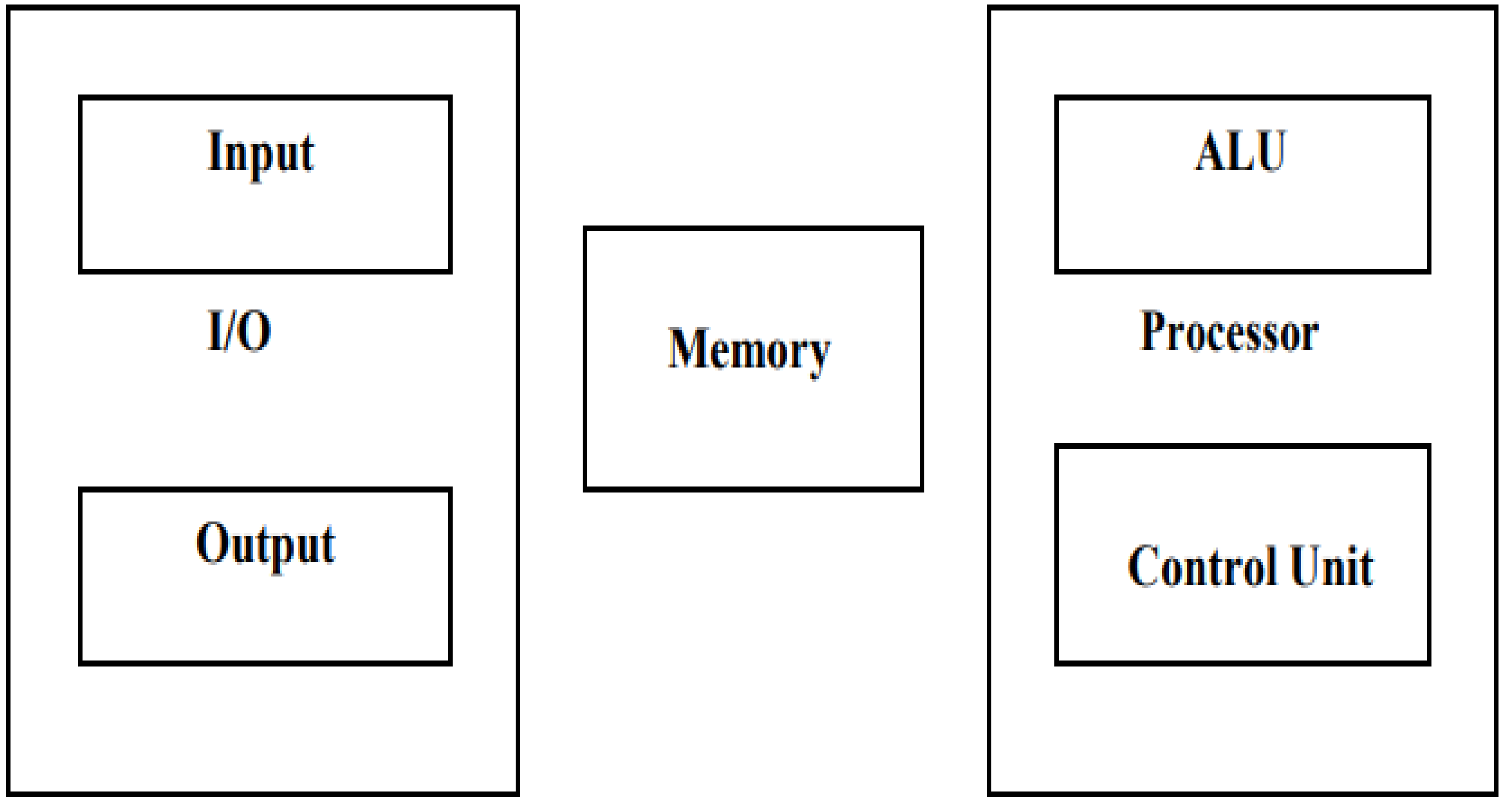
ECT 206 CAMC
Module 1

Algorithms for binary multiplication and division. Fixed and floating-point number representation. **Functional units of a computer, Von Neumann and Harvard computer architectures, CISC and RISC architectures**. **Processor Architecture – General internal architecture, Address bus, Data bus, control bus. Register set – status register, accumulator, program counter, stack pointer, general purpose registers. Processor operation – instruction cycle, instruction fetch, instruction decode, instruction execute, timing response, instruction sequencing and execution (basic concepts, datapath**

Functional Units of a Computer

- A computer can be defined as a fast electronic calculating machine that accepts the (data) digitized input information process it as per the list of internally stored instructions and produces the resulting information
- List of instructions are called programs & internal storage is called computer memory.
- A computer consists of five functionally independent main parts
  - Input
  - Memory
  - Arithmetic Logic Unit (ALU)
  - Output
  - Control unit.



Functional units of computer

Input unit

- The source program/high level language program/coded information/simply data is fed to a computer through input devices keyboard is a most common type.
- Whenever a key is pressed, one corresponding word or number is translated into its equivalent binary code over a cable & fed either to memory or processor.
- Joysticks, trackballs, mouse, scanners etc are other input devices.

Memory unit

- Its function is to store programs and data.
- It is basically to two types

    1. Primary memory

    2. Secondary memory

Primary memory

- Is the one exclusively associated with the processor and operates at the electronics speeds programs must be stored in this memory while they are being executed.
- The memory contains a large number of semiconductors storage cells.
- Each capable of storing one bit of information.
- These are processed in a group of fixed site called word.

- To provide easy access to a word in memory, a distinct address is associated with each word location.
- Addresses are numbers that identify memory location.
- Number of bits in each word is called word length of the computer.
- Programs must reside in the memory during execution.
- Instructions and data can be written into the memory or read out under the control of processor.

- Memory in which any location can be reached in a short and fixed amount of time after specifying its address is called random-access memory (RAM).
- The time required to access one word in called memory access time.
- Memory which is only readable by the user and contents of which can’t be altered is called read only memory (ROM) it contains operating system.
- Caches are the small fast RAM units, which are coupled with the processor and are often contained on the same IC chip to achieve high performance.
- Although primary storage is essential it tends to be expensive.

Secondary memory

- Is used where large amounts of data & programs have to be stored, particularly information that is accessed infrequently.
- Examples: - Magnetic disks & tapes, optical disks (ie CD-ROM’s), floppies etc.,

Arithmetic logic unit (ALU)

- Most of the computer operators are executed in ALU of the processor like addition, subtraction, division, multiplication, etc.
- The operands are brought into the ALU from memory and stored in high speed storage elements called register.
- Then according to the instructions the operation is performed in the required sequence.
- The control and the ALU are may times faster than other devices connected to a computer system.
- This enables a single processor to control a number of external devices such as key boards, displays, magnetic and optical disks, sensors and other mechanical controllers.

Output unit

- These actually are the counterparts of input unit.
- Its basic function is to send the processed results to the outside world.
- Examples:- Printer, speakers, monitor etc.

Control unit

- It effectively is the nerve center that sends signals to other units and senses their states.
- The actual timing signals that govern the transfer of data between input unit, processor, memory and output unit are generated by the control unit.

