CHI Lab Record

Department of Computer Science and Engineering
College of Engineering Trivandrum

Prathyush P.V

November 10, 2015

COLLEGE OF ENGINEERING TRIVANDRUM

CHI LABORATORY RECORD

UNIVERSITY REG. NO. : 12400040
NAME: PRATHYUSH P.V
ROLL NO: 43
CLASS : S7 R
FROM PAGE NO. 1 TO PAGE NO 50

CERTIFIED BONAFIDE RECORD OF WORK DONE BY PRATHYUSH P.V

Thiruvananthapuram

Staff Member-in-charge

Contents

1. Introduction

AIM

Familiarization of the components / Cards inside a computer, standard connectors, cords, different ports,and various computer peripherals

COMPUTER COMPONENTS

MOTHERBOARD

The motherboard is the main component inside the case. It is a large rectangular board with integrated circuitry that connects the other parts of the computer including the CPU, the RAM, the disk drives as well as any peripherals connected via the ports or the expansion slots.

PROCESSOR

A central processing unit (CPU), also referred to as a central processor unit, [1] is the hardware within a computer that carries out the instructions of a computer program by performing the basic arithmetical, logical, and input/output operations of the system. A computer can have more than one CPU; this is called multiprocessing.Two typical components of a CPU are the arithmetic logic unit (ALU), which performs arithmetic and logical operations, and the control unit (CU), which extracts instructions from memory and decodes and executes them, calling on the ALU when necessary.

CHIPSET

A chipset is a set of electronic components in an integrated circuit that manage the data flow between the processor, memory and peripherals. It is usually found in the motherboard of a computer. Because it controls communications between the processor and external devices, the chipset plays a crucial role in determining system performance. Based on Intel Pentium-class microprocessors, the term chipset often refers to a specific pair of chips on the motherboard: the northbridge and the southbridge. The northbridge links the CPU to very high-speed devices, especially RAM and graphics controllers, and the southbridge connects to lower-speed peripheral buses (such as PCI or ISA).

READ ONLY MEMORY

Read-only memory (ROM) is a class of storage medium used in computers and other electronic devices. Data stored in ROM cannot be modified, or can be modified only slowly or with difficulty, so it is mainly used to distribute firmware (software that is very closely tied to specific hardware and unlikely to need frequent updates). Other types of non-volatile memory such as erasable programmable read only memory (EPROM) and electrically erasable programmable read-only memory (EEPROM or Flash ROM) are sometimes referred to, in an abbreviated way, as "read-only memory" (ROM); although these types of memory can be erased and reprogrammed multiple times, writing to this memory takes longer and may require different procedures than reading the memory.

BIOS

In IBM PC compatible computers, the Basic Input/Output System (BIOS), also known as the system BIOS or ROM BIOS, is a de facto standard defining a firmware interface. The fundamental purposes of the BIOS are to initialize and test the system hardware components, and to load a bootloader or an operating system from a mass memory device. The BIOS additionally provides abstraction layer for the hardware, i.e. a consistent way for application programs and operating systems to interact with the keyboard, display, and other input/output devices. Variations in the system hardware are hidden by the BIOS from programs that use BIOS services instead of directly accessing the hardware.

BUSES

Buses connect the CPU to various internal components and to expansion cards for graphics and sound. It is a physical arrangement that provides the same logical functionality as a parallel electrical bus.
The various Bus architectures currently include:

PCI Express

PCI Express (Peripheral Component Interconnect Express), officially abbreviated as PCIe, is a high-speed serial computer expansion bus standard designed to replace the older PCI, PCI-X, and AGP bus standards. PCIe has higher maximum system bus throughput, lower I/O pin count and smaller physical footprint, better performance-scaling for bus devices, a more detailed error detection and reporting mechanism (Advanced Error Reporting (AER)), and native hot-plug functionality.

PCI

PCI, is a local computer bus for attaching hardware devices in a computer. The PCI bus supports the functions found on a processor bus, but in a standardized format that is independent of any particular processor. Devices connected to the bus appear to the processor to be connected directly to the processor bus, and are assigned addresses in the processor's address space.. Typical PCI cards used in PCs include: network cards, sound cards, modems, extra ports such as USB or serial, TV tuner cards and disk controllers.

SATA

Serial ATA (Advance Technology Attachment)(SATA) is a computer bus interface that connects host bus adapters to mass storage devices such as hard disk drives and optical drives. Serial ATA replaces the older AT Attachment standard (ATA later referred to as Parallel ATA or PATA), offering several advantages over the older interface: reduced cable size and cost (seven conductors instead of 40), native hot swapping, faster data transfer through higher signalling rates, and more efficient transfer through an (optional) I/O queuing protocol.

PORTS

A port serves as an interface between the computer and other computers or peripheral devices.Ports are classified as either serial ports or parallel ports based on the mode of data transfer. Hot-swappable ports can be connected while equipment is running. Plug-and-play ports are designed so that the connected devices automatically start handshaking as soon as the hot-swapping is done. USB ports and FireWire ports are plug-and-play.The most commonly used ports in a computer are:

SERIAL PORT

In computing, a serial port is a serial communication physical interface through which information transfers in or out one bit at a time (in contrast to a parallel port).Throughout most of the history of personal computers, data was transferred through serial ports to devices such as modems, terminals and various peripherals.The common applications for the serial port include dial up modems, GPS receivers, Bar code scanners, Serial mouse etc.

