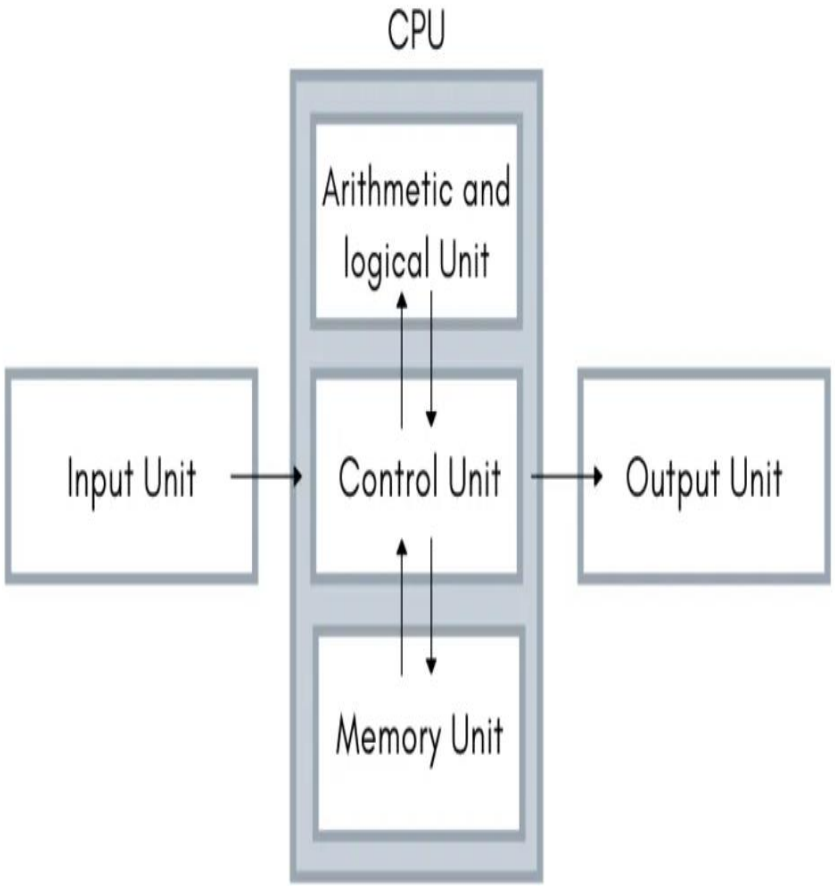
FUNDAMENTAL CONCEPTS

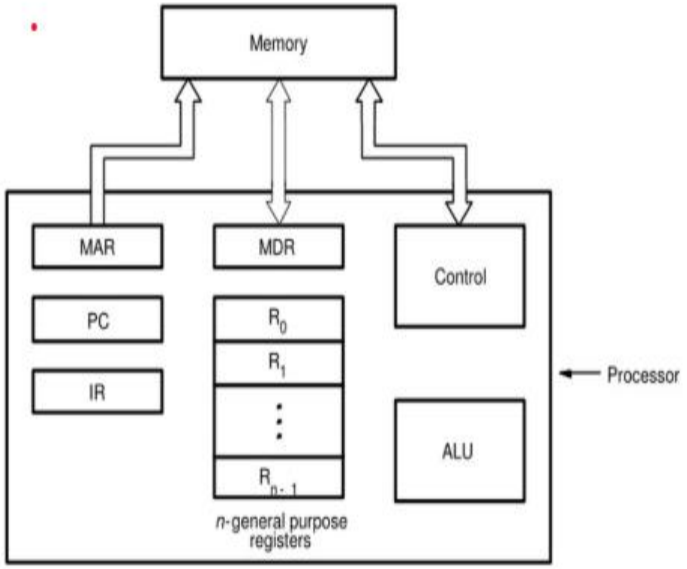
UNIT 2

Fundamental Concepts

- Processor fetches one instruction at a time and perform the operation specified.
- Instructions are fetched from successive memory locations until a branch or a jump instruction is encountered.
- Processor keeps track of the address of the memory location containing the next instruction to be fetched using Program Counter (PC).
- Instruction Register (IR)



Connection Between the Processor and the Memory



Connections between the processor and the memory.

