Computer Organization and Embedded System

Control and Central Processing Units (ACtE0401)

Ishwar Kumar Singh
Computer Engineer
Government of Nepal

4.1 Control and central processing units (ACtE0401) :

- Control Memory,
- Addressing Sequencing,
- Computer Configuration,
- Microinstruction Format,
- Design of Control Unit,
- CPU Structure and Function,
- Arithmetic and Logic Unit,
- Instruction Formats,
- Addressing Modes,
- Data Transfer and Manipulation,
- RISC and CISC Pipelining
- Parallel Processing.

Control Memory

- Any computer that involves microprogrammed control, consists of two memories: Main memory & Control memory.
- The **main memory** consist of **machine instructions and data** which shall be modified by the user.
- The **control memory** consists of **microprograms** that are fixed and cannot be modified frequently.
  - They contain **microinstructions** that specify the **internal control signals** required to execute register **micro-operations**.
- The machine instructions generate a chain of microinstructions in the control memory.

Control Memory

- The function of the control memory is to generate micro-operations, compute the effective address, execute the operation, and return control to fetch phase and continue the cycle.
- The Micro-instruction contains a **Control Word** and a **Sequencing Word**
  - Control Word: All the **control information** required for **one clock cycle**.
  - Sequencing Word: Information needed to decide **next micro-instruction address**.

Important Registers

- Accumulator: This is the most frequently used register used to store data taken from memory.
- Memory Address Registers (MAR): It holds the address of the location to be accessed from memory.
- Memory Data Registers (MDR): It contains data to be written into or to be read out from the addressed location.
- MAR and MDR together facilitate the communication of the CPU and the main memory.

Important Registers

- Program Counter (PC): It contains the memory address of the next instruction to be fetched.
- PC is used to keep the track of the execution of the program or count the number of instructions.
- Instruction Register (IR): The IR holds the instruction which is just about to be executed.
- The instruction from the PC is fetched and stored in IR after which , the CPU starts executing the instruction and the PC points to the next instruction to be executed

Addressing sequencing

- Microinstructions are saved in control memory in groups which are described as routines.

Steps:

- An initial address is loaded into the control address register, when power is turned on.
- This address is usually the address of the first microinstruction that activates the instruction fetch routine.
- The control memory go through the routine that determines the effective address of the operand.
- The micro-operations are then generated which will execute the instruction.

Addressing sequencing

- The transformation from the bits of instruction code into an address in the control memory where routine is located is known as mapping process.
- The control memory required the capabilities of address sequencing as:
  - Incrementing the Control address register
  - Conditional branch or unconditional branch on the basis of the status bit conditions
  - Facility for subroutine calls and returns.
  - Mapping process from the bits of the instruction to an address

Computer Configuration

- Micro-code are required to be generated for control memory and the process is called as microprogramming for the computer configuration
- Two memory Units: Main Memory and Control Memory
- Four registers are associated with the processor units: Program Counter (PC), Address Register (AR), Data Register (DR) and Accumulator (AC)
- The control unit has Control Address Register (CAR) and a Subroutine Register (SBR)
- The transfer of information among the registers in the processor is done through Multiplexers.