PARALLEL PORT

A parallel port is a type of interface found on computers (personal and otherwise) for connecting peripherals. In computing, a parallel port is a parallel communication physical interface. It is also known as a printer port or Centronics port. The parallel port is usually implemented using a 25 pin DB- 25 connector.

USB

Universal Serial Bus (USB) is an industry standard developed in the mid-1990s that defines the cables, connectors and communications protocols used in a bus for connection, communication, and power supply between computers and electronic devices. USB was designed to standardize the connection of computer peripherals (including keyboards, pointing devices, digital cameras, printers, portable media players, disk drives and network adapters) to personal computers, both to communicate and to supply electric power. USB has effectively replaced a variety of earlier interfaces, such as serial and parallel ports, as well as separate power chargers for portable devices. The presently used USB standard is USB 3.0 which has a data rate of upto 5 GB/sec.

SCSI

Small Computer System Interface (SCSI) is a set of standards for physically connecting and transferring data between computers and peripheral devices. The SCSI standards define commands, protocols and electrical and optical interfaces. SCSI is most commonly used for hard disks and tape drives, but it can connect a wide range of other devices, including scanners and CD drives, although not all controllers can handle all devices. SCSI is an intelligent, peripheral, buffered, peer to peer interface. It hides the complexity of physical format. Every device attaches to the SCSI bus in a similar manner. Up to 8 or 16 devices can be attached to a single bus. There can be any number of hosts and peripheral devices but there should be at least one host.

ESATA

Standardized in 2004, eSATA (e standing for external) provides a variant of SATA meant for external connectivity. It uses a more robust connector, longer shielded cables, and stricter (but backwardcompatible) electrical standards.

FIREWIRE

Firewire (IEEE 1394) is a serial bus interface standard for high-speed communications and isochronous real-time data transfer.The system is commonly used to connect data storage devices and DV (digital video) cameras, but is also popular in industrial systems for machine vision and professional audio systems. It is preferred over the more common USB for its greater effective speed and power distribution capabilities. Firewire supports data transfer rates of up to 3200 Mbits/sec.

EXPANSION DEVICES

The expansion card (also expansion board, adapter card or accessory card) in is a printed circuit board that can be inserted into an electrical connector, or expansion slot on a computer motherboard, backplane or riser card to add functionality to computer system via the expansion bus. The primary purpose of an expansion card is to provide or expand on features not offered by the motherboard. Some commonly used expansion cards are:

VIDEO CARD

A video card is an expansion card which generates a feed of output images to a display. Most video cards offer various functions such as accelerated rendering of 3D scenes and 2D graphics, MPEG-2/MPEG-4 decoding, TV output, or the ability to connect multiple monitors (multi-monitor). It is also called a video controller or graphics controller.

SOUND CARD

A sound card (also known as an audio card) is an internal computer expansion card that facilitates the input and output of audio signals to and from a computer under control of computer programs. Typical uses of sound cards include providing the audio component for multimedia applications such as music composition, editing video or audio, presentation, education and entertainment (games) and video projection.

NETWORK INTERFACE CONTROLLER CARD

A network interface controller (NIC) is a computer hardware component that connects a computer to a computer network. The network controller implements the electronic circuitry required to communicate using a specific physical layer and data link layer standard such as Ethernet, WiFi or Token Ring.

TV TUNER CARD

A TV tuner card is a kind of television tuner that allows television signals to be received by a computer. Most TV tuners also function as video capturecards, allowing them to record television programs onto a hard disk much like the digital video recorder (DVR) does.

SECONDARY STORAGE DEVICES

Computer data storage, often called storage or memory, is a technology consisting of computer components and recording media used to retain digital data. It is a core function and fundamental components of computers. In practice, almost all computers use a storage hierarchy, which puts fast but expensive and volatile small storage options close to the CPU and slower but larger, permanent and cheaper

options farther away. The permanent storage is usually referred to as secondary storage. Secondary storage devices can be broadly classified into two:

FIXED MEDIA

HARD DISK DRIVES Hard drive, hard disk, or disk drive is a device for storing and retrieving digital information, primarily computer data. It consists of one or more rigid rapidly rotating discs (often referred to as platters), coated with magnetic material and with magnetic heads arranged to write data to the surfaces and read it from them. An HDD retains its data even when powered off. Data is read in a random-access manner. HDDs are connected to systems by standard interface cables such as SATA (Serial ATA), USB or SAS (Serial attached SCSI) cables. The capacity of modern hard drives ranges from 500 GB to 4 TB.

SOLID STATE DRIVES A solid-state drive (SSD), sometimes called a solid-state disk or electronic disk, is a data storage device that uses solid-state memory to store persistent data with the intention of providing access in the same manner of a traditional block I/O hard disk drive. SSDs are distinguished from traditional magnetic disks such as hard disk drives (HDDs) or floppy disk, which are electromechanical devices containing spinning disks and movable read/write heads. Compared with electromechanical disks, SSDs are typically more resistant to physical shock, run more quietly, have lower access time, and less latency. The capacity of modern SSDs usually ranges from 64 GB to 1 TB.

REMOVABLE MEDIA

OPTICAL DISK DRIVES An optical disc drive (ODD) is a disk drive that uses laser light or electromagnetic waves within or near the visible light spectrum as part of the process of reading or writing data to or from optical discs. Some drives can only read from discs, but recent drives are commonly both readers and recorders, also called burners or writers. Compact discs, DVDs, and Blu-ray discs are common types of optical media which can be read and recorded by such drives.

