NEAR EAST UNIVERSITY

Faculty of Engineering

Department of Computer Engineering

Distance Control through Internet

Graduation Project
COM- 400

Student : Mohammed Al Jabiri (20010815)

Supervisor: Assoc. Prof. Dr. Rahib Abiyev

Nicosia - 2005

Acknowledgement

First of all I am happy to complete the task which I had given with blessing of God and also I am grateful to all the people in my life who have, supported me, advised me, Taught me and who have always encouraged me to follow my dreams and ambitions; My dearest parents, my brothers and sister, my friends and my tutors. They have taught me that no dream is unachievable. As in the words of Walt Disney "If you can dream it, you can do it."

I wish to thank my advisor, Assoc.Prof.Dr. Rahib Abiyev, for intellectual support, encouragement, and enthusiasm, which made this project possible, and his patience for correcting both my stylistic and scientific errors.

As I would like to thank the Near East University Stuff Professors, Assoc.Profs, Assist.Profs, Drs., Mrs., especially Prof.Dr.Fakhreddin Mamedov(Dean), Assoc.Prof.Dr. Adnan Khashman, Assoc.Prof.Dr. Dogan Ibrahim and Mr. Tayseer Alshanableh.

My sincerest thanks must go to my friends, Murad Al-zubi, Mohamed Mdookh, Adham Habboub, Bashar Alawartane, Ghayth Alnajdawi, Osama Alkurd, Muath Ismael and Ismael Elhamss who shared their suggestions and evaluations throughout the completion of my project. The comments from these friends enabled me to present this project successfully.

And above, I thank God for giving me stamina and courage to achieve my objectives.

Mohammed Al Jabiri

ABSTRACT

WAN is a geographically distributed network composed of local area networks (LANs) joined into a single large network using services provided by common carriers. Wide area networks (WANs) are commonly implemented in enterprise networking environments in which company offices are in different cities, states, or countries or on different continents.

WAN technologies were previously limited to expensive leased lines such as T1 lines, slow packet-switching services such as X.25, cheap but low-bandwidth solutions such as modems, and dial-up Integrated Services Digital Network (ISDN) connections, but this has changed considerably in recent years.

Frame relay services provide high-speed packet-switching services that offer more bandwidth than X.25, and virtual private networks (VPNs) created using Internet Protocol (IP) tunneling technologies enable companies to securely connect branch offices by using the Internet as a backbone service.

Intranets and extranets provide remote and mobile users with access to company resources and applications and provide connectivity with business partners and resellers. Wireless networking technologies allow roaming users to access network resources by using cell-based technologies. Digital Subscriber Line (DSL) services provide T1 speeds at much lower costs than dedicated T1 circuits. These and other new technologies continue to evolve and proliferate, allowing enterprise network administrators to implement and administer a highly diverse range of WAN solutions.

TABLE OF CONTENTS

INTRODUCTION

Now a days every where in this world rather a small office or big we need to have a network even in a small office we can have many computers sharing a single or two printers. All this is possible because of networking. There are three main types of networking one which is in a small office called as LAN as local area network. Then there is a kind of networking which is used to connect distant offices means in other works a network in which we can connect LAN of one office to LAN of other office called as WAN. Two or more than two LAN combine to make a WAN and the third type is MAN which is more advance then LAN.

To connect two or more LAN to make WAN we use a device called router as from its name is specified that it routes the data from one network to another network. Router is the main component to make communication between many networks as it has its on operating system and program and it is its responsibility that which data must be sent to which network and this process is called routing.

My first chapter is all about explaining what is WAN and what WAN technologies are. It is an introduction chapter in which I have explained about WAN in detail and what are the features of WAN and what are the devices used in WAN to make communication possible between two networks.

My second chapter is about client server connection through internet. It includes the definition of the client and server and the evolution of the clients and servers in our era. I also explained some architecture of the client and server.

My third chapter is about internet protocols and what is the internet protocol and it classifications and classes. I present also a brief explanation about address resolution protocol (ARP), and internet routing and some information about transmission control protocol (TCP)

The fourth and the last chapter is about controlling using the parallel port in which I controlled a small car using visual basic programming language.
I also used some circuit diagrams and instruments in which I attached the car to the parallel port to move it.

In conclusion the obtained important results and contribution is presented.

Chapter ONE
Wide Area Network

1.1 Overview

A wide-area network (WAN) is a data communications network covering a relatively broad geographic area and often using transmission facilities provided by the common carriers. WAN technologies function at the lower three layers of the OSI reference model: the physical layer, the data link layer, and the network layer.

The following figure shows the relationship between the common WAN technologies and the OSI model:

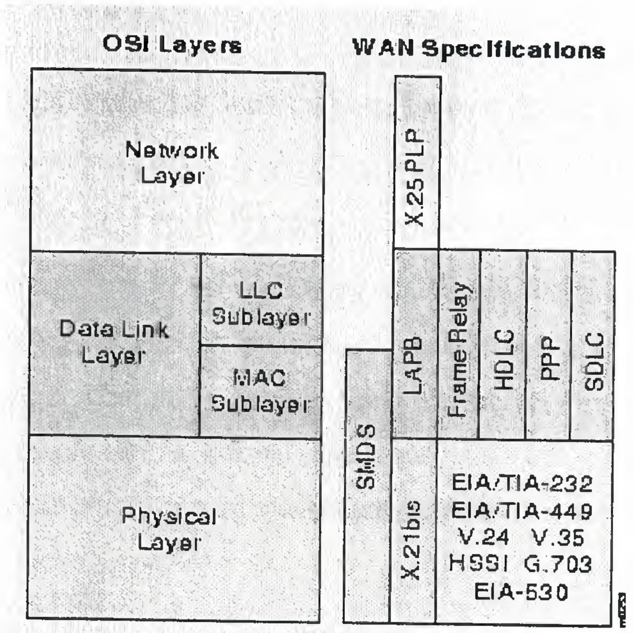


Figure 1.1 Shows WAN Specification and OSI Model

1.2 Point-to-Point Links

A point-to-point link provides a single, reestablished WAN communications path from the customer premises, through a carrier network (the telephone company), to a remote network. Point-to-point links are also known as leased lines. The established path is permanent and is fixed for each remote network reached through the carrier facilities. Point-to-point links are reserved by the carrier company for the private use of the customer.

Point-to-point links allow two types of transmission:

- Datagram transmission -- Datagram transmissions are composed of individually addressed frames.
- Data stream transmission -- Data stream transmissions are composed of a stream of data for which address checking occurs only once.

The following figure illustrates a typical point-to-point link through a WAN:

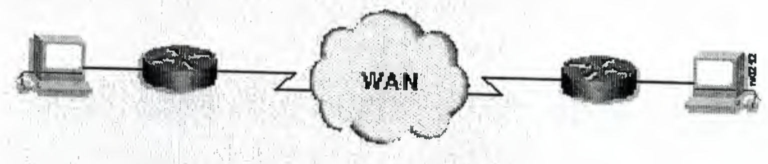


Figure 1.2 Shows a Point-to-Point Link

1.3 Circuit Switching

Circuit switching is a WAN switching method in which a dedicated physical circuit through a carrier network is established, maintained, and terminated for each communication session. Circuit switching, used extensively in telephone company networks, operates much like a normal telephone call. Integrated Services Digital Network (ISDN) is an example of a circuit-switched WAN technology.

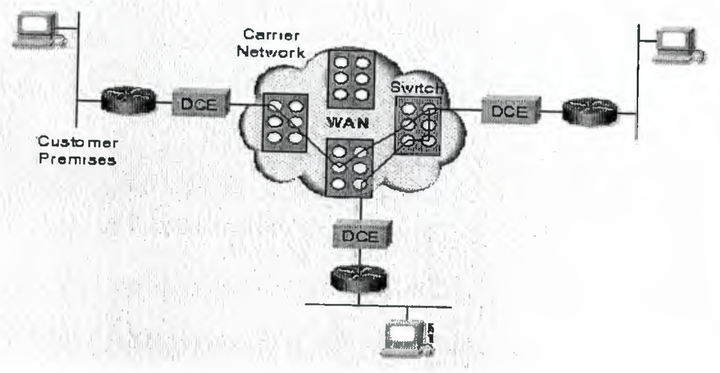


Figure 1.3 Shows a circuit-switched WAN

1.4 Packet Switching

Packet switching is a WAN switching method in which network devices share a single point-to-point link to transport packets from a source to a destination across a carrier network. Statistical multiplexing is used to allow devices to share these circuits. Asynchronous Transfer Mode (ATM), Frame Relay, Switched Multi-megabit Data Service (SMDS), and X.25 are examples of packet-switched WAN technologies.

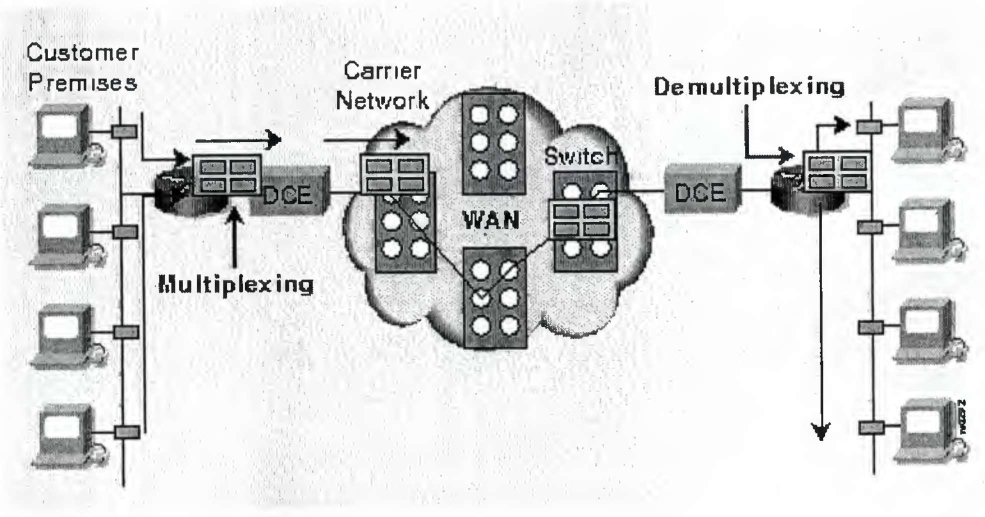


Figure 1.4 Illustrates a packet-switched WAN

1.5 WAN Devices

There are numerous types of devices used in WANs. These include routers, ATM switches, multiplexers, various WAN switches, access servers, modems, CSU/DSUs, and terminal adapters.

1.5.1 WAN Switch

A WAN switch is a multi-port internetworking device used in carrier networks. These devices typically switch Frame Relay, X.25, SMDS, and other WAN traffic. They operate at the data link layer of the Open System Interconnection (OSI) reference model.

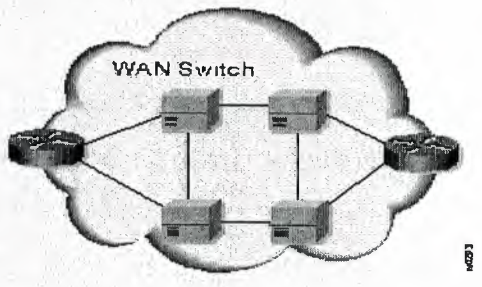


Figure 1.5 Shows Two Routers of a WAN connected by switches:

1.5.2 Access Server

An access server serves as a concentration point for dial-in and dial-out connections.

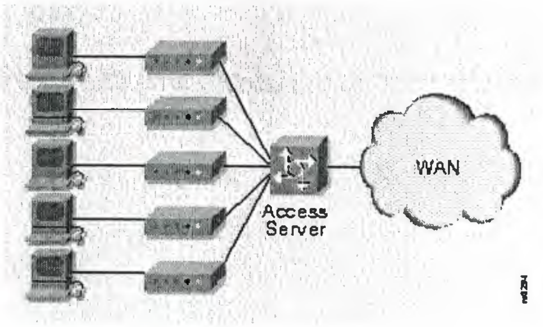


Figure 1.6 Shows an access server concentrating dial-out connections

1.5.3 Modem

A device that converts digital and analog signals, allowing data to be transmitted over voice-grade telephone lines. At the source, digital signals are converted to a form suitable for transmission over analog communication facilities. At the destination, analog signals are returned to their digital form.

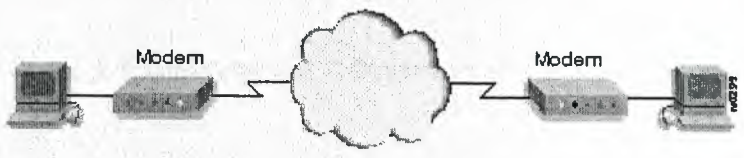


Figure 1.7 Simple modem-to-modem connection

1.5.4 CSU/DSU

A channel service unit/data service unit (CSU/DSU) is a digital interface device (or sometimes two separate digital devices) that adapts the physical interface on a data terminal equipment (DTE) device (such as a terminal) to the interface of a data circuit-terminating (DCE) device (such as a switch) in a switched carrier network. The CSU/DSU also provides signal timing for communication between these devices.

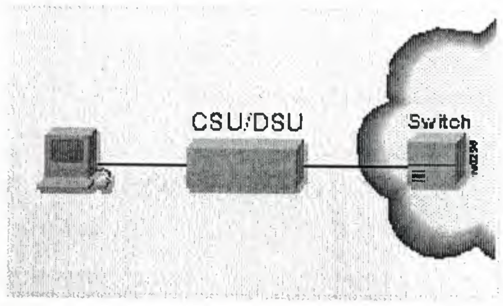


Figure 1.8 The Placement of the CSU/DSU

1.5.5 ISDN Terminal Adapter

An Integrated Services Digital Network (ISDN) terminal adapter is a device used to connect ISDN Basic Rate Interface (BRI) connections to other interfaces such as EIA/TIA-232. A terminal adapter is essentially an ISDN modem.

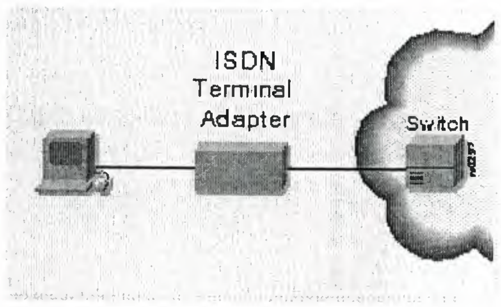


Figure 1.9 The Terminal Adapter in an ISDN Environment

1.6 WAN Technology Types

Following is a list of some of the common WAN technologies:

- Frame Relay

Frame Relay is a high-performance, packet-switched WAN protocol.

- High Speed Serial Interface (HSSI)

HSSI is a network standard for high-speed serial communications over WAN links.

- Integrated Services Data Network (ISDN)

ISDN consists of communication protocols proposed by telephone companies to permit telephone networks to carry data, voice, and other source material.