Microinstruction Format

- A microinstruction format includes 20 bits and are divided into four elements

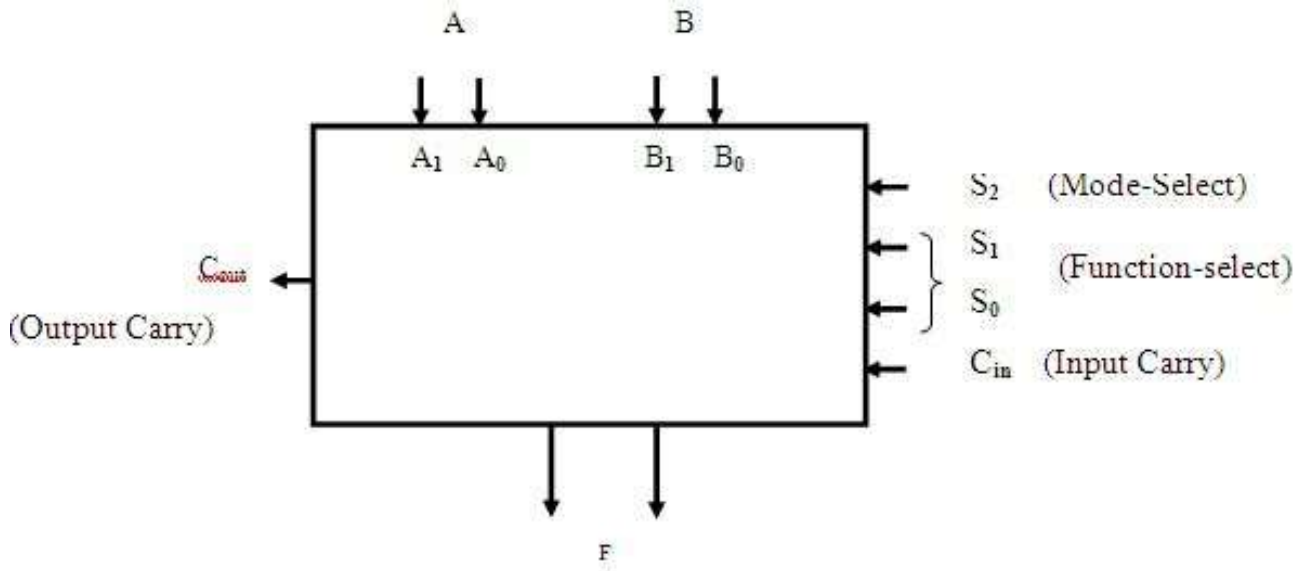
- F1, F2, F3 are the micro-operation fields which determine micro-operations.
- CD is the condition for branching which choose the status bit conditions.
- BR is the branch field which determines the type of branch.
- AD is the address field which specifies a value for the address field of the microinstruction.

Microinstruction Format

- The micro-operations are divided into three fields (F1, F2 & F3) of three bits each.
  - The three bits can define seven different micro-operations, in total there are 21 operations.
- A condition field includes 2 bits which are encoded to define four status bit conditions. (Unconditional Branch: U, Indirect address bit (DR): I, Sign bit of AC: S, Zero value in AC: Z)
- The branch field includes 2 bits which can be used by connecting with the address field to select the address for the next microinstruction. (JMP, CALL, RET, MAP)
- The address filed includes the value (Symbolic address, NEXT, Empty)
- RET, MAP are independent of the values of CD and AD fields.

