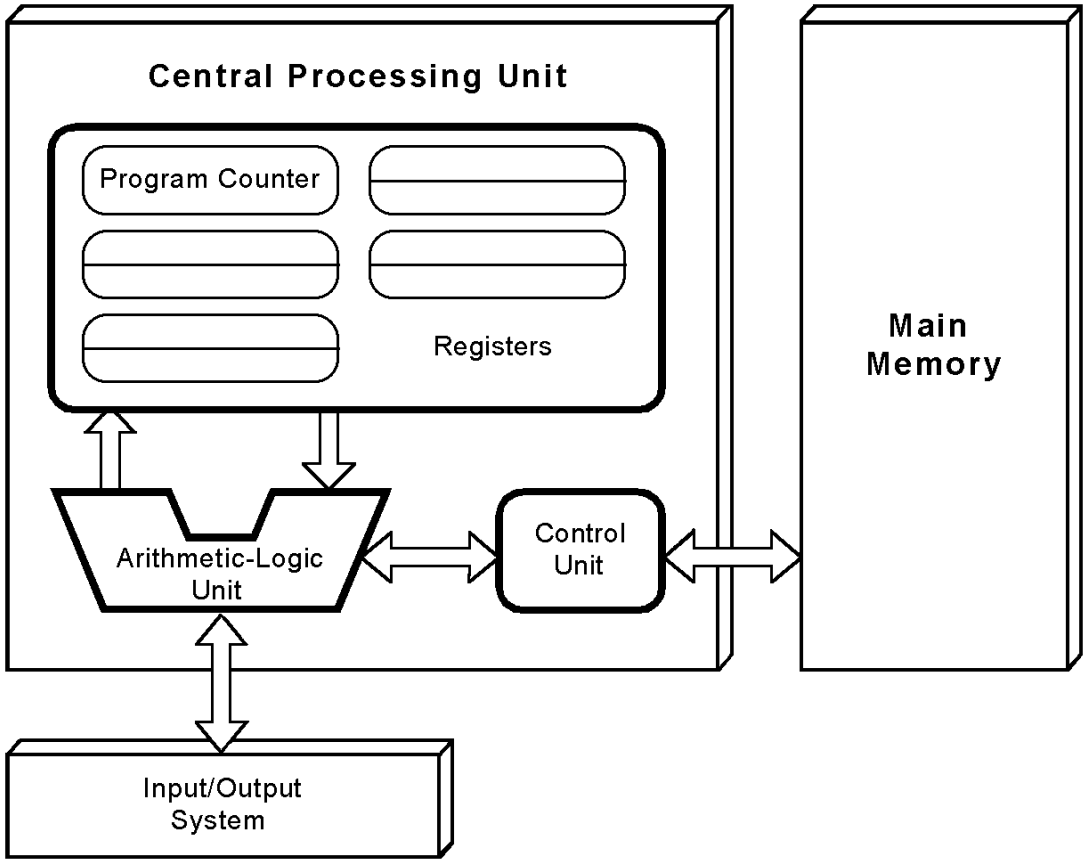
THE VON NEUMANN MODEL

Stored-program computers have become known as **von Neumann Architecture** systems. The von Neumann model is typical uni-processor computer system.

- Today's stored-program computers have the following characteristics:
  - Three hardware systems:
    - A central processing unit (CPU)
    - A main memory system
    - An I/O system
  - The capacity to carry out sequential instruction processing.
  - A single data path between the CPU and main memory.
    - This single path is known as the *von Neumann bottleneck*.

