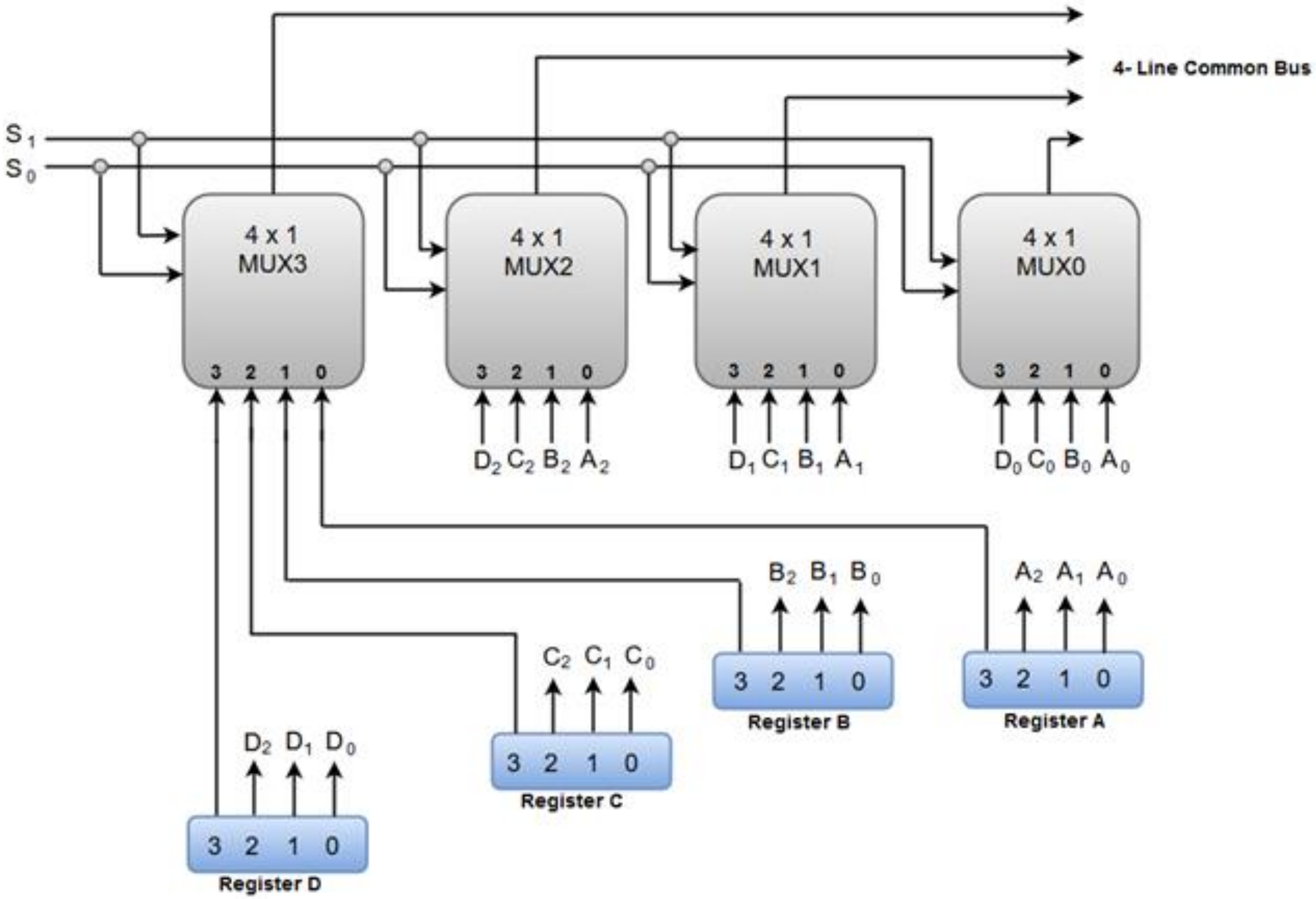