FLOPPY DISK DRIVES They are used for reading and writing to floppy disks, an outdated storage media consisting of a thin disk of a flexible magnetic storage medium. These were once standard on most computers but are no longer in common use. Floppy disks, initially as 8-inch (200 mm) media and later in

5.25-inch (133 mm) and 3.5-inch (90 mm) sizes, were a ubiquitous form of data storage and exchange from the mid-1970s well into the first decade of the 21st century. Floppies are used today mainly for loading device drivers not included with an operating system release.

USB Flash drives A USB flash drive is a data storage device that includes flash memory with an integrated Universal Serial Bus (USB) interface. USB flash drives are typically removable and rewritable, and physically much smaller than an optical disc. Modern USB drives can store data up to 256 GB.

TAPE DRIVES A tape drive is a data storage device that reads and writes data on a magnetic tape. Magnetic tape data storage is typically used for offline, archival data storage. Tape media generally has a favorable unit cost and long archival stability. A tape drive provides sequential access storage, unlike a disk drive, which provides random access storage. A disk drive can move to any position on the disk in a few milliseconds, but a tape drive must physically wind tape between reels to read any one particular piece of data. As a result, tape drives have very slow average seek times. However, the storage capacity of magnetic tapes is considerably more than other secondary storage mediums.

INPUT AND OUTPUT PERIPHERALS

Input and output devices are typically housed externally to the main computer chassis. The following are either standard or very common to many computer systems.

INPUT DEVICES

KEYBOARDS A keyboard is a device to input text and characters by depressing buttons. It is a typewriterstyle device, which uses an arrangement of buttons or keys, to act as mechanical levers or electronic switches. While most keyboard keys produce letters, numbers or signs (characters), other keys or simultaneous key presses can produce actions or execute computer commands.

MOUSE A mouse is a pointing device that functions by detecting two-dimensional motion relative to its supporting surface. Physically, a mouse consists of an object held under one of the user's hands, with one or more buttons. The mouse's motion

typically translates into the motion of a pointer on a display, which allows for fine control of a graphical user interface.

TRACKBALL A trackball is a pointing device consisting of a ball held by a socket containing sensors to detect a rotation of the ball about two axes—like an upside-down mouse with an exposed protruding ball. The user rolls the ball with the thumb, fingers, or the palm of the hand to move a pointer.

TOUCHSCREEN A touchscreen is an electronic visual display that the user can control through simple or multi-touch gestures by touching the screen with one or more fingers. Some touchscreens can also detect objects such as a stylus or ordinary or specially coated gloves. The user can use the touchscreen to react to what is displayed and to control how it is displayed (for example by zooming the text size). The touchscreen enables the user to interact directly with what is displayed, rather than using a mouse, touchpad, or any other intermediate device.

JOYSTICK A joystick is an input device consisting of a stick that pivots on a base and reports its angle or direction to the device it is controlling. Joysticks are often used to control video games, and usually have one or more push-buttons whose state can also be read by the computer.

IMAGE SCANNER In computing, an image scanner—often abbreviated to just scanner—is a device that optically scans images, printed text, handwriting, or an object, and converts it to a digital image. Common examples found in offices are variations of the desktop (or flatbed) scanner where the document is placed on a glass window for scanning. Hand-held scanners, where the device is moved by hand, have evolved from text scanning "wands" to 3D scanners used for industrial design, reverse engineering, test and measurement, orthotics, gaming and other applications. Mechanically driven scanners that move the document are typically used for large-format documents, where a flatbed design would be impractical. Modern scanners typically use a charge-coupled device (CCD) or a Contact Image Sensor (CIS) as the image sensor, whereas older drum scanners use a photomultiplier tube as the image sensor. A rotary scanner, used for high-speed document scanning, is another type of drum scanner, using a CCD array instead of a photomultiplier. Other types of scanners are planetary scanners, which take photographs of books and documents, and 3D scanners, for producing threedimensional models of objects.

MICROPHONE A microphone is an acoustic to electric transducer or sensor that converts sound into an electrical signal. Microphones are used to input audio data into the computer for processing.

OUTPUT DEVICES

PRINTERS In computing, a printer is a peripheral which produces a representation of an electronic document on physical media such as paper or transparency film. Many printers are local peripherals connected directly to a nearby personal computer. Individual printers are often designed to support both local and network connected users at the same time. Depending on the technology used, there can be several variants of printers such as inkjet printers, laser printers, dot matrix printers, thermal printers etc.

COMPUTER MONITORS A monitor or a display is an electronic visual display for computers. The monitor comprises the display device, circuitry and an enclosure. The display device in modern monitors is typically a thin film transistor liquid crystal display (TFT-LCD) thin panel, while older monitors use a cathode ray tube (CRT) about as deep as the screen size.

SPEAKERS Computer speakers, or multimedia speakers, are speakers external to a computer, that disable the lower fidelity built-in speaker. They often have a low-power internal amplifier. The standard audio connection is a 3.5 mm (approximately 1/8 inch) stereo phone connector often color-coded lime green (following the PC 99 standard) for computer sound cards. Analog A/V connectors often use shielded cables to inhibit radio frequency interference (RFI) and noise. Some commonly used connectors are as follows:

RCA An RCA connector, sometimes called a phono connector or cinch connector, is a type of electrical connector commonly used to carry audio and video signals. The connectors are also sometimes casually referred to as A/V jacks.

