Module 2
Overview of Computer Architecture & Organization

https://www.youtube.com/watch?v=jTa0w-MxFJE

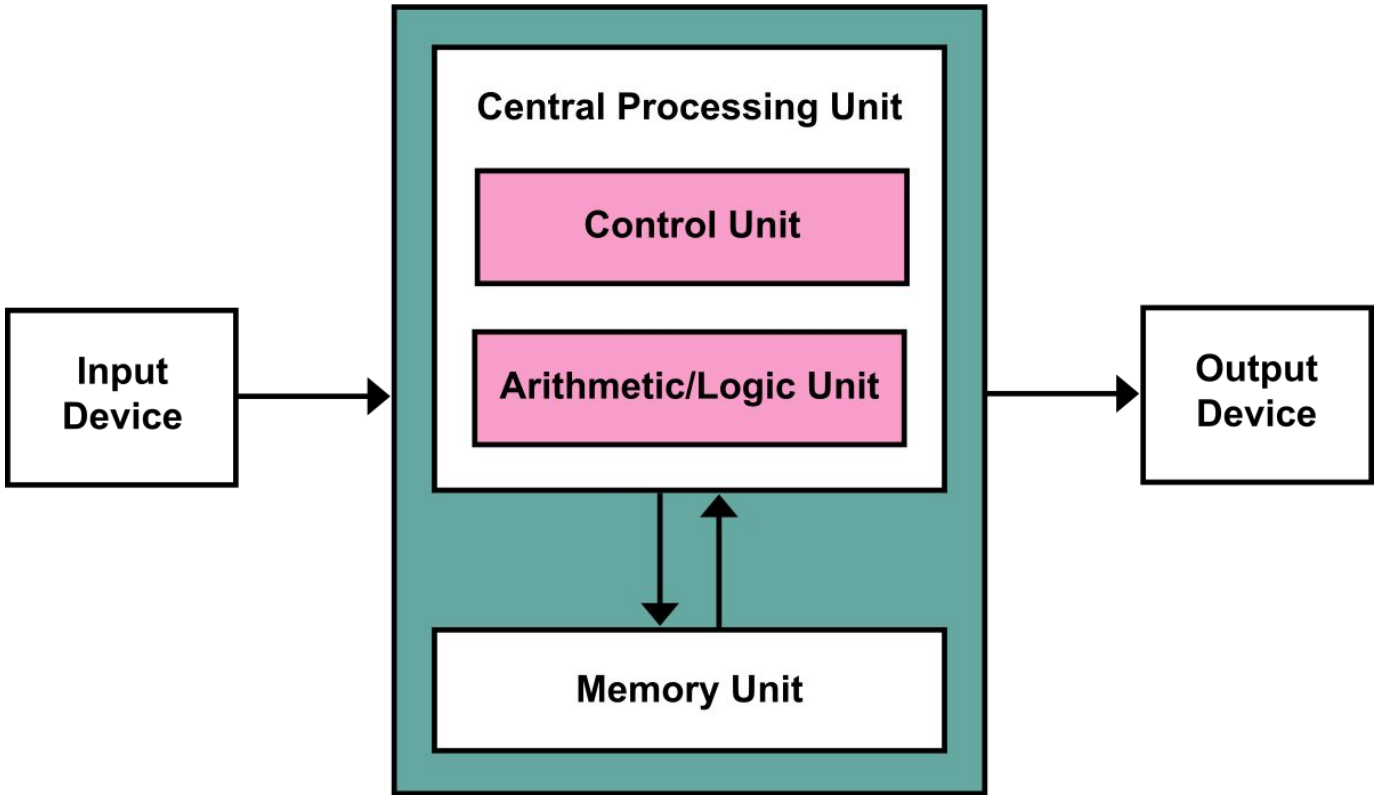
Introduction

- A computer is a combination of **hardware and software** resources which integrate together and provides various functionalities to the user.
- Hardware are the physical components of a computer like the processor, memory devices, monitor, keyboard etc.
- While software is the set of programs or instructions that are required by the hardware resources to function properly.

Introduction

- **Computer architecture** describes those attributes of the system that are visible to the user.
- **Computer organization** deals with implementation of these features and is mostly unknown to the user.

Functional units of computer



Functional units of computer

The components of computer are CPU, memory and I/O devices.These units are connected to each other with the help of buses.