THE VON NEUMANN MODEL

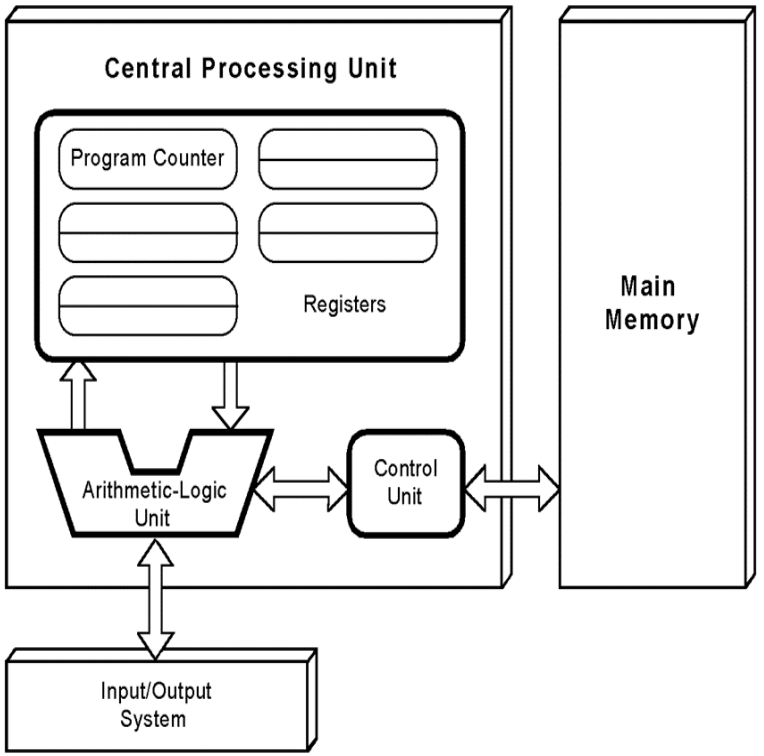
- **This is a general depiction of a von Neumann system:**
- **These computers employ a fetch-decode-execute cycle to run programs as follows . . .**



The memory unit is a single-port device consisting of a MAR and MDR. The memory cells are arranged in the form of several memory words, where each word is the unit of data that can be read or written. All the read and write operations on memory utilize the memory port.