- Point-to-Point Protocol (PPP)

PPP provides router-to-router and host-to-network connections over synchronous and asynchronous circuits.

- Synchronous Data Link Control (SDLC)

SDLC is an IBM bit-synchronous data link layer protocol.

- Switched Multi-megabit Data Service (SMDS)

SMDS is a high-speed, packet-switched WAN technology.

- X.25

X.25 is an ITU-T WAN communications protocol.

1.7 Network Essentials

Network essentials are the things we must have to take care of to establish a good network between two or more networks. It include the OSI reference model which help in complete establishment of the network then we have protocols then we have WAN hardware all these things are very essential for a network

1.7.1 The OSI Model

- OSI is a layer model Developed by ISO it is a seven layer architecture help in communication between two computers.
- International Standards Organization (ISO) specifications for network architecture. Called the Open Systems Interconnect or OSI model.
- Seven layered model, higher layers have more complex tasks. Each layer provides services for the next higher layer. Each layer communicates logically with its associated layer on the other computer.
- Packets are sent from one layer to another in the order of the layers, from top to bottom on the sending computer and then in reverse order on the receiving computer.

OSI Layers Names and a precise description is as follows:

- Presentation.
- Session.
- Transport.
- Network.
- Data Link.
- Physical.
- Application Layer.

1.7.1.1 Presentation Layer

- Determines the format used to exchange data among the networked computers.
- Translates data from a format from the Application layer into an intermediate format.
- Responsible for protocol conversion, data translation, data encryption, data compression, character conversion, and graphics expansion.
- Redirector operates at this level.

1.7.1.2 Session Layer

- Allows two applications running on different computers to establish use and end a connection called a Session.
- Performs name recognition and security.
- Provides synchronization by placing checkpoints in the data stream.
- Implements dialog control between communicating processes.

1.7.1.3 Transport Layer

- Responsible for packet creation.
- Provides an additional connection level beneath the Session layer.
- Ensures that packets are delivered error free, in sequence with no losses or duplications.
- Unpacks, reassembles and sends receipt of messages at the receiving end.
- Provides flow control, error handling, and solves transmission problems.

1.7.1.4 Network Layer

- Responsible for addressing messages and translating logical addresses and names into physical addresses.
- Determines the route from the source to the destination computer.
- Manages traffic such as packet switching, routing and controlling the congestion of data.

1.7.1.5 Data Link Layer

- o Sends data frames from the Network layer to the Physical layer.
- o Packages raw bits into frames for the Network layer at the receiving end.
- o Responsible for providing error free transmission of frames through the Physical layer.

1.7.1.6 Physical Layer

- o Transmits the unstructured raw bit stream over a physical medium.
- o Relates the electrical, optical mechanical and functional interfaces to the cable.
- o Defines how the cable is attached to the network adapter card.
- o Defines data encoding and bit synchronization.

1.7.1.7 Application Layer

- o Serves as a window for applications to access network services.
- o Handles general network access, flow control and error recovery.

1.8 Protocols

- Protocols are rules and procedures for communication.

1.8.1 How Protocols Work?

The Sending Computer does the following jobs:

- Breaks data into packets.
- Adds addressing information to the packet.
- Prepares the data for transmission.

The Receiving Computer does the following jobs:

- Takes the packet off the cable.
- Strips the data from the packet.
- Copies the data to a buffer for reassembly.

- Passes the reassembled data to the application.

1.8.2 Protocol Stacks (or Suites)

- A combination of protocols, each layer performing a function of the communication process.
- Ensure that data is prepared, transferred, received and acted upon.

1.8.3 The Binding Process

- Allows more than one protocol to function on a single network adapter card. (e.g. both TCP/IP and IPX/SPX can be bound to the came card.
- Binding order dictates which protocol the operating systems uses first.
- Binding also happens with the Operating System architecture: for example, TCP/IP may be bound to the NetBIOS session layer above and network card driver below it. The NIC device driver is in turn bound to the NIC.

1.8.4 Standard Stacks

- ISO/OSI.
- IBM SNA (Systems Network Architecture).
- Digital DECnet.
- Novell NetWare.
- Apple AppleTalk.
- TCP/IP.

Protocol types map roughly to the OSI Model into three layers:

Application Level Service Users

- Application Layer.
- Presentation Layer.
- Session Layer.

Transport Services

- Transport Layer.

- Network Layer.
- Data Link Layer.
- Physical Layer.

1.8.5 The IEEE protocols at the Physical Layer

1.8.5.1 802.3 (CSMA /CD - Ethernet)

- Logical bus network.
- Can transmit at 10 Mbps.
- Data is transmitted on the wire to every computer but only those meant to receive respond.
- CSMA /CD protocol listens and allows transmission when the wire is clear.

1.8.5.2 802.4 (Token Passing)

- Bus layout that used token passing.
- Every computer receives all of the data but only the addressed computers respond.
- Token determines which computer can send.

1.8.5.3 802.5 (Token Ring)

- Logical ring network; physical set up as star network.
- Transmits at 4 Mbps or 16 Mbps.
- Token determines which computer can send.

1.9 Important Protocols

1.9.1 TCP/IP

- Provides communications in a heterogeneous environment.
- Routable, defacto standard for internetworking.
- SMTP, FTP, SNMP are protocols written for TCP/IP.
- Disadvantages are size and speed.

1.9.2 NetBEUI

- NetBIOS extended user interface.
- Originally, NetBIOS and NetBEUI were tightly tied together but, NetBIOS has been separated out to be used with other routable protocols. NetBIOS acts as a tool to allow applications to interface with the network; by establishing a session with another program over the network
- NetBIOS operates at the Session layer.
- Small, fast and efficient.
- Compatible with most Microsoft networks.
- Not routable and compatible only with Microsoft networks.

1.9.3 X.25

- Protocols incorporated in a packet switching network of switching services.
- Originally established to connect remote terminals to mainframe hosts.

1.9.4 XNS

- Xerox Network System.
- Developed for Ethernet LANs but has been replaced by TCP/IP.
- Large, slow and produces a lot of broadcasts.

1.9.5 IPX/SPX and NWLink

- Used for Novell networks.
- Small and fast.
- Routable.

1.9.6 APPC

- Advanced Program to Program Communication.
- Developed by IBM to support SNA.
- Designed to enable application programs running on different computers to communicate and exchange data directly.

1.9.7 AppleTalk

- Apple's proprietary protocol stack for Macintosh networks.

1.9.8 OSI Protocol Suite

- Each protocol maps directly to a single layer of the OSI model.

1.9.9 DECnet

- Digital Equipment's proprietary protocol stack.
- Defines communications over Ethernet, FDDI MAN's and WAN's.
- DECnet can also use TCP/IP and OSI protocols as well as its own protocols.
- Routable.

1.10 Network Hardware

Some components can be installed which will increase the size of the network within the confines of the limitations set by the topology. These components can:

- Segment existing LANs so that each segment becomes its own LAN.
- Join two separate LANs.
- Connect to other LANs and computing environments to join them into a larger comprehensive network.

1.10.1 Modems

- Modems share these characteristics.
- A serial (RS-232) interface.
- An RJ-11C telephone line connector.
- Telephones use analog signal; computers use digital signal. A modem translates between those two.
- BAUD refers to the speed of the oscillation of the sound wave on which a bit of data is carried over the telephone wire.
- The BPS can be greater than the baud rate due to compression and encode data so that each modulation of sound can carry more than one bit of data is carried

over the telephone line. For example, a modem that modulates at 28,000 baud can actually send at 115,200 bps => bps is the most important parameter when looking at throughput.

There are 2 types of modems

1.10.1.1 Asynchronous Communications (Async)

- Use common phone lines.
- Data is transmitted in a serial stream.
- Not synchronized, no clocking device => no timing.
- Both sending and receiving devices must agree on a start and stop bit sequence.
- error control
  - A parity bit is used in an error checking and correction scheme called parity checking.
  - It checks to see if the # of bits sent = # of bits received.
  - The receiving computer checks to make sure that the received data matches what was sent.
  - 25 % of the data traffic in async communications consists of data control and coordination.
  - MNP (Microcom Network Protocol) has become the standard for error control.
  - Later LAPM (Link Access Procedure for Modems) is used in V.42 modems (57,600 baud).

1.10.1.2 Synchronous Communication

- Relies on a timing scheme coordinated between two devices to separate groups of bits and transmit them in blocks known as frames.
- NO start and stop bits =. A continuous stream of data because both know when the data starts and stops.
- If there's error, the data is retransmitted.
- some synchronous protocol perform the following that asynchronous protocols don't:
  1. Format data into blocks.

2. Add control info.
3. Check the info to provide error control.

- the primary protocols in synchronous communication are:
    1. Synchronous data link control (SDLC).
    2. High-level data link control (HDLC).
    3. Binary synchronous communication protocol (bisync).
- Synchronous communications are used in almost all digital and network communications.
- 2 types of telephone lines:
    1. Public dial network lines (dial-up lines) - manually dial up to make a connection.
    2. Leased (dedicated) lines - full time connections that do not go through a series of switches, 56 Kbps to 45 Mbps.

1.10.2 Repeaters

- Repeaters
    - EXTEND the network segment by REGENERATING the signal from one segment to the next.
    - Repeaters regenerate BASEBAND, digital signals.
    - Don't translate or filter anything.
    - Is the least expensive alternative.
    - Work at the Physical layer of OSI.
- Both segments being connected must use the same access method e.g. an 802.3 CSMA/CD (Ethernet) LAN segment can't be joined to an 802.5 (Token Ring) LAN segment. Another way of saying this is the Logical Link Protocols must be the same in order to send a signal.
- BUT repeaters CAN move packets from one physical medium to another: for example can take an Ethernet packet from a thinnet coax and pass it on to a fiber-optic segment. Same access method is being used on both segments, just a different medium to deliver the signal.

- They send every bit of data on => NO FILTERING, so they can pass a broadcast storm along from on segment to the next and back. So you want to use a repeater when there isn't much traffic on either segment you are connecting.
- There are limits on the number of repeaters which can be used. The repeater counts as a single node in the maximum node count associated with the Ethernet standard [30 for thin coax].
- Repeaters also allow isolation of segments in the event of failures or fault conditions. Disconnecting one side of a repeater effectively isolates the associated segments from the network.
- Using repeaters simply allows you to extend your network distance limitations. It does not give you any more bandwidth or allow you to transmit data faster.
- Why only so many repeaters are allowed on a single network: "propagation delay". In cases where there are multiple repeaters on the same network, the brief time each repeater takes to clean up and amplify the signal, multiplied by the number of repeaters can cause a noticeable delay in network transmissions.
- It should be noted that in the above diagram, the network number assigned to the main network segment and the network number assigned to the other side of the repeater are the same.
- In addition, the traffic generated on one segment is propagated onto the other segment. This causes a rise in the total amount of traffic, so if the network segments are already heavily loaded, it's not a good idea to use a repeater.
- A repeater works at the Physical Layer by simply repeating all data from one segment to another.

1.10.3 Bridges

- Have all the abilities of a repeater.
- Bridges can
    - Take an overloaded network and split it into two networks, therefore they can divide the network to isolate traffic or problems and reduce the traffic on both segments.
    - Expand the distance of a segment.
    - Link UNLIKE PHYSICAL MEDIA such as twisted-pair (10Base T) and coaxial Ethernet (10Base2).

    - VERY IMPORTANT: they can link UNLIKE ACCESS CONTROL METHODS, on different segments such as Ethernet and Token Ring and forward packets between them. Exam Cram says this is a Translation Bridge that can do this - not all bridges - but my observation is questions don't necessarily mention the distinction.

- Bridges work at the Data Link Layer of the OSI model => they don't distinguish one protocol from the next and simply pass protocols along the network. (use a bridge to pass NetBEUI, a non-routable protocol, along the network).
- Bridges actually work at the MEDIA ACCESS CONTROL (MAC) sub layer. In fact they are sometimes called Media Access Control layer bridges. Here's how they deal with traffic:
    - They listen to all traffic. Each time the bridge is presented with a frame, the source address is stored. The bridge builds up a table which identifies the segment to which the device is located on. This internal table is then used to determine which segment incoming frames should be forwarded to. The size of this table is important, especially if the network has a large number of workstations/servers.
    - They check the source and destination address of each PACKET.
    - They build a routing table based on the SOURCE ADDRESSES. Soon they know which computers are on which segment.
    - Bridges are intelligent enough to do some routing:
        - If the destination address is on the routing table and is on the SAME SEGMENT, the packet isn't forwarded. Therefore, the bridge can SEGMENT network traffic.
        - If the destination address is the routing table, and on a remote segment, the bridge forwards the packet to the correct segment.
        - If the destination address ISN'T on the routing table, the bridge forwards the packet to ALL segments.
        - BRIDGES SIMPLY PASS ON BROADCAST MESSAGES, SO they too contribute to broadcast storms and don't help to reduce broadcast traffic.

- Comparison of Bridges and Repeaters
    - Bridges
        - Regenerate data at the packet level.
        - Accommodate more nodes than repeaters.
        - Provide better network performance than repeaters because they segment the network.

1.10.4 Routers

- Determine the best path for sending data and filtering broadcast traffic to the local segment. They DON'T pass on broadcast traffic.
- Work at the Network layer of OSI => them can switch and route packets across network segments.
- They provide these functions of a bridge:
    - Filtering and isolating traffic.
    - Connecting network segments.
- routing table contains
    1. All known network addresses.
    2. How to connect to other networks.
    3. Possible paths between those routers.
    4. Costs of sending data over those paths.
    5. Not only network addresses but also media access control sub layer addresses for each node.
- Routers
    - Require specific addresses: they only understand network numbers which allow them to talk to other routers and local adapter card addresses.
    - Only pass Packets to the network segment they are destined for.
    - Routers don't talk to remote computers, only to other routers.
    - They can segment large networks into smaller ones.
    - They act as a safety barrier (firewall) between segments.

    - They prohibit broadcast storms, because broadcasts and bad data aren't forwarded.
    - Are slower than most bridges.
    - Can join dissimilar access methods: a router can route a packet from a TCP/IP Ethernet network to a TCP/IP Token Ring network.

- Routers don't look at the destination computer address. They only look at the NETWORK address and they only pass on the data if the network address is known => less traffic.
- Routable protocols:
    - DECnet, IP, IPX, OSI, XNS, DDP (Apple).
    - Routable protocols have Network layer addressing embedded.

- Non-routable protocols:
    - LAT, NetBEUI, DLC.
    - Non-routable protocols don't have network layer addressing.

1.10.6 Hubs

There are many types of hubs:

- Passive hubs are doesn't require power and are simple splitters or combiners that group workstations into a single segment.
- Active hubs require power and include a repeater function and are thus capable of supporting many more connections.
- Intelligent hubs provide
    - Packet switching.
    - Traffic routing.