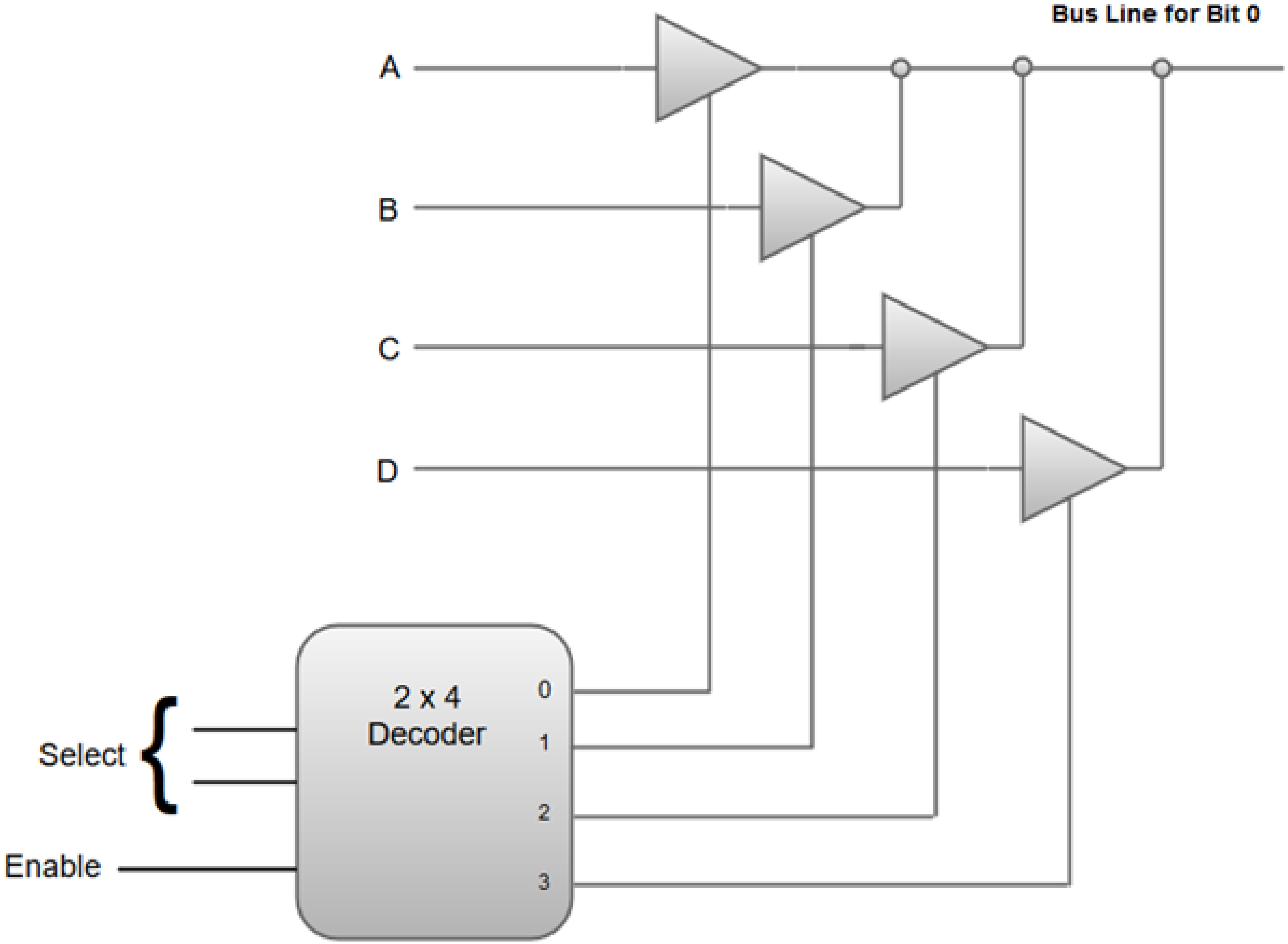
Bus System for 4 Registers