Design of ALU

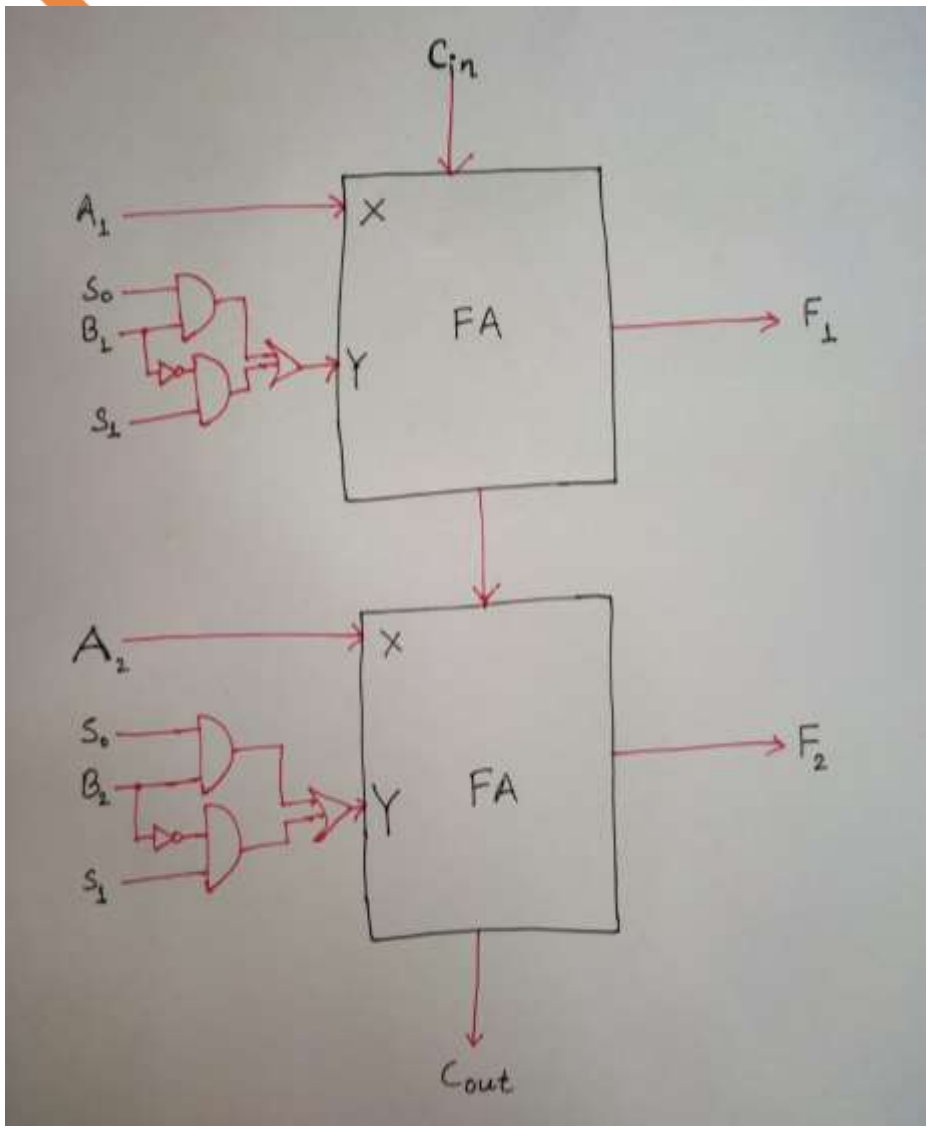
- Has a no of selection lines to select a particulars operation in Unit
- The selection lines are decoded within the ALU so that ‘k’ selection variables can specify up to ‘2^k’ distinct operations.
- S_2 distinguishes between arithmetic and logical operations, S_1 & S_0 specify particular arithmetic or logical operation
- With these 3 selection variables it is possible to Specify 4 arithmetic operation (with S_2 in one state) and 4 logical operations (with S_2 in other state).



Design of ALU

The design of ALU has three stages:

1. Design the Arithmetic Section
2. Design the Logical Section
3. Combine Both Sections

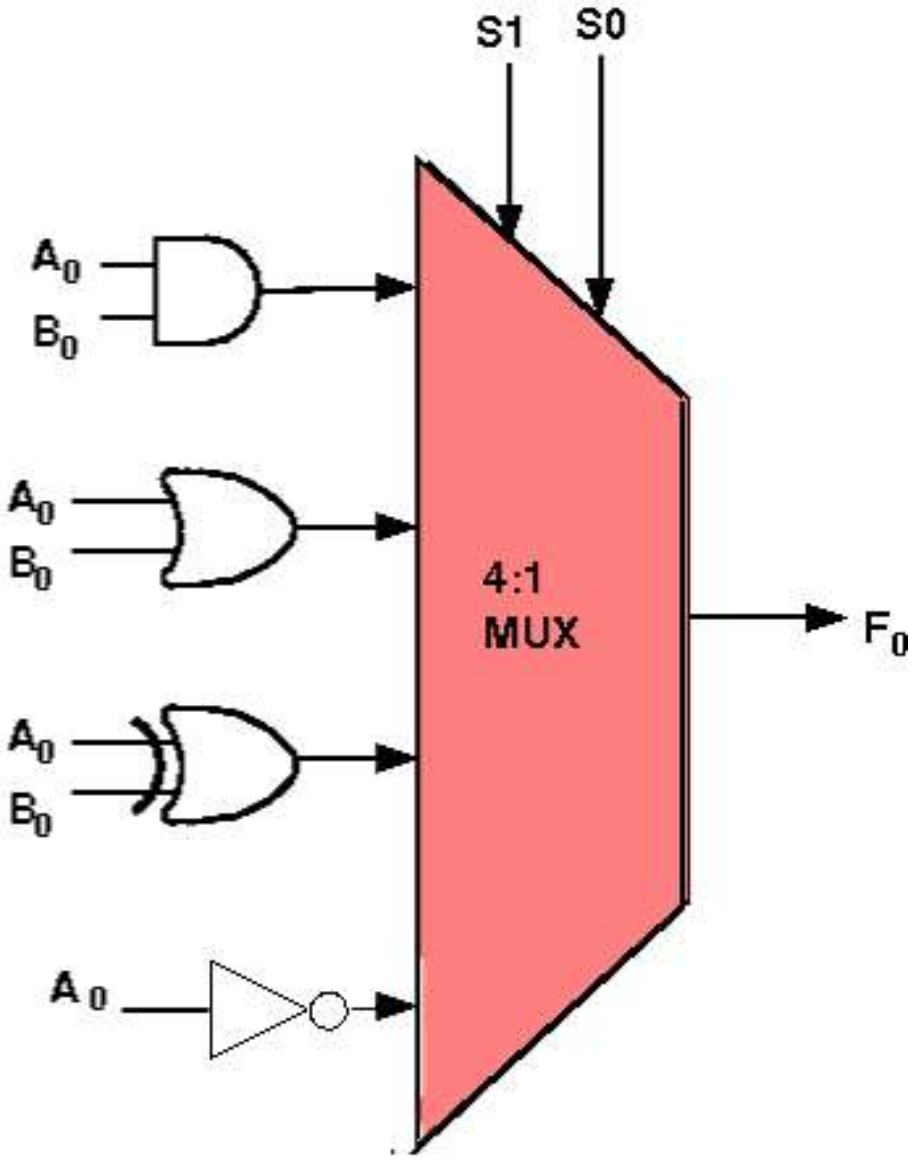


Design of Arithmetic Section

- The basic component of arithmetic circuit is a parallel adder which is constructed with a number of full adder circuits connected in cascade.
- By controlling the data inputs to the parallel adder, it is possible to obtain different types of arithmetic operations.