1.10.7 Gateways

- The TRANSLATOR -- allows communications between dissimilar systems or environments.
- A gateway is usually a computer running gateway software connecting two different segments. For example an Intel-based PC on one segment can both communicate and share resources with a Macintosh computer or an SNA

mainframe. Use gateways when different environments need to communicate. One common use for gateways is to translate between personal computers and mainframes.

- GSNW is a gateway to allow Microsoft clients using SMB to connect to a NetWare server using NCP.
- Gateways work at the Application --> Transport layer.
- They make communication possible between different architectures and environments.
- They perform protocol AND data conversion / translation.
- They take the data from one environment, strip it, and re-package it in the protocol stack from the destination system.
- They repackage and convert data going from one environment to another so that each environment can understand the other environment's data.
- Gateway links two systems don't use the same
  1. Protocols.
  2. Data formatting structure.
  3. Languages.
  4. Architecture.

- They are task specific in that they are dedicated to a specific type of conversion: e.g. "Windows NT Server -> SNA Server Gateway".
- Usually one computer is designated as the gateway computer. This adds a lot of traffic to that segment.
- Disadvantages
  - They slow things down because of the work they do.
  - They are expensive.
  - Difficult to configure.

CHAPTER TWO
CLIENT SERVER CONNECTION THROUGH INTERNET

2.1 Overview

CLIENT/SERVER concepts have roots in early computer systems, so it is useful to briefly review the history of modern computing. Early computers (mainframes) were typically in a "glass room," with special power and air conditioning, and attended to by a priesthood of system programmers.

Users typically shared a pool of "dumb" terminals and had to rely on centralized printing and storage resources. In this environment, the mainframe did all the processing, and users had no local computing horsepower.

In the late 1970s and early 1980s, smaller systems (minicomputers) were developed that required less power and air conditioning. Individual business unit owners wanted their own systems, more suited to their needs. Once a substantial number of applications were developed on these computers, inevitably other departments wanted access to them.

Somewhat concurrently, Apple, IBM and others developed the personal computer. Initially it seemed to have little application beyond that of the hobbyist. However, as powerful, shrink-wrapped applications were developed, and arcane operating systems such as MS-DOS were replaced by more user-friendly systems (e.g., Macintosh, Windows), users began to want to use PCs in the corporate computing environment.

This allowed for a new army of computer users who cared little and understood even less about computer "systems." All they wanted was the latest software applications at their fingertips, but they needed the backup and redundant architecture that previously resided only in the glass room.

Concurrently, systems that were more scalable were introduced using common chip sets and operating systems, with features such as redundant power and multiple processors. They were far more powerful than "personal" computers although they were based largely on the same architecture but they were considerably smaller and less costly than mainframes or even minicomputers.

One last piece of technology was needed to make client/server architecture a reality. In the late 1970s, Xerox developed the standards and technology that we know today as Ethernet. This provided a standard means of linking together computers from different manufacturers and formed the basis for modern local area networks (LANs) and wide area networks (WANs).

At this point, all the pieces were in place to develop client/server systems:

- A strong business need for decentralized computing horsepower
- Standard, powerful computers with user-friendly interfaces
- Mature, shrink-wrapped user applications with widespread acceptance
- Inexpensive, modular systems designed with enterprise-class

2.2 Introduction to Client Server Systems

Most of the initial client/server success stories involve small-scale applications that provide direct or indirect access to transactional data in legacy systems. The business need to provide data access to decision makers, the relative immaturity of client/server tools and technology, the evolving use of wide area networks and the lack of client/server expertise make these attractive yet low risk pilot ventures. As organizations move up the learning curve from these small-scale projects towards mission-critical applications, there is a corresponding increase in performance expectations, uptime requirements and in the need to remain both flexible and scalable. In such a demanding scenario, the choice and implementation of appropriate architecture becomes critical. In fact one of the fundamental questions that practitioners have to contend with at the start of every client/server project is which architecture is more suitable for this project Two Tier or Three Tier architecture.

Architecture affects all aspects of software design and engineering. The architect considers the complexity of the application, the level of integration and interfacing required the number of users, their geographical dispersion, the nature of networks and the overall transactional needs of the application before deciding on the type of architecture. An inappropriate architectural design or a flawed implementation could result in horrendous response times. The choice of architecture also affects the development time and the future flexibility and maintenance of the application. This report defines the basic concepts of client/server architecture, describes the two tier and three tier or N-tier architectures.

2.3 Definition of Client and Server

In the simplest sense, the client and server can be defined as follows:

A **client** is an individual user's computer or a user application that does a certain amount of processing on its own. It also sends and receives requests to and from one or more servers for other processing and/or data.

A **server** consists of one or more computers that receive and process requests from one or more client machines. A server is typically designed with some redundancy in power, network, and computing and file storage. However, a machine with dual processors is not necessarily a server. An individual workstation can function as a server.

Sometimes the term "server" or "client" may refer to the software rather than the computer. Thus, a "mail client" may refer to the mail software that resides on a client machine, rather than the machine itself.

2.4 The Evolution of Clients and Servers

The earliest versions of client/server were merely shared files; when a user needed a file, it was copied from the server to the local machine. When finished, it was returned to the server. It was necessary to develop certain rules to handle conflict and synchronization issues. So as client/server systems evolved, they contained built-in synchronization and sharing engines. Client/server also embodies the concepts of user accounts and sharing of resources the system must separate and keep track of different user's files and applications during a user session, then free up those resources for another user session. As client/server applications evolved, the functionality of the application was separated logically into two parts: the processes requiring the majority of the computing horsepower were put on the server, and the user interface and less processor intensive processes were put on the client.

2.5 Client/Server Examples

A common client/server example is a print server. Most people have probably noticed a temporary lockup or slowdown when a document is printed on a stand alone PC, especially if the document is complex. One can attach a printer to a PC and then share it with other users across the network. However, if everyone on the network simultaneously prints to that shared printer, it would likely lock up or even crash. Therefore, many times a machine is dedicated solely to handle printing a print server. It "serves" print requests to all users, and off-loads this task from local machines. Another example is a mail server which functions much like a post office, receiving mail centrally and delivering individual messages to individual clients.

2.6 Putting It All Together

In a small organization, a single server machine may serve more than one function, if the functions are simple enough. One or more applications may reside on a single server machine, with the server being divided into different "logical" partitions. In a large corporate environment, there may be many servers for separate tasks.

There is typically a primary domain controller (PDC), which authenticates users and controls access and log in to the computer system itself. There may be a mail server, which processes e-mail. There may also be a file server typically containing large disk drives and individual user directories to store user files in a uniform way. And there may be separate application servers for accounting, billing, customer care, Web, e-commerce, database, transaction, manufacturing, inventory, etc. They are typically linked together using integration software (frequently called middleware) so that one can access many server applications from a single (client) machine, through a common interface, typically a browser.

Although client/server in its simplest form is two-tier (server and client), there are newer, more powerful architectures that are three-tier (where application logic lives in the middle-tier and it is separated from the data and user interface) or even n-tier (where there are several middle-tier components within a single business transaction) in nature. Sometimes client/server is referred to as distributed computing; they have the same basic concepts.

2.7 Client Server Architecture

When considering a move to client/server computing, whether it is to replace existing systems or introduce entirely new systems, practitioners must determine which type of architecture they intend to use. The vast majority of end user applications consist of three components: presentation, processing, and data. The client/server architectures can be defined by how these components are split up among software entities and distributed on a network. There are a variety of ways for dividing these resources and implementing client/server architectures. This paper will focus on the "serves" print requests to all users, and off-loads this task from local machines. Another example is a mail server which functions much like a post office, receiving mail centrally and delivering individual messages to individual clients.

2.6 Putting It All Together

In a small organization, a single server machine may serve more than one function, if the functions are simple enough. One or more applications may reside on a single server machine, with the server being divided into different "logical" partitions. In a large corporate environment, there may be many servers for separate tasks.

There is typically a primary domain controller (PDC), which authenticates users and controls access and log in to the computer system itself. There may be a mail server, which processes e-mail. There may also be a file server typically containing large disk drives and individual user directories to store user files in a uniform way. And there may be separate application servers for accounting, billing, customer care, Web, e-commerce, database, transaction, manufacturing, inventory, etc. They are typically linked together using integration software (frequently called middleware) so that one can access many server applications from a single (client) machine, through a common interface, typically a browser.

Although client/server in its simplest form is two-tier (server and client), there are newer, more powerful architectures that are three-tier (where application logic lives in the middle-tier and it is separated from the data and user interface) or even n-tier (where there are several middle-tier components within a single business transaction) in nature. Sometimes client/server is referred to as distributed computing; they have the same basic concepts.

2.7 Client Server Architecture

When considering a move to client/server computing, whether it is to replace existing systems or introduce entirely new systems, practitioners must determine which type of architecture they intend to use. The vast majority of end user applications consist of three components: presentation, processing, and data. The client/server architectures can be defined by how these components are split up among software entities and distributed on a network. There are a variety of ways for dividing these resources and implementing client/server architectures. This paper will focus on the most popular forms of implementation of two-tier and three-tier client/server computing systems.

2.7.1 Two-tier Architecture

Although there are several ways to architect a two-tier client/server system, we will focus on examining what is overwhelmingly the most common implementation. In this implementation, the three components of an application (presentation, processing, and data) are divided among two software entities (tiers): client application code and database server (Figure 2.1). A robust client application development language and a versatile mechanism for transmitting client requests to the server are essential for a two-tier implementation. Presentation is handled exclusively by the client, processing is split between client and server, and data is stored on and accessed via the server. The PC client assumes the bulk of responsibility for application (functionality) logic with respect to the processing component, while the database engine with its attendant integrity checks, query capabilities and central repository functions handles data intensive tasks. In a data access topology, a data engine would process requests sent from the clients. Currently, the language used in these requests is most typically a form of SQL. Sending SQL from client to server requires a tight linkage between the two layers. To send the SQL the client must know the syntax of the server or have this translated via an API (Application Program Interface). It must also know the location of the server, how the data is organized, and how the data is named. The request may take advantage of logic stored and processed on the server which would centralize global tasks such as validation, data integrity, and security. Data returned to the client can be manipulated at the client level for further sub selection, business modeling, "what if" analysis, reporting, etc.

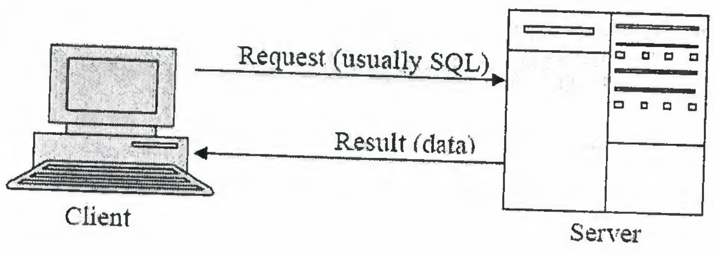


Figure 2.1 Data Access Topology for two-tier architecture. Majority of functional logic exists at the client level

The most compelling advantage of a two-tier environment is application development speed. In most cases a two-tier system can be developed in a small fraction of the time it would take to code a comparable but less flexible legacy system. Using any one of a growing number of PC-based tools, a single developer can model data and populate a database on a remote server, paint a user interface, create a client with application logic, and include data access routines. Most two-tier tools are also extremely robust. These environments support a variety of data structures, including a number of built in procedures and functions, and insulate developers from many of the more mundane aspects of programming such as memory management. Finally these tools also lend themselves well to iterative prototyping and rapid application development (RAD) techniques, which can be used to ensure that the requirements of the users are accurately and completely met.

Two-tier architectures work well in relatively homogeneous environments with fairly static business rules. This architecture is less suited for dispersed, heterogeneous environments with rapidly changing rules.

Since the bulk of application logic exists on the PC client, the two-tier architecture faces a number of potential version control and application re-distribution problems. A change in business rules would require a change to the client logic in each application in a corporation's portfolio which is affected by the change. Modified clients would have to be re-distributed through the network a potentially difficult task given the current lack of robust PC version control software and problems associated with upgrading PCs that are turned off or not "docked" to the network.

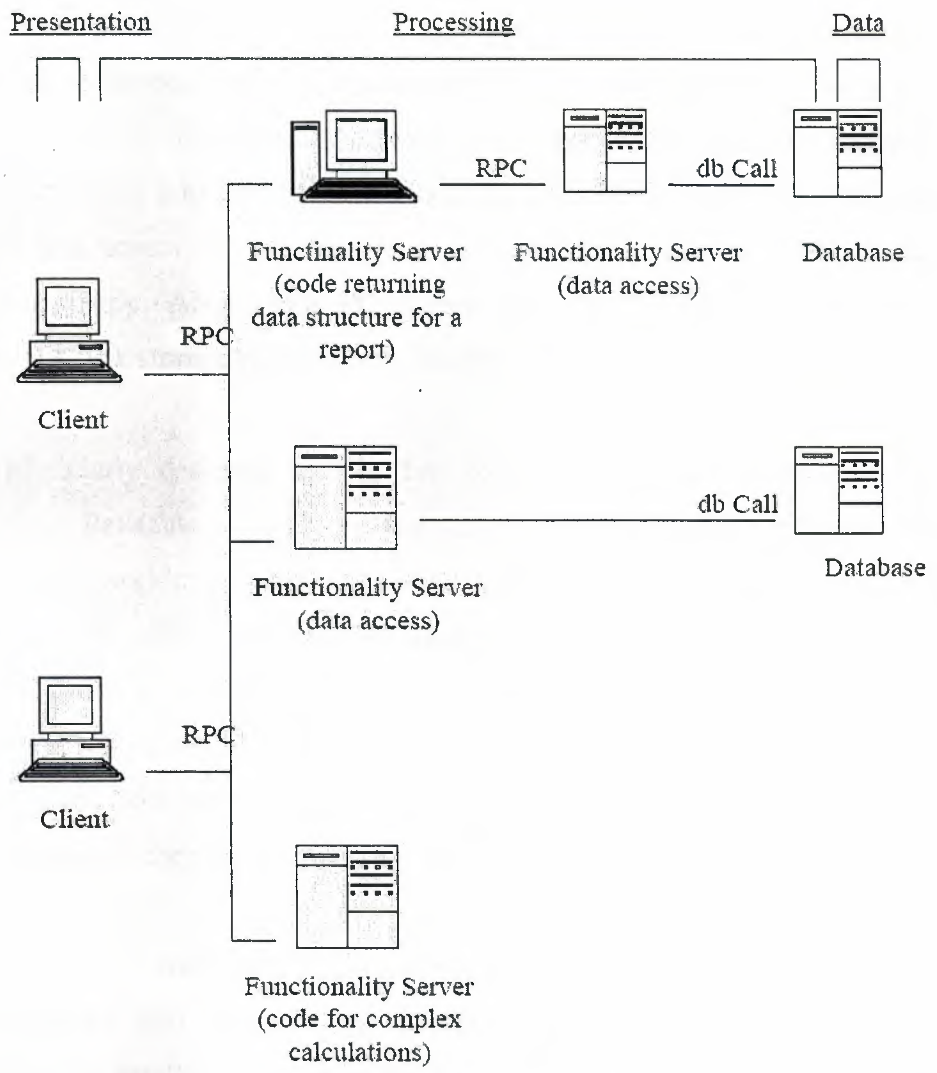
System security in the two-tier environment can be complicated since a user may require a separate password for each SQL server accessed. The proliferation of end-user query tools can also compromise database server security. The overwhelming majority of client/server applications developed today is designed without sophisticated middleware technologies which offer increased security. Instead, end-users are provided a password which gives them access to a database. In many cases this same password can be used to access the database with data-access tools available in most commercial PC spreadsheet and database packages. Using such a tool, a user may be able to access otherwise hidden fields or tables and possibly corrupt data.

