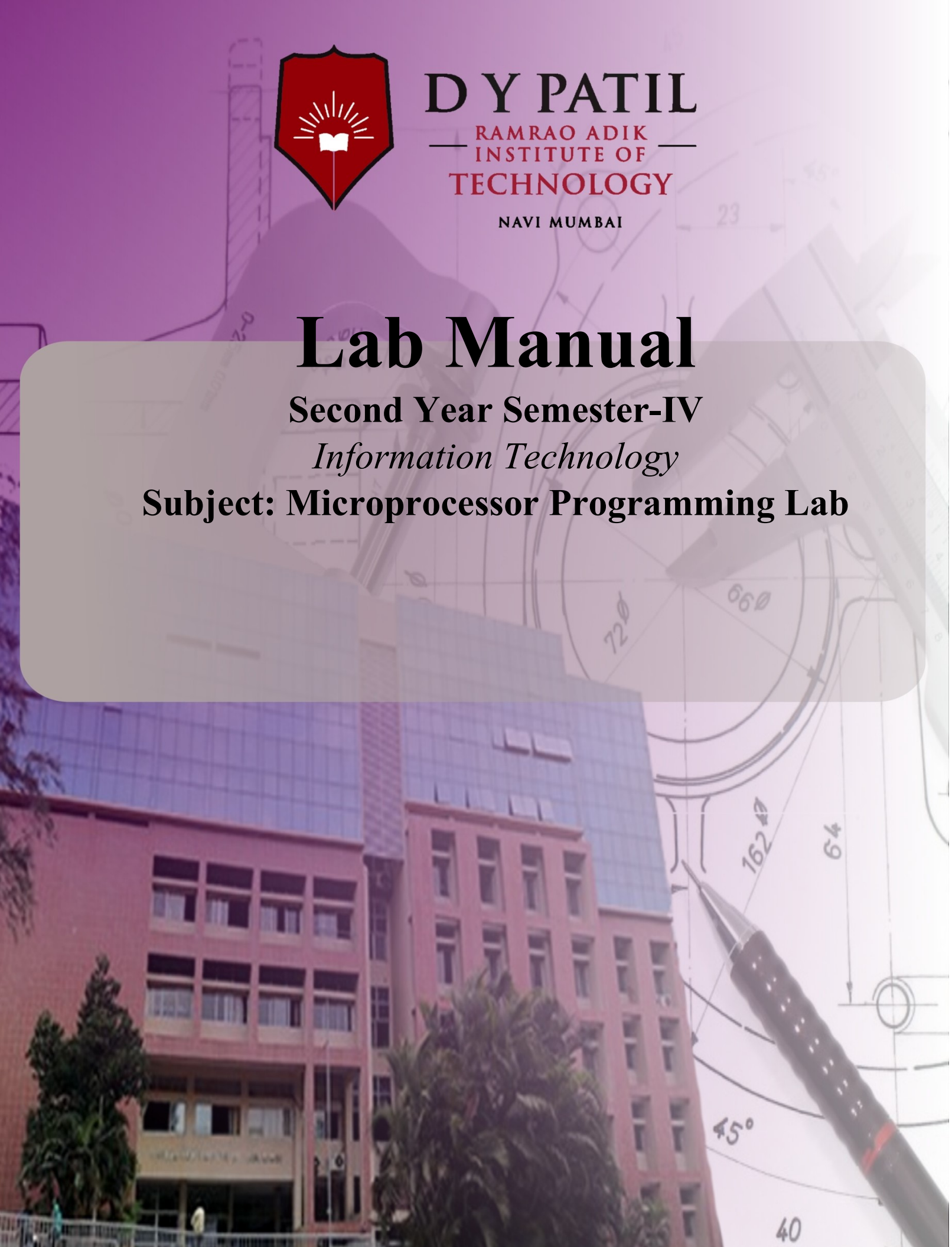
D Y PATIL
RAMRAO ADIK INSTITUTE OF TECHNOLOGY
NAVI MUMBAI

Lab Manual

Second Year Semester-IV

Information Technology

Subject: Microprocessor Programming Lab



Institute Vision, Mission & Quality Policy

Vision

To foster and permeate higher and quality education with value added engineering, technology programs, providing all facilities in terms of technology and platforms for all round development with societal awareness and nurture the youth with international competencies and exemplary level of employability even under highly competitive environment so that they are innovative adaptable and capable of handling problems faced by our country and world at large.

Mission

The Institution is committed to mobilize the resources and equip itself with men and materials of excellence thereby ensuring that the Institution becomes pivotal center of service to Industry, academia, and society with the latest technology. RAIT engages different platforms such as technology enhancing Student Technical Societies, Cultural platforms, Sports excellence centers, Entrepreneurial Development Center and Societal Interaction Cell. To develop the college to become an autonomous Institution & deemed university at the earliest with facilities for advanced research and development programs on par with international standards. To invite international and reputed national Institutions and Universities to collaborate with our institution on the issues of common interest of teaching and learning sophistication.

Quality Policy

ज्ञानधीनं जगत् सर्वम।

Knowledge is supreme.

Our Quality Policy

It is our earnest endeavour to produce high quality engineering professionals who are innovative and inspiring, thought and action leaders, competent to solve problems faced by society, nation and world at large by striving towards very high standards in learning, teaching and training methodologies.

Our Motto: If it is not of quality, it is NOT RAIT!

Dr. Vijay D. Patil
President, RAES

Department Vision & Mission

Vision

To pervade higher and quality education with value added engineering, technology programs to deliver the IT graduates with knowledge, skills, tools and competencies necessary to understand and apply the technical knowledge and to become competent to practice engineering professionally and ethically in tomorrow's global environment. To contribute to the overall development by imparting moral, social and ethical values.

Mission

The mission of the IT department is to prepare students for overall development including employability, entrepreneurship and the ability to apply the technology to real life problems by educating them in the fundamental concepts, technical skills/programming skills, depth of knowledge and development of understanding in the field of Information Technology. To develop entrepreneurs, leaders and researchers with exemplary level of employability even under highly competitive environments with high ethical, social and moral values.

Index

Sr. No.	Contents	Page No.
1.	List of Experiments	iii
2.	Experiment Plan and Lab Outcomes	iv
3.	Study and Evaluation Scheme	v
4.	Experiment No. 1	1
5.	Experiment No. 2	18
6.	Experiment No. 3	28
7.	Experiment No. 4	33
8.	Experiment No. 5	38
9.	Experiment No. 6	42
10.	Experiment No. 7	46
11.	Experiment No. 8	51
12.	Experiment No. 9	55
13.	Experiment No. 10	61
14.	Experiment No. 11	70
15.	Experiment No. 12	77
16.	Experiment No. 13	86

List of Experiments

Sr. No.	Experiment Name
1	Study of PC Motherboard Technology (South Bridge and North Bridge).
2	Disassembling the System Unit & Identifying Internal Components and Connections.
3	Study of various connections and ports used in computer communication.
4	Write 8086 ALP for 16 bit BCD addition.
5	Write 8086 ALP to convert two digit Packed BCD to Unpacked BCD.
6	Write 8086 ALP to move set of numbers from one memory block to another.
7	Write 8086 ALP to count number of 1's and 0's in a given 8 bit number.
8	Write 8086 ALP to search for a given number.
9	Write 8086 ALP to check whether a given string is a palindrome or not.
10	Write 8086 ALP to calculate length of the string and reverse it.
11	Write 8086 ALP to compute the factorial of a positive integer 'n' using recursive procedure.
12	Interfacing Seven Segment Display.
13	Interfacing DAC.

Experiment Plan & Laboratory Outcome

Laboratory Outcomes:

LO1	Apply the fundamentals of assembly level programming of microprocessors.
LO2	Build a program on a microprocessor using arithmetic & logical instruction set of 8086.
LO3	Develop the assembly level programming using 8086 loop instruction set.
LO4	Write programs based on string and procedure for 8086 microprocessor.
LO5	Analyze abstract problems and apply a combination of hardware and software to address the problem
LO6	Make use of standard test and measurement equipment to evaluate digital interfaces.

Module No.	Week No.	Experiment Name	Laboratory Outcome
1	W1	Study of PC Motherboard Technology (South Bridge and North Bridge).	LO1
2	W2	Disassembling the System Unit & Identifying Internal Components and Connections.	
3	W3	Study of various connections and ports used in computer communication.	
4	W4	Write 8086 ALP for 16 bit BCD addition.	LO2 LO6
5	W5	Write 8086 ALP to convert two digit Packed BCD to Unpacked BCD.	
6	W6	Write 8086 ALP to move set of numbers from one memory block to another.	LO3 LO6
7	W7	Write 8086 ALP to count number of 1's and 0's in a given 8 bit number.	
8	W8	Write 8086 ALP to search for a given number.	
9	W9	Write 8086 ALP to check whether a given string is a palindrome or not.	LO4 LO6

10	W10	Write 8086 ALP to calculate length of the string and reverse it.	
11	W11	Write 8086 ALP to compute the factorial of a positive integer 'n' using recursive procedure.	LO4 LO6
12	W12	Interfacing Seven Segment Display.	LO5 LO6
13	W13	Interfacing DAC.	

Study and Evaluation Scheme

Course Code	Course Name	Teaching Scheme			Credits Assigned			
ITL403	Microprocessor Programming Lab	Theory	Practical	Tutorial	Theory	Term work / Practical	Tutorial	Total
			02	--		01	--	01

Course Code	Course Name	Examination Scheme		
ITL403	Microprocessor Programming Lab	Term Work	Oral	Total
		25	25	50

Term Work:

Term work assessment is based on following criteria:

Term Work shall consist of at least 10 to 12 practical's based on the above list. Also Term work Journal must include at least 2 assignments.

Term Work Marks: 25 Marks (Total marks) = 15 Marks (Experiment) + 5 Marks (Assignments) + 5Marks (Attendance)
The final certification and acceptance of TW ensures the satisfactory performance of Laboratory Work and Minimum Passing in the term work.

Oral:

1. Oral exam will be held based on the entire syllabus.

Microprocessor Programming Lab

Experiment No: 1

Aim: Study of PC Motherboard Technology (South Bridge and North Bridge)

Experiment No. 1

1. **Aim :** Study of PC Motherboard Technology (South Bridge and North Bridge).

2. **What you will learn by performing this experiment?**

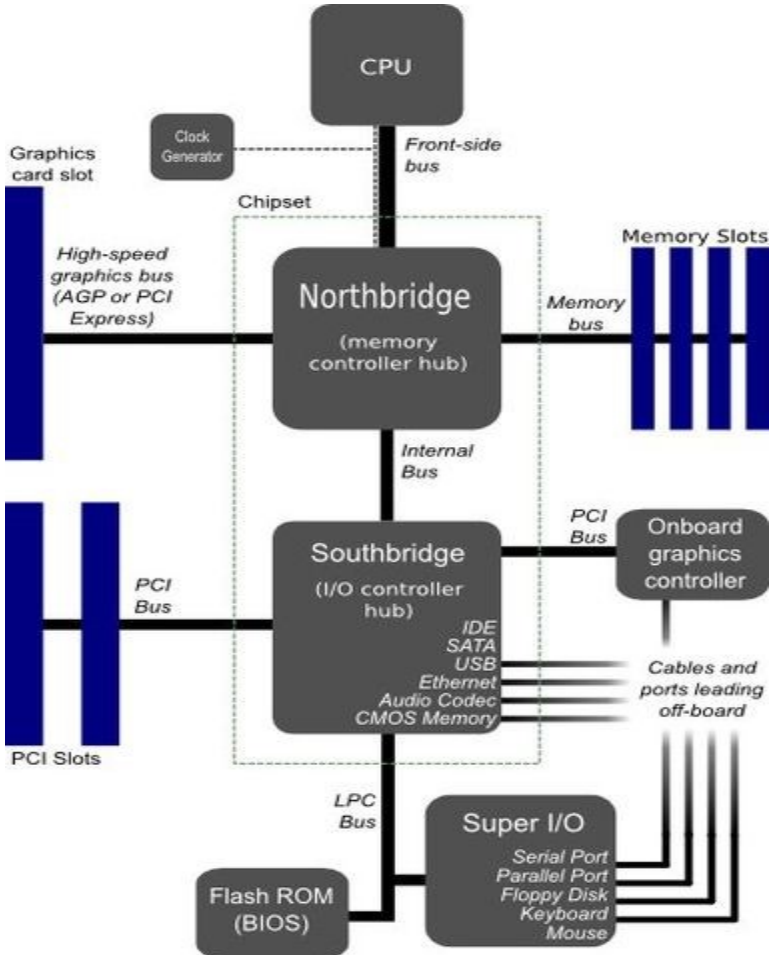
To learn what is PC motherboard, how it is use in computer and different components of ATX motherboard legend.

3. **Apparatus Required:** PC Motherboard

4. **Theory:**

Introduction to the motherboard

A motherboard is an electronic circuit board in a computer which interconnects hardware devices attached to it — which is to say, all of the system hardware. At a minimum it includes one or more Central Processing Units (CPU), and the main processing activity of the computer takes place on it. However, other connected printed circuit boards may contain their own pre-processing or post-processing CPUs, to take some of the load off of the motherboard; these, together with other plug-in boards without CPUs, may be called "daughter boards". It was called a "mother" board in relation to these. A PC motherboard generally has a series of slots, allowing daughter boards to be plugged in directly. Other connectors on the motherboard allow communication through cables with various peripheral devices, both inside and outside the computer case.



ATX motherboard legend

1. Processor socket
2. Chipset
3. RAM slots
4. AGP graphic card slot
5. PCI slots
6. CNR modem slot
7. Audio chip
8. I/O chip
9. BIOS
10. ATX power connector

11. Floppy drive connector

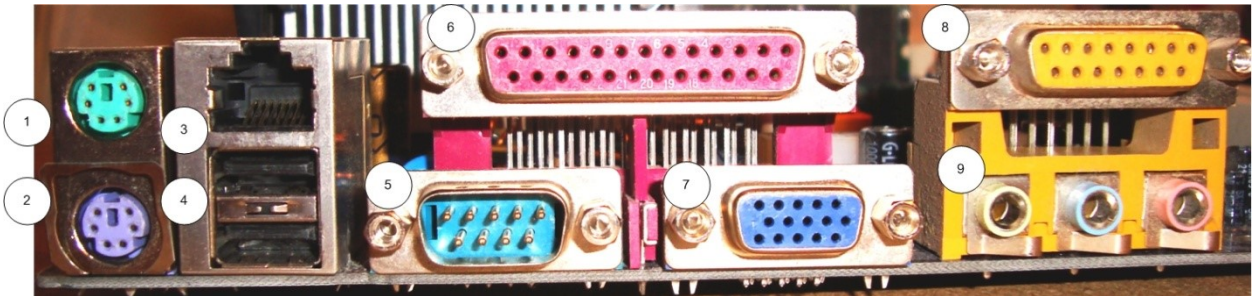
12. ATA connectors

13. Connectors for buttons, indicator lights etc.

The external ports (connections) are along the top edge (right hand side). A face-on view is shown below.