Computer Architecture

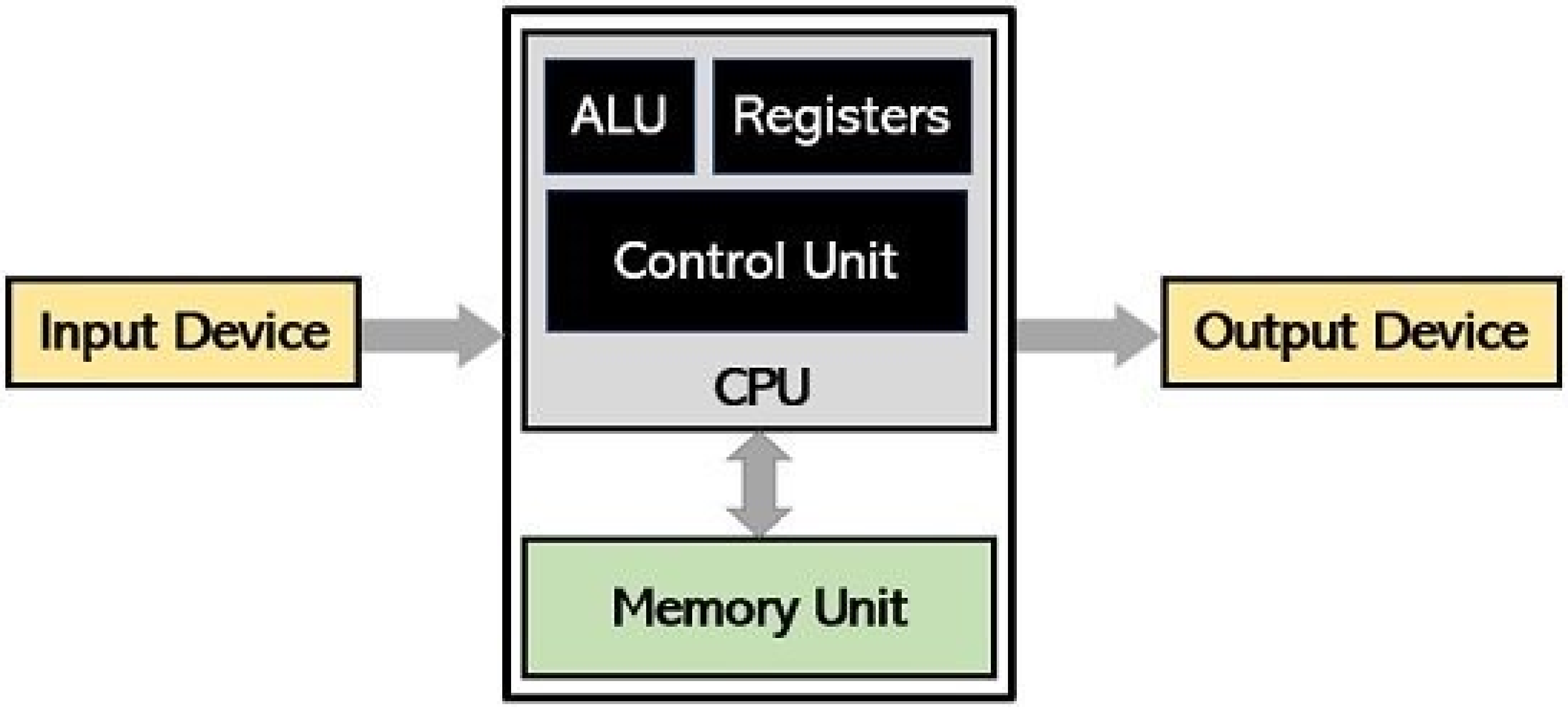
- Computer architecture is *a set of rules and methods that describe the functionality, organization, and implementation of computer systems*.
- There are three categories of computer architecture
  – System Design
  – Instruction Set Architecture(ISA)
  – Micro Architecture

- **Instruction set architecture (ISA):** defines the machine code that a processor reads and acts upon as well as the word size, memory address modes, processor registers, and data type.
- **Microarchitecture:** also known as "**computer organization**", this describes how a particular processor will implement the ISA. The size of a computer's CPU cache for instance, is an issue that generally has nothing to do with the ISA.
- **Systems design:** includes all of the other hardware components within a computing system, such as data processing other than the CPU (e.g., direct memory access), virtualization, and multiprocessing

- There are two basic models of computer architecture
  - Von Neumann Architecture
  - Harvard Architecture
- The significant difference between Von Neumann and Harvard architecture arises according to the way the CPU is separated from the memory.
- In both these architecture, two different ways are used by which memory is accessed by the CPU.

Von Neumann Architecture

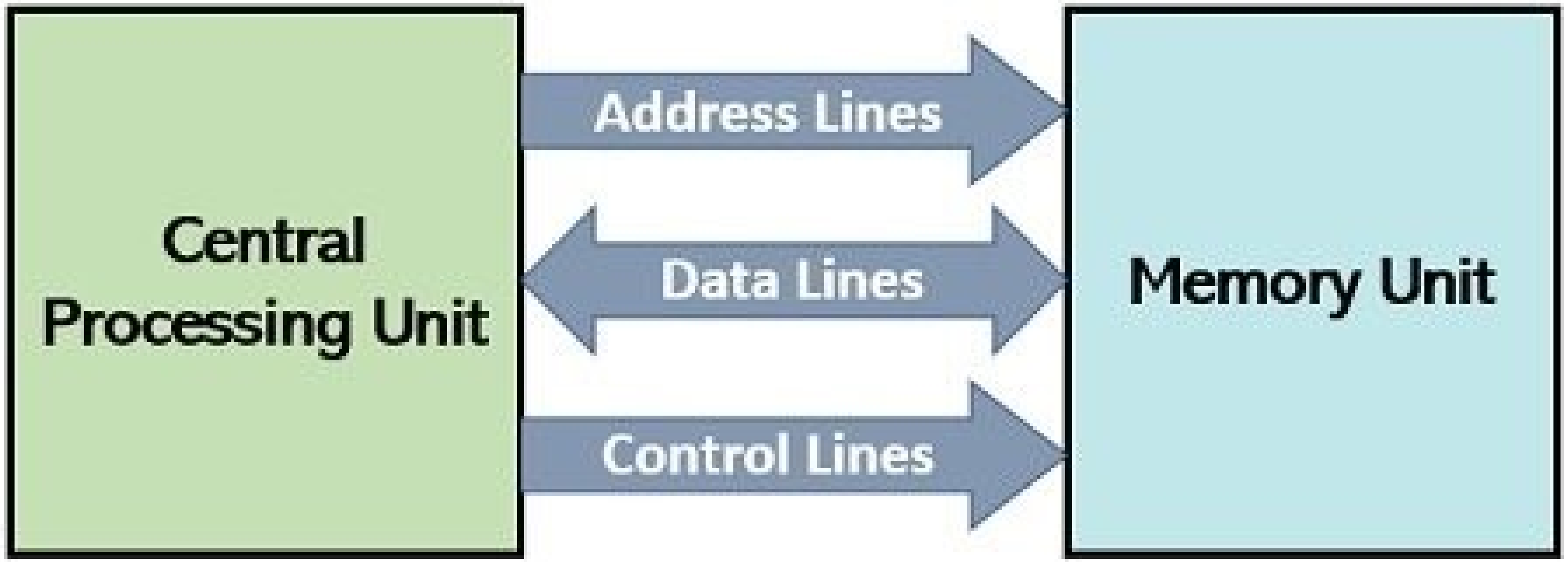
- Also known as Princeton architecture
- Proposed by John Von Neumann in 1945
- A computer architecture that uses a single memory unit within which both data and instructions get stored is known as Von Neumann Architecture
- It is an architecture where the data and programs are subjected to shared memory.



Von Neumann Architecture

- There are three major components that constitute this architecture:
  - CPU
  - Memory
  - I/O interface

