process. No signal on the input port, forces the idle signal through the buffers to push the possible remain bits out of the hub.

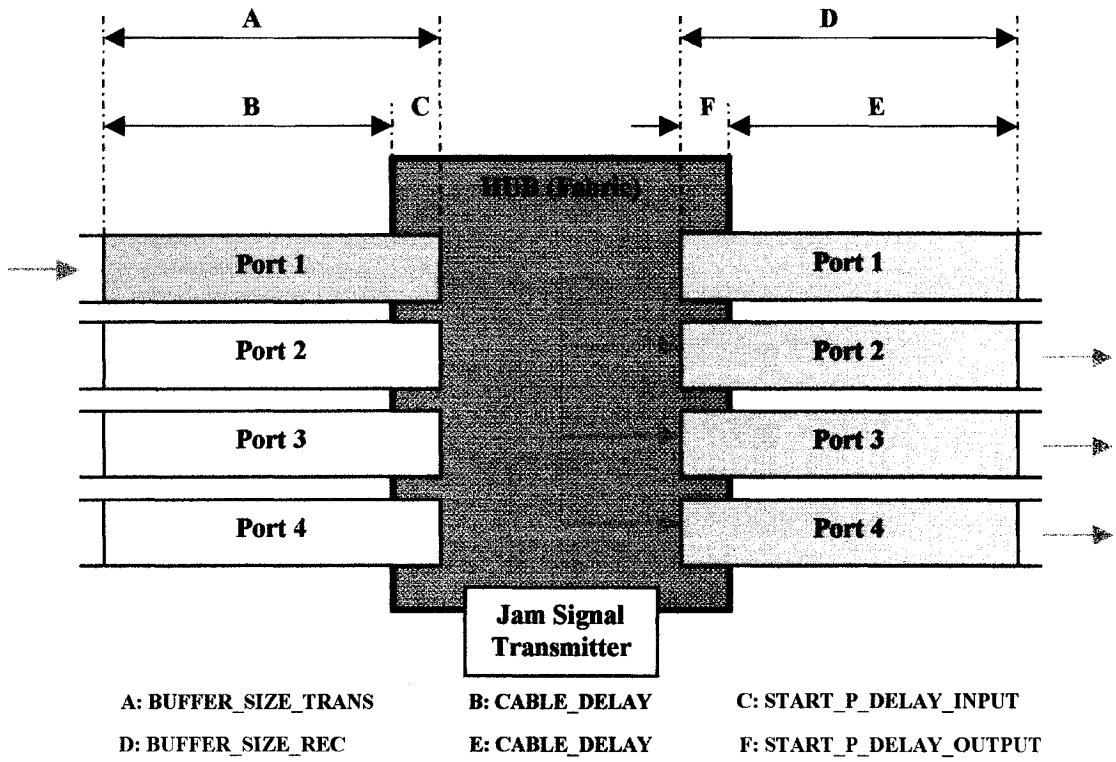


Figure 3-19: Simplified smart hub functionality diagram for regular operation

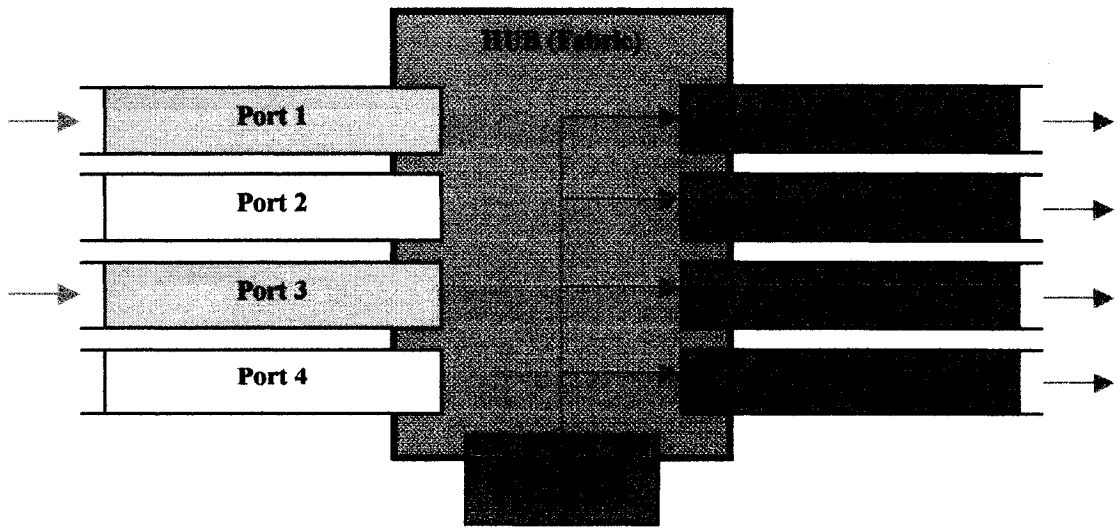


Figure 3-20: Simplified smart hub functionality diagram in collision operation

Table 3-1: Parameters for Hub model

CODE PARAMETER	DEFINITION	VALUE
MAX_BUFFER_SIZE_TRANS	Max (B+C) in Figure 3-22	*22 BT
MAX_BUFFER_SIZE_REC	Max (F+E) in Figure 3-22	*19 BT
BUFFER_SIZE_TRANS	B+C	19 BT
BUFFER_SIZE_REC	F+E	16 BT
MAX_PORT	Maximum number ports of hub in simulation	4
START_P_DELAY_INPUT	Start of packet propagation delay in input ports (C)	**16 BT
START_P_DELAY_OUTPUT	Start of packet propagation delay in output ports (F)	**13 BT
START_COLL_JAM_DELAY_INPUT	Start of collision jam delays in input ports	15.5 BT
START_COLL_JAM_DELAY_OUTPUT	Start of collision jam delays in output ports	11.5 BT
CABLE_LENGTH	Length of the cable connected from NIC to HUB ports	50m
P_DELAY	Propagation delay	0.006 µs/m
CABLE_DELAY	***Cable delay (B or E)	***3 BT

*For 10BaseT based on Max. 100m Segments.

** For 10BaseT as media.

***For 10BaseT as media, Where: CABLE_DELAY =(CABLE_LENGTH * P_DELAY)/BT, BT=0.1 µs

Note that, velocity of propagation (VOP) for twisted pair cables varies between, 0.45c to 0.78c [39][40][41], where c is, speed of the light in vacuum (300*10^6 m/s), depends on the category of cable hence propagation delay equal with 0.004 µs/m will be used for CAT5e cable or higher in Fast Ethernet simulation.

3.4.4 Simulation and Testing

The objective is to test the propagation delay caused by the hub block, distribution of signal and jam generator. Figure 3-21 shows the simulation configuration of this test.

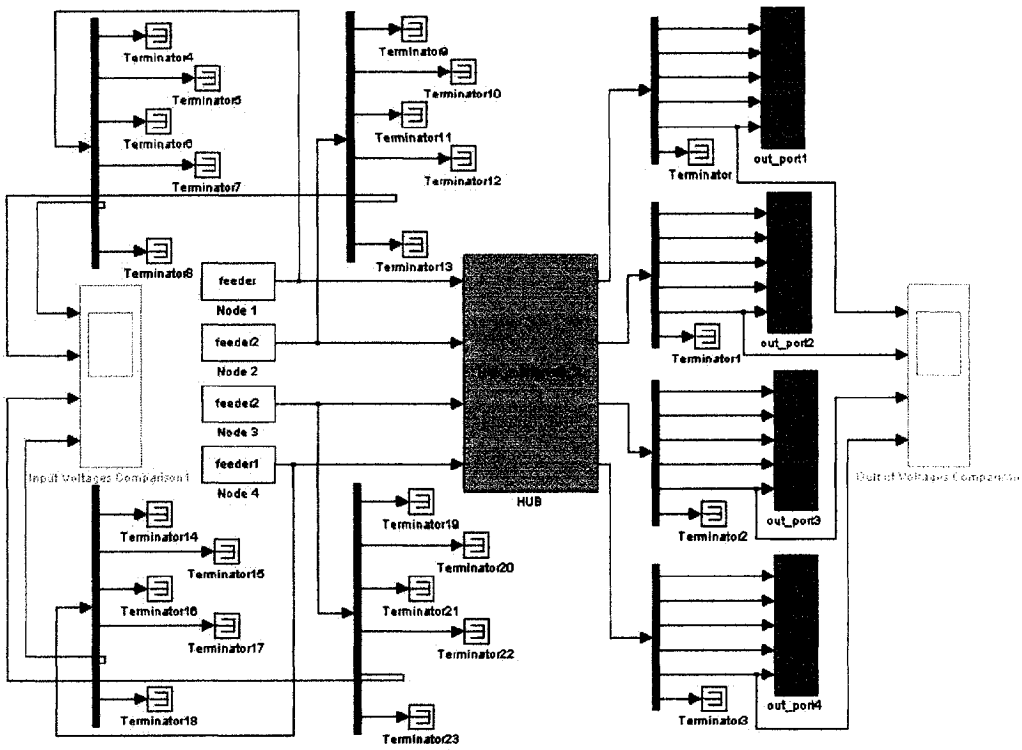


Figure 3-21: Simulation configuration used to test the Hub model

By looking at the following graph (Figure 3-22), we can simply see that we have a small delay due to the cable and input/output ports.

Ports 2 and 3 connected to feeder2 are not active for the entire of simulation time. Port 1 connected to feeder, shows activity during simulation and hub distribute this signal for the whole ports but itself. Port 4, connected to feeder1 has an activity at time 1010 and causes collision and generates jam signals for 32 bits long. Jam signal could be seen on the all output ports.

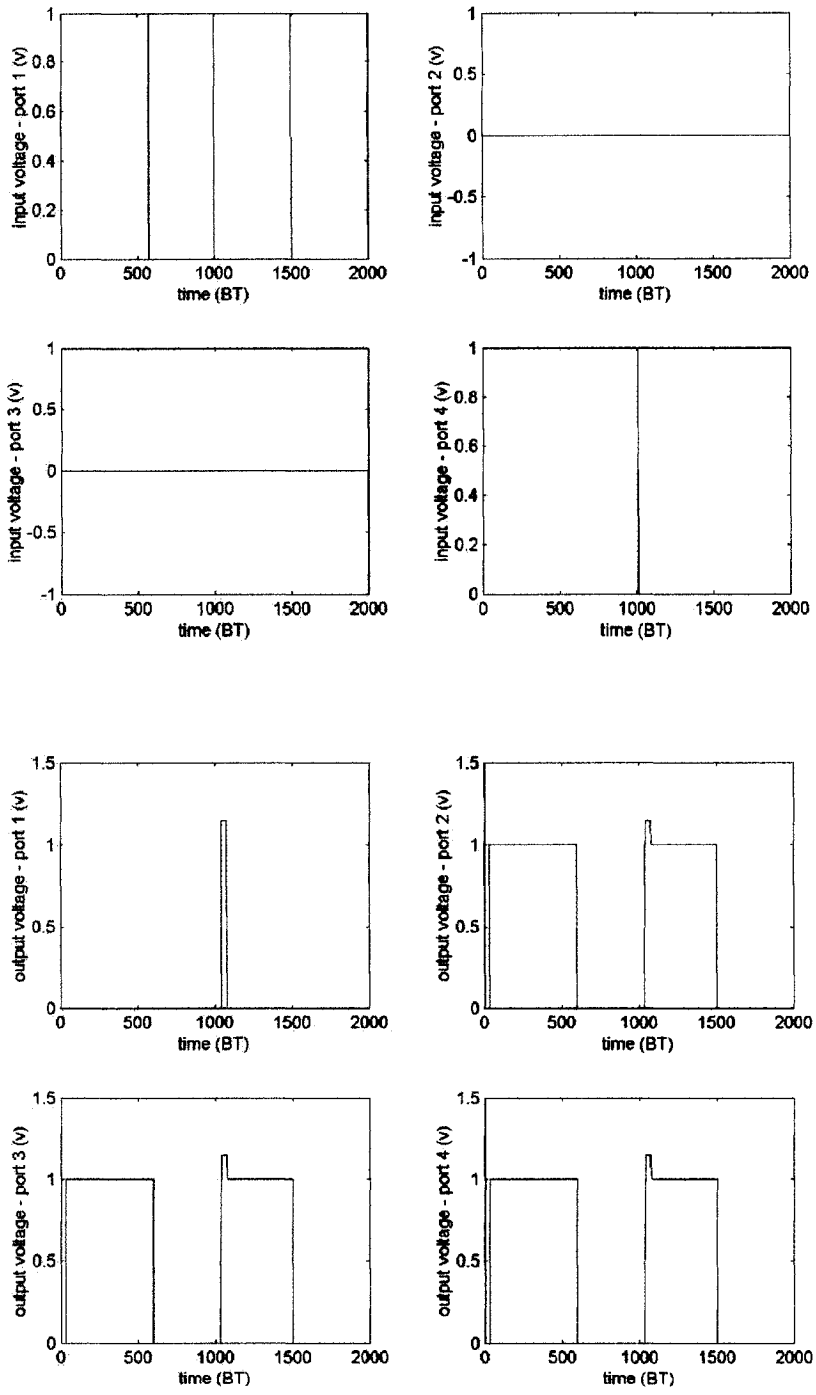


Figure 3-22: Voltage of inputs and output ports for the Hub

3.5 Switch

Full duplex is another advancement of Ethernet switching. As mentioned in the previous chapter, in a totally switched network, nodes only communicate with the switch and never directly with each other. Switched networks, employ either twisted pair or fiber optic cabling, both of which use separate conductors for sending and receiving data. In this type of environment, Ethernet stations can forgo the collision detection process and transmit at will, since they are the only potential devices that can access the medium. This allows end stations to transmit to the switch at the same time that the switch transmits to them, achieving a collision-free environment. A vital difference between a hub and a switch is that all the nodes connected to a hub share the bandwidth among themselves, while a device connected to a switch port has the full bandwidth all to itself.

LAN switches rely on packet switching. The switch establishes a connection between two segments just long enough to send the current packet. Incoming packets (part of an Ethernet frame) are saved in a temporary memory area (buffer); the MAC address existed in the frame's header is read and then compared to a list of addresses maintained in the switch's lookup table. In an Ethernet-based LAN, an Ethernet frame contains a normal packet as the payload of the frame, with a special header that includes the MAC address information for both source and destination of the packet.

Packet-based switches use one of three methods for routing traffic:

Cut-through: As soon as the switch detects a packet it reads the MAC address then stores the 6 bytes that make up the address information and immediately

begins to transfer the packet to the destination node, even as the rest of the packet is coming into the switch.

Store-and-forward: Save the entire packet to the buffer, proceed CRC error checking or check for other problems before sending. A packet with any error would be discarded. Otherwise, the switch looks up the MAC address and sends the packet to the destination. Store and forward switches are slower but they do not forward the corrupted frame.

Fragment-free: A less common method is fragment-free. It works like cut-through except that it stores the first 64 bytes of the packet before sending it on. The reason for this is that most errors, and all collisions, occur during the initial 64 bytes of a packet.

Many switches combine the two methods, using cut-through until a certain error level is reached and then changing over to store-and-forward. Another kind of switch, called Backpressure switch, works above layer 2. It can send overflow message to the workstation when the workstation sends too much packets and causes the buffer full. Very few switches are strictly cut-through, since this provides no error correction.

A switch includes a number of input and output ports and a fabric. The ports can communicate with the stations and/or the other switches and the fabric will receive the packet from the input port and forward it to output port.

The design goal of switch includes three concerns: throughput, scalability and cost.

The main problem of switch design is contention in case of high and particular throughput. For example, several input ports want to send message to one output port at the same time and will overflow it. There are three kinds of buffer architecture design:

Crossbar Switch: A Crossbar switch has n buffers for all the input ports in each output buffer, which can let n inputs send message to an output simultaneously. The complexity of building such architecture grows in proportional to n^2.

Knockout Switch: In this design, it assumes that no more than k input ports would send message to an output, so there are only k buffers in an output port. The complexity of building such architecture grows in proportional to n if k is much less than n.

Shared-media Switch: A shared-media switch uses a big buffer for all the input and outputs ports (you can divide it into n*n part to form a crossbar switch). The packet from each input port will be saved to the big buffer and then will be sent to the appropriate output consequently. Therefore, it needs n times faster memory to read and write the packet.

On the other hand, we can also categorize the switches by their buffer locations as follow:

Input Buffering: Packets are stored in the input queue until the switch fabric is available. This may lead to HOL (Head Of Line) blocking. It happens when packets blocked in the input ports while they have contention for the output port. (Can potentially reduce the efficiency of a switch to 60% of the maximum offered load, therefore should be avoided.)

Output Buffering: Packets are stored in the output queue while the output port is busy or paused.

Central Buffering: All ports use one memory pool, which causes lower efficiency for the higher port numbers.

There are also some other types like crossbar buffering, both input/output ports buffering, virtual output buffering or the other approaches which are combination of those above mentioned types and created by the manufacturers.

Note that, conceptually in the virtual output queue switch architecture, the input buffer appears as multiple queues (one per output port) so we have no longer HOL as an issue.

Most Ethernet switches use a very interesting system called transparent bridging to create their address lookup tables. Transparent bridging is a technology that allows a switch to learn everything it needs to know about the location of nodes on the network without the network administrator having to do anything.

3.5.1 Assumptions

In this research, full duplex switches with store and forward mechanism has been modeled and developed. Since the same topology and cable policy in length and type used in hub, have been also developed in switch; we see the same buffer structure and size for the input/output ports in here too.

While we do not have spanning tree and address learning in this approach, before beginning of simulation, we should setup the address look up table in switch in order to define the connection of each port. Usually the last port would be open to a non-specified

node. Note that each port will be connected to only one station and all the nodes send the same frame size.

It happens that more than one node have data for the same destination. In this case we used the priority mechanism to define which station should send first. Lower node index has the higher priority.

After several times correspondence and conversations with the most famous switch manufacturers like 3COM, NETGEAR and D-Link we reached this conclusion that no information concerning the port buffering or fabric design architecture would be accessible. On the other hand any architecture or buffering, which lead us to our objective, has no backlash (like HOL) and is compatible with our simulation circumstances could be acceptable. A modified type matrix crossbar structure constitutes the buffer management of this switch. We will discuss this in section 3.5.3. When a switch detects an upcoming buffer overflow; it transmits a pause control packet to the sender, requesting to stop transmission for a certain amount of time. After this time, the sender may restart the transmission. If sufficient buffers at the receiver become free in the meantime; the switch can readmit transmission by sending a "PAUSE" frame. We did not apply this in our approach for the following reasons:

- As we consider unlimited buffer in our developed code, there is no problem concerning overflow.
- An overload only happens due to unpredictable conditions. So, it would not be necessary in a properly designed Ethernet automation system. Transmitting high priority packets can help to avoid these critical situations. [8]

- Though the store and forward mechanism used in this switch, we never consider CRC error checking, since we do not model degradation or attenuation or even noise in this research.

3.5.2 Model Description

Figures 3-23, 24, 25 illustrate the algorithm of the modeled switch in this approach. A complete description can be found in section 3.5.3.

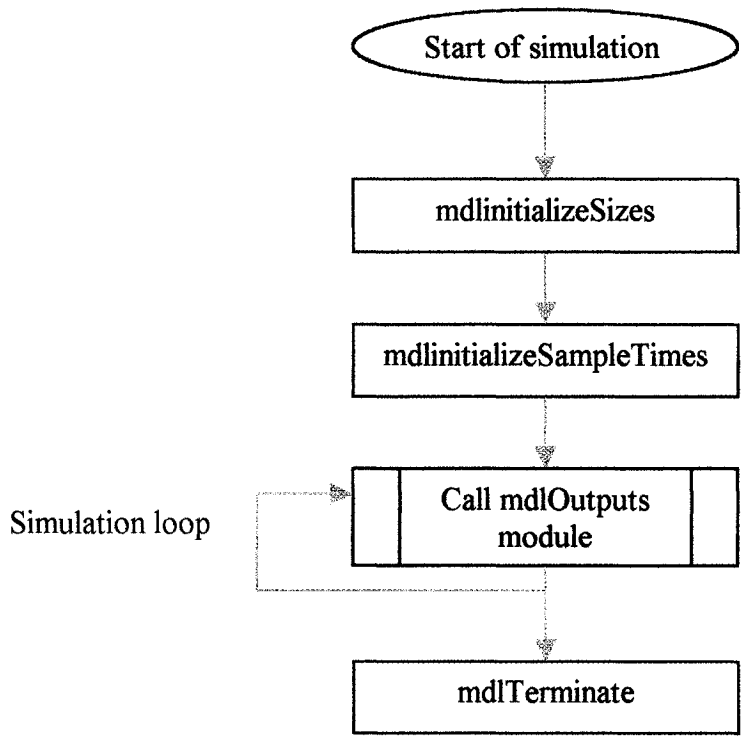


Figure 3-23: Modeled switch algorithm

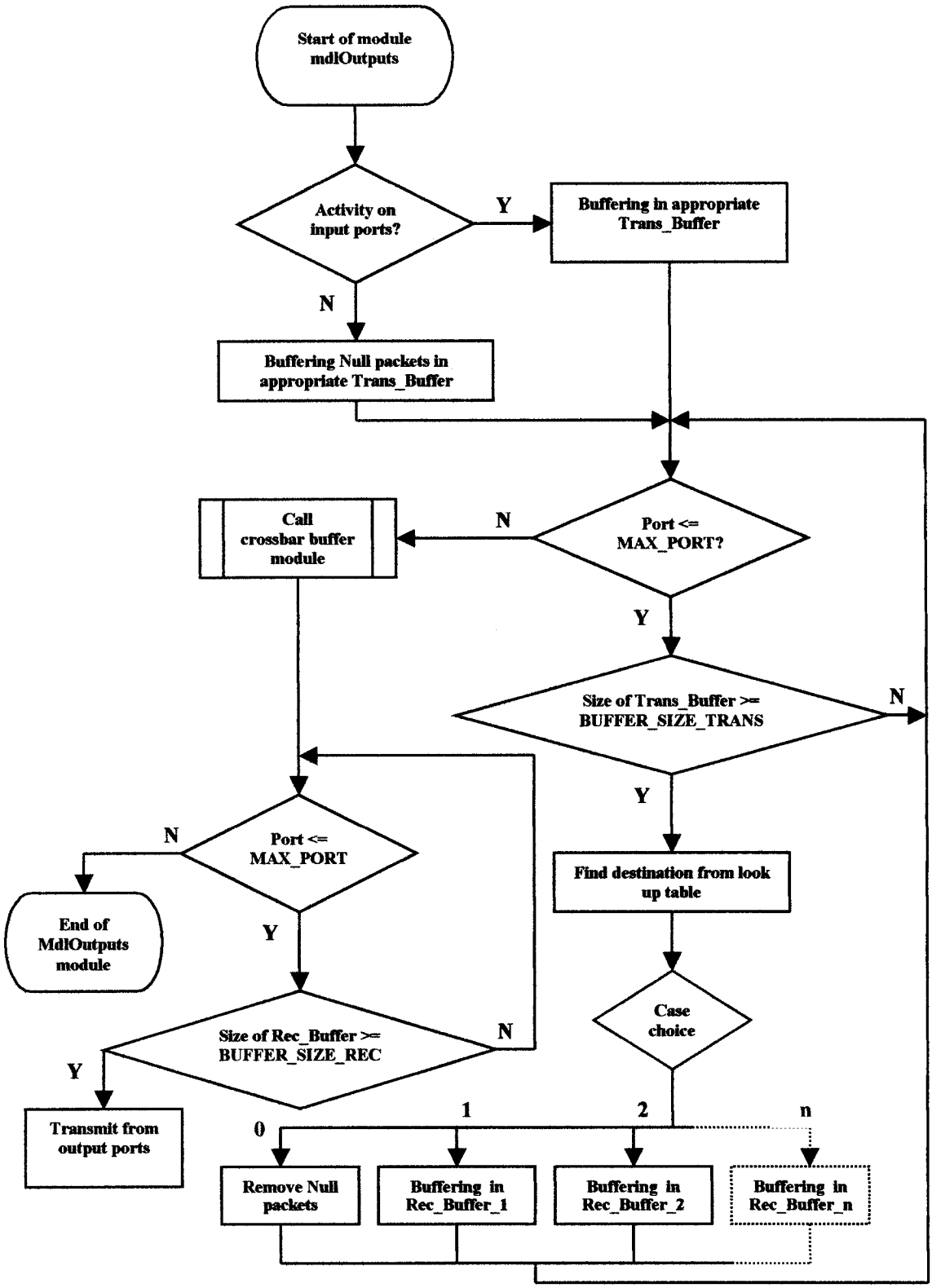


Figure 3-24: mdlOutput function algorithm

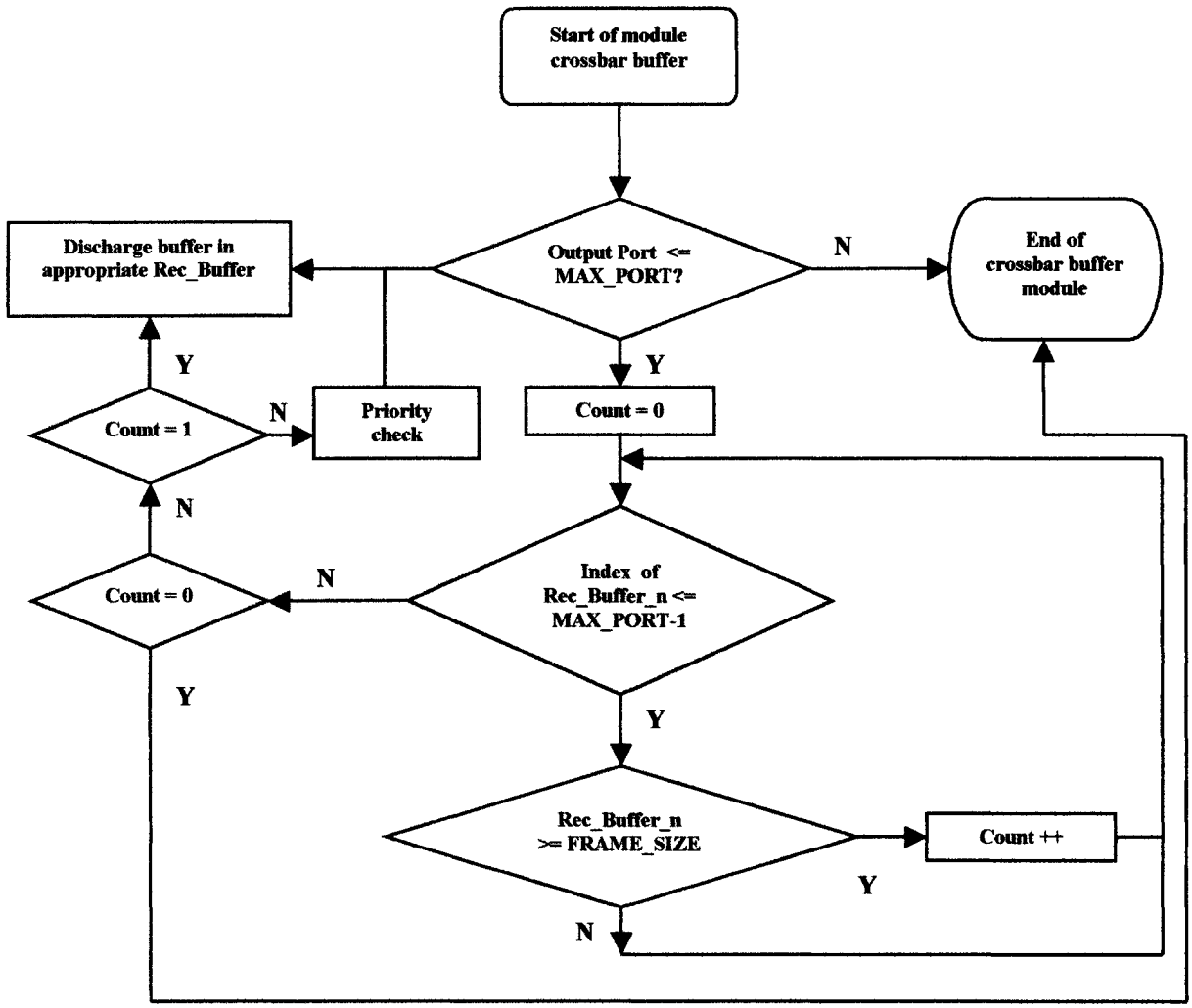


Figure 3-25: Crossbar buffer module

3.5.3 Simulation Description

As already mentioned we have designed two types of input/output port buffers equal to the number of ports in our switch. The third party is the core buffering mechanism, which has the responsibility to do the store and forward. Simulation starts with checking the ports and line up the incoming bits in their appropriate input "Trans-Buffer" port. Then after they pass through these buffers, or on the other words, they pass through the cable and show up on the input ports, they will look up for their destination ports in the table address and will place in their corresponding buffers. Having an " n "port switch, each input port will have access to "n-1" output buffers as we never send data to its source. The following figure (Figure 3-26) illustrates this idea. Bits will gather in these buffers to reach the frame size and then will release from switch out by moving in the "Rec-Buffer". From now on, they are on their way to their destinations after the total time it taken to rise up on the output port and propagate in the second line of twisted pair cable. If we have data from two or more source to one destination (or port) simultaneously and have no high priority packets then we should somehow compromise it. Either randomly choose destination or make our own priority for them, could manage this case. As we have already said, the latter has been chosen for this scenario.

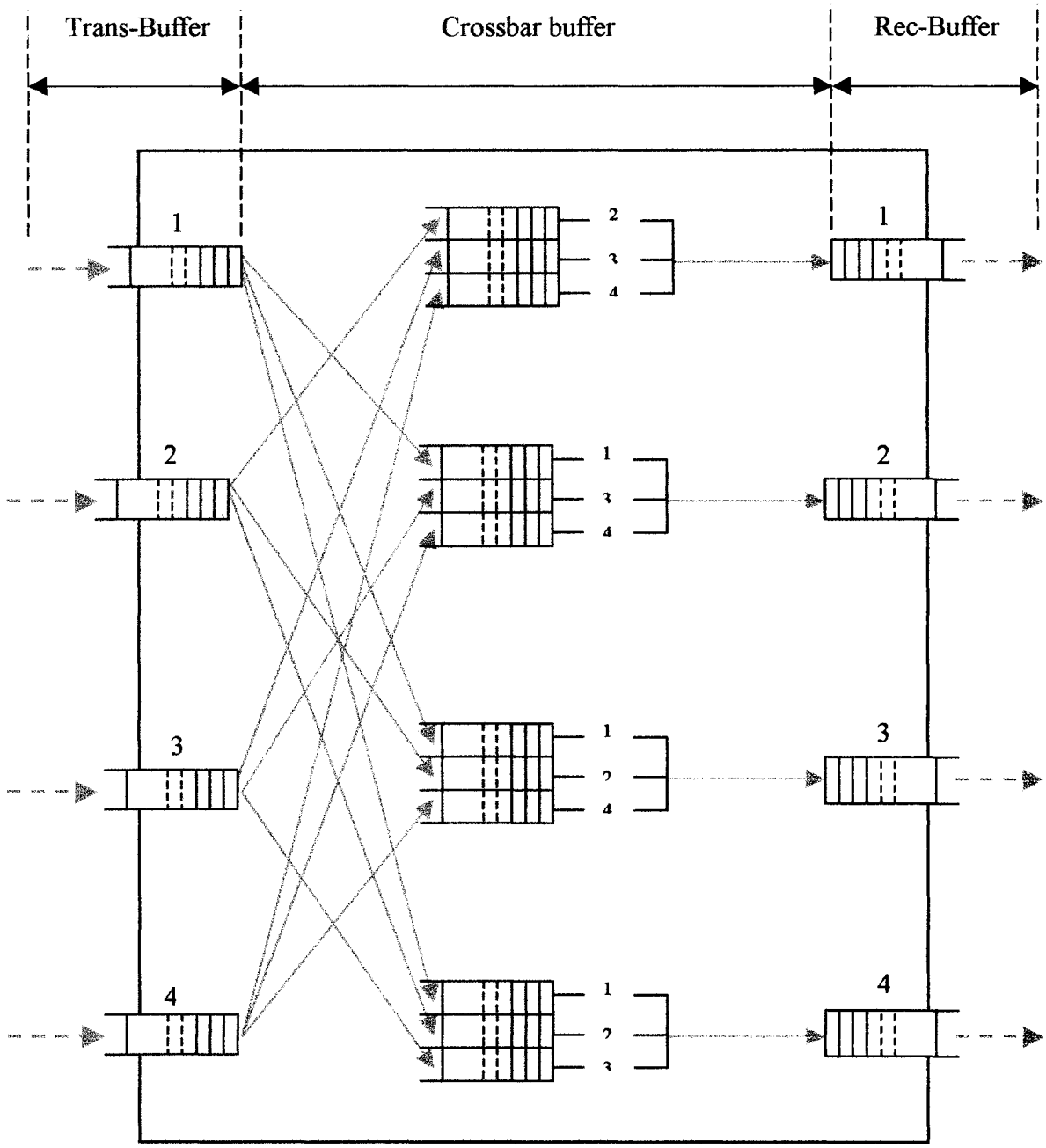


Figure 3-26: Four-port switch with crossbar buffering architecture

Table 3-2: Designated parameters in Switch

CODE PARAMETER	DEFINITION	VALUE
MAX_BUFFER_SIZE_TRANS	Max buffer size for trans. line in code, due to the ma permissible length of segment	*22 BT
MAX_BUFFER_SIZE_REC	Max buffer size for rec. line in code, due to the max permissible length of segment	*19 BT
BUFFER_SIZE_TRANS	START_P_DELAY_INPUT+ CABLE_DELAY	19 BT
BUFFER_SIZE_REC	START_P_DELAY_OUTPUT+CABLE_DELAY	16 BT
MAX_PORT	Maximum number ports of switch in simulation	8
START_P_DELAY_INPUT	Start of packet propagation delay in input ports	**16 BT
START_P_DELAY_OUTPUT	Start of packet propagation delay in output ports	**13 BT
CABLE_LENGTH	Length of the cable connected from NIC to switch ports	50m
P_DELAY	Propagation delay	0.006 μs/m
CABLE_DELAY	***Cable delay (B or E)	***3 BT

*For 10BaseT based on Max. 100m Segments.