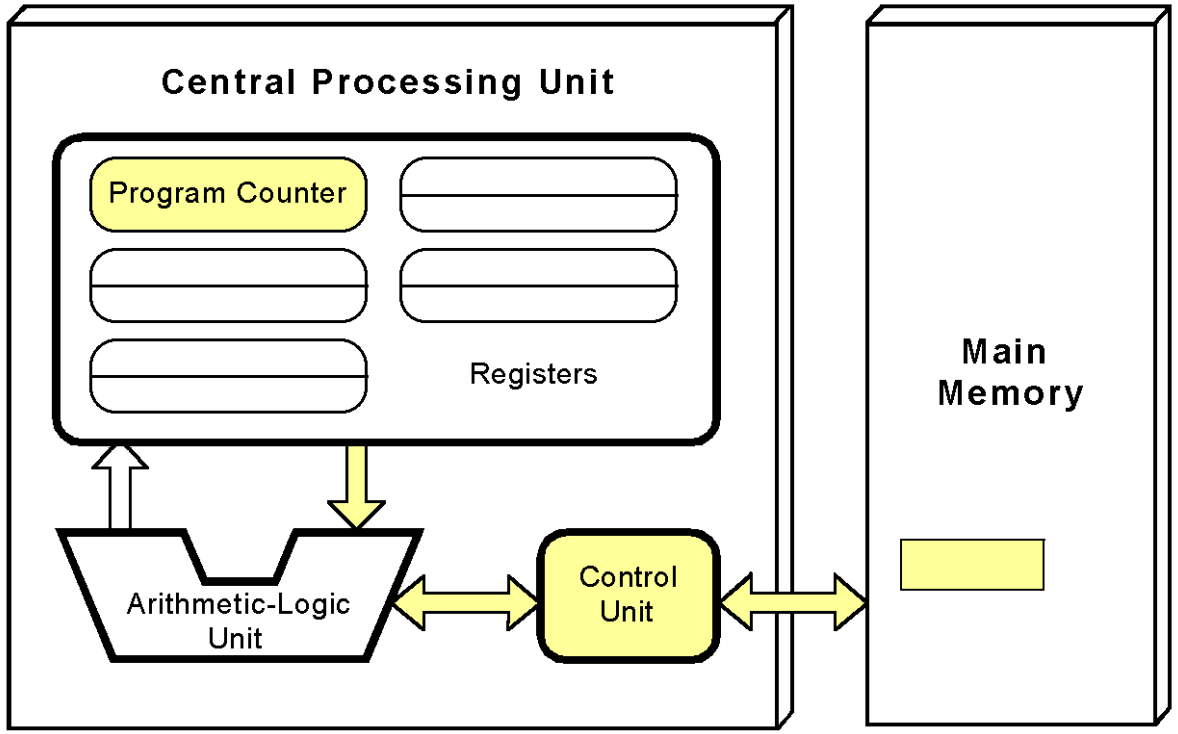
THE VON NEUMANN MODEL

- The ALU perform the arithmetic and logic operation on the data item in the accumulator (ACC) and MBR and ACC retain the results of such operation.
- The control unit consists of a program Counter (PC) that contain the address of the instruction to be fetched and an Instruction register (IR) into which the instructions are fetched from the memory for execution.
- In practice the I/O may also occur directly between the memory and I/O devices without utilizing any processor registers.
- The control unit manages this flow through the use of appropriate control signal.



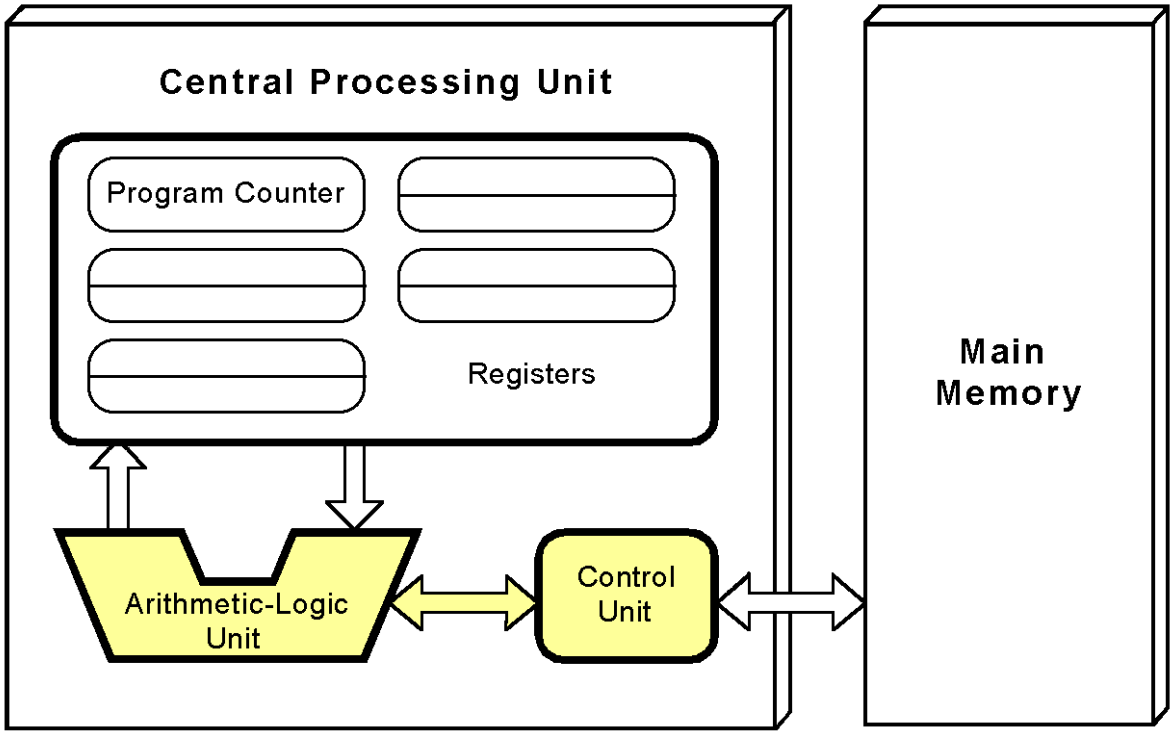
THE VON NEUMANN MODEL

- **The control unit fetches the next instruction from memory using the program counter to determine where the instruction is located.**