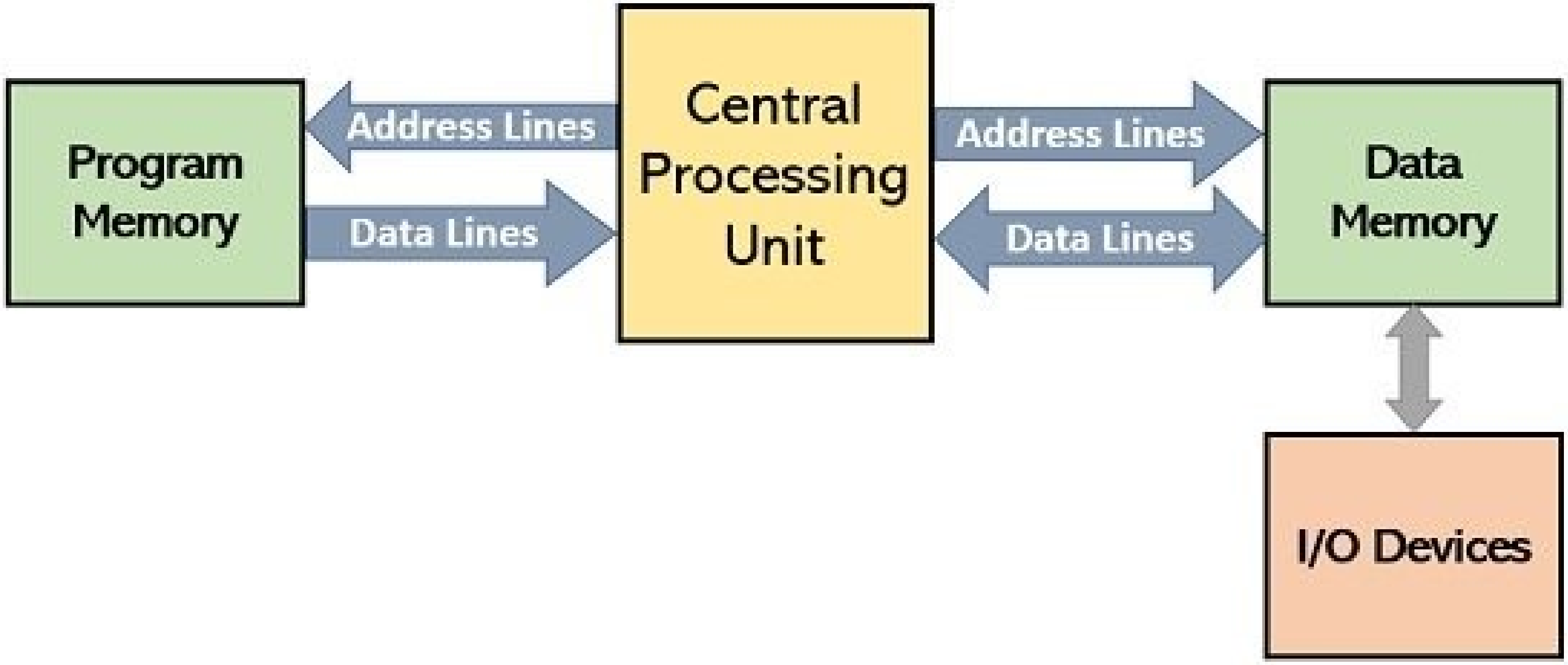
- In this architecture, data and instructions both reside in a single memory unit hence a single set of buses is used by the CPU to access the memory.
- After the execution of the program, data gets stored in memory from where it is provided to output devices


Central Processing Unit
Address Lines
Data Lines
Control Lines
Memory Unit

Harvard Architecture

- A computer architecture where the memory unit is divided into two parts for individually storing data and instructions is known as Harvard architecture.
- This means, unlike Von Neumann architecture, here data memory and instruction memory is in separate format.

- In this architecture, the CPU operates in a somewhat similar manner as in Von Neumann architecture.
- But as here two separate memory units are used thus separates buses are used for data transferring and instruction fetching.
- Thus, execution speed is very much faster than Von Neumann architecture



Harvard Architecture

- Efficient resource utilization occurs as instructions bits are sometimes more than the data bits thereby permitting different cell sizes.
- Also, the use of separate memories for both data and instructions helps in the minimization of execution time.
- However, in the Harvard model, the central processing unit present must be more efficient so that it can handle two sets of buses and allows simultaneous data transfer and instruction fetching.

Basis for Comparison	Von Neumann Architecture	Harvard Architecture
Basic	Data and instructions reside within a single memory unit.	Data and instruction are provided 2 different memory units.
Memory system	Single	Dual
Required space	Less	Comparatively more
Set of address/ data bus	One	Two
Development cost	Low	Comparatively more
Efficiency	Less	More

Execution speed	Slow	Comparatively fast
Operation	Simple	Complex
Performance offered	Low	Comparatively high
Clock cycle	Single instruction is executed in minimum two clock cycles.	Single instruction is executed in one clock cycle.
Feature	Data transfer and instruction fetching do not occur simultaneously.	Data transfer and instruction fetch take place at the same time.
Space utilization	Good	Not so good
Applications	PCs, workstations, notebooks, etc.	Microcontrollers, digital signal processing, etc.

ISA

- ISA describes the abstract model of a computer.
- It describe how an instruction is going to execute within its architecture.
- ISA defines the supported instructions, data types, registers, the hardware support for managing main memory, fundamental features (such as the memory consistency, addressing modes, virtual memory), and the input/output model of a family of implementations of the ISA.

- Based on ISA there are two models of computer architecture
  - Complex Instruction Set Computer(CISC)
  - Reduced Instruction Set Computer(RISC)

CISC

- It is a computer in which single instructions can execute several low-level operations such as load from memory, an arithmetic operation and a memory store.
- They are capable of multi-step operations or addressing modes within single instructions.
- Well known microprocessors and microcontrollers.
  - Motorola 6800, 6809 and 68000-families
  - the Intel 8080, iAPX432 and x86-family
  - the Zilog Z80, Z8 and Z8000-families
  - the National Semiconductor 32016 and NS320xx-line
  - the MOS Technology 6502-family
  - the Intel 8051-family; and others.

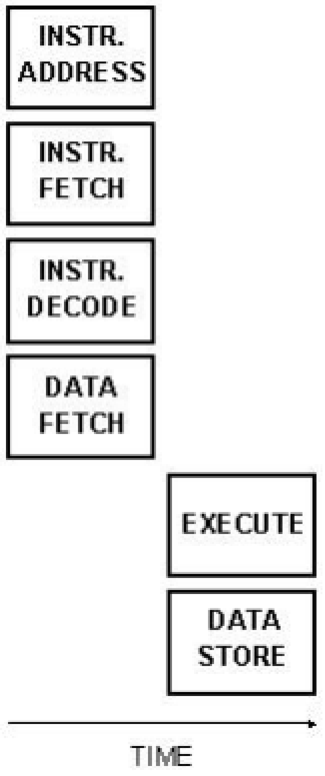
- It is a computer in which single instructions can execute several low-level operations
- They are capable of multi-step operations or addressing modes within single instructions.
- Has a complex instruction set.
- Variable length encoding of instructions.

- Instruction execution takes a varying number of clock cycles.
- Large number of addressing modes for the operations and instructions, the CISC computer generally requires fewer instructions than – RISC computers to perform the computation. E.g. MUL
- Programs writing for CISC architecture tend to take less space in memory

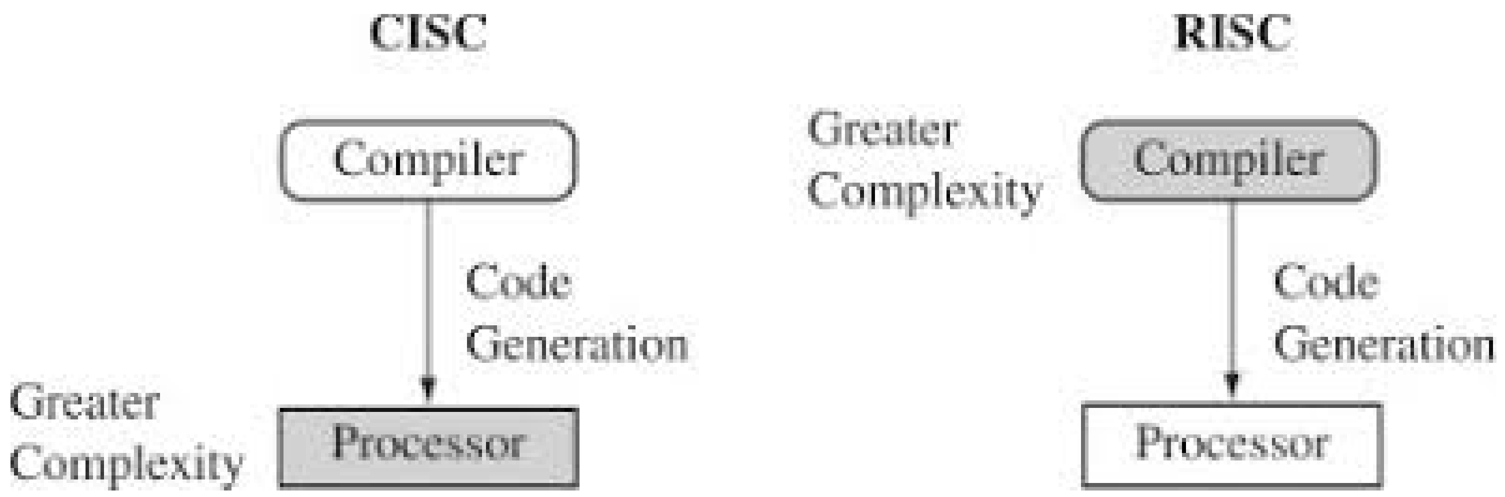
- Arithmetic and other instructions may read inputs from or write their outputs to the memory system, in addition to use of GPRs register file by arithmetic and other instructions
- Since less program memory size, its execution need less RAM memory.