Bus System for 4 Registers:



Bus line with three state buffer:

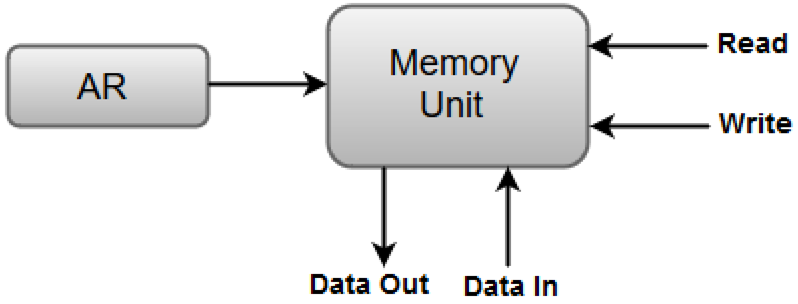


Memory Transfer

- Most of the standard notations used for specifying operations on memory transfer are stated below.

- The transfer of information from a memory unit to the user end is called a **Read** operation.
- The transfer of new information to be stored in the memory is called a **Write** operation.
- A memory word is designated by the letter **M**.
- We must specify the address of memory word while writing the memory transfer operations.
- The address register is designated by **AR** and the data register by **DR**.
- Thus, a read operation can be stated as:
- Read: **DR ← M [AR]**

- The Read statement causes a transfer of information into the data register (DR) from the memory word (M) selected by the address register (AR).
- And the corresponding write operation can be stated as:
- Write: M [AR] ← R1
- The Write statement causes a transfer of information from register R1 into the memory word (M) selected by address register (AR).



Register Transfer Language

Registers in Computer Architecture

- Register is a very fast computer memory, used to store data/instruction in-execution.
- A **Register** is a group of flip-flops with each flip-flop capable of storing **one bit** of information. An *n-bit* register has a group of *n flip-flops* and is capable of storing binary information of *n-bits*.
- A register consists of a group of flip-flops and gates. The flip-flops hold the binary information and gates control when and how new information is transferred into a register. Various types of registers are available commercially. The simplest register is one that consists of only flip-flops with no external gates.

Loading the Registers

- The transfer of new information into a register is referred to as loading the register. If all the bits of register are loaded simultaneously with a common clock pulse than the loading is said to be done in parallel.

Register Transfer Language

- The symbolic notation used to describe the micro-operation transfers amongst registers is called **Register transfer language**.
- The term **register transfer** means the availability of **hardware logic circuits** that can perform a stated micro-operation and transfer the result of the operation to the same or another register.

- The word **language** is borrowed from programmers who apply this term to programming languages. This programming language is a procedure for writing symbols to specify a given computational process.

Following are some commonly used registers:

- **Accumulator**: This is the most common register, used to store data taken out from the memory.
- **General Purpose Registers**: This is used to store data intermediate results during program execution. It can be accessed via assembly programming.
- **Special Purpose Registers**: Users do not access these registers. These registers are for Computer system,
    - **MAR:** Memory Address Register are those registers that holds the address for memory unit.
    - **MBR:** Memory Buffer Register stores instruction and data received from the memory and sent to the memory.
    - **PC:** Program Counter points to the next instruction to be executed.
    - **IR:** Instruction Register holds the instruction to be executed.

Register Transfer

- Information transferred from one register to another is designated in symbolic form by means of replacement operator.
- R2 ← R1
- It denotes the transfer of the data from register R1 into R2.

- Normally we want the transfer to occur only in predetermined control condition. This can be shown by following if-then statement: if (P=1) then (R2 ← R1)
- Here P is a control signal generated in the control section.

Control Function

- A control function is a Boolean variable that is equal to 1 or 0. The control function is shown as:
- **P: R2 ← R1**
- The control condition is terminated with a colon. It shows that transfer operation can be executed only if P=1.
-

Micro-Operations

- The operations executed on data stored in registers are called micro-operations. A micro-operation is an elementary operation performed on the information stored in one or more registers.
- **Example:** Shift, count, clear and load.