Register

The registers are an integral part of the CPU.

They are a type of **memory** that can be accessed very quickly compared to other types of memory. The pieces of information they hold are needed very often. They can be used to **store data** and **control information** during a fetch-decode-execute cycle or they can be used to hold values that are generated as part of the ALU working on data.

SOME OF THE REGISTERS

- **Program Counter (PC)**
- **Instruction Register (IR)**
- **Memory Address Register (MAR)**
- **Memory Data Register (MDR)**

- General Purpose Registers

A register is a collection of flip-flops. Single bit digital data is stored using flip-flops. By combining many flip-flops, the storage capacity can be extended to accommodate a huge number of bits. We must utilize an n-bit register with n flip flops if we wish to store an n-bit word.

General Purpose Registers (GPRs) are essential components within a CPU, serving as temporary storage locations for data that is actively being processed. Unlike special-purpose registers, which are dedicated to specific tasks, GPRs are versatile and can store a wide variety of data, including operands for arithmetic and logic operations, memory addresses, or intermediate results.

They provide fast access to frequently used data, reducing the need to fetch from slower main memory and thereby speeding up program execution. Their contents can be read/written by programmer /user through instructions.

- PROGRAM COUNTER

The program counter, also known as the instruction pointer or simply PC, is a fundamental component of a computer's central processing unit (CPU). It is a special register that keeps track of the memory address of the next instruction to be executed in a program.

- INSTRUCTION REGISTER

  The Instruction Register (IR) is a crucial component of a computer's Central Processing Unit (CPU). It holds the instruction currently being executed or decoded. The IR is part of the control unit of the CPU and plays a vital role in the instruction cycle, which includes fetching, decoding, and executing instructions.

Function and Workflow

The instruction register temporarily stores the instruction fetched from memory. This instruction is then decoded to determine the operation to be performed and the operands involved. The decoded instruction is executed by the CPU, and the results are stored back in memory or other registers.

Step-by-Step Process

1. Fetch Phase: The CPU fetches the next instruction from memory and loads it into the instruction register. This step ensures that the CPU knows what operation to execute next.
2. Decode Phase: The instruction in the IR is decoded to interpret the command. This involves determining the operation and the operands.
3. Execute Phase: The decoded instruction is executed. For example, an addition operation would add the contents of two registers and store the result in a third register.

- Memory Address Register (MAR)

  The Memory Address Register (MAR) is a crucial component within a computer's CPU. It is responsible for storing the memory address from which data will be fetched to the CPU registers or the address to which data will be sent and stored via the system bus. Essentially, the MAR holds the memory location of data that needs to be accessed during the execution phase of an instruction.

Key Functions of MAR

- Fetching Data: When the CPU needs to read data from memory, the MAR holds the address of the data to be fetched.
- Storing Data: When the CPU needs to write data to memory, the MAR holds the address where the data will be stored.
- Coordination with MDR: The MAR works in conjunction with the Memory Data Register (MDR) to facilitate the transfer of data between the CPU and memory.

- Memory Data Register (MDR)

  The Memory Data Register (MDR) The Memory Data Register (MDR) is a crucial component in CPU architecture, serving as a buffer for data during transfer to and from main memory. It works with the Memory Address Register (MAR) to facilitate read and write operations, influencing the speed and efficiency of data processing.