- Hardware required to implement the CISC processor is More complex, since it would have to be able to fetch instruction operands from memory, so the CISC processor would probably have a longer cycle time (or would require more cycles to execute each instruction) than the RISC processor

- One instruction in a CISC architecture for ADD (r1), (r2), (r3) without load-store architecture
- Assume – that the appropriate memory addresses are present in r1, r2, and r3 at the start of the instruction sequence
- Four instructions are required to implement the same function
- LD r4, (r2)
- LD r5, (r3)
- ADD r6, r4, r5
- ST (r1), r6



ure 3 **<u>Typical CISC Architecture – Stack Design</u>**


CISC
Compiler
Code Generation
Greater Complexity
Processor
RISC
Greater Complexity
Compiler
Code Generation
Processor

RISC

- A Reduced instruction set computer, or RISC is a computer with a small, highly optimized set of instructions, rather than the more specialized set often found in other types of architecture

- One Cycle Execution Time
- RISC processors have a CPI (Clock Per Instruction) of one cycle.
- This is due to the optimization of each instruction on the CPU
- **Pipelining :-** A technique that allows for simultaneous execution of parts or stages of instructions to more efficiently process instructions.
- RISC has a performance oriented architecture based on exploitation of parallelism through pipelining

- Five pipeline stages are

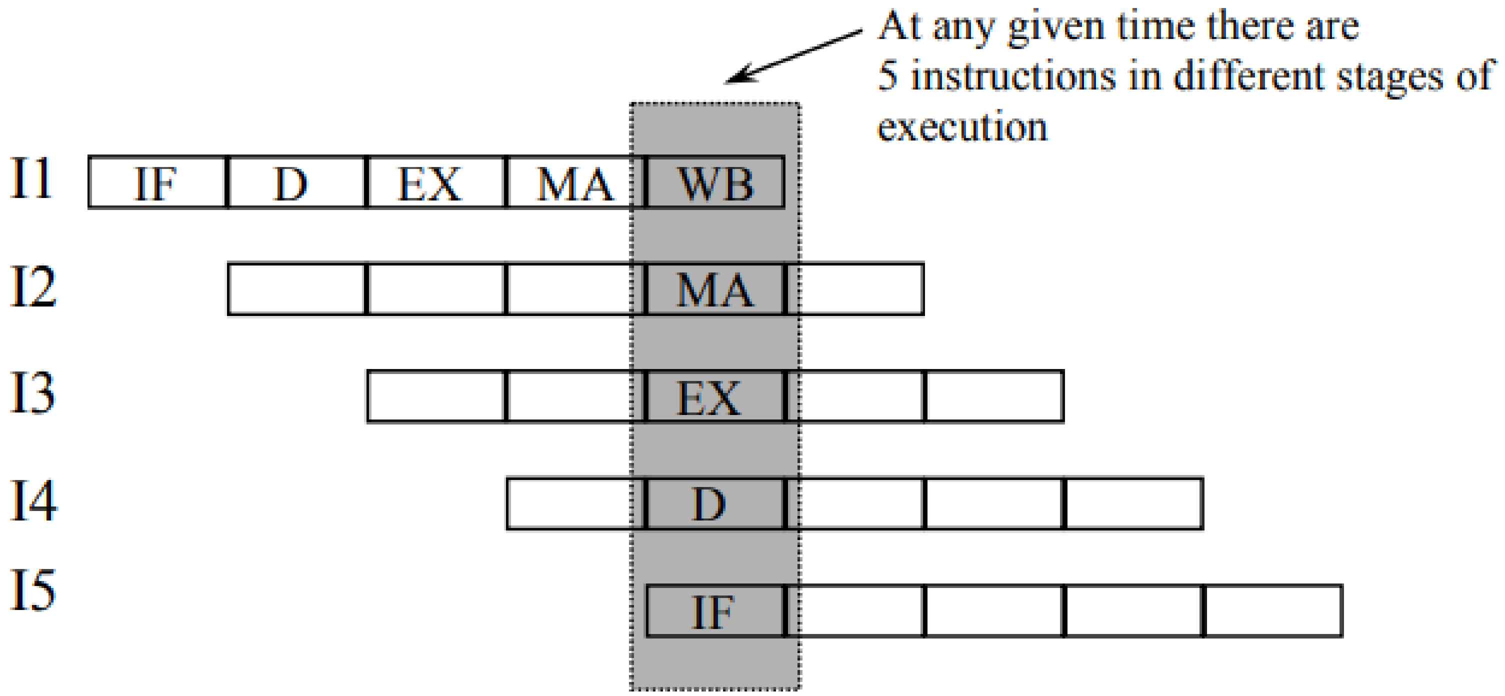
    IF – Instruction Fetch

    ID – Instruction Decode

    EX – Execute

    MA – Memory Access

    WB – Write Back


At any given time there are 5 instructions in different stages of execution
I1
IF
D
EX
MA
WB
I2
MA
I3
EX
I4
D
I5
IF

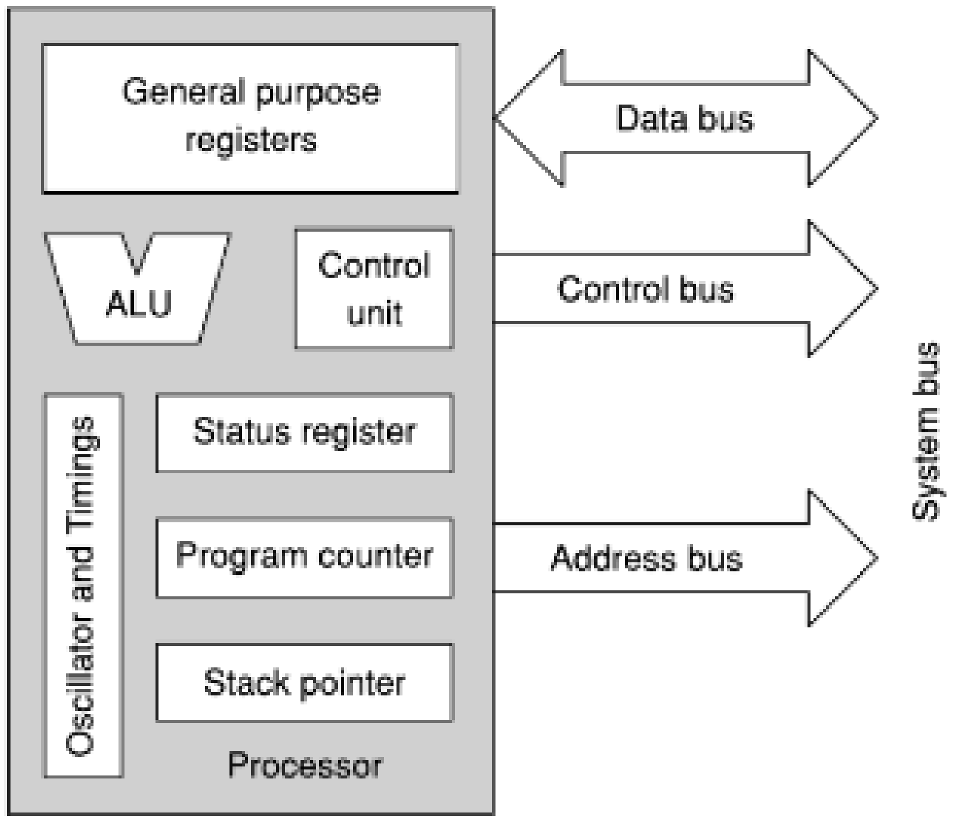
Advantages

- Large Number of Registers .
  - The RISC design philosophy generally incorporates a larger number of registers to prevent in large amount of interactions with memory.
- RISC is architecture having Load/Store architecture
  - Load-store architecture means that arithmetic and logic instructions do not use operand at the memory but operands must first load in the registers
- One instruction per cycle execution rate, CPI-1.0
  - Possible only through the use of pipelining
- Optimized Compiler
  - Close coupling between the architecture and the compiler. Compiler “knows” about the pipeline.
- Harvard Architecture
  - Separation of instruction and data cache resulting in increased memory bandwidth