Ports

Ports are used by a motherboard to interface with electronics both inside and outside of the computer. Integrated ports are those that are part of, directly wired to, the motherboard. Internal integrated ports are used to connect devices inside the system unit. Externalports may be connected to the motherboard directly (integrated) or by circuit boards that are inserted into slots on the motherboard. It is often possible to add new external ports by inserting such a circuit board into an open slot. The external integrated ports are generally grouped together as shown below. Expansion card ports are arranged in a row of equal sized rectangular slots



Typical integrated motherboard ports (see top view above) 1. & 2. PS/2 (mouse, keyboard) 3. RJ-45 Ethernet 4. USB 5. Serial (terminal, modem) 6. parallel (printer) 7. VGA monitor 8. DA-15 (old networking) 9. Audio

PS/2

PS/2 ports were for connecting peripherals such as your mouse (1 above) and keyboard (2 above) to the computer, but are now outdated. PS/2 based mice and keyboards have now been replaced by USB ports as the popular standard. This trend for USB over PS/2 started in circa 2004.

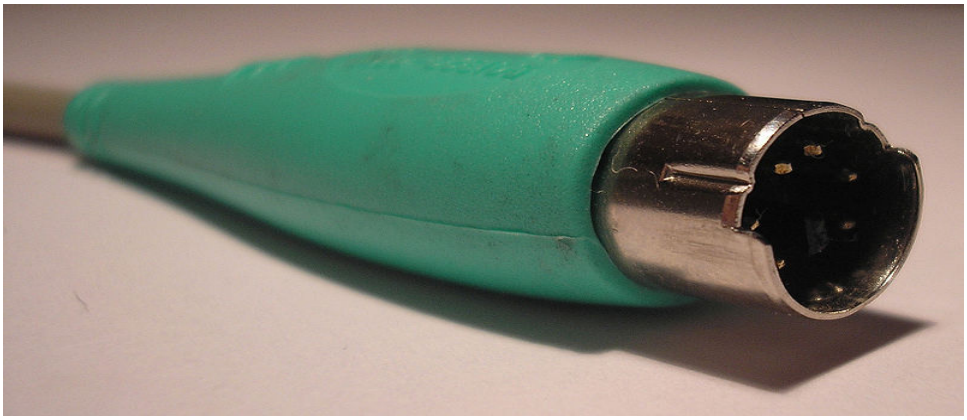


Figure: PS/2 connector

RJ-45

Wired Ethernet connection (3 above). Looks like a (bigger) telephone/modem jack. The cable itself is referred to by its category (e.g. CAT 5) and basic type, UTP (Unshielded Twisted Pair).

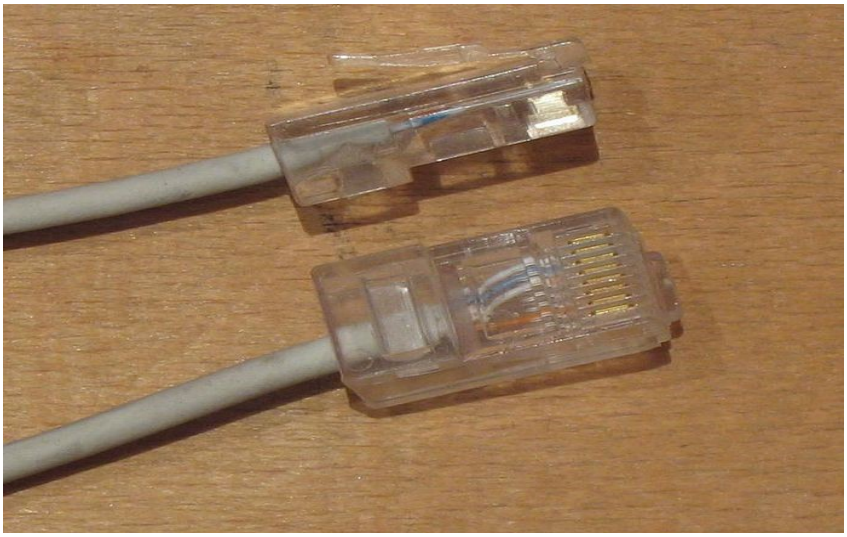
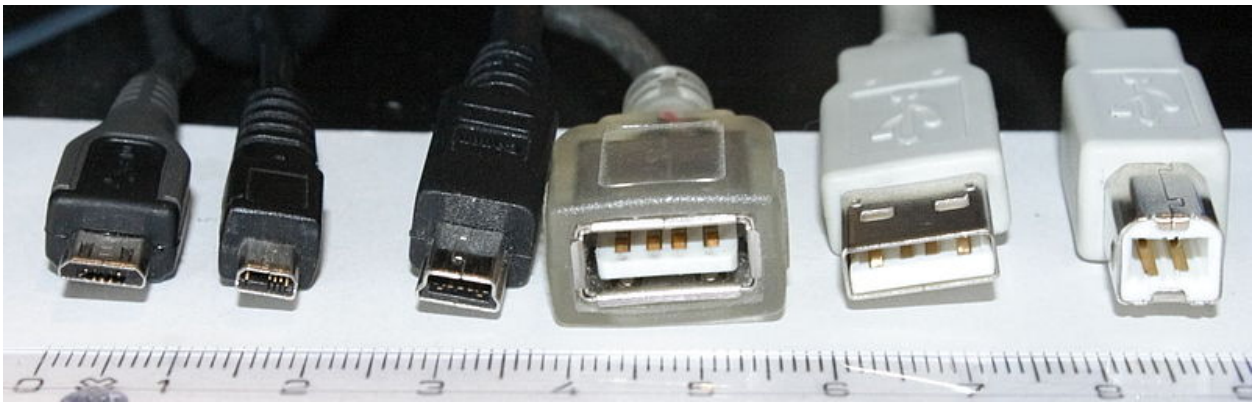


Figure: Ethernet cable

USB

USB (4 above), or Universal Serial Bus, is a connectivity specification, currently at version 3 (V3). They are very common today, connecting flash drives and many peripherals. Modern desktop systems have should have 4-8 on the back of the computer and at least two on the front. USB is one of the most successful interconnect in computing history. V1 operates at 1.5 Mbps (low speed) or 12 Mbps (full speed), V2 (high speed) at 480 Mbps, and V3 (super speed) at up to 5Gbps. It can be found in over 2 billion PC and mobile devices. USB has strong consumer brand recognition and a reputation for ease-ofuse. USB connectors are sometimes used to supply power, generally to recharge handheld devices like a smartphone.



Different USB connectors. From left to right: male Micro USB B-Type, proprietary (not USB), male Mini USB (5-pin) B-type, female A-type, male A-type, male B-type. Shown

with a centimeter ruler. Female A-type connector (4th from left) is "upside down" to show the pins

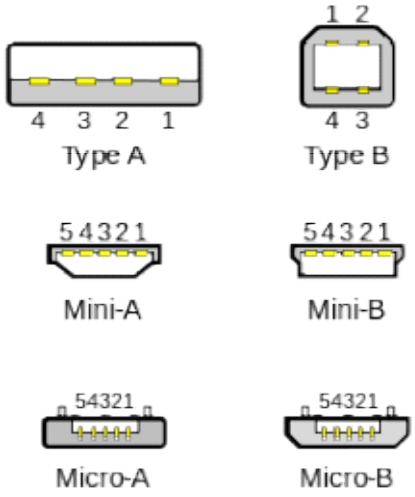


Figure: USB connector

What is a Computer Bus?

A computer bus carries data from one device to another. It's more like an electricity bus than the vehicle. In terms of vehicles, the (computer) bus would be the road being traveled, with the vehicles on the road being the data. Serial bussesallow one vehicle at a time, while parallel busses support multiple vehicles. Modern serial busses are faster because the data (vehicles) travel at a greater speed.

Serial

An outdated piece of technology, serial ports (5 above) were most often used to connect the mouse and keyboard. By circa 2000, most personal computers stopped relying on serial ports and were replaced by PS/2 and/or USB ports.

Figure: Serial connector

Parallel

Parallel ports (6 above) are used to connect other peripherals such as joysticks, and more commonly, printers. Similar to the serial port, this technology is slowly being phased out in favor of USB. Parallel ports can still be found in many motherboards today.

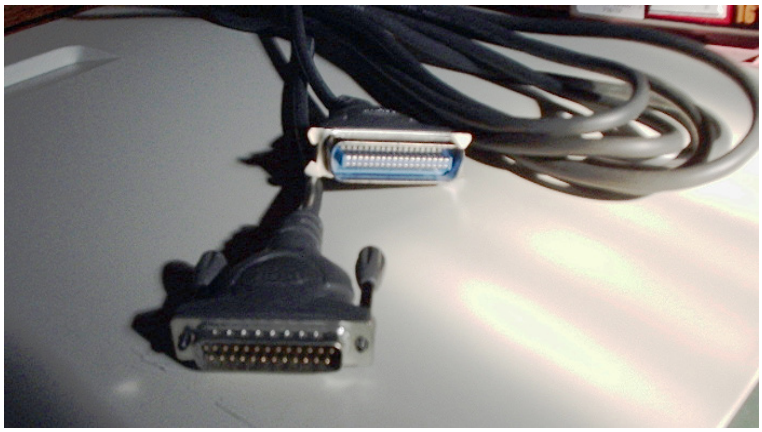


Figure: Parallel printer cable

VGA

A VGA (7 above), or Video Graphics Array, connector is used to connect a monitor or other video equipment. The same connector is sometimes used for high definition television and is sometimes called an RGB connector.

Figure: VGA cable for monitor

DA-15

The DA-15 port shown (8 above) has been used for network connectivity and for video output.

Audio

The audio input and stereo output ports (9 above) connect to external speakers, a microphone, head sets, and possibly a game. The external ports are color codedby industry standard.

Figure: Audio ports

Firewire

Technically known as the IEEE 1394 interface, but dubbed by Apple as Firewire (not shown above), this connection medium hoped to surpass USB in terms of speed and popularity. While it did outperform USB v2 in speed tests, uptake was very limited due to the existing

widespread use of USB. Firewire is the standard for high definition audio and video transfer and may be found on many digital camcorders. Also known by the brand names i.LINK and Lynx.

Figure: Firewire PCI Expansion Card

Figure:Firewire cables: 4 & 6 circuit

eSATA

eSATA (Serial Advanced Technology Attachment) is a computer bus interface for connecting to mass storage devices such as hard disk drives and optical drives inside the system box. eSATA is an external port for drives.

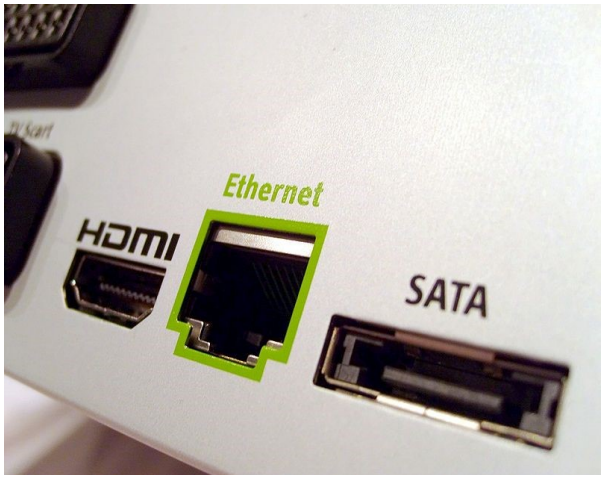


Figure: eSata Port

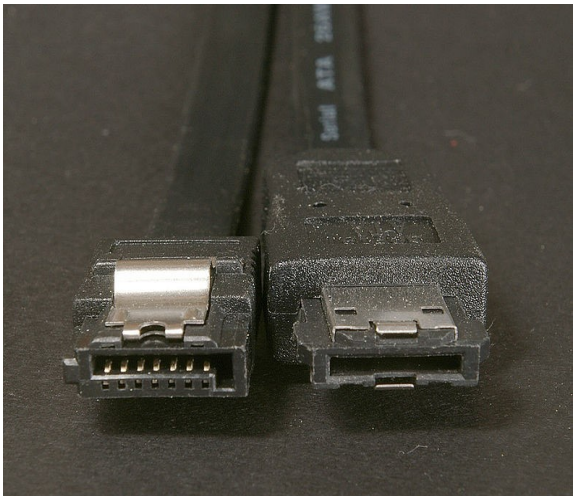


Figure:eSata Cable

Modem

For many years the telephone, or voice band, modem (not shown above) was the primary means of connecting desktop and laptop PCs to the Internet. Modems have not gone away, as broadband connections are not available in all areas. A standard telephone modem uses your existing analog telephone line at speeds up 56 Kbps. The speed is limited by the quality of the phone line connection — extraneous noise lowers the actual throughput.

Broadband connections (to be discussed in the section on networks) also use modems of a different type. They are still called modems because their essential function is the same — they convert digital signals to analog and modulate them on an analog medium when sending information, and reverse this process when receiving it. Hence, they MOdulate and DEModulate. Modem ports and connectors look a lot like Ethernet (RJ-45). A lot of networking was initially done with phone cables. Technically, they are called Category 3 UTP.

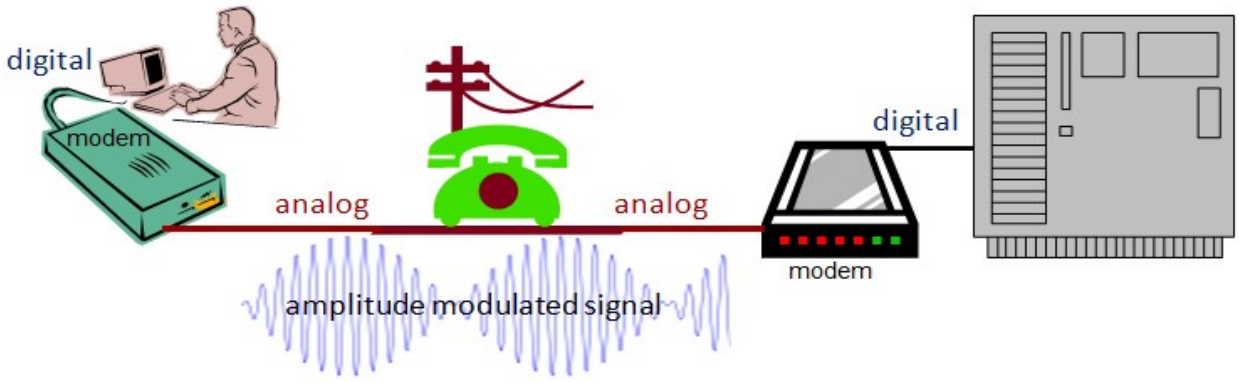


Figure: modem functions

SCSI

Pronounced "scuzzy", SCSI (not shown above) was used primarily as a connection interface for tape drives and hard disk drives. SCSI has been superseded in favor of newer and cheaper technologies such as USB and Firewire in the home market. SCSI provides a high-speed (up to 5 Gbps) bus interface that allows up to 8 or 16 devices to be attached. It is still used for high-end workstations and servers.