1. **Memory:** It is used to store program and data.
2. **I/O devices:** It is used to accept input and produce desired output.

Functional units of computer

3. **CPU:** It consist of ALU and control unit.
    - ALU: It performs arithmetic operations.
    - Control unit: It stores instructions into the register and gives signal in proper sequence to execute a instruction.

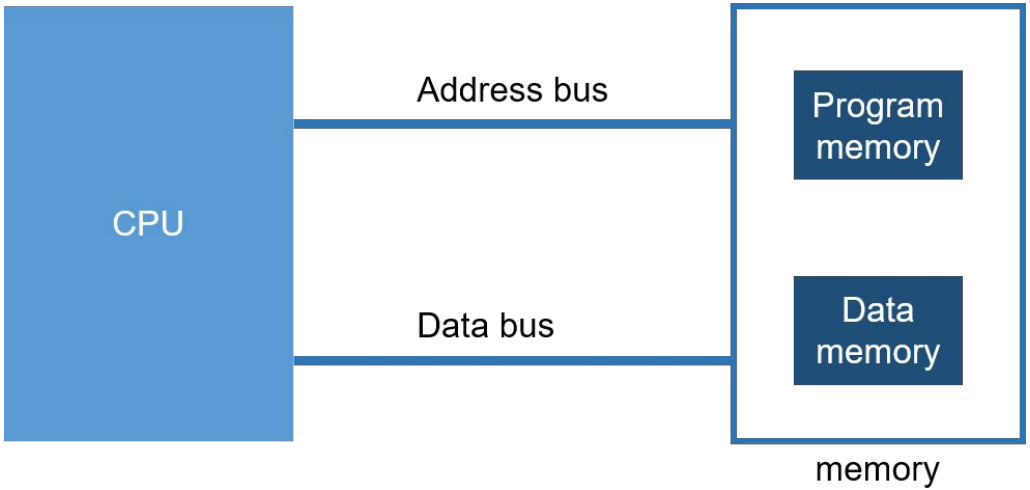
The functions of these units are data processing, data storage , data movement and control.

Von Neumann & Harvard Architecture

- There are basically two types of digital computer architectures.
- The first one is called Von Neumann architecture and later Harvard architecture was adopted for designing digital computers.

Von Neumann Architecture

The architecture which uses common bus to access data memory and program memory is called as von neumann architecture



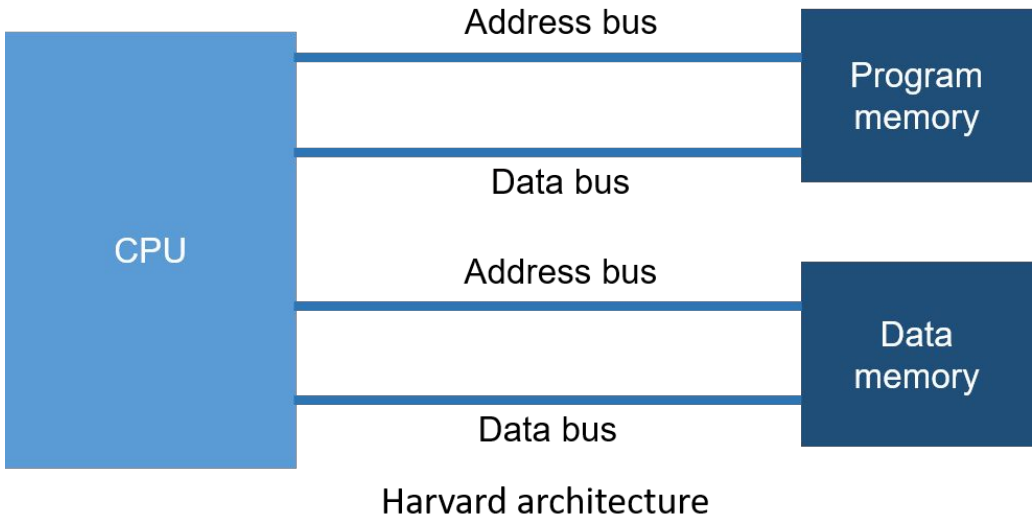
Von neumann architecture

Von Neumann Architecture

- One shared memory for instructions (program) and data with one data bus and one address bus between processor and memory.
- Instructions and data have to be fetched in sequential order (known as the Von Neuman Bottleneck), limiting the operation bandwidth. Its design is simpler than that of the Harvard architecture.
- Most of the general purpose microprocessors such as motorola 6800, intel 8086 use this architecture.