Negatives

- An operation might need two, three or more instructions to accomplish a task
- More memory access might be needed
- Execution speed may be reduced in certain applications.
- Usually leads to longer programs, which needs larger memory to space to store.
- Difficult to program in machine level and assembly level(More time consuming)

CISC	RISC
Larger number of instructions – from 120 to 350	Relatively fewer instructions – less than 100
Employs a variety of data types and larger number of addressing modes	Relatively fewer addressing modes
Variable length instruction format	Fixed length instructions, usually 32 bits, easy to decode instructions
Instructions manipulate operands residing in memory	Mostly register to register operations
Small code sizes, high cycles per second	Low cycles per second, large code sizes
Memory to memory : "LOAD" and "STORE" incorporated in instructions	Register to register : "LOAD" and "STORE" are independent instructions
The number of cycles per instructions (CPI) varies from 1-20 depending on the type of instructions	The number of CPI is one as it uses pipelining
GPRs varies from 8-32, but no support is available for parameter passing and functions call	Large number of GPRs are available that are primarily used for global registers and support is available for parameter passing and functions call
Micro programmed control unit	Hardwired control unit
Emphasis on hardware	Emphasis on software



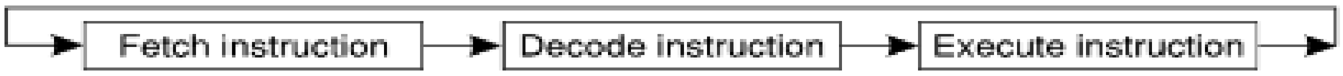
General internal architecture of a processor

General Internal Architecture

- **ALU** – Performs all arithmetic and logical operations.
- **General Purpose Registers** – Every processor offers a set of general purpose registers for various storage and operations
- **Status Register** – Accommodates the status of different arithmetic and logical operations, which might be necessary for conditional branching.
- **Program Counter** – Holds the address of the next instruction word to be fetched from external memory.

- **Stack Pointer** – Indicates the address of the stack-top
- **Control Unit** – Responsible for generating all control signals and general working of the processor. This is achieved by the instructions with the help of internal clock, which is maintained by **oscillator unit**
- **Oscillator and Timing Circuit** –
  - The timing and oscillatory operations are performed here
  - **The internal clock signal is generated with the help of this block**

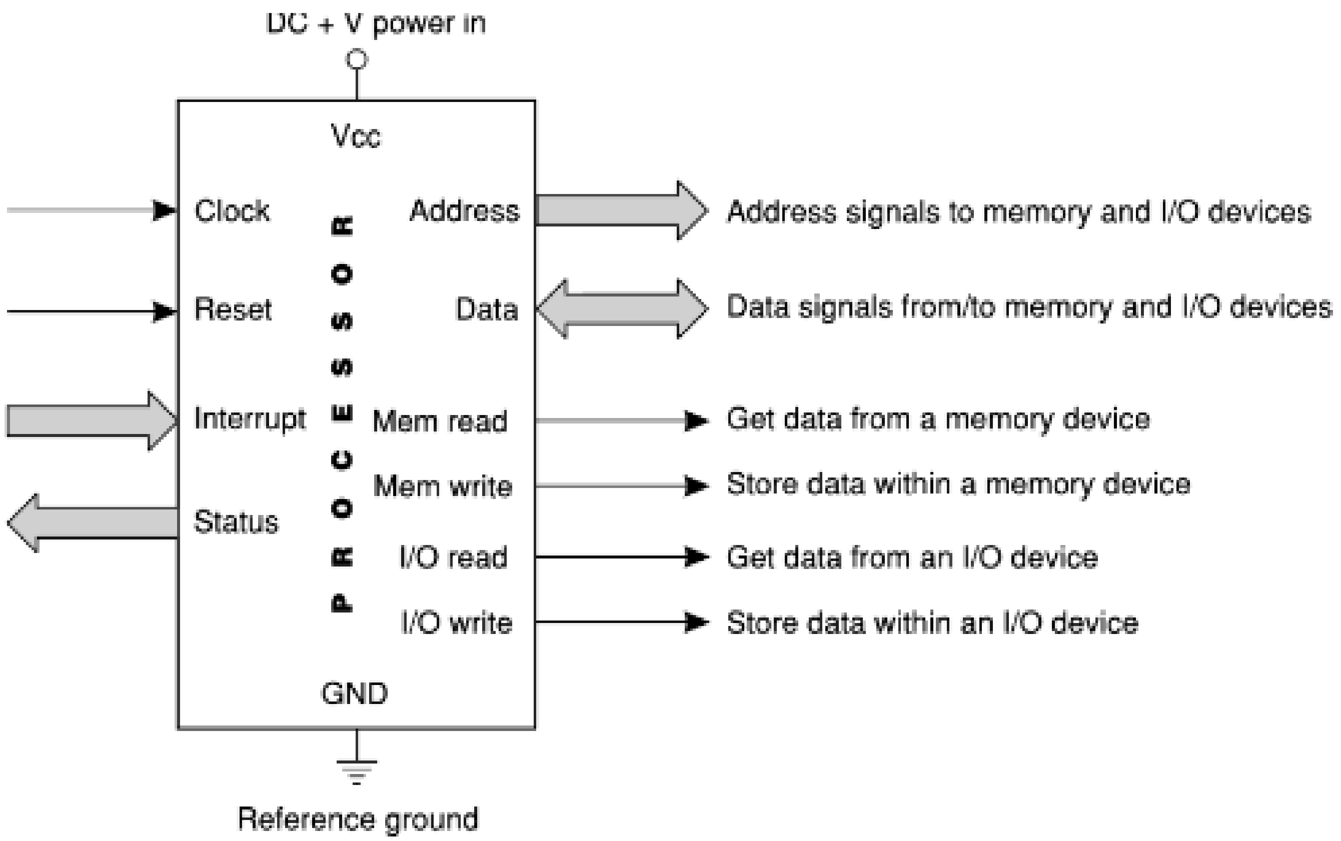
- The architectural details of a processor are meant for executing a software.
- Both hardware and software must be dealt concurrently for any computer or any processor.
- Only hardware or only software would not be able to achieve any tangible outcome.
- The basic duty of any processor is to fetch, decode and execute instructions as long as it is powered on
- Unless it is a microcontroller, these instructions are available outside the physical boundary of the processor, within memory chips(ICs).
- These memory chips are electrically connected with the processor through a bunch of wires designated as the bus



Basic function of any processor

- Apart from fetching the opcode of executable instructions, sometimes it might be necessary for the processor to load or store operands in the external memory, if indicated so by the on-going instruction.
- Generally, the operation of a processor is sequential, which is diverted to another sequence due to conditional branching, subroutine calls and returning from subroutines, or to respond against any eventful external interrupt signal.
- Memory devices are not the only category of peripheral devices necessary for a processor to be function, there are input/output devices also connected with the processor through the bus

- Every processor offers three major types of bus.
- Address Bus – **Unidirectional**
  - It carries address signals from processor to all external devices around it, memory and I/O.
- Data Bus – **Bi-directional**
  - Since data must come in and also go out of the processor
- Control Bus
- - Most of the control signals also move out of the processor.