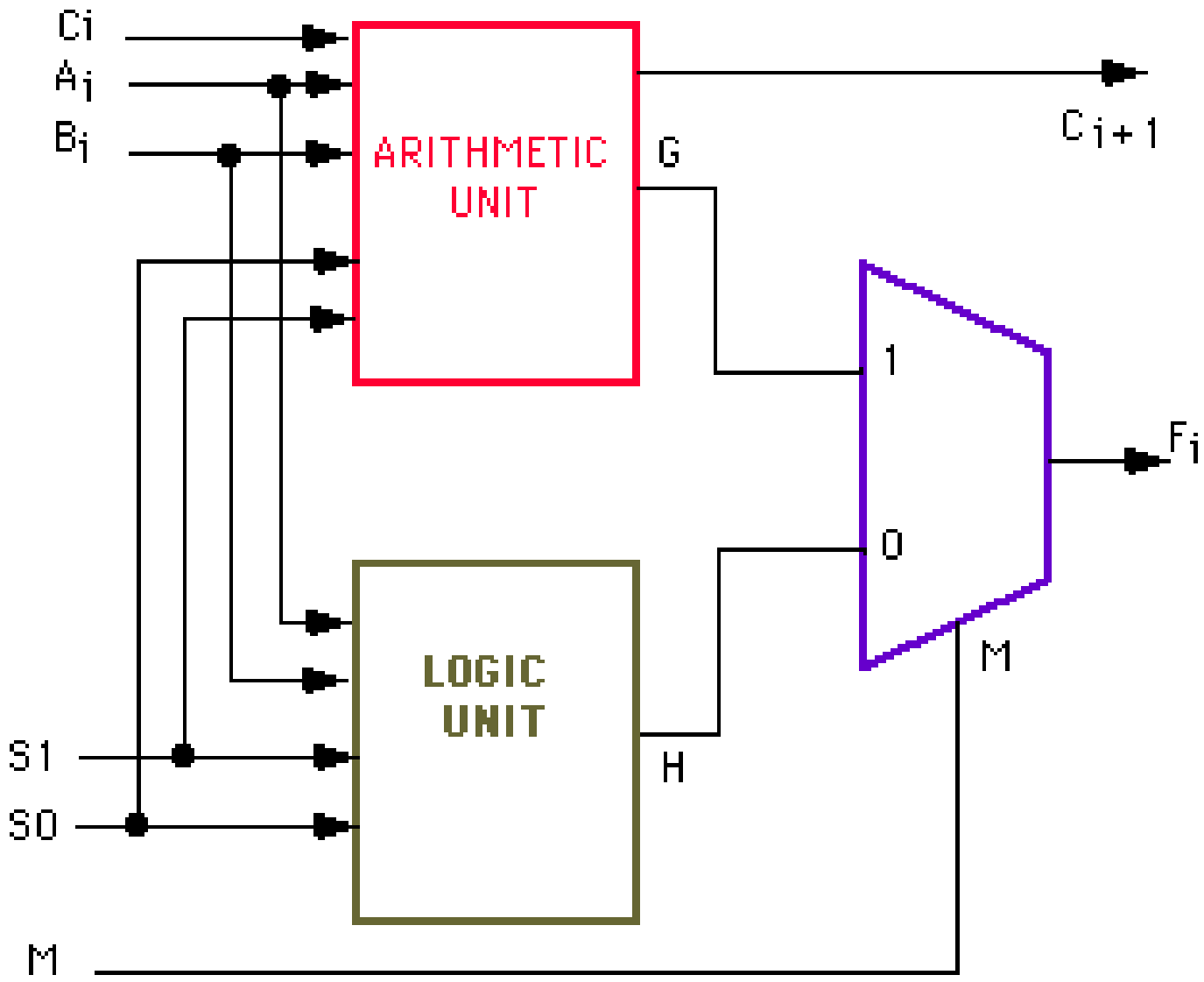
Design of Logical Section

- The basic components of logical circuit are AND, OR, XOR and NOT gate circuits connected accordingly with multiplexer.
- Each of four logic operations is generated through a gate that performs the required logic.
- The selection lines of multiplexer choose one of the gate operation and directs its value to the output.



Combining Both Sections

- By combining both Arithmetic and Logical Section, A mode select differentiates between Arithmetic and Logical Operation.