Executing an Instruction

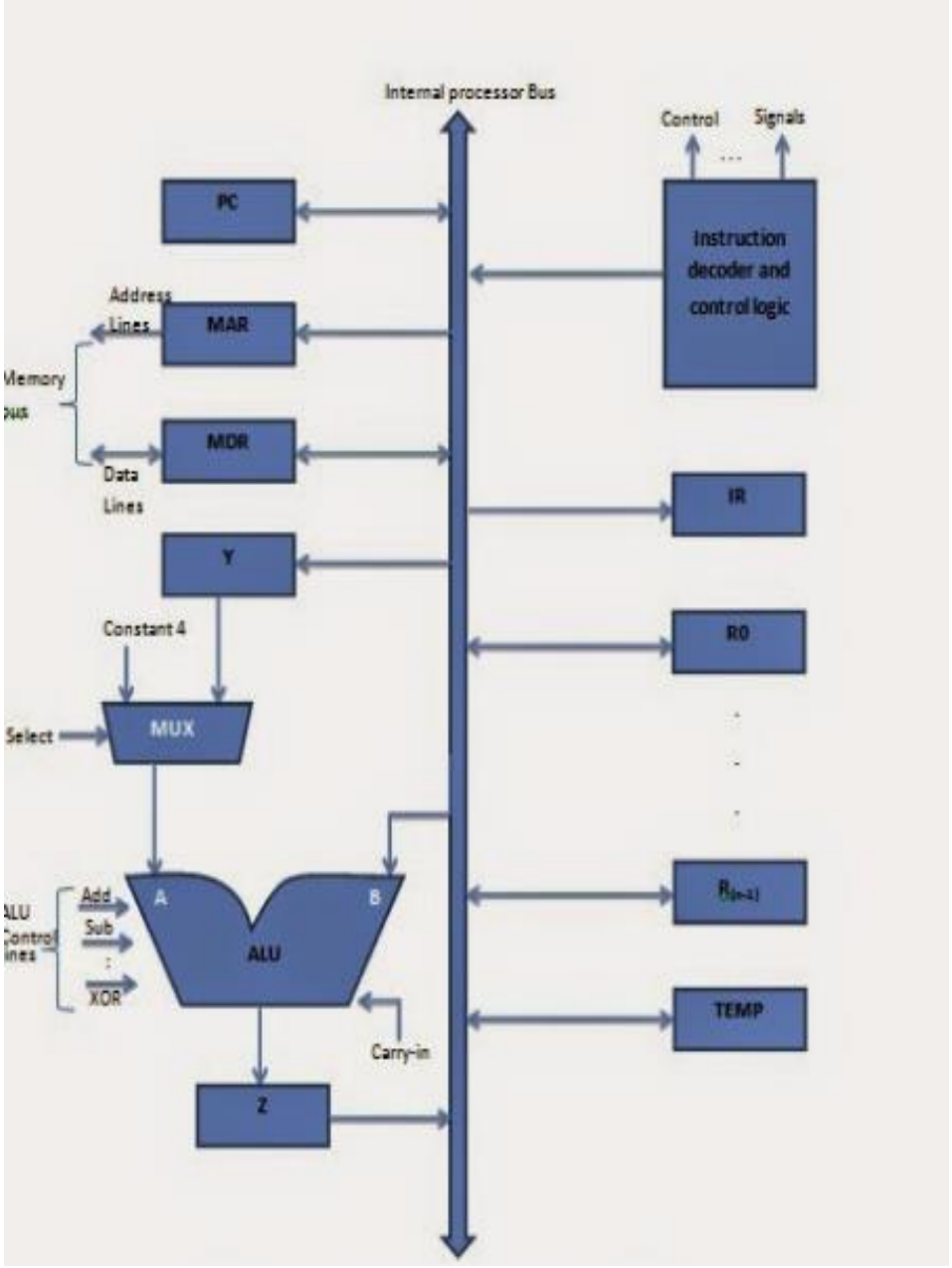
- Fetch the contents of the memory location pointed to by the PC. The contents of this location are loaded into the IR (fetch phase).

$$IR \leftarrow [[PC]]$$

- Assuming that the memory is byte addressable, increment the contents of the PC by 4 **(Fetch phase)**.

$$PC \leftarrow [PC] + 4$$

- Carry out the actions specified by the instruction in the IR **(Execution phase)**.



SINGLE BUS ORGANISATION

Single Bus Organization

- ALU
- Registers for temporary storage
- Various digital circuits for executing different micro operations.(gates, MUX, decoders, counters).
- Internal path for movement of data between ALU and registers.
- Driver circuits for transmitting signals to external units.
- Receiver circuits for incoming signals from external units.