DC + V power in
Vcc
Clock
Reset
Interrupt
Status
PROCESSOR
Address
Data
Mem read
Mem write
I/O read
I/O write
GND
Address signals to memory and I/O devices
Data signals from/to memory and I/O devices
Get data from a memory device
Store data within a memory device
Get data from an I/O device
Store data within an I/O device
Reference ground

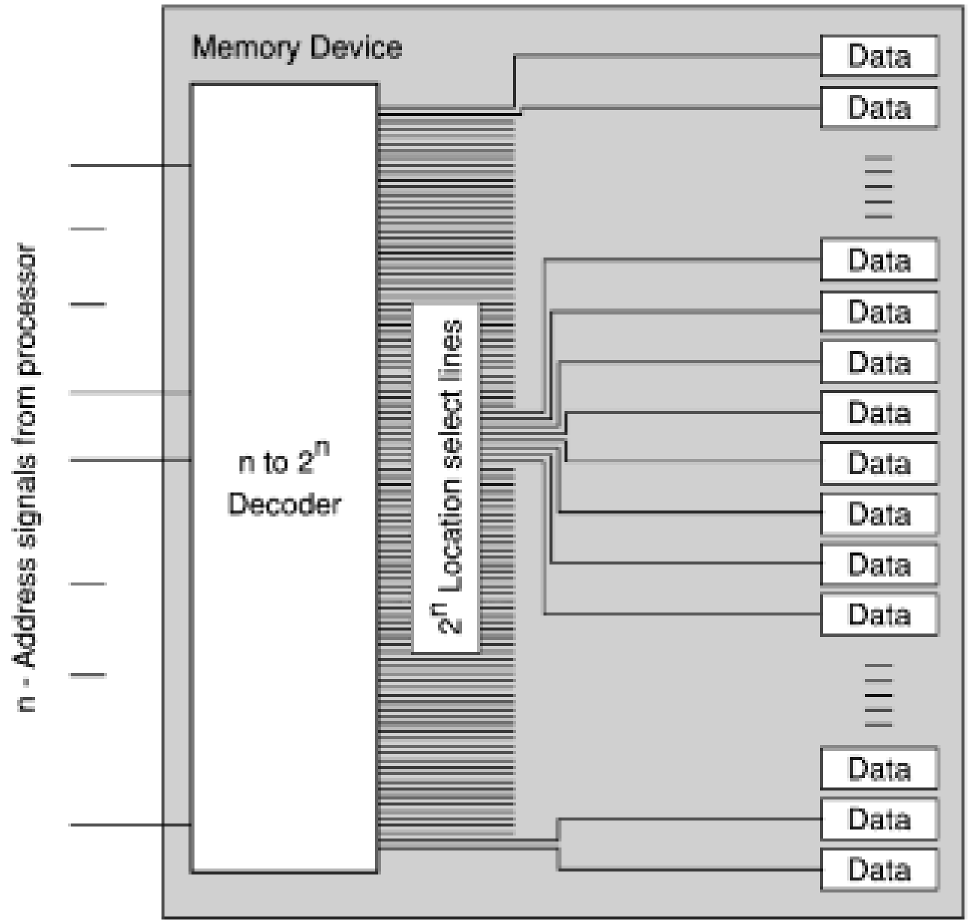
External signals of a generic processor (Note: the bus having multiple signals is shown by double lined arrow)

Address Bus and Addressing

- The width of address bus or number of address lines available from any processor indicates its maximum memory size handling capability.
- The number of memory locations (bytes or words as the case might be)addressable by n address lines is $2^{n.}$

  i.e. **16 address lines – 2^{16} - 64K locations**

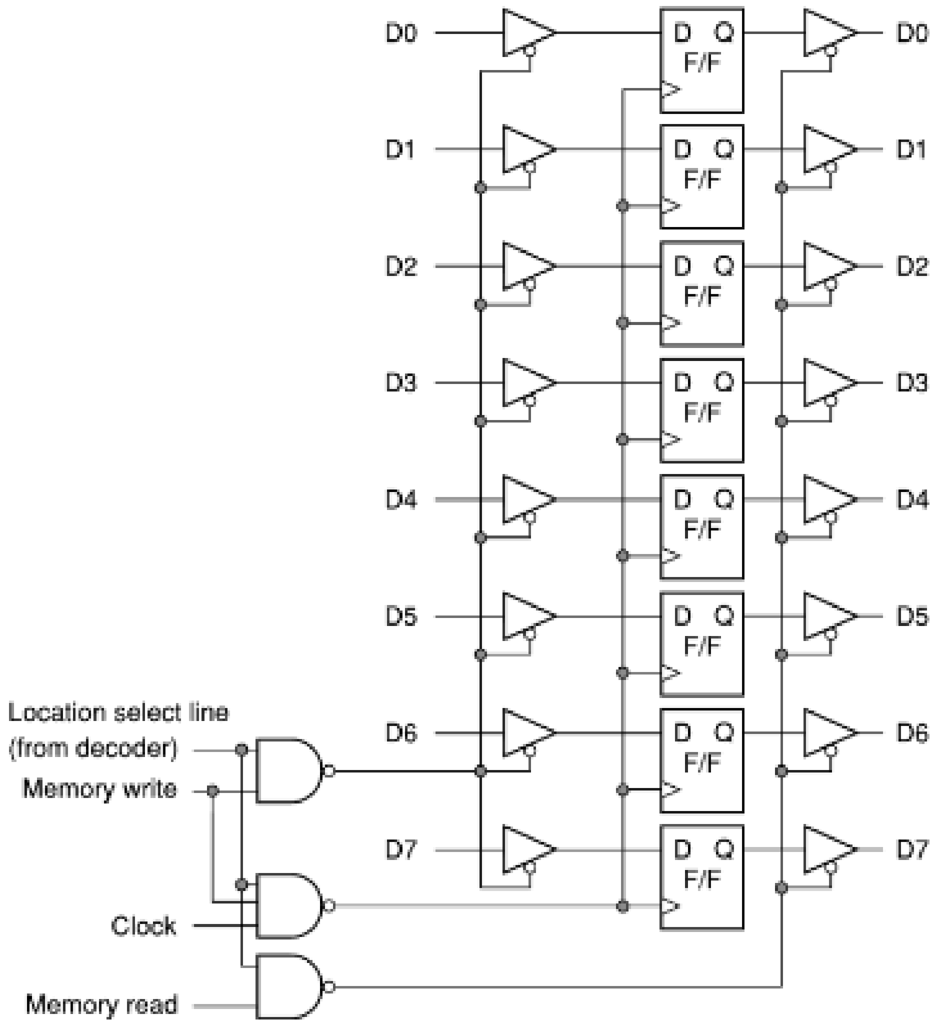
  20 address lines – 2^{20} – 1M locations
- Address bus helps in locating any desired data with any memory or I/O device.
- These address signals are decoded by a decoder inside the memory or I/O device to target the desired location**.**


Memory Device
n - Address signals from processor
n to 2^n Decoder
2^n Location select lines
Data

Addressing of memory location by the processor

Data Bus and Data Flow Control

- The width of data bus of any processor indicates its simultaneous handling capability of the maximum number of bits.
- Generally, a processor is designated by its data bus width.
- E.g.: 8-bit processor – capable of communicating 8-bit of data at the same time or having an 8-bit data bus.
- 16 –bit processor has 16 parallel data lines for data communications.
- The flow of data is bi-directional, depending upon whether the processor is interested in reading from or writing into the device(memory or I/O).
- This intention of the processor is expressed through its control signals(read and write). Depending upon this indication(read or write), the device(memory or I/O) enables the appropriate 3-state buffer to allow the flow of data signals from the data location already selected by address signals.
- The identical type of 3-state buffers is also present at the processor end in its data bus.



5 Data flow mechanism between memory and processor

Control Bus

- Number and functions of control signals, constituting the control bus varies widely with the processor itself.
- However, two of its important signals are READ and WRITE.
- Condition of these control signals indicate whether the present operation, intended by the processor is expecting the data in (READ) or sending the data out (WRITE).
- As the processor is to interact with two types of devices, memory and I/O, in general four read/write signals are offered.
- A few status signals are also available from the processor, apart from power input signals.
- Two more input signals are essential for all processors, namely clock and reset.
- Apart from these, a few external interrupt input signals are also provided in all processors.
- The purpose of all these signals is to execute any program. The programs are composed of individual instructions.