Client tools and the SQL middleware used in two-tier environments are also highly proprietary and the PC tools market is extremely volatile. The client/server tools market seems to be changing at an increasingly unstable rate. In 1994, the leading client/server tool developer was purchased by a large database firm, raising concern about the manufacturer's ability to continue to work cooperatively with RDBMS vendors which compete with the parent company's products. The number two toolmaker lost millions and has been labeled as a takeover target. The tool which has received some of the brightest accolades in early 1995 is supplied by a firm also in the midst of severe financial difficulties and management transition. This kind of volatility raises questions about the long term viability of any proprietary tool an organization may commit to. All of this complicates implementation of two-tier systems migration from one proprietary technology to another would require a firm to scrap much of its investment in application code since none of this code is portable from one tool to the next.

2.7.2 Three tier or N-tier Architecture

The tree tier architecture (Figure 2.2) attempts to overcome some of the limitations of the two-tier scheme by separating presentation, processing, and data into separate, distinct software entities (tiers). The same types of tools can be used for presentation as were used in a two-tier environment; however these tools are now dedicated to handling just the presentation. When calculations or data access is required by the presentation client, a call is made to a middle tier functionality server. This tier can perform calculations or can make requests as a client to additional servers. The middle tier servers are typically coded in a highly portable, non-proprietary language such as C. Middle-tier functionality servers may be multi-threaded and can be accessed by multiple clients, even those from separate applications.

Although three-tier systems can be implemented using a variety of technologies, the calling mechanism from client to server in such as system is most typically the (remote procedure call or) RPC. Since the bulk of two-tier implementations involve SQL messaging and most three-tier systems utilize RPCs, it is reasonable to examine the merits of these respective request/response mechanisms in a discussion of architectures. RPC calls from presentation client to middle-tier server provide greater overall system flexibility than the SQL calls made by clients in the two-tier architecture. This is because in an RPC, the requesting client simply passes parameters needed for the request and specifies a data structure to accept returned values (if any). Unlike most two-tier implementations, the three-tier presentation client is not required to "speak" SQL. As such, the organization, names, or even the overall structure of the back-end data can be changed without requiring changes to PC-based presentation clients. Since SQL is no longer required, data can be organized hierarchically, relationally, or in object format. This added flexibility can allow a firm to access legacy data and simplifies the introduction of new database technologies.



Presentation
Processing
Data
RPC
db Call
Functinality Server (code returning data structure for a report)
Functionality Server (data access)
Database
RPC
Client
db Call
Functionality Server (data access)
Database
RPC
Client
Functionality Server (code for complex calculations)

Figure 2.2 Three Tier Architecture. Most of the logic processing is handled by functionality servers. Middle-tier code can be accessed and utilized by multiple clients

In addition to the openness stated above, several other advantages are presented by this architecture. Having separate software entities can allow for the parallel development of individual tiers by application specialists. It should be noted that the skill sets required to develop c/s applications differ significantly from those needed to develop mainframe-based character systems. As examples, user interface creation requires an appreciation for platform and corporate UI standards and database design requires a commitment to and understanding of the enterprise's data model. Having experts focus on each of these three layers can increase the overall quality of the final application.

The three-tier architecture also provides for more flexible resource allocation. Middle-tier functionality servers are highly portable and can be dynamically allocated and shifted as the needs of the organization change. Network traffic can potentially be reduced by having functionality servers strip data to the precise structure required before distributing it to individual clients at the LAN level. Multiple server requests and complex data access can emanate from the middle tier instead of the client, further decreasing traffic. Also, since PC clients are now dedicated to just presentation, memory and disk storage requirements for PCs will potentially be reduced.

Modularly designed middle tier code modules can be re-used by several applications. Reusable logic can reduce subsequent development efforts, minimize the maintenance workload, and decrease migration costs when switching client applications. In addition, implementation platforms for three tier systems such as OSF/DCE offer a variety of additional features to support distributed application development. These include integrated security, directory and naming services, server monitoring and boot capabilities for supporting dynamic fault-tolerance, and distributed time management for synchronizing systems across networks and separate time zones.

There are of course drawbacks associated with three-tier architecture. More code in more places also increases the likelihood that a system failure will effect an application so detailed planning with an emphasis on the reduction/elimination of critical paths is essential. Three tiers bring with it an increased need for network traffic management, server load balancing, and fault tolerance.

2.8 Advantages of Client Server Computing

There are many advantages to client/server architecture including that subsystems can be optimized for a particular set of applications. Systems can grow modularly, as different applications grow. Then more powerful subsystems can be installed without wasting resources on other applications. "Forklift upgrades," where an entire system is replaced, are theoretically kept to a minimum.

With most of the crucial applications and data residing on centralized machines, or clusters of machines, systems can be engineered to high standards of reliability and availability.

2.9 Summary

In conclusion, client/server architecture has become the dominant structure for corporate computing in both small and large organizations. It combines the best concepts of centralized, robust infrastructure with decentralized capability and control in other words, it gives both IT managers and end users what they want and need. If implemented properly, client/server architecture achieves the best balance between complexity, cost and ease of use, with excellent scalability and reliability.

CHAPTER THREE
INTERNET PROTOCOLS

3.1 Overview

The Internet protocols are the world's most popular open-system (nonproprietary) protocol suite because they can be used to communicate across any set of interconnected networks and are equally well suited for LAN and WAN communications. The Internet protocols consist of a suite of communication protocols, of which the two best known are the Transmission Control Protocol (TCP) and the Internet Protocol (IP). The Internet protocol suite not only includes lower-layer protocols (such as TCP and IP), but it also specifies common applications such as electronic mail, terminal emulation, and file transfer. This chapter provides a broad introduction to specifications that comprise the Internet protocols. Discussions include IP addressing and key upper-layer protocols used in the Internet. Specific routing protocols are addressed individually in Part 6, Routing Protocols.

Internet protocols were first developed in the mid-1970s, when the Defense Advanced Research Projects Agency (DARPA) became interested in establishing a packet-switched network that would facilitate communication between dissimilar computer systems at research institutions. With the goal of heterogeneous connectivity in mind, DARPA funded research by Stanford University and Bolt, Beranek, and Newman (BBN). The result of this development effort was the Internet protocol suite, completed in the late 1970s.

TCP/IP later was included with Berkeley Software Distribution (BSD) UNIX and has since become the foundation on which the Internet and the World Wide Web (WWW) are based.

Documentation of the Internet protocols (including new or revised protocols) and policies are specified in technical reports called Request For Comments (RFCs), which are published and then reviewed and analyzed by the Internet community. Protocol refinements are published in the new RFCs. To illustrate the scope of the Internet protocols, Figure 3.1 maps many of the protocols of the Internet protocol suite and their corresponding OSI layers. This chapter addresses the basic elements and operations of these and other key Internet protocols.

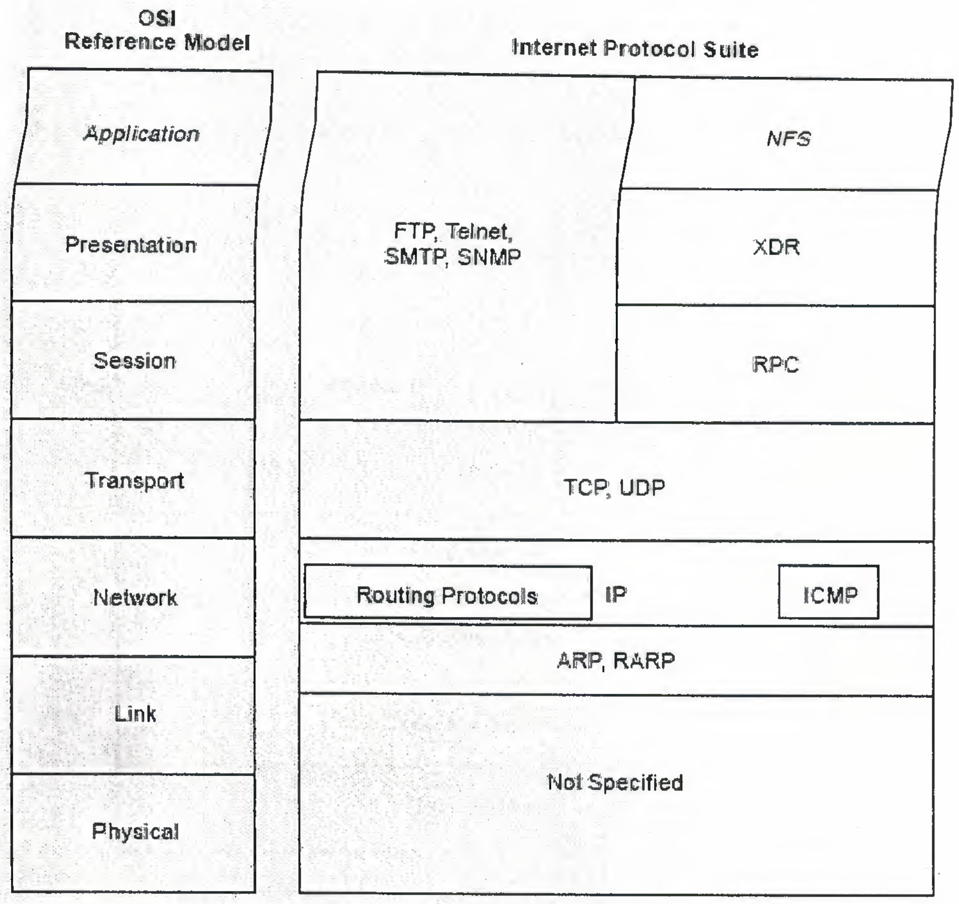


Figure 3.1 Internet protocols span the complete range of OSI model layers.

3.2 Internet Protocol (IP)

The Internet Protocol (IP) is a network-layer (Layer 3) protocol that contains addressing information and some control information that enables packets to be routed. IP is documented in RFC 791 and is the primary network-layer protocol in the Internet protocol suite. Along with the Transmission Control Protocol (TCP), IP represents the heart of the Internet protocols.

IP has two primary responsibilities:

- Providing connectionless, best-effort delivery of datagrams through an internet work.
- Providing fragmentation and reassembly of datagrams to support data links with different maximum-transmission unit (MTU) sizes.

3.2.1 IP Packet Format

An IP packet contains several types of information, as illustrated in Figure 3.2.

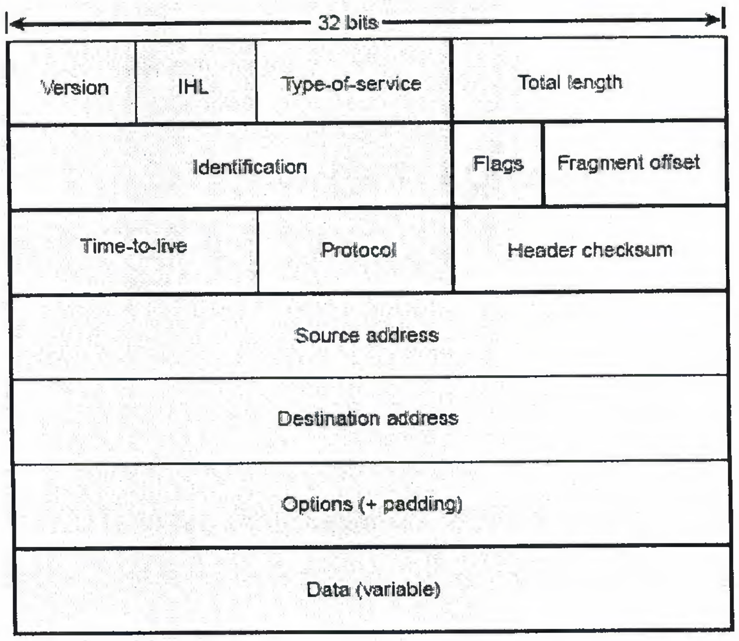


Figure 3.2 Fourteen fields comprise an IP packet.

The following discussion describes the IP packet fields illustrated in Figure 3.2:

• *Version*—indicates the version of IP currently used.

• *IP Header Length (*IHL)—Indicates the datagram header length in 32-bit words.

• *Type-of-Service*—Specifies how an upper-layer protocol would like a current datagram to be handled, and assigns datagrams various levels of importance.

• *Total Length*—specifies the length, in bytes, of the entire IP packet, including the data and header.

• *Identification*—contains an integer that identifies the current datagram. This field is used to help piece together datagram fragments.

• *Flags*—consist of a 3-bit field of which the two low-order (least-significant) bits control fragmentation. The low-order bit specifies whether the packet can be fragmented. The middle bit specifies whether the packet is the last fragment in a series of fragmented packets. The third or high-order bit is not used.

• *Fragment Offset*—indicates the position of the fragment's data relative to the beginning of the data in the original datagram, which allows the destination IP process to properly reconstruct the original datagram.

• *Time-to-Live*—maintains a counter that gradually decrements down to zero, at which point the datagram is discarded. This keeps packets from looping endlessly.

• *Protocol*—Indicates which upper-layer protocol receives incoming packets after IP processing is complete.

• *Header Checksum*—helps ensure IP header integrity.

• *Source Address*—specifies the sending node.

• *Destination Address*—specifies the receiving node.

• *Options*—Allows IP to support various options, such as security.

• *Data*—Contains upper-layer information.

3.2.2 IP Addressing

As with any other network-layer protocol, the IP addressing scheme is integral to the process of routing IP datagrams through an internetwork. Each IP address has specific components and follows a basic format. These IP addresses can be subdivided and used to create addresses for subnetworks, as discussed in more detail later in this chapter.

Each host on a TCP/IP network is assigned a unique 32-bit logical address that is divided into two main parts: the network number and the host number. The network number identifies a network and must be assigned by the Internet Network Information Center (InterNIC) if the network is to be part of the Internet. An Internet Service Provider (ISP) can obtain blocks of network addresses from the InterNIC and can itself assign address space as necessary. The host number identifies a host on a network and is assigned by the local network administrator.