Harvard Architecture

- Harvard architecture uses separate buses for instructions and data.



Harvard architecture

Harvard Architecture

- The instruction address bus and instruction bus used for reading instructions from memory. The address bus and data bus are used for writing and reading data to and from memory.

Comparison

Von Neumann Architecture	Harvard Architecture
1. It requires less hardware	1. It requires more hardware
2. Von-Neumann Architecture requires less space.	2. Harvard Architecture requires more space.
3. Von Neumann Architecture is based on the stored-program concept.	3. Harvard Architecture is based on relay-based computer models.
4. Controlling becomes simpler since either data or instructions are to be fetched at a time.	4. Controlling becomes complex since data and instructions are to be fetched simultaneously.
5. In Von Neumann Architecture, Common bus used for transferring instructions and data.	5. In Harvard Architecture, Separate buses are used to transfer instructions and data.
6. Speed of execution is slower since it cannot fetch the data and instructions at the same time.	6. Speed of execution is faster because the processor fetches data and instructions simultaneously .
7. Von Neumann Architecture is cheaper than Harvard architecture.	7. Harvard Architecture is more expensive than Von Neumann's architecture.
8. It is mainly used in personal computers (PC)	8. It is mainly used in micro-controllers and signal processing.

Microprocessor

It is a programmable device which has all the functions of CPU of a computer.

CPU performs following functions:

- Access information
- Store information
- Execute and process information
- Give result in desired form

Introduction to 8086 Microprocessor

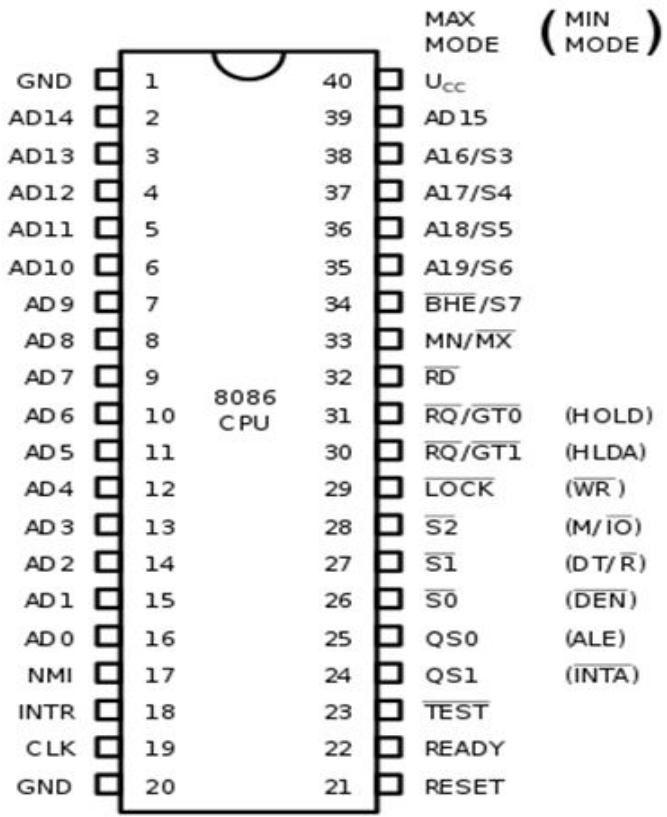
- It is a 16-bit microprocessor which means is that the ALU and the internal registers work with 16 bit of binary data at a time.
- It has 16 data lines and 20 address lines.
- It is capable to do multiplication and division also.
- It has two modes of operation: maximum mode and minimum mode.

Introduction to 8086 Microprocessor

- **Maximum mode:** In this, various control signals of microprocessor are generated by some external logic.

- **Minimum mode:** In this, various control signals of microprocessor are generated by its own.

Introduction to 8086 Microprocessor



Introduction to 8086 Microprocessor

Three types of signals available in 8086:

1. Common mode signal
2. Minimum mode signal
3. Maximum mode signal

1. **Common mode signal**

- **AD0-AD15 (Address Data Bus):** Bidirectional address/data lines. These are low order address bus. They are multiplexed with data. When these lines are used to transmit memory address, the symbol A is used instead of AD, for example, A0- A15.
- **A16 - A19 (Output):** High order address lines. These are multiplexed with status signals.