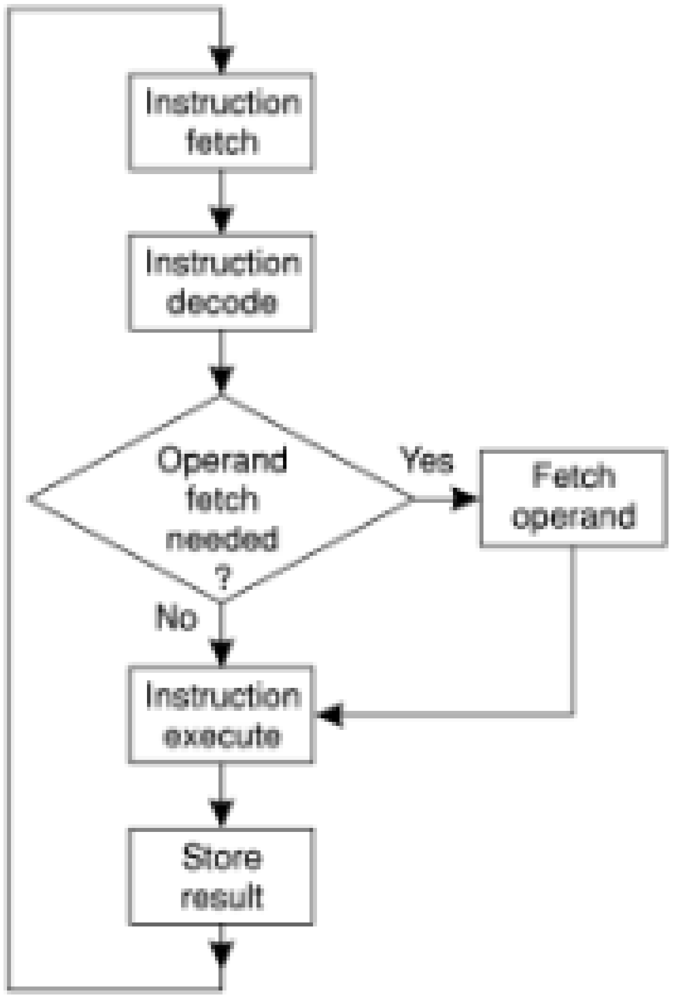
Processor Operation

- The job of the processor is to execute programs, which are composed of multiple instructions.
- Instructions executable by the processor are always in machine code.
- Programs developed with High Level Language(HLL)instructions are first changed to this machine code, understandable by the processor.
- Machine code instructions are extremely primitive and simple
- E.g. Copy a data byte from external memory to internal register or vice versa
- Add two number available within the processor registers
- If the result of subtraction is zero, then skip next three

Instruction Cycle

- To execute any type of instruction including those that are cited above , the processor should perform the following steps
- Fetch
- Decode
- Execute

- **Combination of these steps is known as an instruction cycle**.

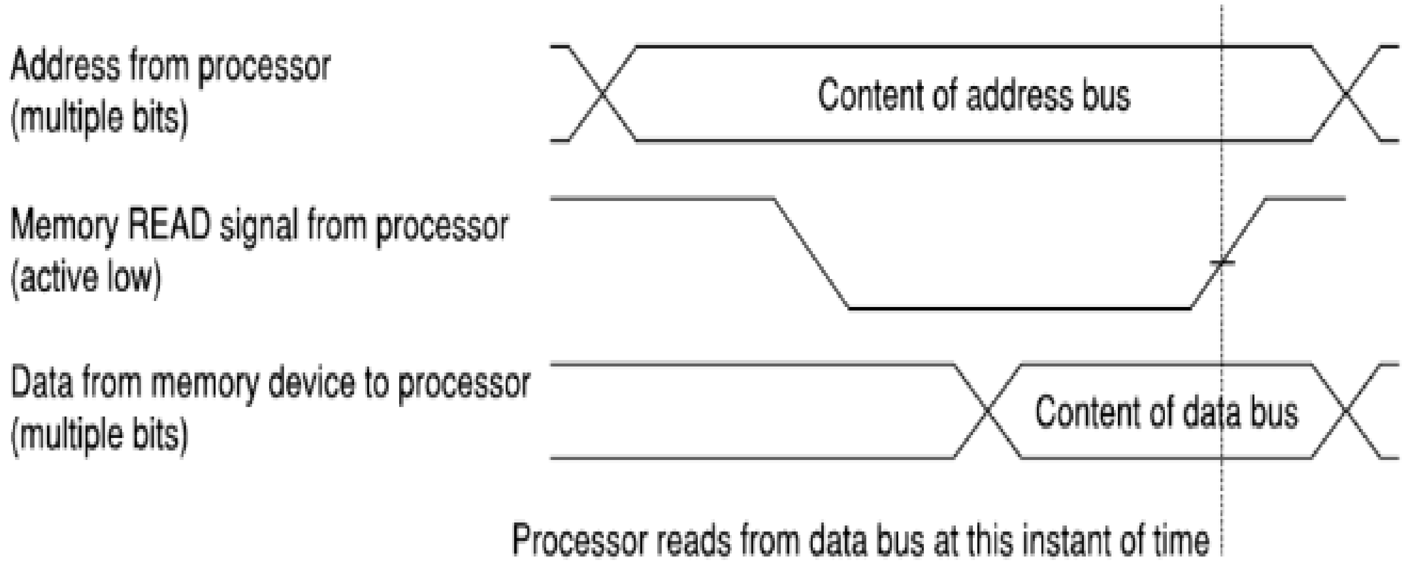


Flowchart for simplified instruction cycle

- It may be observed from the flowchart that after fetching the instruction in the form of its opcode and decoding it, the processor checks for any eventful operand fetch, which might be necessary for some instructions.
- If found necessary, then the operand is fetched from memory and then the instruction is executed.
- Finally, the result of the instruction is stored and the whole cycle is repeated.

Instruction Fetch

- The first step is to fetch the instruction byte(s) from external memory.
- This external memory is a vast are containing many bytes of instructions.
- So, the processor must pin-point the correct location of this large memory are to extract the target byte
- It was already indicated that every memory location(byte in majority of cases) has a unique binary address.
- After receiving this address, the duty of the memory device is to decode the address to locate the target byte and place it on the data bus, so that the content of that address is available for the processor.


Address from processor
(multiple bits)
Content of address bus
Memory READ signal from processor
(active low)
Data from memory device to processor
(multiple bits)
Content of data bus
Processor reads from data bus at this instant of time

Timing diagram for instruction fetch

- For the purpose of instruction fetch, the processor places an address Composed of multiple bits of binary information, on the address bus.
- Simultaneously, the processor also sends a memory read signal through its control bus.
- When these signals reach the memory device, the data are sent to the processor automatically by the memory device.
- Schematically this transaction is depicted and is known as timing diagram.
- Address signals, generated by the processor, are stable at this stage to ensure a valid data transaction.

Chip Select

- Out of so many devices interfaced with address, data and control bus the correct is identified with the help of chip-select input.(CE or CS)
- If this input is not activated, the device does not react with the system bus communications.
- Using a part of the address lines and a suitable decoder, the processor activates only one device during any communication and that solves the problem.
- This technique is known as address decoding and device selection.
- A processor is assisted with a memory decode and an I/O decoder to target the correct device, whish is of current interest.

Active Low/Active High

- Generally for active low signals, the transactions are carried out when it switches from low to high.
- In the case of active high signals, the process occurs at the time when the signal switches high to low.