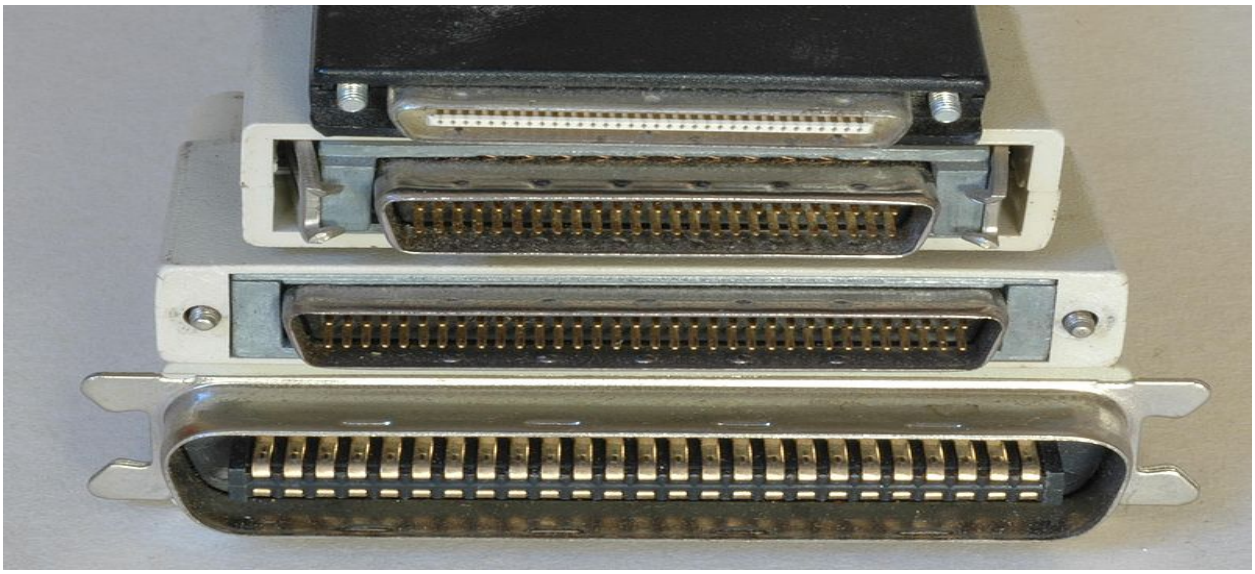


Figure: SCSI connectors

Slots

On the typical desktop PC, a number of slots are provided on the motherboard (see image above) for attaching devices. One use of these slots is to expand or add to the ports available for external devices, as noted above. Graphics, audio and networking may be incorporated on

the motherboard, but are often handled by (daughter) circuit boards added into slots. These are often pre-installed on your computer, but may be replaced (upgraded) as needed. Adding devices to laptop computers, after purchase, is problematic. Simply opening and closing the case can be a challenge and is not recommended for the novice, although manufactures usually provide fairly easy access to the RAM memory and hard disk. To allow for expansion through external devices, laptops often have an external slot (or port).

ExpressCard

The ExpressCard and slot (not shown above) is used primarily on laptop computers. It replaces the older PC Card (also called PCMCIA) shown at right. The ExpressCard comes in two sizes, although the ExpressCard/34 may be used in an ExpressCard/54 slot. Hardware that may be plugged into a computer via an ExpressCard includes connect cards, FireWire 800 (1394B), USB 3.0, 1Gb/sec Ethernet, Serial ATA external stick drives, solid-state drives, external enclosures for desktop size PCI Express graphics cards, wireless network interface cards, TV tuner cards, common access card (CAC) readers, and sound cards.[src] Other card slots may also be available, such a PC cards, smart cards, and secure digital cards.

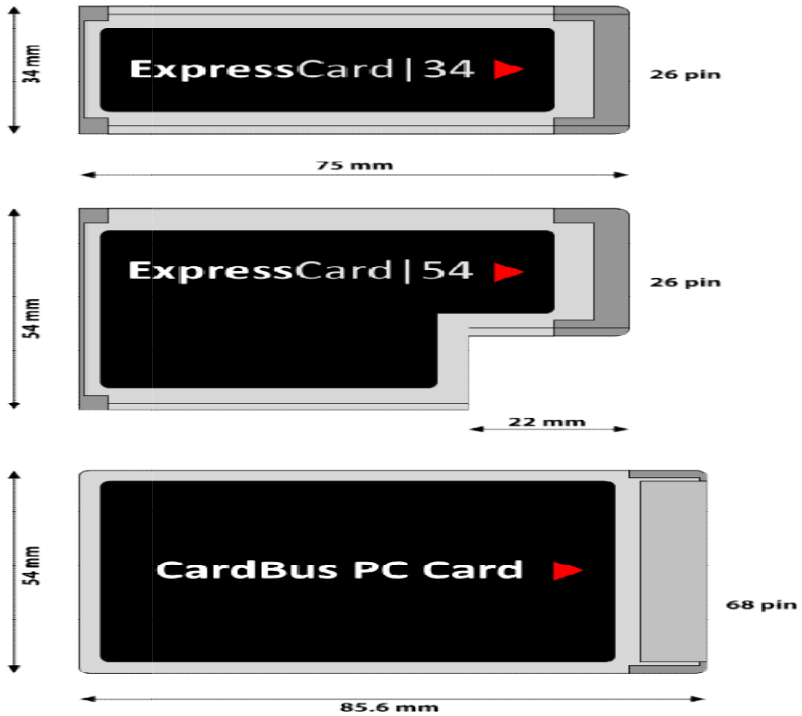


Figure: Comparison of Express Card and PC card

Graphics Card

Graphics cards are also called video cards or a video adapter. They are in all PCs, but may be integrated on the motherboard. Graphic cards generate output images that can be displayed on the monitor. While many graphics cards are built into the motherboard these days, enthusiasts will invest in stand-alone graphics cards with more powerful processing capabilities. This allows for heavy image editing, or better rendering and frame rates in computer games. Graphics cards are designed to offload rendering from the CPU. Graphics cards are powered by the motherboard and require a PCI-X or PCIe slot to install. Some cards require more power and thus will need a 6-8 pin connector that runs directly to the power supply. Graphics cards also include on-board memory for efficient rendering. Typical sizes include 128-1024MB of memory. Today, high end graphics cards have multiple core processors that are largely parallel to increase texture fill and process more 3D objects in real time.

Figure:Graphics Card Model

Sound Card

A sound card, also referred to as an audio card, facilitates the input and output of audio signals to and from a computer under the control of computer programs. Sound cards for computers were uncommon until 1988, which left the single internal PC speaker as the only way early PC software could produce sound and music. Uses of a sound card include the

audio component's for multimedia applications such as games, video/audio editing software and music composition. Most computers today have sound capabilities built into the motherboard, while others require additional expansion cards. The external ports are color coded by industry standard.

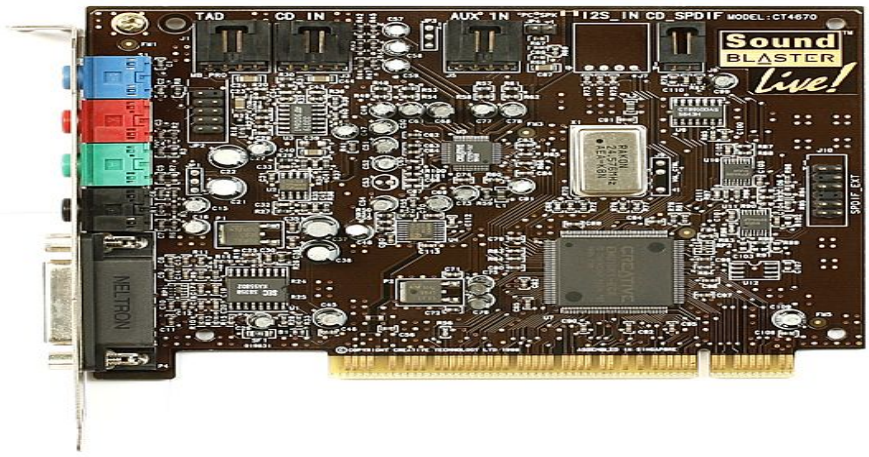


Figure: Creative Labs Soundblaster Live Gold edge at bottom is slotted into motherboard. External ports show in profile on left

Network Interface Card

A Network Interface Card (NIC), also called a network card, network adapter, orLAN Adapter is a piece of computer hardware designed to allow computers to communicate over a computer network. Used for remote communication via cable. Data is transmitted over a cable network. The NIC connects computers to the Internet and other devices, such as printers. Many modern motherboards have NICs built in by default. Most laptops also provide a wireless adapter or wireless network interface controller (WNIC). This device allows your laptop to communicate via a radio transceiver with a wireless (Wi-Fi) network. A WNIC can be added using a USB port or ExpressCard slot.

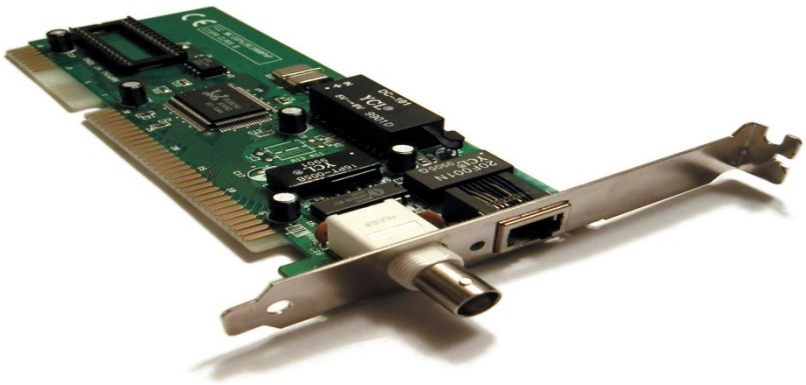


Figure: Network card with both BNC "Thinnet" (left) and Ethernet RJ-45 (right) connectors.

5. Conclusion

Hence We have studied *s*tudy of PC Motherboard Technology.

6. QUIZ / Viva Questions:

- What is use of motherboard?
- What is NIC?
- Explain Graphics Card in detail
- What is port?
- Explain Sound Card in detail
- Explain Express Card in detail
- What is SCSI?
- What is a Computer Bus?

7. References:

1. Scott Mueller,"Upgrading and repairing PCs", Pearson,

2. John Uffenbeck, "8086/8088 family: Design Programming and Interfacing:"Pearson Education

3. K Bhurchandi, "Advanced Microprocessors & Peripherals", Tata McGraw-Hill Education

Microprocessor Programming Lab

Experiment No: 2

Aim: Disassembling the System Unit & Identifying Internal Components and Connections

Experiment No. 2

1. **Aim:** Disassembling the System Unit & Identifying Internal Components and Connections.

2. **What you will learn by performing this experiment?**

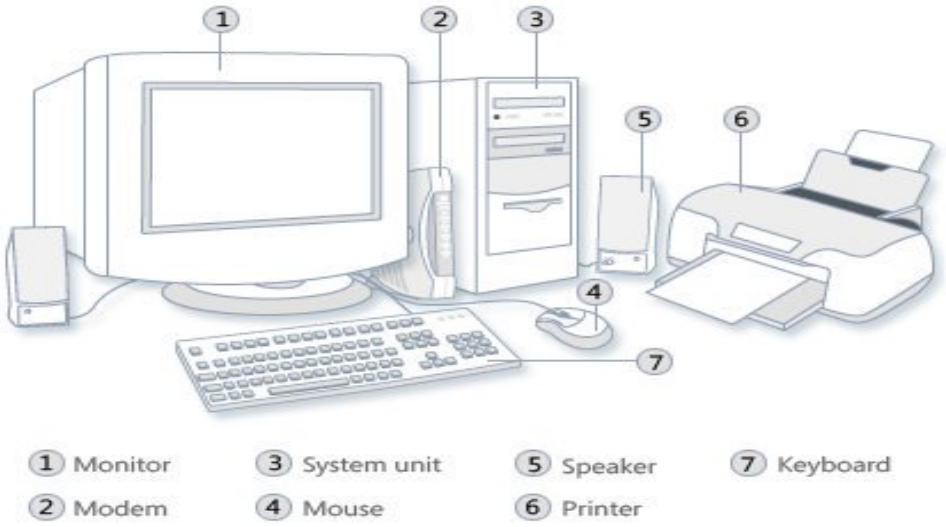
   To learn how to disassemble System Unit & Identify Internal Components and Connections.

3. **Apparatus Required:** Computer

4. **Theory:**

   Disassembling of System Unit & Identifying Internal Components and Connections.

Parts of Computer System



The computer system is made up of following external devices:

- CPU Cabinet
- Monitor
- Keyboard
- Mouse
- Printer/scanner [if attached]

Disassembling the computer system

Detach the power cable:

The disassembling of the computer system starts with externally connected device detachment. Make sure the computer system is turned off, if not then successfully shut down the system and then start detaching the external devices from the computer system. It includes removing the power cable from electricity switchboard, then remove the cable from SMPS (switch mode power supply) from the back of the CPU Cabinet. Do not start the disassembling without detaching the power cable from the computer system. Now remove the remaining external devices like keyboard, mouse, monitor, printer or scanner from the back of CPU cabinet.

Figure: switch off the power supply and detach power cable

Remove the Cover:

The standard way of removing tower cases used to be to undo the screws on the back of the case, slide the cover back about an inch and lift it off. The screwdrivers as per the type of screw are required to do the task.

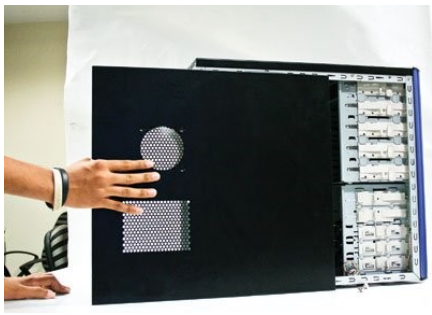


Figure: remove the CPU cabinet cover

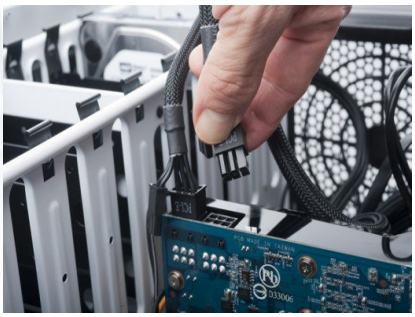


Figure: detach internal cables

Remove the adapter cards:

Make sure if the card has any cables or wires that might be attached and decide if it would be easier to remove them before or after you remove the card. Remove the screw if any, that holds the card in place. Grab the card by its edges, front and back, and gently rock it lengthwise to release it.

Figure: uninstall internal cards

Remove the drives:

Removing drives is easier. There can be possibly three types of drives present in your computer system, Hard disk drive, CD/DVD/Blue-ray drives, floppy disk drives (almost absolute now a day). They usually have a power connector and a data cable attached from the device to a controller card or a connector on the motherboard. CD/DVD/Blue Ray drive may have an analog cable connected to the sound card for direct audio output.