Types of Micro-Operations

- The micro-operations in digital computers are of 4 types:
- Register transfer micro-operations : transfer binary information from one register to another.
- Arithmetic micro-operations : perform arithmetic operations on numeric data stored in registers.
- Logic micro-operations perform : bit manipulation operation on non-numeric data stored in registers.
- Shift micro-operations: perform shift micro-operations performed on data.

Arithmetic Micro-Operations

- Some of the basic micro-operations are addition, subtraction, increment and decrement.
- Add Micro-Operation
- It is defined by the following statement:
- R3 ← R1 + R2
- The above statement instructs the data or contents of register R1 to be added to data or content of register R2 and the sum should be transferred to register R3.

- Subtract Micro-Operation
- Let us again take an example:

- R3 ← R1 + R2' + 1
- In subtract micro-operation, instead of using minus operator we take 1's compliment and add 1 to the register which gets subtracted, i.e

R1 - R2 is equivalent to R3 ← R1 + R2' + 1

Increment/Decrement Micro-Operation

- Increment and decrement micro-operations are generally performed by adding and subtracting 1 to and from the register respectively.

- R1 ← R1 + 1
- R1 ← R1 – 1
- Symbolic Designation Description
- R3 ← R1 + R2 Contents of R1+R2 transferred to R3.
- R3 ← R1 - R2 Contents of R1-R2 transferred to R3.
- R2 ← (R2)' Compliment the contents of R2.
- R2 ← (R2)' + 1 2's compliment the contents of R2.
- R3 ← R1 + (R2)' + 1 R1 + the 2's compliment of R2 (subtraction).
- R1 ← R1 + 1 Increment the contents of R1 by 1.
- R1 ← R1 - 1 Decrement the contents of R1 by 1.

Logic Micro-Operations

- These are binary micro-operations performed on the bits stored in the registers. These operations consider each bit separately and treat them as binary variables.

- Let us consider the X-OR micro-operation with the contents of two registers R1 and R2.

- P: R1 ← R1 X-OR R2
- In the above statement we have also included a Control Function.

- Assume that each register has 3 bits. Let the content of R1 be 010 and R2 be 100. The X-OR micro-operation will be:

Shift Micro-Operations

- These are used for serial transfer of data. That means we can shift the contents of the register to the left or right. In the **shift left** operation the serial input transfers a bit to the right most position and in **shift right** operation the serial input transfers a bit to the left most position.

There are three types of shifts as follows:

- a) Logical Shift

- It transfers 0 through the serial input. The symbol "shl" is used for logical shift left and "shr" is used for logical shift right.