3.2.3 IP Address Format

The 32-bit IP address is grouped eight bits at a time, separated by dots, and represented in decimal format (known as *dotted decimal notation*). Each bit in the octet has a binary weight (128, 64, 32, 16, 8, 4, 2, 1). The minimum value for an octet is 0, and the maximum value for an octet is 255. Figure 3.3 illustrates the basic format of an IP address.

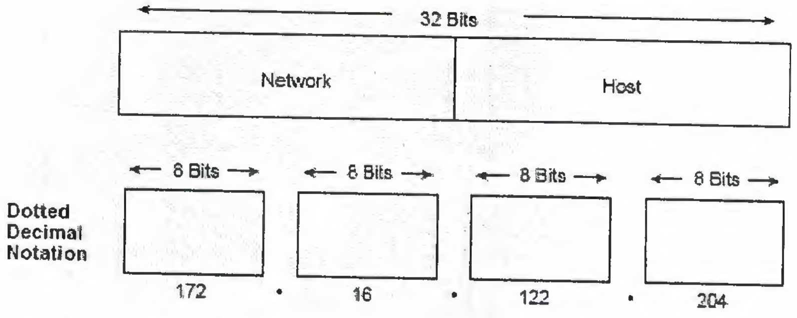


Figure 3.3 an IP address consists of 32 bits, grouped into four octets.

3.2.4 IP Address Classes

IP addressing supports five different address classes: A, B, C, D, and E. only classes A, B, and C are available for commercial use. The left-most (high-order) bits indicate the network class. Table 3.1 provides reference information about the five IP address classes.

Table 3.1 Reference Information about the Five IP Address Classes

IP Address Class	Format	Purpose	High-Order Bit(s)	Address Range	No. Bits Network/Host	Max. Hosts
A	N.H.H.H[1]	Few large organizations	0	1.0.0.0 to 126.0.0.0	7/24	16,777, 214[2] ($2^{24}-2$)
B	N.N.H.H	Medium-size organizations	1, 0	128.1.0.0 to 191.254.0.0	14/16	65, 543 ($2^{16}-2$)
C	N.N.N.H	Relatively small organizations	1, 1, 0	192.0.1.0 to 223.255.254.0	22/8	245 ($2^{8}-2$)
D	N/A	Multicast groups (RFC 1112)	1, 1, 1, 0	224.0.0.0 to 239.255.255.255	N/A (not for commercial use)	N/A
E	N/A	Experimental	1, 1, 1, 1	240.0.0.0 to 254.255.255.255	N/A	N/A

1 N = Network number, H = Host number.
2 One address is reserved for the broadcast address, and one address is reserved for the network.

Figure 3.4 illustrates the format of the commercial IP address classes. (Note the high-order bits in each class.)

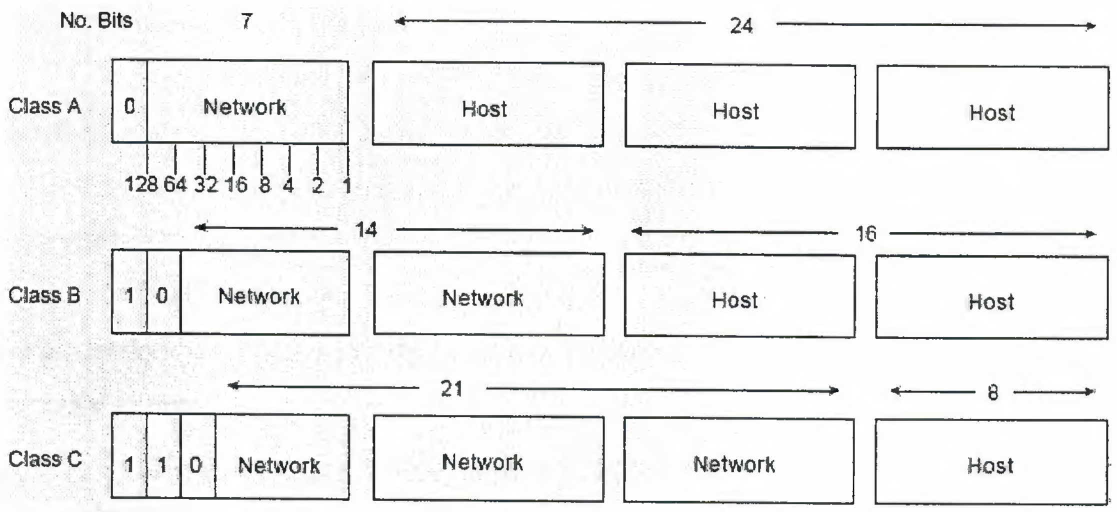


Figure 3.4 IP address formats A, B, and C are available for commercial use.

The class of address can be determined easily by examining the first octet of the address and mapping that value to a class range in the following table. In an IP address of 172.31.1.2, for example, the first octet is 172. Because 172 fall between 128 and 191, 172.31.1.2 is a Class B address. Figure 3.5 summarizes the range of possible values for the first octet of each address class.

Address Class	First Octet in Decimal	High-Order Bits
Class A	1 Ð 126	0
Class B	128 Ð 191	10
Class C	192 Ð 223	110
Class D	224 Ð 239	1110
Class E	240 Ð 254	1111

Figure 3.5 a range of possible values exists for the first octet of each address class.

3.2.5 IP Subnet Addressing

IP networks can be divided into smaller networks called sub networks (or subnets). Sub netting provides the network administrator with several benefits, including extra flexibility, more efficient use of network addresses, and the capability to contain broadcast traffic (a broadcast will not cross a router).

Subnets are under local administration. As such, the outside world sees an organization as a single network and has no detailed knowledge of the organization's internal structure.

A given network address can be broken up into many sub networks. For example, 172.16.1.0, 172.16.2.0, 172.16.3.0, and 172.16.4.0 are all subnets within network 171.16.0.0. (All 0s in the host portion of an address specifies the entire network.)

IP Subnet Mask A subnet address is created by "borrowing" bits from the host field and designating them as the subnet field. The number of borrowed bits varies and is specified by the subnet mask. Figure 3.6 shows how bits are borrowed from the host address field to create the subnet address field.

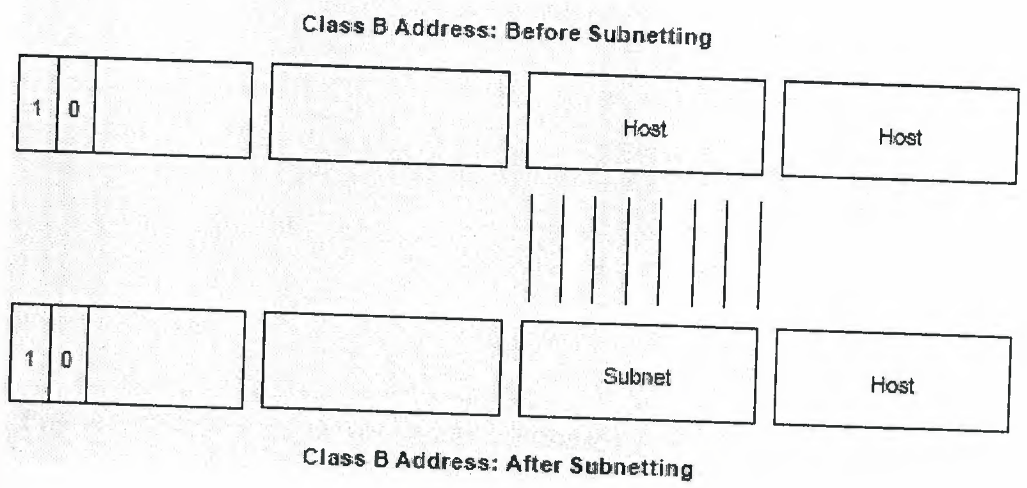

Class B Address: Before Subnetting
1
0
Host
Host
1
0
Subnet
Host
Class B Address: After Subnetting

Figure 3.6 Bits are borrowed from the host address field to create the subnet address field.

Subnet masks use the same format and representation technique as IP addresses. The subnet mask, however, has binary 1s in all bits specifying the network and sub network fields, and binary 0s in all bits specifying the host field. Figure 3.7 illustrates a sample subnet mask.

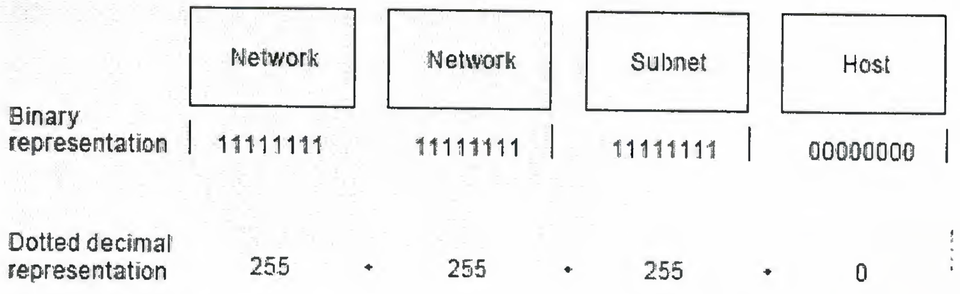


Figure 3.7 a sample subnet mask consists of all binary 1s and 0s.

Subnet mask bits should come from the high-order (left-most) bits of the host field, as Figure 3.8 illustrates. Details of Class B and C subnet mask types follow. Class A addresses are not discussed in this chapter because they generally are sub netted on an 8-bit boundary.

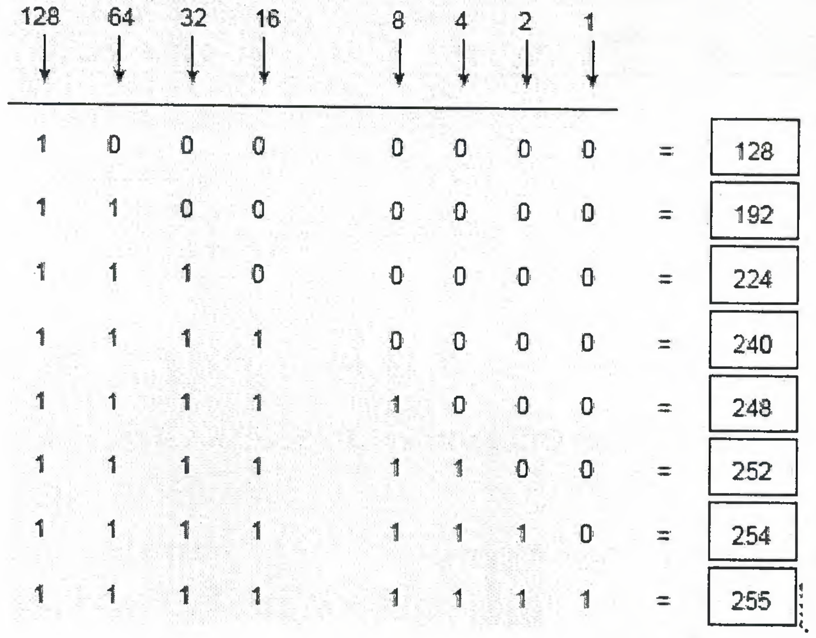


Figure 3.8 Subnet mask bits come from the high-order bits of the host field.

The default subnet mask for a Class B address that has no sub netting is 255.255.0.0, while the subnet mask for a Class B address 171.16.0.0 that specifies eight bits of sub netting is 255.255.255.0. The reason for this is that eight bits of sub netting or 28 – 2 (1 for the network address and 1 for the broadcast address) = 254 subnets possible, with 28 – 2 = 254 hosts per subnet.

The subnet mask for a Class C address 192.168.2.0 that specifies five bits of sub netting is 255.255.255.248.With five bits available for sub netting, 25 – 2 = 30 subnets possible, with 23 – 2 = 6 hosts per subnet.

The reference charts shown in table 3.2 and table 3.3 can be used when planning Class B and C networks to determine the required number of subnets and hosts, and the appropriate subnet mask..

Table 3.2 Class B Sub netting Reference Chart

Number of Bits	Subnet Mask	Number of Subnets	Number of Hosts
2	255.255.192.0	2	16382
3	255.255.224.0	6	8190
4	255.255.240.0	14	4094
5	255.255.248.0	30	2046
6	255.255.252.0	62	1022
7	255.255.254.0	126	510
8	255.255.255.0	254	254
9	255.255.255.128	510	126
10	255.255.255.192	1022	62
11	255.255.255.224	2046	30
12	255.255.255.240	4094	14

Table 3.3 Class C Sub netting Reference Chart

Number of Bits	Subnet Mask	Number of Subnets	Number of Hosts
2	255.255.255.192	2	62
3	255.255.255.224	6	30
4	255.255.255.240	14	14
5	255.255.255.248	30	6
6	255.255.255.252	62	2

3.3 How Subnet Masks are used to determine the Network Number

The router performs a set process to determine the network (or more specifically, the subnetwork) address. First, the router extracts the IP destination address from the incoming packet and retrieves the internal subnet mask. It then performs a *logical AND* operation to obtain the network number.

This causes the host portion of the IP destination address to be removed, while the destination network number remains. The router then looks up the destination network number and matches it with an outgoing interface. Finally, it forwards the frame to the destination IP address. Specifics regarding the logical AND operation are discussed in the following section.

3.3.1 Logical AND Operation

Three basic rules govern logically "ANDing" two binary numbers. First, 1 "ANDed" with 1 yields 1. Second, 1 "ANDed" with 0 yields 0. Finally, 0 "ANDed" with 0 yields 0. The truth table provided in table 3.4 illustrates the rules for logical AND operations.

Table 3.4 Rules for Logical AND Operations

Input	Input	Output
1	1	1
1	0	0
0	1	0
0	0	0

Two simple guidelines exist for remembering logical AND operations: Logically "ANDing" a 1 with a 1 yields the original value, and logically "ANDing" a 0 with any number yields 0.

Figure 3.9 illustrates that when a logical AND of the destination IP address and the subnet mask is performed, the subnetwork number remains, which the router uses to forward the packet.

		Network	Subnet	Host
Destination IP Address	171.16.1.2		00000001	00000010
Subnet Mask	255.255.255.0		11111111	00000000
			00000001	00000000
			1	0

Figure 3.9 Applying a logical AND the destination IP address and the subnet mask Produces the sub network number.

3.4 Address Resolution Protocol (ARP)

For two machines on a given network to communicate, they must know the other machine's physical (or MAC) addresses. By broadcasting Address Resolution Protocols (ARPs), a host can dynamically discover the MAC-layer address corresponding to a particular IP network-layer address.

After receiving aMAC-layer address, IP devices create an ARP cache to store the recently acquired IP-to-MAC address mapping, thus avoiding having to broadcast ARPS when they want to re-contact a device. If the device does not respond within a specified time frame, the cache entry is flushed.