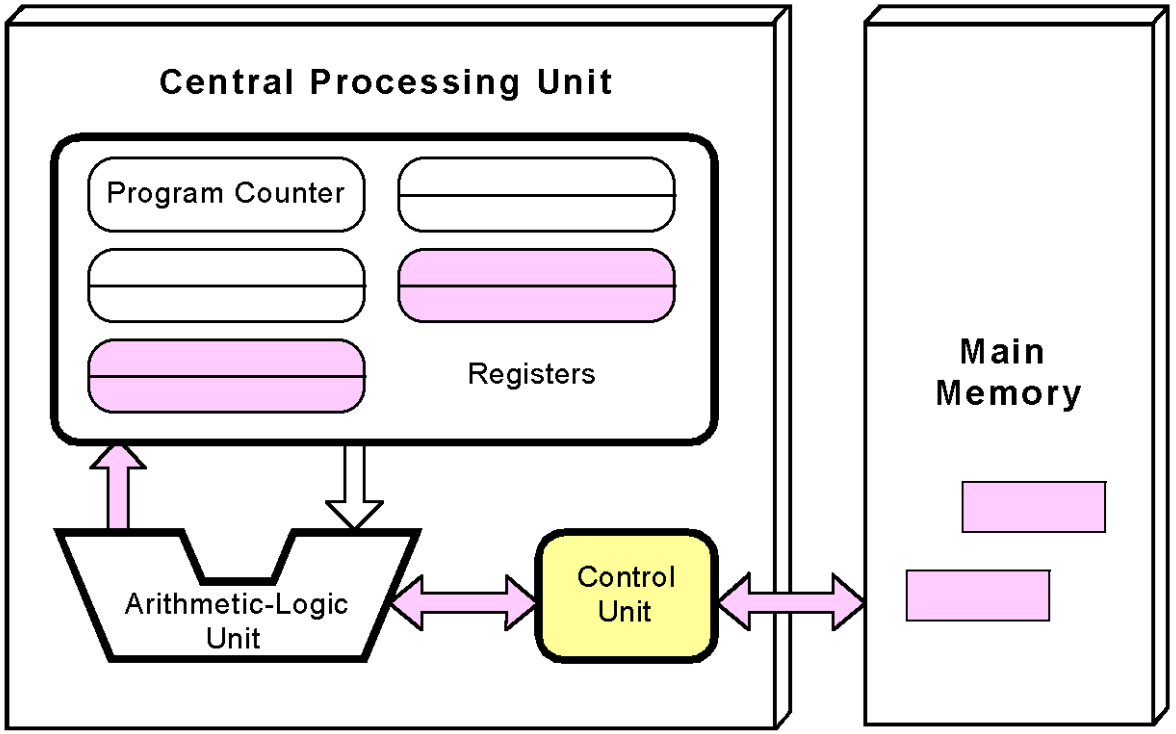
THE VON NEUMANN MODEL

- **The instruction is decoded into a language that the ALU can understand.**



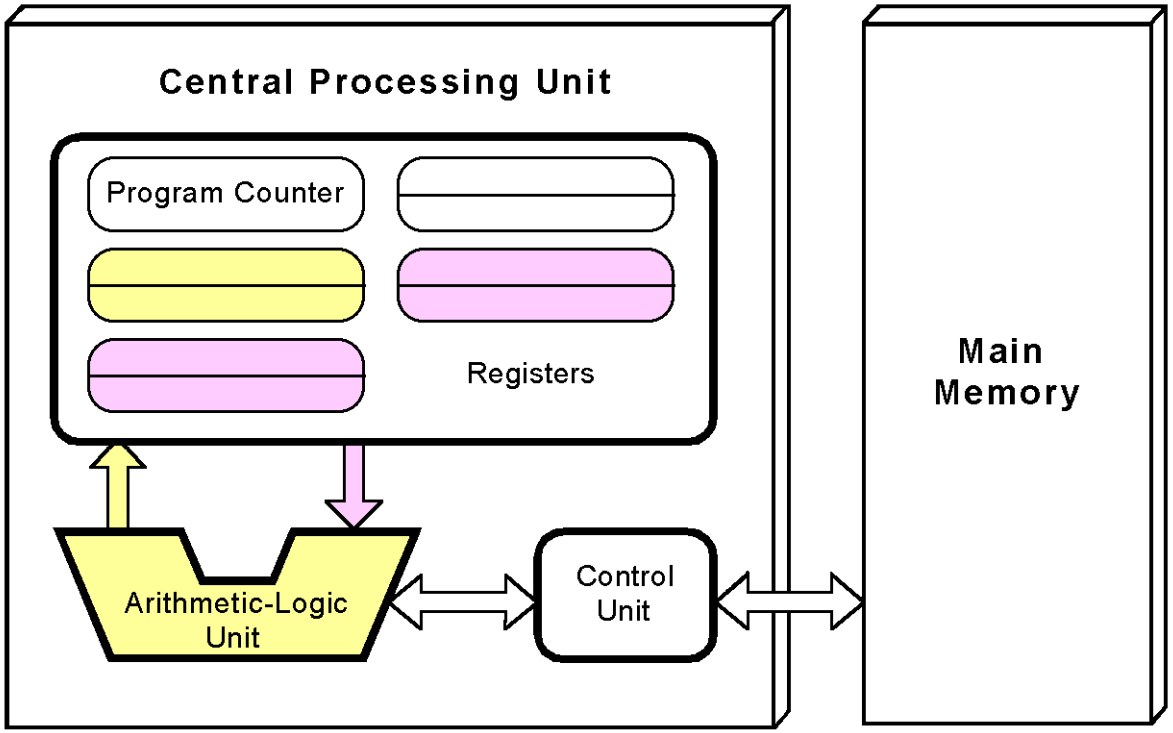
THE VON NEUMANN MODEL

- **Any data operands required to execute the instruction are fetched from memory and placed into registers within the CPU.**



THE VON NEUMANN MODEL

- **The ALU executes the instruction and places results in registers or memory.**



NON-VON NEUMANN MODELS

- Conventional stored-program computers have undergone many incremental improvements over the years.
- These improvements include adding specialized buses, floating-point units, and cache memories, to name only a few.
- But enormous improvements in computational power require departure from the classic von Neumann architecture.
- Adding processors is one approach.

NON-VON NEUMANN MODELS

- In the late 1960s, high-performance computer systems were equipped with dual processors to increase computational throughput.
- In the 1970s supercomputer systems were introduced with 32 processors.
- Supercomputers with 1,000 processors were built in the 1980s.
- In 1999, IBM announced its Blue Gene system containing over 1 million processors.
- Parallel processing is only one method of providing increased computational power.

Von Neumann and Harvard architectures

Von-Neuman	Harvard
First digital computer architecture. Introduced stored program concept	Computer architecture based on Harvard Mark 1
One memory module for data and instructions	Have different memory modules for data and instructions.
common bus for data and instructions	Individual buses for data and instructions
CPU takes 2 clocks to execute one instruction. Because fetch data before executing an instruction.	Can execute instruction one clock cycle.
CPU can not fetch instructions and data read/write at the same time.	CPU can not fetch instructions and data read/write at the same time.
Slow	Fast

REGISTERS

- It is a special temporary storage location (extremely fast memory spaces) within the CPU.
- Used to store the results of executions of the instructions in CPU.
- Registers quickly accept, store and transfer data and instructions that are being used immediately by the CPU.
- A Register can also be considered as a group of flip-flops with each flip-flop capable of storing one bit of information.
- A register with *n flip-flops* is capable of storing binary information of n-bits.

REGISTER ORGANIZATION

- Different computers have different set of register.
- They differ in the number of registers, register types, and the length of each register.
- All CPU registers has their unique identity and special role in computer system.
- **General-purpose registers** can be used for multiple purposes and assigned to a variety of functions by the programmer.
- **Special-purpose registers** are restricted to only specific functions. In some cases, some registers are used to hold data and cannot be used in calculations of operand addresses.
- **Address registers** may be dedicated to a particular addressing mode or may be used as address general purpose. Address registers must be long enough to hold the largest address.