** For 10BaseT as media.

***For 10BaseT as media, Where: CABLE_DELAY =(CABLE_LENGTH * P_DELAY)/BT, BT=0.1 μs

Note that, velocity of propagation (VOP) for twisted pair cables varies between, 0.45c to 0.78c [39][40][41], where c is, speed of the light in vacuum (300*10^6 m/s), depends on the category of cable hence propagation delay equal with 0.004 μs/m will be used for CAT5e cable or higher in Fast Ethernet simulation.

3.5.4 Simulation and Testing

Having the same feeding method, we can also easily test the functionality of our switch. Note that for the whole designed parts, we also checked the codes by controlling their log out files, which is too bulky and complex to demo in here. Simulation configuration for

this test is illustrated in Figure 3-27. The following table shows the feeding basis of our test simulation. The first and second columns correspond to the look up table of our 8-port switch.

Table 3-3: Feeding basis of switch test simulation

Ports	Connected Node	Feeder	Destination Node
1	1	f12	2
2	3	f34	4
3	8	f84	4
4	5	f53	3
5	4	f42	2
6	2	f23	3
7	6	f61	1
8	Open*	f72	2

* In this simulation node 7.

As it seen in Table 3-3, we have packets only destined to the following nodes: 1,2,3,4 which are correspond to the output ports: 1,6,2,5 and the risen voltage should be seen on them after an appropriate delay caused by cable, ports and store and forward mechanism.

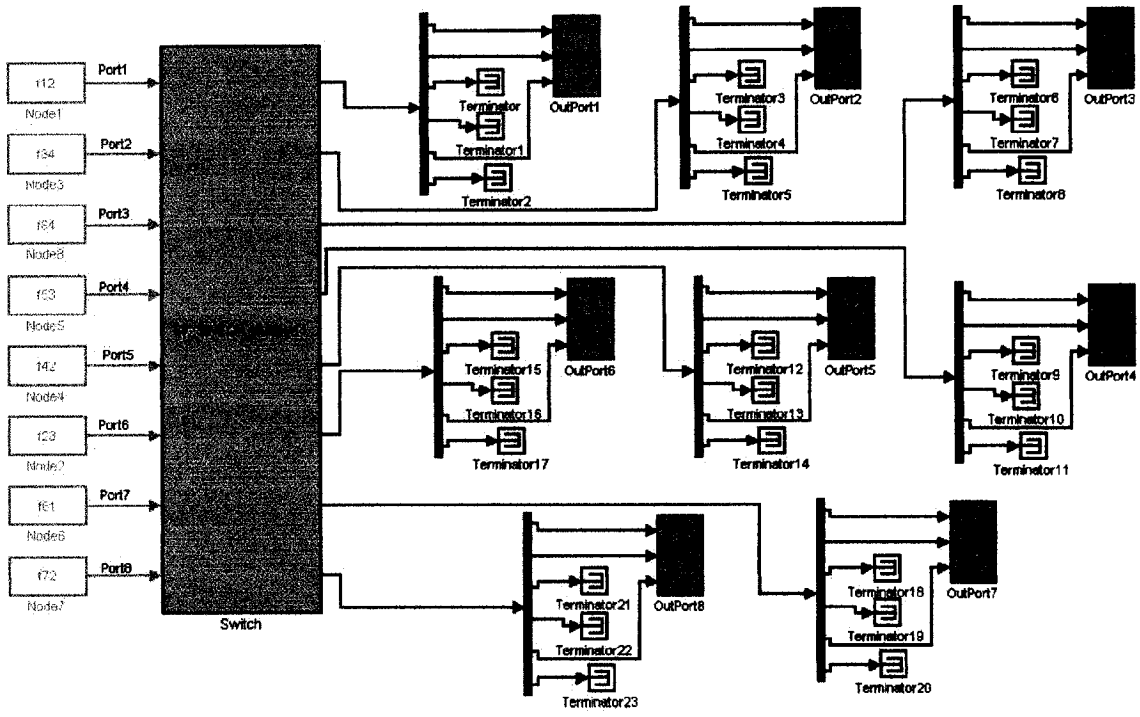


Figure 3-27: Simulation configuration used to test the Switch model

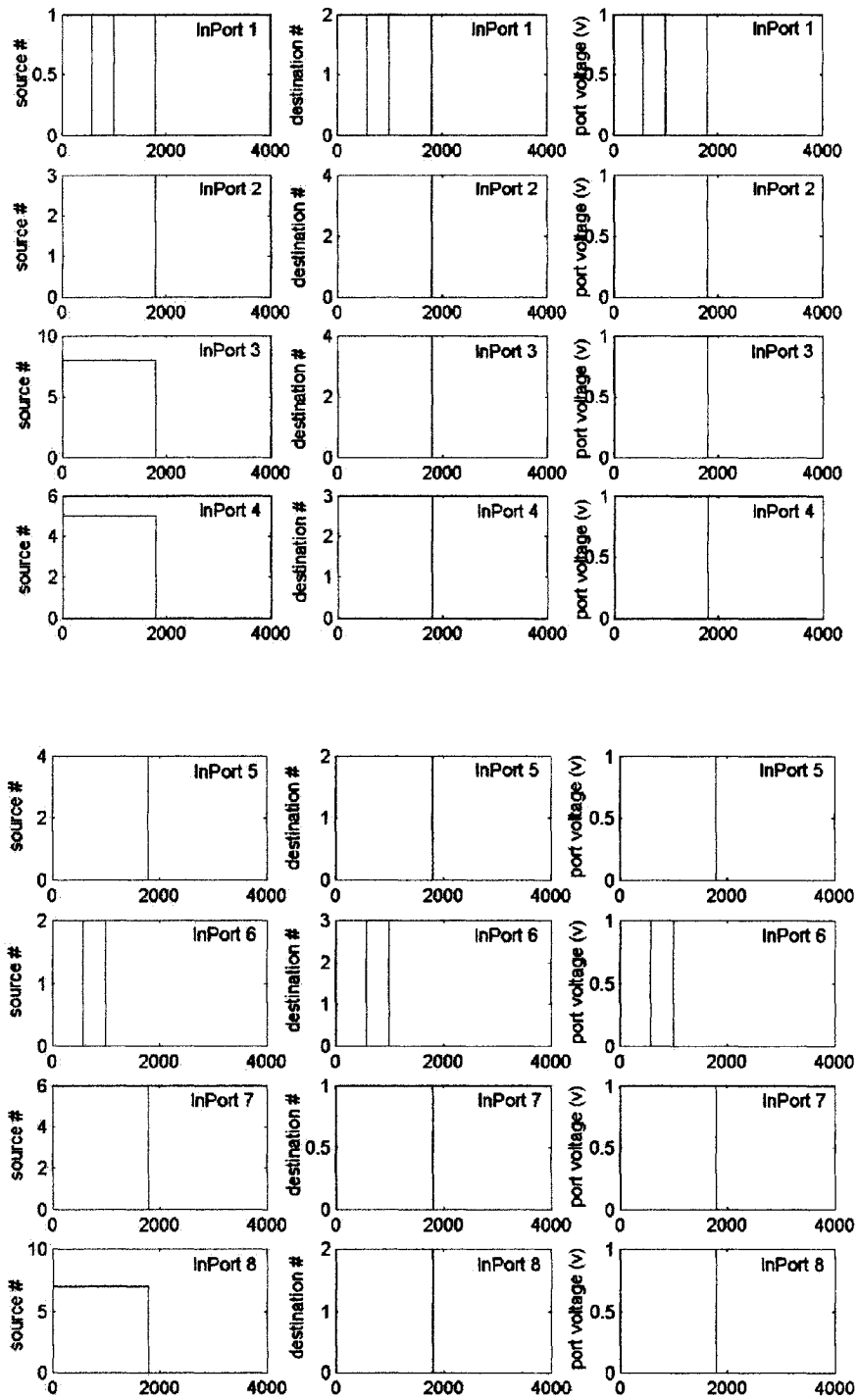


Figure 3-28: Comparison of inputs to switch ports

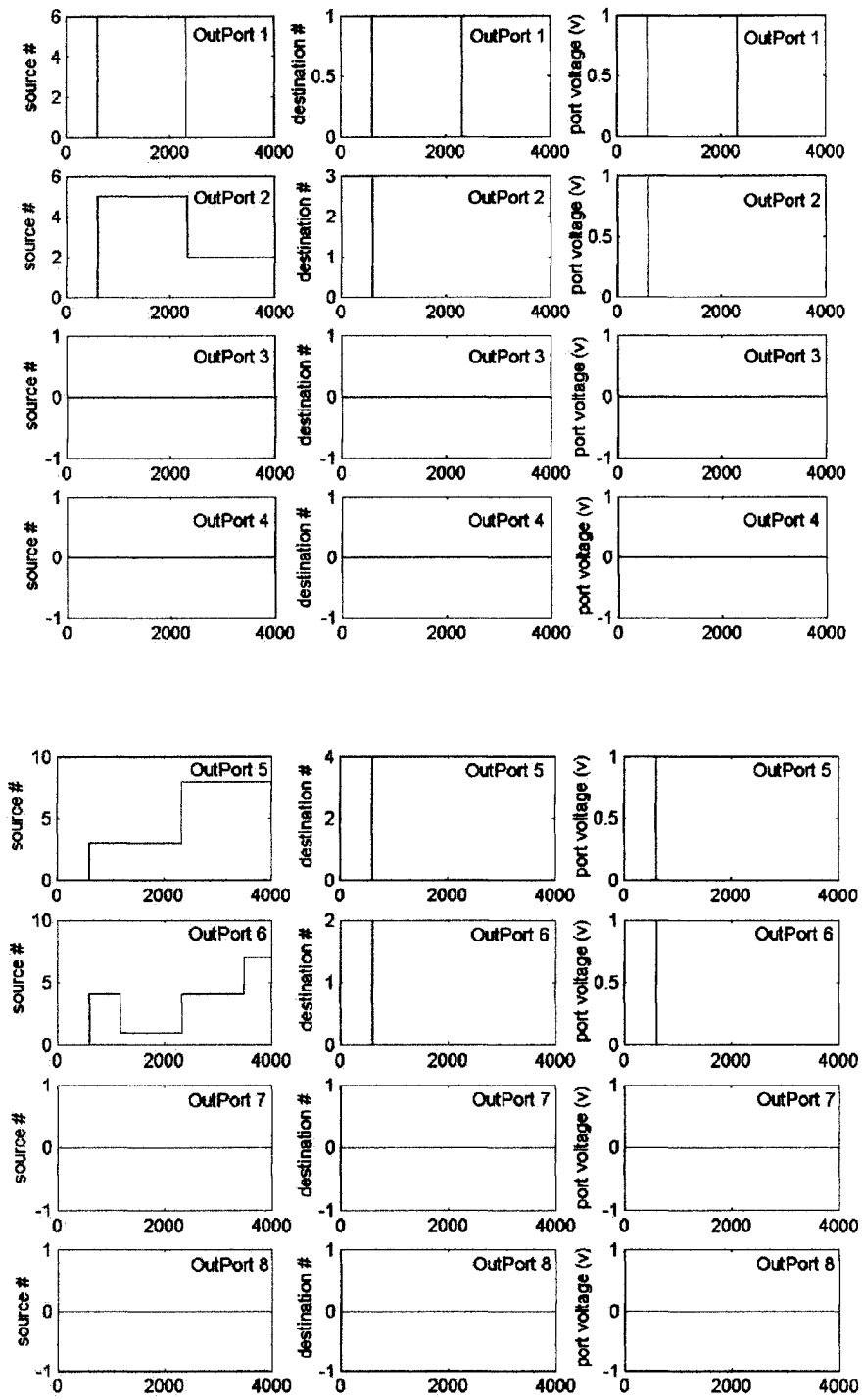


Figure 3-29: Comparison of outputs to switch ports

Designated propagation delay of each ports, store and forward mechanism, simultaneously receiving two or more packets destined for one node and control appropriate multicast functionality of switch were the main objectives of this test.

3.6 Interconnections of Connections

Different conventional topologies have been tested as seen in the following:

3.6.1 Model 1: Half Duplex NIC with Coaxial Cable as Bus

Figure 3-30 shows the configuration for this test. System parameters can be found in Table 3-4 and results for this model can be seen in Figure 3-31.

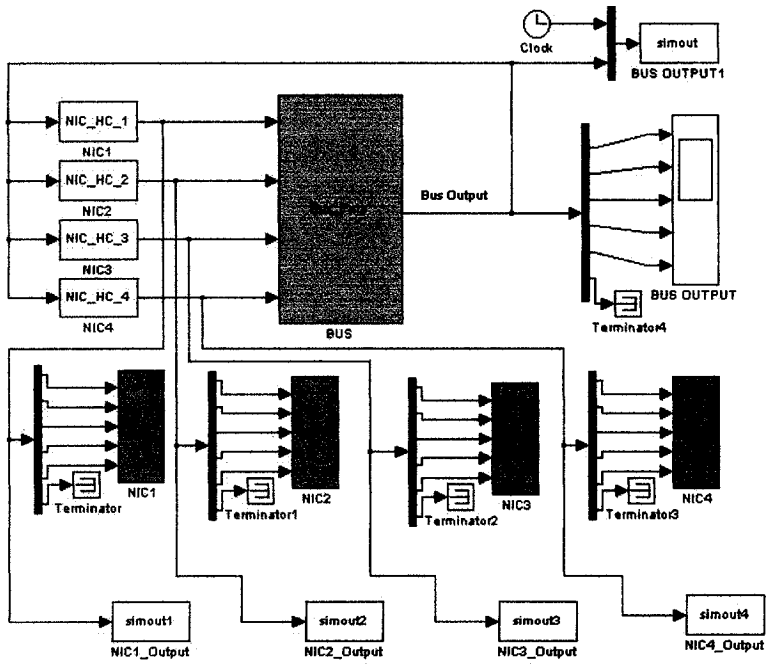


Figure 3-30: Interconnection of four NICs in half duplex mode with 10Base5 (Coaxial cable) Bus

Table 3-4: Model 1, system parameters

OPERATION MODE	**Half Duplex**
MAU TYPE	**10Base5**
***CABLES LENGTH (m)**	**60**
BT (μs)	**0.1**
SLOT TIME (μs)	**51.2**
IFG (μs)	**9.6**
SIMULATION TIME (BT)	**20000**
SIMULATION TYPE	**Fixed Step, Discrete (No Continuous States)**
SAMPLE TIME (BT)	**1**
NUMBER OF NODES	**4**
RECEIVING BUFFER SIZES	**100**
TRANSMISION BUFFER SIZES	**100**
****FRAME SIZE (bit)**	**576**
*****JAM VOLTAGE (volt)**	**1.2**
DATA RATE (Mbps)	**10**

*Length of the cable, connected from each node (NIC) to BUS.

**Preamble included.

***In this simulation based on this formula: V_JAM = V_XMIT + 1.0/(MAX_NODE_ETHERNET+1)

Where: MAX_NODE_ETHERNET=4.0 and V_XMIT =1.0

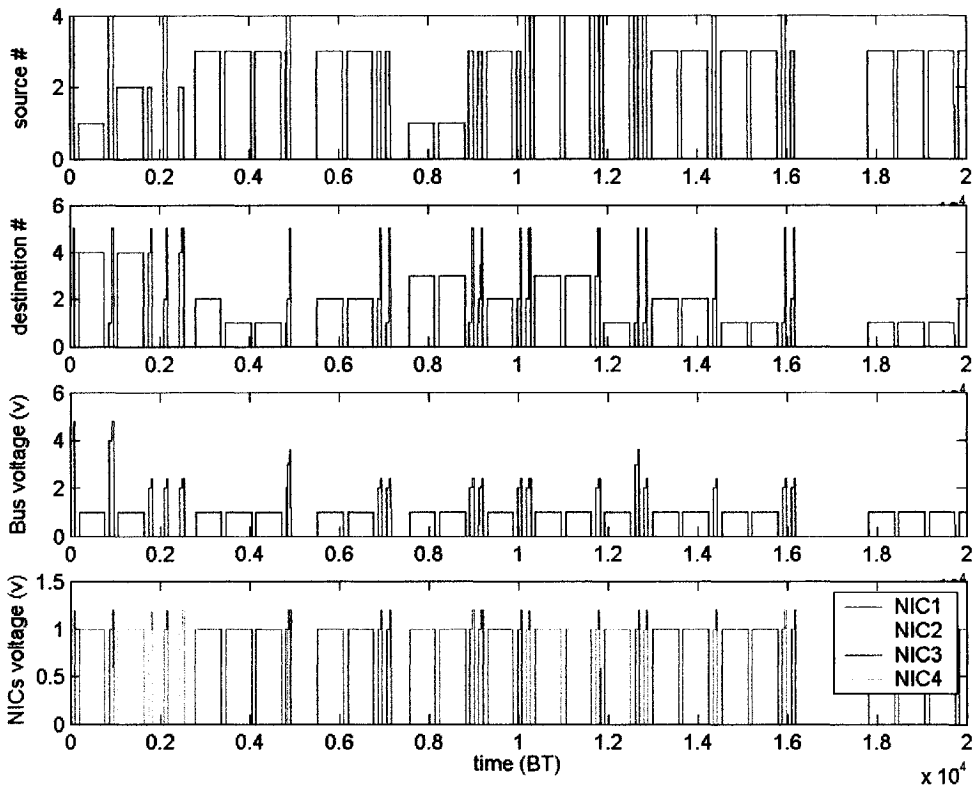


Figure 3-31: Signal activity on the BUS

Note that, destination nodes for each packet are chosen randomly within the three other nodes, and zero releasing policy is used in this test. Different releasing policies will be discussed more in chapter 5.

3.6.2 Model 2: Half Duplex NIC, Twisted Pair Cable and 4 Ports Hub

Figure 3-32 shows the configuration for this test. System parameters can be found in Table 3-5 and results for this model can be seen in Figure 3-33.

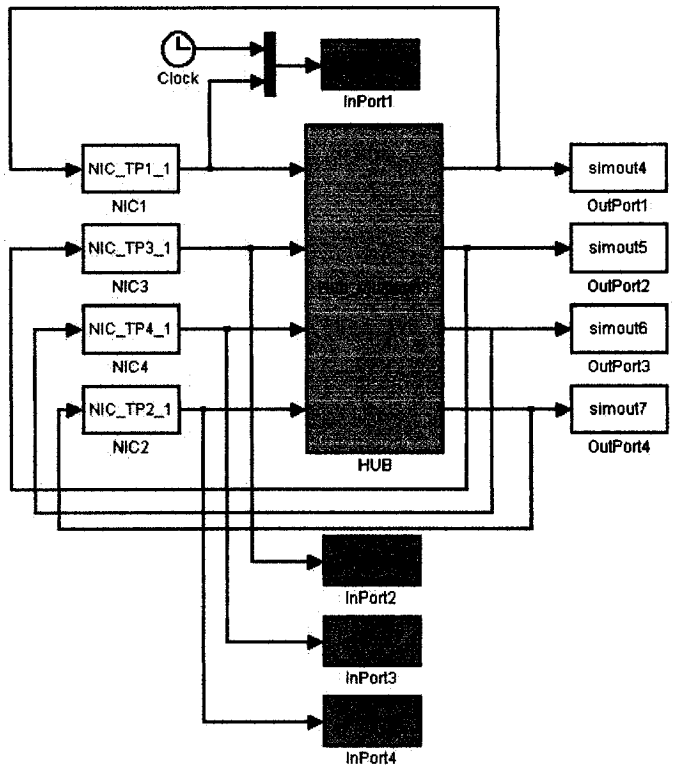


Figure 3-32: Interconnection of four NIC in half duplex mode with 10BaseT(Twisted pair) as media and four-port Hub

Note that, destination nodes for each packet are chosen randomly within the three other nodes, and zero releasing policy is used in this test. Different releasing policies will be discussed more in chapter 5.

Table 3-5: Model 2, system parameters

OPERATION MODE	**Half Duplex**
MAU TYPE	**10BaseT**
***CABLES LENGTH (m)**	**50**
BT (μs)	**0.1**
SLOT TIME (μs)	**51.2**
IFG (μs)	**9.6**
SIMULATION TIME (BT)	**10000**
SIMULATION TYPE	**Fixed Step, Discrete (No Continuous States)**
SAMPLE TIME (BT)	**1**
NUMBER OF NODES	**4**
RECEIVING BUFFER SIZES	**100**
TRANSMISION BUFFER SIZES	**100**
****FRAME SIZE (bit)**	**576**
*****JAM VOLTAGE (volt)**	**1.2**
DATA RATE (Mbps)	**10**
NUMBER OF HUB 'S PORT	**4**

*Length of the cable, connected from each node (NIC) to HUB.

**Preamble included.

***In this simulation based on this formula: V_JAM = V_XMIT + 1.0/(MAX_NODE_ETHERNET+1)

Which: MAX_NODE_ETHERNET=4.0 and V_XMIT =1.0

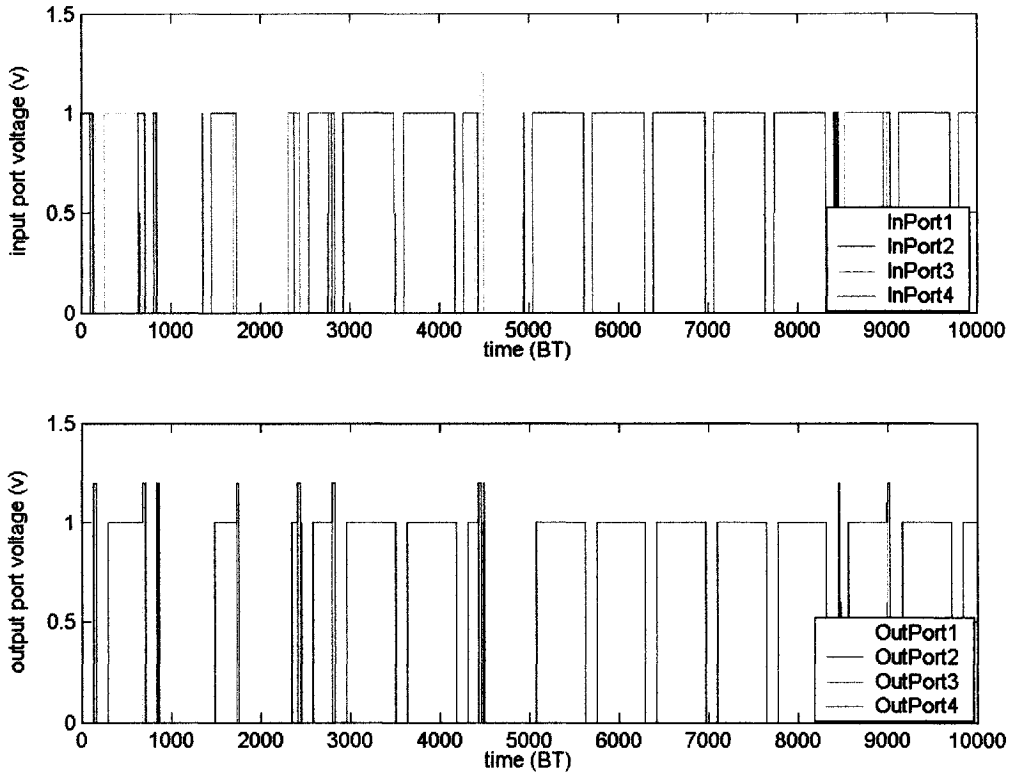


Figure 3-33: Illustration of voltages on the input / output ports of Hub model

3.6.3 Model 3: Full Duplex NIC, Twisted Pair Cable and 8 Ports Switch

In this test, like model 2, destination nodes for each packet are chosen randomly within the seven other nodes, and one more time, zero releasing policy has been applied in this test. Figure 3-34 shows the configuration for this test. System parameters can be found in Table 3-6 and results for this model can be seen in Figure 3-35.

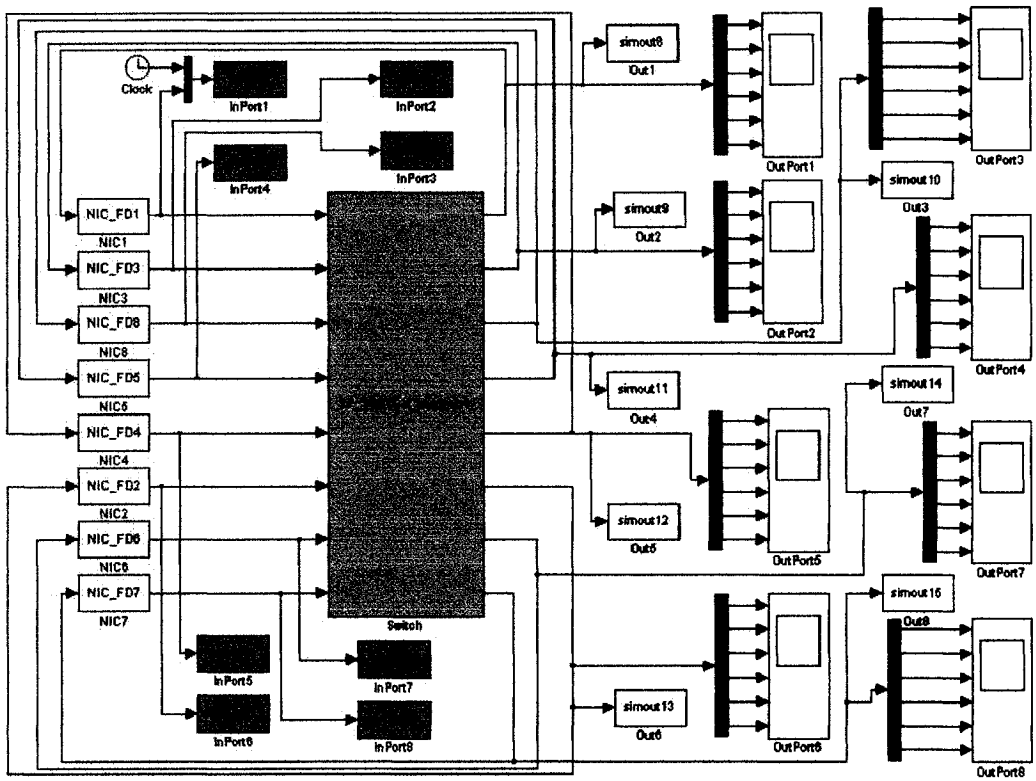


Figure 3-34: Interconnection of four NICs in half duplex mode

with 10BaseT(Twisted pair) as media and eight-port switch

Table 3-6: Model 3, system parameters

OPERATION MODE	Half Duplex
MAU TYPE	10BaseT
*CABLES LENGTH (m)	50
BT (μs)	0.1
SLOT TIME (μs)	51.2
IFG (μs)	9.6
SIMULATION TIME (BT)	3000
SIMULATION TYPE	Fixed Step, Discrete (No Continuous States)
SAMPLE TIME (BT)	1
NUMBER OF NODES	8
RECEIVING BUFFER SIZES	100
TRANSMISION BUFFER SIZES	100
**FRAME SIZE (bit)	576
DATA RATE (Mbps)	10
NUMBER OF SWITCH'S PORT	8

*Length of the cable, connected from each node (NIC) to Switch.

**Preamble included.

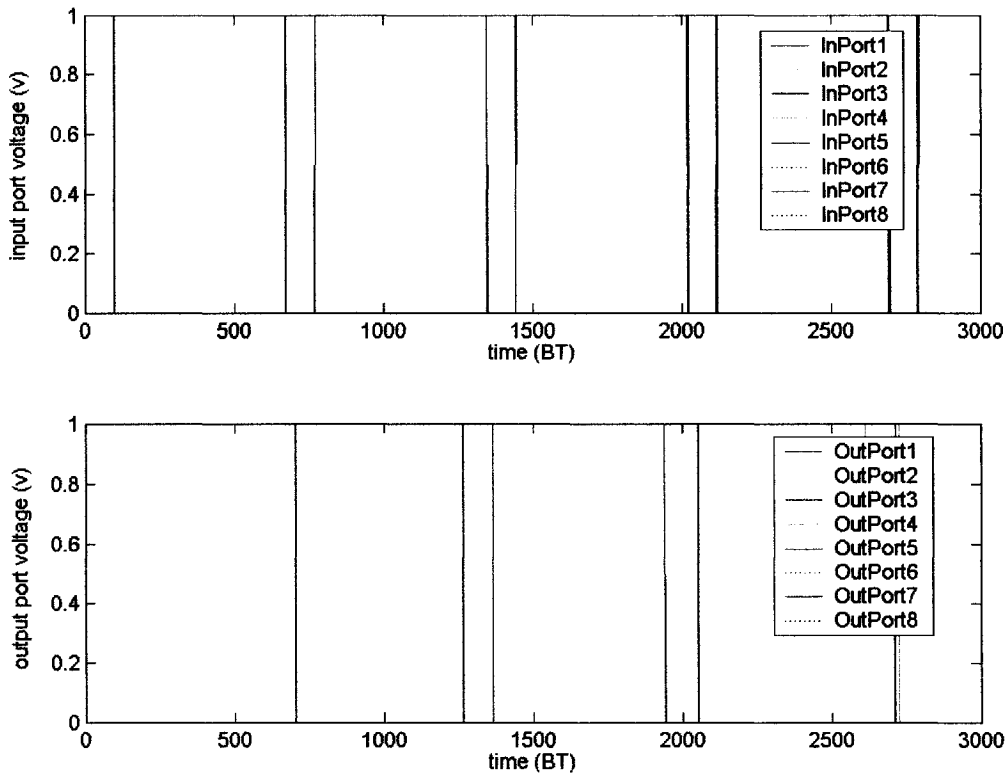


Figure 3-35: Voltages on the input / output ports of the Switch model

CHAPTER 4: PARAMETER IDENTIFICATION

In this chapter first we introduce the most important parameter involved in the simulation. Then we discuss the performance parameters of a network as well as their definitions. Later, we focus on latency quantifications and measurement methods for each component. Eventually the designated parameters of each model, which concern the latency issue, are compared with the experimental data. There are several parameters involved in simulation model. For instance, we can name number of nodes, message period and message size. A list of the parameters are given in Tables 4-1a,b.

Note that, parameters, which are designed, exclusively for a device, and those, which are only related to the functionality of S-functions, are not listed in the following tables (Table 4-1a,b and Table 4-2).

