Computer Organization
Unit 1
Dt.21.11.17

Topics covered:
Course outline and schedule
Introduction

General information

Course : Computer Organization

Instructor : Dr E V Prasad

Email : profevprasad@yahoo.com

Lecture time : Thursday 10:00pm – 12:00 pm.

Office hours : By appointment.

(I will hang around for a few minutes at the end of each class).

Online Resources:

Supporting staff: Mrs. R Seeta Sireesha , Asst. professor

Email : sitasireesha@gvpcew.ac.in

Office hours : Discuss with the RSS

Course Objective

- Describe the general organization and architecture of computers.
- Identify computers' major components and study their functions.
- Introduce hardware design issues of modern computer architectures.
- Learn assembly language programming.
- Build the required skills to read and research the current literature in computer architecture.

Textbooks

1. “Computer Organization,” by Carl Hamacher, Zvonko Vranesic and Safwat Zaky. Fifth Edition McGraw-Hill, 2002.

2. “SPARC Architecture, Assembly Language Programming and C,” Richard P. Paul, Prentice Hall, 2000.

Course topics

1. Introduction (Unit 1):Basic concepts, overall organization.
2. Machine Instruction and Programs(unit 2): fetch/execute cycle, basic addressing modes, instruction sequencing, assembly language and stacks. The role of Stacks and Queues in computer programming equation Logic Instructions shift and Rotate Instructions CISC vs. RISC architectures.
3. Logic Circuits Fundamentals(Unit 3) : Logic Gates, Combinational Circuits, Sequential circuits
4. Type of Instructions(Unit 4):Arithmetic and Logic InstructionsBranch Instructions Addressing Modes Input/output Operations

Cont..

5. CPU architecture (unit 5): Single-bus CPU, Multiple-bus CPU Hardwired control, and Microprogrammed control.
6. Processing Unit (unit 6): Register Transfers, Performing An Arithmetic Or Logic Operation Fetching A Word From Memory Execution of Complete Instruction Wide Branch Addressing Microinstructions with next –Address Field
7. Arithmetic (unit 7): Integer arithmetic and floating-point arithmetic.
8. Memory architecture (unit 8): Memory hierarchy, Primary memory, Cache memory, virtual memory.

Cont..

9. Cache Memories (unit 9): MappingFunctions, Interleaving

10. Input / Output organization (unit 10): I/O device addressing, I/O data transfers, Synchronization, DMA, Interrupts, Channels, Bus transfers, and Interfacing.

11. Bus (unit 11) : Synchronous Bus, Asynchronous Bus, Peripheral Component Interconnect (PCI) Bus, Universal Serial Bus (USB)

12. Pipelining(unit 12) : Arithmetic & Instruction Pipelining, Data Hazards, Vector Processing.

Grading System

Midterm Exam 1 : (15) Units 1,2 and 3 during 15^{th} -20^{th} jan 2018

Midterm Exam 2: (15) Units 4,5 and 6 during 19^{th}-24^{th} mar 2018

Weightage : higher for better performed Exam: 2/3 : 1/3

Assignments : (5) - 6 homework assignments.

Tutorial s: Practice -3 problem solving sessions

On-line Quiz 1 s Units 1,2 and 3 during 15^{th} -20^{th} jan 2018

On-line Quiz 2 s Units 4,5 and 6 during 19^{th}-24^{th} mar 2018

Weightage : higher for better performed Exam: 2/3 : 1/3

Final : (70) during 2^{nd}-14^{th} april 2018 - All topics.

Course topics, exams and assignment calendar

Week #1 - Introduction

Week #2 - Addressing methods

- Lab. Assignment 1.

Week #3 - Addressing methods.

Week #4 - Logic Circuit Fundamentals

- Lab. Assignment 2.

Week #5 - CPU Architecture.

- Lab. Assignment 3.

Week #6 - CPU Architecture.

- Lab. Assignment 4.

Week #7 - Arithmetic.

Course topics, exams and assignment calendar

Week #8		**Midterm Exam**
Week #9	-	Arithmetic
Week #10:	-	Memory architecture.
		- Lab. Assignment 5.
Week #11	-	Memory organization
		- Lab. Assignment 6.
Week #12	-	I/O organization.
Week #13	-	I/O devices.
		- Lab. Assignment 7.
Week #14	-	Pipelining
Week #15		**Final Exam**

Grading policy

- Grading of assignments/exams is handled by the Supporting staff (RSS), if you cannot resolve a problem with RSS, see me.
- Assignments may be submitted by ?hard copy and email. Hard copy will be accepted, but you have to submit in the department office to stamp the date. Please submit all the assignments to the RSS.
- Late assignments are penalized by a loss of 33% per day late (so 3 days late is as late as you can get). Solution will be posted on the course web page No assignments will be accepted after the solution is posted.
- The weeks during which exams will be held have been announced. The actual day of that week, (Thursday) when the exam will be held will be announced two weeks prior to the exam.

If you have any conflict with the exam date, please see me in advance.

Important prerequisite material

- Digital Logic Course is fundamental to the Computer Organization Course.
- Review issues in:
  - Basic computer organization: CPU, Memory, I/O, Registers.
  - Fundamentals of combinatorial design and sequential design.
  - Simple ALU, simple register design
  - Bus structure
  - system software
  - Performance
  - Historical perspective

Reading

- Reading the text is imperative.
- Computer architecture especially processor design, changes rapidly.

You really have to keep up with the changes in the industry.

This is especially important for job interviews later.

Feedback

Please provide informal feedback early and often, before the formal review process.