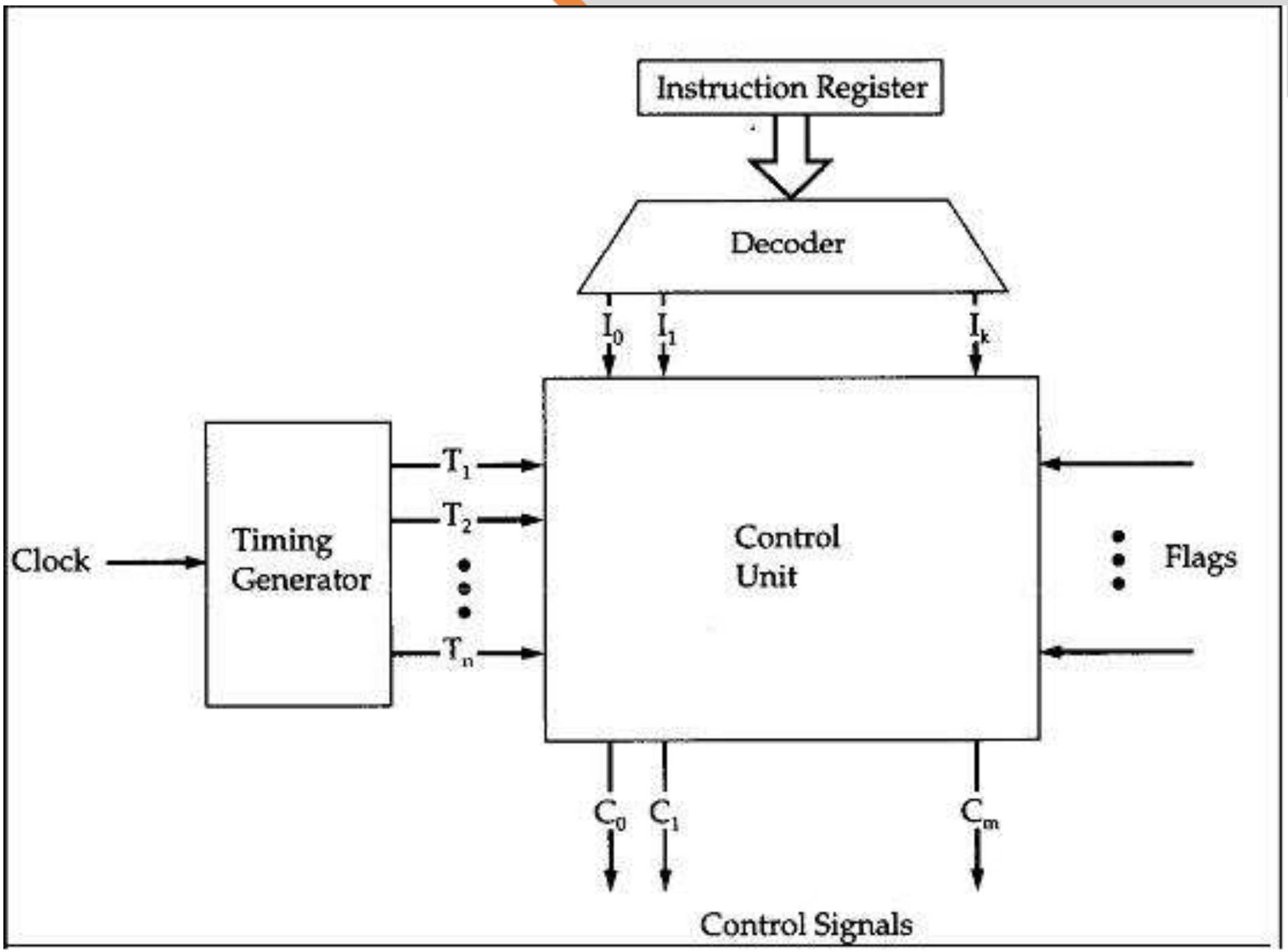
Design of CU

- The function of the control unit in a digital computer is to initiate sequences of micro-operations.
- In a bus-organized systems, the control signals that specify micro-operations, are groups of bits that select the paths in multiplexers, decoders and arithmetic logic units.
- There are two approaches in the design of CU :
  - When the control signals are generated by hardware using conventional logic design techniques, the control unit is said to be hardwired.
  - A control unit whose binary control variables are stored in memory is called a microprogrammed control unit.

Hardwired Control Unit

- In this, CU is designed as clocked Sequential circuit.
- It is implemented with logic gates, flip-flops, decoders, multiplexers & other logic building blocks.
- Its i/p signals are transformed into set of o/p logic signal which are known as control signals.
- for each control signal, a Boolean expression of that signal as a function of the inputs is driven.