1. **Common mode signal**

- **BHE/S7 (Output):** Bus High Enable/Status. During T1, it is low. It enables the data onto the most significant half of data bus, D8-D15. 8-bit device connected to upper half of the data bus use BHE signal. It is multiplexed with status signal S7. S7 signal is available during T3 and T4.

1. **Common mode signal**

- **RD (Read):** For read operation. It is an output signal. It is active when LOW.
- **Ready (Input):** The addressed memory or I/O sends acknowledgment through this pin. When HIGH, it denotes that the peripheral is ready to transfer data.
- **RESET (Input):** System reset. The signal is active HIGH.
- **CLK (input):** Clock 5, 8 or 10 MHz.

1. **Common mode signal**

- **INTR:** Interrupt Request.
- **NMI (Input):** Non-maskable interrupt request.
- **TEST (Input):** Wait for test control. When LOW the microprocessor continues execution otherwise waits.
- **VCC:** Power supply +5V dc.
- **GND:** Ground.

2. Minimum mode signal

- **INTA (Output):** Pin number 24 interrupts acknowledgement. On receiving interrupt signal, the processor issues an interrupt acknowledgment signal. It is active LOW.
- **ALE (Output):** Pin no. 25. Address latch enable. It goes HIGH during T1. The microprocessor 8086 sends this signal to latch the address into the Intel 8282/8283 latch.

2. Minimum mode signal

- **DEN (Output):** Pin no. 26. Data Enable. When Intel 8287/8286 octal bus transceiver is used this signal. It is active LOW.
- **DT/R (output):** Pin No. 27 data Transmit/Receives. When Intel 8287/8286 octal bus transceiver is used this signal controls the direction of data flow through the transceiver. When it is HIGH, data is sent out. When it is LOW, data is received.

2. Minimum mode signal

- **M/IO (Output):** Pin no. 28, Memory or I/O access. When this signal is HIGH, the CPU wants to access memory. When this signal is LOW, the CPU wants to access I/O device.
- **WR (Output):** Pin no. 29, Write. When this signal is LOW, the CPU performs memory or I/O write operation.

3. **Maximum mode signal**

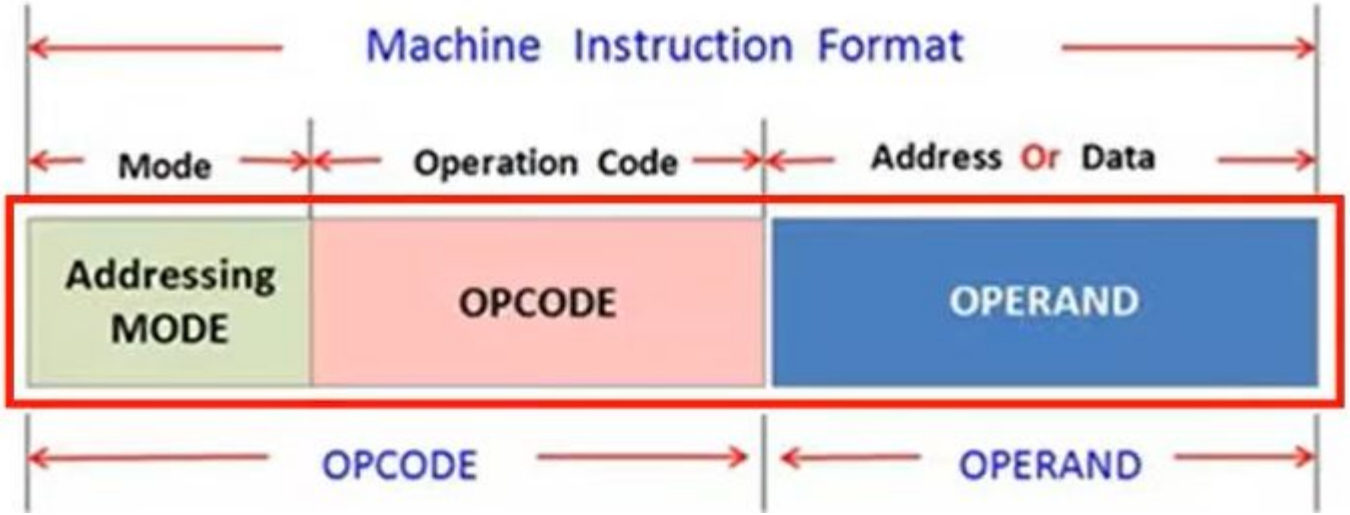
- **S0, S1, S2 (Output):** Pin numbers 26, 27, 28 Status Signals. These signals are connected to the bus controller of Intel 8288 which generates memory and I/O access control signals.
- **LOCK (Output):** Pin no. 29. It is an active LOW signal. When this signal is LOW, all interrupts are masked and no HOLD request is granted.
- **RG/GT1, RQ/GT0 (Bidirectional):** Pin numbers 30, 31, Local Bus Priority Control. Other processors ask the CPU by these lines to release the local bus.