Basic Functional Unit of a Computer

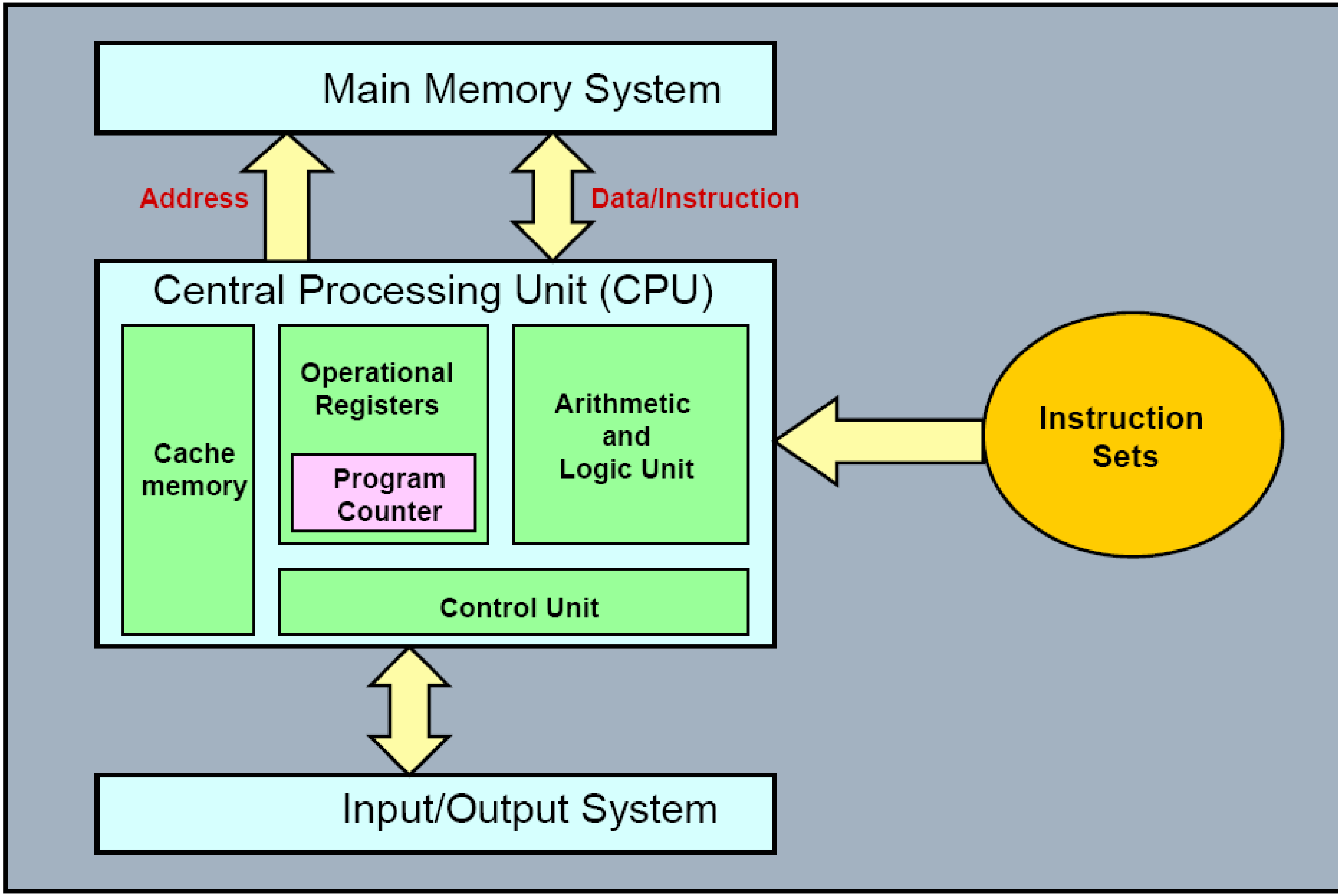


Figure 1.1. (a) Basic functional units of a computer

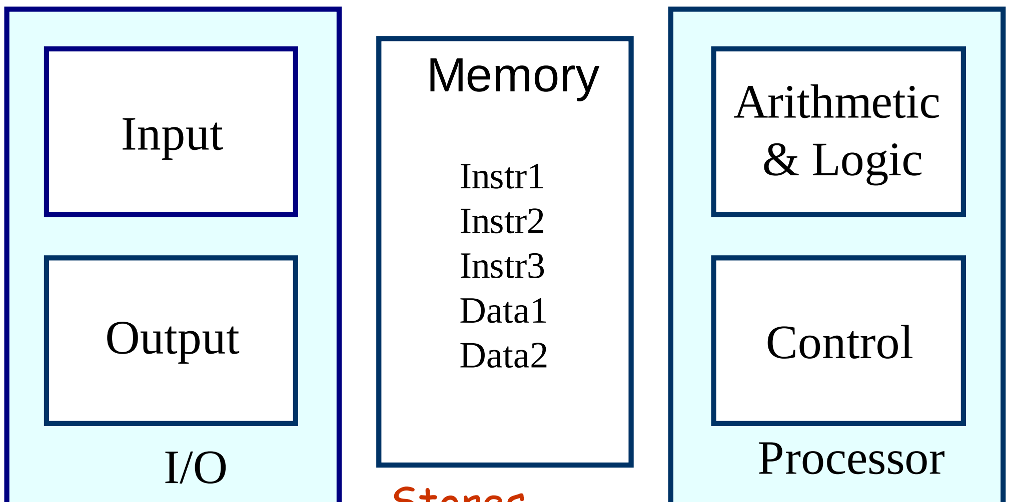
What is a computer?

- a computer is a sophisticated electronic calculating machine that:
  - Accepts input information,
  - Processes the information according to a list of internally stored instructions and
  - Produces the resulting output information.
- Functions performed by a computer are:
  - Accepting information to be processed as input.
  - Storing a list of instructions to process the information.
  - Processing the information according to the list of instructions.
  - Providing the results of the processing as output.
- What are the functional units of a computer?

Functional units of a computer

Input unit accepts information:
- Human operators,
- Electromechanical devices (keyboard)
- Other computers

Arithmetic and logic unit(ALU):
- Performs the desired operations on the input information as determined by instructions in the memory



Output unit sends results of processing:
- To a monitor display,
- To a printer

Stores information:
- Instructions,
- Data

Control unit coordinates various actions
- Input,
- Output
- Processing

Information in a computer -- *Instructions*

- Instructions specify commands to:
  - Transfer information within a computer (e.g., from memory to ALU)
  - Transfer of information between the computer and I/O devices (e.g., from keyboard to computer, or computer to printer)
  - Perform arithmetic and logic operations (e.g., Add two numbers, Perform a logical AND).
- A sequence of instructions to perform a task is called a program, which is stored in the memory.
- Processor fetches instructions that make up a program from the memory and performs the operations stated in those instructions.
- What do the instructions operate upon?

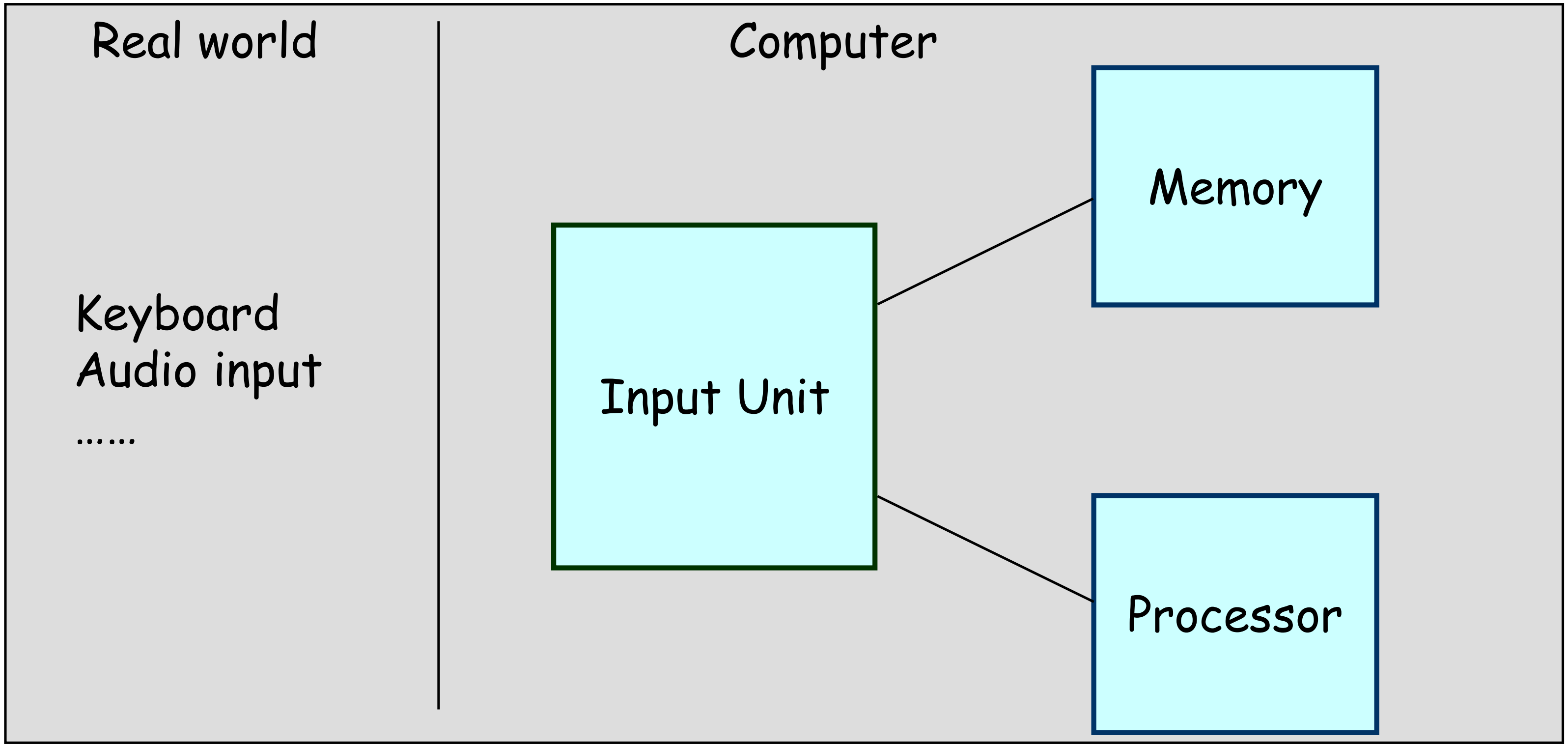
Information in a computer -- **Data**

- Data are the "operands" upon which instructions operate.
- Data could be:
  - Numbers,
  - Encoded characters.
- Data, in a broad sense means any digital information.
- Computers use data that is encoded as a string of binary digits called bits.

Input unit

Binary information must be presented to a computer in a specific format. This task is performed by the input unit:

- Interfaces with input devices.
- Accepts binary information from the input devices.
- Presents this binary information in a format expected by the computer.
- Transfers this information to the memory or processor.



Memory unit

- Memory unit stores instructions and data.
  - Recall, data is represented as a series of bits.
  - To store data, memory unit thus stores bits.
- Processor reads instructions and reads/writes data from/to the memory during the execution of a program.
  - In theory, instructions and data could be fetched one bit at a time.
  - In practice, a group of bits is fetched at a time.
  - Group of bits stored or retrieved at a time is termed as "word"
  - Number of bits in a word is termed as the "word length" of a computer.
- In order to read/write to and from memory, a processor should know where to look:
  - "Address" is associated with each word location.

Memory unit (contd..)

- Processor reads/writes to/from memory based on the memory address:
  - Access any word location in a short and fixed amount of time based on the address.
  - Random Access Memory (RAM) provides fixed access time independent of the location of the word.
  - Access time is known as "Memory Access Time".
- Memory and processor have to "communicate" with each other in order to read/write information.
  - In order to reduce "communication time", a small amount of RAM (known as Cache) is tightly coupled with the processor.
- Modern computers have three to four levels of RAM units with different speeds and sizes:
  - Fastest, smallest known as Cache
  - Slowest, largest known as Main memory.

Memory unit (contd..)