- Another type of registers is used to hold processor status bits or flags. These bits are set by the CPU as the result of the execution of an operation.
- **Memory Access Registers:**
  - Memory Data Register (MDR)
  - Memory Address Register (MAR)

  The basic operations of memory are write operation and read operation, the registers MDR and MAR are respectively used for these operations.

  The MDR and MAR are used exclusively by the CPU. These registers are not directly accessible by the developers.

Write and Read operations

- In order to perform a write operation into a specified memory location, MDR and MAR are used as following way:
  - The word to be stored into memory location is first loaded by the CPU into MDR.
  - The address of the location into which the word is to be stored is loaded by the CPU into a MAR
  - A write signal is issued by the CPU
- Similarly, to perform read operation, the MDR and MAR used following way:
  - The address of the location from which the word is to be read is loaded into the MAR.
  - A read signal is issued by the CPU.
  - The required word will be loaded by the memory into the MDR ready for use by the CPU.

Instruction Fetching Registers

- There are two main registers are involved for fetching an instruction for execution:
  - Program Counter (PC)
  - Instruction Register (IR)
- Program Counter (PC) is used to keep the track of execution of the program. The PC contains the address of the next instruction to be fetched. The fetched instruction is loaded in the IR for execution.
- After a successful instruction fetch, the PC is updated to the point to the next instruction to be executed. PC also functions to count the number of instructions. The incrementation of PC depends on the type of architecture being used. If we are using 32-bit architecture, the PC gets incremented by 4 every time to fetch the next instruction.

- In computing, the instruction register (IR) or current instruction register (CIR) is the part of a CPU's control unit that holds the instruction currently being executed or decoded.
- In simple processors, each instruction to be executed is loaded into the instruction register, which holds it while it is decoded, prepared and ultimately executed, which can take several steps.

Condition Registers

- Condition registers, or the flags are used to maintain the status information. Some architecture contain a special program status word (PSW) register. The PSW contain bits that are set by the CPU to indicate the current status of an executing program.
- A condition register is a flag register which indicates some condition produced by the execution of an instruction or controls certain operations of the CPU.
- A 16 bit flag register in the CPU contains nine active flags out of that six conditional or status flag and three machine control flag.
- Conditional flags or status flag:
  - Carry flag (CF): indicate a carry after addition or a borrow after subtraction, also indicates error conditions.
  - Parity Flag (PF): is a logic “0” for odd parity and a logic “1” for even parity.
  - Auxiliary carry flag (AF): important for BCD addition and subtraction; hold a carry (borrow) after addition (subtraction) between bits position 3 and 4.
  - Zero Flag (ZF): indicates that the result of an arithmetic or logic operation is zero.
  - Sign Flag (SF): indicates arithmetic sign of the result after an arithmetic operation.
  - Overflow Flag (OF): a condition that occurs when signed numbers are added or subtracted. An overflow indicates that the result has exceeded the capacity of the machine.