Table 4-1a: General simulation parameters

CODE PARAMETER	DEFINITION	VALUE
NODE	Virtual MAC address of each node	-
MAX_NODE_ETHERNET	Maximum number of nodes in network	-
V_XMIT	Transmit voltage	1 v
V_IDLE	Idle voltage	0 v
V_CD	Collision trigger voltage in coaxial media	2 v
MAX_TRANS_BUFFER_SIZE	Maximum transmission buffer size	100 frame
MAX_REC_BUFFER_SIZE	Maximum receive buffer size	100 frame
FRAME_SIZE	Frame size	*576 bits
ATTEMPT_LIMIT	Attempt limit in BEB algorithm	16
BACKOFF_LIMIT	Back off limit in BEB algorithm	10
JAM_SIZE	Jam pattern length	32 bits
CABLE_LENGTH	Cable Length, connection from node to device	-
MAX_FRAME_SIZE	Maximum permissible frame size	*1526 octet
MIN_FRAME_SIZE	Minimum permissible frame size	*576 bits

*Preamble included

Table 4-1b: Network simulation parameters

10 Mbps Ethernet Parameters (Mbps_10)		
BT	**Bit Time**	**0.1 µs**
SLOT_TIME	**Slot time**	**512 BT**
INTER_FRAME_GAP	**Inter frame gap (IFG)**	**9.6 µs**
100 Mbps Ethernet Parameters (Mbps_100)		
BT	**Bit Time**	**0.01 µs**
SLOT_TIME	**Slot time**	**512 BT**
INTER_FRAME_GAP	**Inter frame gap (IFG)**	**0.96 µs**
1000 Mbps Ethernet Parameters (Mbps_1000)		
BT	**Bit Time**	**0.001 µs**
SLOT_TIME	**Slot time**	**4096 BT**
INTER_FRAME_GAP	**Inter frame gap (IFG)**	**0.096 µs**
BURST_LIMIT	**Burst limit**	**65536**

Table 4-2: Simulation parameters used in code and Simulink interface

CODE'S PARAMETER	DEFINITION	VALUE
MAX_INPUT_SIGNAL_WIDTH	Width of the input signals to block	*6
MAX_OUTPUT_SIGNAL_WIDTH	Width of the output signals from block	*6
sample_time	Sample time in ssSetSampleTime	1
offset_time	Offset time in ssSetSampleTime	0.0
P_DELAY	Propagation delay in cables	**
MAX_PACKET_ID	Max ID number for the transmitted packets	-
MAX_SEG	Max permissible length of segment (m)	**
Start time/Stop time	Set in Simulink interface	-
Type: Fixed step, discrete	Set in Simulink interface	-
Fixed step size	Set in Simulink interface	-

* In mechanical simulation, depends on the data size could be vary.

**Based on the media in simulation could be vary.

4.1 Definition of Network Performance Parameters

The simulation program records the time delay history of each message and is able to calculate network performance statistics such as the average time delay, efficiency and utilization of the network.

Average Time Delays: Based on the running time, we can calculate the average time delays from each node of network, using the following equation [14]:

$$T_{avg\text{-}delay} = (1/N) \sum_{i \varepsilon N_{node}} \left[\frac{\sum_{j=1}^{M(i)} T_{delay}^{(i,j)}}{M(i)} \right] \qquad \text{Eqn. (4-3)}$$

Where:

N = Total number of nodes,

N_{node} = Set of nodes,

M(i) = The number of message requested at node i

Network Efficiency: The ratio of the total transmitting time to the time used to send messages (including queuing time, blocking time and so on) defines the efficiency of a network [14]:

$$P_{eff} = \frac{\sum i \varepsilon N_{node} \sum_{j=1}^{M(i)} T_{tx}^{(i,j)}}{T_{sum\text{-}delay}} \qquad \text{Eqn. (4-1)}$$

Note that, P_{eff} leans toward one, means that, all the time delays are due the transmission delay and the performance is good. On the other hand, P_{eff} leans toward zero, shows that most of time delay is due to collision and message contention. We will

discuss more about the latency in section 4.2. It is better to notify that T_{tx} is the same as ΔT_3 and $T_{sum\text{-}delay}$ is equal to T_{lat} in Figure 2-21.

Network Utilization: The ratio of the total transmission time to the total running time defines the utilization of network. It also describes the percentage of effective bandwidth used by the nodes [14]:

$$P_{util} = \frac{\sum_{i \varepsilon N_{node}} \sum_{j=1}^{M(i)} \left(T_{tx}^{(i,j)} + T_{retx}^{(i,j)} \right)}{T_{max}} \qquad \text{Eqn. (4-2)}$$

Where:

$T_{retx}^{(i,j)}$ Is the time taken to retransmit the (i , j)th message.

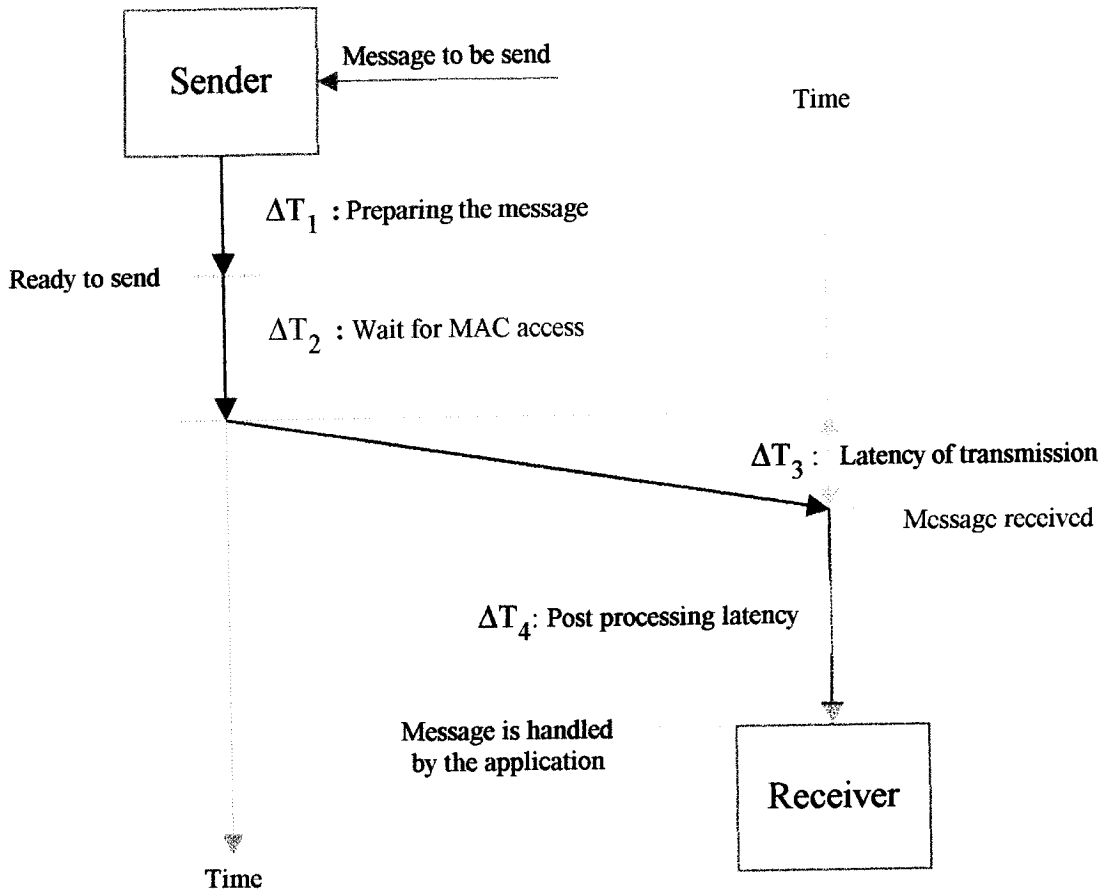
T_{max} Is the total running time.

P_{util} leans toward one, means, network is saturated and we have to redesign the layout or reassign the traffic load. Note that, high load condition can saturate the network. Meanwhile, P_{util} leans toward zero, means, there is sufficient bandwidth left on the network.

4.2 Latency Quantification and Measurement

There are different requirements, which can be defined for real-time communication: receive variance, scan time jitter (See Appendix G), throughput, and latency [16]. In the following, latency quantification is discussed.

In a simplified model, a sender wants to signal an event by sending a message to receiver. The maximal delay of this message, ΔT_{lat} (the latency), is a possible real-time constraint, to be fulfilled by the communication system.



$$\Delta T_{lat} = \Delta T_1 + \Delta T_2 + \Delta T_3 + \Delta T_4$$

Figure 4-1: Latency diagram

As is illustrated in Figure 4-1,the latency is composed of different part delays for different reasons:

1. **Preparation of message:** First the sender has to prepare the message and run through the upper layers of the protocol stack. This takes time ΔT_1, which is equal to, computation time, ΔT_{scomp}, plus the encoding time, ΔT_{code}.
2. **Wait for access:** Once the message is ready to transmit, the sender has to wait until the medium access control (MAC) gets control over the communication system. After ΔT_2 , which is the sum of the queue time, ΔT_{queue}, and the blocking time, ΔT_{block}, the message can be transmitted. The queue time is the time a message waits in the buffer at the source node while previous messages in the queue are sent. It depends on the blocking time of previous messages in the queue, the periodicity of messages, and the processing load. The blocking time is the time a message must wait once a node is ready to send it and includes the waiting time while the other nodes are sending messages and the time needed to resend the messages in case of a collision. As we already know blocking time is a non-deterministic factor because of the BEB algorithm in Ethernet.
3. **Transmission:** The transmission of the information over the network takes ΔT_3 time delay. This includes the total transmission time of a message, ΔT_{frame}, and the propagation delay of the network, ΔT_{prop}. They

both depend on message size, data rate, and the length of the network cable.

4. **Message received:** The message is processed by the receiver and passed to the destination application; we call this post processing time, ΔT_4 , which is the sum of decoding time, ΔT_{decode}, and the computation time, ΔT_{dcomp}, at the destination node.

The latency in this model is the sum of all mentioned delays [14]:

$$\Delta T_{lat} = \Delta T_1 + \Delta T_2 + \Delta T_3 + \Delta T_4 \qquad \text{Eqn. (4-4)}$$

$$= (\Delta T_{scomp} + \Delta T_{code}) + (\Delta T_{queue} + \Delta T_{block}) + (\Delta T_{frame} + \Delta T_{prop}) + (\Delta T_{decode} + \Delta T_{dcomp})$$

Typically, ΔT_1 and ΔT_4 compared with ΔT_2 and ΔT_3 are constant. They are negligible in our simulation since they are a limitation of computer processing parameters rather than network physical and protocol parameters.

Note that within the duration of test or simulation, measured latency is not constant and can be vary from a minimum to a maximum number. The difference between these two numbers is called "Jitter" and is a performance stability factor of the system.

Regarding experiments, which has been done in Control and Information System (CIS) laboratory of Concordia University [37]; identified parameters of components are compared with the results.

Obviously, operating system delay, crossover cable delay, parallel and I/O ports delays involve in the following experiments. (For more information please refer to [37].)

4.3 Network Interface

Some NICs come with software diagnostic tools, which check the operation of NIC and are usually called [38]:

Internal diagnostic test: checks different aspects of the internal hardware of NIC. (Up to the transmit/receive circuitry)

Loop back test: checks the proper functionality of transmits and receives circuitry and is only applicable to 10Base2 media (Coaxial cable), as two terminators with a BNC TEE should be used (Figure 4-2).

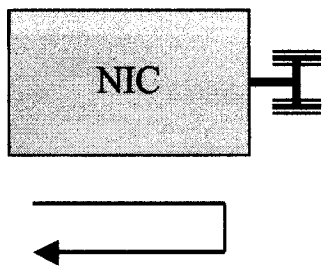


Figure 4-2: Coaxial loop back plug

Echo–Server test: This method is used for the Fast Ethernet NIC latency measurement test in this approach and does not depend on the media type (See Figure 4-3a for the Coax as media). As we use twisted pair cable as medium, two NICs are connected via a crossover cable (Figure 4-3b). One acts as an Echo server and the other, which is under the test is the Echo client. The latter sends a packet to echo server that echoes the packet back.

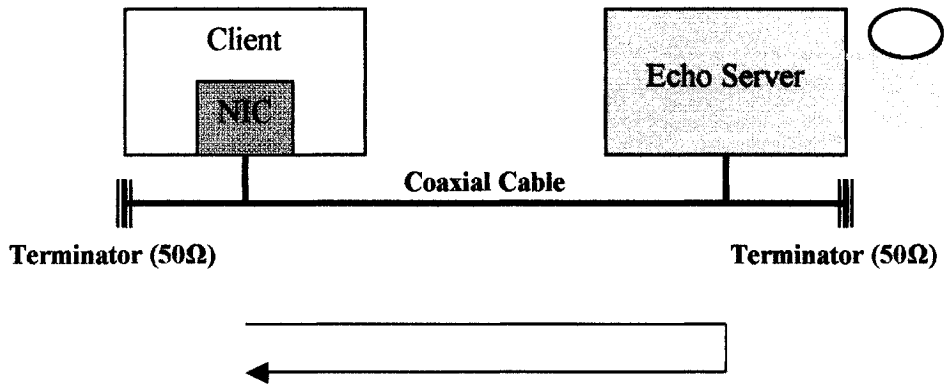


Figure 4-3a: Echo server test (coax medium)

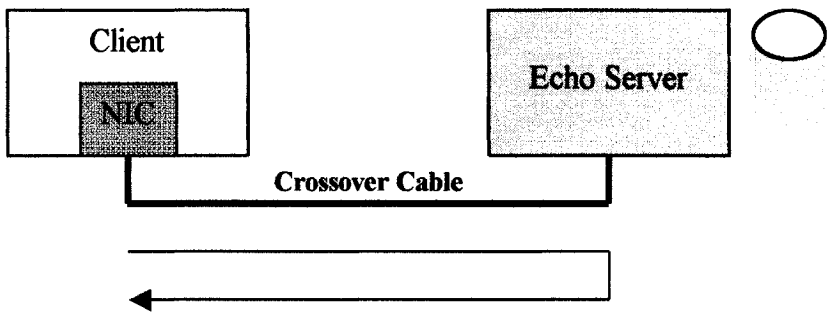


Figure 4-3b: Echo server test (crossover cable)

Crossover cable length is less than 3 m in these tests, so the resulting delay (ΔT_{prop}) is negligible. As we only have two computers, this kind of cable should be applied. We can either build or purchase this cable. The crossover cable is built with the transmit pins on one end of the cable connected to the receive data pins on the other end of the crossover cable, and vice versa. Figure 4-4 shows the crossover wiring required for 10BaseT and 100BaseT systems.

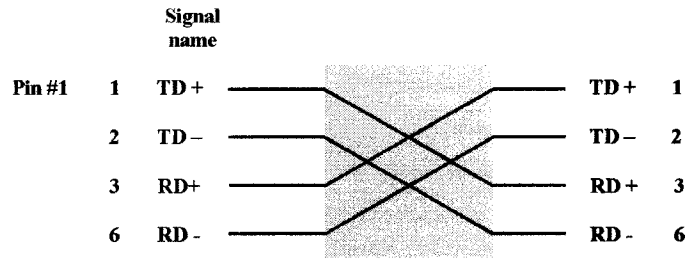


Figure 4-4: 10BaseT and 100BaseT crossover cable wiring [3]

SMC1244 10/100M,Ethernet card that is based on Realtek RTL8139 chip series (Fast Ethernet card) [4], has been tested. Parallel ports are used to measure latency of cards (Figure 4-5).

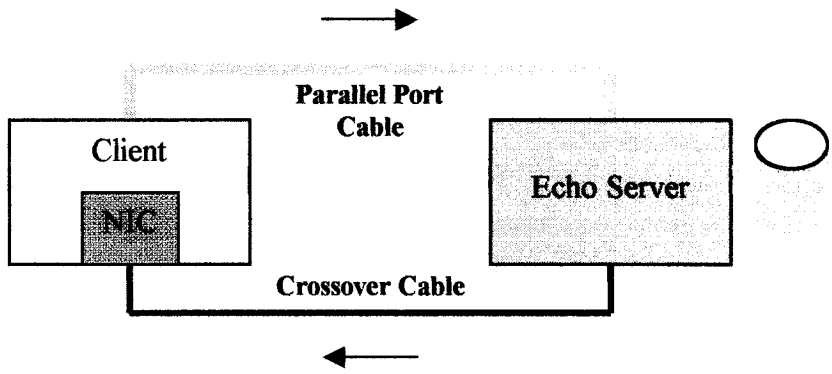


Figure 4-5: Echo server test (crossover cable and parallel port as feedback)

Test 1: Measurement of the message latency (One-way latency), which includes one transmitting and one receiving latencies by using parallel port as feedback (Figure 4-5):

- Operation procedure:
  - Step1: Server reads the clock, gets time T_1 and sends a packet to the NIC. Client is polling the NIC register, waiting for the packet.

- Step2: Server is polling parallel port. Client receives the packet and sends a high voltage signal to the parallel port.
- Step3: Server receives the signal and reads clock again, gets time T_2.

Table 4-3: Latency definitions for test 1

LATENCY	DEFINITION
T_{dclock}	**Parallel port time delay of clock reading**
T_{dtx_NIC}	**NIC time delay of Tx**
T_{drx_NIC}	**NIC time delay of Rx**
T_{dtx_p}	**Parallel port time delay of Tx**
T_{drx_p}	**Parallel port time delay of Rx**
T_{dround}	**Round trip time delay**

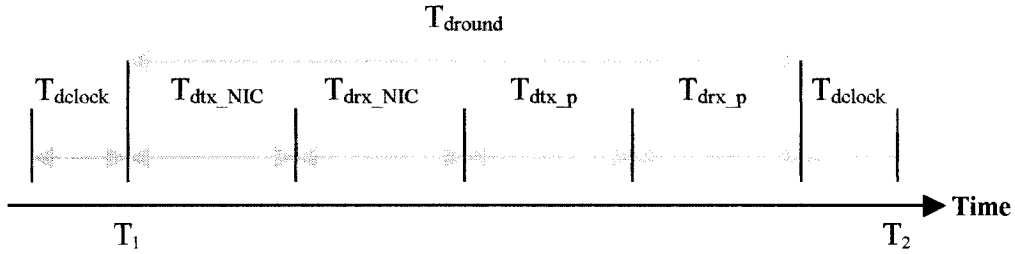


Figure 4-6: Latency definitions for test 1

Note that, length of the network cable and parallel port cable; both are less then 3m, so the resulting latency (ΔT_{prop}) by medium is negligible. This test has been repeated in different hardware and packet size combinations.

Figure 4-6 and Table 4-3 illustrate the definition of our latencies in this test. The hardwares used in this test are listed in Table 4-4.

Table 4-4: Hardware for test1

NIC	SMC1244 10/100M
Medium Type	Crossover cable
Cable	C5e Network cable

The following graph (Figure 4-7) shows the one way measured latencies in this test corresponding to the packet sizes. (For the NIC round trip latency test, see Appendix J.)

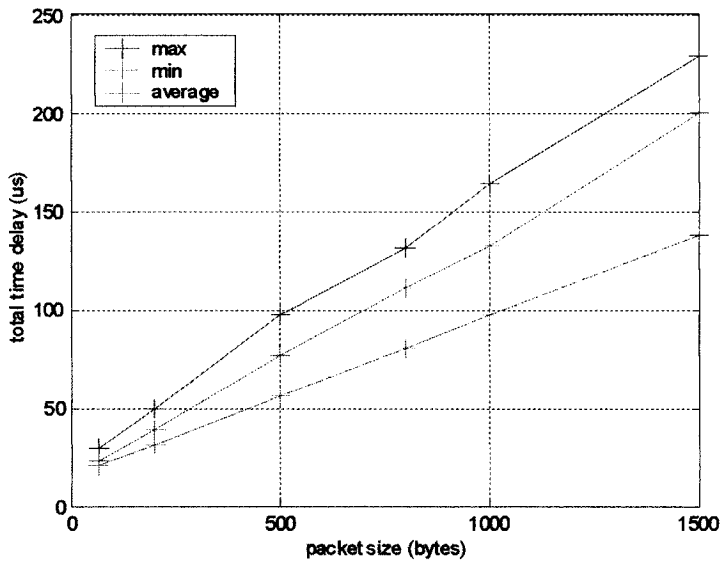


Figure 4-7: Test 1 - Crossover cable plus parallel port feed back

Test 2: Measurement of the transmitting latency by polling a specific register (Ethernet NIC transmit latency test). Results can be seen in Figure 4-8. Transmit registers are used to test the packet sending time delay. The RTL8139 NIC has registers to show if

the packet is totally sent out. If we keep polling this register, we could get the time when transmit is finished.

- Operation procedure:
  - Step1: send out 1 to parallel port.
  - Step2: send out packet to the NIC and check the status register until the packet is totally sent out. This check blocks the computer until it finishes.
  - Step3: send out 0 to parallel port.
  - Step4: wait for amount of time. (300μs)

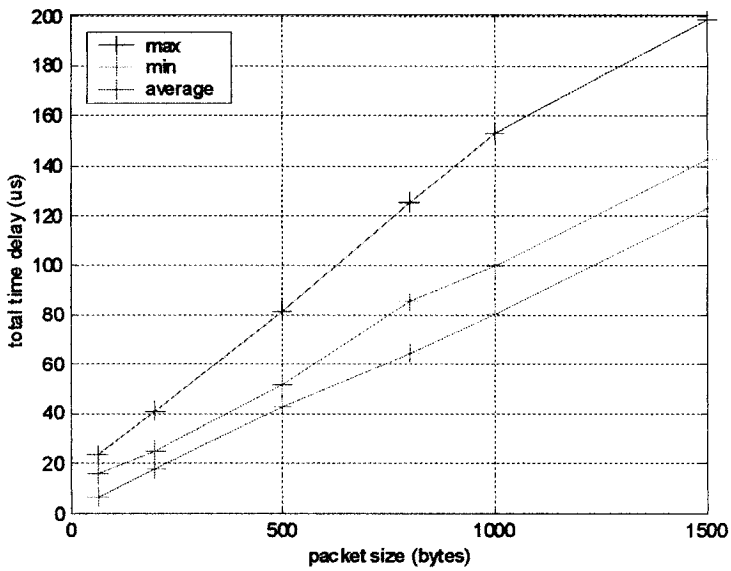


Figure 4-8: Summery of Tx time delay test

Figure 4-9, shows pulses that generated by transmission of 1500 byte size packets and have been observed by Oscilloscope. The high level period is the packet transmission time, which is about 125μs.

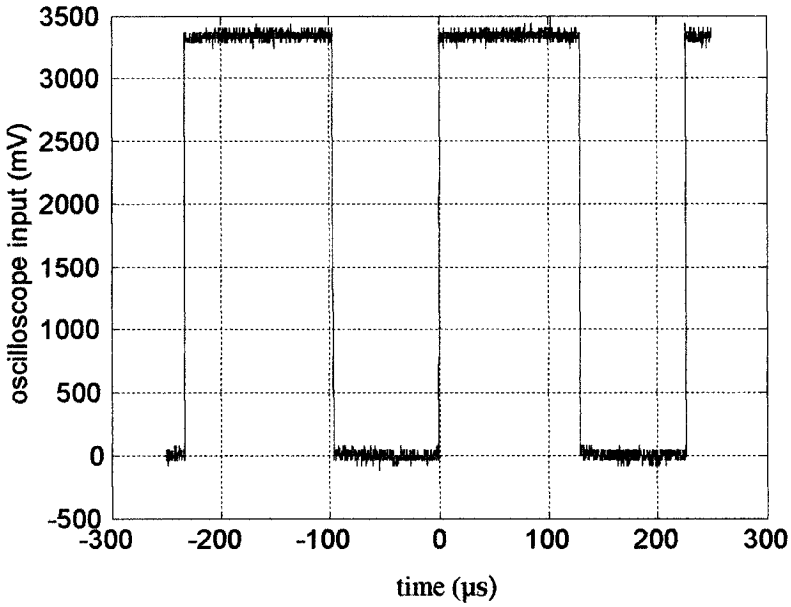


Figure 4-9: Observation of packet sending time

Test 3: Ethernet NIC driver buffer the packets prior of their transmission by NIC. In case of exceeding number of packets from the buffer size limit, some of the packets will be lost. To avoid of this problem, there is a minimum time interval for putting the packets in buffer, which should be theoretically equal to sending them out on the network. Test 3 measures this value or in other words measures how close packets could be sent out which can be a factor for data rates of a card.

This test implemented by two nodes, which are connected with a crossover cable. Note that two operating system (Windows and RTX) has been applied in this experiment. Test has been repeated with different packet sizes.

- Operation procedure:

Server:

- Step1: Server sends a packet.
- Step2: wait a time Tr

Client:

- Client keeps receiving.

Note that receive NIC is an integrated Intel 10/100.

Table 4-5: Minimum time space between two packets (Test 3: results)

Packet Size (Byte)	Theoretical minimum time space (µs)	Tr Test result (µs)
1500	120	128
1000	80	78
64	5.12	1

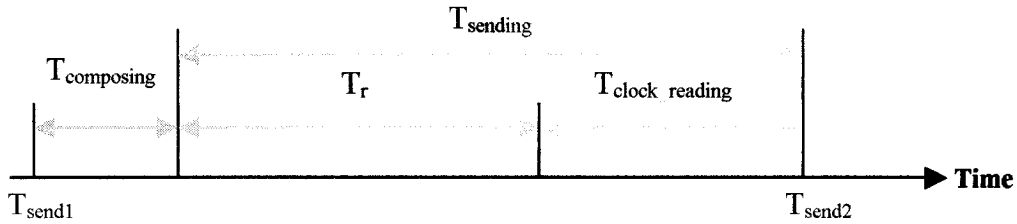


Figure 4-10: Latency definitions for Test 3

Table 4-5 is a comparison of the theoretical with test result of minimum time interval between two packets. Considering the clock reading time, which is around 5µs,test result is about the theoretical packets sending time.

Note that in this approach, regarding the scope, the only dedicated latency in NICs is the recent one (minimum time interval).

4.4 Hub

In order to test the latency in hub two computers connected via hub using twisted pair, type C5e as medium and parallel ports as feed back. Cables are short enough to neglect the resulting delays by them. In the following, these results for two different brand name models have been brought. This test has been repeated for different packet sizes.

- Operation procedure:
  - Step1: Server reads the clock, gets time T_1 and sends a packet to the NIC. Client is polling the NIC register, waiting for the packet.
  - Step2: Server is polling the NIC register. Client receives the packet and sends back the packet to server.
  - Step3: Server receives the packet and reads clock again, gets time T_2.

Definitions of the latencies in this test are illustrated in Table 4-6 and Figure 4-11.

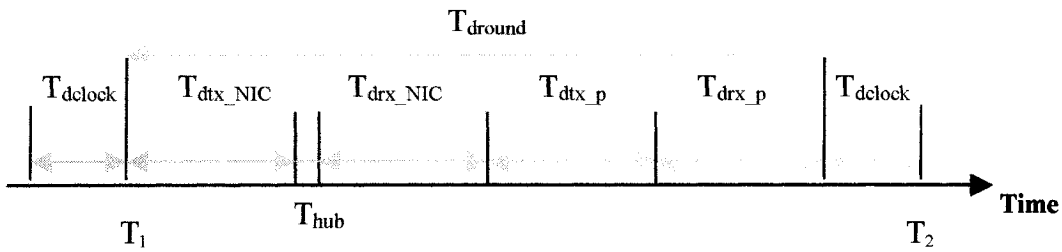


Figure 4-11: Latency definitions for the Hub test

Table 4-6: Latency definitions for the Hub test

LATENCY	DEFINITION
T_{dclock}	**Time delay of clock reading**
T_{dtx_NIC}	**NIC time delay of Tx**
T_{drx_NIC}	**NIC time delay of Rx**
T_{dround}	**Round trip time delay**
T_{hub}	**Time delay of hub**
T_{dtx_p}	**Parallel port time delay of Tx**
T_{drx_p}	**Parallel port time delay of Rx**

Obviously to find out about the pure delay caused by the hub, results in Figures 4-12 and/or 4-13 should be subtracted from Figure 4-7. For instance, average latency for a 1000 bytes packet size is about 132μs in Figure 4-7 and 142μs in Figure 4-12 while the average latency for the same packet size is 137 μs in Figure 4-13. So we have 5 to 10μs measured latency for hub in two different models in average.

Considering the maximum and minimum curves in those mentioned graphs and their differences, hub's latencies have tendency to less than microseconds, which is negligible and also reasonable with theoretical values.

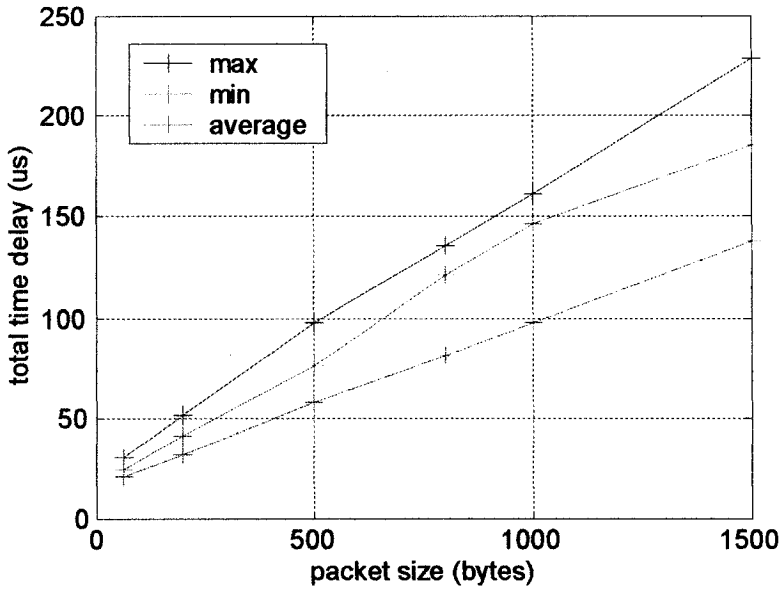


Figure 4-12: 3COM Office Connect 10/100 Dual Speed Hub 8 time delay test

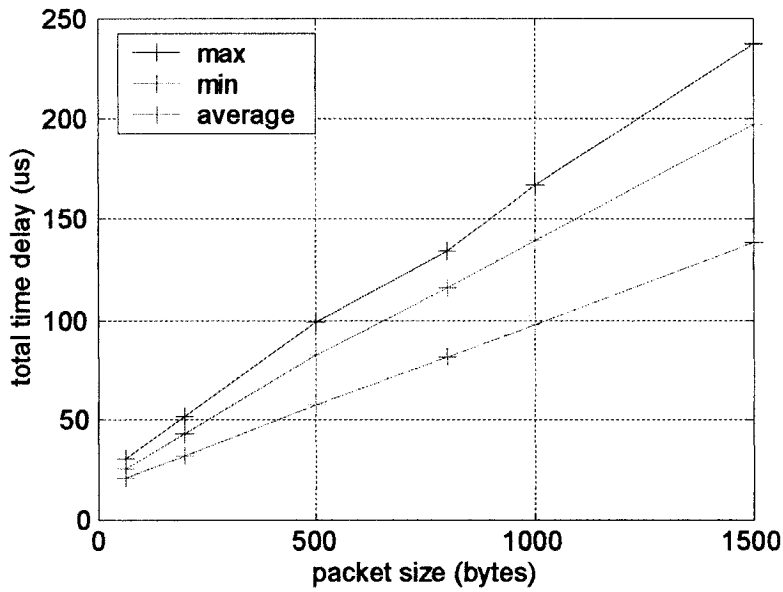


Figure 4-13: NETGEAR DS108 Dual Speed Hub time delay test

Designated latencies for the hub in this approach only come from propagation delays in ports (Appendix E) and they are also in nanoseconds range based on 10Mbps media, which may slightly change for 100Mbps.

Theoretically, hubs have no major effects on the total latencies on the networks and data communications because of its functionalities and fabric. Latency is always about nanoseconds [42]; hence there is almost no major discrepancy between the model and real application and no reason to modify the codes.

4.5 Switch

Operation procedures and physical connections to measure the latencies in Switch are similar to the hubs (Section 4-4). Definitions of the latencies have been mentioned in Table 4-7 and Figure 4-14. (Test: 10000 cycles, takes time about 30000000.0 μs)

Table 4-7: Latency definitions for the Switch test

LATENCY	DEFINITION
T_{dclock}	**Time delay of clock reading**
T_{dtx_NIC}	**NIC time delay of Tx**
T_{drx_NIC}	**NIC time delay of Rx**
T_{dround}	**Round trip time delay**
T_{switch}	**Time delay of switch**
T_{dtx_p}	**Parallel port time delay of Tx**
T_{drx_p}	**Parallel port time delay of Rx**

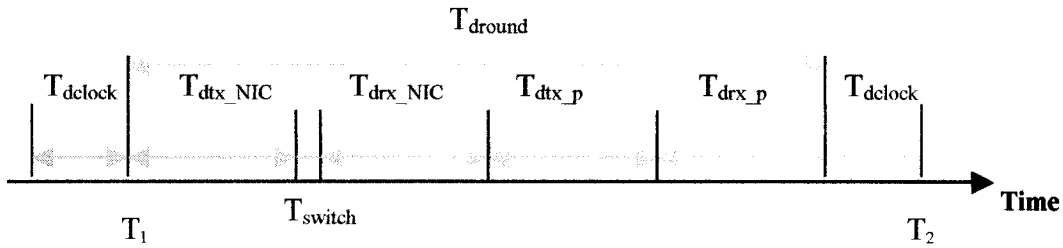


Figure 4-14: Latency definitions for the Switch test

In the following, the results for two different brand name models have been brought:

One more time, using Figure 4-7 can help us to find the pure time delay of switch. Assuming a 1000 bytes packet size, from Figure 4-7,average time delay is about 132μs. In Figure 4-15 we have 224μs latency for this size of packet.