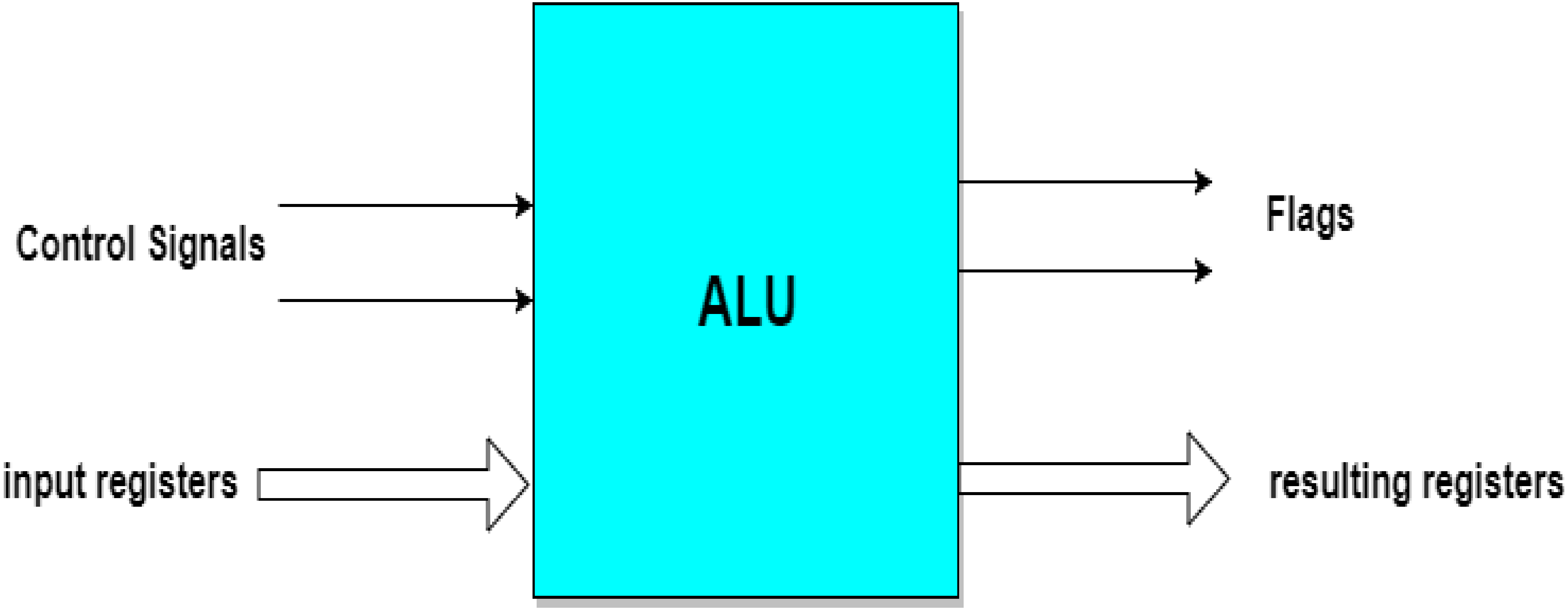
- R1 ← she R1
- R1 ← she R1
- The register symbol must be same on both sides of arrows.

- **b) Circular Shift**
- This circulates or rotates the bits of register around the two ends without any loss of data or contents. In this, the serial output of the shift register is connected to its serial input. **"cil"** and **"cir"** is used for circular shift left and right respectively.
-

- **c) Arithmetic Shift**
- This shifts a signed binary number to left or right. An **arithmetic shift left** multiplies a signed binary number by 2 and **shift left** divides the number by 2. Arithmetic shift micro-operation leaves the sign bit unchanged because the signed number remains same when it is multiplied or divided by 2.

Arithmetic Logical Unit

- Instead of having individual registers performing the micro-operations, computer system provides a number of registers connected to a common unit called as Arithmetic Logical Unit (ALU).
- ALU is the main and one of the most important unit inside CPU of computer. All the logical and mathematical operations of computer are performed here.
- The contents of specific register is placed in the in the input of ALU. ALU performs the given operation and then transfer it to the destination register.


Control Signals
input registers
ALU
Flags
resulting registers

Instruction Cycle

- An instruction cycle, also known as **fetch-decode-execute cycle** is the basic operational process of a computer. This process is repeated continuously by CPU from boot up to shut down of computer.
- Following are the steps that occur during an instruction cycle:
- Fetch
- Decode
- Execute

1. Fetch the Instruction

- The instruction is fetched from memory address that is stored in PC(Program Counter) and stored in the instruction register IR. At the end of the fetch operation, PC is incremented by 1 and it then points to the next instruction to be executed.