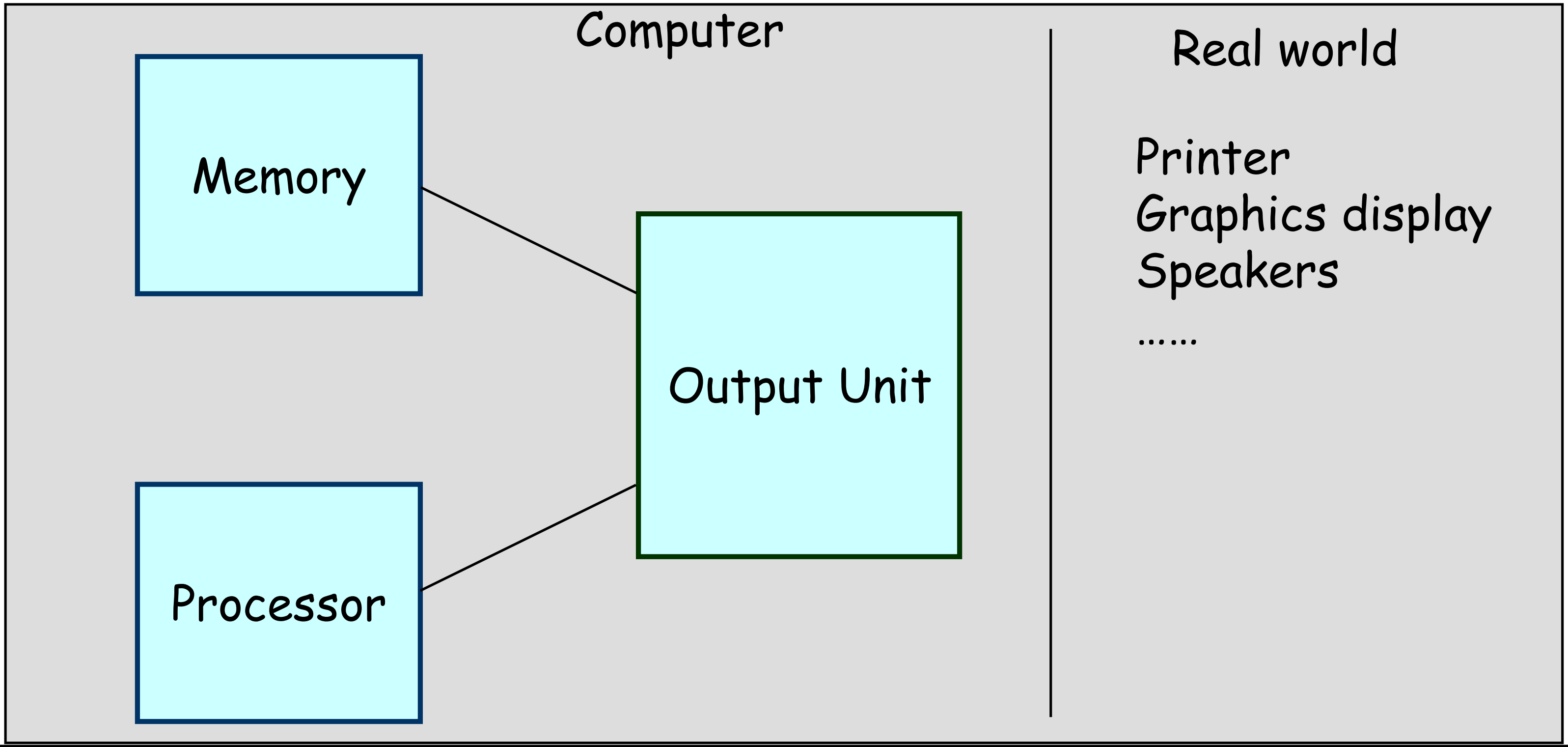
- Primary storage of the computer consists of RAM units.
  - Fastest, smallest unit is Cache.
  - Slowest, largest unit is Main Memory.
- Primary storage is insufficient to store large amounts of data and programs.
  - Primary storage can be added, but it is expensive.
- Store large amounts of data on secondary storage devices:
  - Magnetic disks and tapes,
  - Optical disks (CD-ROMS).
  - Access to the data stored in secondary storage in slower, but take advantage of the fact that some information may be accessed infrequently.
- Cost of a memory unit depends on its access time, lesser access time implies higher cost.

Arithmetic and logic unit (ALU)

- Operations are executed in the Arithmetic and Logic Unit (ALU).
  - Arithmetic operations such as addition, subtraction.
  - Logic operations such as comparison of numbers.
- In order to execute an instruction, operands need to be brought into the ALU from the memory.
  - Operands are stored in general purpose registers available in the ALU.
  - Access times of general purpose registers are faster than the cache.
- Results of the operations are stored back in the memory or retained in the processor for immediate use.

Output unit

- Computers represent information in a specific binary form. Output units:
  - Interface with output devices.
  - Accept processed results provided by the computer in specific binary form.
  - Convert the information in binary form to a form understood by an output device.



Control unit

- Operation of a computer can be summarized as:
  - Accepts information from the input units (Input unit).
  - Stores the information (Memory).
  - Processes the information (ALU).
  - Provides processed results through the output units (Output unit).
- Operations of Input unit, Memory, ALU and Output unit are coordinated by Control unit.
- Instructions control "what" operations take place (e.g. data transfer, processing).
- Control unit generates timing signals which determines "when" a particular operation takes place.

A Typical Instruction

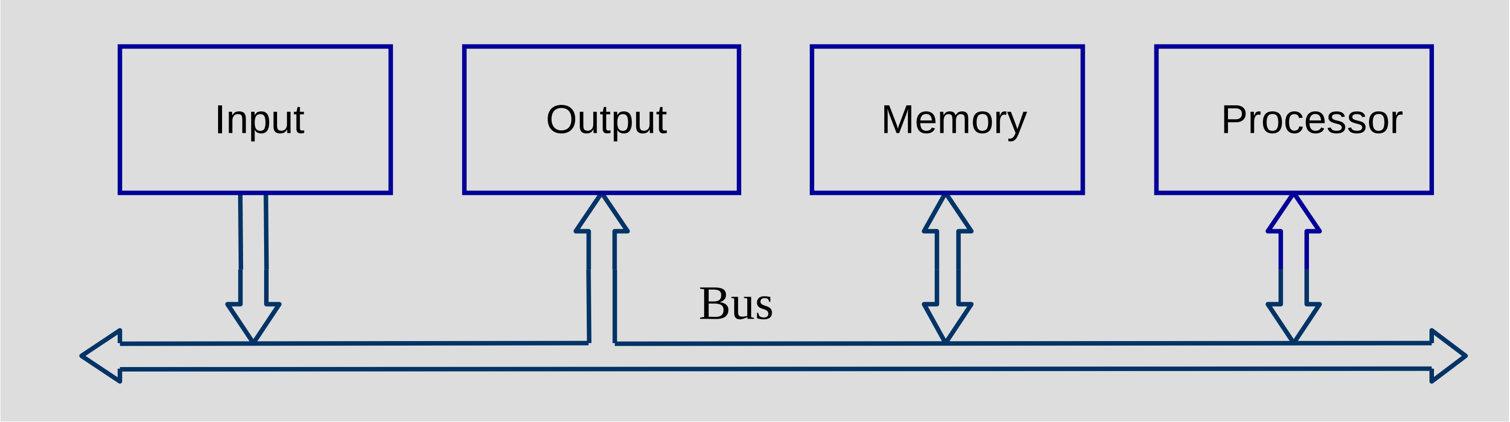
- Add LOCA, R0
  - Add the operand at memory location LOCA to the operand in a register R0 in the processor.
  - Place the sum into register R0.
  - The original contents of LOCA are preserved.
  - The original contents of R0 is overwritten.
  - Instruction is fetched from the memory into the processor – the operand at LOCA is fetched and added to the contents of R0 – the resulting sum is stored in register R0.

Separate Memory Access and ALU Operation

- Load LOCA, R1
- Add R1, R0
  - Whose contents will be overwritten?