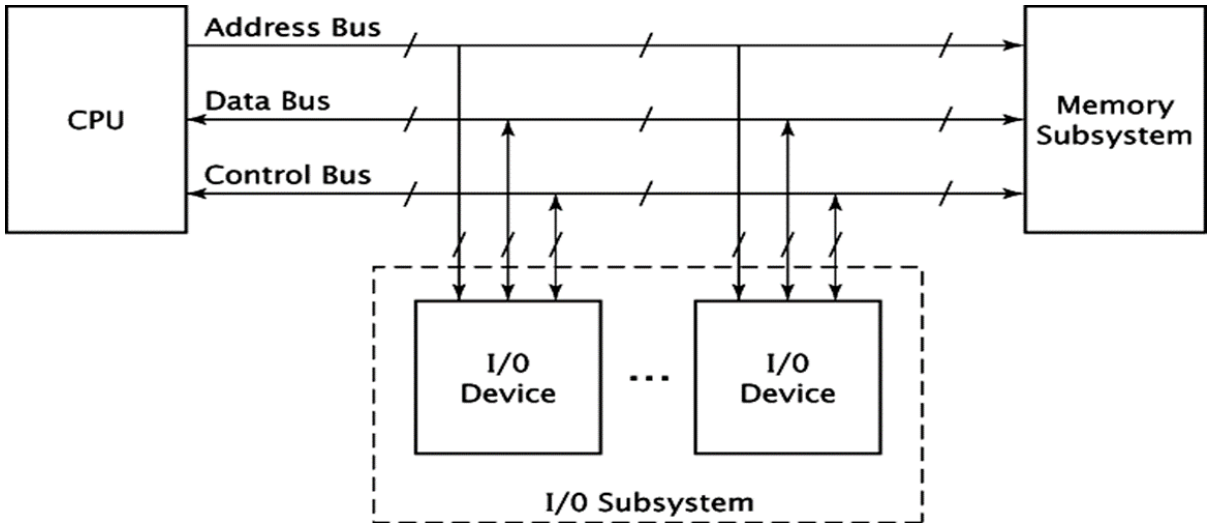
- Machine control flags: are deliberately set or reset with specific instructions you put in your program.
  - Trap Flag (TF): used for single stepping through a program
  - Interrupt Flag (IF): used to allow or prohibit the interruption of a program
  - Direction Flag (DF): used with string instruction

Special purpose Registers

- Index Register
  - Index register used in the index addressing mode, to obtain the address of the operand by adding the content of the register.
  - The Index register holds an address displacement.
  - Index addressing is indicated in the instruction by including the name of the index register in the parentheses and using the symbol X to indicate the constant to be added.
- Segment Pointers
  - In order to support segmentation, the address issued by the processor should consist of segment number (base) and a displacement (or an offset) within the segment. A segment Register holds the address of the base of the segment.
- Stack Pointers
  - Is used to indicate the stack location that can be addressed

INTERCONNECTIONS OF UNITS..

- Set of wires used for interconnection is known as system bus which carry group of bits (information) in a controlled manner.
- It is further divided into three logical units, namely the address bus, the data bus, and the control bus.



System buses

- Set of wires, that interconnects all the components (subsystems) of a computer
  - A source component sources out data onto the bus
  - A destination component inputs data from the bus
- May have a hierarchy of buses
  - Address, data and control buses to access memory and an I/O controller.
  - Second set of buses from I/O controller to attached devices/peripherals
  - Peripheral Component Interconnection (PCI) bus is an example of a very common local bus

ADDRESS BUS

- CPU reads/writes data from the memory by addressing a unique location; outputs the location of the data (aka address) on the address buss; memory uses this address to access the proper data
- Each I/O device (such as monitor, keypad, etc) has a unique address as well (or a range of addresses); when accessing a I/O device, CPU places its address on the address bus. Each device will detect if it is its own address and act accordingly
- The address bus is uni-directional, the CPU always sends out the address, no other devices will send an address.

Data bus

- When the CPU fetches data from memory, it first outputs the address on the address bus, then the memory outputs the data onto the data bus; the CPU reads the data from data bus
- When writing data onto the memory, the CPU outputs first the address on the address bus, then outputs the data onto the output bus; memory then reads and stores the data at the proper location
- Data bus is bi-directional – data is sent from the CPU to memory for a store command, from CPU to I/O for an output command, from memory to CPU for a load command, from I/O to CPU for an input.

Control Bus

- The control bus is responsible for making CPU, memory and I/O devices work together as a functional system, carrying signals that report the status (ready, not ready) of various units.
- The function of a control bus is to determine and instruct according to the operation type (Read or Write). For example, if the processor or an I/O device wants to read or write a value from memory, the control bus will specify it.