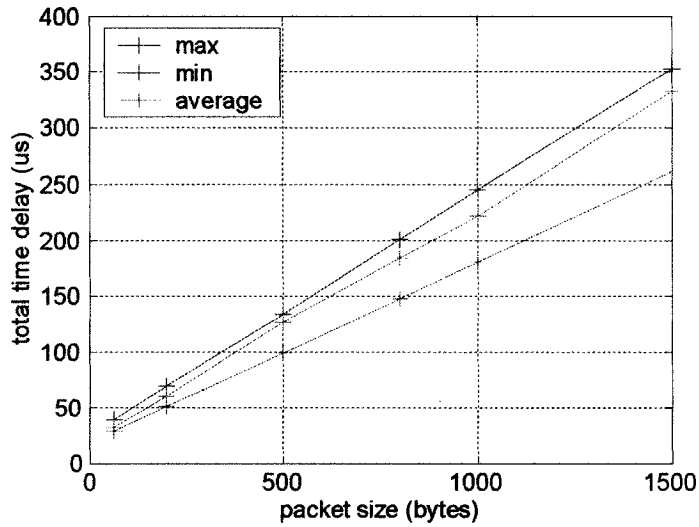


Figure 4-15: 3COM 3C16477 Baseline Gigabit Switch time delay test

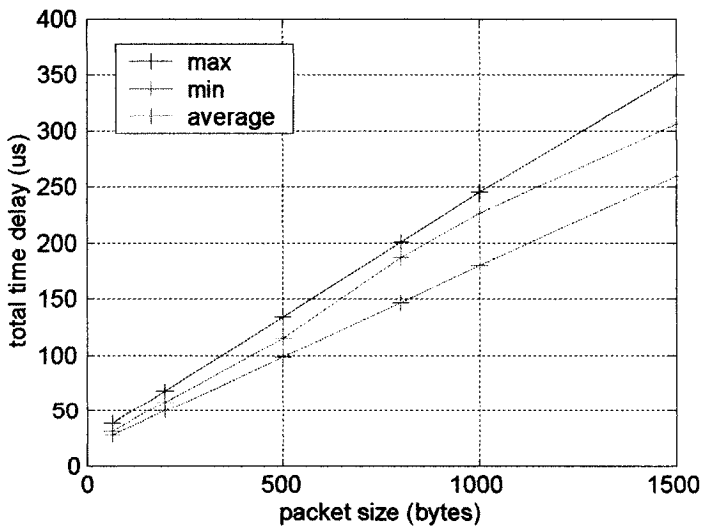


Figure 4-16: NETGEAR FS108 Fast Ethernet Switch time delay

The difference or the switch delay is about 92μs. If the switch works in store and forward mechanism it should store the whole packet prior to transmit. Therefore we have a time delay equal to the packet size multiply by BT, which are 80μs for 1000 bytes.

Rest of the delay, which in average is about 12μs, belongs to the other functions in switch like, CRC error checking, spanning tree and address learning (Section 3.5.1). As it is seen in experiments and their graphs, this number varies between about 3 to 14 μs for different frame size. For sure it also depends on the hardware and/or operating system. In order to achieve higher accuracy and/or reliability in the switch model, constant latency of 9μs, considered and will be added to the model. We named the most important general parameters involved in simulation and discussed the definition of performance parameters of a network, which will be used, in the next chapter.

The bottom line is, except for the switch in which minor modification in designated latency is necessary, other component's parameters have reasonable values.

CHAPTER 5: MODEL TESTING AND VERIFICATION

In the previous chapter, first we identified the parameters involved in this study and then compared them with experimental data, helping to test and in one case modifying the models.

The following chapter, testing of model, will be done through the interconnection of components in different conditions. Note that, the obtained results, which will be discussed in this chapter, are only aim to illustrate the behaviour of this simulator and not to performance measurements of a network system.

In order to test our model, we consider half duplex Fast Ethernet (100Mbps) control network as a communication backbone for a networked control system connecting sensors, actuators, and other controllers. The key parameters of the corresponding network will be studied when used in a control situation, including network utilization and efficiency, magnitude of the expected time delay, and characteristics of time delays. Simulation results are presented for several different scenarios and/or topologies through three tests, and the network performances will be discussed as well.

All the simulation tests have used the same zero releasing policy. Three releasing policies could be considered:

- **Zero releasing policy:** Assumes every node tries to send first messages at the start of simulation (t = 0) and send a new message every period.

- **Random releasing policy:** Assumes a random start time for each node, but still each node sends a new message every period.
- **Scheduled releasing policy:** The start sending time is scheduled to occur (to the extent possible) when the network is available to the node; this occurs in a polled connection. [14]

Zero releasing policy has been chosen for these tests, as it has the worst contention within the above policies. This happens when a system powers up and there is no pre-scheduling of messages or when a strobe requests from the master. Period is constant within the tests and it is equal to BT.

In designing a networked control system, both the effective bandwidth and the sensors' sampling rate must be considered. Although high sampling rates improve the control performance in traditional control systems, they also induce high traffic loads on the network medium.

Though we have a high sampling rate, number of messages during the simulation time has been limited to lower the traffic loads on the network medium and for the stability reason and to escape from the intensive channel capture issue. The number of messages in the queue of each node defines the stability of a network itself. If this number is larger than a certain constant or tends to infinity as time increases, the network is said to be unstable (even though we assume infinite buffer size). We have already faced this problem when we tested our model in chapter 3 (section 3.6). By looking at the Figures 3-31 and 3-33, we will find out most of the transmission within the time of simulation concerns to one or two specified sources. It means that some nodes capture the

channel for the most of a limited simulation time. This causes less chance for the other nodes to transfer more messages within that time, and increases the time delay and therefore degrades the control system performance sufficiently, which causes the instability of the control network system. However, note that it is possible to have a system with an unstable network but a stable control system and vice versa. [26]

As we have already mentioned in chapter 3, the BLAM mechanism could be a fix for this issue by changing the back off algorithm to make fairer access to the channel, but IEEE has not decided to standardize it yet.

Table 5-1 includes the whole common parameters within the following tests in this chapter.

Table 5-1: Parameters for all tests in chapter 5

OPERATION MODE	**Half Duplex**
MAU TYPE	**100BaseT**
***CABLES LENGTH (m)**	**50**
BT (μs)	**0.01**
SLOT TIME (μs)	**5.12**
IFG (μs)	**0.96**
SIMULATION TYPE	**Fixed Step, Discrete (No Continuous States)**
SAMPLE TIME (BT)	**1**
RECEIVING BUFFER SIZES	**500**
DATA RATE (Mbps)	**100**

*Length of the cable, connected from each node (NIC) to Switch.

Note, each test run several times within the reasonable simulation time to achieve the accurate average results (Jitters for these tests can be seen in Appendix H).

5.1 Test 1

Changing the number of nodes in network can affect the performance of the control system. Looking at the following graphs proves this matter.

Table 5-2 includes the exclusive parameters for test 1:

Table 5-2: Parameters for Test 1

SIMULATION TIME (BT)	**180000**
NUMBER OF MESSAGES PER NODE	**10**
SAMPLE TIME (BT)	**1**
NUMBER OF NODES	**2,4,6,8**
FRAME SIZE (BYTE)	***72**

*Preamble included.

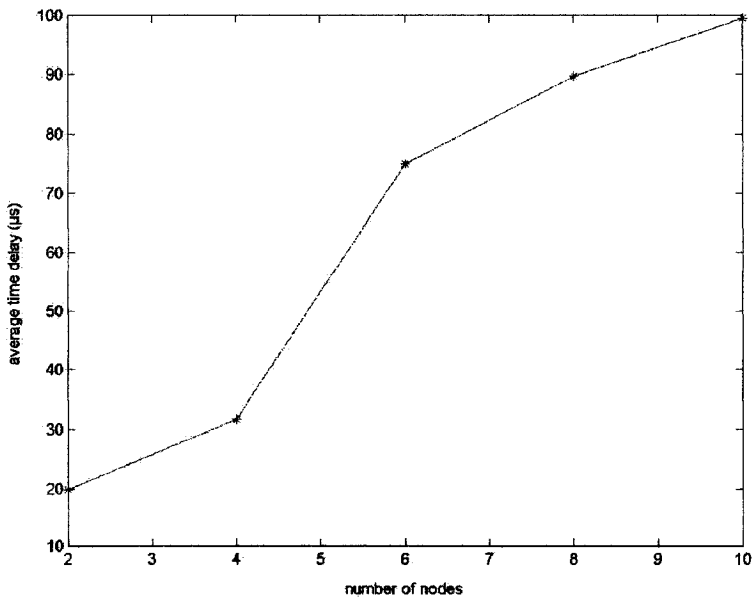


Figure 5-1: Test 1 - Average time delay versus number of network nodes

As it is seen in Figure 5-1, increasing in number of nodes causes higher traffic load and consequently higher latency. We can also see in Figure 5-2 that increase in number of nodes incline the network efficiency and utilization to higher values.

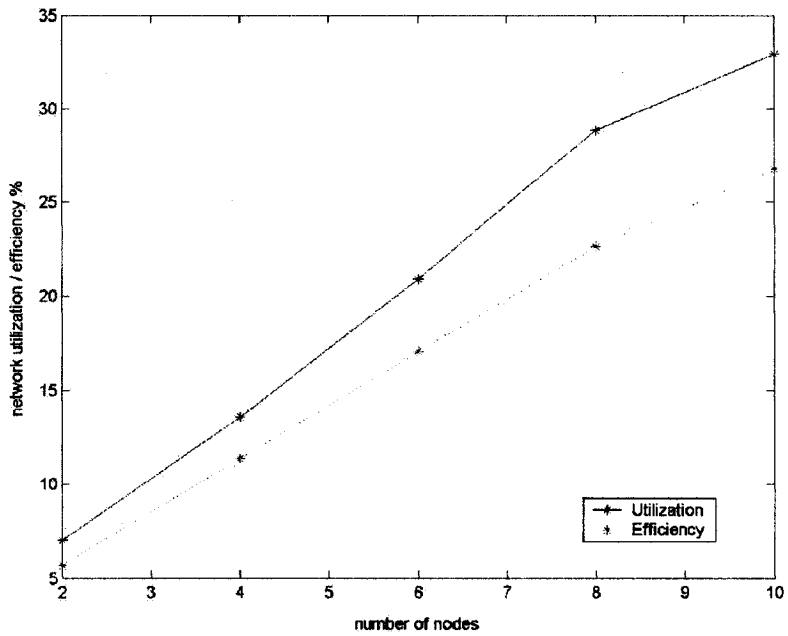


Figure 5-2: Test 1 - Network utilization/efficiency versus number of network nodes

Note that simulation time has been estimated and chosen for this test somehow to cover transmission of the whole designated number of messages for a 10-node network and has been kept for the rest.

5.2 Test 2

In this test, number of messages transmitted within the simulation time change and the affect of this change on network parameters can be seen in Figures 5-3 and 5-4.

Table 5-3 includes the exclusive parameters for test 2:

Table 5-3: Parameters for Test 2

SIMULATION TIME (BT)	***150000**
NUMBER OF MESSAGES PER NODE	**5,10,15,20,25,30,35**
SAMPLE TIME (BT)	**1**
NUMBER OF NODES	**4**
FRAME SIZE (BYTE)	****72**

*Time simulation for NUMBER OF MESSAGE PER NODE = 30, 35 is equal with 300000 BT
**Preamble included.

Note that simulation time has been estimated and chosen for this test somehow to cover transmission of the whole designated maximum number of messages for a 4-node network and has been kept for the others.

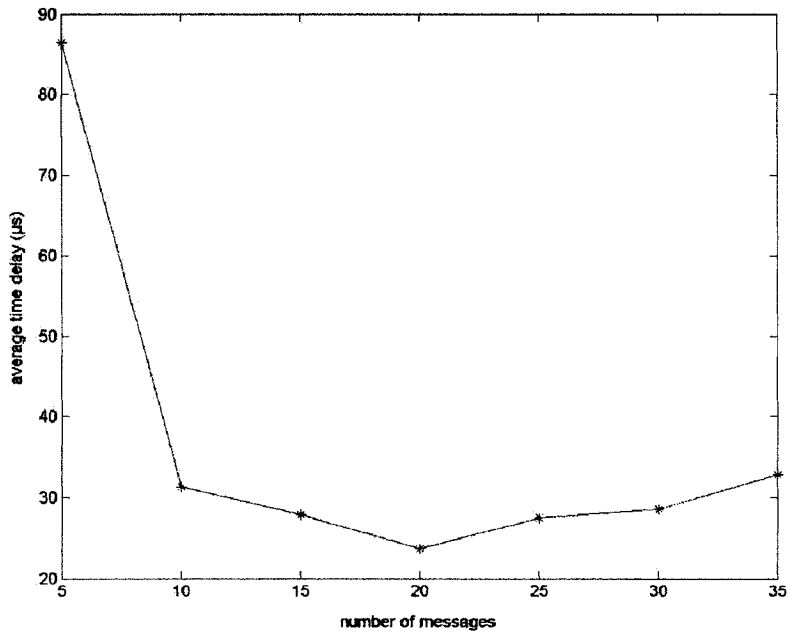


Figure 5-3: Test 2 - Average time delay versus number of messages per node to transmit

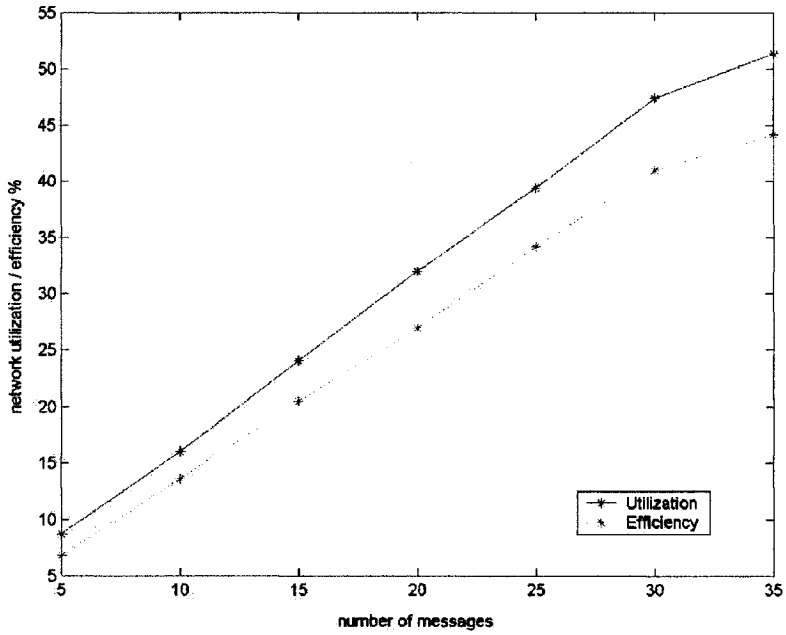


Figure 5-4: Test 2 - Network utilization / efficiency versus number of messages per node

As it is seen, increasing the number of messages buffered to send from each node decreases the latency for a while and then after getting to a turning point slightly increases. In addition, efficiency and utilization of network within a limited run time increases with growing the number of messages as we expect.

5.3 Test 3

Effect of frame size on network parameters for two different networks, one with two nodes and the other with four nodes, is discussed next.

5.3.1 Test 3.1: Two Nodes

For the two-node system we use the parameters in Table 5-4:

Table 5-4: Parameters for Test 3.1

SIMULATION TIME (BT)	250000
NUMBER OF MESSAGES PER NODE	5
SAMPLE TIME (BT)	1
NUMBER OF NODES	2
FRAME SIZE (BYTE)	* 72,500,1000

*Preamble included.

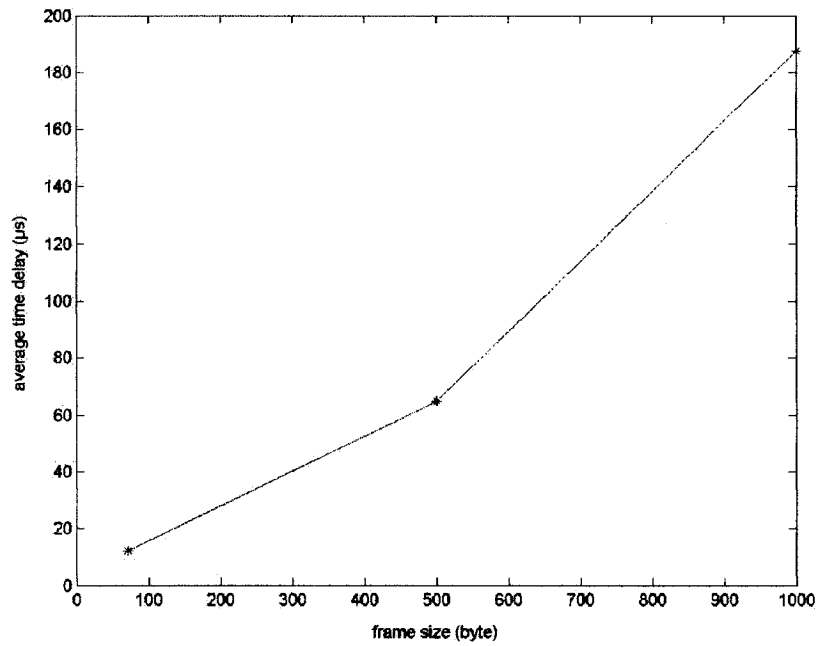


Figure 5-5: Test 3 - Average time delay versus frame size for a two-node network

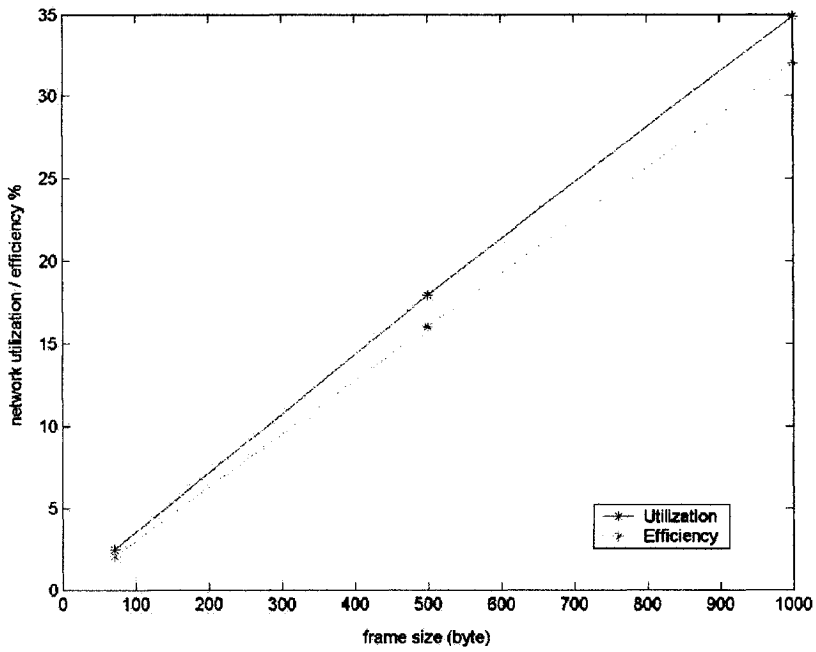


Figure 5-6: Test 3 - Network utilization / efficiency versus frame size for a two-node network

5.3.2 Test 3.2: Four Nodes

System parameters for a test with four-node, have been brought in Table5-5.

Table 5-5: Parameters for Test 3.2

SIMULATION TIME (BT)	**520000**
NUMBER OF MESSAGES PER NODE	**5**
SAMPLE TIME (BT)	**1**
NUMBER OF NODES	**4**
FRAME SIZE (BYTE)	*** 72,500,1000**

*Preamble included.

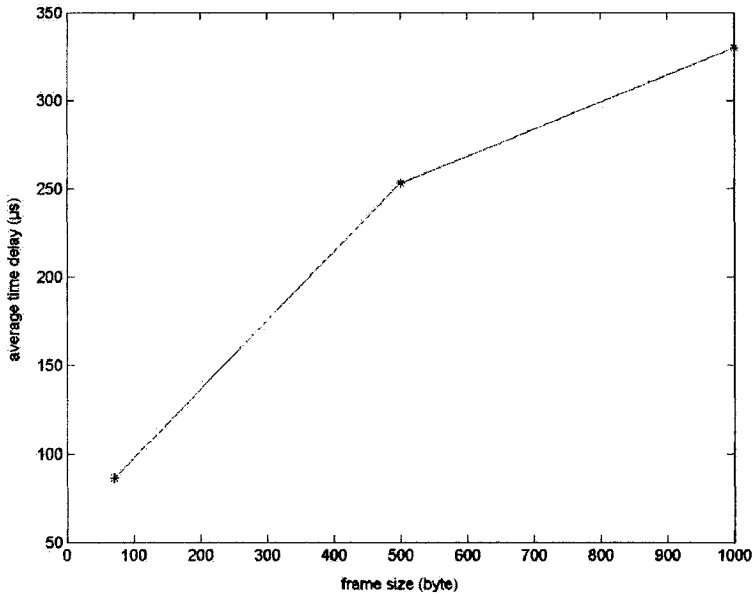


Figure 5-7: Test 3 - Average time delay versus frame size for a four-node network

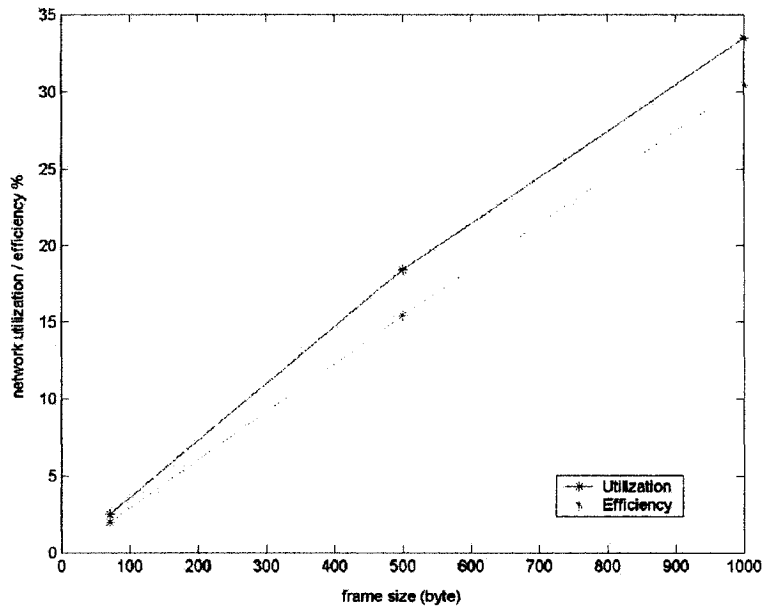


Figure 5-8: Test 3 - Network utilization / efficiency versus frame size for a four-node network

As we expect, both tests, for 2 and 4 nodes, show that by increasing the frame size we have higher latency, network efficiency and utilization. Figure 5-9 illustrates the comparison of latency in 2 and 4-node network for different frame sizes and effect of traffic load in this aspect.

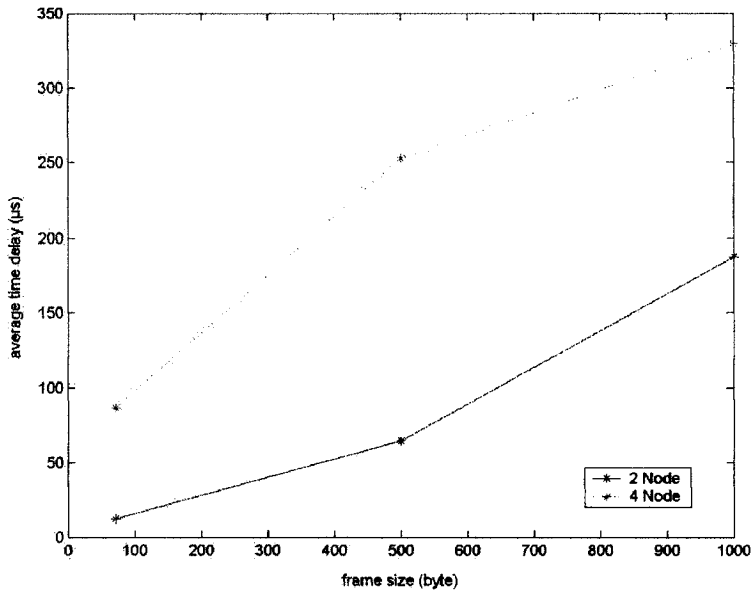


Figure 5-9: Test 3 – Average time delay versus frame size comparison for 2 and 4 node in networks

Note that for utilization comparison, the simulation time directly involves the result and should be equal in different tests. However it is not completely legitimate, it can be seen in Figure 5-11 that accidentally both curves, are really close. As from the previous test (Figure 5-2), we already know that within a constant frame size, number of nodes can affect the utilization. The same discussion stands for efficiency (Figure 5-10). Though in this case, the simulation time is not directly involved in efficiency computation.

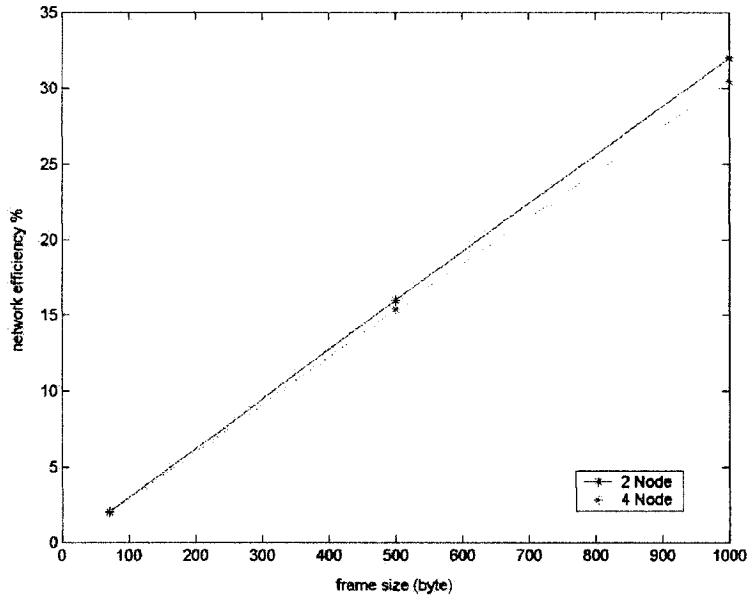


Figure 5-10: Test 3 – Network efficiency delay versus frame size comparison for 2 and 4 node networks

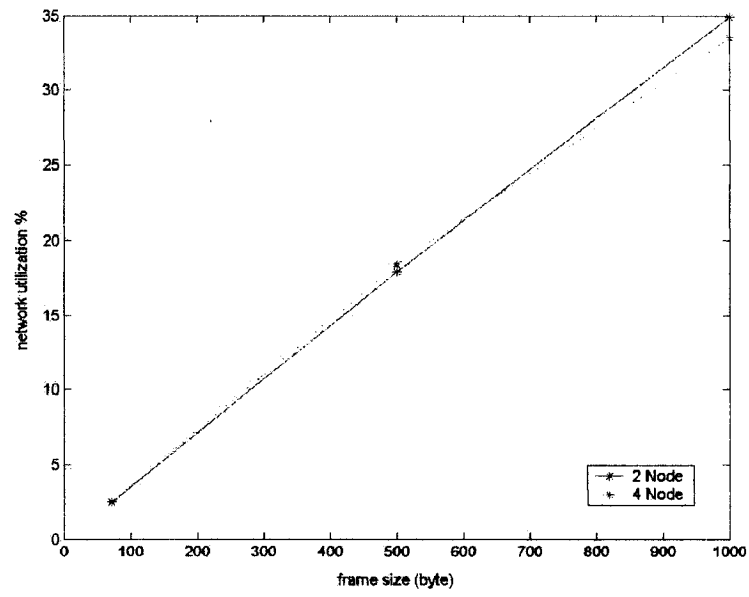


Figure 5-11: Test 3 – Network utilization versus frame size comparison for 2 and 4 node networks

In Test 1, increasing the number of node, and Test 2, number of messages in each node, challenged the performance of network. Frame sizes are changed in Test 3 and we monitored the performance of system while we repeat this test for 2 and 4 number of node in system.

Performance evaluation of the model through these three tests and the expected and reasonable results prove the reliability of this model for other applications of a distributed network control system, and lead us to study their behaviours in different circumstances and topologies.

In the next chapter, testing of our model will be done while we simulate interconnecting of a mechanical system via an Ethernet network as a backbone, using switch in a Full Duplex mode data communication.

CHAPTER 6: SIMULATION OF NETWORKED MECHANICAL SYSTEMS

Mechanical systems and subsystems are mostly formulated in dynamic equations of motion and solved by advanced numerical techniques. Nowadays, the attentions are drawn to parallel computation. This causes applications on supercomputing facilities and distributed network implementation, as a potential to enhance the performance.

In this chapter, the application of our model in distributed control systems will be discussed. We start with component's models on a single processor, then divide the system into multiple subsystems and run them simultaneously on different processors (Nodes) which provides the computational power required to run complex models in real-time.

Mass spring as a simplest mechanical system can be the best example in this category. Though, a simple mechanical system does not illustrate the computational power as a virtue of a distributed system but help us to show the functionality of our model by resorting to simplicity.

Switched Ethernet and its merits in industrial applications already are discussed in previous chapters. Switched networks employ either twisted pair or fibre optic cabling.

Following simulations are based on Ethernet with 10 Mbps using twisted pair cables.

6.1 Dynamic Equations of Motion

Figure 6-1 shows the mechanical model, studied in this thesis.

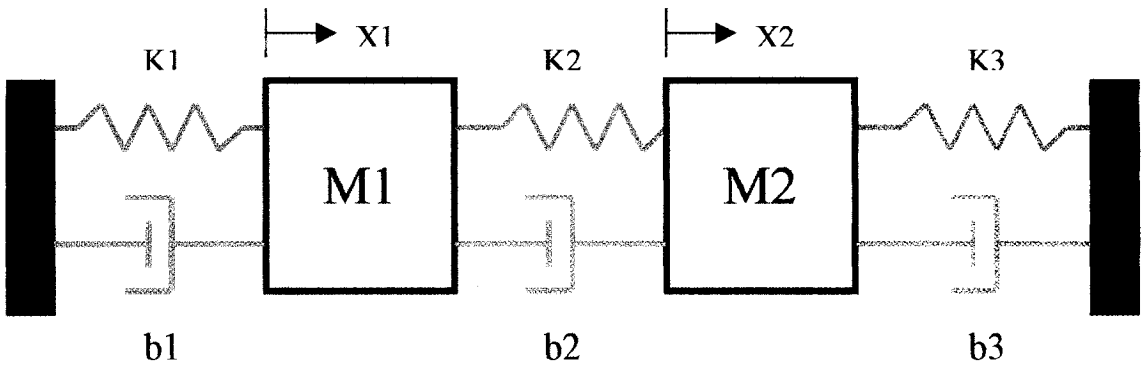


Figure 6-1: Mass spring system

Having the illustrated system in Figure 6-1 in hand, we can draw the free body diagrams for each mass in Figures 6-2a and 6-2b.