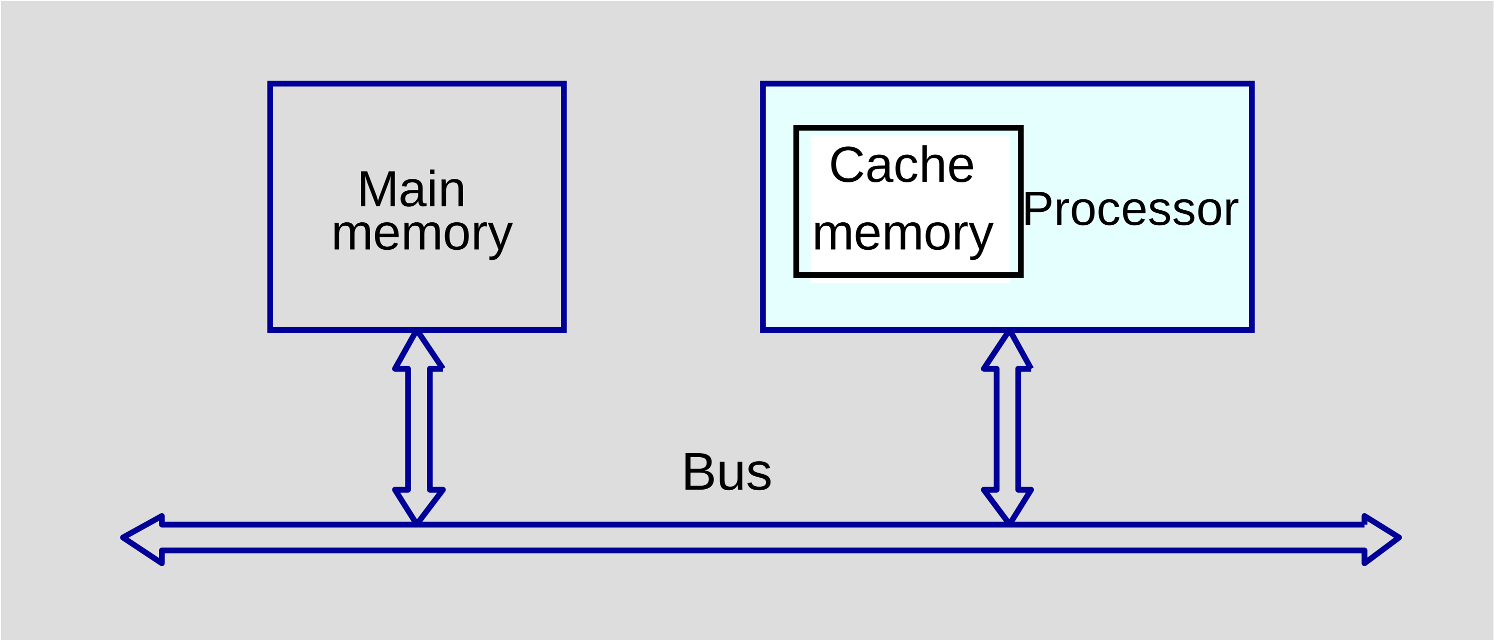
How are the functional units connected?

- For a computer to achieve its operation, the functional units need to communicate with each other.
- In order to communicate, they need to be connected.



- Functional units may be connected by a group of parallel wires.
- The group of parallel wires is called a ***bus***.
- Each wire in a bus can transfer one bit of information.
- The number of parallel wires in a bus is equal to the word length of a computer

Organization of cache and main memory



Why is the access time of the cache memory lesser than the access time of the main memory?

Registers

- In addition to the ALU and the control circuitry, the processor contains number of registers used for several different purposes.
- Instruction register (IR)
  - It holds the instruction that is currently being executed.
  - Its output is available to the control circuits which generates the timings ignals that control the various processing elements involved in executing the instruction.
- Program counter (PC)
  - PC is another specialized register.
  - It keeps track of the execution of a program. It contains the memory
  - address of the next instructionto be fetched and executed.
  - During the execution of an instruction, the contents of the PC updated to correspond to the address of the next instruction to executed.
  - PC *points* to the next instruction that is to be fetched from the memory.

Registers

two registers facilitate communication with the memory

- ❖ Memory address register (MAR)

  holds the address of the memory location to be accessed.

- ❖ Memory data register (MDR)

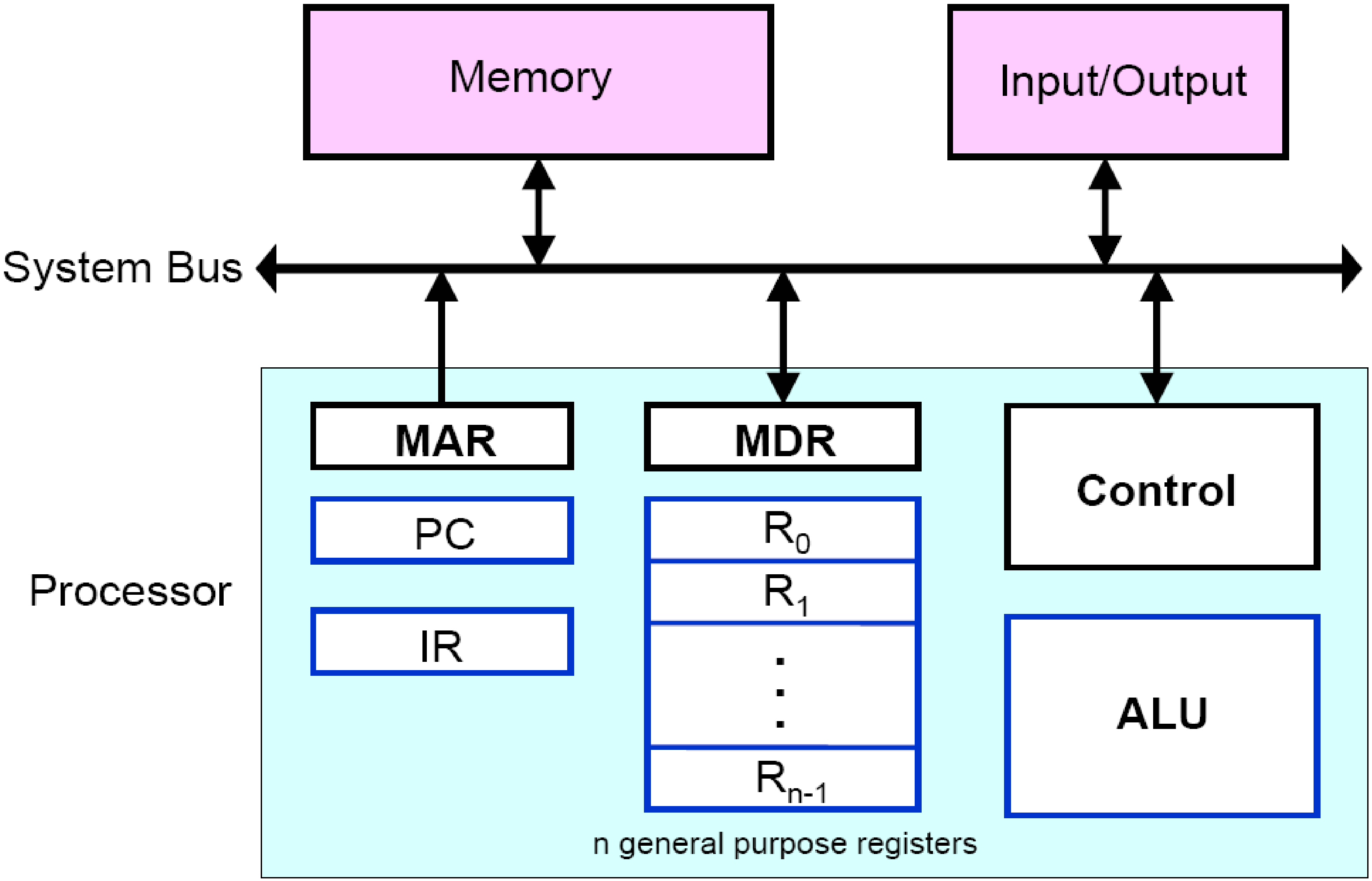
  MDR contains the data to be written into or read out of the addressed location.

- ❖ General-purpose register (R_0 – R_{n-1})