In addition to the Reverse Address Resolution Protocol (RARP) is used to map MAC-layer addresses to IP addresses. RARP, which is the logical inverse of ARP, might be used by diskless workstations that do not know their IP addresses when they boot. RARP relies on the presence of a RARP server with table entries of MAC-layer-to-IP address mappings.

3.5 Internet Routing

Internet routing devices traditionally have been called gateways. In today's terminology, however, the term gateway refers specifically to a device that performs application-layer protocol translation between devices. Interior gateways refer to devices that perform these protocol functions between machines or networks under the same administrative control or authority, such as a corporation's internal network. These are known as autonomous systems. Exterior gateways perform protocol functions between independent networks.

Routers within the Internet are organized hierarchically. Routers used for information exchange within autonomous systems are called interior routers, which use a variety of Interior Gateway Protocols (IGPs) to accomplish this purpose. The Routing Information Protocol (RIP) is an example of an IGP.

Routers that move information between autonomous systems are called exterior routers. These routers use an exterior gateway protocol to exchange information between autonomous systems. The Border Gateway Protocol (BGP) is an example of an exterior gateway protocol.

IP Routing IP routing protocols are dynamic. Dynamic routing calls for routes to be calculated automatically at regular intervals by software in routing devices. This contrasts with static routing, where routers are established by the network administrator and do not change until the network administrator changes them.

An IP routing table, which consists of destination address/next hop pairs, is used to enable dynamic routing. An entry in this table, for example, would be interpreted as follows: to get to network 172.31.0.0, send the packet out Ethernet interface 0 (E0).

IP routing specifies that IP datagrams travel through inter-networks one hop at a time. The entire route is not known at the onset of the journey, however. Instead, at each stop, the next destination is calculated by matching the destination address within the datagram with an entry in the current node's routing table.

Each node's involvement in the routing process is limited to forwarding packets based on internal information. The nodes do not monitor whether the packets get to their final destination, nor does IP provide for error reporting back to the source when routing anomalies occur. This task is left to another Internet protocol, the Internet Control-Message Protocol (ICMP), which is discussed in the following section.

3.6 Internet Control Message Protocol (ICMP)

The *Internet Control Message Protocol (ICMP)* is a network-layer Internet protocol that provides message packets to report errors and other information regarding IP packet processing back to the source. ICMP is documented in RFC 792.

3.6.1 ICMP Messages

ICMPs generate several kinds of useful messages, including Destination Unreachable, Echo Request and Reply, Redirect, Time Exceeded, and Router Advertisement and Router Solicitation. If an ICMP message cannot be delivered, no second one is generated. This is to avoid an endless flood of ICMP messages.

When an ICMP destination-unreachable message is sent by a router, it means that the router is unable to send the package to its final destination. The router then discards the original packet. Two reasons exist for why a destination might be unreachable. Most commonly, the source host has specified a nonexistent address. Less frequently, the router does not have a route to the destination.

Destination-unreachable messages include four basic types: network unreachable, host unreachable, protocol unreachable, and port unreachable. *Network-unreachable messages* usually mean that a failure has occurred in the routing or addressing of a packet. *Host-unreachable messages* usually indicate delivery failure, such as a wrong subnet mask. *Protocol-unreachable messages* generally mean that the destination does not support the upper-layer protocol specified in the packet.
Port-unreachable messages imply that the TCP socket or port is not available.

An ICMP echo-request message, which is generated by the ping command, is sent by any host to test node reachability across an internetwork. The ICMP echo-reply message indicates that the node can be successfully reached.

An ICMP Redirect message is sent by the router to the source host to stimulate more efficient routing. The router still forwards the original packet to the destination. ICMP redirects allow host routing tables to remain small because it is necessary to know the address of only one router, even if that router does not provide the best path. Even after receiving an ICMP Redirect message, some devices might continue using the less-efficient route.

An ICMP Time-exceeded message is sent by the router if an IP packet's Time-to-Live field (expressed in hops or seconds) reaches zero. The Time-to-Live field prevents packets from continuously circulating the internetwork if the internetwork contains a routing loop. The router then discards the original packet.

3.6.2 ICMP Router-Discovery Protocol (IDRP)

IDRP uses Router-Advertisement and Router-Solicitation messages to discover the addresses of routers on directly attached subnets. Each router periodically multicasts Router-Advertisement messages from each of its interfaces. Hosts then discover addresses of routers on directly attached subnets by listening for these messages. Hosts can use Router-Solicitation messages to request immediate advertisements rather than waiting for unsolicited messages.

IRDP offers several advantages over other methods of discovering addresses of neighboring routers.

Primarily, it does not require hosts to recognize routing protocols, nor does it require manual configuration by an administrator.

Router-Advertisement messages enable hosts to discover the existence of neighboring routers, but not which router is best to reach a particular destination. If a host uses a poor first-hop router to reach a particular destination, it receives a Redirect message identifying a better choice.

3.7 Transmission Control Protocol (TCP)

The TCP provides reliable transmission of data in an IP environment. TCP corresponds to the transport layer (Layer 4) of the OSI reference model. Among the services TCP provides are stream data transfer, reliability, efficient flow control, full-duplex operation, and multiplexing.

With stream data transfer, TCP delivers an unstructured stream of bytes identified by sequence numbers. This service benefits applications because they do not have to chop data into blocks before handing it off to TCP. Instead, TCP groups bytes into segments and passes them to IP for delivery.

TCP offers reliability by providing connection-oriented, end-to-end reliable packet delivery through an internetwork. It does this by sequencing bytes with a forwarding acknowledgment number that indicates to the destination the next byte the source expects to receive. Bytes not acknowledged within a specified time period are retransmitted. The reliability mechanism of TCP allows devices to deal with lost, delayed, duplicate, or misread packets. A time-out mechanism allows devices to detect lost packets and request retransmission.

TCP offers efficient flow control, which means that, when sending acknowledgments back to the source, the receiving TCP process indicates the highest sequence number it can receive without overflowing its internal buffers.
Full-duplex operation means that TCP processes can both send and receive at the same time.

Finally, TCP's multiplexing means that numerous simultaneous upper-layer conversations can be multiplexed over a single connection.

3.7.1 TCP Connection Establishment

To use reliable transport services, TCP hosts must establish a connection-oriented session with one another. Connection establishment is performed by using a "three-way handshake" mechanism.

A three-way handshake synchronizes both ends of a connection by allowing both sides to agree upon initial sequence numbers. This mechanism also guarantees that both sides are ready to transmit data and know that the other side is ready to transmit as well. This is necessary so that packets are not transmitted or retransmitted during session establishment or after session termination.

Each host randomly chooses a sequence number used to track bytes within the stream it is sending and receiving. Then, the three-way handshake proceeds in the following manner:
The first host (Host A) initiates a connection by sending a packet with the initial sequence number (X) and SYN bit set to indicate a connection request. The second host (Host B) receives the SYN, records the sequence number X, and replies by acknowledging the SYN (with an ACK = X + 1).

Host B includes its own initial sequence number (SEQ = Y). An ACK = 20 means the host has received bytes 0 through 19 and expects byte 20 next. This technique is called *forward acknowledgment.* Host A then acknowledges all bytes Host B sent with a forward acknowledgment indicating the next byte Host A expects to receive (ACK = Y + 1). Data transfer then can begin.

3.7.2 TCP Sliding Window

A *TCP sliding window* provides more efficient use of network bandwidth than PAR because it enables hosts to send multiple bytes or packets before waiting for an acknowledgment.

In TCP, the receiver specifies the current window size in every packet. Because TCP provides a byte-stream connection, window sizes are expressed in bytes. This means that a window is the number of data bytes that the sender is allowed to send before waiting for an acknowledgment. Initial window sizes are indicated at connection setup, but might vary throughout the data transfer to provide flow control. A window size of zero, for instance, means "Send no data."

In a TCP sliding-window operation, for example, the sender might have a sequence of bytes to send (numbered 1 to 10) to a receiver who has a window size of five. The sender then would place a window around the first five bytes and transmit them together. It would then wait for an acknowledgment.

The receiver would respond with an ACK = 6, indicating that it has received bytes 1 to 5 and is expecting byte 6 next. In the same packet, the receiver would indicate that its window size is 5. The sender then would move the sliding window five bytes to the right and transmit bytes 6 to 10. The receiver would respond with an ACK = 11, indicating that it is expecting sequenced byte 11 next. In this packet, the receiver might indicate that its window size is 0 (because, for example, its internal buffers are full). At this point, the sender cannot send any more bytes until the receiver sends another packet with a window size greater than 0.

3.7.3 TCP Packet Format

Figure 3.10 illustrates the fields and overall format of a TCP packet.

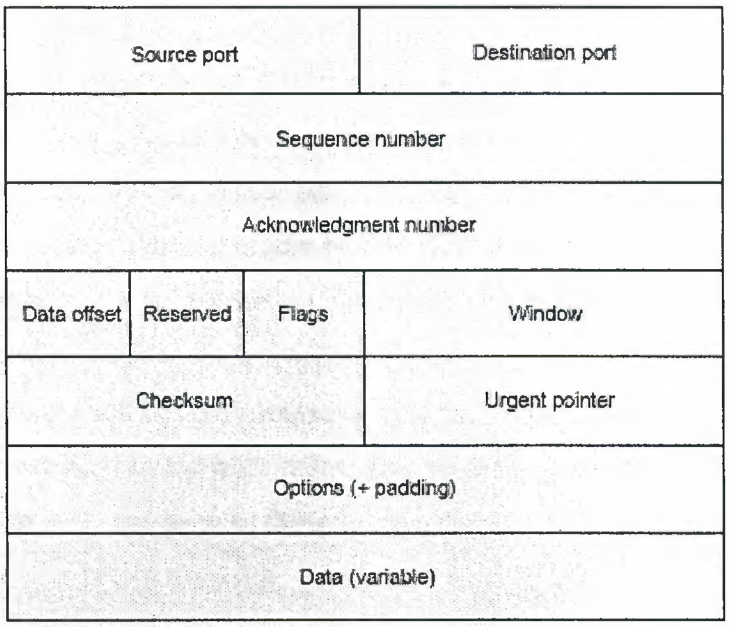


Figure 3.10 Twelve fields comprise a TCP packet.

3.7.4 TCP Packet Field Descriptions

The following descriptions summarize the TCP packet fields illustrated in Figure 3.10:

• *Source Port* and *Destination Port*—Identifies points at which upper-layer source and destination processes receive TCP services.

• *Sequence Number*—usually specifies the number assigned to the first byte of data in the current message. In the connection-establishment phase, this field also can be used to identify an initial sequence number to be used in an upcoming transmission.

• *Acknowledgment Number*—contains the sequence number of the next byte of data the sender of the packet expects to receive.

• *Data Offset*—indicates the number of 32-bit words in the TCP header.

• *Reserved*—Remains reserved for future use.

• *Flags*—carry a variety of control information, including the SYN and ACK bits used for connection establishment, and the FIN bit used for connection termination.

• *Window*—Specifies the size of the sender's receive window (that is, the buffer space available for incoming data).

• *Checksum*—indicates whether the header was damaged in transit.

• *Urgent Pointer*—Points to the first urgent data byte in the packet.

• *Options*—Specifies various TCP options.

• *Data*—Contains upper-layer information.

3.8 Positive Acknowledgment and Retransmission (PAR)

A simple transport protocol might implement a reliability-and-flow-control technique where the source sends one packet, starts a timer, and waits for an acknowledgment before sending a new packet. If the acknowledgment is not received before the timer expires, the source retransmits the packet. Such a technique is called *positive acknowledgment and retransmission* (PAR).

By assigning each packet a sequence number, PAR enables hosts to track lost or duplicate packets caused by network delays that result in premature retransmission. The sequence numbers are sent back in the acknowledgments so that the acknowledgments can be tracked.

PAR is an inefficient use of bandwidth, however, because a host must wait for an acknowledgment before sending a new packet, and only one packet can be sent at a time.

3.9 User Datagram Protocol (UDP)

The User Datagram Protocol (UDP) is a connectionless transport-layer protocol (Layer 4) that belongs to the Internet protocol family. UDP is basically an interface between IP and upper-layer processes. UDP protocol ports distinguish multiple applications running on a single device from one another.

Unlike the TCP, UDP adds no reliability, flow-control, or error-recovery functions to IP. Because of UDP's simplicity, UDP headers contain fewer bytes and consume less network overhead than TCP.

UDP is useful in situations where the reliability mechanisms of TCP are not necessary, such as in cases where a higher-layer protocol might provide error and flow control.

UDP is the transport protocol for several well-known application-layer protocols, including Network File System (NFS), Simple Network Management Protocol (SNMP), Domain Name System (DNS), and Trivial File Transfer Protocol (TFTP).

The UDP packet format contains four fields, as shown in Figure 3.11. These include source and destination ports, length, and checksum fields.

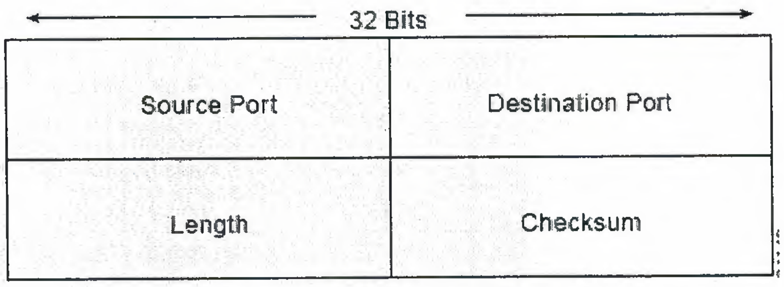


Figure 3.11 A UDP packet consists of four fields.

Source and destination ports contain the 16-bit UDP protocol port numbers used to de-multiplex data-grams for receiving application-layer processes. A length field specifies the length of the UDP header and data. Checksum provides an (optional) integrity check on the UDP header and data.

3.10 Internet Protocols Application-Layer Protocols

The Internet protocol suite includes many application-layer protocols that represent a wide variety of applications, including the following:

- *File Transfer Protocol (FTP)*—Moves files between devices
- *Simple Network-Management Protocol (SNMP)*—primarily reports anomalous network conditions and sets network threshold values
- *Telnet*—serves as a terminal emulation protocol
- *X Windows*—Serves as a distributed windowing and graphics system used for communication between X terminals and UNIX workstations
- *Network File System (NFS), External Data Representation (XDR), and Remote Procedure Call (RPC)*—Work together to enable transparent access to remote network resources
- *Simple Mail Transfer Protocol (SMTP)*—Provides electronic mail services
- *Domain Name System (DNS)*—translates the names of network nodes into network addresses

Table 3.5 lists these higher-layer protocols and the applications that they support.