Single Bus Organization - contd.

Program Counter (PC)

- Keeps track of execution of a program
- Contains the memory address of the next instruction to be fetched and executed.

Memory Address Register (MAR)

- Holds the address of the location to be accessed.
- I/P of MAR is connected to Internal bus and an O/P to external bus.

Memory Data Register (MDR)

- It contains data to be written into or read out of the addressed location.
- It has 2 inputs and 2 outputs.
- Data can be loaded into MDR either from memory bus or from internal processor bus.
- The data and address lines are connected to the internal bus via MDR and MAR

Single Bus Organization - contd.

Registers

- The processor registers R_0 to R_{n-1} vary considerably from one processor to another.
- Registers are provided for general purpose used by programmer.
- Special purpose registers-index & stack registers.
- Registers Y, Z & TEMP are temporary registers used by processor during the execution of some instruction.

Multiplexer

- Select either the output of the register Y or a constant value 4 to be provided as input A of the ALU.
- Constant 4 is used by the processor to increment the contents of PC.

Single Bus Organization - contd.

ALU

- B input of ALU is obtained directly from processor-bus.
- As instruction execution progresses, data are transferred from one register to another, often passing through ALU to perform arithmetic or logic operation.

Data Path

- The registers, ALU and interconnecting bus are collectively referred to as the data path.

- DATA PATH

A data path (or datapath) is a critical component of a computer's architecture, consisting of various fu nctional units such as:

- Arithmetic Logic Units (ALUs): Perform arithmeti c and logical operations.
- Registers: Temporary storage locations for data b eing processed.
- Buses:Communication pathways that transfer da ta between components within the CPU.

An Instruction Can Be Executed By Performing One Or More Of The Following Operations

- Transfer a word of data from one processor register to another or to the ALU
- Perform an arithmetic or a logic operation and store the result in a processor register
- Fetch the contents of given memory location and load them into processor register
- Store a word of data from a processor register into a given memory location

- An Instruction code is a group of bits that instructs the computer to perform a specific operation.

- It is divided into two parts and the most important parts of an instruction code is its operation code or opcode. The second part is the address part which specifies the memory address.

1. Register Transfers

- The input and output gates for register Ri are controlled by signals is Ri_{in} and Ri_{out}
  - Ri_{in} is set to 1 – data available on common bus are loaded into Ri.
  - Ri_{out} is set to 1 – the contents of register are placed on the bus.
  - Ri_{out} is set to 0 – the bus can be used for transferring data from other registers .
- All operations and data transfers within the processor take place within time-periods defined by the processor clock.
- When edge-triggered flip-flops are not used, 2 or more clock-signals may be needed to guarantee proper transfer of data. This is known as multiphase clocking.

Data Transfer Between Two Registers

Example:

- Transfer the contents of R1 to R4.
- Enable output of register R1 by setting $R1_{out}$=1. This places the contents of R1 on the processor bus.
- Enable input of register R4 by setting $R4_{in}$=1. This loads the data from the processor bus into register R4.