Registers

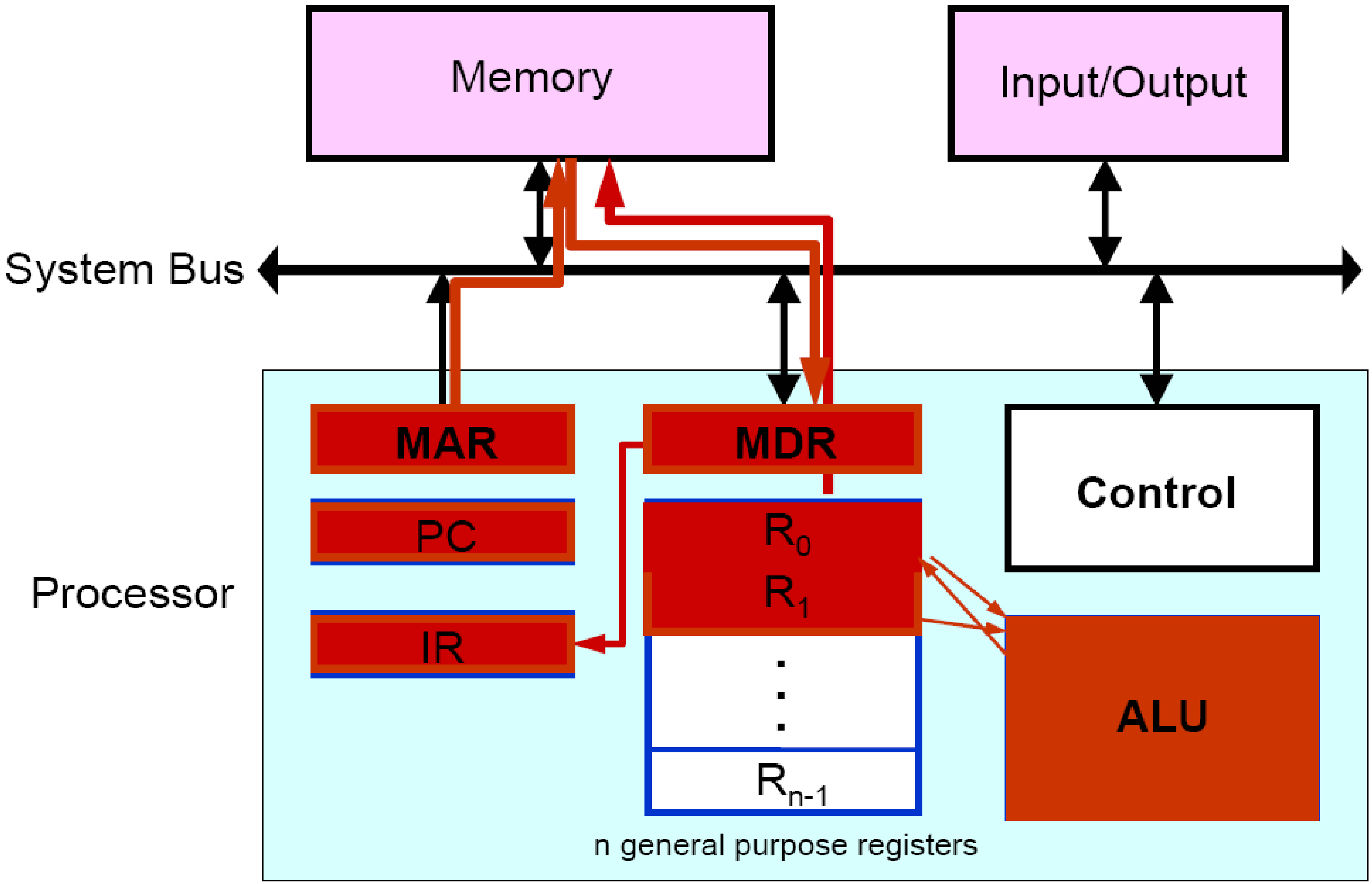
- In addition to the ALU and the control circuitry, the processor contains a
  a number of registers used for several different purposes
- Instruction register (IR)
  - It holds the instruction that is currently being executed.
  - Its output is available to the control circuits which generates the timings ignals that control the various processing elements involved in executing the instruction.
- Program counter (PC)
- General-purpose register (R_0 – R_{n-1})
- Memory address register (MAR)
- Memory data register (MDR)

Computer Components: Top-Level View



Connections between Processor and memory

Basic Operational Concepts

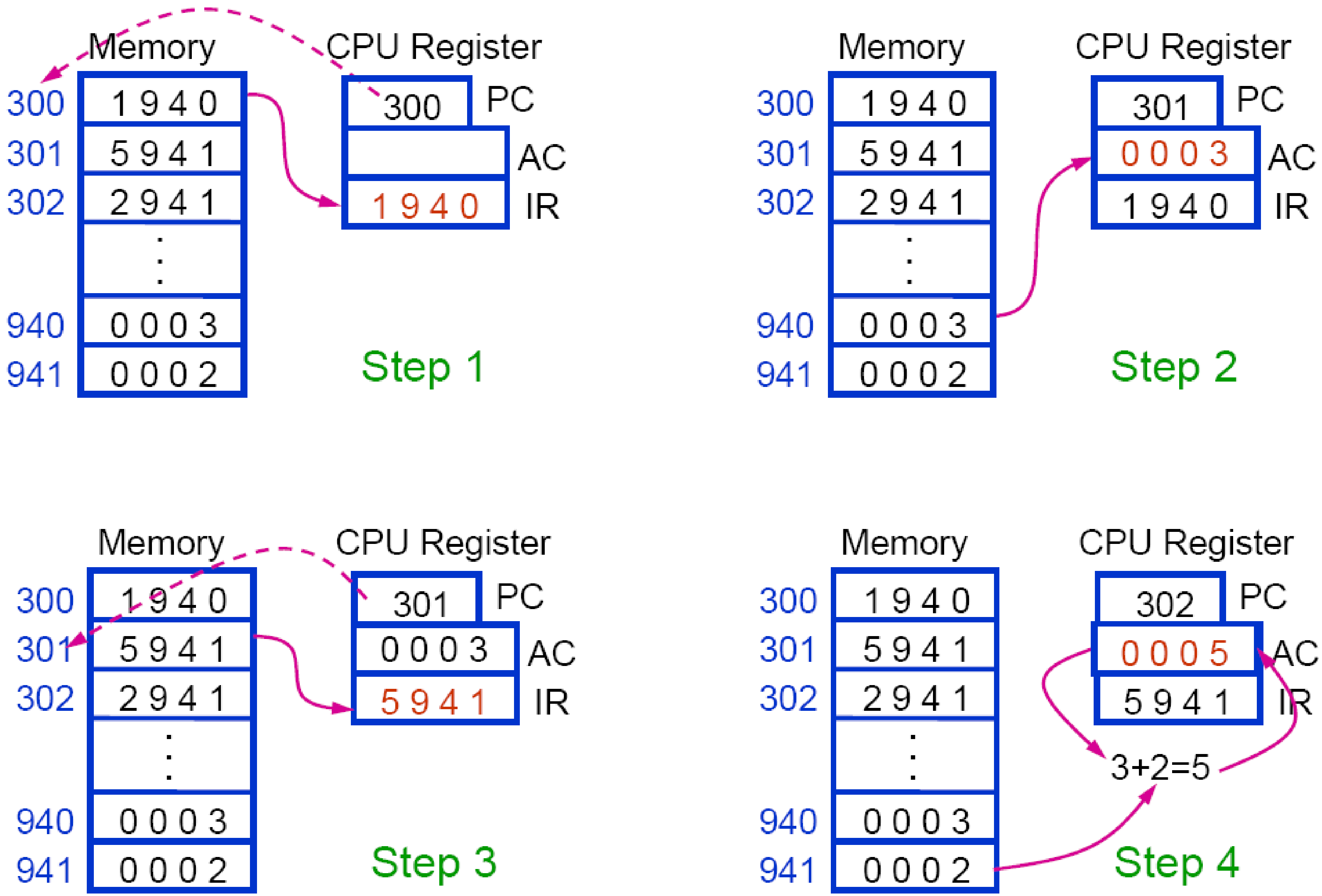


Typical Operating Steps

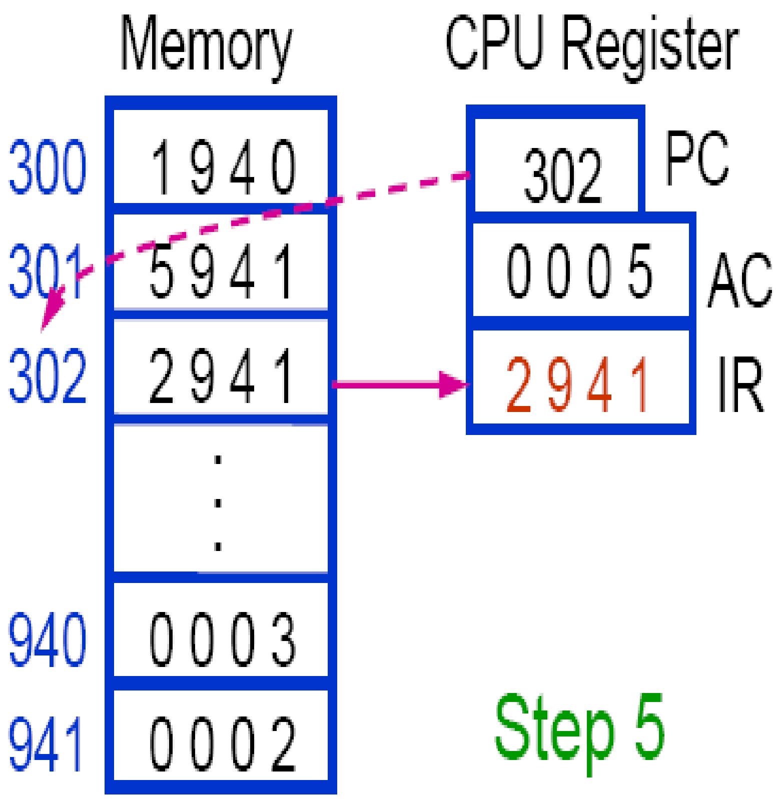
- Programs reside in the memory through input devices
- PC is set to point to the first instruction
- The contents of PC are transferred to MAR
- A Read signal is sent to the memory
- The first instruction is read out and loaded into MDR
- The contents of MDR are transferred to IR
- Decode and execute the instruction
- Get operands for ALU
  - General-purpose register
  - Memory (address to MAR – Read – MDR to ALU)
- Perform operation in ALU
- Store the result back
  - To general-purpose register
  - To memory (address to MAR, result to MDR – Write)
- During the execution, PC is incremented to the next instruction