Instruction Decode

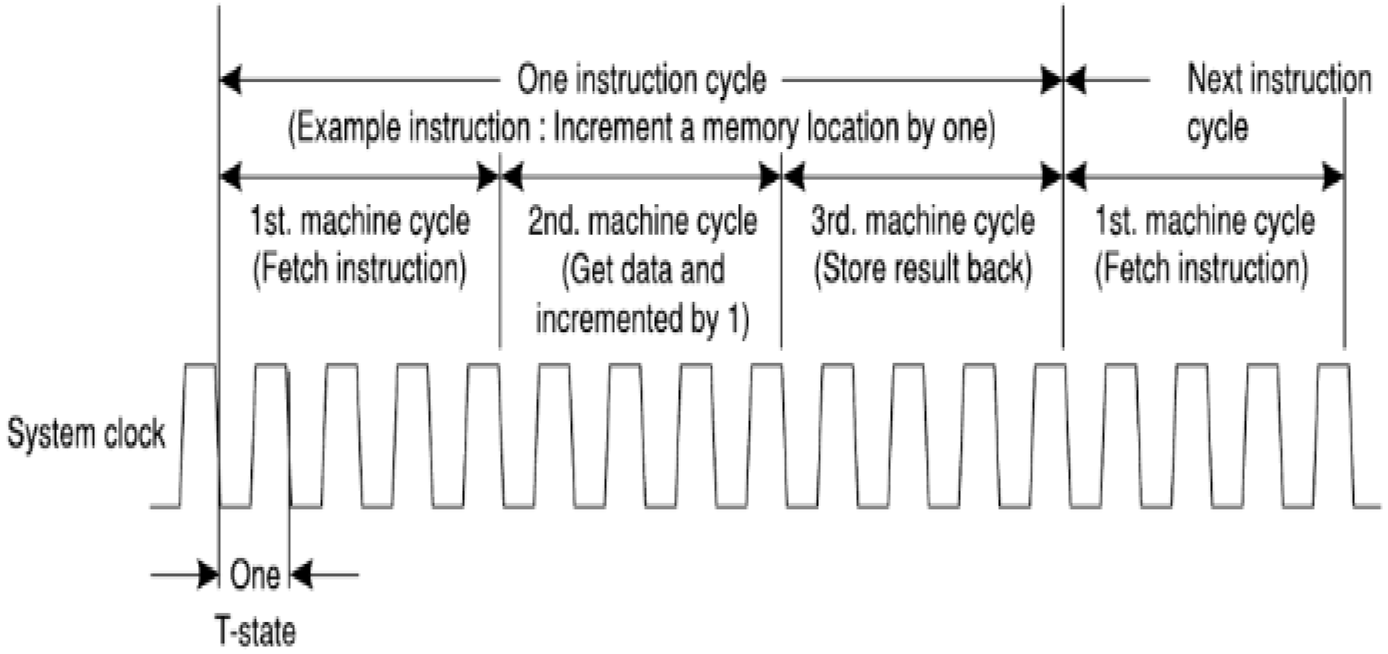
- After receiving the instruction code byte within itself, the processor becomes busy in understanding it.(What to do?)
- This part is known as instruction decode, carried out within the processor itself.
- After the completion of instruction decoding, the processor knows whether to fetch operands from external memory or to increment a register by one or to store a register content in external memory location.
- This instruction decoding may be implemented through hardware.
- Instruction decoding may be implemented through software known as micro-programming.
- This demands a miniature processor within the processor itself, completely devoted for instruction decoding and its execution.

Instruction Execute

- This is the last and final phase of an instruction’s execution. Depending upon the instruction, one or several operations are implemented by the processor.
- Once this part is complete, the processor looks forward for the next instruction fetch-decode-execute, and the process continues.

Machine Cycle and T-States

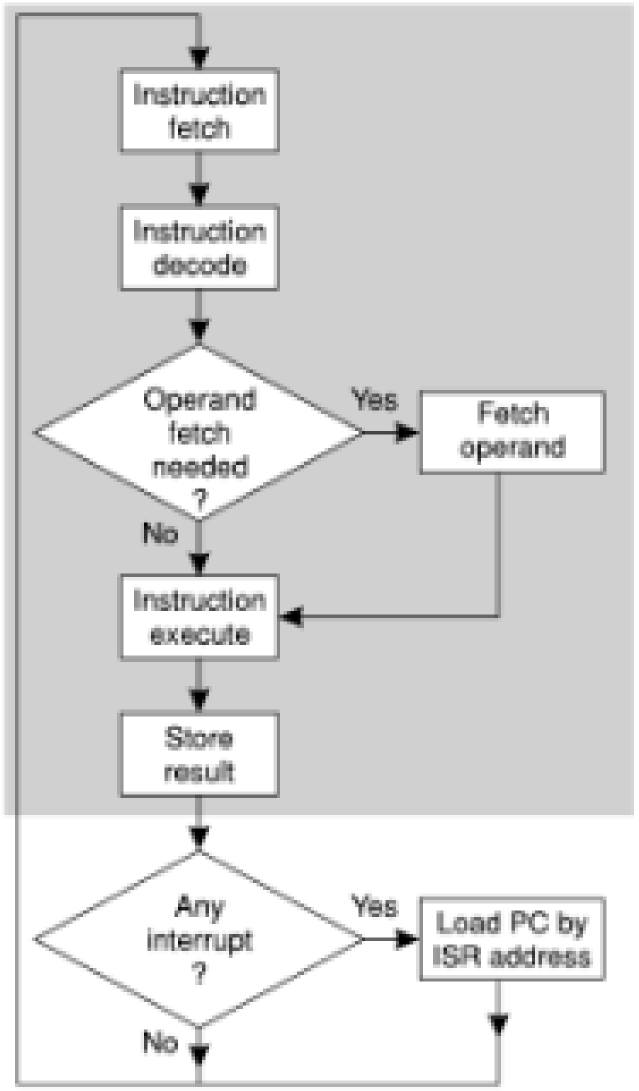
- An instruction cycle has one or more machine cycles and every machine cycle is composed of several T-states.
- A machine cycle is the step or time-slice during which 1-byte(or one word) of data are transacted between the processor and some external device.
- Generally this external device is the memory device or in exceptional cases it might be an I/O device also.
- To transact 1-byte of information, one machine cycle must be executed by the processor.


One instruction cycle
(Example instruction : Increment a memory location by one)
Next instruction cycle
1st. machine cycle (Fetch instruction)
2nd. machine cycle (Get data and incremented by 1)
3rd. machine cycle (Store result back)
1st. machine cycle (Fetch instruction)
System clock
One T-state

Example of instruction cycle, machine cycle and T-state correlation

Timings , Control and Response

- Timing and control play very important roles in smooth and efficient functioning of any processor.
- Interrupt is an external asynchronous signal, which forces the processor to carry out something special for it by branching to a pre-defined address and thus executing a special program segment know as interrupt service routine(ISR).
- As this is an asynchronous signal, it may be activated at any time during the execution of any instruction by the processor.
- However the processor cannot leave an instruction’s execution half-way to start doing something else for the sake of such an interrupting signal.
- To solve this problem, processors reserve a particular time-slot for checking the existence of any interrupt input signal during the execution of each and every instruction.
- E.g.: in 8085 the penultimate T state of last machine cycle of any instruction for this interrupt checking.



Modified flowchart for simplified instruction cycle

Register Set

- To perform internal operations, all processors offer some internal registers, which can store temporary information or some operands.
- Similar to read/write memory, these registers are nothing but a combination of several flip-flops. Most of these registers are user (programmer) accessible and a few are not.
- The number of user accessible register varies from processor to processor.
- Those processors that are memory oriented offers lesser number of internal registers as it expects data or operands would mainly be stored and manipulated within the read/write memory(RAM)of the system.
- E.g.: Motorola 6800

- Some Processors are register oriented which offers a larger number of internal registers for the user.
- Eg:Zilog Z80
- It may be noted that the program execution time for a processor would be less if the data are available within itself rather than looking outside for them.
- More Internal registers means more complexity in instruction decoding as each register would demand a separate instruction to be provided by the instruction set of the processor
- **Temporary registers are purely for processor's own use.**

Status Register

- Every Processor performs some arithmetic or logical operations generating some results.
- Depending upon whether the result is zero for negative or produced a carry or odd/even parity, some additional actions might have to be taken by the programmer.
- Status registers solves this problem by offering the result status of the last performed arithmetic or logical operation through its pre-assigned bits.
- Generally , each bit of this status register is assigned for one particular indication
- E.g.: carry, parity, zero, overflow etc.
- These bits act as flags and their conditions(true or false)help the program to decide further course of actions and dictate the conditional program branching

Accumulator

- Accumulator also known as result register.
- In earlier processors, result of all arithmetic or logical operations were made available only in the accumulator .
- In more recent register-to-register architecture, all relevant registers available within the processor may contain the result of similar operations.

Program Counter

- This is one of the most important registers within any processor as it is responsible for holding the address of the memory location for next instruction byte/word to be fetched by the processor.
- After fetching every instruction byte, this is automatically incremented by one to point to the next byte.
- The only exception for this auto-increment of the program counter is in the case of program branching, when it is reloaded by a new value.
- This counter is always initialized during system reset so that the first executable instruction byte is fetched from a pre-defined location of the memory

Stack Pointer

- System stack is a RAM area, which is earmarked by the programmer to accommodate important information,
- e.g, return address or register values, in last-in-first-out(LIFO) sequence.
- Stack pointer always points to the top of the stack area.

General Purpose Registers

- These registers are available within the processor for temporary data storage and manipulation.
- For arithmetic or logical operations, one of the two operands must be within these registers(the other one should be in the accumulator). As already indicated, number of these registers vary, depending on the processor

References

- Subrata Ghoshal, Computer Architecture and Organization: From 8085 to Core2Duo and beyond, Pearson, 2011