Input and Output Gating for Registers

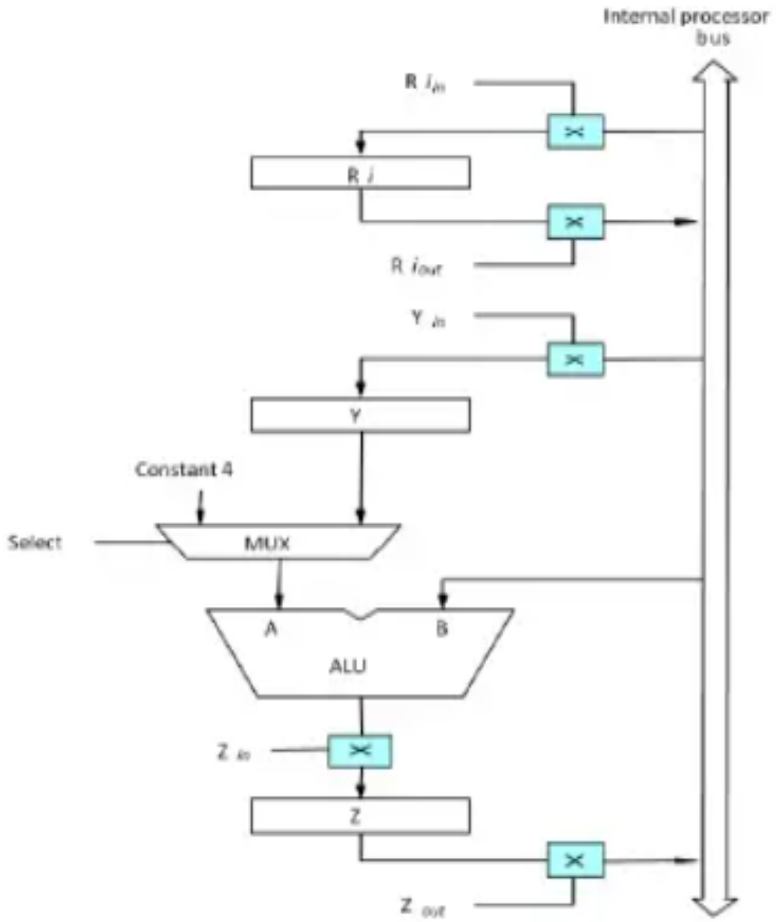


Figure. Input and Output Gating for the Registers

Input and Output Gating for One Register Bit

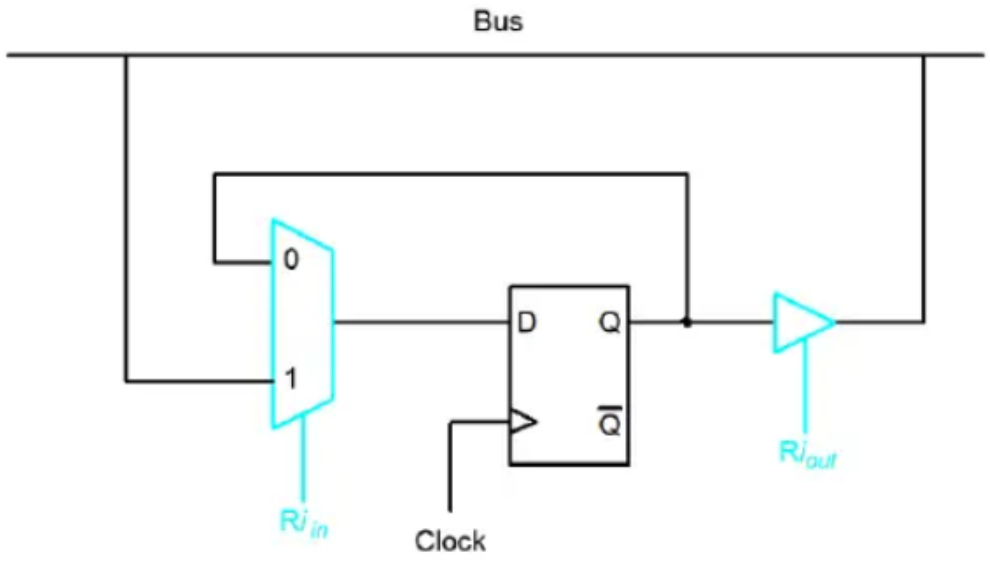


Figure. Input and Output Gating for One Register Bit

Input and Output Gating for One Register Bit - contd

- A 2-input multiplexer is used to select the data applied to the input of an edge-triggered D flip-flop.
  - When Ri_{in}=1, mux selects data on bus. This data will be loaded into flip-flop at rising-edge of clock.
  - When Ri_{in}= 0, mux feeds back the value currently stored in flip-flop.
- Q output of flip-flop is connected to bus via a tri-state gate.
  - When Ri_{out}= 0, gate's output is in the high-impedance state. (This corresponds to the open circuit state of a switch).
  - When Ri_{out}=1, the gate drives the bus to 0 or 1, depending on the value of Q.