Hardwired Control Unit



Hardwired Control Unit

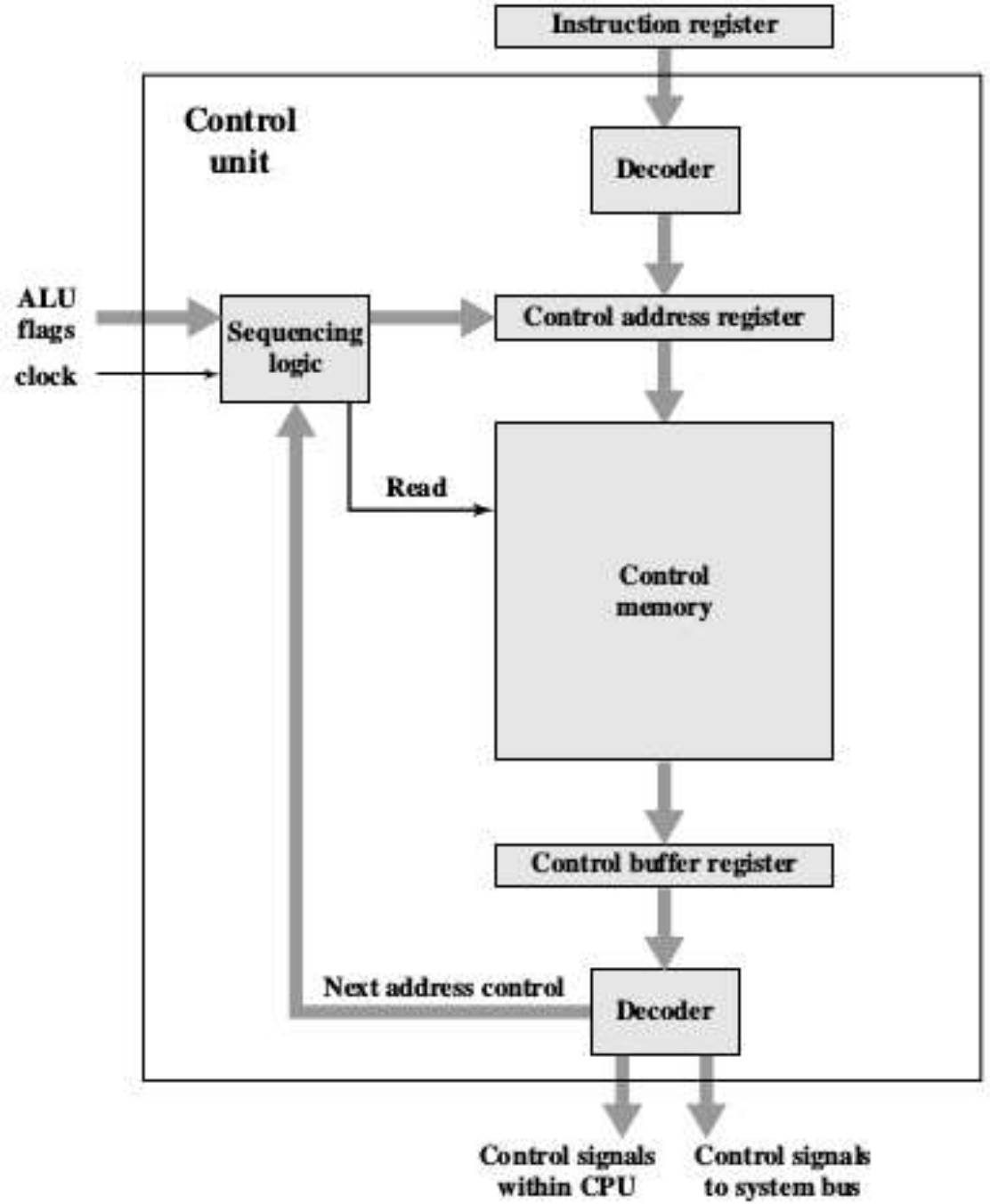
Problems With Hardwired Designs

- Complex sequencing & micro-operation logic
- Difficult to design and test
- Inflexible design
- Difficult to add new instructions

Micro-Programmed CU

- In this all the control functions that can simultaneously be activated are grouped together to form Control Words and are stored in a separate memory called Control Memory.
- The control words are fetched from the control memory and the Individual control field that constitutes a control word are routed to various functional units to enable appropriate processing hardware.
- When the parts of processing unit are driven sequentially, a particular task is performed.
- A computer that employs a microprogrammed control unit will have two separate memories:
  - A main memory
  - A control memory

Micro Programmed CU



Micro-Programmed CU

Merits

- Eases tasks of design & implementation of CU.
- Provides the support to the family concept.
- Well-structured control organization
- Systematic, flexible and less error prone

Demerits

- May cost more due to cost of control Memory
- Lower speed due to retrieval process of micro-instruction

Comparison between two Approaches

- The most important advantage of micro-programming is that it provide a well-structured control organization & control functions are systematically transformed into programing discipline & a regular memory replaces combinational logic circuit.
- With micro-programming, the contexts of the micro program can be change easily but a small change in hardwired CU may leads to redesign of entire CU
- The design of CU with control memory is more expensive than hardwired CU
- The microprogramming approach simplifies documentation and service training of the system.

Parallel Processing

- Parallel processing is a term used to denote a large class of techniques that are used to provide simultaneous data-processing tasks for the purpose of increasing the computational speed of a computer system.
- The purpose of parallel processing is to speed up the computer processing capability and increase its throughput.
- The amount of hardware increases with parallel processing, and with it, the cost of the system increases.
- There are a variety of ways that parallel processing can be classified:
  - Internal Organization of the processors
  - Interconnection structure between processors
  - The flow of information through the system

Parallel Processing

- M.J. Flynn considers the organization of a computer system by the number of instructions and data items that are manipulated simultaneously.
  - Single Instruction stream, Single Data stream (SISD)
  - Single Instruction stream, Multiple Data stream (SIMD)
  - Multiple Instruction stream, Single Data stream (MISD)
  - Multiple Instruction stream, Multiple Data stream (MIMD)

SISD

- Represents the organization of a single computer containing a control unit, a processor unit, and a memory unit.
- Instructions are executed sequentially and the system may or may not have internal parallel processing capabilities.
- Single Instruction is performed on a single set of data in a sequential form.
- Parallel processing may be achieved by means of multiple functional units or by pipeline processing.