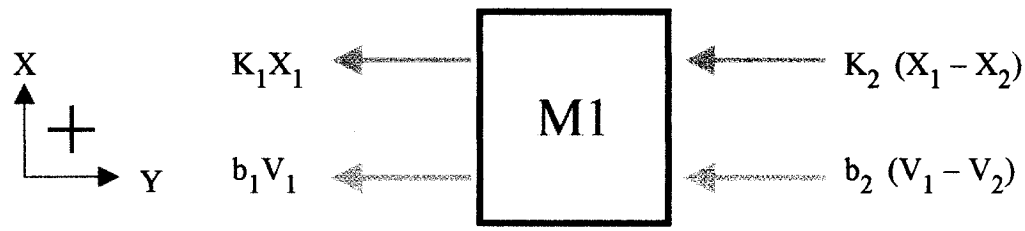


Figure 6-2a: Free body diagram for M1

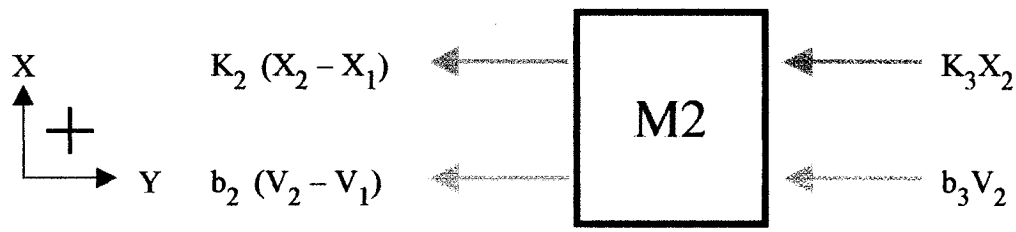


Figure 6-2b: Free body diagram for M2

According to the free body diagrams illustrated in Figures 6-2a and 6-2b, we have the following dynamic equations of motions:

$$(1)\quad \begin{cases} M_1\dot{V}_1 = -K_1X_1 - K_2(X_1 - X_2) - b_1V_1 - b_2(V_1 - V_2) & \text{Eqn. (6-1a)} \\ \dot{X}_1 = V_1 & \text{Eqn. (6-1b)} \end{cases}$$

$$(2)\quad \begin{cases} M_2\dot{V}_2 = -K_3X_2 - K_2(X_2 - X_1) - b_3V_2 - b_2(V_2 - V_1) & \text{Eqn. (6-2a)} \\ \dot{X}_2 = V_2 & \text{Eqn. (6-2b)} \end{cases}$$

Combining (1) and (2) results in the following state space equation (Eqn. 6-3,6-4):

$$\begin{bmatrix} \dot{X}_1 \\ \dot{V}_1 \\ \dot{X}_2 \\ \dot{V}_2 \end{bmatrix} = \begin{bmatrix} 0 & \overbrace{1}^{A_1} & 0 & \overbrace{0}^{B_1} \\ -(K_1+K_2)/M_1 & -(b_1+b_2)/M_1 & K_2/M_1 & b_2/M_1 \\ 0 & 0 & 0 & 0 \\ K_2/M_2 & b_3/M_2 & \underbrace{\quad}_{A_2} & \underbrace{\quad}_{B_2} \end{bmatrix} \begin{bmatrix} X_1 \\ \dot{X}_1 \\ X_2 \\ \dot{X}_2 \end{bmatrix} \quad \text{Eqn. (6-3)}$$

$$Y = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} X_1 \\ \dot{X}_1 \\ X_2 \\ \dot{X}_2 \end{bmatrix} \quad \text{Eqn. (6-4)}$$

In order to have a discrete system we write the following equations:

$$(Z_1(k+1) - Z_1(k)) / \Delta t = \dot{Z}_1 = A_1 Z_1(k) + B_1 Z_2(k) \qquad \text{Eqn. (6-5)}$$

$$(Z_2(k+1) - Z_2(k)) / \Delta t = \dot{Z}_2 = A_2 Z_2(k) + B_2 Z_1(k) \qquad \text{Eqn. (6-6)}$$

Solve the above equations:

$$Z_1(k+1) = A_{1d} Z_1(k) + B_{1d} Z_2(k) \qquad \text{Eqn. (6-7)}$$

$$Z_2(k+1) = A_{2d} Z_2(k) + B_{2d} Z_1(k) \qquad \text{Eqn. (6-8)}$$

In which:

$$A_{1d} = I + A_1 \Delta t \qquad \text{Eqn. (6-9a)}$$

$$A_{2d} = I + A_2 \Delta t \qquad \text{Eqn. (6-9b)}$$

$$B_{1d} = B_1 \Delta t \qquad \text{Eqn. (6-10a)}$$

$$B_{2d} = B_2 \Delta t \qquad \text{Eqn. (6-10b)}$$

6.1 Simulation Using One Processor

In this stage, equations of motion are computed in only one computer using the parameters mentioned in Table 6-1 to study the behaviour of mass-spring system. Equation for each mass, inserted in one state space block, communicates with each other directly. This procedure is illustrated in Figure 6-3. Simulation parameters have been brought in Table 6-2. Output results for each mass is shown in Figure 6-4.

Note that, only the communication aspect of this system is considered in this study and mechanical behaviour of system is not in our scope.

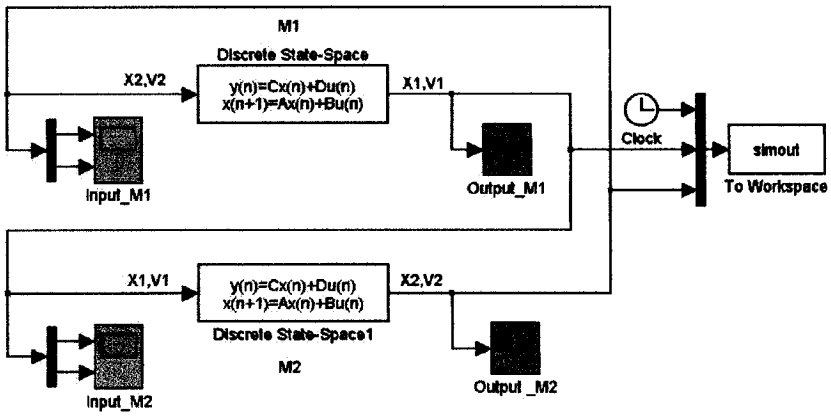


Figure 6-3: Simulation configuration of mass-spring system, simulation on one computer

Table 6-1: Mechanical system parameters

K_1 (N/m)	20000
K_2 (N/m)	40000
K_3 (N/m)	80000
b_1(s.N/m)	6
b_2(s.N/m)	8
b_3(s.N/m)	10
M_1(Kg)	0.004
M_2(Kg)	0.008
Δt (s)	0.1e-6
X_{01}(m)	4
X_{02}(m)	3
V_{01}(m/s)	4
V_{02}(m/s)	3

Table 6-2: Simulation parameters

OPERATION MODE	**Full Duplex**
MAU TYPE	**10BaseT**
***CABLES LENGTH (m)**	**50**
BT (μs)	**0.1**
SLOT TIME (μs)	**51.2**
IFG (μs)	**9.6**
SIMULATION TYPE	**Fixed Step, Discrete (No Continuous States)**
SAMPLE TIME (BT)	**1**
NUMBER OF NODES	**2**
RECEIVING BUFFER SIZES	**100**
TRANSMISION BUFFER SIZES	**100**
****FRAME SIZE (byte)**	**100**
DATA RATE (Mbps)	**10**
NUMBER OF SWITCH'S PORT	**8**

*Length of the cable, connected from each node (NIC) to Switch for the test in clause 6.3.

**Preamble included.

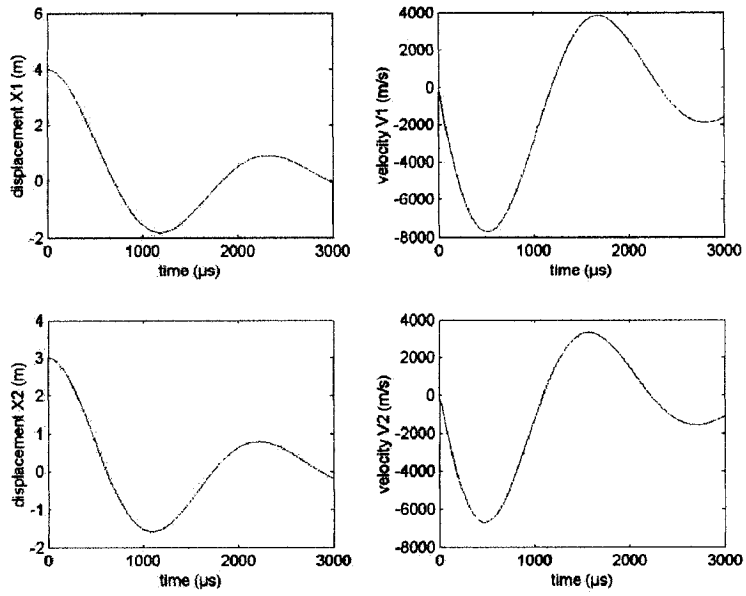


Figure 6-4: Simulation of mass-spring model

6.3 Distributed Simulation

In a real-time Ethernet network, equations can be put into two different computers, which are connected together. They built a small network in which data can transfer. If the two computers are synchronized well; results should be the same as for one computer. This architecture can be expended to large systems. Simulation of the mechanical system proves this fact. Each discrete state- space equation connects to a NIC and transfers the data via a switch in full duplex Ethernet mode to another node. Note that mechanical and simulation parameters are the same as the last experience in Section 6-2. (Table 6-1 and Table 6-2)

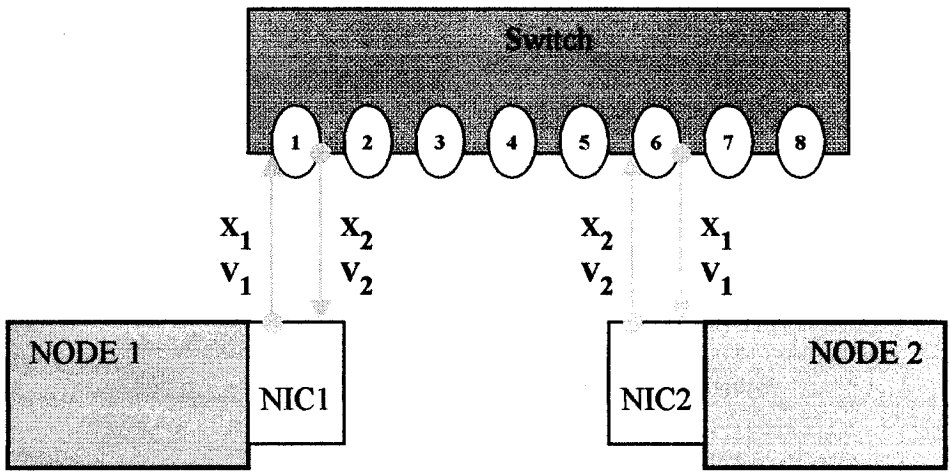


Figure 6-5: Schematic connection of nodes and switch in the model

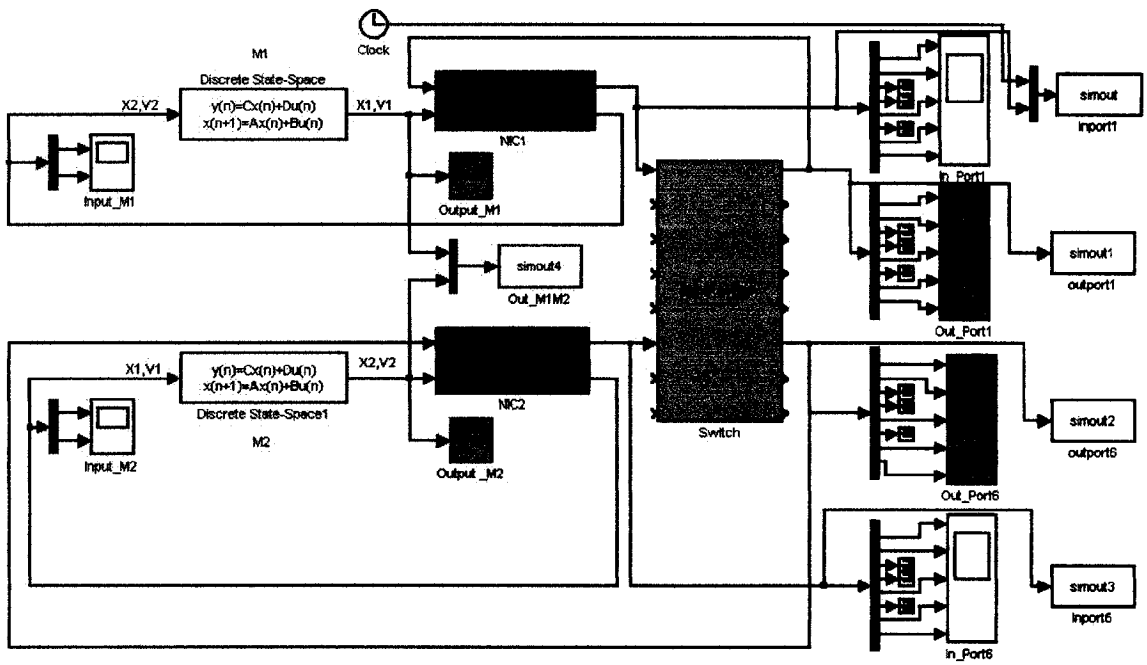


Figure 6-6: Distributed mass spring model using full duplex Ethernet protocol

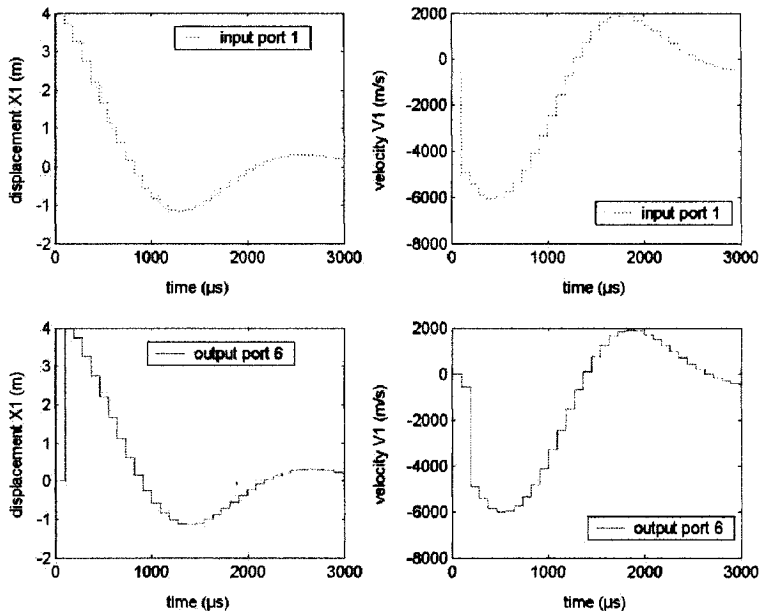


Figure 6-7: Illustration of delay caused by switch on transmitted signal from M1 to M2

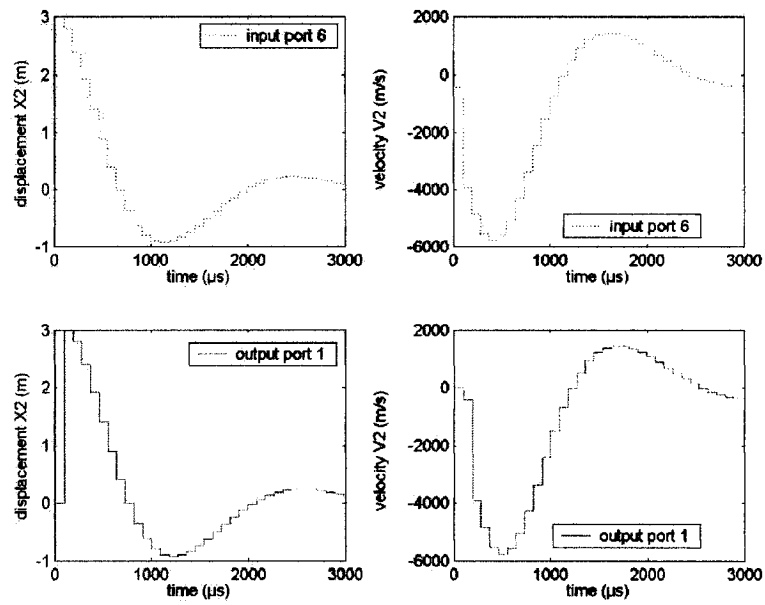


Figure 6-8: Illustration of delay caused by switch on transmitted signal from M2 to M1

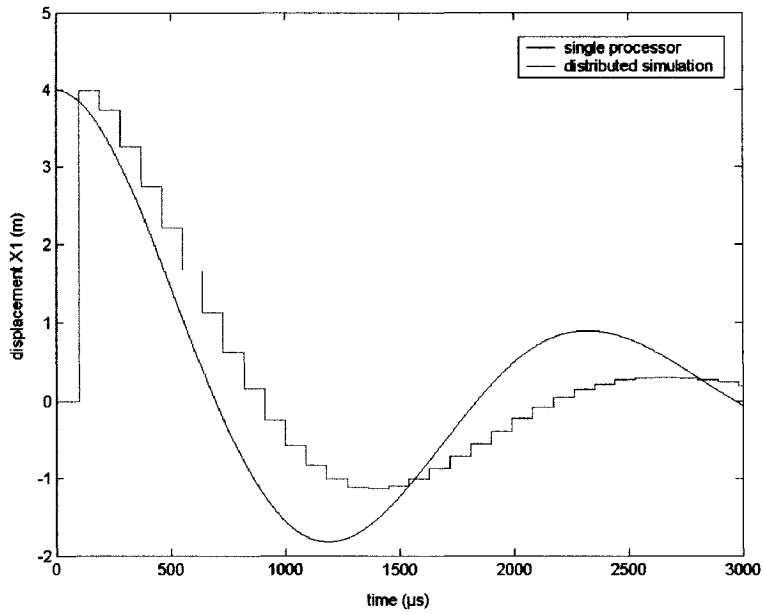


Figure 6-9: Comparison of the signals in one processor simulation model and distributed simulation *after* switch

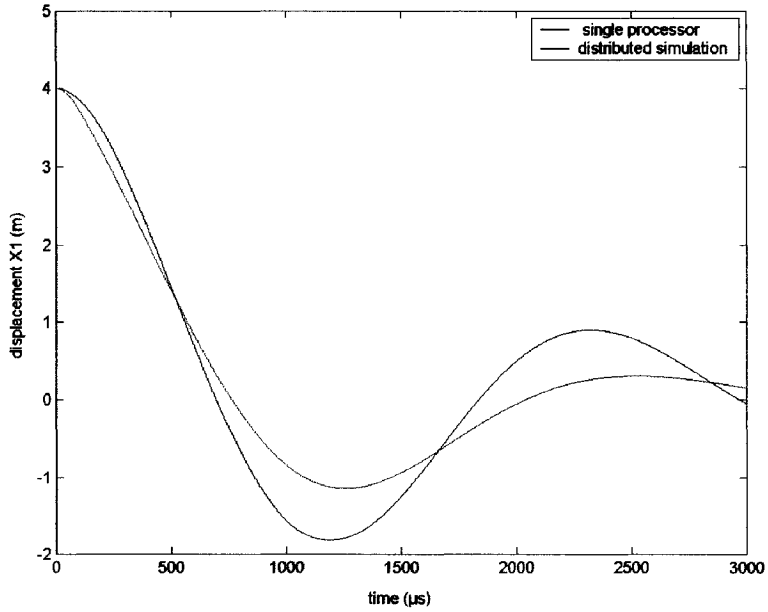


Figure 6-10: Comparison of the signals in one processor simulation model and distributed simulation

In this simulation node 1, is connected to port 1 and addresses its data to node 2, which is connected to port 6 of our switch in this model (Figure 6-6). These signals leaves NICs with the designated time delay caused by network cards and are destined to the switch.

Major latency happens when signals go through the switch. That is because of the store and forward mechanism and fabric of switches. This latency can be seen in Figures 6-7 and 6-8.

As we have already expected, in this method, no major changes in signals, in compare with the one processor model could be seen except the delays caused by the components involved in data communication (Figures 6-9,6-10).

In this particular example releasing policies are not an issue as we are in full duplex mode where no collision happens.

Nevertheless, if we increase the number of nodes, which may consequently cause simultaneous transmissions for the same destination nodes, releasing policy can be a major factor from latency point of view.

Varying the number of stations, changing the communication profiles of stations in different topologies and with different link capacities affect the real-time capability of a switched Ethernet and can be part of future works.

Based on the reasonable results maintain in this thesis and the prior discussion, we can certify functionality of all the components and models of this approach. Consequently it can be used for further researches in this field.

CHAPTER 7: CONCLUSIONS AND FUTURE WORKS

7.1 Conclusions

In this thesis, an innovative modular modelling and simulation approach is developed to predict the behaviour of distributed mechanical systems based on Ethernet (IEEE 802.3 protocol) networks. The main results and contributions of this thesis are summarized as follows:

1. Modelling and Simulation of Ethernet Networks

The steps leading to the development of the models were presented in chapter 3 of this document, based on a detailed description of the system and fundamental knowledge of networking given in chapter 2. Development involved the detailed modelling of the main components of a network system. These include Network Interface Card (NIC), bus, hub and switch components. Each component is expressed within a FSM and/or algorithm and is written in C++ language, which is encapsulated in S-function of Matlab® Simulink®. All developed components are tested and described in detail with procedures individually to confirm their designated functionality.

NIC received the most attention and credit in this approach as it is responsible to define the contention protocol and frame structure. NIC developed in both half and full duplex mode and also compatible with both coaxial (10Base5) and twisted pair (10BaseT and 100BaseT) medias.

It was also necessary to have the Bus as an active entity of a Thicknet Ethernet. In this research, 10Base5 media (coaxial cable) characteristics are the basis of modelling the propagation delay in medium.

Hubs or multi port repeaters developed to employ as a heart (concentration point) of an active star topology and propagate signal through the networks. In this approach, we assumed a semi smart hub, which was not only a data distributor but also worked as a collision detector and Jam generator.

Switched Ethernet with its all merits like filter network traffic and real-time applications was the inspiration of developing a full duplex switch with store and forward mechanism and crossbar architecture.

2. Parameter Identification

Desired accuracy of this system is based on achieving the main goal that is predict the overall time delay of transmitted packets and estimate the real-time performance of Ethernet / mechanical systems in different topologies for both normal and abnormal operating conditions. Hence, we kept the essential parameters (general and network simulation parameters listed in Table 4-1a,b and Table 4-2), which were concerned to this objective and ignore the others to simplify the model.

Definition of network performance parameters and latency quantification in this approach was discussed in sections 4.1 and 4.2.

3. Testing and Verification

Interconnection of the network was tested and verified the desired functionality of system through three models in different modes and topology in section 3.6. This

work therefore perfectly fulfills the requirements of the first objective which was innovation of a new modular / scalable simulator that could be able to simulate a networked distributed control of a mechanical system based on Ethernet protocol (IEEE 802.3) and still presenting all the necessary features to pursue the second (or main) goal of this thesis.

Methods and detailed procedure of each experiment to measure the average time delay of NIC, Hub and Switch were explained in sections 4.3, 4.4 and 4.5. Results are then compared with modeled components and necessary modification applied to the corresponding parts. While changing the parameters in both simulation and network within three tests in chapter 5 gave us a vast experience of observing the behaviour of a network system, we also checked and proved the reliability of the system in different conditions. This work therefore fulfilled the requirements of the first and second objective and this research accomplished with simulation of a combined Ethernet / mechanical system as an example of this simulator's application.

7.2 Future Works

These models are now part of the Simulink library and ready to be used as a tool for evaluating future design changes and are interesting platform from which to conduct further research.

Using the vast capability of Simulink and particularly its S-functions with the inserted flexibility in the developed codes provides the required computational power to integrate and run complex mechanical models in real-time in future.

Study of algorithms to define the configuration of switched Ethernet for real-time control applications, evaluation of different topologies and implementation of embedded switched Ethernet and data servers systems in the device level to improve the efficiency, ease of use of accessing collected measurement data by reducing infrastructures are the further steps to follow. This also causes reducing in wiring cost and optimizes using the physical volume of system.

For the future work we can also automatically divide a model into multiple subsystems, use the external mode and real-time workshop for hard real-time distributed processing and run them simultaneously on different processors which provides the computational power required running complex models in real-time simulations.

More experiences by varying the number of stations, changing the communication profiles (e.g. period of feeding data to system) of stations in different topologies and with different link capacities affect the real-time capability of a switched Ethernet and can be part of future works.

Minor modifications in NIC, would be results in having access to a Gigabit Ethernet simulator.

Further works on Switch architecture and also other functionalities of this component like working in half duplex mode, learning ability and spam algorithm can extend the application of this simulator in the other field of researches in networking.

REFERENCES

[1] Wang, J. and S. Keshav, "Efficient and Accurate Ethernet Simulation" Cornell Network Research Group (C/NRG), Department of Computer Science, Cornell University, Ithaca, NY 14853-7501, 24th Conference on Local Computer Networks, Oct.17-20, 1999,Lowell, Massachusetts.

[2] IEEE Std 802.3, "Part 3:Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications", 2000 Edition.

[3] Charles, E. Spurgeon, "Ethernet : The Definitive Guide " O'Reilly & Associates, Inc, 1st Edition, 2000.

[4] Realtek 3.3V Single Chip Fast Ethernet Controller With Power Management RTL8139C (L)+, Realtek Semiconductor Corp, Product specification, 2001.

[5] Philips Semiconductors, Low-power coaxial Ethernet transceiver, NE83Q92, Product specification, 1995.

[6] Sadiku, N.O. and M. Ilyas, "Simulation of Local Area Networks", CRC Press Inc, 1995.

[7] Amato, V. "Cisco Networking Academy Program: Second-Year Companion Guide", 1999 Cisco Systems, Inc.

[8] Vonnahme, E., Ruping S. and U. Ruckert, "Measurement in Switched Ethernet Networks Used for Automation Systems", Heinz Nixdorf Institute, System and Circuit Technology, University of Paderborn, WFCS Sept. 2000, Porto, Portugal.

[9] Potter, D. "Using Ethernet for Industrial I/O and Data Acquisition", National Instruments Corporation, 0-7803-5276-9/99 @ 1999 IEEE.

[10] Rohel, B., "Distributed Virtual Reality, *An Overview*", first draft: June 1995.

[11] Using Simulink®,"SIMULINK®, Dynamic System Simulation for MATLAB®", Version 3, 1999.

[12] Writing S-Functions, "SIMULINK®, Model-Based and System-Based Design", Version 4, 2001.

[13] Shay, W.A., "Understanding Data Communications and Networks", 2nd Edition, Cole Publishing Company, 1999.

[14] Lian,F.L., Moyne, J.R., and M.T. Dawn, "Performance Evaluation of Control Networks : Ethernet, ControlNet, and DeviceNet ", University of Michigan, 2001.

[15] Krommenacker, N., Rondeau, E. and Thierry D., "Study of Algorithms to define The Cabling Plan of Switched Ethernet for Real-Time Applications" centre de Recherché en Automatique de Nancy: CRAN-CNRS ESA 7039,Universite Henri Poincare-Nancy, France, 0-7803-7241-7/01 @ 2001 IEEE.

[16] Felser M., "Ethernet TCP/IP in Automation, *A Short Introduction to Real-Time Requirements*," Bern Institute of Engineering and Architecture, Bern University of Applied Sciences, Switzerland, 0-7803-7241-7/01 @ 2001 IEEE.

[17] Jasperneite, J. and N. Peter, "Switched Ethernet for factory Communication", Germany, 0-7803-7241-7/01 @ 2001 IEEE.

[18] Martinsson, A. "Scheduling of Real-Time Traffic in a Switched Ethernet Network", Master Thesis, Department of Automation Control, Lund Institute of Technology, March 2002.

[19] Wheelis, J.D. "Process control communications: Token Bus, CSMA/CD, or Token Ring" ISA Transactions, vol. 32, no. 2, pp. 193-198, July 1993.

[20] Koubias, S.A. and G.D. Papadopoulos," Modern fieldbus communication architectures for real-time industrial applications," Computers in Industry, vol. 26, no. 3, pp. 243-252, Aug. 1995.

[21] Tanenbaum, A.S., "Computer Networks" Upper Saddle River, NJ, Prentice-Hall Inc., 3rd Edition, 1996.

[22] Ramakrishnan, K.K. and H. Yang, "The Ethernet capture effect: Analysis and solution," in 19th Conference on Local Computer Networks, Minneapolis, MN, pp. 228-240,Oct. 1994.

[23] Khanna, V.K. and S. Singh, "An improved `piggyback Ethernet' protocol and its analysis," Computer Networks and ISDN Systems, vol. 26, no. 11, pp. 1437-1446,Aug.1994.

[24] Eidson, J. and W. Cole, "Ethernet rules closed-loop system," *InTech*, pp. 39-42,June 1998.

[25] Moyne, J.R. , N. Najafi, D. Judd and A. Stock," Analysis of Sensor/Actuator Bus Interoperability Standard Alternatives for Semiconductor Manufacturing," in Sensors Expo Conference Proceedings, Sep. 1994.

[26] Hazzard, B., " Binary Logarithmic Arbitration Method", Convergent Technology Center, Department of Electrical and Computer Engineering, Worcester Polytechnic Institute, http://www.ece.wpi.edu/courses/ee535/hwk98/hwk3cd98/brianh/brianh.html.

[27] Boggs, D. R., Mogul, J. C. and C. A. Kent, "Measured capacity of an Ethernet: myths and reality," Proceedings of the SIGCOMM'88 Symposium on Communications Architectures and Protocols.

[28] Gonsalves, T. A., " Measured performance of the Ethernet, Advances in Local Area Networks", Edited by K. Kummerle, F. A. Tobagi, and J. O. Limb, pp. 383-410, IEEE Press: New York, 1987.

[29] Hughes, H.D. and L. Li, " Simulation model of an Ethernet, Computer Performance", Vol. 3, No. 4, pp. 210-217, December 1982.

[30] Marino, P. and A. Del Rio, " An accurate and fast CSMA/CD simulator, Micro processing and Microprogramming ", Vol. 39, No. 2-5, pp. 187-190, Dec. 1993.

[31] O'Reilly, P. J. P. and J. L. Hammond Jr., " An efficient simulation technique for performance studies of CSMA/CD local networks", IEEE Journal on Selected Areas in Communications, Vol. SAC-2, No. 1, pp.238-249, Jan. 1984.

[32] Prasad, K. and R. Patel, " Performance analysis of Ethernet based on an event driven simulation algorithm", Proceedings Conference on Local Computer Networks, 1988.

[33] Prasad, K. and A. Singhal, " Simulation of Ethernet performance based on single server and single queue model ", Proceedings of the 12th Conference on Local Computer Networks, Oct. 1987.

[34] Smith, W.R. and R. Y. Kain, " Ethernet performance under actual and simulated loads", Proceedings of 16th Conference on Local Computer Networks, pp. 569-581, 1991.

[35] Takagi, H. and L. Kleinrock, "Throughput analysis for persistent CSMA System", IEEE Trans. on Communications, Vol. COM-33, No. 7, pp. 627-638, July 1985.

[36] Tsui, L.Y. and O. M. Ulgen, " On modelling local area networks", 1988 Winter Simulation Conference Proceedings, pp. 842-849.

[37] Lu, J.," Design of Ethernet based Real-Time Distributed Systems ", MASc. Thesis, in progress, 2003.

[38] http://www.thelinuxreview.com/howto/intro_to_networking/c4642.htm

[39] Knickerbocker, J., "Identifying Cable Plant Faults, Quickly and Remotely Using VCT", May 2003.

[40] B&B electronics, Manufacturing Company, Technical Article #11, "Cable Selection for RS-422 and RS-485 Systems", Jan 1999.

[41] General Cable Australia Pty Ltd.

http://www.faqs.org/faqs/LANs/cabling-faq/section-21.html.