The power may be attached using one of two connectors, a Molex connector or a Berg connector for the drive. The Molex connector may require to be wiggled slightly from side to side and apply gentle pressure outwards. The Berg connector may just pull out or it may have a small tab which has to be lifted with a screwdriver.

Now Pull data cables off from the drive as well as motherboard connector. The hard disk drive and CD/DVD drives have two types of data cables. IDE and SATA cables. The IDE cables need better care while being removed as it may cause the damage to drive connector pins. Gently wiggle the cable sideways and remove it. The SATA cables can be removed easily by pressing the tab and pulling the connector straight back.

Now remove the screws and slide the drive out the back of the bay.

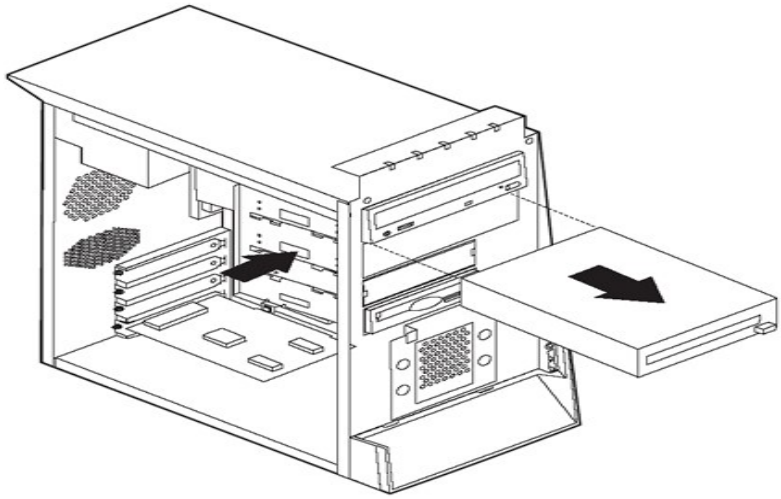


Figure: remove CD/DVD drives

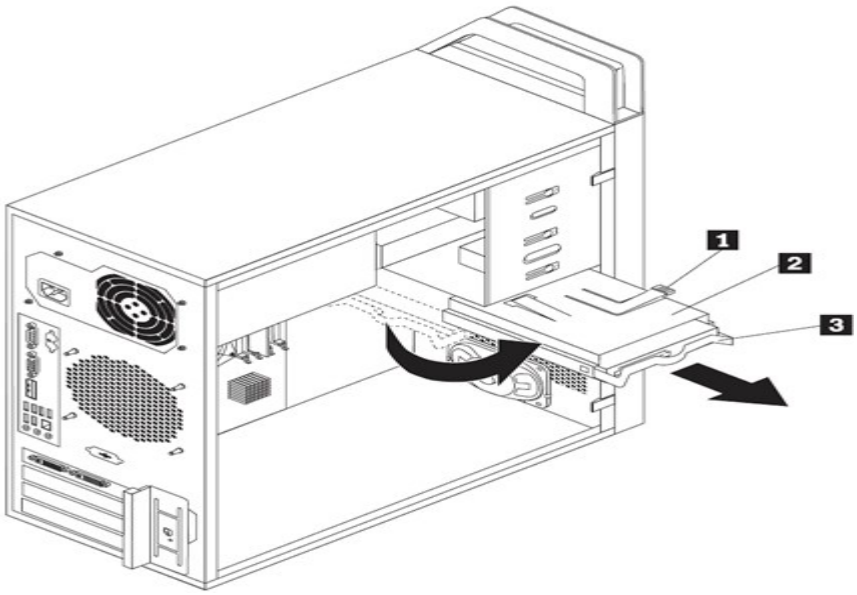


Figure: remove Hard Disk Drive

Remove the memory module:

Memory modules are mounted on the motherboard as the chips that can be damaged by manual force if applied improperly. Be careful and handle the chip only by the edges. SIMMs and DIMMs are removed in a different way:

- SIMM - gently push back the metal tabs while holding the SIMM chips in the socket. Tilt the SIMM chip away from the tabs until a 45% angle. It will now lift out of the socket. Put SIMM in a safe place.

- DIMM- There are plastic tabs on the end of the DIMM sockets. Press the tabs down and away from the socket. The DIMM will lift slightly. Now grab it by the edges and place it safely. Do not let the chips get dust at all.

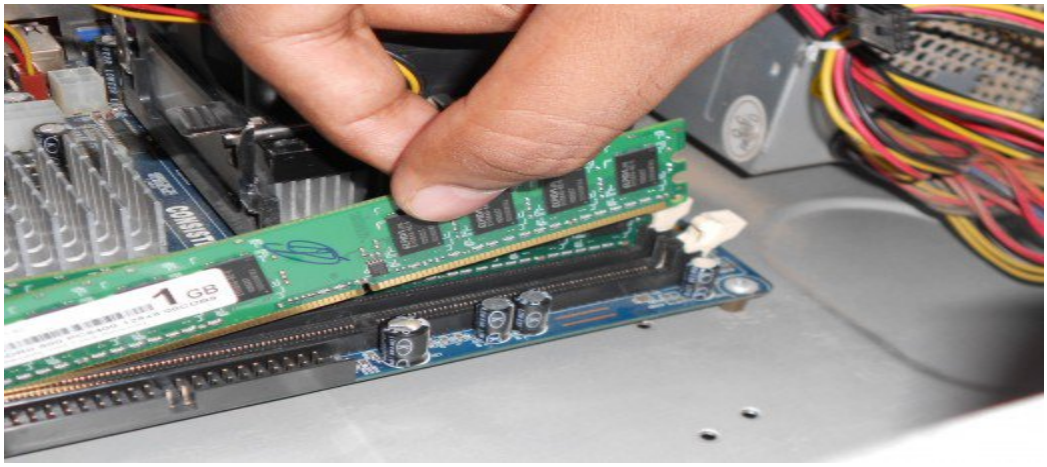


Figure: remove RAM

Remove the power supply:

The power supply is attached into tower cabinet at the top back end of the tower. Make sure the power connector is detached from the switchboard. Start removing the power connector connected to motherboard including CPU fan power connector, cabinet fan, the front panel of cabinet power buttons and all the remaining drives if not detached yet.

Now remove the screws of SMPS from the back of the cabinet and the SMPS can be detached from the tower cabinet.

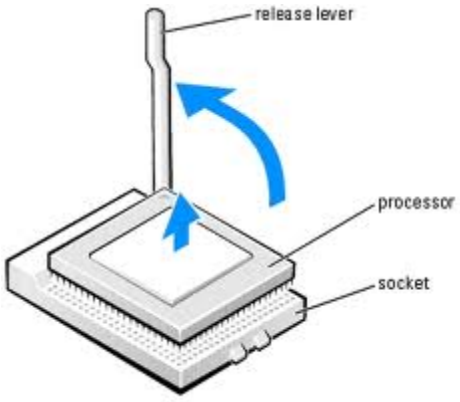


Figure: remove processor

Remove the motherboard:

Before removing all the connectors from the motherboard, make sure u memorize the connectors for assembling the computer if required, as that may require connecting the connectors at its place. Remove the screws from the back of the motherboard and you will be able to detach it from the cabinet. Now remove the CPU fan from the motherboard. The heat sink will be visible now which can be removed by the pulling the tab upward. Finally, the processor is visible now, which can be removed by the plastic tab which can be pulled back one stretching it side way**.**

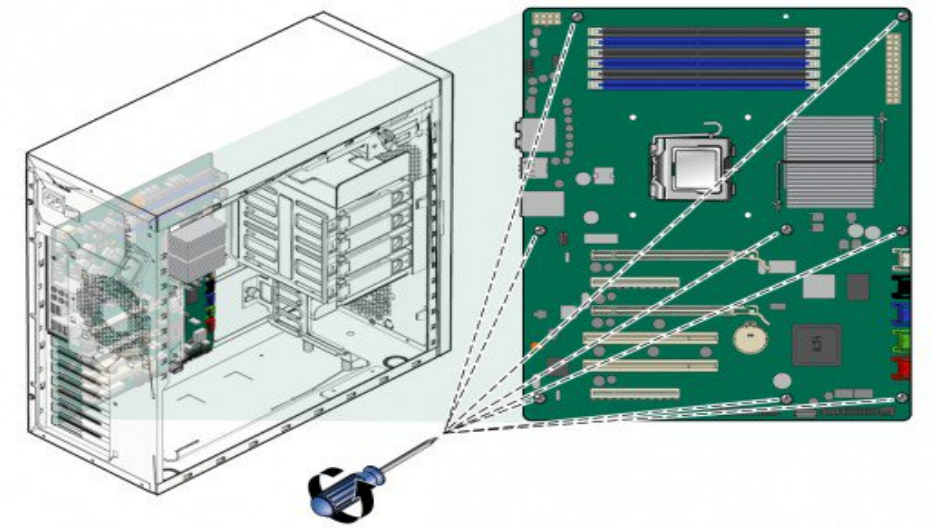


Figure: Remove the motherboard

Assembling the computer system

The assembling of the computer system is exactly the opposite of disassembling operation. Before starting assembling the computer system, make sure you have the screws and a screwdriver for those.

The first step for assembling the computer system starts with mounting the processor on the processor socket of the motherboard. To mount the process, you don't need to apply any force. The special ZIF (zero insertion force) sockets are usually used to prevent any damage to the processor pins. Once the processor is mounted, the heat sink will be attached on top of the processor. The CPU fan is also attached on top of the heat sink.

Now the motherboard is to be fixed vertically in the tower case and the screws are fixed from behind of the motherboard.

Now line up the power supply at the top back end of the cabinet and screw it. The power connectors for motherboard power supply and CPU fan power supply are to be connected. If the cabinet cooling FAN is required then it is to be screwed at the back end grill of the cabinet and its power connector is to be connected from SMPS.

Install the CD/DVD drives at the top front end of the cabinet and screw it. Install the Hard disk drive and floppy disk drive below CD/DVD drive and screw it. Make sure once screwed there is no vibration in either of the CD/DVD, Hard disk or Floppy disk drives.

Now select the appropriate data cable and connect one end of the cable to its drive socket and another end at its appropriate connector on the motherboard. For SATA hard disk drive or CD/DVD drives use SATA cable and its power cable, else use IDE data cable. Do the proper jumper settings as per the usage requirement.

It is time now to mount the memory modules on the motherboard by aligning the RAM to its socket on the motherboard and press it downward. Make sure the side tab are fixed into the RAM notch. If not, you may still have to press a bit.

Install the internal cards to its socket and attach the cables or power cable to it. The selection of right socket or slot is required as per the type of socket.

Cover the tower by placing it and pressing towards front side and screw it.

Connect the external devices with CPU at its appropriate socket. It includes mouse and keyboard at PS2 or USB connectors. Monitor at the video output socket. Connect the power cable to the back of tower in SMPS. Plug in the power cable to the electric board.

5. Conclusion:

Hence We have studied Disassembling the System Unit & Identifying Internal Components and Connections.

6. QUIZ / Viva Questions:

- What is motherboard?
- How to disassemble the system Unit? List the steps.
- What are the external devices of computer system?

7. References:

1. Scott Mueller,"Upgrading and repairing PCs", Pearson,
2. John Uffenbeck, "8086/8088 family: Design Programming and Interfacing:"Pearson Education
3.K Bhurchandi, "Advanced Microprocessors & Peripherals", Tata McGraw-Hill Education

Microprocessor Programming Lab

Experiment No. : 3

Study of various connections and ports used in computer communication

Experiment No. 3

1. **Aim:** Study of various connections and ports used in computer communication.

2. **What you will learn by performing this experiment?**

   Students can understand what are the different ports are available fin computer for communication in the system.

3. **Apparatus Required:** Computer and ports

4. **Theory:**

A port is a physical docking point using which an external device can be connected to the computer. It can also be programmatic docking point through which information flows from a program to the computer or over the Internet.

Characteristics of Ports

A port has the following characteristics −

- External devices are connected to a computer using cables and ports.
- Ports are slots on the motherboard into which a cable of external device is plugged in.
- Examples of external devices attached via ports are the mouse, keyboard, monitor, microphone, speakers, etc.

Let us now discuss a few important types of ports −

1. Serial Port

- Used for external modems and older computer mouse
- Two versions: 9 pin, 25 pin model
- Data travels at 115 kilobits per second

2. Parallel Port

- Used for scanners and printers
- Also called printer port
- 25 pin model
- IEEE 1284-compliant Centronics port

3. PS/2 Port

- Used for old computer keyboard and mouse
- Also called mouse port
- Most of the old computers provide two PS/2 port, each for the mouse and keyboard
- IEEE 1284-compliant Centronics port

4. Universal Serial Bus (or USB) Port

- It can connect all kinds of external USB devices such as external hard disk, printer, scanner, mouse, keyboard, etc.
- It was introduced in 1997.
- Most of the computers provide two USB ports as minimum.
- Data travels at 12 megabits per seconds.
- USB compliant devices can get power from a USB port.

5. VGA Port

- Connects monitor to a computer's video card.
- It has 15 holes.
- Similar to the serial port connector. However, serial port connector has pins, VGA port has holes.

6. Power Connector

- Three-pronged plug.
- Connects to the computer's power cable that plugs into a power bar or wall socket.

7. Firewire Port

- Transfers large amount of data at very fast speed.
- Connects camcorders and video equipment to the computer.
- Data travels at 400 to 800 megabits per seconds.
- Invented by Apple.
- It has three variants: 4-Pin FireWire 400 connector, 6-Pin FireWire 400 connector, and 9-Pin FireWire 800 connector.

8. Modem Port

- Connects a PC's modem to the telephone network.

9. Ethernet Port

- Connects to a network and high speed Internet.
- Connects the network cable to a computer.
- This port resides on an Ethernet Card.
- Data travels at 10 megabits to 1000 megabits per seconds depending upon the network bandwidth.

10. Game Port

- Connect a joystick to a PC
- Now replaced by USB

11. Digital Video Interface, DVI port

- Connects Flat panel LCD monitor to the computer's high-end video graphic cards.
- Very popular among video card manufacturers.

12. Sockets

- Sockets connect the microphone and speakers to the sound card of the computer.