VGA A Video Graphics Array (VGA) connector is a three-row 15-pin DE-15 connector which carries video signals. The 15-pin VGA connector is found on many video cards, computer monitors, and high definition television sets.

DVI Digital Visual Interface (DVI) is a video display interface developed by the Digital Display Working Group (DDWG). The digital interface is used to connect a video source to a display device, such as a computer monitor.

HDMI HDMI (High-Definition Multimedia Interface) is a compact audio/video interface for transferring uncompressed video data and compressed/uncompressed digital audio data from a HDMIcompliant device ("the source device") to a compatible computer monitor, video projector, digital television, or digital audio device.

2. Computer Assembly

AIM

To study the assembly of a Computer from its components

PROCEDURE

Following are the steps involved in assembling a computer from its components:

Step 1: Open the case by removing the side panels.

Step 2: Install the Power Supply.

Power supply installation steps include the following:

Insert the power supply into the case.

Align the holes in the power supply with the holes in the case.

Secure the power supply to the case using the proper screws.

Step 3: Attach Components to the Motherboard

i) CPU on Motherboard

The CPU and motherboard are sensitive to electrostatic discharge.
The CPU is secured to the socket on the motherboard with a locking assembly.
Apply thermal compound which helps keep the CPU cool.

ii) Heat Sink/Fan Assembly

The Heat Sink/Fan Assembly is a two-part cooling device.
The heat sink draws heat away from the CPU.

iii) Install RAM

RAM provides temporary data storage for the CPU and should be installed in the motherboard before the motherboard is placed in the computer case.
iv) Install motherboard in the computer case.

Step 4: Install the Hard Disk drive in the 3.5 inch internal drive bay using screws.

Step 5: Install optical disc drive and floppy disc drive in the external drive bays provided for the same.

Step 6: Install Adapter cards: Expansion cards such as NIC cards, video cards, sound cards etc. should be installed in the slots (PCI, PCIe) provided for the same in the motherboard.

Step 7: Connect all the internal power and data cables. Power cables distribute electricity from the power supply to the motherboard and other components. Data cables transmit data between the motherboard and storage devices, such as hard drives.

Step 8: Close the case by reattaching the side panels. Connect the external cables.

Step 9: Boot computer for first time. The BIOS will perform a Power On Self Test (POST) to check all of the internal components. If a device is malfunctioning, an error or beep code will be generated.

Step 10: If the computer is functioning properly, install a suitable operating system.

3. Data communication through the serial port

AIM:

To transfer
1) Single data between 8051 microcontroller ICs.
2) an array of numbers between 8051 microcontroller ICs.

ALGORITHM:

Single data transfer
Sender:

STEP 1:SBUF <- Element to be transfered
Receiver:
STEP 1: Clear R1 bit
STEP 2: When R1 is set, move SBUF into an external memory location

PROGRAM:

Sender:

```
MOV  SBUF   ,#FF
```

```
LOOP:  SJMP  LOOP
```

Receiver:

```
MOV  DPTR,#4500
CLR  R1
DATA_NOT_READY:  JNB  R1,DATA_NOT_READY
MOV  A,SBUF
MOVX  @DPTR,A
```

Sample Output

At the receiver side
(4200): FF

ALGORITHM:

Array transfer
Sender:

STEP 1: R1 <- count
STEP 2: Clear RI
STEP 3: A <- Element to be transfered
STEP 4: SBUF <- A
STEP 5: If RI bit is set, decrement R1 and goto step 2
Receiver:

STEP 1: Clear RI bit

STEP 2: If RI bit is set, A <- SBUF
STEP 3: Store A in a memory location
STEP 4: SBUF <- 01
STEP 5: Goto step 1

PROGRAM:

Sender:

```
MOV DPTR,#4200
MOV R1,0A
SEND_LOOP: CLR 98
MOV A,@DPTR
MOV SBUF,A
INC DPTR
LOOP: JNB 98,LOOP
DJNZ R1,SEND_LOOP
END_LOOP: SJMP END_LOOP
```

Receiver:

```
MOV DPTR,#4500
REC_LOOP: CLR 98
LOOP: JNB 98,LOOP
MOV A,SBUF
MOVX @DPTR,A
INC DPTR
MOV SBUF,#01
SJMP REC_LOOP
```

Sample Input

At the sender side
01 02 03 04 05 06

Sample Output

At the receiver side
01 02 03 04 05 06

RESULT:

The program has been executed and the output verified

4. COM Port base address

AIM:

To find the base address of the COM port in a system.

THEORY:

COM is the original, yet still common, name of the serial port interface on IBM-PC compatible computers. It might refer not only to physical ports, but also to virtual ports, such as ports created by Bluetooth or USB-to-serial adapters. COM ports are usually associated with 4 memory addresses as shown below:

NAME	ADDRESS	IRQ
COM1	0x3f8	4
COM2	0x2f8	3
COM3	0x3e8	4
COM4	0x2e8	3

The base addresses for the COM ports can be read from the BIOS data area. The addresses in the BIOS data area that store the COM port base addresses are as follows:

Start Address	Function
0000:0400	COM1's Base Address
0000:0402	COM2's Base Adress
0000:0403	COM3's Base Address
0000:0404	COM4's Base Address

PROCEDURE:

Execute the following program to find the base address of the COM port.