[42] Industrial networking and open control, Vol 8, Issue 6.

http://www.industrialnetworking.co.uk/mag/v8-6/f_latency.html

APPENDIX A: STATE DIAGRAM OF DISTRIBUTED CSMA/CD SIMULATION [1]

In this approach medium is a passive entity:

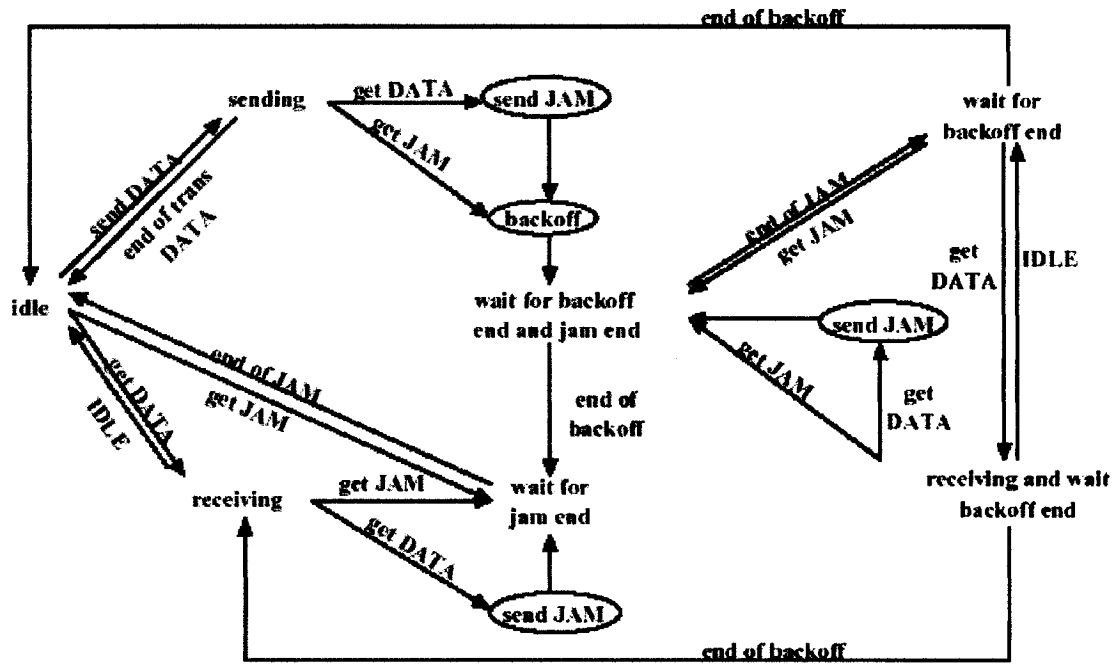


Figure A-1: State diagram of distributed CSMA/CD simulation [1]

APPENDIX B: TRANSCEIVERS

Figure B-1, illustrates a connection diagram for a low-power coaxial Ethernet transceiver:

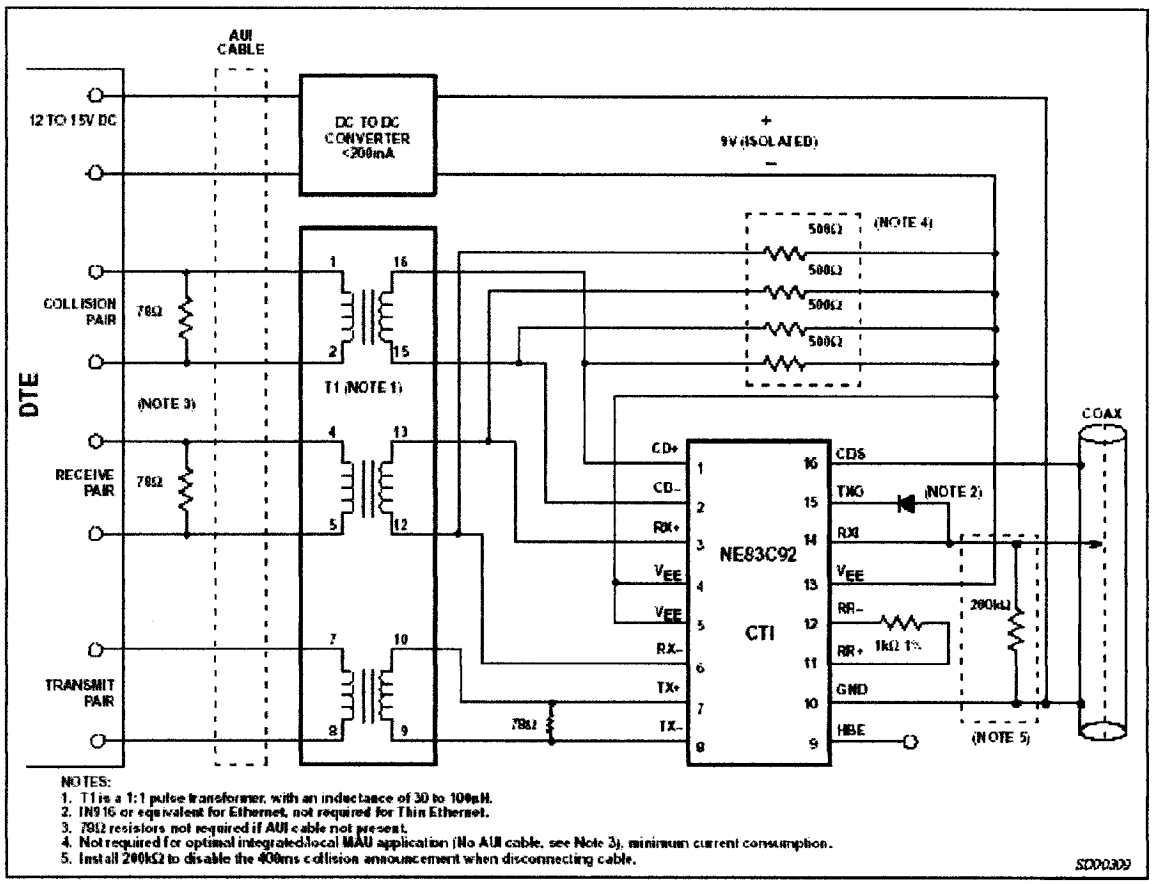


Figure B-1: Connection diagram for standard 8392 applications [5]

For the electrical and/or timing characteristics, and functional descriptions please refer to Reference 5.

APPENDIX C: NYQUIST AND SHANNON RESULTS

Nyquist's Theorem [13]: Nyquist showed that if F is the maximum frequency that medium could transmit; the receiver can completely reconstruct a signal by sampling it 2f times per second. Note that he assumed absolutely a noiseless channel.

Now suppose that, the transmitter changed the signals at intervals of (1/2f) or in other words the baud rate is 2f. We can have the results of the Nyquist theorem, which states:

$$\text{Bit rate} = \text{baud rate} * n = 2f * n$$

Where, n is the number of bits in string. If B is the number of different components, then: $B = 2 \wedge n$ or $n = \log_2(B)$, so we can write:

$$\text{Bit rate} = 2f * \log_2(B)$$

Shannon's Theorem [13]: Shannon considered noisy channel and related the maximum data rate not only to the frequency but also to the signal-to-noise ratio:

$$\text{Bit rate} = \text{bandwidth} * \log_2(1 + S/N)$$

Where: S/N means, signal-t-noise ratio (dB or bels).

APPENDIX D: SIMULATION LOOP IN S-FUNCTIONS

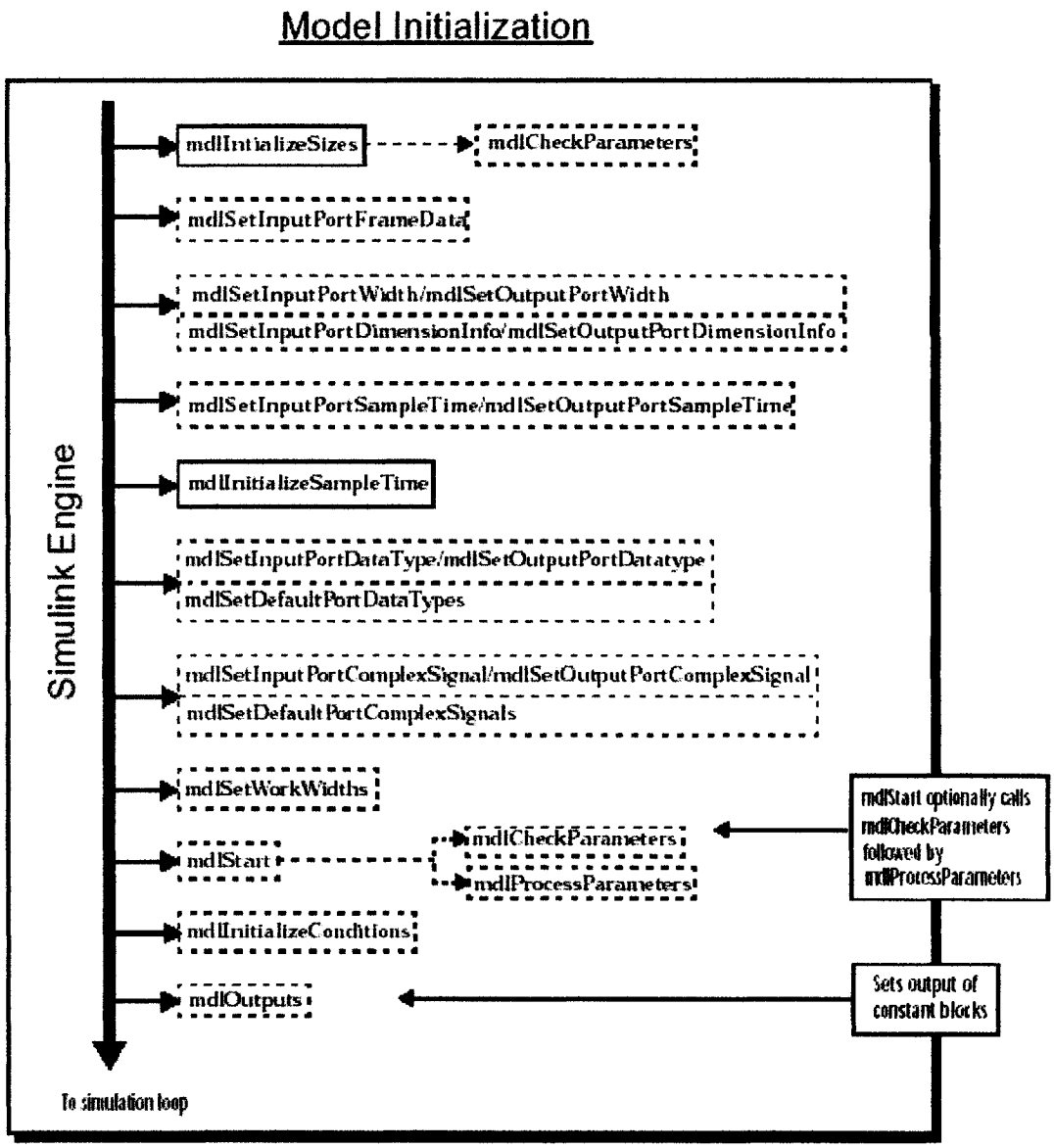


Figure D-1: Model initialization in simulation[12]

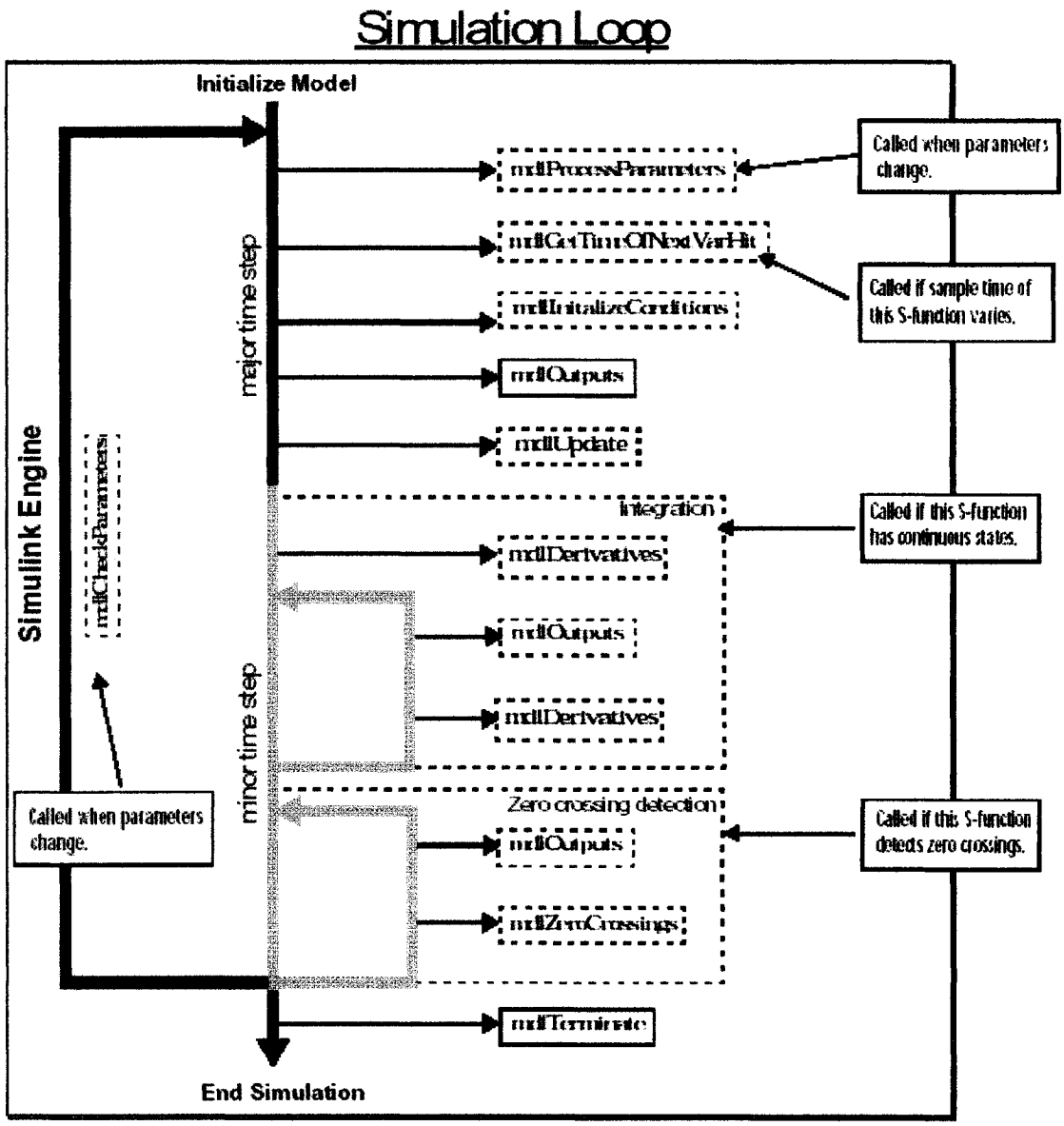


Figure D-2: Simulation loop in simulink [12]

APPENDIX E: PORT'S DELAYS IN HUB

Table E-1: Start of propagation delays (Repeater unit delay of 8 BT plus) [2]

MAU type	Input (BT)	Output (BT)
10BASE5	6.5	3.5
10BASE2	6.5	3.5
FOIRL	3.5	3.5
10BASE-T	8	5
10BASE-FP	3	4
10BASE-FB	2	2
10BASE-FL	5	5

Table E-2: Start of collision jam delays (Repeater unit delay of 6.5BT plus) [2]

MAU type	Input (BT)	Output (BT)
10BASE5	*9	3.5
10BASE2	*9	3.5
FOIRL	3.5	3.5
10BASE-T	9	5
10BASE-FP	11.5	1
10BASE-FB	3.5	2
10BASE-FL	3.5	5

*This does not include collision rise time on the coaxial media. For the worst-case round trip delay calculation, collision rise time plus MAU propagation delay.

APPENDIX F: COMPATIBILITY INTERFACES

Five important compatibility interfaces are defined. [2]

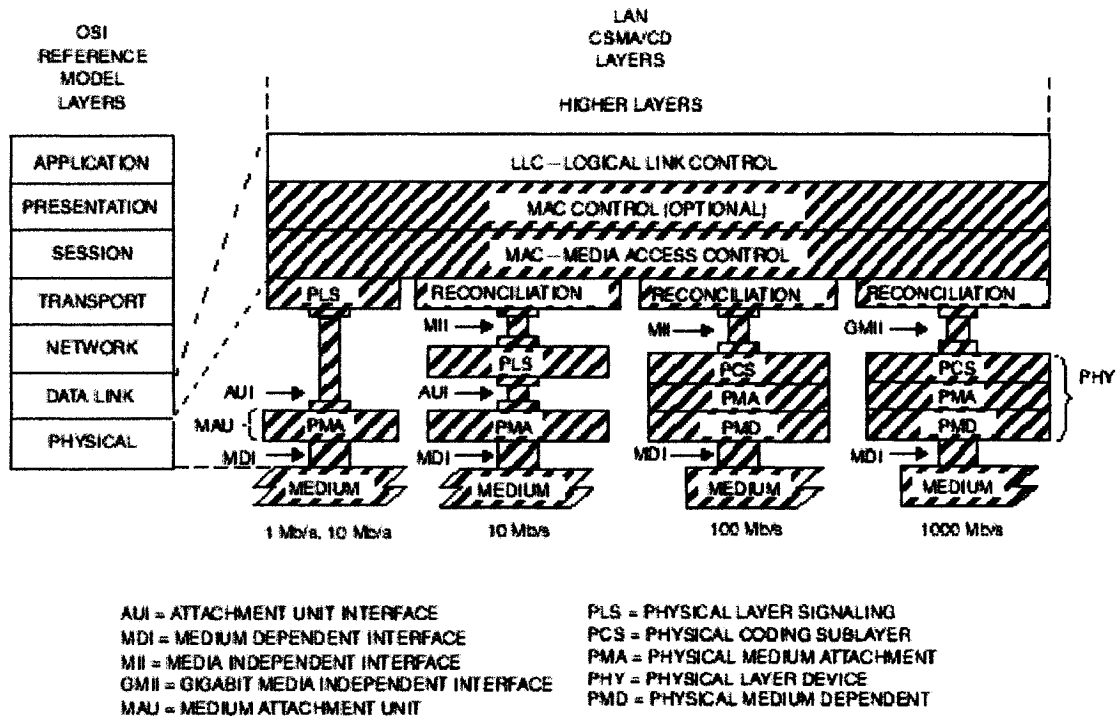


Figure F-1: LAN standard relationship to the OSI reference model [2]

a) *Medium Dependent Interfaces (MDI):* In order to communicate in a compatible manner, all stations should follow the exact specification of physical media signals, and to the procedures that define correct behavior of a station.

b) *Attachment Unit Interface (AUI):* Most DTEs are located a little far from their connection to the physical cable. A circuitry exists in the Medium Attachment Unit (MAU) directly adjacent to the physical cable, while the majority of the hardware and all of the software is placed within the DTE. The AUI is defined as a second compatibility

interface. While conformance with this interface is not strictly necessary to ensure communication, it is highly recommended, since it allows maximum flexibility in intermixing MAUs and DTEs. The AUI may be optional or not specified for some implementations of this standard that are expected to be connected directly to the medium and so do not use a separate MAU or its interconnecting AUI cable. The PLS and PMA are then part of a single unit, and no explicit AUI implementation is required.

c) *Media Independent Interface (MII):* Some DTEs are connected to a remote PHY, and/or to different medium dependent PHYs. The MII is defined as a third compatibility interface. While conformance with implementation of this interface is not strictly necessary to ensure communication, it is highly recommended, since it allows maximum flexibility in intermixing PHYs and DTEs. The MII is optional.

d) *Gigabit Media Independent Interface (GMII):* The GMII is designed to connect a gigabit-capable MAC or repeater unit to a gigabit PHY. While conformance with implementation of this interface is not strictly necessary to ensure communication, it is highly recommended, since it allows maximum flexibility in intermixing PHYs and DTEs at gigabit speeds. The GMII is intended for use as a chip-to-chip interface. No mechanical connector is specified for use with the GMII. The GMII is optional.

e) *Ten-bit Interface (TBI):* The TBI is provided by the 1000BASE-X PMA sub layer as a physical instantiation of the PMA service interface. The TBI is highly recommended for 1000BASE-X systems, since it provides a convenient partition between the high-frequency circuitry associated with the PMA sub layer and the logic functions associated

with the PCS and MAC sub layers. The TBI is intended for use as a chip-to-chip interface. No mechanical connector is specified for use with the TBI. The TBI is optional. We can also see the relationship between 10BaseT media and OSI model in Figure F-2:

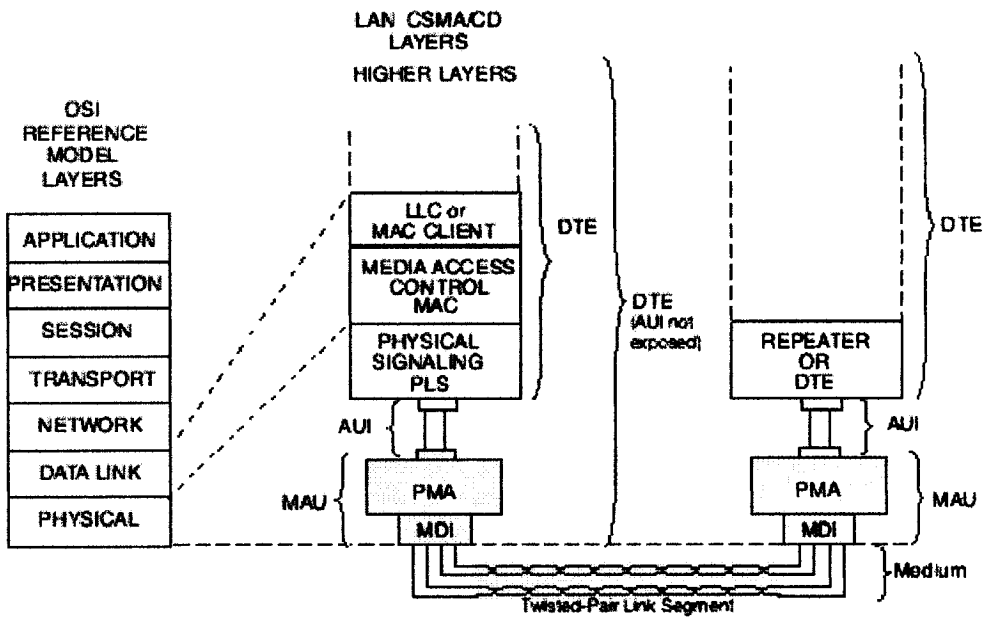


Figure F-2: 10BASE-T relationship to OSI reference model and the IEEE 802.3 CSMA/CD LAN model [2]

APPENDIX G: RECEIVE VARIENCE / SCAN TIME JITTER

Receive variance: Very often in control applications the same value must be used on two different places at the same time. For consistency reasons the value should be the same in a certain time delay. This variance Trv, depends on the requirements of the application.

There exist two variants: The system sends one message for every destination or there are multi cast messaged used. In the unicast message, the latency of the message adds together, to form the receive variance. In the multicast message, only the differences of the delay of the network equipment lead to receive variance.

Scan time jitter: For distributed control-loops the message has to be transmitted in cyclic intervals to all devices involved in the loop. That the control-loop can work correctly, the scan time has to be implemented with a limited jitter. In practice, a jitter of less than 5% of the scan time is requested. The scan time varies on the sum of the receive variance of the different massages. The effect of the jitter can be limited, if a multicast message is used to synchronize the field devices. In this case the jitter is limited to the receive variance of the multicast message. This holds as long as the scan time is not longer as the cycle time of application.

APPENDIX H: TEST RESULTS FOR CHAPTER 5

In the following, parts of results for each test of chapter 5 have been brought:

TEST 1:

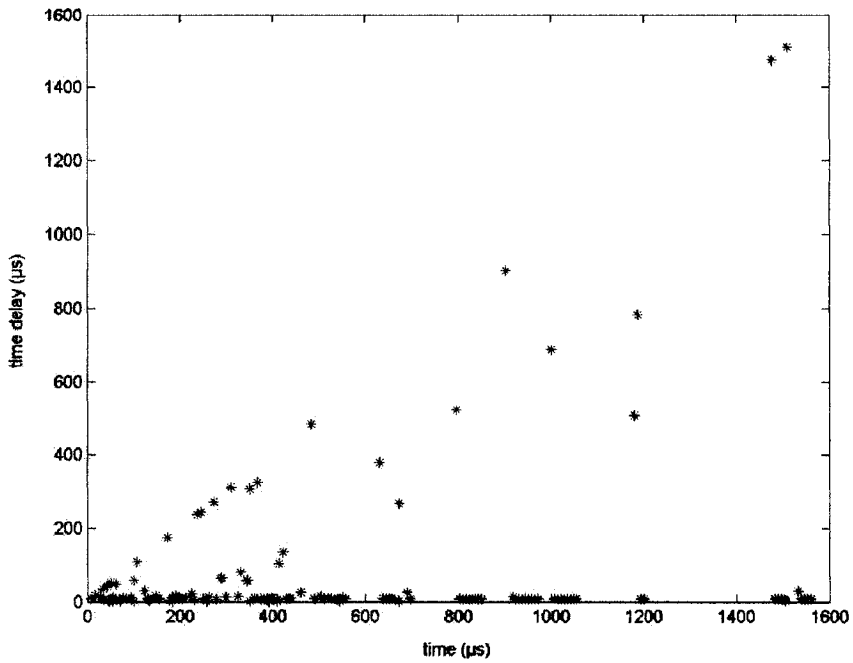


Figure H-1: Test 1-Measurement of time delay for a 10-node network

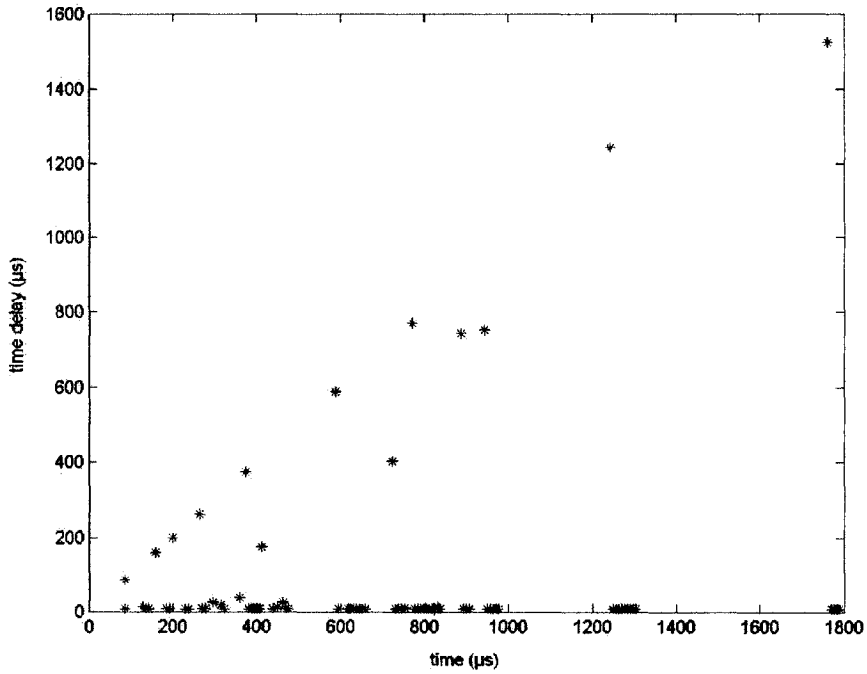


Figure H-2: Test 1-Measurement of time delay for an 8-node network

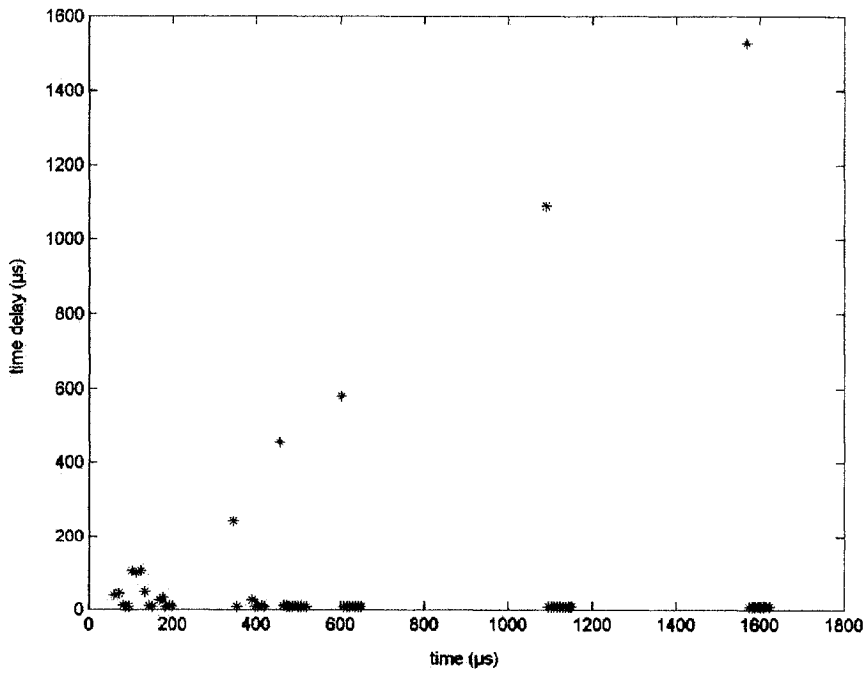


Figure H-3: Test 1-Measurement of time delay for a 6-node network

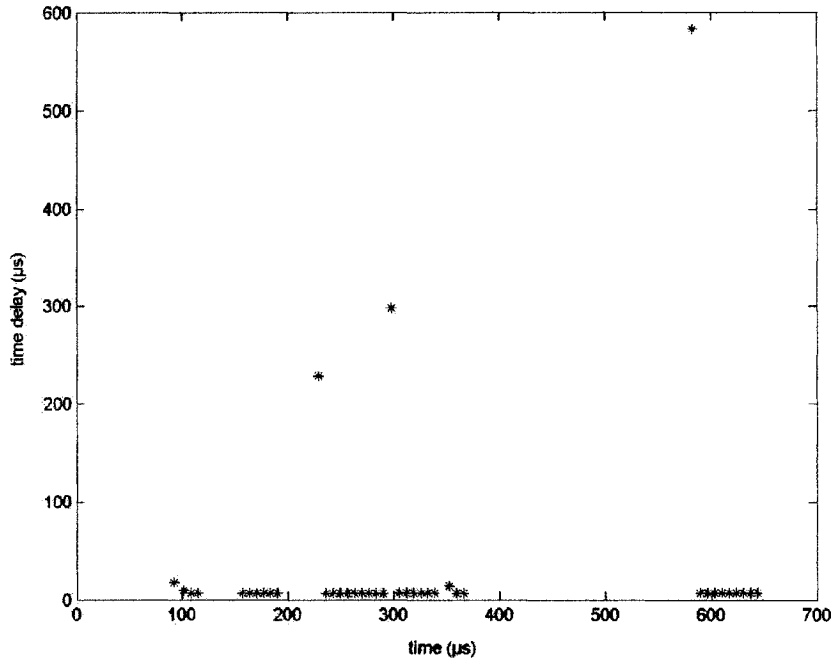


Figure H-4: Test 1-Measurement of time delay for a 6-node network

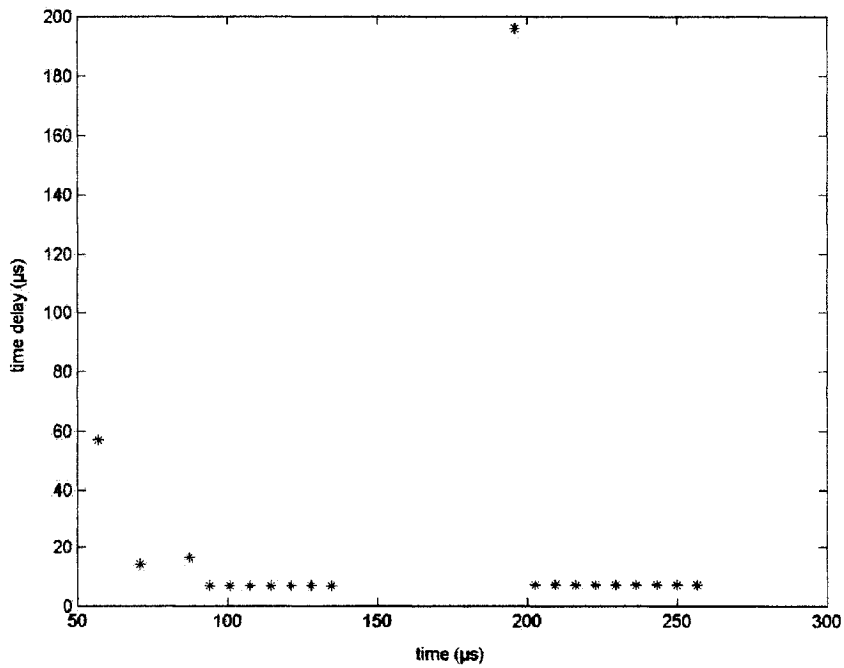


Figure H-5: Test 1-Measurement of time delay for a 2-node network

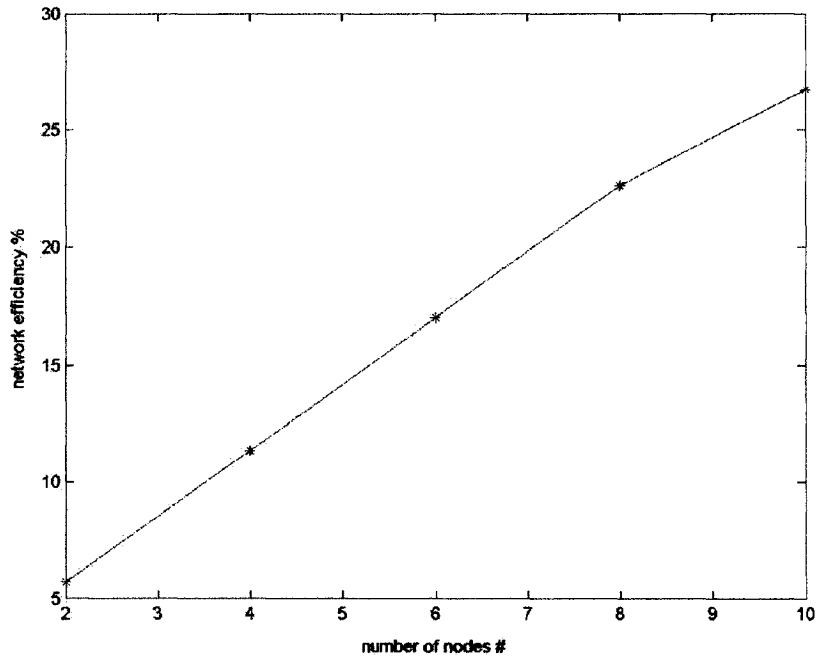


Figure H-6: Test 1 - Network efficiency versus number of nodes in the network

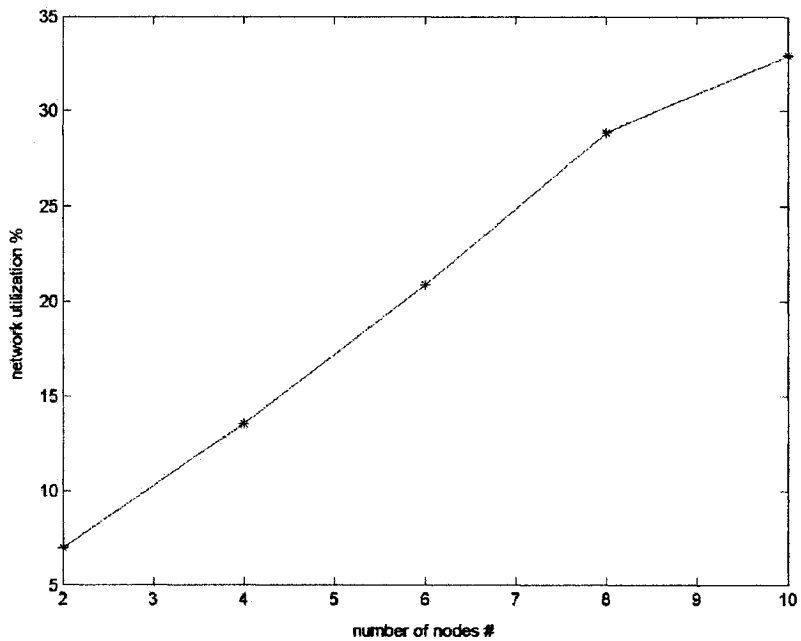


Figure H-7: Test 1 - Network utilization versus number of nodes in the network

TEST 2:

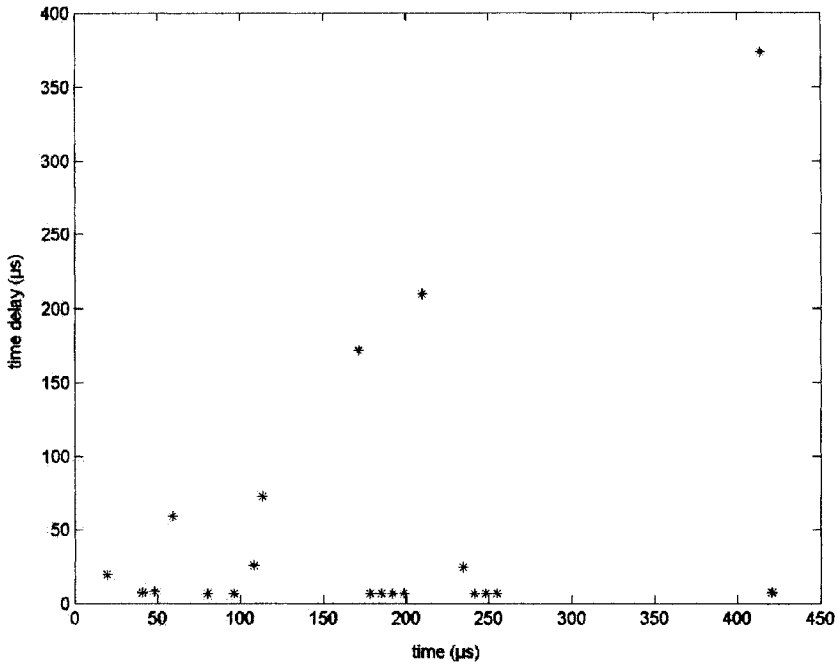


Figure H-8: Test 2-Measurement of time delay for 5 messages per node

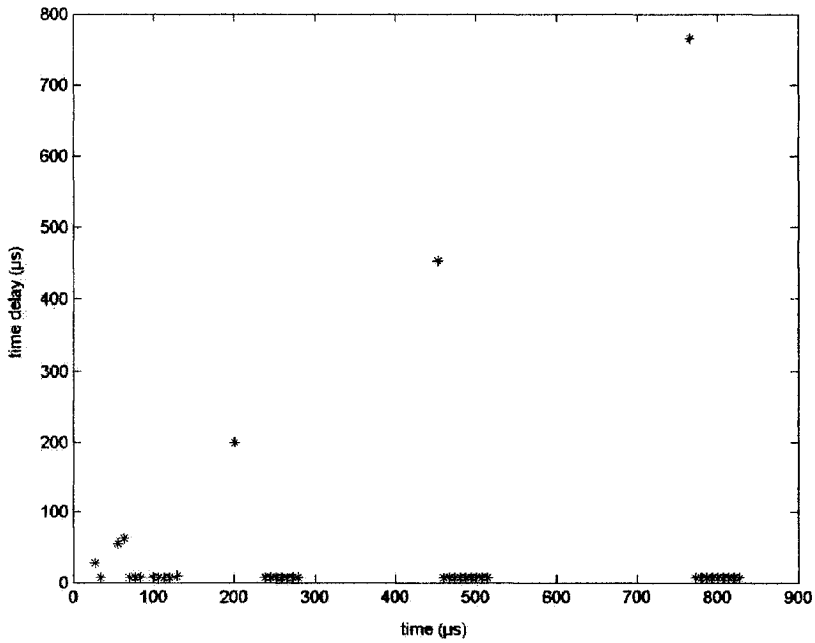


Figure H-9: Test 2-Measurement of time delay for 10 messages per node

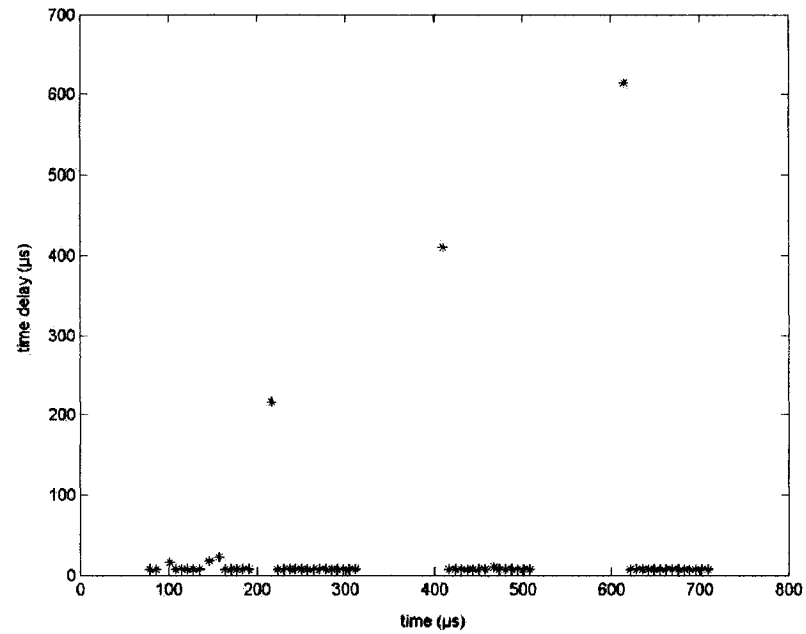


Figure H-10: Test 2-Measurement of time delay for 15 messages per node

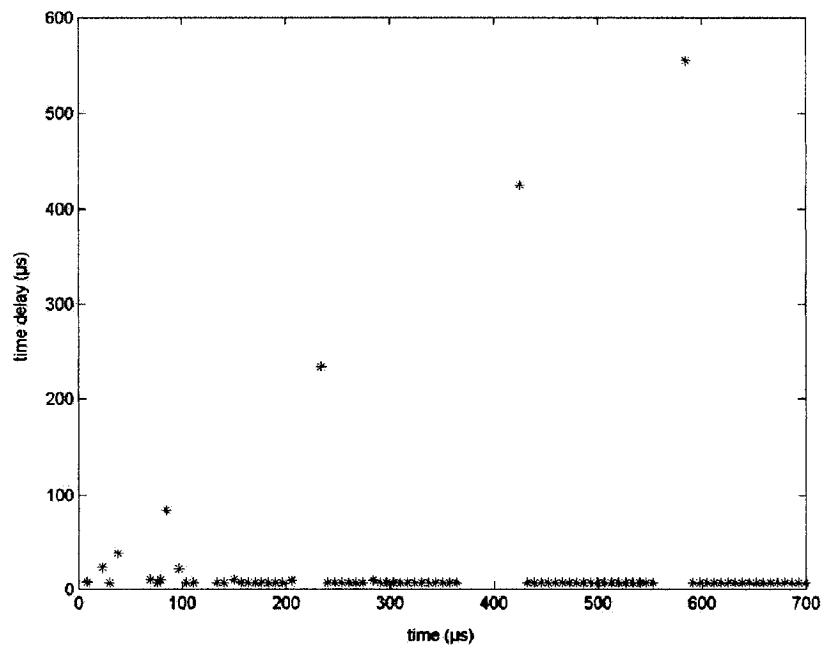


Figure H-11: Test 2-Measurement of time delay for 20 messages per node

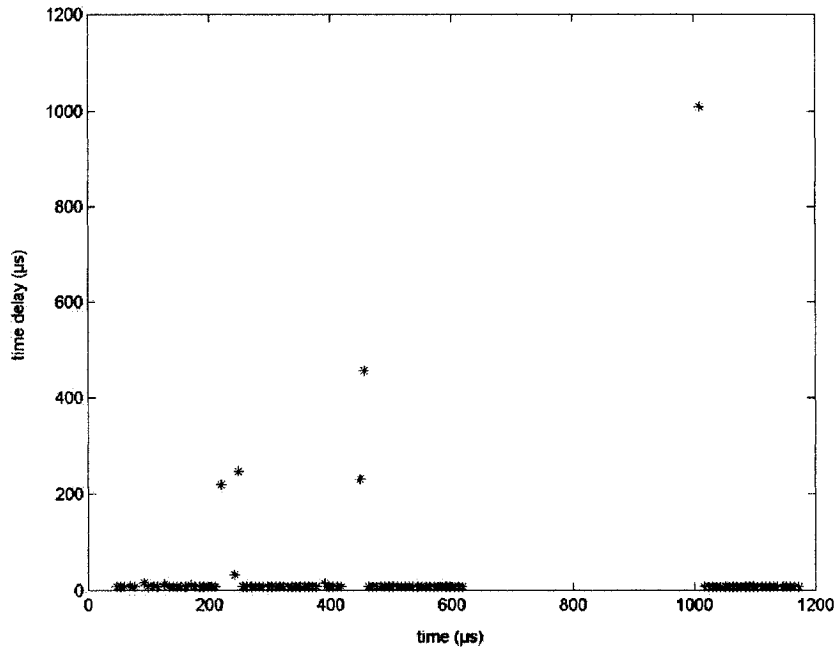


Figure H-12: Test 2-Measurement of time delay for 25 messages per node

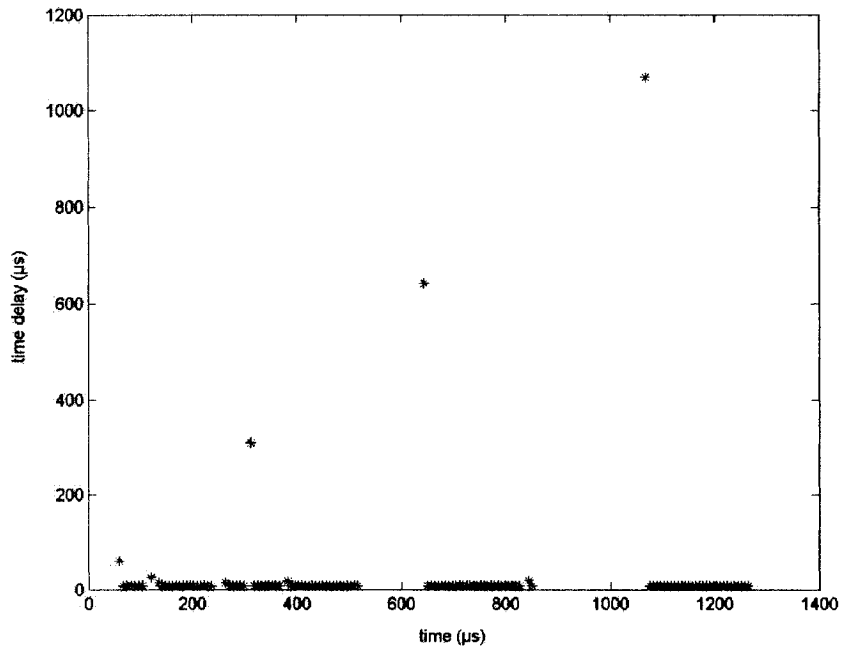


Figure H-13: Test 2-Measurement of time delay for 30 messages per node

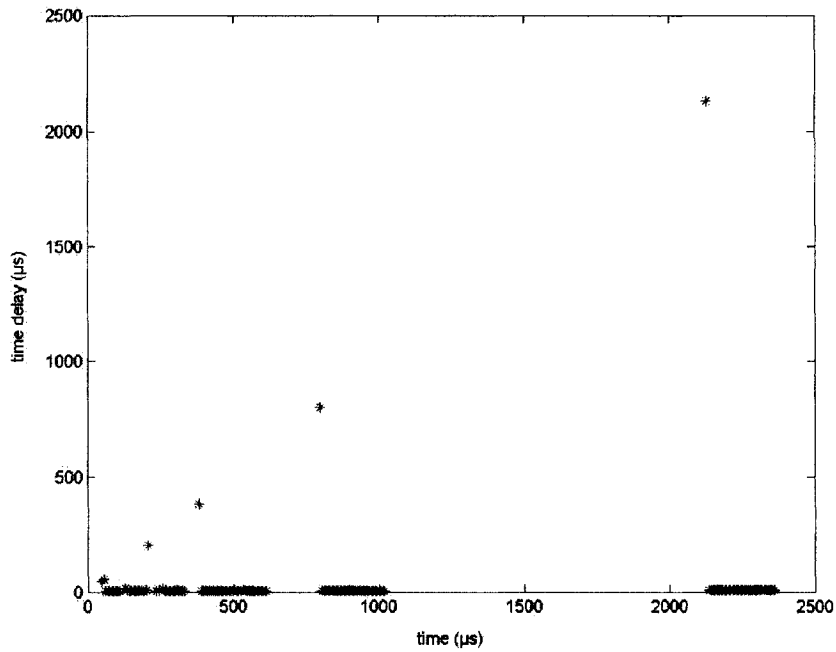


Figure H-14: Test 2-Measurement of time delay for 35 messages per node

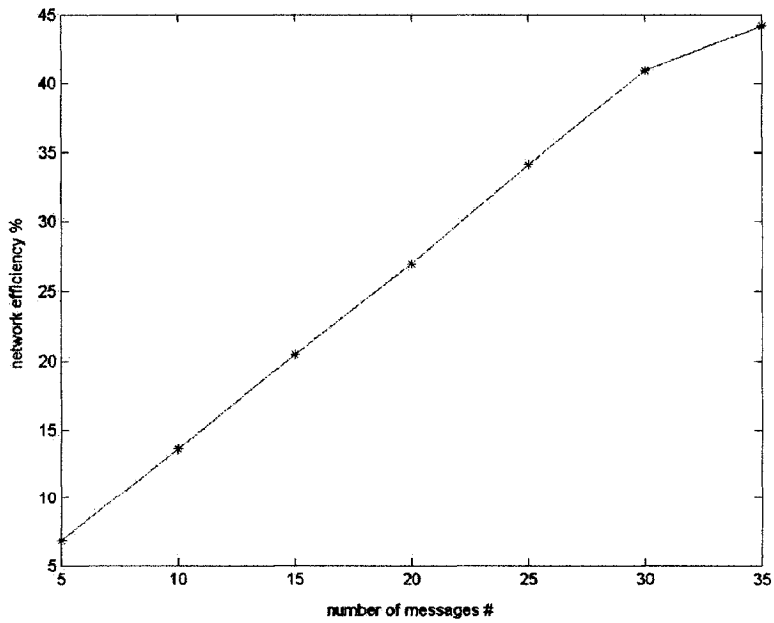


Figure H-15: Test 2 - Network efficiency versus number of messages per node to transmit

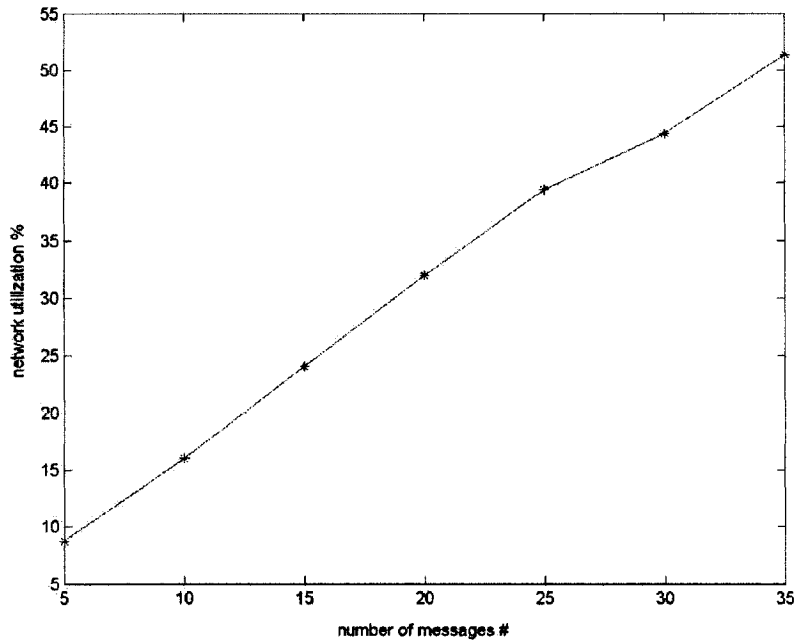


Figure H-16: Test 2 - Network utilization versus number of messages per node to transmit

TEST 3:

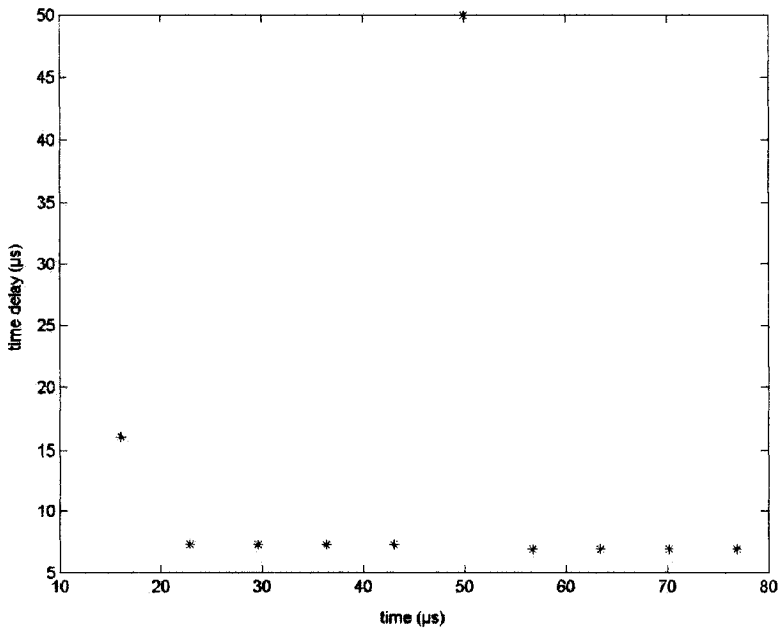


Figure H-17: Test 3-Measurement of time delay (2 node, frame size 72 bytes)

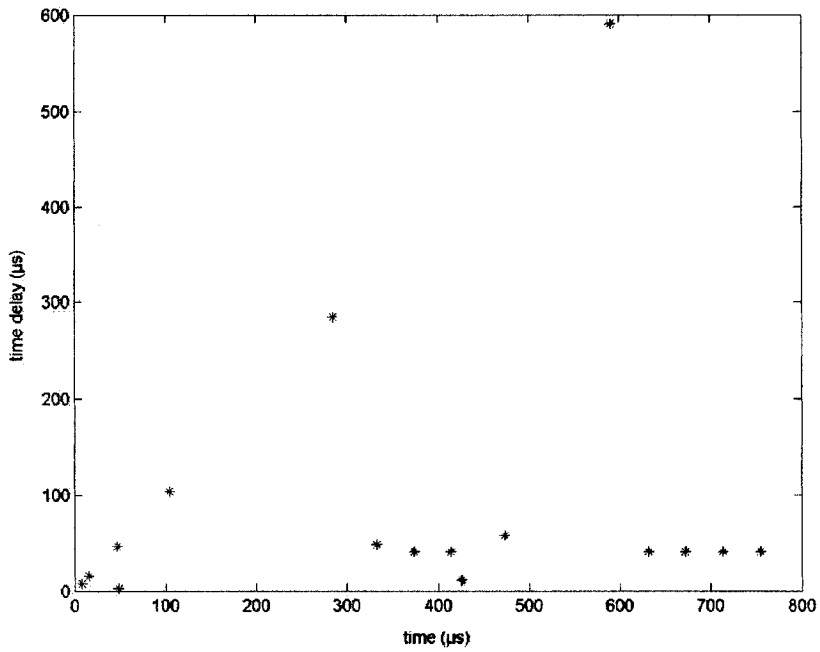


Figure H-18: Test 3-Measurement of time delay (2 node, frame size 500 bytes)

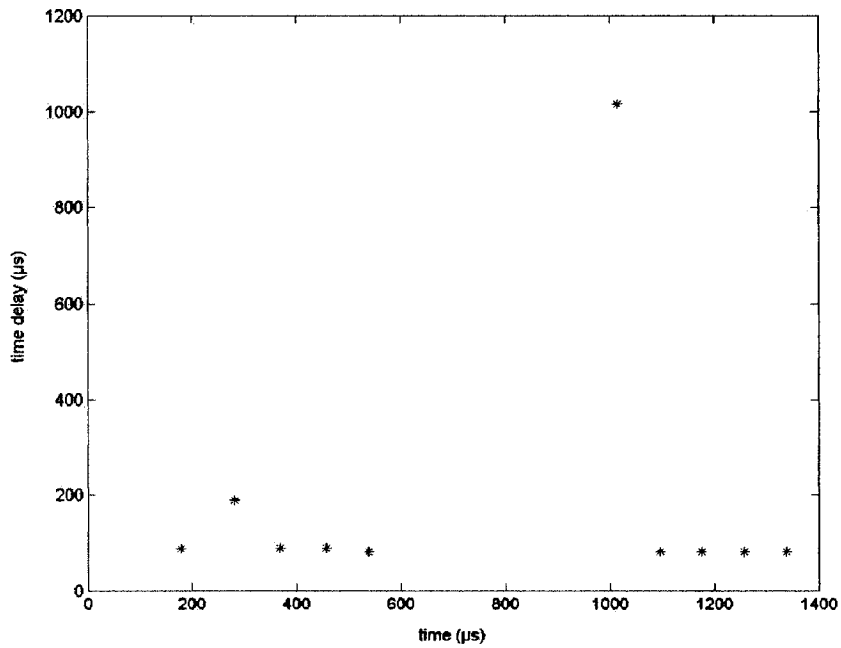


Figure H-19: Test 3-Measurement of time delay (2 node, frame size 1000 bytes)

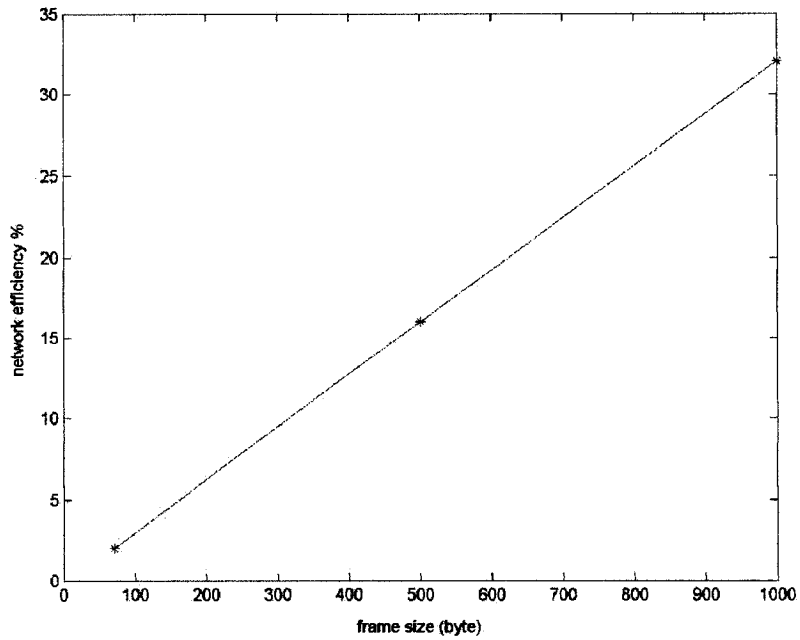


Figure H-20: Test 3 - Network efficiency versus frame size for two-node network

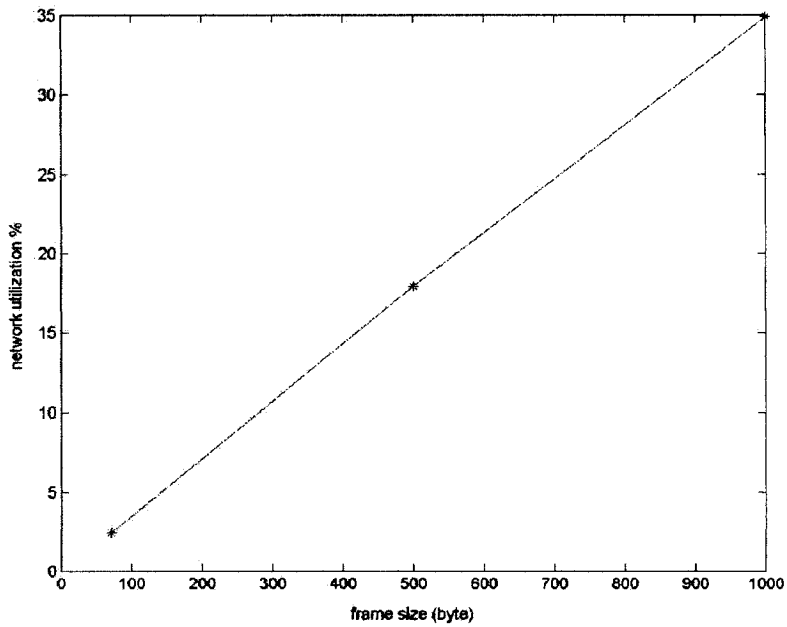


Figure H-21: Test 3 - Network utilization versus frame size for two-node network

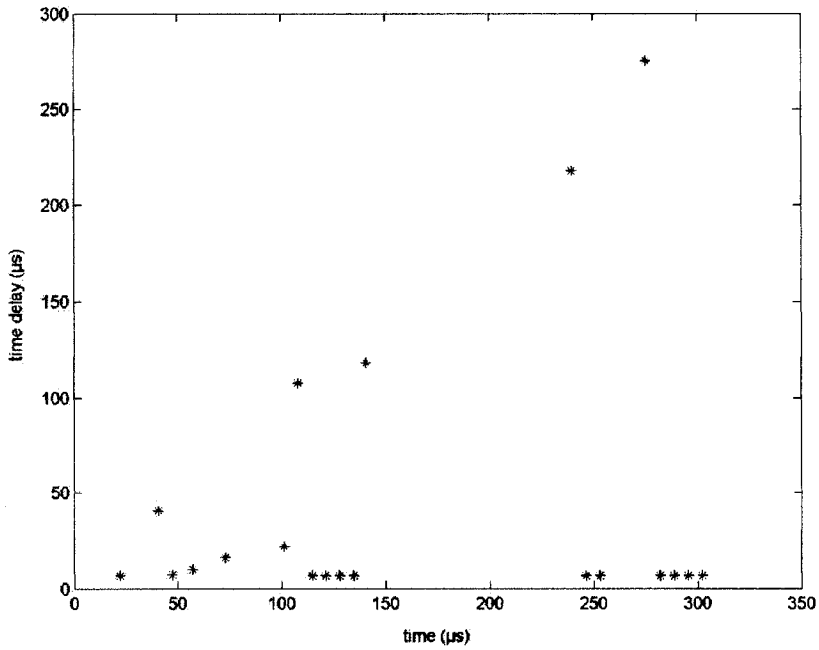


Figure H-22: Test 3-Measurement of time delay (4 node, frame size 72 bytes)

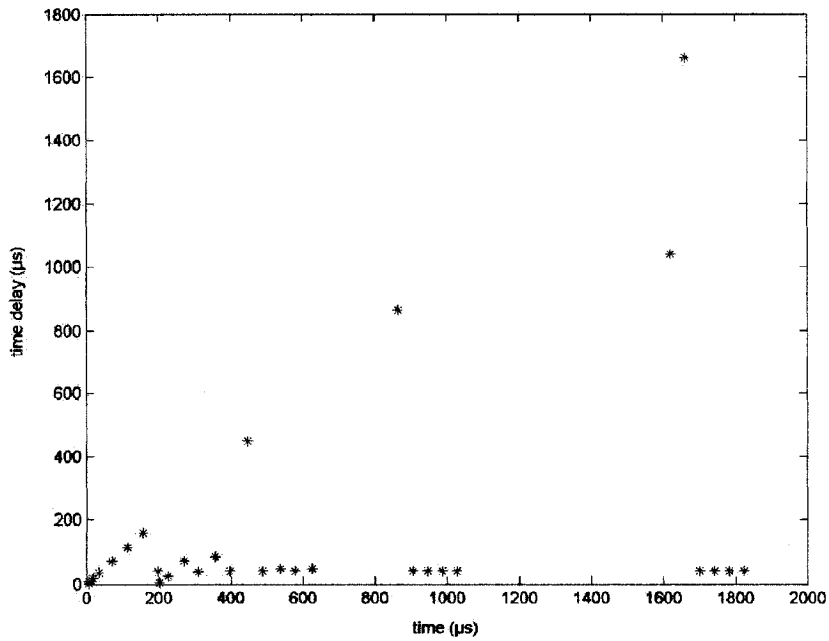


Figure H-23: Test 3-Measurement of time delay (4 node, frame size 500 bytes)

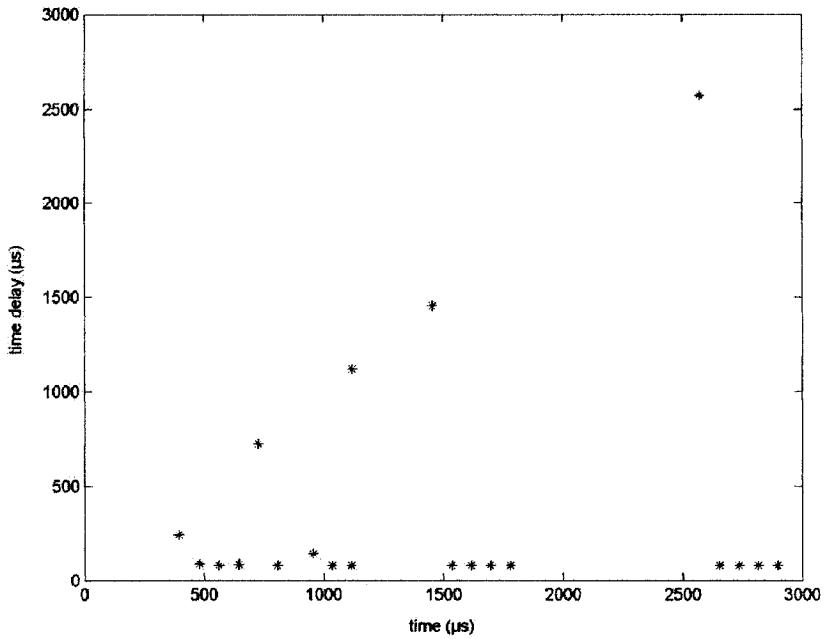


Figure H-24: Test 3-Measurement of time delay (4 node, frame size 1000 bytes)

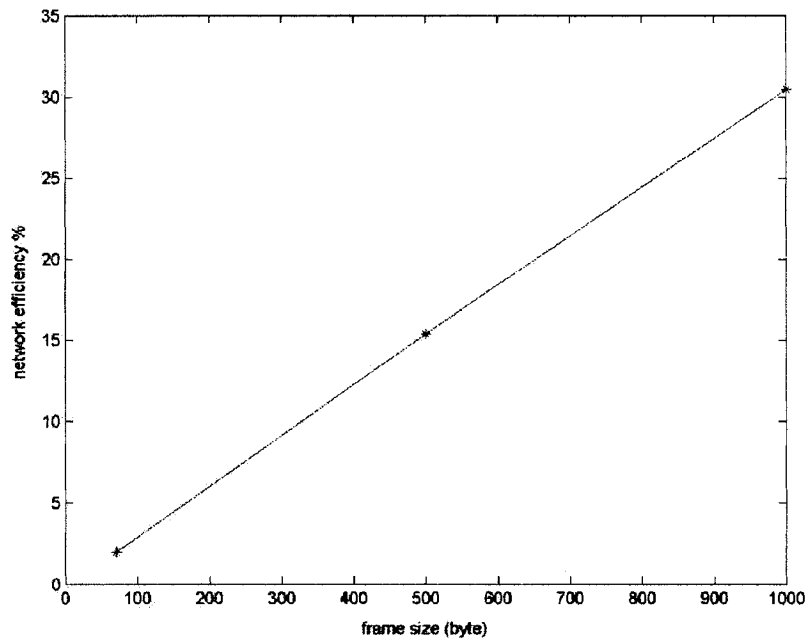


Figure H-25: Test 3 - Network efficiency versus frame size for four-node network

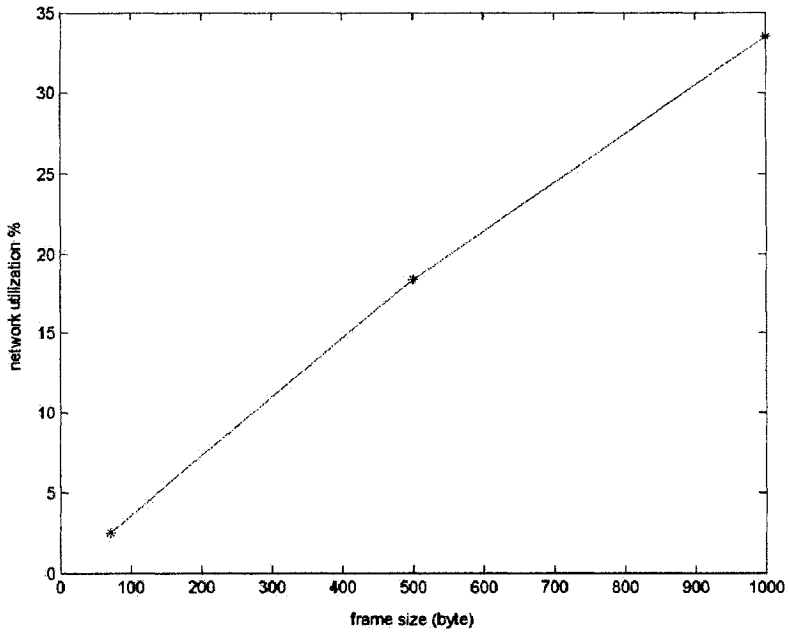


Figure H-26: Test 3 - Network utilization versus frame size for four-node network

APPENDIX I: ETHERNET AND REAL-TIME CONTROL APPLICATIONS

The research in real-time communications can be grouped into two categories: Soft real-time or best effort communication and hard real-time. The primary performance objective of the first category is to maximize the percentage of data packets successfully transmitted within a predetermined time constraint. Voice and video package communication are included in this category of application in which usually losing a certain amount of packets are tolerable and an unsuccessfully delivered packet within a certain time limit is considered lost. On the other hand, industrial networks in which several packets, such as alarm and control signals must reach their destinations before a time deadline are the classical applications of hard real-time networks and they aim to improve the throughput while ensuring all time constrained packets to meet their time deadlines. [20]

Ethernet communication systems should fulfill the requirements regarding determinism, queues, consistence and cyclic control.