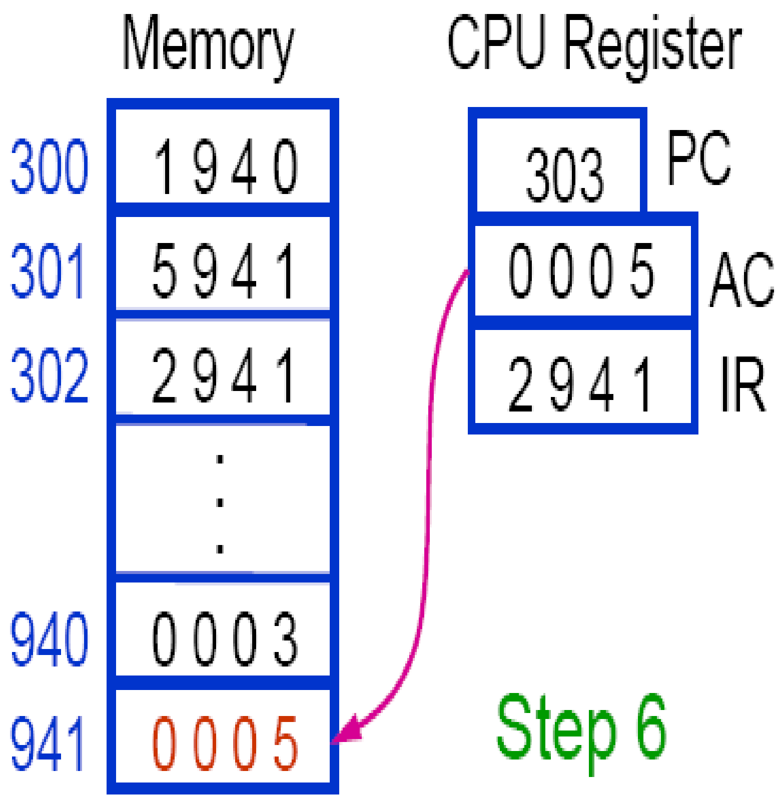
A Partial Program Execution

Example: Add 2 numbers and store the results



A Partial Program Execution Example





Interrupt

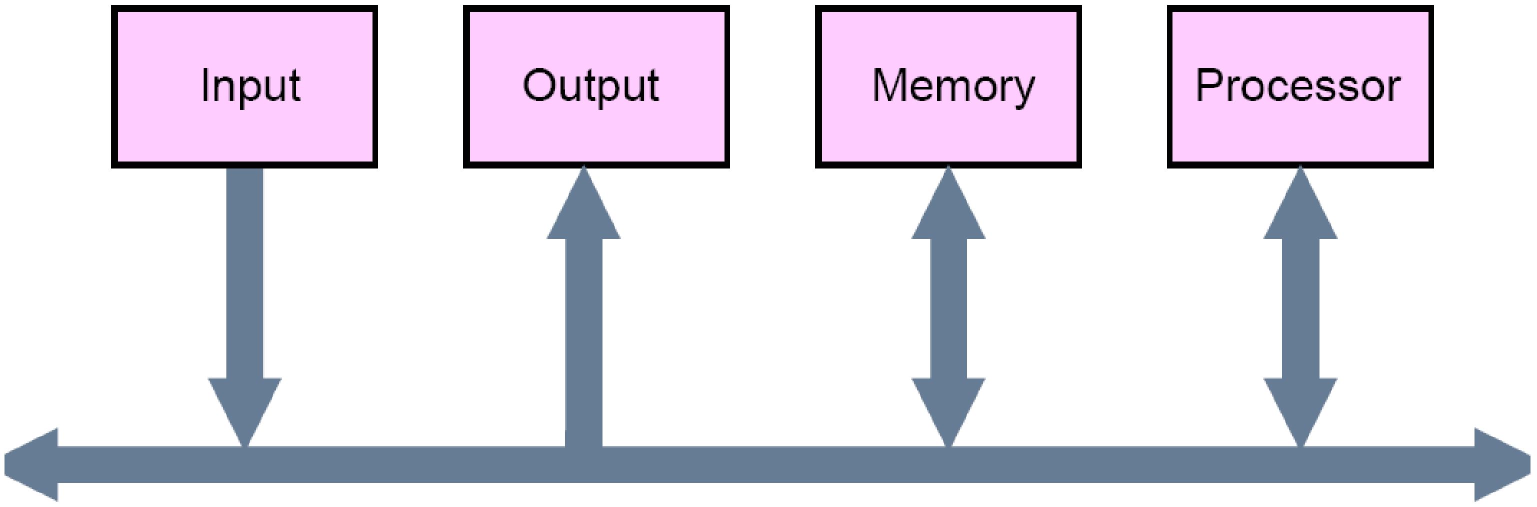
- Normal execution of programs may be interrupted if some device requires urgent servicing
  - To deal with the situation immediately, the normal execution of the current program must be interrupted
- Procedure of interrupt operation
  - The device raises an interrupt signal
  - The processor provides the requested service by executing an appropriate interrupt-service routine
  - The state of the processor is first saved before servicing the interrupt
    - Normally, the contents of the PC, the general registers, and some control information are stored in memory
  - When the interrupt-service routine is completed, the state of the processor is restored so that the interrupted program may continue

Classes of Interrupts

- Program
  - Generated by some condition that occurs as a result of an instruction execution such as arithmetic overflow, division by zero, attempt to execute an illegal machine instruction, or reference outside a user's allowed memory space
- Timer
  - Generated by a timer within the processor. This allows the operating system to perform certain functions on a regular basis
- I/O
  - Generated by an I/O controller, to signal normal completion of an operation or to signal a variety of error conditions
- Hardware failure
  - Generated by a failure such as power failure or memory parity error

Bus Structures

- A group of lines that serves a connecting path for several devices is called a bus
  - In addition to the lines that carry the data, the bus must have lines for address and control purposes
  - The simplest way to interconnect functional units is to use a single bus, as shown below



Drawbacks of the Single Bus Structure

- The devices connected to a bus vary widely in their speed of operation
  - Some devices are relatively slow, such as printer and keyboard
  - Some devices are considerably fast, such as optical disks
  - Memory and processor units operate are the fastest parts of a computer
- Efficient transfer mechanism thus is needed to cope with this problem
  - A common approach is to include buffer registers with the devices to hold the information during transfers
  - An another approach is to use two-bus structure and an additional transfer mechanism
    - A high-performance bus, a low-performance, and a bridge for transferring the data between the two buses. ARMA Bus belongs to this structure

Software

- In order for a user to enter and run an application program, the computer must already contain some system software in its memory

- System software is a collection of programs that are executed as needed to perform functions such as
  - Receiving and interpreting user commands
  - Running standard application programs such as word processors, etc, or games
  - Managing the storage and retrieval of files in secondary storage devices
  - Controlling I/O units to receive input information and produce output results

functions of system **Software**

- Translating programs from source form prepared by the user into object form consisting of machine instructions
- Linking and running user-written application programs with existing standard library routines, such as numerical computation packages
- System software is thus responsible for the coordination of all activities in a computing system

Operating System

- Operating system (OS)
  - This is a large program, or actually a collection of routines, that is used to control the sharing of and interaction among various computer units as they perform application programs
- The OS routines perform the tasks required to assign computer resource to individual application programs
  - These tasks include assigning memory and magnetic disk space to program and data files, moving data between memory and disk units, and handling I/O operations
- In the following, a system with one processor, one disk, and one printer is given to explain the basics of OS
  - Assume that part of the program's task involves reading a data file from the disk into the memory, performing some computation on the data, and printing the results

HOW OS manages to execute one application program ?

Consider a system with one processor, one disk, and one printer

- Sequence of the steps involved in running one application program.

- Assume that the application program has been compiled from a high-level language form into a machine language form and stored on the disk.
- The first step is to transfer this file into the memory.
- When the transfer is completed, execution of the program is started.
- Assume that part of the program's task involves reading a data file from the disk into the memory, performing some computation on the data, and printing the results.
- When execution of the program reaches the point where the data file is needed, the program requests the OS to transfer the data file from the disk to the memory.
- The OS performs this task and passes execution control back to the application program, which then proceeds to perform the required computation.