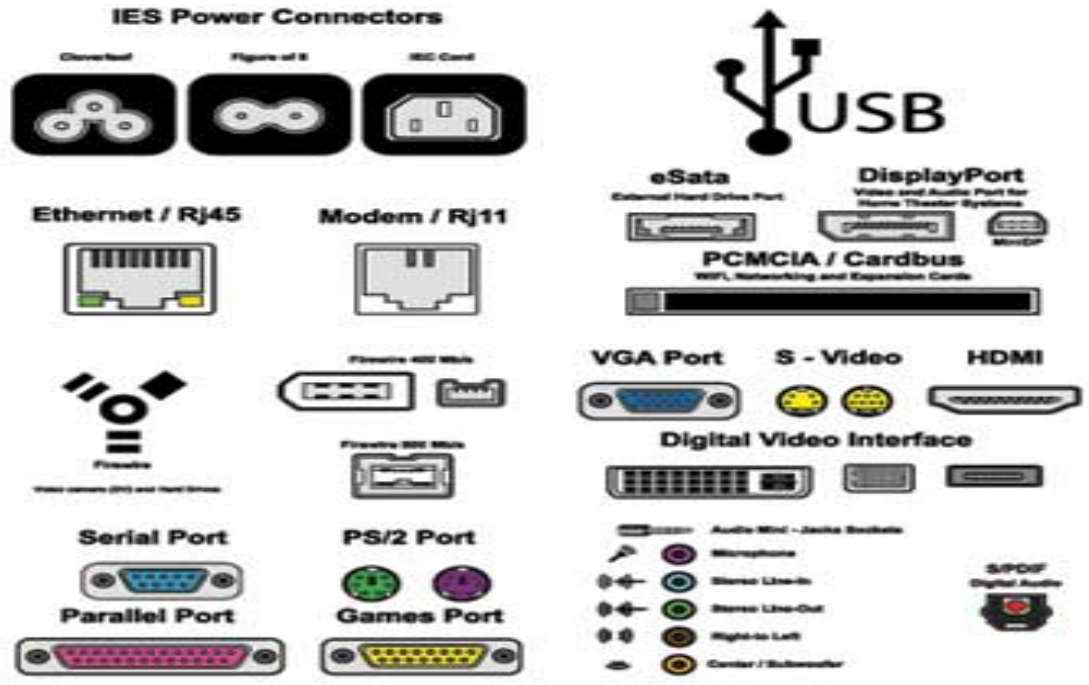


Figure : Different Ports and Their Connections

5. Conclusion: Hence we studied different ports and their connection used in computer communication**.**

6. QUIZ / Viva Questions:

- What is port?
- Differentiate between Serial Port and Parallel Port.\
- What is PS/2?
- What use of is VGA, HDMI

7. References:

1. Scott Mueller,"Upgrading and repairing PCs", Pearson,
2. John Uffenbeck, "8086/8088 family: Design Programming and Interfacing:"Pearson Education
3. K Bhurchandi, "Advanced Microprocessors & Peripherals", Tata McGraw-Hill Education

Microprocessor Programming Lab

Experiment No. : 4

Aim : Write 8086 ALP for 16 bit BCD addition

Experiment No. 4

1. **Aim:** Write 8086 ALP for 16 bit BCD addition.

2. What you will learn by **performing this experiment?**

   Students will learn what BCD number system is and understand how to do addition of BCD using DAA instruction.

3. **Apparatus Required:** Emulator 8086, TASM 1.4 for 64 bits

4. **Theory:**

   BCD (Binary Coded Decimal)

   BCD, or binary-coded decimal, represents the 10 decimal digits in terms of binary numbers.

   BCD Number Representation

   The decimal digits 0 through 9 are represented by the 4-bit binary strings 0000 through 1001.

   Just as in conventional decimal addition, BCD addition is performed one decimal digit at a time. The question is, what happens when the sum exceeds what can be represented in 4 bits? Stated differently, what are the conditions under which a carry is generated to the next highest-order BCD digit?

   For example, let's consider the addition of the two BCD digits 5 and 3:

```
5 =   0101
3 =   0011
      1000 = 3
```

   Now consider the sum of 5 and 8:

```
5 =   0101
8 =   1000
      1101 = 13!
```

   The sum is 11012 = 13, but this result should be correctly represented as 0001 0011 in BCD notation. Fortunately, there is a simple way to find the correct result. We add 6 (0110) to the digit sum if it exceeds 9. Let's examine the following cases:

```
5 =  0101                    9 =  1001
3 =  1000                    7 =  0111
     1101 = 13 in decimal        10000 = 16 in decimal
   + 0110                      + 0110
    10011 = 13 in BCD           10110 = 16 in BCD
```

In both cases, by adding six we obtain the correct answer in BCD.

DAA: Decimal Adjust AL after Addition

Adjusts the sum of two packed BCD values to create a packed BCD result. The AL register is the implied source and destination operand. The DAA instruction is only useful when it follows an ADD instruction that adds (binary addi-tion) two 2-digit, packed BCD values and stores a byte result in the AL register. The DAA instruction then adjusts the contents of the AL register to contain the correct 2-digit, packed BCD result. If a decimal carry is detected, the CF and AF flags are set accordingly.

5. Algorithm:

1. Start
2. Declare the variables of word size a,b,c,d
3. Initialize code and data segment
4. Move value of a to AX register
5. Move value of b to BX register
6. Add with carry AX, BX and store the result into AX
7. Adjust the decimal in AL after addition to create a packed BCD result
8. Move value of AL to CL register
9. Move value of AH to AL register
10. Add with carry AL and 00 and store the result into AL
11. Adjust the decimal in AL after addition to create a packed BCD result
12. Move value of AL to CH register

13. Move 00 to AX register
14. Add with carry AX and 00 and store the result into AX
15. Move value of CX to C variable.
16. Move value of AX to D variable.
17. Stop

6. Program:

```
data segment
    a dw 9102h
    b dw 1109h
    c dw ?
    d dw ?
data ends
code segment
    assume cs:code, ds:data
    start:
    mov ax,data
    mov ds,ax
    mov ax,a
    mov bx,b
    adc ax,bx
    daa
    mov cl,al
    mov al,ah
    adc al,00
    daa
    mov ch,al
    mov ax,00
```

```
    adc ax,00
    mov c,cx
    mov d,ax
    int 3h
code ends
end start
```

7. Conclusion:

Hence we studied ALP for 16 bit BCD addition.

8. QUIZ / Viva Questions:

- What is BCD?
- What is ALP?
- Differentiate between Compiler, Interpreter and Assembler.
- What is Assembler Directives? List all in this program.
- Explain working of DAA.

9. References:

1. Scott Mueller,"Upgrading and repairing PCs", Pearson,
2. John Uffenbeck, "8086/8088 family: Design Programming and Interfacing:"Pearson Education
3. K Bhurchandi, "Advanced Microprocessors & Peripherals", Tata McGraw-Hill Education

Microprocessor Programming Lab

Experiment No. : 5

Aim : Write 8086 ALP to convert two digit Packed BCD to Unpacked BCD

Experiment No. 5

1. **Aim:** Write 8086 ALP to convert two digit Packed BCD to Unpacked BCD.
2. **What you will learn by performing this experiment?**

   How the packed BCD numbers are converted into unpacked BCD number.
3. **Apparatus Required:**

   Emulator 8086, TASM 1.4 for 64 bits
4. **Theory:**

   BCD (Binary Coded Decimal)

   BCD, or binary-coded decimal, represents the 10 decimal digits in terms of binary numbers.

   BCD Number Representation

   The decimal digits 0 through 9 are represented by the 4-bit binary strings 0000 through 1001.

   To convert packed BCD number , BCD number is logically ANDed with 0f and fo so that BCD number is unpacked.

   Example: if variable declared is a=32

   Then 32 ANDed with OFH then we will get 02 or if it is ANDed with FOH then it will get 30 then after shifting the number 4 times right we will get the 03.

   So it is called packed BCD number to unpacked BCD number. So if it is 16 bit number.

5. **Algorithm:**

   1. Initialize the data memory.
   2. Load number into register AL.
   3. Mask the lower nibble.
   4. Rotate 4 times left to make MSB digit = LSB.
   5. Display the digit.
   6. Load number in AL.

7. Mask upper nibble.
8. Display the result.
9. Stop.

6. Program:

```
DATA SEGMENT
MSG1 DB "ENTER NUMBER : $"
DIGIT1 DB ?
DIGIT2 DB ?
UD1 DB ?
UD2 DB ?
DATA ENDS
CODE SEGMENT
ASSUME DS:DATA,CS:CODE
START:
MOV AX,DATA
MOV DS,AX
LEA DX,MSG1
MOV AH,9
INT 21H
MOV AH,1
INT 21H
SUB AL,30H
MOV DIGIT1,AL
MOV AH,1
INT 21H
SUB AL,30H
```

```
MOV DIGIT2,AL
MOV AH,DIGIT1
MOV AL,DIGIT2
MOV CL,4
ROL AH,CL
ADD AL,AH
MOV AH,0
MOV BL,AL
AND AL,0F0H
AND BL,0FH
MOV UD2, BL
MOV CL,04H
ROL AL,CL
MOV UD1,AL
CODE ENDS
END START
```

7. **Conclusion**: Hence we have studied ALP to convert packed BCD to unpack the BCD number.
8. **QUIZ / Viva Questions:**
   - What is packed BCD?
   - What is unpacked BCD?
9. **References:**

1. Scott Mueller,"Upgrading and repairing PCs", Pearson,
2. John Uffenbeck, "8086/8088 family: Design Programming and Interfacing:"Pearson Education
3. K Bhurchandi, "Advanced Microprocessors & Peripherals", Tata McGraw-Hill Education

Microprocessor Programming Lab

Experiment No. : 6

Write 8086 ALP to move set of numbers from one memory block to another

Experiment No. 6

1. **Aim:** Write 8086 ALP to move set of numbers from one memory block to another.
2. **What you will learn by performing this experiment?**

   To learn how to move set of numbers from one memory block to another.
3. **Apparatus Required:**

   Emulator 8086, TASM 1.4 for 64 bits
4. **Theory:**

   Move set of numbers from one memory block to another can be done using two ways:

   - Nonoverlap memory transfer
   - Overlap memory transfer

   Program for non-overlapped and overlapped block transfer of array elements. Takes the array elements from the user and also the number of elements to be overlapped in overlapped transfer. Block transfer here, refers to moving of block of data within the memory to a different location. In non-overlapped transfer we move the data to a completely new location. It is easily accomplished by copying the data using two pointers, one data byte at a time. In overlapped transfer the data block is shifted slightly from the present position, thus, some of the starting elements may overlap with the old position of the last elements in the array. They are therefore copied in reverse order.
5. **Algorithm:**

   1. Initialize the data memory with array block and count with array size.
   2. For Nonoverlap memory transfer

   a.) Load effective address of array block to SI

   b.) Load effective address of array block+5 to DI

   c.) Move size of array to CX

   d.) Move data pointed by SI memory location to AL

   e.) Move data of AL memory location to pointed by DI

   f.) Increment SI

   g.) Increment DI

h.) Repeat from d till CX become 0

3. For Overlap memory transfer

a.) Load effective address of array block+4 to SI

b.) Load effective address of array block1+7 to DI

c.) Move size of array to CX

d.) Move data pointed by SI memory location to AL

e.) Move data of AL memory location to pointed by DI

f.) Decrement SI

g.) Decrement DI

h.) Repeat from d till CX become 0

4. Stop.

6. Program:

```
data segment
    block db 10h,20h,30h,40h,50h
   count dw 0005h
data ends

code segment
   assume cs:code,ds:data
   start:
   mov ax,data
   mov ds,ax
; Nonoverlap Block transfer
   lea si,block
   lea di,block+5
   mov cx,count
l1: mov al,[si]
   mov [di],al
   inc si
```

```
    inc di
    loop l1

Overlap Block transfer
    lea si,block1+4
    lea di,block1+7
    mov cx,count
l2: mov al,[si]
    mov [di],al
    dec si
    dec di
    loop l2
code ends
end start
```

7. **Conclusion**: Hence we have studied ALP to move set of numbers from one memory block to another.
8. **QUIZ / Viva Questions:**
   - What is Nonoverlap memory transfer?
   - What is Overlap memory transfer?
9. **References:**

1. Scott Mueller,"Upgrading and repairing PCs", Pearson,
2. John Uffenbeck, "8086/8088 family: Design Programming and Interfacing:"Pearson Education
3. K Bhurchandi, "Advanced Microprocessors & Peripherals", Tata McGraw-Hill Education

Microprocessor Programming Lab

Experiment No. : 7

Write 8086 ALP to count number of 1's and 0's in a given 8 bit number

Experiment No. 7

1. **Aim:** Write 8086 ALP to count number of 1's and 0's in a given 8 bit number.
2. **What you will learn by performing this experiment?**

   In this experiment Student will come to know how to use the control transfer instructions. Using conditional and unconditional instructions we can check number of one and zeros in the give binary number.

3. **Apparatus Required:**

   Emulator 8086, TASM 1.4 for 64 bits

4. **Theory:**

   The conditional jumps that are listed as pairs are actually the same instruction. The assembler provides the alternate mnemonics for greater clarity within a program listing. Conditional jump instructions contain a displacement which is added to the EIP register if the condition is true. The displacement may be a byte, a word, or a double-word. The displacement is signed; therefore, it can be used to jump forward or backward.

Mnemonic	Condition Tested	"Jump If..."
JA/JNBE	(CF or ZF) = 0	above/not below nor equal
JAE/JNB	CF = 0	above or equal/not below
JB/JNAE	CF = 1	below/not above nor equal
JBE/JNA	(CF or ZF) = 1	below or equal/not above
JC	CF = 1	carry
JE/JZ	ZF = 1	equal/zero
JNC	CF = 0	not carry
JNE/JNZ	ZF = 0	not equal/not zero
JNP/JPO	PF = 0	not parity/parity odd
JP/JPE	PF = 1	parity/parity even

Signed Conditional Transfers

Mnemonic	Condition Tested	"Jump If..."
JG/JNLE	((SF xor OF) or ZF) = 0	greater/not less nor equal
JGE/JNL	(SF xor OF) = 0	greater or equal/not less
JL/JNGE	(SF xor OF) = 1	less/not greater nor equal
JLE/JNG	((SF xor OF) or ZF) = 1	less or equal/not greater

JNO	OF = 0	not overflow
JNS	SF = 0	not sign (positive, including 0)
JO	OF = 1	overflow
JS	SF = 1	sign (negative)

5. Algorithm:

1. Start
2. Declare the variable to hold 16 bit number.
3. Load register AX *(Accumulator)* with the content of memory location
4. Set a counter to loop through each bits of number, Say CX=10H
5. Set a counter to store the number of one's (1's) in the byte, Say D
6. Set a counter to store the number of zero's (0's) in the byte Say B.
7. Rotate the content of accumulator to left through carry
8. Move value of AL to CL register If no carry found then increase the counter Z by 1
9. Else increase the counter D by one *(if carry found)*
10. Decrease counter CX by one
11. Repeat till counter CX becomes 0
12. Terminate the program. *(When C is 0)*
13. Stop

6. Program:

```
DATA SEGMENT
NO DW 5648H
Z DW ?
O DW ?
DATA ENDS

CODE SEGMENT
ASSUME CS:CODE, DS:DATA
START:
MOV AX, DATA
MOV DS, AX
MOV AX, NO
MOV BX, 00H
```

```
MOV CX, 10H
MOV DX, 00H

UP:
ROL AX,1
JC ONE
INC BX
JMP NXT

ONE:
INC DX

NXT:
DEC CX
JNZ UP

MOV Z, BX
MOV O, DX

INT 3
CODE ENDS
END START
```

7. **Conclusion:**

   Hence we have studied 8086 ALP to count number of 1's and 0's in a given 8 bit number.