Table 3.5 Higher-Layer Protocols and Their Applications

Application	Protocols
File transfer	FTP
Terminal emulation	Telnet
Electronic mail	SMTP
Network management	SNMP
Distributed file services	NFS, XDR, RPC, X Windows

CHAPTER FOUR
DESIGN OF CONTROL DEVICE THROUGH PARALLEL PORT

4.1 Overview

In the computer world, a *port* is a set of signal lines that the microprocessor, or CPU, uses to exchange data with other components.

Typical uses for ports are communicating with printers, modems, keyboards, and displays, or just about any component or device except system memory. Most computer ports are digital, where each signal, or bit, is 0 or 1. A parallel port transfers multiple bits at once, while a serial port transfers a bit at a time (though it may transfer in both directions at once).

Along with the RS-232 serial port, the parallel port is a workhorse of PC communications. On newer PCs, you may find other ports such as SCSI, USB, and IrDA, but the parallel port remains popular because it's capable, flexible, and every PC has one.

The term *PC-compatible,* or *PC* for short, refers to the IBM PC and any of the many, many personal computers derived from it. From another angle, a PC is any computer that can run Microsoft's MS-DOS operating system and whose expansion bus is compatible with the ISA bus in the original IBM PC. The category includes the PC, XT, AT, PS/2, and most computers with 80x86, Pentium, and compatible CPUs. It does not include the Macintosh, Amiga, or IBM mainframes, though these and other computer types may have ports that are similar to the parallel port on the PC.

The original PC's parallel port had eight outputs, five inputs, and four bidirectional lines. These are enough for communicating with many types of peripherals.

On many newer PCs, the eight outputs can also serve as inputs, for faster communications with scanners, drives, and other devices that send data to the PC.
The parallel port was designed as a printer port, and many of the original names for the port's signals (*Paper End, Auto Linefeed*) reflect that use. But these days, you can find all kinds of things besides printers connected to the port. The term *peripheral* or *peripheral device* is a catch-all category that includes printers, scanners, modems, and other devices that connect to a PC.

4.2 Port Types

As the design of the PC evolved, several manufacturers introduced improved versions of the parallel port. The new port types are compatible with the original design, but add new abilities, mainly for increased speed.

Speed is important because as computers and peripherals have gotten faster, the jobs they do have become more complicated, and the amount of information they need to exchange has increased. The original parallel port was plenty fast enough for sending bytes representing ASCII text characters to a dot-matrix or daisy-wheel printer. But modern printers need to receive much more information to print a page with multiple fonts and detailed graphics, often in color. The faster the computer can transmit the information, the faster the printer can begin processing and printing the result.

A fast interface also makes it feasible to use portable, external versions of peripherals that you would otherwise have to install inside the computer. A parallel-port tape or disk drive is easy to move from system to system, and for occasional use, such as making back-ups, you can use one unit for several systems. Because a backup may involve copying hundreds of Megabytes, the interface has to be fast to be worthwhile.

4.2.1 Original (SPP)

The parallel port in the original IBM PC, and any port that emulates the original port's design, is sometimes called the *SPP*, for standard parallel port, even though the original port had no written standard beyond the schematic diagrams and documentation for the IBM PC. Other names used are *AT-type* or *ISA-compatible.*

The port in the original PC was based on an existing Centronics printer interface. However, the PC introduced a few differences, which other systems have continued.

SPPs can transfer eight bits at once to a peripheral, using a protocol similar to that used by the original Centronics interface. The SPP doesn't have a byte-wide input port, but for PC-to-peripheral transfers, SPPs can use a Nibble mode that transfers each byte 4 bits at a time. Nibble mode is slow, but has become popular as a way to use the parallel port for input.

4.2.2 PS/2-type (Simple Bidirectional)

An early improvement to the parallel port was the bidirectional data port introduced on IBM's model PS/2. The bidirectional port enables a peripheral to transfer eight bits at once to a PC. The term *PS/2-type* has come to refer to any parallel port that has a bidirectional data port but doesn't support the EPP or ECP modes described below. Byte mode is an 8-bit data-transfer protocol that PS/2-type ports can use to transfer data from the peripheral to the PC.

4.2.3 EPP

The EPP (enhanced parallel port) was originally developed by chip maker Intel, PC manufacturer Zenith, and Xircom, a maker of parallel-port networking products.

As on the PS/2-type port, the data lines are bidirectional. An EPP can read or write a byte of data in one cycle of the ISA expansion bus, or about 1 microsecond, including handshaking, compared to four cycles for an SPP or PS/2-type port. An EPP can switch directions quickly, so it's very efficient when used with disk and tape drives and other devices that transfer data in both directions. An EPP can also emulate an SPP, and some EPPs can emulate a PS/2-type port.

4.2.4 ECP

The ECP (extended capabilities port) was first proposed by Hewlett Packard and Microsoft. Like the EPP, the ECP is bidirectional and can transfer data at ISA-bus speeds. ECPs have buffers and support for DMA (direct memory access) transfers and data compression. ECP transfers are useful for printers, scanners, and other peripherals that transfer large blocks of data. An ECP can also emulate an SPP or PS/2-type port, and many ECPs can emulate an EPP as well.

4.2.5 Multi-mode Ports

Many newer ports are multi-mode ports that can emulate some or all of the above types. They often include configuration options that can make all of the port types available, or allow certain modes while locking out the others.

4.3 System Resources

The parallel port uses a variety of the computer's resources. Every port uses a range of addresses, though the number and location of addresses varies. Many ports have an assigned IRQ (interrupt request) level, and ECPs may have an assigned DMA channel. The resources assigned to a port can't conflict with those used by other system components, including other parallel ports.

4.4 Addressing

The standard parallel port uses three contiguous addresses, usually in one of these ranges:

3BCH, 3BDH, 3BEH

378H, 379H, 37AH

278H, 279H, 27AH

The first address in the range is the port's base address, also called the Data register or just the port address. The second address is the port's Status register, and the third is the Control register. EPPs and ECPs reserve additional addresses for each port. An EPP adds five registers at *base address + 3* through *base address + 7*, and an ECP adds three registers at *base address + 400h* through *base address + 402h.* For a base address of 378h, the EPP registers are at 37Bh through 37Fh, and the ECP registers are at 778h through 77Fh.

On early PCs, the parallel port had a base address of 3BCh. On newer systems, the parallel port is most often at 378h. But all three addresses are reserved for parallel ports, and if the port's hardware allows it, you can configure a port at any of the addresses.

IBM's Type 3 PS/2 port also had three additional registers, at *base address +3* through *base address + 5*, and allowed a base address of 1278h or 1378h. Most often, DOS and Windows refer to the first port in numerical order as *LPT1*, the second, *LPT2*, and the third, *LPT3.* So on bootup, LPT1 is most often at 378h, but it may be at any of the three addresses. LPT2, if it exists, may be at 378h or 278h, and LPT3 can only be at 278h. Various configuration techniques can change these assignments, however, so not all systems will follow this convention.

LPT stands for *line printer*, reflecting the port's original intended use. If your port's hardware allows it, you can add a port at any unused port address in the system.

Not all software will recognize these non-standard ports as LPT ports, but you can access them with software that writes directly to the port registers.

4.5 Interrupts

Most parallel ports are capable of detecting interrupt signals from a peripheral. The peripheral may use an interrupt to announce that it's ready to receive a byte, or that it has a byte to send. To use interrupts, a parallel port must have an assigned interrupt-request level (IRQ).

Conventionally, LPT1 uses IRQ7 and LPT2 uses IRQ5. But IRQ5 is used by many sound cards, and because free IRQ levels can be scarce on a system, even IRQ7 may be reserved by another device. Some ports allow choosing other IRQ levels besides these two.

Many printer drivers and many other applications and drivers that access the parallel port don't require parallel-port interrupts. If you select no IRQ level for a port, the port will still work in most cases, though sometimes not as efficiently, and you can use the IRQ level for something else.

4.6 DMA Channels

ECPs can use direct memory access (DMA) for data transfers at the parallel port. During the DMA transfers, the CPU is free to do other things, so DMA transfers can result in faster performance overall. In order to use DMA, the port must have an assigned DMA channel, in the range 0 to 3.

4.7 Port Options

There is no standard method for configuring a port. Some ports, especially older ones, use jumper blocks or switches to select different options. Others allow configuring in software, using a utility provided on disk. A port on a system motherboard may have configuration options in the system setup screens (the CMOS setup) that you can access on bootup. On ports that meet Microsoft's Plug and Play standard, Windows 95 can automatically assign an available port address and IRQ level to a port.

Some ports allow a choice of just one or two of the three conventional base addresses. A few allow you to choose any uncommitted address, including nonstandard ones. On some boards, the jumpers or switches are labeled, which is extremely handy when you don't have other documentation (or can't find it). If your port supports ECP transfers, assign it an IRQ level and DMA channel if possible. Most ECP drivers do use these, and if they're not available, the driver will revert to a slower mode.

4.8 Port Hardware

The parallel port's hardware includes the back-panel connector and the circuits and cabling between the connector and the system's expansion bus. The PC's microprocessor uses the expansion bus's data, address, and control lines to transfer information between the parallel port and the CPU, memory, and other system components.

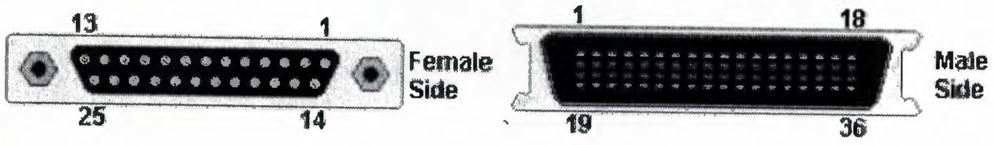


Figure 4.1 Parallel Port.

4.9 Connectors

The PC's back panel has the connector for plugging in a cable to a printer or other device with a parallel-port interface. Most parallel ports use the 25-contact D-sub connector shown in Figure 4.1. The shell (the enclosure that surrounds the contacts) is roughly in the shape of an upper-case D. Other names for this connector are the subminiature D, DB25, D-shell, or just D connector. The IEEE 1284 standard for the parallel port calls it the IEEE 1284-A connector.

Newer parallel ports may use the new, compact, 36-contact IEEE 1284-C connector. The connector on the computer is female, where the individual contacts are sockets, or receptacles. The cable has a mating male connector, whose contacts are pins, or plugs.

The parallel-port connector is usually the only female 25-pin D-sub on the back panel, so there should be little confusion with other connectors. Some serial ports use a 25-contact D-sub, but with few exceptions, a 25-pin serial D-sub on a PC is male, with the female connector on the cable; the reverse of the parallel-port convention. (Other serial ports use 9-pin D-subs instead.)

SCSI is another interface whose connector might occasionally be confused with the parallel ports. The SCSI interface used by disk drives, scanners, and other devices usually has a 50-contact connector, but some SCSI devices use a 25-contact D-sub that is identical to the parallel-port's connector. If you're unsure about which is the parallel-port connector, check your system documentation. When all else fails, opening up the enclosure and tracing the cable from the connector to an expansion board may offer clues.

The pin outs of DB25 connector is shown in the picture below:

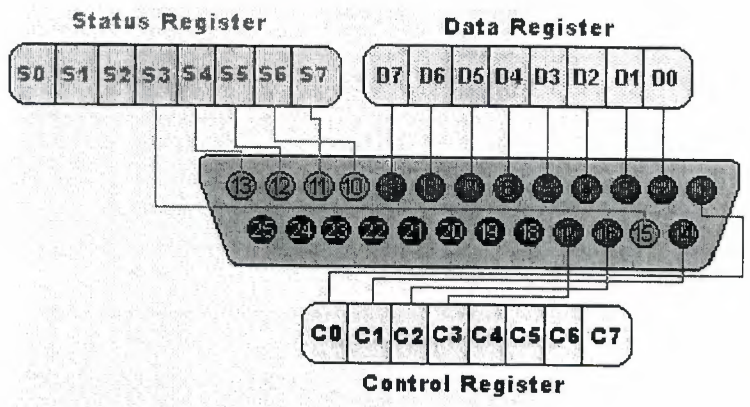


Figure 4.2 Port Configurations.

The lines in DB25 connector are divided in to three groups, they are

1) Data lines (data bus)
2) Control lines
3) Status lines

As the name refers, data is transferred over data lines, Control lines are used to control the peripheral and of course, the peripheral returns status signals back to computer through Status lines. These lines are connected to Data, Control and Status registers internally. The details of parallel port signal lines are given below:

Table 4.1 Parallel Port Signal Lines.

Pin No (DB25)	Signal name	Direction	Register - bit	Inverted
1	nStrobe	Out	Control-0	Yes
2	Data0	In/Out	Data-0	No
3	Data1	In/Out	Data-1	No
4	Data2	In/Out	Data-2	No
5	Data3	In/Out	Data-3	No
6	Data4	In/Out	Data-4	No
7	Data5	In/Out	Data-5	No
8	Data6	In/Out	Data-6	No
9	Data7	In/Out	Data-7	No
10	nAck	In	Status-6	No
11	Busy	In	Status-7	Yes
12	Paper-Out	In	Status-5	No
13	Select	In	Status-4	No
14	Linefeed	Out	Control-1	Yes
15	nError	In	Status-3	No
16	nInitialize	Out	Control-2	No
17	nSelect-Printer	Out	Control-3	Yes
18-25	Ground	-	-	-

4.10 Parallel Port Registers

As we know the Data, Control and status lines are connected to there corresponding registers inside the computer. So by manipulating these registers in program , one can easily read or write to parallel port with programming languages like 'C' or VISUAL BASIC.

The registers found in standard parallel port are:

1) data register
2) Status register
3) Control register

As there names specifies, Data register is connected to Data lines, Control register is connected to control lines and Status register is connected to Status lines. (Here the word connection does not mean that there is some physical connection between data/control/status lines. The registers are virtually connected to the corresponding lines.). So what ever you write to these registers, will appear in corresponding lines as voltages, Of course, you can measure it with a multi-meter. And What ever you give to Parallel port as voltages can be read from these registers (with some restrictions). For example, if we write '1' to Data register, the line Data0 will be driven to +5v. Just like this, we can programmatically turn on and off any of the data lines and Control lines.

Where these registers are?

In an IBM PC, these registers are IO mapped and will have unique address. We have to find these addresses to work with parallel port. For a typical PC, the base address of LPT1 is 0x378 and of LPT2 is 0x278. The data register resides at this base address, status register at base address + 1 and the control register is at base address +2. So once we have the base address, we can calculate the address of each registers in this manner. The table below shows the register addresses of LPT1 and LPT2

Table 4.2 Register Addresses

Register	LPT1	LPT2
data register(base address + 0)	0x378	0x278
status register (base address + 1)	0x379	0x279
control register (base address + 2)	0x37a	0x27a

4.11 Programming Concepts

Almost all programming languages allow programmers to access parallel port using some library functions. For example, Borland C is providing "Inportb" and "Outportb" functions to read or write IO mapped peripherals. Visual Basic does not have any functions or support to access parallel port directly, but it is possible to add such capabilities to your VB application by writing a dll in VC++ and calling its exported functions from VB. VC++ provides two functions to access IO mapped peripherals, '_inp' for reading and '_outp' for writing. These functions are declared in "conio.h".