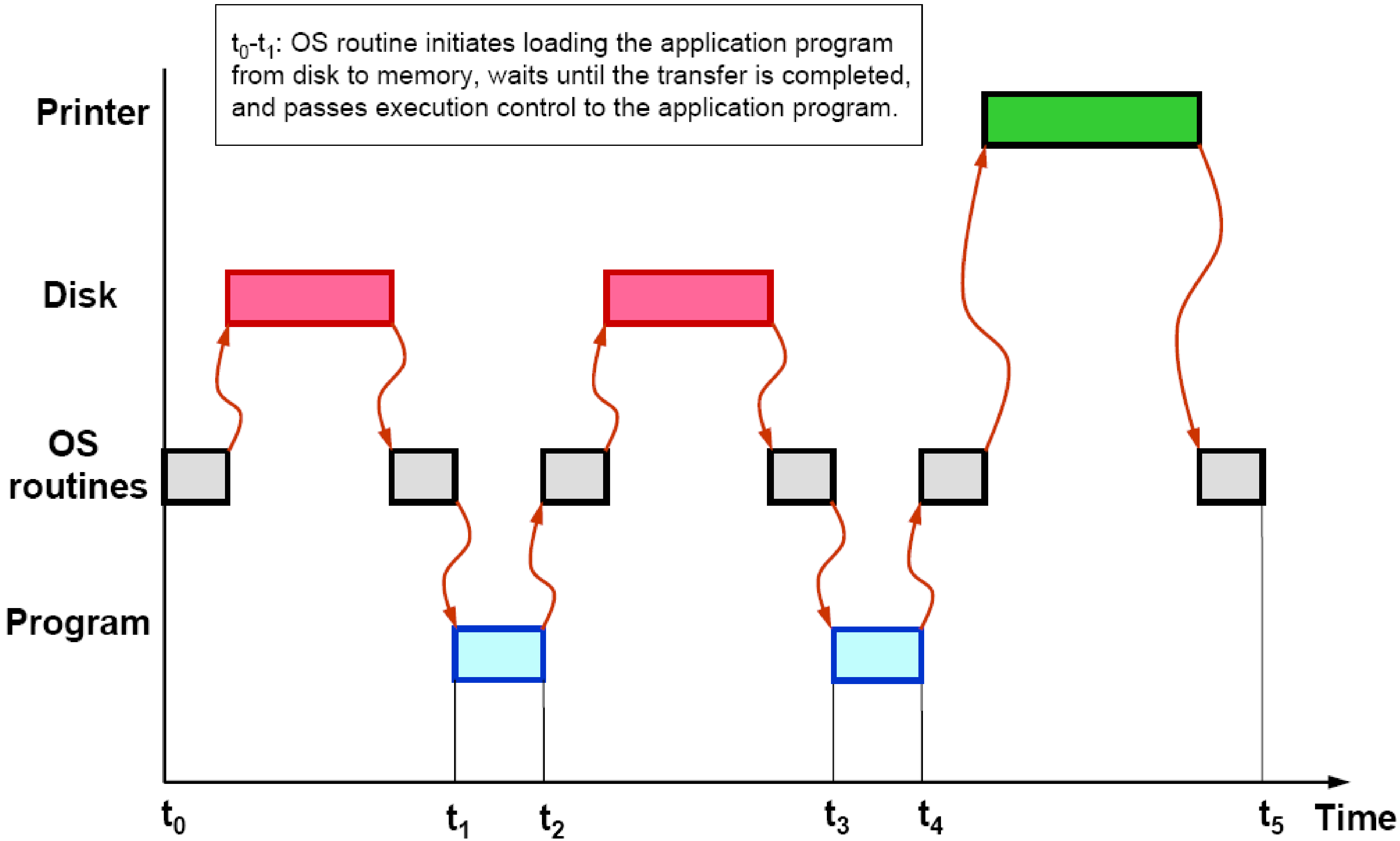
Cont..

- When the computation is completed and the results are ready to be printed, the application program again sends a request to the OS.
- An OS routine is then executed to cause the printer to print the results.
- i.e. execution control passes back and forth between the application program and the OS routines .
- A convenient way to illustrate this sharing of the processor execution time is by a time-line diagram, such as that shown in Figure 1.4.

Execution of more than one application program at a time

- Computer resources can be used more efficiently if several application programs are to be processed.
- Notice that the disk and the processor are idle during most of the time period {t_4 to t_5) .
- The OS can load the next program to be executed into the memory from the disk while the printer is operating.
- Similarly, during t_o to t_1, the OS can arrange to print the previous program's results while the current program is being loaded from the disk.
- Thus, the OS manages the concurrent execution of several application programs to make the best possible use of computer resources.
- This pattern of concurrent execution is called *multi-programming* or *multitasking*

User Program and OS Routine Sharing

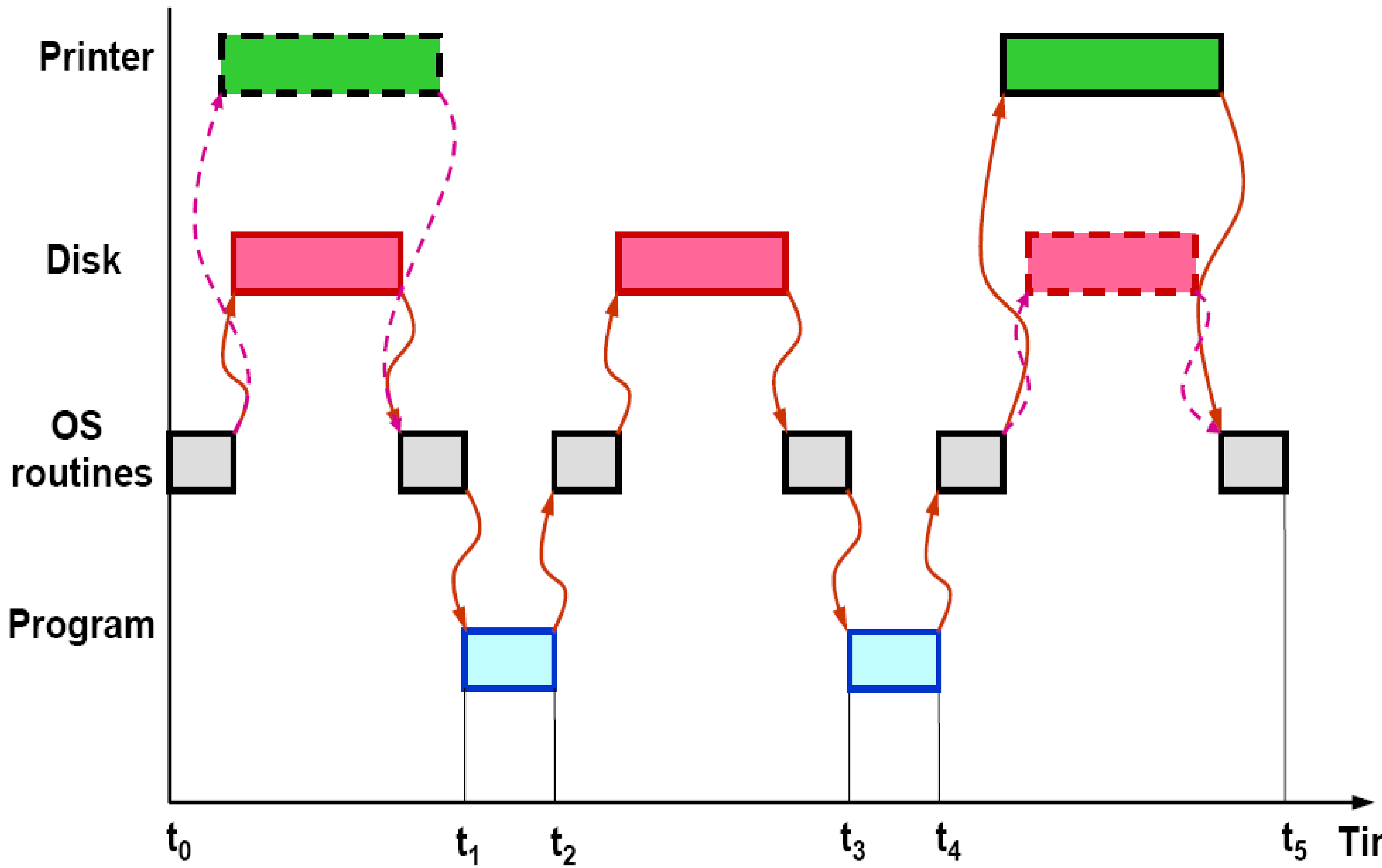


time-line diagram

Cont..

- During the time period t_o to t_1|, an OS routine initiates loading the application program from disk to memory, waits until the transfer is completed, and then passes execution control to the application program.
- A similar pattern of activity occurs during period t_2 to t_3 and period t_4 to t_5 when the OS transfers the data file from the disk and prints the results.
- At t_5 , the OS may load and execute another application program.

Multiprogramming or Multitasking



Performance

- The speed with which a computer executes programs is affected by the design of its hardware and its machine language instructions

- Because programs are usually written in a high-level language, performance is also affected by the compiler that translates programs into machine languages

- For best performance, the following factors must be considered
  - Compiler
  - Instruction set
  - Hardware design

Performance

- Processor circuits are controlled by a timing signal called a clock
  - The clock defines regular time intervals, called clock cycles
- To execute a machine instruction, the processor divides the action to be performed into a sequence of basic steps, such that each step can be completed in one clock cycle
- Let the length P of one clock cycle, its inverse is the clock rate, $R=1/P$
- Basic performance equation
  - $T=(N \times S)/R$, where T is the processor time required to execute a program, N is the number of instruction executions, and S is the average number of basic steps needed to execute one machine instruction
  - Note: these are not independent to each other
  - How to improve T?