8. **QUIZ / Viva Questions:**

- What is JC and JMP instruction ?
- Applying same logic wap to check one's and zero's for 16 bit number.
- Explain ROL instruction.
- Differentiate conditional and unconditional statements
- Differentiate RCR and ROL

9. References:

1. Scott Mueller,"Upgrading and repairing PCs", Pearson,
2. John Uffenbeck, "8086/8088 family: Design Programming and Interfacing:"Pearson Education
3. K Bhurchandi, "Advanced Microprocessors & Peripherals", Tata McGraw-Hill Education

Microprocessor Programming Lab

Experiment No. : 8

Write 8086 ALP to search for a given number

Experiment No. 8

1. **Aim:** Write 8086 ALP to search for a given number.
2. **What you will learn by performing this experiment?**

   Student will learn how to search for a given number.
3. **Apparatus Required:**

   Emulator 8086, TASM 1.4 for 64 bits

4. **Theory:**

Declare an array

To declare an array, specify the name of array, the dimension of array, the size of every element and the special system words DUP.

Size of element can be DB(for byte) or DW (for word that means 2 bytes)

```
A    DB      1,2,3,4,5,6,7,8,9,10
B    DB      DUP (?)
```

A is an byte array with initial values 1,2,3,4,5,6,7,8,9,10. B is an array without initial values.

Access array elements

This part is easy similar to other language, you need the right index and then access element.

That's means if want to access the first element of the array A, just write: A[0].

5. **Algorithm:**
   1. Start
   2. Declare array to hold the numbers.
   3. Declare variable to hold the number to be searched
   4. Take number to be searched in AL
   5. Set Load the effective address of array in SI
   6. Set a counter to loop through each of number of array, Say CX=04H.
   7. Move the first element of array in BL.
   8. Compare Al and BL contents.

9. If numbers are equal or zero flag is set then msg is “FOUND
10. If zero flag is not set then take the next element of array by increment SI
11. Decrement the counter
12. Compare numbers
13. Do this till counter is not equal to zero.
14. Terminate the program. *(When C is 0)*

6. Program:

```
DATA SEGMENT
STRING1 DB 11H,22H,33H,44H,55H
MSG1 DB "FOUND$"
MSG2 DB "NOT FOUND$"
SE DB 33H
DATA ENDS

PRINT MACRO MSG
MOV AH, 09H
LEA DX, MSG
INT 21H
INT 3
ENDM

CODE SEGMENT
ASSUME CS:CODE, DS:DATA
START:
MOV AX, DATA
MOV DS, AX
MOV AL, SE
LEA SI, STRING1
MOV CX, 04H

UP:
MOV BL,[SI]
CMP AL, BL
```

```
JZ FO
INC SI
DEC CX
JNZ UP
PRINT MSG2
JMP END1

FO:
PRINT MSG1

END1:
INT 3
CODE ENDS
END START
```

7. **Conclusion**: Hence we have studied 8086 ALP to search for a given number
8. **QUIZ / Viva Questions:**
   - What is array ?
   - How to access array elements.

9. **References:**

1. Scott Mueller,"Upgrading and repairing PCs", Pearson,
2. John Uffenbeck, "8086/8088 family: Design Programming and Interfacing:"Pearson Education
3. K Bhurchandi, "Advanced Microprocessors & Peripherals", Tata McGraw-Hill Education

Microprocessor Programming Lab

Experiment No. : 9

Write 8086 ALP to check whether a given string is a palindrome or not

Experiment No. 9

1. **Aim:** Write 8086 ALP to check whether a given string is a palindrome or not.
2. **What you will learn by performing this experiment?**

   Student will learn how to check whether a given string is a palindrome or not.
3. **Apparatus Required:**

   Emulator 8086, TASM 1.4 for 64 bits
4. **Theory:**

   CMPSB: Compares byte at address DS:SI with byte at address ES:DI and sets the status flags accordingly.

   Compares the byte, word, or double word specified with the first source operand with the byte, word, or double word specified with the second source operand and sets the status flags in the FLAGS register according to the results. Both the source operands are located in memory. The address of the first source operand is read from either the DS: registers (depending on the address-size attribute of the instruction respectively). The address of the second source operand is read from either the ES:DI registers (again depending on the address-size attribute of the instruction). The DS segment may be overridden with a segment override prefix, but the ES segment cannot be overridden.

   REPE chain instruction

   Repeat following CMPSB, CMPSW, SCASB, SCASW instructions while ZF = 1 (result is Equal), maximum CX times.

   Algorithm:

   check_cx:

   if CX <> 0 then

   do following chain instruction

   CX = CX - 1

   if ZF = 1 then:

   go back to check_cx

   else

   exit from REPE cycle

else

exit from REPE cycle

5. Algorithm:

1. Initialize the data memory with str1, strlen1, strrev, str_palin,str_not_palin
2. Initialize data and extra segment.
3. Move value of strlen1 into CX register
4. Load the value of strlen1 into CX register
5. Load the effective address of str1 into SI
6. Load the effective address of strrev into DI
7. Add SI with strlen1
8. Add SI with -2
9. Move the character pointed by SI to AL register
10. Move the character in AL to memory pointed by DI
11. Decrement SI
12. Increment DI
13. Goto 8
14. Move the character pointed by SI to AL register
15. Move the character in AL to memory pointed by DI
16. Increment DI
17. Move '$' to DL
18. Move the character in DL to memory pointed by DI
19. Load str1 to SI
20. Load strrev to DI
21. Compare character pointed by SI with character pointed by DI
22. Repeat till equal
23. Jump if not equal to 25
24. Display the message 'String is Palindrome.'
25. Display the message 'String is not Palindrome.'
26. Stop.

4. Stop.

6. Program:

```
Data Segment
  str1 db 'MADAM','$'
  strlen1 dw $-str1
  strrev db 20 dup(' ')
  str_palin db 'String is Palindrome.','$'
  str_not_palin db 'String is not Palindrome.','$'
Data Ends

Code Segment
  Assume cs:code, ds:data

  Begin:
    mov ax, data
    mov ds, ax
    mov es, ax
    mov cx, strlen1
    add cx, -2

    lea si, str1
    lea di, strrev

    add si, strlen1
    add si, -2
    L1:
      mov al, [si]
      mov [di], al
```

```
    dec si
    inc di
    loop L1
    mov al, [si]
    mov [di], al
    inc di
    mov dl, '$'
    mov [di], dl
    mov cx, strlen1

Palin_Check:
    lea si, str1
    lea di, strrev
    repe cmpsb
    jne Not_Palin

Palin:
    mov ah, 09h
    lea dx, str_palin
    int 21h
    jmp Exit

Not_Palin:
    mov ah, 09h
    lea dx, str_not_palin
    int 21h

Exit:
    mov ax, 4c00h
    int 21h
```

Code Ends

End Begin

7. **Conclusion**: Hence we have studied 8086 ALP to check whether a given string is a palindrome or not

8. **QUIZ / Viva Questions:**
   - What is Palindrome?
   - List some of the strings which are Palindrome?
   - Explain repe instruction.
   - Explain jne instruction.
   - Explain CMPS instruction

9. **References:**

1. Scott Mueller,"Upgrading and repairing PCs", Pearson,
2. John Uffenbeck, "8086/8088 family: Design Programming and Interfacing:"Pearson Education
3. K Bhurchandi, "Advanced Microprocessors & Peripherals", Tata McGraw-Hill Education

Microprocessor Programming Lab

Experiment No. :10

Write 8086 ALP to calculate length of the string and reverse it

Experiment No. 10

1. **Aim:** Write 8086 ALP to calculate length of the string and reverse it.
2. **What you will learn by performing this experiment?**

   To learn how to handle string related data, how to calculate length of it and how to reverse it.
3. **Apparatus Required:**

   Emulator 8086, TASM 1.4 for 64 bits

4. **Theory:**

   8086 String

   The 8086 microprocessor is equipped with special instructions to handle string operations. By string we mean a series of data words or bytes that reside in consecutive memory locations. The string instructions of the 8086 permit a programmer to implement operations such as to move data from one block of memory to a block elsewhere in memory. A second type of operation that is easily performed is to scan a string and data elements stored in memory looking for a specific value. Other examples are to compare the elements and two strings together in order to determine whether they are the same or different.

   In string $ indicates end of string

   Interrupt 21h

   AH = 02h -WRITE CHARACTER TO STANDARD OUTPUT

   Entry: DL = character to write

   Return: AL = last character output

   Notes:

   - ^C/^Break are checked
   - the last character output will be the character in DL unless DL=09h on entry, in which case AL=20h as tabs are expanded to blanks

- if standard output is redirected to a file, no error checks (write- protected, full media, etc.) are performed

AH = 09h - WRITE STRING TO STANDARD OUTPUT

Entry: DS:DX -> '$'-terminated string

Return: AL = 24h

Notes: ^C/^Break are checked

AH = 0Ah - BUFFERED INPUT

Entry: DS:DX -> buffer

Return: buffer filled with user input

Notes:

- ^C/^Break are checked
- reads from standard input

5. Algorithm:

Algorithm 1: 8086 ALP to calculate length of the string

1. Initialize the data memory with variables STR1, STR2, STR3, STR4, INSTR1, NEWLINE, LN,N with $, S.
2. Load effective address of INSTR1 to SI
3. Display the string STR1 on standard output device
4. Read the string in INSTR1 with int 21h with function 0A
5. Display the string STR2 on standard output device
6. Display the string in INSTR1 on standard output device
7. Direct Method :
    1. Display the string STR3 on standard output device
    2. Move value of INSTR1+1 to BL register.
    3. Add 30 to BL
    4. Move BD value to DL.
    5. Display the length on standard output device
8. Indirect Method:

1. Display the string STR4 on standard output device
2. Add SI with 2
3. Move 00 to AX register
4. Compare the byte pointed by SI with '$'
5. Jump if equal to 9
6. Increment SI
7. Add AL with 1
8. Jump to 4
9. Subtract 1 from AL
10. Add AL with 30
11. Display the length on standard output device

9. Stop.

Algorithm 2: 8086 ALP to display the reverse of the string

1. Initialize the data memory with str1, strlen1, strrev.
2. Initialize the data segment and extra segment
3. Load the value of strlen1 into CX register
4. Load the effective address of str1 into SI
5. Load the effective address of strrev into DI
6. Add SI with strlen1
7. Add SI with -2
8. Move the character pointed by SI to AL register
9. Move the character in AL to memory pointed by DI
10. Decrement SI
11. Increment DI
12. Goto 8
13. Move the character pointed by SI to AL register
14. Move the character in AL to memory pointed by DI
15. Increment DI

16. Move '$' to DL
17. Move the character in DL to memory pointed by DI
18. Display the string with int 21h to standard output device
19. Stop.

6. Program:

; Program 1: 8086 ALP to calculate length of the string

```
DATA SEGMENT
    STR1 DB "ENTER YOUR STRING HERE ->$"
    STR2 DB "YOUR STRING IS ->$"
    STR3 DB "LENGTH OF STRING IS(DIRECT) ->$"
    STR4 DB "LENGTH OF STRING IS(COUNT) ->$"
    INSTR1 DB 20 DUP("$")
    NEWLINE DB 10,13,"$"
    LN DB 5 DUP("$")
    N DB "$"
    S DB ?
DATA ENDS
CODE SEGMENT
    ASSUME DS:DATA,CS:CODE
START:
    MOV AX,DATA
    MOV DS,AX
   LEA SI,INSTR1
;GET STRING
    MOV AH,09H
    LEA DX,STR1
    INT 21H
   MOV AH,0AH
   MOV DX,SI
   INT 21H
```

```
        MOV AH,09H
        LEA DX,NEWLINE
        INT 21H
;PRINT THE STRING

        MOV AH,09H
        LEA DX,STR2
        INT 21H
       MOV AH,09H
      LEA DX,INSTR1+2
      INT 21H
    MOV AH,09H
    LEA DX,NEWLINE
    INT 21H

;PRINT LENGTH OF STRING (DIRECT)
        MOV AH,09H
        LEA DX,STR3
        INT 21H
       MOV BL,INSTR1+1
        ADD BL,30H
        MOV AH,02H
        MOV DL,BL
        INT 21H
        MOV AH,09H
        LEA DX,NEWLINE
        INT 21H
;PRINT LENGTH OF STRING ANOTHER WAY
        MOV AH,09H
```

```
        LEA DX,STR4
        INT 21H
       ADD SI,2
       MOV AX,00
L2:CMP BYTE PTR[SI],"$"
       JE L1
       INC SI
      ADD AL,1
     JMP L2

L1:SUB AL,1
    ADD AL,30H

  MOV AH,02H
  MOV DL,AL
  INT 21H
 MOV AH,4CH
 INT 21H
CODE ENDS
END START
```

; Program 2: 8086 ALP to display the reverse of the string

```
Data Segment
  str1 db 'good morning','$'
  strlen1 dw $-str1
  strrev db 20 dup(' ')
Data Ends

Code Segment
   Assume cs:code, ds:data
  Begin:
```

```
    mov ax, data
    mov ds, ax
    mov es, ax
    mov cx, strlen1
    add cx, -2
    lea si, str1
    lea di, strrev
    add si, strlen1
    add si, -2
    L1:
      mov al, [si]
      mov [di], al
      dec si
      inc di
      loop L1
      mov al, [si]
      mov [di], al
      inc di
      mov dl, '$'
      mov [di], dl
    Print:
      mov ah, 09h
      lea dx, strrev
      int 21h
    Exit:
      mov ax, 4c00h
      int 21h
Code Ends
End Begin
```

7. **Conclusion**: Hence we have studied ALP to calculate length of the string and reverse it.
8. **QUIZ / Viva Questions:**
   - What is int 21h with ah=02h?
   - What is int 21h with ah=09h?
   - What is int 21h with ah=0Ah?
   - What is mean by '$' in string?
9. **References:**

1. Scott Mueller,"Upgrading and repairing PCs", Pearson,
2. John Uffenbeck, "8086/8088 family: Design Programming and Interfacing:"Pearson Education
3. K Bhurchandi, "Advanced Microprocessors & Peripherals", Tata McGraw-Hill Education

Microprocessor Programming Lab

Experiment No. : 11

Write 8086 ALP to compute the factorial of a positive integer ‘n’ using recursive procedure

Experiment No.11

1. **Aim:** Write 8086 ALP to compute the factorial of a positive integer 'n' using recursive procedure.

2. **What you will learn by performing this experiment?**

   Students will learn what is procedure, how to call it, how to return from it to main program.

3. **Apparatus Required:**

   Emulator 8086, TASM 1.4 for 64 bits

4. **Theory:**

   Procedures

   A procedure is a set of code that can be branched to and returned from in such a way that the code is as if it were inserted at the point from which it is branched to. The branch to procedure is referred to as the *call,* and the corresponding branch back is known as the *return*. The return is always made to the instruction immediately following the call regardless of where the call is located.

   1. Calls, Returns, and Procedure Definitions

   The CALL instruction not only branches to the indicated address, but also pushes the return address onto the stack. The RET instruction simply pops the return address from the stack. The registers used by the procedure need to be stored before their contents are changed, and then restored just before their contents are changed, and then restored just before the procedure is excited.