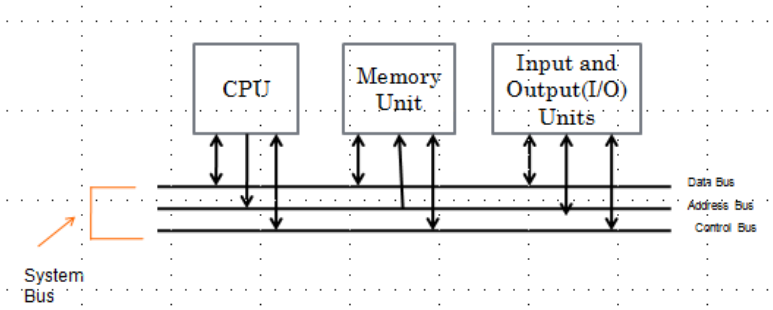
CONTROL BUS

- Address and data buses consist of *n* lines, which combine to transmit one *n* bit value; control bus is a collection of individual control signals
- These signals indicate whether the data is to be read into or written out the CPU, whether the CPU is accessing memory or an IO device, and whether the I/O device or memory is ready for the data transfer
- The control bus is bi-directional, the CPU sends out commands to memory (read/write) and I/O devices (input, output, possible fast forward, rewind, format) and requests (are you available?) and the devices can send responses (busy/available) and interrupts (calls for attention).

Processing of Information

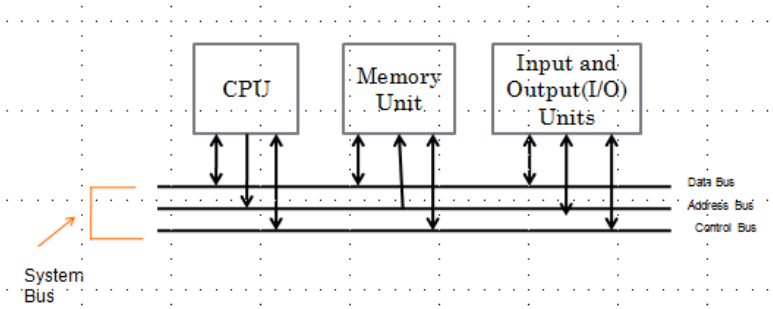
- The bus is common to all the units in the computer. Before sending some information on the bus, an unit should verify whether the bus is free or occupied with some communication started by some other unit.
- CPU is the bus master in a computer which decides who should control the bus when more than one unit wants the bus at the same time.
- An unit who needs the bus makes a request to the CPU and waits sanction. Till the CPU issues sanction, the requesting unit does not attempt to use the bus.

Interconnection of Computers Units via Bus



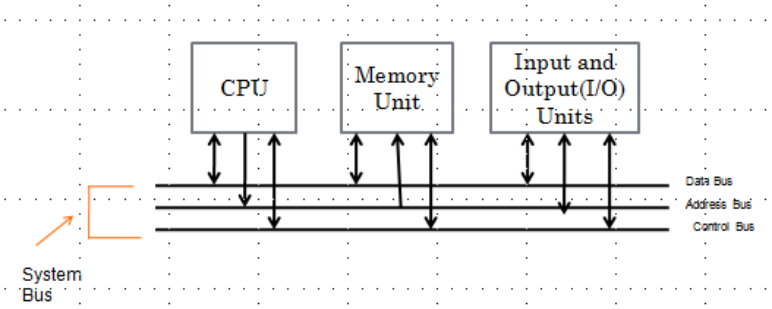
- Shows how the system bus interconnects the processor, memory and I/O devices.
- Both processor and memory units hold a bi-directional relationship with the control and data bus.

Interconnection of Computers Units via Bus



- In case of an address bus, the communication with processor and memory is unidirectional.
- Processor provides location of data (stored in the register) to be fetched from the memory to the address bus and the data carries the required data to the processor.

Interconnection of Computer Units via BUS



- I/O devices have a bi-directional relationship with the system bus.

Additionally, computers may have multiple buses

local bus – connects registers, ALU and control unit together (also an on-chip cache if there is one)

system bus – connects CPU to main memory

expansion or I/O bus – connects system bus to I/O devices