4.12 Hardware and the program

I will be basically modifying a remote controlled car to be computer controlled using visual basic programming language; so a few electronic components and an RC car are required.

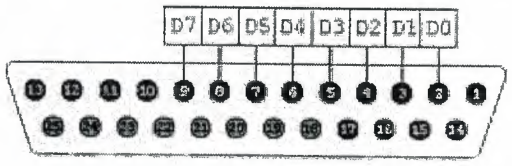
What we may need?

- A small RC car.
- 2 6v SPST Relays.
- 2 IN4148 Diodes.
- 2 IN2002 Diodes.
- 1 BC 547 Transistor.
- 2 4.7 K resistors.
- 3+ meters ribbon wire.
- 25 pin D-type male connector.
- A battery eliminator.

4.12.1 The parallel port

A parallel port can send (by varying the voltage levels) 8 bits in parallel (1 byte), simultaneously to a connected device, for example, a printer. The parallel port that you find at the back of your PC is a 25-pin, D-type female connector. 25 pins allow for powerful communication with devices, but we will restrict ourselves to just sending data out of the parallel port.

As mentioned, a parallel port can send 8 bits simultaneously. The pins that are used to do so are 8 pins from pin numbers 2 to 9. Other pins that we will use are the ground pins (numbers 18, 19, 20, 21, 22, 23, 24 and 25). So many (redundant) ground pins are used for parity checking.



The output pins on the parallel port use TTL (Transistor-Transistor Logic) to pass signals as voltage levels. Voltage levels are either '1' or 'high' (a voltage level of 5V) or '0' or 'low', signified by 0 volts. This is the theory. In practice, the actual voltages will not be exactly these, but around them. The objective of the interfacing circuitry is to identify these voltage levels on the parallel port and use them for driving our car. As for the current, the parallel port itself can source only 4.6 mA of current, which is very less, except for say lighting up LEDs. For driving the motors in the car, we will need more current. This will be provided by a separate power source.

The voltage levels at the parallel port are interpreted as digital signals to make the car perform specific tasks. Pin number 2 is the 0th bit and progressively pin 9 is the 7th bit of the byte that is sent to the connected device through the parallel port. Hence, a value of 11111111 in binary or 255 in decimal would mean that all pins are at a voltage level of 5V.

Generalizing this, a pin status of 00000001 (only the first data pin, i.e. the 2nd pin on the parallel port, or the 0th bit, is active) would be signified by a value 1, or the decimal equivalent of the binary number that corresponds to the desired high–low (one or zero) configuration of the data pins.

Port is the I/O address of the first parallel port attached on your computer, where it might be 0x378.

An important point to note is that this setting of value remains the same even after the program has quit, until the PC is rebooted in which case the parallel port is reset. So, any good programmer should, before exiting the program, set the values back to where he started from. For this, you have to obviously read them in when you start the program.

Now consider a motor inside the car. A typical DC motor can have three states: clockwise motion, anticlockwise motion and no motion. To generate control signals for these three states we will need 2 bits (bit 0 & 1) in which the first bit will used to forward motion and the second bit will be used for the backward motion.

0 0 off

1 0 clockwise motion

0 1 anticlockwise motion

1 1 not needed

So, to create a clockwise motion of the motor we will have to send a value of 1 to the parallel port, similarly a value of 2 for anticlockwise motion.

4.12.2 The circuit

The circuitry is a buffer circuit to drive a relay using +5v volts from the parallel port. The relays, in turn, are connected to the DC motors in the car. The circuit is very basic as shown in the picture, and you can put the components together on a PCB.

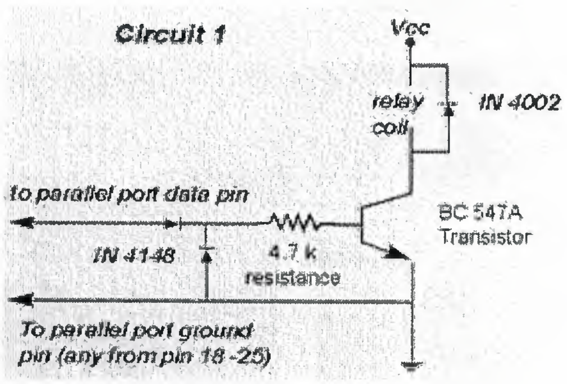

Circuit 1
Vcc
relay coil
IN 4002
to parallel port data pin
BC 547A Transistor
IN 4148
4.7 k resistance
To parallel port ground pin (any from pin 18-25)

Figure 4.3 circuit connection.

The parallel port is a very sensitive electronic device that can fail if the wiring is faulty or if there's a short circuit. In newer motherboards, where the parallel port is built into the board, the fault can domino through the port to your motherboard. The two IN4148 diodes are used as safeguards, but also we have to make sure that the circuit is correctly made before we connect it to the parallel port. Also, we have to check the IN4002 diode because this diode protects the circuit from any back EMF generated from the relay. To be doubly sure, we can replace the IN 4148 diodes with 5.1 V Zener diodes.

The outputs to be connected to the parallel port can be connected through a 25 PIN D type male plug, which can then be connected to the parallel port.

By sending a 1 (binary 00000001) , you can make the motor move clockwise and by sending a 2 you can make the motor move anticlockwise, These translate into forward/backward.

4.12.3 The Program Code

The program code is written visual basic format, and through it we can control the movement of the car using that program or the keyboard.

But as I mentioned before Visual Basic does not have any functions or support to access parallel port directly, but it is possible to add such capabilities to your VB application by writing a Dynamic Link Library (dll) in VC++ and calling its exported functions from VB.

I downloaded that DLL file from the internet and it's called "inpout32.dll" and it has to be copied to the system directory in the windows. After this step you have to add this code as a module in the VB:

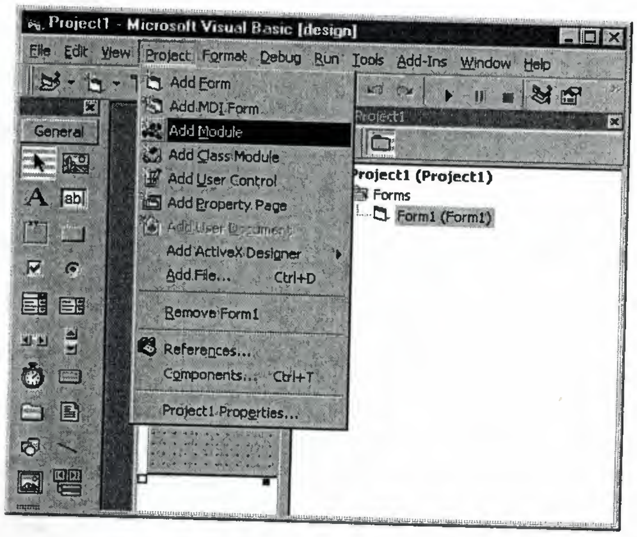


Figure 4.4 Adding a module.

The Module:

Public Declare Function Inp Lib "inpout32.dll" _

Alias "Inp32" (ByVal PortAddress As Integer) As Integer

Public Declare Sub Out Lib "inpout32.dll" _

Alias "Out32" (ByVal PortAddress As Integer, ByVal Value As Integer)

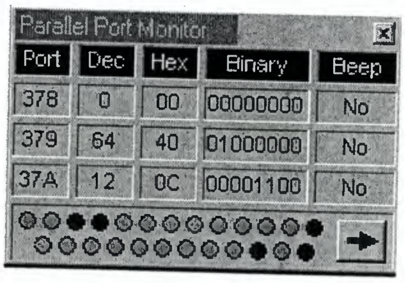
The VB code is:

```
Option Explicit
Dim Keys(255) As Boolean                    (KeyCode goes from 0 to 255)
Dim StopLoop As Boolean         (Its used to stop the Do-Loop of the Form_Load)
Private Sub Command1_Click()
Text2.Text = str(Inp(Val("&H" + Text1.Text)))
End Sub
Private Sub Command2_Click()
Out Val("&H" + Text1.Text), Val(Text2.Text)
End Sub
Private Sub Command3_Click()
End
End Sub
Private Sub Form_KeyDown(KeyCode As Integer, Shift As Integer)
If KeyCode = vbKeyUp Then                                (or KeyCode=38)
Out Val("&H" + Text1.Text), Val(1)
Keys(KeyCode) = True
ElseIf KeyCode = vbKeyDown Then                          (or KeyCode=40)
Out Val("&H" + Text1.Text), Val(2)
Keys(KeyCode) = True
ElseIf KeyCode = vbKeyRight Then                         (or KeyCode=39)
 Out Val("&H" + Text1.Text), Val(5)
Keys(KeyCode) = True
ElseIf KeyCode = vbKeyLeft Then                          (or KeyCode=37)
Out Val("&H" + Text1.Text), Val(9)
Keys(KeyCode) = True
End If
End Sub
Private Sub Form_KeyUp(KeyCode As Integer, Shift As Integer)
Keys(KeyCode) = False
Out Val("&H" + Text1.Text), Val(0)
End Sub
Private Sub Form_Load()
```

```
Dim str As String

 Me.Show
 Me.KeyPreview = True
 StopLoop = False
Do
 str = ""
      If Keys(vbKeyUp) Then str = str + "FORWARD" & vbCrLf
     If Keys(vbKeyDown) Then str = str + "BACKWARD" & vbCrLf
     If Keys(vbKeyLeft) Then str = str + "LEFT" & vbCrLf
     If Keys(vbKeyRight) Then str = str + "RIGHT" & vbCrLf
     lblKeys = str
DoEvents
  Loop Until StopLoop
End Sub
Private Sub Form_Unload(Cancel As Integer)
StopLoop = True
End Sub
```

I used to monitor my parallel port a program which is called "Parallel port monitor" and it's easily helped me in recognizing the parallel port.


Parallel Port Monitor

Port	Dec	Hex	Binary	Beep
378	0	00	00000000	No
379	64	40	01000000	No
37A	12	0C	00001100	No

Figure 4.5 Parallel Port Monitor.

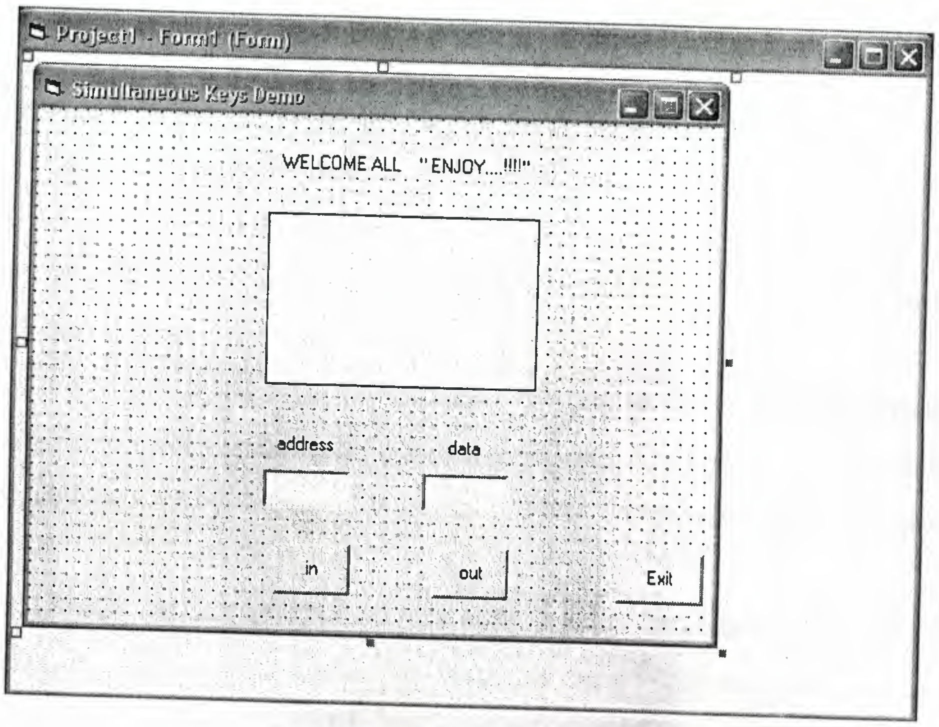


Figure 4.6 The VB program.

CONCLUSION

From my project about WAN and its technologies and features; I conclude that as there is need for advancement in any field of life with due time and according to the needs of the day. Like there can be a simple WAN working but may be it can slow when too many networks connected together due to heavy traffic there can be problem like path is not secure.

Also sometimes we need information very fast as this is the era of multimedia and if we have to send a multimedia file over a network it will take too much time so we have new and new technologies of WAN in order to compensate with the needs of the day.

All these WAN technologies which I have explained are not essential components to make communication but there are some WAN essentials which include software and hardware part in which communication without it may not be possible.

About routers they are just to rout the data. There can be many kinds of routers some can be smarter and some can be just routing. Smarter routers are those especially who can calculate the cost of sending a data over a network as how much time will take and how many ways must be used to make the cost low and time fast. So, while using a WAN its technologies and routing may not be essential but important.

REFERENCES

[1] Mc Graw Hill "TCP/IP Protocol Suite". Behrouz A. Forouzan 2nd edition.2003. Pages 19-40, 60-80, 89-108, 119-137, 228-236.

[2] Dougle E. Comer "Computer Network & Internet".

[3] "Wide Area Networks", 2nd edition.

[4] www.open.ac.uk.

[5] www.intelinfo.com

[6] www.bethanybc.edu/distanteducation.htm

[7] www.research.ct.byu.edu/dlc/main.html

[8] www.openp2p.com/pub/gl/95.

[9] www.openp2p.com/pub/gl/9.

[10] www.intelligenteu.com

[11] www.e-learningcenter.com

[12] http://www.linlsys.com/faqs/default/asp?fqid=17

[13] http://map.sdsu.edu/gisbook/ch3.htm

[14] www.citap.com

[15] www.freebooks.by.ru/view/clientservercomputind/csc03.htm

[16] www.faqs.org/faqs/client-server-faq/

[17] www.schools.nsw.edu.au/adminsupport/schtechnologies/networking/index.php

[18] http://roboticsindia.com/modules.php

[19] http://www.codeproject.com/useritems/Visual_Basic_Game.asp

[20] http://www.control.com/1003297924/index_html

[21] http://www.a1vbcode.com/app.asp?ID=2781

[22] www.logix4u.net

[23] http://www.geocities.com/SiliconValley/Bay/8302/epp.pdf (33kb)

[24] http://www.geekhideout.com