   A CALL may be direct or indirect and intrasegment or intersegment. If the CALL is intersegment, the return must be intersegment. Intersegment call must push both (IP) and (CS) onto the stack. The return must correspondingly pop two words from the stack. In

the case of intrasegment call, only the contents of IP will be saved and retrieved when call and return instructions are used.

Procedures are used in the source code by placing a statement of the form at the beginning of the procedure

Procedure name PROC Attribute and by terminating the procedure with a statement

Procedure name ENDP

The attribute that can be used will be either NEAR or FAR. If the attribute is NEAR, the RET instruction will only pop a word into the IP register, but if it is FAR, it will also pop a word into the CS register.

A procedure may be in:

1. The same code segment as the statement that calls it.

2. A code segment that is different from the one containing the statement that calls it, but in the same source module as the calling statement.

3. A different source module and segment from the calling statement.

In the first case, the attribute could be NEAR provided that all calls are in the same code segment as the procedure. For the latter two cases the attribute must be FAR. If the procedure is given a FAR attribute, then all calls to it must be intersegment calls even if the call is from the same code segment. For the third case, the procedure name must be declared in EXTRN and PUBLIC statements.

2 Saving and Restoring Registers

When both the calling program and procedure share the same set of registers, it is necessary to save the registers when entering a procedure, and restore them before returning to the calling program.

MSK PROC NEAR

PUSH AX

PUSH BX

PUSH CX

POP CX

POP BX

POP AX

RET

MSK ENDP

3 Procedure Communications

There are two general types of procedures; those operate on the same set of data and those that may process a different set of data each time they are called. If a procedure is in the same source module as the calling program, then the procedure can refer to the variables directly.

When the procedure is in a separate source module it can still refer to the source module directly provided that the calling program contains the directive.

PUBLIC ARY, COUNT, SUM

EXTRN ARY: WORD, COUNT: WORD, SUM: WORD

4 Recursive Procedures

When a procedure is called within another procedure it called recursive procedure. To make sure that the procedure does not modify itself, each call must store its set of parameters, registers, and all temporary results in a different place in memory

5. Algorithm:

1. Declare the variables
2. Initialize code and data segments
3. Load effective address of MSG2 into DX register.
4. Accept the input from user in AL register
5. Move AL into N
6. Move 1 into AX
7. Move N into BL
8. Move 0 into BH
9. Call procedure FACTORIAL
10. Move AX into FACT variable

Procedure FACTORIAL

i. Compare the value of BX with 1
ii. Jump if equal at label L1
iii. Push the contents of BX onto Stack Memory
iv. Decrement BX by 1
v. Call procedure FACTORIAL
vi. Pop the contents of BX from Stack Memory
vii. Multiply the contents of BX with AX and store the result into AX.

6. Program:

```
DATA SEGMENT
   MSG1 DB "ENTER NUMBER : $"
   N DB ?
   FACT DW ?
DATA ENDS
CODE SEGMENT
ASSUME CS:CODE, DS:DATA
START:
      MOV AX, DATA
      MOV DS, AX
      LEA DX,MSG1
      MOV AH,9
      INT 21H
      MOV AH,1
      INT 21H
      SUB AL,30H
      MOV N,AL
      MOV AX, 1
      MOV BL, N
      MOV BH, 0
      CALL FACTORIAL
      MOV FACT, AX
      MOV AH, 4CH
      INT 21H

      FACTORIAL PROC
      CMP BX, 1
```

```
        JE L1
        PUSH BX
        DEC BX
        CALL FACTORIAL
        POP BX
        MUL BX
L1:     RET
        FACTORIAL ENDP
CODE ENDS
END START
```

7. **Conclusion:** Hence We have studied ALP to compute the factorial of a positive integer 'n' using recursive procedure.
8. **QUIZ / Viva Questions:**
    - What is procedure?
    - Explain in detail recursive procedure.
    - Differentiate between FAR and NEAR procedure.
    - How to pass parameters to procedure?
    - Explain RET instruction.
    - What is stack in 8086 and its use in procedure.
9. **References:**

1. Scott Mueller,"Upgrading and repairing PCs", Pearson,
2. John Uffenbeck, "8086/8088 family: Design Programming and Interfacing:"Pearson Education
3. K Bhurchandi, "Advanced Microprocessors & Peripherals", Tata McGraw-Hill Education

Microprocessor Programming Lab

Experiment No. : 12

Interfacing Seven Segment Display

Experiment No.12

1. **Aim:** Study of Interfacing Seven Segment Display with 8086 using 8255
2. **What you will learn by performing this experiment?**
   Students will learn Interfacing Seven Segment Display with 8086 using 8255.
3. **Apparatus Required**: 7-SEG Display Card, Interface Cable, Datasheets

4. **Theory:**

Introduction to 7-Segment display

PS-ADD-ON 7SEG card has 6 Nos. number of Common Anode **7segment displays** arranged in order with driver by using NPN transistors, All the **7-segment displays** segment lines and digit selections and power lines are terminated with 20-pin connector, to interface development kits.

7-SEG Display Characteristics

- The High Efficiency Red source color devices are made with Gallium Arsenide Phosphide on Gallium Phosphate Orange Light Emitting Diode.
- 0.56 inch digit height.
- Low current operation.
- Excellent character appearance.
- Easy mounting on P.C. boards or sockets.
- Mechanically rugged.

Specifications

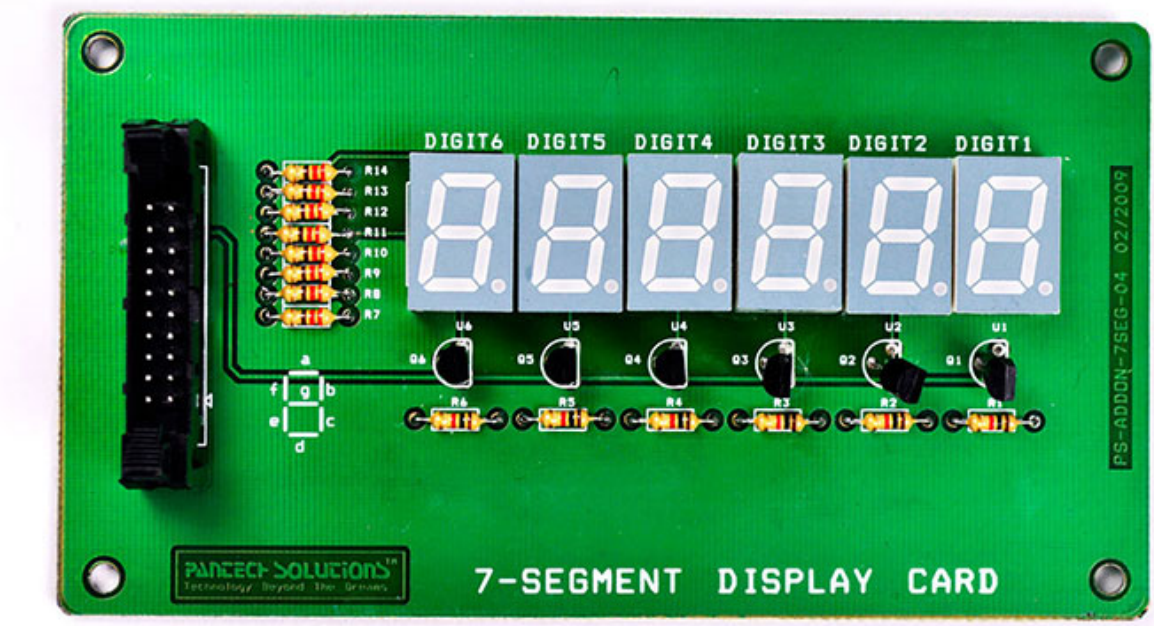
- Digital Outputs
  - 6 Nos. of Common Anode (7-SEG) Display.
- 20-pin Box Header
  - All LED and Switch | Power lines terminated at box connector

- 20-pin FRC Cable
    - To connect host boards (Microcontroller/Processor/FPGA Kits)

Card Features

- 6 Nos. common anode **7-segment Display**
- Transistors driver Circuit for displays
- 20-pin Box Connector

Hardware Description of seven segment display Card

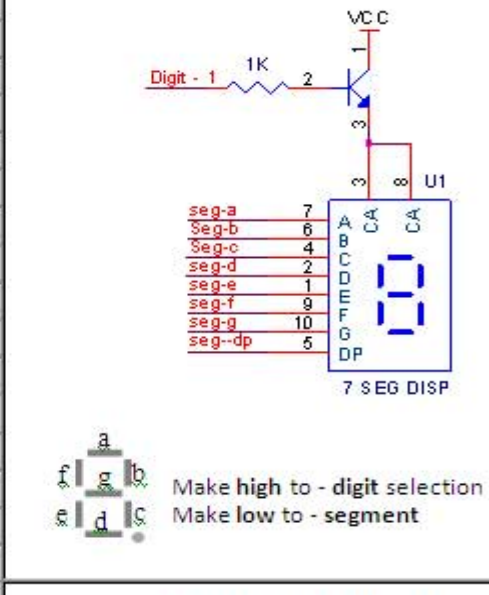


7-SEG Display card has 6 nos. of common anode **7segment displays** arranged in order with driver by using NPN transistors, all the**7-segment displays** segment lines and digit selections and power lines are terminated with 20-pin connector, to interface development kits.

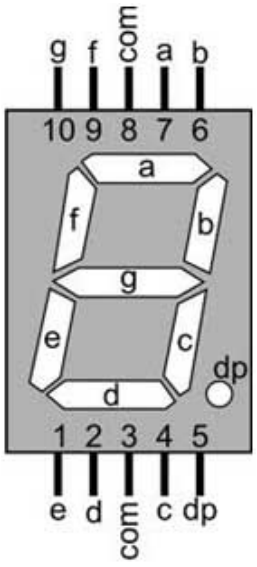
Standard: gray face, white segment.

RoHS compliant.

Hardware Configuration of 7 segments With 8086

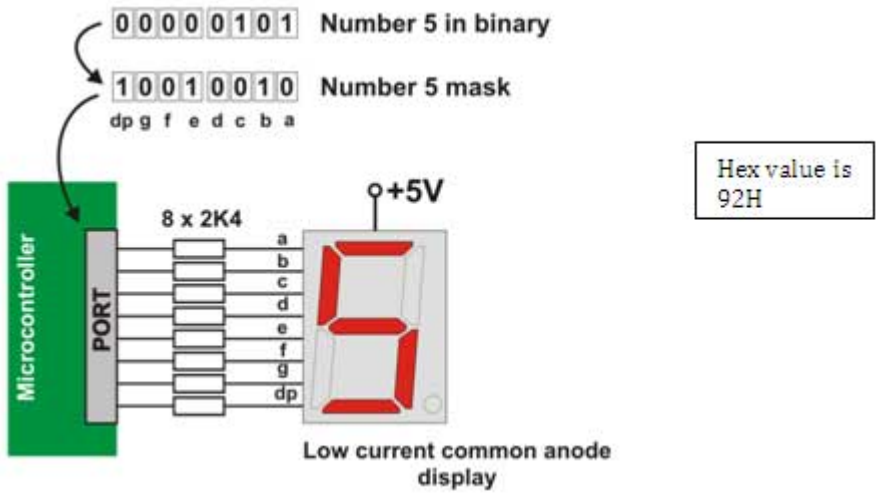
	20PIN CONNECTOR	MODULES	7-SEGMENT DISPLAY CARD
DIGIT SELECT LINES	1	Digit - 1	
	2	Digit – 2	
	3	Digit – 3	
	4	Digit - 4	
	5	Digit – 5	
	6	Digit - 6	
	7	NC	
	8	NC	
SEGMENT LINES	9	Seg - a	
	10	Seg – b	
	11	Seg – c	
	12	Seg – d	
	13	Seg – e	
	14	Seg – f	
	15	Seg – g	
	16	Seg – dp	
PWR	17,19	Vcc	Supply form ARM7 Tyro Kit
	18,20	Gnd	

WHAT IS A 7-SEG DISPLAY?



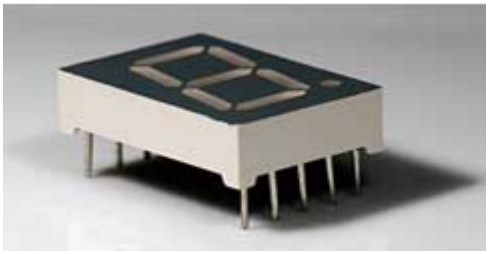
7-segment Display, it is composed of 8 LEDs, **7 segments** are arranged as a rectangle for symbol displaying and there is an additional segment for decimal point displaying. In order to simplify connecting, anodes and cathodes of all diodes are connected to the common pin so that there are common anode displays and common cathode displays, respectively. Segments are marked with the letters from A to G, plus DP, as shown in the figure on the left. On connecting, each diode is treated separately, which means that each must have its own current limiting resistor.

A **seven segment display**, as its name indicates, is composed of seven elements. Individually on or off, they can be combined to produce simplified representations of the Arabic numerals. Often the **seven segments** are arranged in an oblique (slanted) arrangement, which aids readability. In most applications, the **seven segments** are of nearly uniform shape and size (usually elongated hexagons, though trapezoids and rectangles can also be used), though in the case of adding machines, the vertical segments are longer and more oddly shaped at the ends in an effort to further enhance readability.



7-segment Display

TYPES AND DISPLAY FORMAT



The hex decimal data corresponding to the segments which have to glow for displaying a character is output to port B

Display Format

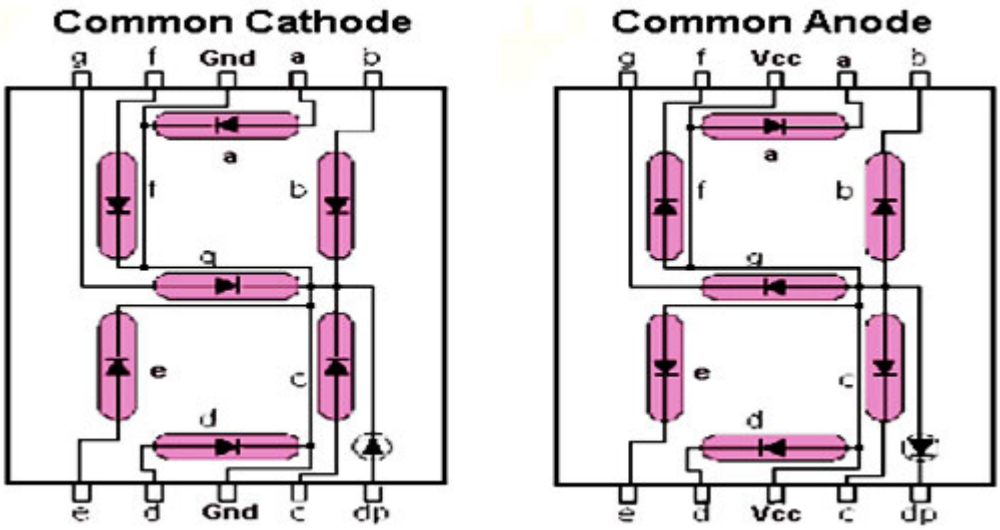
D	D	D	D	D	D	D	D
D	G	F	E	D	C	B	A

Logic ‘0' in the above said format will turn ON that particular segment. Logic ‘1' will keep the segment OFF. The data for turning ON the display is through 8255.