I.1 Determinism

The latency for the message should never takes longer than the specific maximum delay of an application. Most of the part-time latencies can be calculated. Figure 4-1 illustrates the most important latencies, which occur during the transmission from one node to another. The most critical part is ΔT_2.

There are three possibilities to get a more deterministic behaviour of the Ethernet: Limited traffic reduces the number of collision. With a reduction to less than 10% the risk of non-deterministic and unacceptable delays is ignorable. In fact we can separate the real-time network from the IT world by routers or bridges. This is a non-efficient solution but it is used and proposed already for ten years by different manufacturers. [16]

- Giving a deterministic behaviour similar the one used by Fieldbus by putting a Master–Slave system on top of Ethernet is the easiest solution. The Master controls the bus access and possible maximal delay can be calculated based on the polling algorithm of the master. In This solution one device is the Master and polls the other as Slave. Normal IT devices, not designed for these Master-Slave rules, are not allowed to be on the same segment. This solution is most efficient, when no switches and only hubs are used. [16]
- By applying the switch to segment the bus in several collision domains we can avoid the collision. It is the most proposed solution today and in the best case we can get one domain for each device and have no collision anymore. [16] Note that this solution has the major inconvenience that switches are blocking which can be prevented by special signalling name Backpressure with half duplex connections and the MAC Control Pause Mechanism for the full duplex. Moreover at least at the moment they are still very expensive devices. [8]

The system parameters that affect real-time capabilities in switched Ethernet networks are as follows: [17]

- ✓ Number of stations

- ✓ The station communication profiles
- ✓ Type of topology and the resultant number of switch.
- ✓ Link capacity (10,100,1000,10000 Mbps)
- ✓ Packet scheduling strategy of the transit system (switches) and the stations.
- ✓ The thinking time, ΔT_1 within the stations, which comprises the processing time for communications, requests within the stations.

The interconnection between the different switches must be analyzed because a bad management of the network-cabling plan can generate bottlenecks and can slow down the network traffic. Several methods have been proposed to design architectures of Switched-Ethernet networks. The following figure is an example, which illustrates a physical network architecture constituted of a star topology where a federative (or backbone) switch interconnects the whole end switches (gathering the end nodes). When the spanning tree algorithms detect a failure in the backbone, the linear topology replaces the star topology. [15]

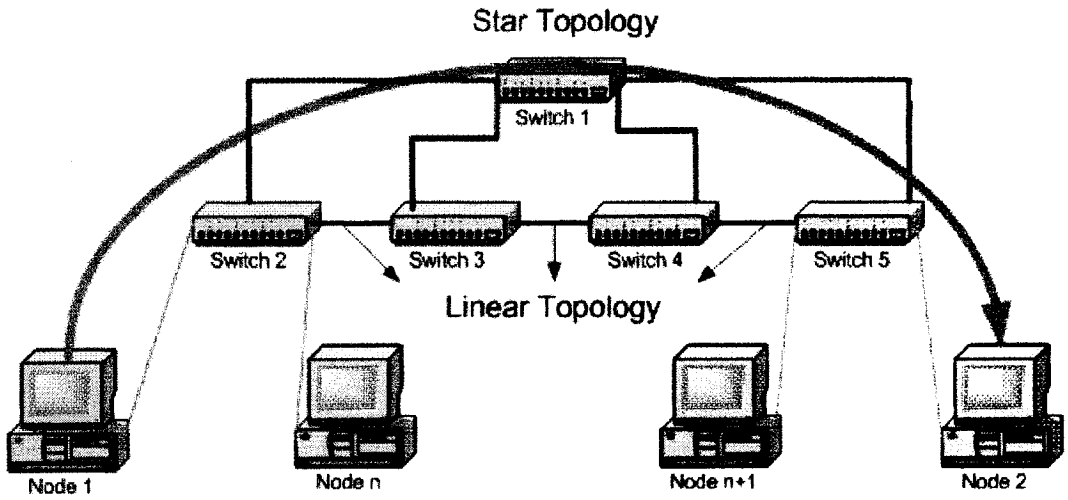


Figure I-1: Hybrid topology studied [15]

The linear topology and the redundant topology corresponds to the star architectures is one way to obtain the best end-to-end delay times. Note that in an industrial context, the parameters of the mean delays are not enough to measure the adequacy of the communication system in regard to the constraints of the real-time applications. It is necessary to evaluate others parameters such as end to end maximum delays or the end to end jitters. [15]

Meanwhile, the star topology requires the maximum cabling, therefore a bus-like topology, which assigns precisely one switch to each devices and create the same topology as in field bus technology, is suggested. In case of this bus-like topology it is imaginable, that future devices have an embedded switch functionality to reduce infrastructures and wiring costs. On the other hand based on maximum number of switches, the bus-like topology is not the best choice in regard to real-time capability. [17]

I.2 Queues

It is common to increase the bit rate speed to reduce the ΔT_2 and ΔT_3 (Figure4-1), hence more messages coming now to the receiver in the same time. If the receiver is not able to handle all these messages at the speed they get in, they are queued in the communication stack. It is also the same for the switches. If the communication capacity between the switches is lower than the sum of the capacity to the end users, the switch has to either drop or queue the messages, which adds an additional delay [16]. For the adaptation to Ethernet and TCP/IP there exits two different solutions to this problem:

Priority: Define a clear strategy for IP priority and define separate queues for real-time control information. Although UDP telegrams will be used and not TCP but there is no need to modify the communication stack. [16]

New Protocol: Invent a new protocol on top of the Ethernet based on the buffered principals or use an existing Fieldbus protocol on top of Ethernet replacing TCP/IP, therefore there is no compatibility with the existing IP Protocol anymore. Be advise that, Fieldbus do not work with the queues and it use a buffer mechanism. The new message overwrites the old value in the buffer, so it is on the base of this idea that the only last valid value is interesting and not the sequence of values. [16]

I.3 Consistence

There is a problem to implement consistent communication in a distributed system with the Client-Server model. We have to define two Client-Server relations while we want to set a value in two field-devices from a controller in a consistent way. On one communication network, these services request messages are always passed one

after the other, so there is automatically a $\Delta T_{const} > \Delta T_{lat}$ [16]. To get acceptable consistence in Ethernet based network there exist different approaches:

Speed: Just increasing the bit rate on the system reduces mainly ΔT_2 and ΔT_3 and therefore causes $\Delta T_{lat} < \Delta T_{const}$. This method creates the queues problem.

Fieldbus: Use an existing fieldbus protocol on top of Ethernet, which implements the publisher-subscriber model.

New Protocol: A new protocol on top of the Ethernet, base on Publisher-Subscriber model should be invented.

Note that, in the publisher-subscriber approach which is used by Fieldbus the consumer of a value subscribes to the publisher of an output variable of a function. The producer of the variable publish the actual value on the communication network and all subscribers can take the same value on the same time from the same message. The only difference in the delay is the difference in transmission delay, ΔT_3, due to the signal runtime on the communication cable and the different ΔT_4 in the receivers.

I.4 Cyclic Control

For distributed control-loops the messages have to be transmitted in cyclic intervals with the limited jitter. For the Ethernet, two solutions are possible:

Master Scheduler: By using the Fieldbus master-slave principle also over the Ethernet we get the cycle control like this. It has the disadvantage that the availability of the system depends on the availability of the central scheduler. The central scheduler ensures that the timing constraints on the bus system are fulfilled and sends the corresponding

synchronization messages. In a polled system it calls all values in the correct cycle interval.

Time Stamp: Synchronize the clocks in all devices. The messages are sent with time stamps. The receiver correct the different receive variant to the specified maximum latency. [16]

APPENDIX J: NIC ROUND TRIP LATENCY TEST

Measuring of the two transmitting and two receiving latencies, which still use network NIC as feedback.

Operation procedure:

- Step1: Server reads the clock, gets time T_1 and sends a packet to the NIC. Client is polling the NIC register, waiting for the packet.
- Step2: Server is polling the NIC register. Client receives the packet and sends back the packet to server.
- Step3: Server receives the packet and reads clock again, gets time T_2.

Figure J-1 and Table J-1 illustrate the definition of latencies for this test.

Obviously: $T_{dtx\_NIC} + T_{drx\_NIC} = (T_2 - T_1 - T_{dclock}) / 2.0$

Table J-1: Latencies definition

LATENCY	DEFINITION
T_{dclock}	**Time delay of clock reading**
T_{dtx_NIC}	**NIC time delay of Tx**
T_{drx_NIC}	**NIC time delay of Rx**
T_{dround}	**Round trip time delay**

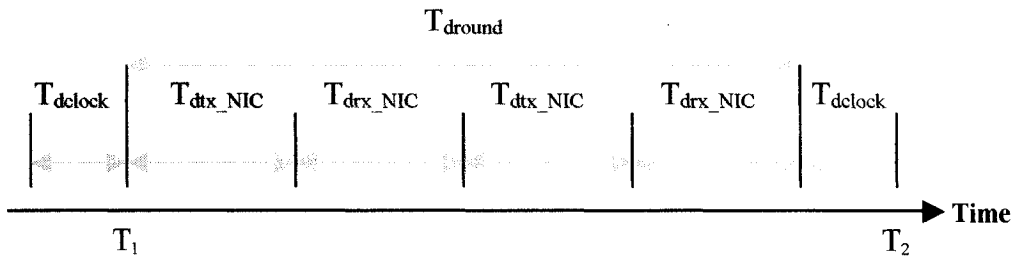


Figure J-1: Definition of latencies

As the length of the network cables is less then 3 m, the resulting time delay by cable, is negligible.

APPENDIX K: APPLICATIONS OF SIMULATION

In the following, we discuss the most interesting applications of combined Ethernet / mechanical system simulation:

K.1 Distributed Control System Design and Testing

Control systems and subsystems are mostly being formulated and integrated in dynamic equations of motion. Most real life control systems are so complex that their descriptions by a mathematical model are beyond the capabilities of an analyst and are also so complicated to be solved within a single processor in a reasonable amount of time (See Appendix I for soft and / or hard real-time applications). Hence, we can break the control system to several subsystems (nodes), which are connected together through an Ethernet protocol real-time control system as possible.

At this point the potential for using parallel computation to enhance performance is receiving the most attention, with application to both supercomputing facilities and distributed-network implementation.

Simulation of this system helps to observe the parallel computation in different topologies and / or abnormal conditions while we enhance the real-time behaviour of interconnected control system.

K.2 Congestion Control Design and Testing

A potential serious consequence of growth in networks is congestion, which needs routing strategies to deal with increasing number of packets and any failure in the network links.

While we are talking about using simulation to monitor the performance, flow control aspect comes up and hence we can see the merits of applying the simulation for the congestions control. These can be detected before packets are dropped, and the network layout can be adapted to the traffic.

K.3 Distributed Virtual Environment Design and Testing

The idea behind distributed Virtual Reality (VR) is very simple; a simulated world runs not on one computer system, but on several. The computers are connected over a network (possibly the global Internet) and people using those computers are able to interact in real time, sharing the same virtual world. Each of the computers participating in the simulation is called a "host". On each host there are a number of "entities" (things in the virtual environment) that communicate their changing state by sending "update messages". The specific entity that corresponds to a human participant's virtual body is called their "avatar".

There are lots of challenges that we'll be facing in designing a distributed VR standard. The three basic challenges are:

- Compatibility
- Limited bandwidth
- Latency (lag)

K.4 Higher Level Protocol Design and Testing

Computers attached to an Ethernet can send application data to one another using high-level protocol software, such as the TCP/IP protocol suite used on the worldwide Internet or TDMA (Time Division Multiple Access). The high-level protocol packets are carried between computers in the data field of Ethernet frames. The system of high-level protocols carrying application data and the Ethernet system are independent entities that cooperate to deliver data between computers. Since the Ethernet system is unaffected by

the contents of the data field in the frame, different sets of computers running different high-level network protocols can share the same Ethernet. [3]

K.5 Advanced Multi –Media Application Design and Testing

According to the growing demands of multi-media (voice, data, video) applications in industry and their high bandwidth characteristics, nowadays Ethernet with more than Gigabit data rate is the best solution to satisfy these needs by its high and flexible range of bandwidth.

APPENDIX L: MEDIAS AND THEIR CHARACTERISTICS

Table L-1: UTP categories and their characteristics

Category	Bandwidth (MHz)	Data Rate (Mbps)	Twist Length (cm)
3	16	16	7.5-10
4	20	20	
5	100	100	0.6-0.85
5E	100	1000	0.6-0.85
6	250	1000	
7	600	1000	

Table L-2: Media systems and their characteristics

IEEE	Media System	Type	Max Length (m)	Encoding	Comments	Full Duplex Support
802.3e	1Base5	TP	500			
802.3a	10Base2	Coax	185	Manchester	Thin 0.25in Diameter	No
802.3	10Base5	Coax	500	Manchester	Thick 0.5in Diameter	No
802.3i	10BaseT	TP	100	Manchester	2pair CAT 3/4/5	Yes
802.3b	10Broad36	Coax	3600		75 ohm	No
802.3u	100BaseTX	TP	100	4B/5B	2pair CAT 5	Yes
802.3u	100BaseFX	2Multimode Optical Fibers	HD: 412 FD:2000	4B/5B	62.5/125 Micron	Yes
802.3u	100BaseVG*	TP	250		CAT 5 Token passing	
802.3ab	1000BaseT	TP	100	4D-PAM5	4pair CAT 5E / 6	Yes
802.3z	1000BaseSX	2Multimode Optical Fibers		8B/10B	Short Wave length	Yes
802.3z	1000BaseLX	Single Mode Fiber		8B/10B	Long wave Length	Yes

*Voice Grade

APPENDIX M: DATA ENCODING

Digitally encoded data are represented by a sequence of 0s and 1s. There are two popular encoding schemes named "NRZ (Non Return to Zero)" and "Manchester" encoding. In the first case, 0 is the high and 1 is the low voltage. In order to synchronize the clocks in both transmitters and receivers, Manchester encoding is invented.

In Manchester Line Encoding, there is a transition at the middle of each bit period. The mid-bit transition serves as a clocking mechanism (and also as data). A low to high transition represents a 1 and a high to low transition represents a 0.

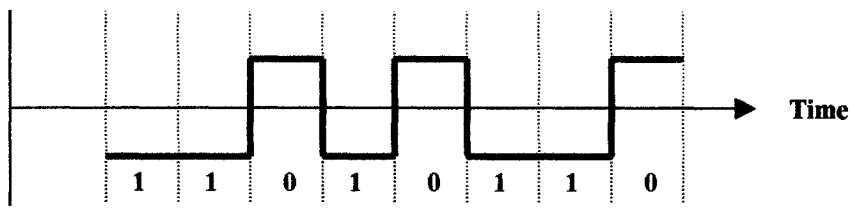


Figure M-1: NRZ encoding

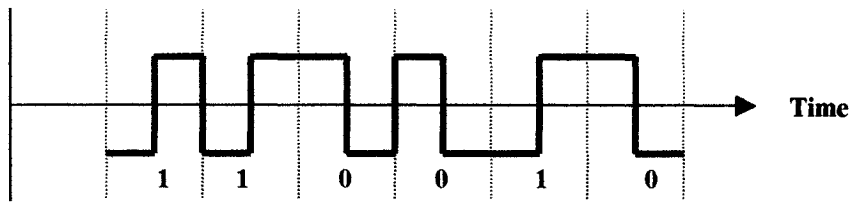


Figure M-2: Manchester encoding

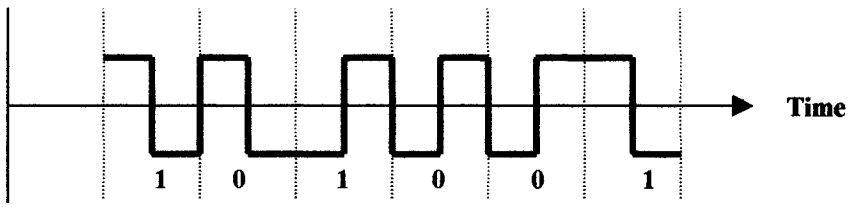


Figure M-3: Differential Manchester Encoding

Manchester line encoding has no DC component and there is always a transition available for synchronizing receives and transmits clocks. Some call it self-clocking line encoding or self-synchronizing code. It benefits need to less amount of bandwidth comparing to other line encoding.

Differential Manchester encoding is a variation of this method, which needs even less bandwidth than the first one. The difference is in the beginning of the interval. A 0 causes the signal to change at the beginning of the interval and a 1 causes the signal to remain where it was at the end of the previous interval. This characteristic leads to another advantage; we do not need to mark the wires to indicate the higher voltage, thus makes the wires cheaper than before.

APPENDIX N: NETWORKING PROTOCOL MODELS

N.1 TCP/IP Model

The Internet is based on a family of protocols called TCP/IP. The IP (Internet Protocol) part is the lowest level and it handles addressing and routing. IP packets can be sent over Ethernet, fiber optics, telephone lines, radio links, or any other medium that can move bits. Above IP are two other protocols: TCP and UDP. TCP, the Transmission Control Protocol, provides a connection-oriented and reliable byte stream form of communication. Applications can use TCP to open a logical connection to another host on the Internet, send data down that connection and receive data back. TCP handles the complex process of breaking the stream of data into smaller pieces, doing checksums on the data to make sure it arrives intact, sorting the packets as they arrive to make sure they are in the correct order, requesting retransmission when needed, handling flow control, and lots more. TCP uses IP to actually move the packets to other hosts.

TCP is the protocol on which most Internet applications run. For instance, SMTP defines the protocol used for the delivery of mail messages over the Internet. Telnet protocol allows users to log into remote computers via the Internet. FTP, allows Internet users to transfer files from remote computers. DNS provides a mapping of the host names to addresses.

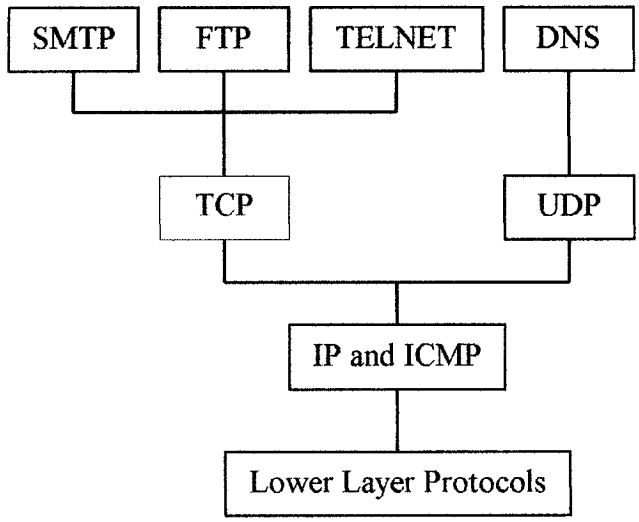


Figure N-1: Internet protocols

There is no official TCP/IP protocol model, as in the case of OSI. However, based on the protocol standards that have been developed, it is possible to organize the communications task into five relatively independent layers.

1. **Physical:** Defines the characteristics of the transmission medium, signalling rate and signal-encoding scheme.
2. **Network Access:** Concerns with the logical interface between an end system and a network.
3. **Internet:** Concerns with the routing of data from source to destination host on one or more networks connected by routers.
4. **Transport:** Provides end-to-end data transfer service. This layer may include reliability mechanisms. It hides the details of the underlying network or networks from the application layer.

5. **Application:** Provides communication between processes or applications on separate hosts.

Figure N-2 shows a comparison between the OSI reference model and the TCP/IP model.

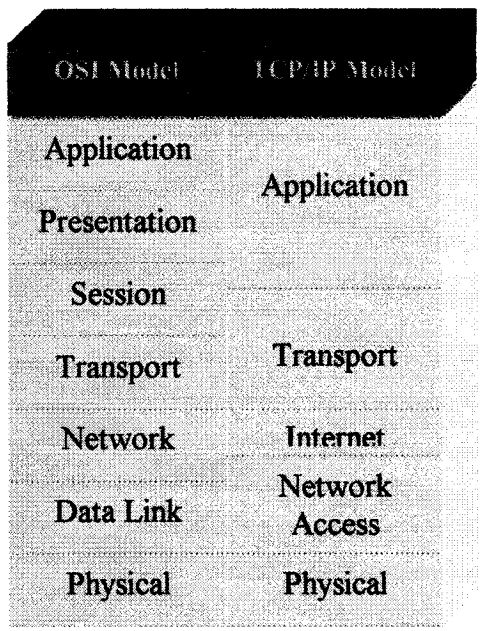


Figure N-2: OSI model and TCP/IP model

N.1.1 Sockets

Because so many networks applications run under the UNIX operating system, in particular Berkley UNIX, a separate set of TCP primitives was designed to be used by UNIX systems. Through these primitives, UNIX applications running on different computers connected to the Internet can communicate using a mechanism call " Socket".

In the other word, Sockets are a construct that allows client/sever models between programs running on different computers connected by the Internet. A client application

can connect to a server if it knows the server's address and the server's port number. The client can then send requests to the server. The server can accept connections from a variety of clients and respond accordingly. The server complemented the client's action by reading the file, dividing it up into small pieces that it stored into a series of packets, and sending the packets to the client. Many Internet applications follow this model. [13]

N.2 UDP/IP Model

UDP, the User Datagram Protocol, is quite different from TCP. It provides a connectionless, unreliable way of sending datagram from one host to another. It does not mean that UDP is somehow error-prone; it just means that there is no guarantee that a specific packet will arrive. However, if a packet does arrive, it will arrive intact. UDP is used by NFS (the Network File System), NTP (Network Time Protocol) and several others.

There are some important advantages in using UDP instead of TCP. The problem with TCP lies in its strengths. The fact that it does all kinds of flow control, error checking and sequencing means that it has a fair amount of overhead; this translates into lag, which is one of our enemies. UDP packets are relatively lightweight. Although they get lost occasionally, there will never be any congestion; they will not clog things up, they will just be discarded. Also, unlike TCP, UDP packets can be broadcasted on subnets (since there is no logical connection involved). However, because UDP is unreliable each message must be complete and self-contained. [10]

APPENDIX O: RING TOPOLOGY

Like a bus network, rings have the chained nodes (Figure O-1). The difference is that the end of the network comes back around to the first node, creating a complete circuit. In a ring network, each node takes a turn sending and receiving information by using a token. The token, along with any data, travels from the first node to the second node, which copies the data addressed to it and adds any data it wishes to send. Then, the second node passes the token and data to the third node, and so on until it comes back around to the first node which removes it and puts the token or another frame back onto the ring. Only the node with the token is allowed to send data. All the other nodes must wait for the token to come to them.

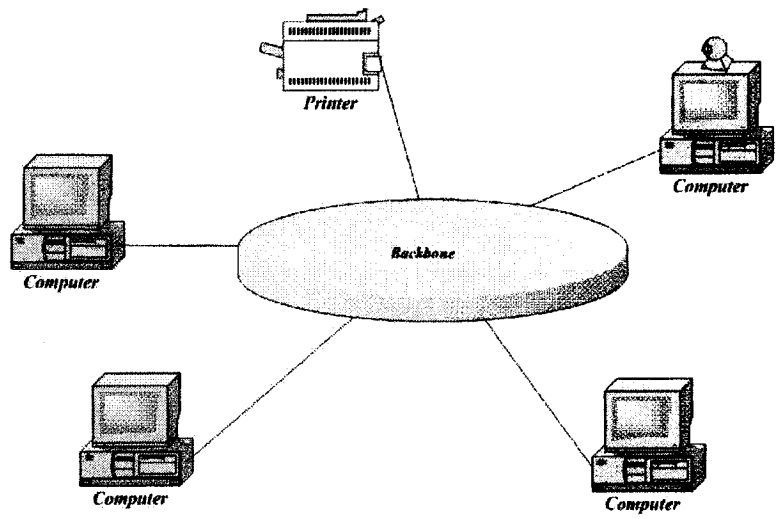


Figure O-1: Ring topology

GLOSSARY

Attachment Unit Interface (AUI): In 10 Mb/s CSMA/CD, the interface between the Medium Attachment Unit (MAU) and the data terminal equipment (DTE) within a data station. Note that the AUI carries encoded signals and provides for duplex data transmission. (See IEEE 802.3 Clauses 7 and 8.)

Bit rate (BR): The total number of bits per second transferred to or from the Media Access Control (MAC). For example, 100BASE-T has a bit rate of one hundred million bits per second (108 b/s).

Bit Time (BT): The duration of one bit as transferred to and from the Media Access Control (MAC). The bit time is the reciprocal of the bit rate. For example, for 100BASE-T the bit rate is 10–8 s or 10 ns. [2]

Broadband local area network (LAN): A local area network in which information is transmitted on modulated carriers, allowing coexistence of multiple simultaneous services on a single physical medium by frequency division multiplexing. (See IEEE 802.3 Clause 11.)

Carrier sense: In a local area network, an ongoing activity of a data station to detect whether another station is transmitting. The carrier sense signal indicates that one or more DTEs are currently transmitting.

Channel: A band of frequencies dedicated to a certain service transmitted on the broadband medium. (See IEEE 802.3 Clause 11.)

Coaxial cable segment: A length of coaxial cable made up from one or more coaxial cable sections and coaxial connectors, and terminated at each end in its characteristic impedance.

Collision: A condition that results from concurrent transmissions from multiple data terminal equipment (DTE) sources within a single collision domain.

Collision domain: A single, half duplex mode CSMA/CD network. If two or more Media Access Control (MAC) sub layers are within the same collision domain and both transmit at the same time, a collision will occur. MAC sub layers separated by a repeater are in the same collision domain. MAC sub layers separated by a bridge are within different collision domains. (See IEEE 802.3.)

Data terminal equipment (DTE): Any source or destination of data connected to the local area network.

Fiber optic cable: A cable containing one or more optical fibers as specified in IEEE 802.3, 15.3.1.

Fiber Optic Inter-Repeater Link (FOIRL): A Fiber Optic Inter-Repeater Link segment and its two attached Medium Attachment Units (MAUs). (See IEEE 802.3 Clause 15)

Jabber: A condition when a station transmits for a period of time longer than the maximum permissible packet length, usually due to a fault condition.

Jabber function: A mechanism for controlling abnormally long transmissions.

Link: The transmission path between any two interfaces of generic cabling.

Logical Link Control (LLC): The data link sub layer and is an IEEE mechanism for identifying the data carried in an Ethernet frame.

Media Access Control (MAC): The data link sub layer that is responsible for transferring data to and from the Physical Layer.

Media Independent Interface (MII): A transparent signal interface at the bottom of the Reconciliation sub layer. (See IEEE 802.3 Clause 22.)

Medium Attachment Unit (MAU): A device containing an Attachment Unit Interface (AUI), Physical Medium Attachment (PMA), and Medium Dependent Interface (MDI) that is used to connect a repeater or data terminal equipment (DTE) to a transmission medium.

Medium Dependent Interface (MDI): The mechanical and electrical interface between the transmission medium and the Medium Attachment Unit (MAU) (10BASE-T) or PHY (100BASE-T, 1000BASEX, or 1000BASE-T).

Packet: Consists of a data frame as defined previously, preceded by the Preamble and the Start Frame Delimiter, encoded, as appropriate, for the Physical Layer (PHY) type.

Pause: A mechanism for full duplex flow control. (See IEEE 802.3Annex 31B.)

Physical Layer entity (PHY): Within IEEE 802.3, the portion of the Physical Layer between the Medium Dependent Interface (MDI) and the Media Independent Interface (MII), or between the MDI and GMII, consisting of the Physical Coding Sub layer (PCS), the Physical Medium Attachment (PMA), and, if present, the Physical Medium Dependent (PMD) sub layers. The PHY contains the functions that transmit,

receive, and manage the encoded signals that are impressed on and recovered from the physical medium. (See IEEE 802.3 Clauses 23–26, 32, 36, and 40.)

Physical Signaling Sub layer (PLS): In 10BASE-T, that portion of the Physical Layer contained within the data terminal equipment (DTE) that provides the logical and functional coupling between the Medium Attachment Unit (MAU) and the Data Link Layer.

Twisted-pair cable: A bundle of multiple twisted pairs within a single protective sheath. (From ISO/ IEC 11801: 1995.)

Unshielded Twisted-Pair cable (UTP): An electrically conducting cable, comprising one or more pairs, none of which is shielded. There may be an overall shield, in which case the cable is referred to as unshielded twisted-pair with overall shield. (From ISO/IEC 11801: 1995.)