Pipeline and Superscalar Operation

- Instructions are not necessarily executed one after another.
- The value of S doesn't have to be the number of clock cycles to execute one instruction.
- Use of Pipelining – overlapping the execution of successive instructions.
- Use multiple functional units
- Goal is to reduce S (could become <1!)

Performance Improvement

- Pipelining and superscalar operation
  - Pipelining: by overlapping the execution of successive instructions
  - Superscalar: different instructions are concurrently executed with multiple instruction pipelines. This means that multiple functional units are needed.
- The processor and a relatively small cache memory can be fabricated on a single integrated circuit chip

Performance Improvement

- Clock rate improvement
  - Improving the integrated-circuit technology makes logic circuits faster (R inereases), which reduces the time needed to complete a basic step.
  - Reducing amount of processing done in one basic step also makes it possible to reduce the clock period, P.
  - However, if the actions that have to be performed by an instruction remain the same, the number of basic steps needed may increase
    - Reduced instruction set computers (RISC) and complex instruction set computers (CISC)

CISC and RISC

- Reduce the number of basic steps to execute
  - Tradeoff between N and S
  - A key consideration is the use of pipelining
- S is close to 1 even though the number of basic steps per instruction may be considerably larger
- It is much easier to implement efficient pipelining in processor with simple instruction sets

Compiler

- A compiler translates a high-level language program into a sequence of machine instructions.
- To reduce N, we need a suitable machine instruction set and a compiler that makes good use of it.
- Goal – reduce $N \times S$
- A compiler may not be designed for a specific processor; however, a high-quality compiler is usually designed for, and with, a specific processor.

Performance Measurement

- T is difficult to compute.
- Measure computer performance using benchmark programs.
- System Performance Evaluation Corporation (SPEC) selects and publishes representative application programs for different application domains, together with test results for many commercially available computers.
- SPEC rating is a measure of the combined effect of all factors affecting performance.(Compiler,OS,CPU and of the computer being tested)Memory how much fast the computer under test
- Compile and run (no simulation)
- Reference computer

$$SPEC\ rating = \frac{\text{Running time on the reference computer}}{\text{Running time on the computer under test}}$$

$$SPEC\ rating = \left(\prod_{i=1}^{n} SPEC_i\right)^{\frac{1}{n}}$$

Where n is the number of programs in the suite.

Multiprocessors and Multicomputers

Multiprocessor computer

- Execute a number of different application tasks in parallel
- Execute subtasks of a single large task in parallel
- All processors have access to all of the memory – shared-memory multiprocessor
- Cost – processors, memory units, complex interconnection networks

Multicomputers

- Each computer only have access to its own memory
- Exchange message via a communication network – message-passing multicomputers

Representation of Basic Information

- The basic functional units of computer are made of electronics circuit and it works with electrical signal.
- There are two basic types of electrical signals, namely, analog and digital.
- In present days most of the computers are digital in nature and we will deal with Digital Computer in this course.
- Computer is a digital device, which works on two levels of signal . We say these two levels of signal as High and Low.
- To make it convenient for understanding, we use some logical value, say,

LOW (L) - will represent 0V and
HIGH (H) - will represent 5V

Basic Working Principle of a Computer

- Working of a computer with the help of a small hypothetical computer considering only CPU and memory.
- Assume that somehow we have stored the program and data into
- main memory.
- CPU can perform the job depending on the program stored in main memory.
- Consider an ALU which can perform four arithmetic operations and four logical operations
- To distinguish between arithmetic and logical operation, we may use a signal line.

  0-in that signal-represents an arithmetic operation

  1- in that signal-and represents a logical operation.
- In the similar manner, we need another two signal lines to distinguish

  between four arithmetic operations.

The different operations and their binary code is as follows:

Arithmatic	Logical
000 ADD	100OR
001 SUB	101AND
010 MULT	110NAND
011 DIV	111ADD

Consider the part of control unit, its task is to generate the appropriate signal at right moment.

- There is an instruction decoder in CPU which decodes this information in such a way that computer can perform the desired task.
- The simple model for the decoder may be considered that there is three input lines to the decoder and correspondingly it generates eight output lines.
- Depending on input combination only one of the output signals will be generated and it is used to indicate the corresponding operation of ALU.
- In our simple model, we use three storage units in CPU, Two -- for storing the operand and one -- for storing the results.
- These storage units are known as register.

- But in computer, we need more storage space for proper functioning of the Computer. Some of them are inside CPU, which are known as register.
- Other bigger junk of storage space is known as primary memory or main memory.
- The CPU can work with the information available in main memory only.
- To access the data from memory, we need two special registers one is known as Memory Data Register (MDR) and the second one is Memory Address Register (MAR).
- Data and program is stored in main memory. While executing a program, CPU brings instruction and data from main memory, performs the tasks as per the instruction fetch from the memory.
- After completion of operation, CPU stores the result back into the memory.

Main Memory Organization

- Main memory unit is the storage unit, There are several location for storing information in the main memory module.
- The capacity of a memory module is specified by the number of memory locations and the information stored in each location.
- A memory module of capacity 16 X 4 indicates that, there are 16 location in the memory module and in each location, we can store 4 bit of information.
- We have to know how to indicate or point to a specific memory location. his is done by address of the memory location.
- We need two operation to work with memory.
- READ Operation: This operation is to retrive the data from memory and bring it to CPU register
- WRITE Operation: This operation is to store the data to a memory location from CPU register

- With the help of one signal line, we can differentiate these two operations. If the content of this signal line is

  0 - we say that we will do a READ operation; and if it is
  1- we do WRITE operation.

- To transfer the data from CPU to memory module and vice-versa, we need some connection. This is termed as **DATA BUS**.
- How to specify a particular memory location where we want to store our data or from where we want to retrieve the data.
- This can be done by the memory address. Each location can be specified with the help of a binary address.
- If we use 4 signal lines, we have 16 different combinations in these four lines.
- To distingush 16 location, we need four signal lines.
- These signal lines used to identify a memory location is termed as ADDRESS BUS.
- Size of address bus depends on the memory size. For a memory module of capacity of 2^n location, we need n address lines, that is, an address bus of size n.

Address Decoder

- As for example, consider a memory module of 16 location and each location can store 4 bit of information .The size of address bus is 4 bit and the size of the data bus is 4 bit .The size of address decoder is 4 X 16. There is a control signal named R/W.
- If R/W = 0, we perform a READ operation and

  if R/W = 1, we perform a WRITE operation .
- If the contents of address bus is 0101 and contents of data bus is 1100 and R/W = 1, then 1100 will be written in location 5.
- If the contents of address bus is 1011 and R/W=0, then the contents of location 1011 will be placed in data bus.

Memory Instruction

- Instructions for data transfer from main memory to CPU and vice versa.
- In our hypothetical machine, we use three signal lines to identify a particular instruction. If we want to include more instruction, we need additional signal lines.

Instruction	Code	Meaning
1000	LDAI	imm Load Reg A with data that is given in the program
1001	LDAA	addr Load Reg A with data from memory location addr
1010	LDBI	imm. Load register B with data
1011	LDBA	addr Load Reg B with data from memory location addr
1100	STC	Store the value of Reg C in memory location addr
1101	HALT	Stop the execution
1110	NOP	No operation
1111	NOP	No operation

- With this additional signal line, we can go upto 16 instructions.

  When the signal of this new line is 0, it will indicate the ALU operation.

  For signal value equal to 1, it will indicate 8 new instructions. So, we can design 8 new memory access instructions.

- We have added 6 new instructions.

  Still two codes are unused, which can be used for other purposes.

  We show it as NOP means No Operation.

Control Unit

- We have seen that for ALU operation, instruction decoder generated the signal for appropriate ALU operation.
- Apart from that we need many more signals for proper functioning of the computer. Therefore, we need a module, which is known as control unit, and it is a part of CPU.
- The control unit is responsible to generate the appropriate signal.
- As for example, for LDAI instruction, control unit must generate a signal which enables the register A to store in data into it.
- One major task is to design the control unit to generate the appropriate signal at appropriate time for the proper functioning of the computer.

- Example 1.

Consider a simple problem to add two numbers and store the result in memory, say we want to add 7 to 5.

To solve this problem in computer, we have to write a computer program.

The program is machine specific, and it is related to the instruction set of the machine

- For our hypothetical machine, the program is as follows.

Instruction	Binary	HEX	Memory Location
LDAI 5	1000 0101	8 5	(0, 1)
LDBI 7	1010 0111	A 7	(2, 3)
ADD	0000	0	(4)
STC 15	1100 1111	C F	(5, 6)
HALT	1101	D	(7)

- Example 2.
- Say that the first number is stored in memory location 13 and the second data is stored in memory location 14. Write a program to Add the contents of memory location 13 and 14 and store the result in memory location 15.

Instruction	Binary	HEX	Memory Location
LDAA 13	1000 0101	8 5	(0, 1)
LDBA 14	1010 0111	A 7	(2, 3)
ADD	0000	0	(4)
STC 15	1100 1111	C F	(5, 6)
HALT	1101	D	(7)