PROGRAM:

```
#include <stdio.h>
#include <string.h>

int main()
{
      char s[100],string[100];
      int com=0;

      //--Executing BASH script to read all messages in ker
      //  search for the string "ttyS" in these messages an
      //  redirect the search results into a text file for

      system("dmesg | grep ttyS > out.txt");
```

```
        FILE *fp;
        fp=fopen("out.txt","r");

        while( !feof(fp) )
        {
                fscanf(fp, "%s", string);
                if( strcmp(string ,"I/O") == 0 )
                {
                        fscanf(fp, "%s", s);
                        printf("COM PORT%d ADDRESS IS %s\n",com,s);
                }
        }

        fclose(fp);
        return 0;
}
```

RESULT

The program has been executed and the output, verified.

5. Parallel Port Inter-Computer Communication

AIM:

To realize inter-computer communication through the parallel port

THEORY:

For inter computer communication, the bidirectional mode of the parallel port should be enabled. This is controlled by bit number 6 of the control port associated with the parallel port. When it is cleared to 0, data values can be output to the data pins. When it is set to 1, external data values can be read from the data pins.

PROCEDURE:

STEP 1: Connect parallel ports of two computers

STEP 2: Enter programs for sender and receiver into the corresponding computers

STEP 3: Compile and run the program in sender

STEP 4: Compile and run the program in receiver

STEP 5: Observe data. as entered at sender side, at receiver side

PROGRAM:

SENDER:

```
#include <stdio.h>
#include <unistd.h>

int main(){
      int addr;
      addr = 0x378;
      ioperm(addr, 3, 1);

      //--configure as sender
      outb(0x0B, addr+2);

      //--send data 0xFF
      outb(0xFF, addr);

      ioperm(addr, 3, 0);
      return 0;
}
```

RECEIVER:

```
#include <stdio.h>
#include <unistd.h>

int main(){
      int addr;
      addr = 0x378;
      ioperm(addr ,3 ,1);
```

```
        //--configure as receiver
        outb(0x2B, addr+2);

        //--receive data
        printf("\nReceived %x\n", inb(addr) );

        ioperm(addr, 3, 0);
        return 0;
}
```

RESULT:

The program has been executed and the output verified

6. LED Ring Counter

AIM:

To light up LEDs in a ring counter pattern using the parallel port

THEORY:

A parallel port is a type of interface found on computers (personal and otherwise) for connecting peripherals. In computing, a parallel port is a parallel communication physical interface. It is also known as a printer port or Centronics port. The port is composed of 4 control lines, 5 status lines and 8 data lines. The connector for the parallel port is a 25 pin D-Type female connector. The parallel port pinout is as shown below:

The parallel port has three commonly used port addresses as shown below:

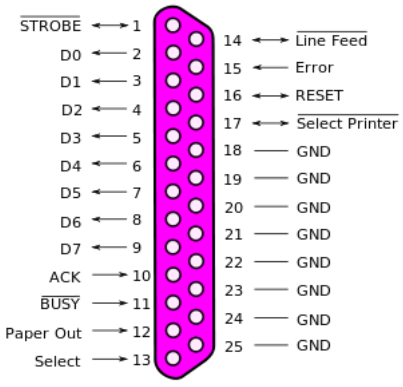

Figure 6.1: Parallel port pinout

Address	Notes
3BCh - 3BFh	Used for parallel ports which were incorporated onto Video cards - doesn't support ECP addresses
378h - 37Fh	Usual address for LPT1
278h - 27Fh	Usual address for LPT2

Table 6.1: Parallel port base addresses

The parallel port is associated with three software registers namely- the data port, the status port, and the control port. These ports can be used to read/write data and control the operation of the parallel port. A detailed description of the individual bits in these registers is given below:

The data port is simply used for outputting data on the Parallel Port's data lines (Pins 2-9). This register is normally a write only port. However, modern parallel ports are often bi-directional, and data can be read from the data port address.

The Status Port is a read only port. Any data written to this port

Offset	Name	Read/Write	Bit No.	Properties
Base + 0	Data Port	Write	Bit 7	Data 7
			Bit 6	Data 6
			Bit 5	Data 5
			Bit 4	Data 4
			Bit 3	Data 3
			Bit 2	Data 2
			Bit 1	Data 1
			Bit 0	Data 0

Table 6.2: Data Port register bits

will be ignored. The Status Port is made up of 5 input lines (Pins 10,11,12,13 & 15), an IRQ status register and two reserved bits.

Offset	Name	Read/Write	Bit No.	Properties
Base + 1	Status Port	Read Only	Bit 7	Busy
			Bit 6	Ack
			Bit 5	Paper Out
			Bit 4	Select In
			Bit 3	Error
			Bit 2	IRQ
			Bit 1	Reserved
			Bit 0	Reserved

Table 6.3: Status Port register bits

The Control Port (base address + 2) was intended as a write only port. When a printer is attached to the Parallel Port, four "controls" are used. These are Strobe, Auto Linefeed, Initialize and Select Printer, all of which are inverted except Initialize.

The printer would not send a signal to initialize the computer, nor would it tell the computer to use auto linefeed. However these four outputs can also be used for inputs. If the computer has placed a pin high (e.g. +5v) and your device wanted to take it low, you would effectively short out the port, causing a conflict on that pin. Therefore these lines are "open collector" outputs (or open drain for CMOS devices). This means that it has two states. A low state (0v) and a high impedance state (open circuit).