2. Decode the Instruction

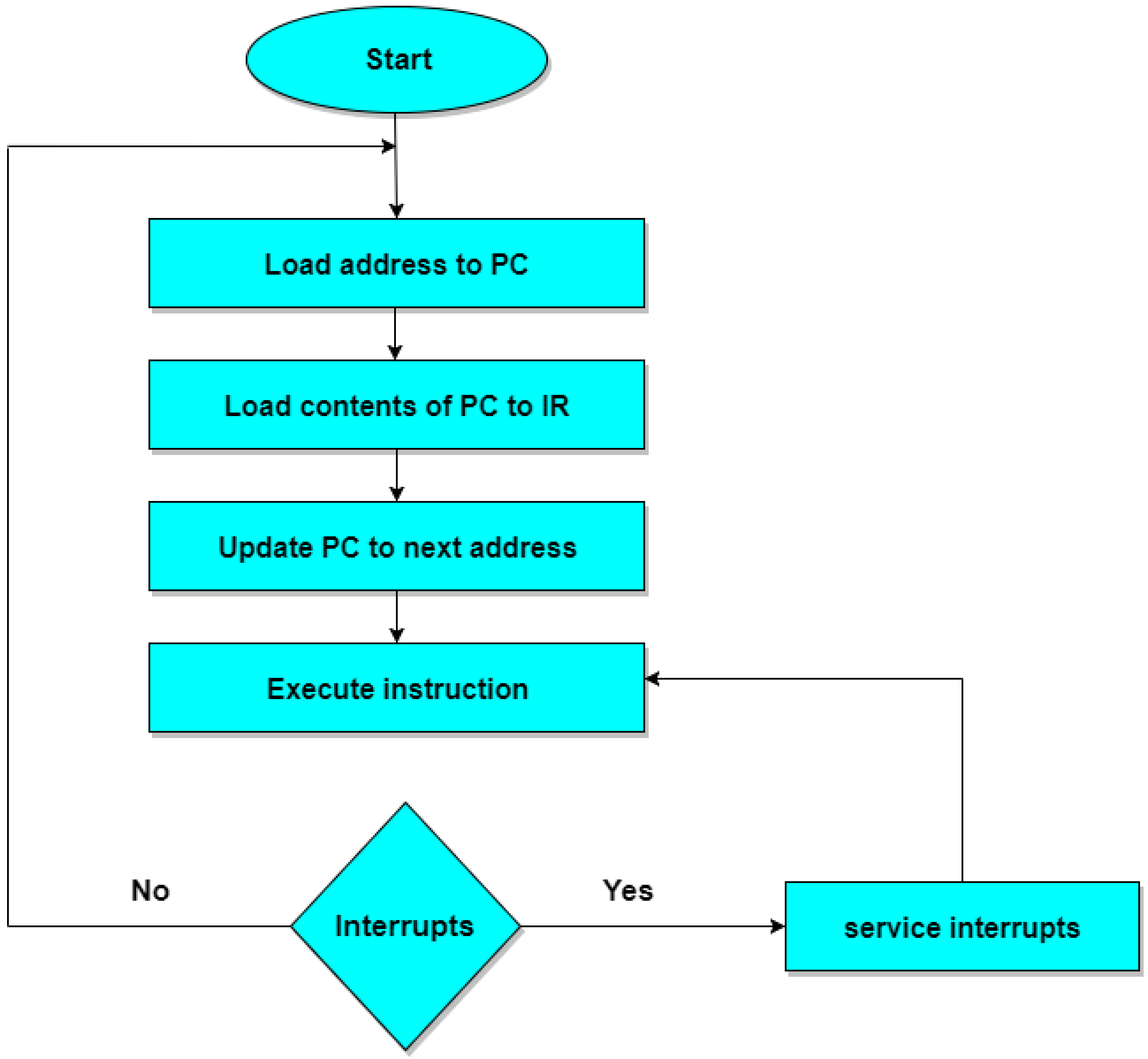
- The instruction in the IR is executed by the decoder.

3. Read the Effective Address

- If the instruction has an indirect address, the effective address is read from the memory. Otherwise operands are directly read in case of immediate operand instruction.
-

4. Execute the Instruction

- The Control Unit passes the information in the form of control signals to the functional unit of CPU. The result generated is stored in main memory or sent to an output device.


Start
Load address to PC
Load contents of PC to IR
Update PC to next address
Execute instruction
No
Interrupts
Yes
service interrupts