7 – Segment Types

There are two types of **seven segment display** available, namely,

- Common anode display and
- Common cathode display



A) 7 – Segment Display Interfacing With 8051 Trainer Kit

IN 8051 WE HAVE SINGLE 8255 IC

GPIO- I (8255) J1 Connector

PORTS	ADDRESS
Control port	4003
PORT A	4000
PORT B	4001
PORT C	4002

B) 7 – Segment Display Interfacing with 8086 Trainer Kit

IN 8086 WE HAVE TWO 8255 IC'S

GPIO- I (8255)

PORTS	ADDRESS
Control port	FF26
PORT A	FF20
PORT B	FF22
PORT C	FF24

GPIO- II (8255)

PORTS	ADDRESS
Control port	FF36
PORT A	FF30
PORT B	FF32
PORT C	FF34

5. Program:

ADDRESS	OPCODE	MNEMONICS
1200	B0 80	MOV AL, 80
1202	BA 36 FF	MOV DX, FF36
1205	EE	OUT DX, AL
1206	8D 36 00 13	START: LEA SI, [1300]
120A	B9 06 00	MOV CX, 000A
120D	B3 7F	MOV BL, 7F
120F	88 D8	AGN: MOV AL, BL
1211	BA 30 FF	MOV DX, FF30
1214	EE	OUT DX, AL
1215	8A 04	MOV AL, [SI]
1217	BA 32 FF	MOV DX, FF32
121A	EE	OUT DX, AL
121B	E8 07 00	CALL DLY
121E	46	INC SI
121F	D0 CB	ROR BL, 1
1221	E2 EC	LOOP AGN
1206	EB E1	JMP START
1225	BA FF FF	DLY: MOV DX, FFFF
1228	4A	M: DEC DX
1229	75 FD	JNZ M
122B	C3	RET

6. **Conclusion:** Hence we have studied Interfacing Seven Segment Display.

7. **QUIZ / Viva Questions:**

   - Draw and explain Seven Segment Display?

8. **References:**

1. Scott Mueller,"Upgrading and repairing PCs", Pearson,
2. John Uffenbeck, "8086/8088 family: Design Programming and Interfacing:"Pearson Education
3. K Bhurchandi, "Advanced Microprocessors & Peripherals", Tata McGraw-Hill Education

Microprocessor Programming Lab

Experiment No. : 13

Interfacing DAC

Experiment No.12

1. **Aim:** Study of Interfacing DAC.
2. **What you will learn by performing this experiment?**

   Students will learn Interfacing DAC interfacing of DAC 0808 kit to 8086 kit.
3. **Apparatus Required:**

   8086 Microprocessor Kit, CRO, SMPS, Keyboard, DAC 0808 Card, Interfacing Cables.
4. **Theory:**

   Performance parameters

   a. Resolution: This is the number of possible output levels the DAC is designed to reproduce. This is usually stated as the number of bits it uses, which is the base two logarithm of the number of levels.

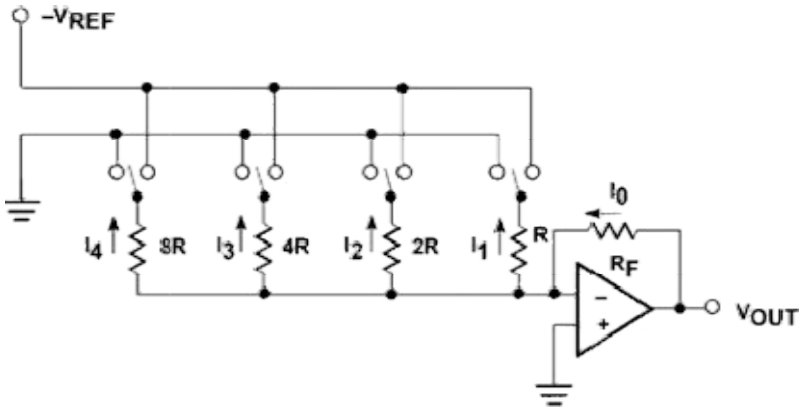
   b. Maximum sampling rate: This is a measurement of the maximum speed at which the DACs circuitry can operate and still produce the correct output.

   c. Monotonicity: This refers to the ability of a DAC's analog output to move only in the direction that the digital input moves

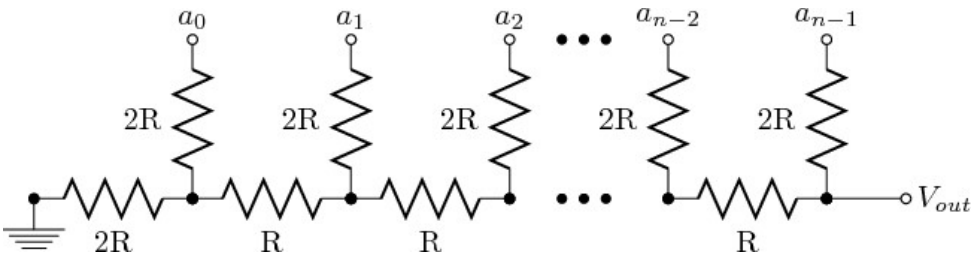
   d. Settling time: settling time is the interval between a command to update (change) its output value and the instant it reaches its final value, within a specified percentage.

   Different techniques of conversions:

   a. Binary weighted resistor D/A converter



b. R/2R Ladder D/A converter,



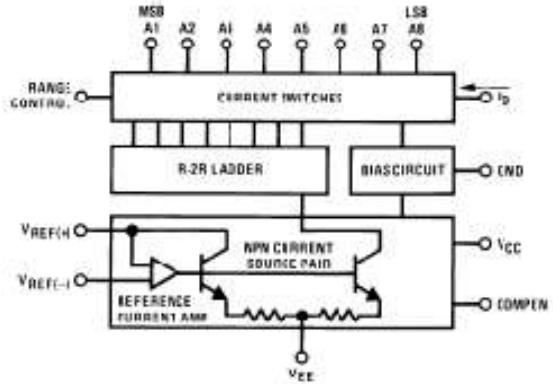
Sources of errors in DAC:

a. Differential Nonlinearity error: DNL error is the difference between the ideal and the measured output responses for successive DAC codes. An ideal DAC response would have analog output values exactly one code (LSB) apart (DNL = 0).

b. Offset error: Offset error, often called 'zero-scale' error, indicates how well the actual transfer function matches the ideal transfer function at a single point. For an ideal data converter, the first transition occurs at 0.5LSB above zero. For a DAC, offset error is the analog output response to an input code of all zeros.

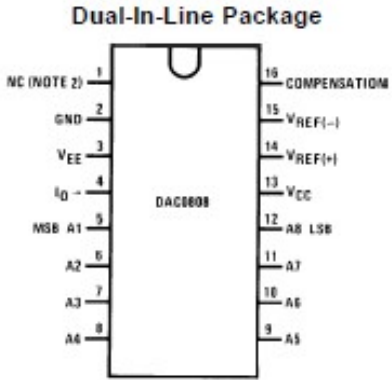
c. Gain error: indicates how well the slope of an actual transfer function matches the slope of the ideal transfer function. Gain error is usually expressed in LSB or as a percent of full-scale range (%FSR), and it can be calibrated out with hardware or in software. Gain error is the full-scale error minus the offset error.

Block Diagram of DAC 0808:



Pin Diagram & Features of DAC 0808

Pin Diagram



Features:

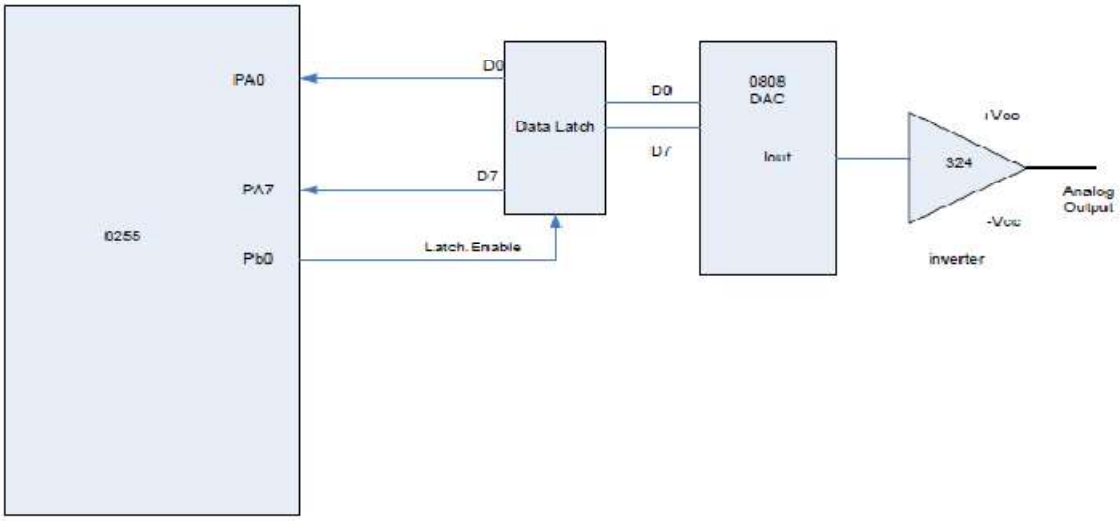
- Relative accuracy: ±0.19% error maximum
- Full scale current match: ±1 LSB typ
- Fast settling time: 150 ns typ
- Noninverting digital inputs are TTL and CMOS compatible
- High speed multiplying input slew rate: 8 mA/μs
- Power supply voltage range: ±4.5V to ±18V
- Low power consumption: 33 mW @ ±5V
-

1. Square wave – variable duty cycle and frequency.

We are asked to generate a square wave using DAC interface. To generate square wave we will output FFH and then 00H on port A of 8255. The output of 8255 (port A) is compulsory connected to the DAC 0808. We also have to vary duty cycle. Variation in Duty cycle will automatically change the frequency as:

Duty cycle = Ton / Ton + Toff = Ton / Tfrequency

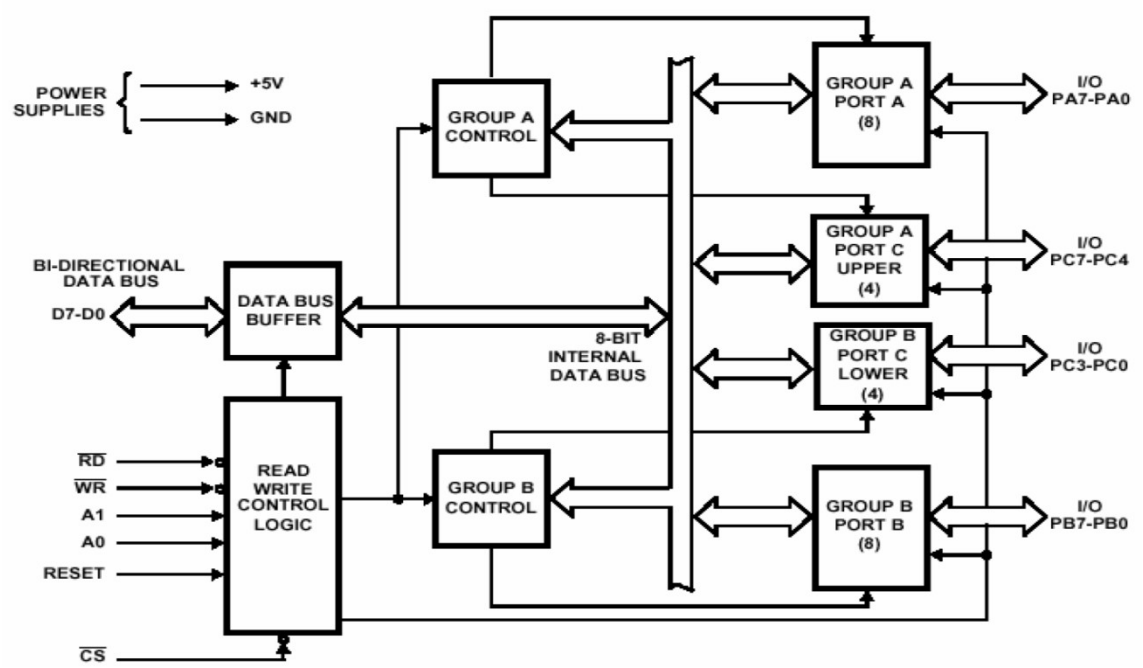
According to output duty cycle requirement, we can change the Delay between the two outputs FFH (for output high) and 00H (for output low.)



2. Ramp Wave: variable direction.

We are asked to generate a ramp wave using DAC interface. To generate ramp wave we will output 00 to FFH. If we want a ramp wave with reverse direction then, we output in reverse manner. If we want ramp wave in forward direction, we will initialize, al = 00H, otherwise AL = FFH for reverse direction

Figure shows 8255 PPI:



5. Procedure:

Square wave

Step 1 : Initialize 8255

Step 2 : Initialize latch

Step 3 : Display 00 .

Step 4 : Call delay

Step 5 : Display FF.

Step 6 : Call delay

Step 7 : Goto step 3

Ramp wave

1. Initialize 8255

2. Initialize latch

3. Send 00 on port A

4. Display on CRO.

5. Increment the content by 1

6. Go to step 4

6. Program:

Square Wave:

```
        MOV CX,1100
        MOV DX,8006 ; Address of Control Word Register
        MOV AL,80H
        OUT DX,AL
        MOV DX,8000 ;Address of Port A
XX:     MOV AL,00H
        OUT DX,AL
        INT AAH
        MOV AL,FFH
        OUT DX,AL
        INT AAH
        JMP XX
```

Ramp Wave:

```
        MOV CX,1100
        MOV DX,8006 ; Address of Control Word Register
        MOV AL,80H
        OUT DX,AL
UP:     MOV DX,8000 ;Address of Port A
        MOV AL,00H
XX:     OUT DX,AL
        INC AL
        CMP AL,FFH
        JNE XX
        JMP UP
```

7. Conclusion: Hence we have studied Interfacing Seven Segment Display.

8. QUIZ / Viva Questions:

- Discuss the organization and architecture of 8255 PPI with a functional block diagram.
- Explain various modes of operations of 8255PPI.

- Explain in detail: Centronics Printer Parallel Interfacing using 8255.
- Explain control word of 8255 for I/O mode and BSR mode.
- Explain:
- Resolution
- Maximum sampling rate
- Monotonicity
- Settling time
- Explain with the diagram R/2R Ladder D/A converter
- List Sources of errors in DAC & explain
- Draw & explain Block Diagram of DAC 0808
- List the Features of DAC 0808

9. References:

1. Scott Mueller,"Upgrading and repairing PCs", Pearson,
2. John Uffenbeck, "8086/8088 family: Design Programming and Interfacing:"Pearson Education
3. K Bhurchandi, "Advanced Microprocessors & Peripherals", Tata McGraw-Hill Education