Bits 4 & 5 are internal controls. Bit four will enable the IRQ (See Using the Parallel Ports IRQ) and Bit 5 will enable the bi-directional port meaning so as to enable input of 8 bits using (DATA0-7). This mode is only possible if the card supports it. Bits 6 & 7 are reserved. Any writes to these two bits will be ignored.

Offset	Name	Read/Write	Bit No.	Properties
Base + 2	Control Port	Read/Write	Bit 7	Unused
			Bit 6	Unused
			Bit 5	Enable Bi-Directional Port
			Bit 4	Enable IRQ via ACK line
			Bit 3	Select Printer
			Bit 2	Initialize Printer (Reset)
			Bit 1	Auto Linefeed
			Bit 0	Strobe

Table 6.4: Control Port register bits

PROCEDURE:

STEP 1: Setup a circuit connecting the parallel port to 4 LEDs.

STEP 2: Compile and run the program given below in Linux environment with super user privilege.

The four LEDs will be seen lighting up in the ring counter pattern.

PROGRAM:

```
#include <stdio.h>
#include <sys/io.h>
#include <unistd.h>

void main(){

      int base = 0x378;
```

```
        if( ioperm(base, 1, 1) == -1){
              printf("\nError granting permission.Terminating.
              return;
        }else{}

        int i;
        //--outputting powers of 2 from 1 to 16, one in every

        for(i=1 ; ; i=1)
              for(; i<16 ; sleep(1), i = i<<1)
                    outb(i);

        ioperm(base, 1, 0);
        return;
}
```

RESULT:

The program has been executed and the output, verified.

7. Testing of gates

AIM:

To study the user of parallel ports in the testing of digital IC's

PROCEDURE:

STEP 1: Start

STEP 2: Connect the parallel port IC 7408 as per the circuit diagram.

STEP 3: Compile and run the program using the super user privilege in the Linux environment.

STEP 4: Power supply for the logic gates should be came from the parallel ports. For this purpose keep one of the dateline in the parallel ports always in 1 logic state. As these equates to a constant voltage source of +5 volts and connect same to the Vcc pin of the IC. Ground of the IC should be connected to common ground of the parallel ports

STEP 5: Stop.

For the following program to run the Pin D4 is always kept in the high state and individual truth table values are input to the gate by varying the logic state of the pins D0 & D1. By observing the output of gates for each input associated by whether the LED light

up or not. The truth table of the gates can be verified.

PROGRAM:

```
#include<studio.h>
#include<unistd.h>
void main(){
int addr=0x378;
ioperm(addr,3,1);
outb(0x10,addr);
sleep(1);
outb(0x11,addr);
sleep(1);
outb(0x12,addr);
sleep(1);
outb(0x13,addr);
sleep(1);
}
```

RESULT:

The program has been executed and the output verified

8. Parallel Port Data Input

AIM:

To input an 8-bit data word from an external source using the parallel port

THEORY:

An external data source can be simulated by a DC power supply. A voltage value of +5 volt corresponds to a logic state of 1, and the ground of the power supply corresponds to a logic state of 0. Any combination of 8 bits can be input to the data pins of the parallel port by varying the voltage levels at the pins accordingly. The parallel port should be configured to read external data by modifying bit number 6 in the control port.

PROCEDURE:

Setup a circuit connecting the parallel port to a power supply. Compile and run the program (given next) in a Linux Environment with superuser privilege. The external input bit string will be read and displayed to the user until *Ctrl+C* is pressed.

PROGRAM:

```
#include <stdio.h>
#include <sys/io.h>
#include <unistd.h>
#include <inttypes.h>
#include <signal.h>

int running = 1;

void signalhandler(int sig){ running = 0; }

void main(){

 //--Instructs the program to execute signalhandler when C
 //--SIGINT is a constant in inttypes.h defined as correspo

       signal(SIGINT , signalhandler);

       int addr = 0x378;

       if( ioperm(addr, 4, 1) == -1){
            printf("Unable to grant permission.Terminating.
            return;
       }else{}

       //--configuring to read input
       outb(0x2b, addr+2);

       while(running){
```

```
            printf("\nRead : %x from external source", inb(
            sleep(1);
        }

        ioperm(addr, 4, 0);
        return;
}
```

RESULT:

The program has been executed and the output, verified.

9. Decimal to Hexadecimal Conversion

AIM:

To convert decimal number to hexadecimal number.

ALGORITHM:

STEP 1: Start
STEP 2: A <-Decimal number
STEP 3: Take a copy of A to B
STEP 4: AND A with 0F then store it in location
STEP 5: AND A with F0 and store it in another location
STEP 6: Swap separated upper and lower nibbles of A
STEP 7: B <- 0A
STEP 8: Multiply A with B
STEP 9: Add A with lower nibbles
STEP 10: Store A
STEP 11: Stop

PROGRAM:

```
MOV DPTR,#4200H
 MOVX A,@DPTR
 MOV B,#0A
 MUL AB
 MOV B,A
 INC DPTR
 MOVX A,@DPTR
 ADD A,B
 MOV DPTR,#4300H
 MOVX @DPTR,A
 HLT: SJMP HLT
```

Sample Input

(4200): 23

Sample Output

(4300): 07 (4301): 01

RESULT:

The program has been executed and the output verified

10. Multibyte Addition

Problem Definition

To perform multi byte addition of two numbers (16 bit)

Algorithm

1 start
2 move the content of memory locations 4501,4503,to R0 and A
3 ADD R0, A and store result into 4602
4 move the content of memory locations 4500,4502,to R0 and A
5 ADD with carry R0, A and store result into 4601
6 if there is a carry then place 01 to 4600
7 else place 00 to 4600
8 end

Program

```
CLR C
MOV DPTR,#4501
MOVX A,@DPTR
```

```
MOV R0,A
MOV DPTR,#4503
MOVX A,@DPTR
ADD A,R0
MOV DPTR,#4602
MOVX @DPTR,A
MOV DPTR,#4500
MOVX A,@DPTR
MOV R0,A
MOV DPTR,#4502
MOVX A,@DPTR
ADDC A,R0
MOV DPTR,#4601
MOVX @DPTR,A
MOV DPTR,#4600
MOV A,#00
JNC NO
MOV A,#01
NO :MOVX @DPTR,A
HLT :SJMP HLT
```

Sample Input

(4500): 12 34
(4502): 56 78

Sample Output

(4600): 00
(4601): 68
(4602): AC

Result

Program executed successfully and the output is verified.

11. BCD to ASCII Conversion

AIM:

To convert BCD number to an ASCII number.

ALGORITHM:

STEP 1: Start
STEP 2: A <- BCD number
STEP 3: Separate upper and lower nibble by ANDing with F0
STEP 4: Swap the content of A
STEP 5: Perform OR with 30
STEP 6: Separate lower nibble
STEP 7: OR it with 30 to convert to ASCII
STEP 8: Store A
STEP 9: Stop

PROGRAM:

```
MOV DPTR,#4200H
MOVX A,@DPTR
MOV R1,A
ANL A,#0F
```

```
ADD A,#30
MOV DPTR,#4300H
MOVX @DPTR,A
MOV A,R1
ANL A,#F0
SWAP A
ADD A,#30
INC DPTR
MOVX @DPTR,A
HLT: SJMP HLT
```

Sample Input

(4200): 64

Sample Output

(4300): 34 (4301): 36

RESULT:

The program has been executed and the output verified

12. Search From an Array

Problem Definition

To search for a number from a list of 10 numbers

Algorithm

1.Start
2.Move the searching element to R1 register
3.Move the array size to R0
4.DPTR points to searching location of the arraty
5.Move the content of DPTR to A
6.Compare A with content of R1
7.If not equal, jump to 9
8.If the element is found in the array, set the memory
location 4400 with 01 jump to end
9.Increment DPTR
10.Decrement R0, if not equal to 0, jump to 6
11.If element is not found, set memory location 4400 with 0
12.Stop

Program

```
MOV R0,#10
MOV DPTR,#4200
LOOP1:MOVX A,@DPTR
CJNE A,4300H,NO
MOV 4350H,#01
SJMP HLT
NO :INC DPTR
DJNZ R0,LOOP1
HLT :SJMP HLT
```

Sample Inputs

(4200): 07 05 08 09 03 01 02 0A 04 06
(4300): 06

Sample Outputs

(4350): 01

Result

Program executed successfully and the output is verified.

13. Hexadecimal to Decimal Conversion

AIM:

To convert hexadecimal number to decimal number.

ALGORITHM:

STEP 1: Start
STEP 2: A<-Hexadecimal number
STEP 3: B<-10
STEP 4: Divide A by B
STEP 5: Store reminder in a register
STEP 6: B<-10
STEP 7: Divide A by B
STEP 8: Move the quotient in A to any other register
STEP 9: A<-B
STEP 10: Swap(upper-nibble(A),lower-nibble(A))
STEP 11: Perform OR operation between A and remainder stored
STEP 12: Store A
STEP 13: Stop

PROGRAM:

```
MOV DPTR,#4200H
MOVX A,@DPTR
MOV B,#64
DIV AB
MOV DPTR,#4300
MOVX @DPTR,A
MOV A,B
MOV B,#0A
DIV AB
INC DPTR
MOVX @DPTR,A
INC DPTR
MOV A,B
MOVX @DPTR,A
HLT: SJMP HLT
```

Sample Input

(4200): FF

Sample Output

(4300): 01 (4301): 02 (4302): 07

RESULT:

The program has been executed and the output verified

14. Factorial

Problem Definition

To find the factorial of an input number.

Algorithm

1 start
2 move the number to register R0
3 move 1 to register A
4 multiply R0 with A
5 decrement R0
6 if R0 greater than 0 goto 3
7 end

Program

```
MOV DPTR,#4200H
MOVX A,@DPTR
MOV B,A
MOV A,#01
LOOP1 :MOV R1,B
```

```
MOV R2,A
MUL AB
MOV B,R1
DJNZ B,LOOP1
MOV DPTR,#4300H
MOVX @DPTR,A
HLT :SJMP HLT
```

Sample Input

(4200): 05

Sample Output

(4300): 78

Result

Program executed successfully and the output is verified.