SIMD

- Represents an organization that includes many processing units under the supervision of a common control unit.
- All processors receive the same instruction from the control unit but operate on different items of data
- The shared memory unit must contain multiple modules so that it can communicate with all the processors simultaneously.

MISD

- MISD structure is only of theoretical interest since no practical system has been constructed using this organization.

MIMD

- MIMD organization refers to a computer system capable of processing several programs at the same time.
- Flynn's classification depends on the distinction between the performance of the control unit and the data-processing unit.
- It emphasizes the behavioral characteristics of the computer system rather than its operational and structural interconnections.

Pipelining

- Pipelining is technique decomposing a sequential process into sub-operations, with each sub-process being executed in a special dedicated segment that operates concurrently with all other segments.
- The overlapping of computation is made possible by associating a register with each segment in the pipeline
- The registers provide isolation between each segment so that each can operate on distinct data simultaneously.
- Pipelining is one type of parallel processing which does not fit Flynn's classification.

Pipelining

- There are various reasons why the pipeline cannot operate at its maximum theoretical rate.
    - Different segments may take different times to complete their sub operation.
    - It is not always correct to assume that a non-pipe circuit has the same time delay as that of an equivalent pipeline circuit.
- There are two areas of computer design where the pipeline organization is applicable.
    - Arithmetic pipeline
    - Instruction pipeline

Arithmetic Pipeline

- Arithmetic Pipeline divides an arithmetic operation into sub-operations for execution in pipeline segments.
- Usually found in very high speed computers
- Used to implement floating-point operations, multiplication of fixed-point numbers and similar computations in scientific problem.

Instruction Pipeline

- Instruction Pipeline operates on a stream of instructions by overlapping fetch, decode and execute phases.
- Reads consecutive instructions from memory while previous Instructions are being executed in other segment
- In case of branching instruction after branch may be discarded.

Pipeline Conflicts/Hazards

- In general, there are three major difficulties that cause the instruction pipeline to deviate from its normal operations
  - Data Dependency
  - Resource conflicts
  - Branch Difficulties

Data Dependency

- Also called as Data Hazard Hazzard which arises when an instruction depends on the result of previous instruction, but this result is not yet available.
- Collision of data or address occur when an instruction can't procced because the previous instructions didn't complete certain operations.
- Address Dependency occurs when an operand address can't be calculated because the information needed by the addressing mode is not available.
- Pipelined computer deals with such conflicts between data dependencies in variety of ways:
  - Hardware Interlocks
  - Operand forwarding
  - Delayed Load

Hardware Interlocks

- Interlock is a circuit that detects instruction whose source operands are destination of instructions farther up in the pipeline
- Detection of this situations causes the instruction whose source is not available to be delayed by enough cycle to resolve the conflict.
- Maintains the program sequence by using hardware to insert the required delays.

Operand Forwarding

- Uses special hardware to detect a conflict and then avoid it by routing data through special paths between pipeline segments
- May require additional hardware paths through Mux as well as circuit that detects Conflict

Delayed Load

- The compiler is designed to detect a data Conflict and re-order instructions as necessary to delay the loading of the conflicting data by inserting no-operation instruction

Resource Conflicts

- Known as structural Hazzard
- Arises when two segments access to Memory at the same time
- Can be resolved by using separate memories for instructions and data.

Branch Conflicts

- Known as Control Hazard which arises from branch and other instructions that change the value of PC
- Break the sequence of instruction stream: Conditionally or Unconditionally.
- Pipelined computer employ various hardware techniques to minimize the performance degradation
  - **Pre-fetch target instruction:** to pre-fetch the target instruction in addition to the instruction following the Branch. Both are saved until the branch is executed.

Branch Conflicts

- **Branch target Buffer (BTB):** It is an associative memory included in the fetch segment of the pipeline. Each entry in the BTB consists of the address of a previously executed branch instruction and the target instruction for that branch
- **Branch prediction:** Uses some additional logic to guess the outcome of a conditional branch instruction before it is executed
- **Delayed Branch:** The compiler detects the branch instructions and re-arranges the machine language code sequence by inserting useful instructions that keep the pipeline operating without interruption

Thank You.