2. **Minimum mode signal**

- **HLDA (Output):** Pin no. 30, Hold Acknowledgment. It is sent by the processor when it receives HOLD signal. It is active HIGH signal. When HOLD is removed HLDA goes LOW.
- **HOLD (Input):** Pin no. 31, Hold. When another device in microcomputer system wants to use the address and data bus, it sends HOLD request to CPU through this pin. It is an active HIGH signal.

Addressing modes

The term addressing modes refers to the way in which the operand of an instruction is specified. It specifies a rule for interpreting or modifying the address field of the instruction before the operand is actually executed.



Addressing modes

The 8086 microprocessors have 8 addressing modes.

Two addressing modes have been provided for instructions which operate on register or immediate data.

1. Register Addressing

In this, the operands / values are stored within any of the internal registers itself. So, the processing on these operands is done either in those registers or by shifting their values to some other registers. The operand is placed in one of the 16-bit or 8-bit general purpose registers.

Example

- MOV AX, CX
- ADD AL, BL
- ADD CX, DX

2. Immediate Addressing

In this addressing mode, the operands are specified within the instructions which means is that the instruction will either contain the values itself or will contain the operands whose values are required.

Example

- MOV AL, 35H
- MOV BX, 0301H
- MOV [0401], 3598H

3. Direct Addressing

In direct addressing mode, the operands offset is given in the instruction as an 8-bit or 16-bit displacement element.

Example

- ADD AL, [0301]

The instruction adds the content of the offset address 0301 to AL. the operand is placed at the given offset (0301) within the data segment DS.

4. Register Indirect Addressing

The operand's offset is placed in any one of the registers BX, BP, SI or DI as specified in the instruction.

Example

- MOV AX, [BX]

It moves the contents of memory locations addressed by the register BX to the register AX.

5. Based Addressing

The operand's offset is the sum of an 8-bit or 16-bit displacement and the contents of the base register BX or BP. BX is used as base register for data segment, and the BP is used as a base register for stack segment.

Effective address (Offset) = [BX + 8-bit or 16-bit displacement].

Example

- MOV AL, [BX+05]; an example of 8-bit displacement.
- MOV AL, [BX + 1346H]; example of 16-bit displacement.

6. Indexed Addressing

The offset of an operand is the sum of the content of an index register SI or DI and an 8-bit or 16-bit displacement.

Offset (Effective Address) = [SI or DI + 8-bit or 16-bit displacement]

Example

- MOV AX, [SI + 05]; 8-bit displacement.
- MOV AX, [SI + 1528H]; 16-bit displacement.

7. Base Indexed Addressing

The offset of operand is the sum of the content of a base register BX or BP and an index register SI or DI.

Effective Address = [BX or BP] + [SI or DI]

Here, BX is used for data segment, and BP is used for stack segment.

Example

- ADD AX, [BX + SI]
- MOV CX, [BX + SI]

8. Base Indexed with displacement Addressing

In this mode of addressing, the operand's offset is given by:

Effective Address (Offset) = [BX or BP] + [SI or DI] + 8-bit or 16-bit displacement

Example

- MOV AX, [BX + SI + 05]; 8-bit displacement
- MOV AX, [BX + SI + 1235H]; 16-bit displacement

Other Addressing modes:

- **Program Memory Addressing Mode:** These types of addressing modes are used in branch instructions like JMP or CALL instruction.

  Example: JMP 0006H
- **Stack Memory Addressing Mode:** The stack memory addressing mode is used whenever you perform a push or pop operation.

  Example: PUSH BX

Instruction Set of 8086

Instructions are classified on the basis of functions they perform. They are categorized into the following main types:

- Data Transfer instruction
- Arithmetic Instructions
- Logical Instructions
- Rotate Instructions
- String Instructions