15. Fibonacci

Problem Definition

To generate a Fibonacci series with the limit read as the input

Algorithm

1.Start
2.Point DPTR to memory location 4500
3.Move the count to register
4.Move 00 to A
5.Move A to memory location 4500
6.Move A to R1
7.Decrement count
8.Increment DPTR
9.Move 01 to A register, then to R2 register
10.Decrement count
11.Move R1 to A
12.Add A with R2
13.Move A to R3
14.Move content of DPTR to A, then move R2
15.Decrement DPTR

16.Move R3 to A, then move to the desired memory location
17.Decrement R0
18.If R0 is not zero,loop
19.Stop

Program

```
MOV DPTR,#4200H
MOVX A,@DPTR
MOV R7,A
MOV A,#00
MOV R1,#01
MOV DPTR,#4300H
LOOP1 :MOV @DPTR,A
INC DPTR
ADD A,R1
MOV R2,A
MOV A,R1
MOV R3,A
MOV A,R2
MOV R1,A
MOV A,R3
DJNZ R7,LOOP1
HLT :SJMP HLT
```

Sample Inputs

(4200): 07

Sample Outputs

(4300): 00 01 01 02 03 05 08

Result

Program executed successfully and the output is verified.

16. Largest number in an array

Largest number in an array

AIM:

To find the largest number in an array of 10 numbers.

ALGORITHM:

STEP 1: Start

STEP 2: Get first element to R1 register

STEP 3: Get the next element in A register

STEP 4: Subtract the R1 register from the A register.

STEP 5: Check the carry flag.

STEP 6: If carry is not there, then A register is greater that means 2nd data is larger than 1stData. So exchange the A and B register.

STEP 7: If carry is there, then B register is greater that means 1st data is larger than 2 nd Data. So do not exchange the A and B register.

STEP 8: Check the R0 register. If R0 is not equal to zero, repeat from the 2 nd step else Move the R1 register to a memory location.

STEP 11: Stop

PROGRAM:

```
MOV R0,#10
MOV DPTR,#4200H
MOVX A,@DPTR
MOV R1,A
INC DPTR
DEC R0
LOOP1 :MOVX A,@DPTR
MOV R2,A
SUBB A,R1
JC NO
MOV R1,R2
NO :INC DPTR
DJNZ R0,LOOP1
MOV DPTR,#4300H
MOV A,R1
MOVX @DPTR,A
HLT :SJMP HLT
```

Sample Input

(4200): 07 05 09 03 01 02 0A 04 06

Sample Output

(4300): 0A

RESULT:

The program has been executed and the output verified

17. Soring in ascending order

AIM:

To sort an array of numbers in ascending order.

ALGORITHM:

STEP 1: Start
STEP 2: Store the count in R0
STEP 3: Decrement R0 if R0!=0, goto step 5
STEP 4: Stop
STEP 5: R1 <- R0
STEP 6: A <- next element in array, 40H <- next element in array
STEP 7: Compare A and 40H
STEP 8: If A > 40H, swap the contents
STEP 9: Decrement R1. If R1!=0 goto step 5, otherwise goto step 2

PROGRAM:

```
MOV DPTR,#5000H
MOVX A,@DPTR
MOV R0,A
```

```
LABEL6 : DJNZ R0,LABEL1
LABEL2 : SJMP LABEL2
LABEL1 : MOV A,R0
MOV R1,A
MOV DPTR,#5001
MOVX A,@DPTR
MOV R2,A
INC DPTR
LABEL5 : MOVX A,DPTR
MOV 40H,A
MOV A,R2
CJNE A,40H,LABEL3
LABEL4 : DJNZ R1,LABEL5
SJMP LABEL6
LABEL3 : JC LABEL4
MOVX @DPTR,A
MOV A,DPL
DEC A
MOV DPL,A
MOV A,40H
MOVX @DPTR,A
INC DPTR
HALT : SJMP HALT
```

Sample Input

(5000): 05 (5001): 02 (5002): 04 (5003): 0F (5004): 03 (5005): 07

Sample Output

(5001): 02 (5002): 03 (5003): 04 (5004): 07 (5005): 0F

RESULT:

The program has been executed and the output verified

18. Soring in descending order

AIM:

To sort an array of numbers in descending order.

ALGORITHM:

STEP 1: Start
STEP 2: Store the count in R0
STEP 3: Decrement R0 if R0!=0, goto step 5
STEP 4: Stop
STEP 5: R1 <- R0
STEP 6: A <- next element in array, 40H <- next element in array
STEP 7: Compare A and 40H
STEP 8: If A < 40H, swap the contents
STEP 9: Decrement R1. If R1!=0 goto step 5, otherwise goto step 2

PROGRAM:

```
MOV DPTR,#5000H
MOVX A,@DPTR
MOV R0,A
```

```
LABEL6 : DJNZ R0,LABEL1
LABEL2 : SJMP LABEL2
LABEL1 : MOV A,R0
MOV R1,A
MOV DPTR,#5001
MOVX A,@DPTR
MOV R2,A
INC DPTR
LABEL5 : MOVX A,DPTR
MOV 40H,A
MOV A,R2
CJNE A,40H,LABEL3
LABEL4 : DJNZ R1,LABEL5
SJMP LABEL6
LABEL3 : JC LABEL4
SJMP LABEL4
MOVX @DPTR,A
MOV A,DPL
DEC A
MOV DPL,A
MOV A,40H
MOVX @DPTR,A
INC DPTR
HALT : SJMP HALT
```

Sample Input

(5000): 05 (5001): 02 (5002): 04 (5003): 0F (5004): 03 (5005): 07

Sample Output

(5001): 0F (5002): 07 (5003): 04 (5004): 03 (5005): 02

RESULT:

The program has been executed and the output verified