CHAPTER 6

Processor Organization

Learning objectives

- Introduction
- Computer Registers Set (Register Organization)
- Datapath
- CPU Instruction Cycle
- Control Unit

INTRODUCTION

The CPU (Central Processing Unit) or Processor is responsible for fetching, decoding and executing an instruction. A CPU is the brains of a computer. To be more specific, a CPU fetches program instructions from RAM (input), interprets and processes it (execution) and then sends back the computed results so that the relevant components can carry out the instructions. In this chapter we introduce the basic functions of processor.

Regarding computing power, the CPU is the most important element of a computer system. The CPU is consisted of thin layers of thousands of transistors. Each transistor receives a set of inputs and produces output. Transistors hold a key role in functioning of CPU as they make computer able to count and perform logical operations which is called processing. It processes the instructions that it collects by decoding the code in programs. Computers use two types of storage. Primary storage and secondary storage. The CPU mainly interacts with primary storage or main memory, referring to it for both instructions and data.

There are four important functions of CPU,

1. **Fetch :** the process to read the instruction from memory
2. **Decode :** the process to interpret the instruction
3. **Execute :** the process to perform the required arithmetic and logical operation.
4. **Write back :** the process to transfer the result of an execution of instruction.

The main components of CPU help it in performing various functions. The components of a CPU work together, and their making/manufacturing determine the complexity of operations as well as how fast they can be carried out.

Small Concept

Fetch ⇔ Read instruction, Decode ⇔ Convert to machine code
Execute ⇔ Run instruction, Write back ⇔ Store result

The three components of the CPU are following,

1. Arithmetic Logic Unit
2. Control Unit
3. Register Set (Memory Unit)

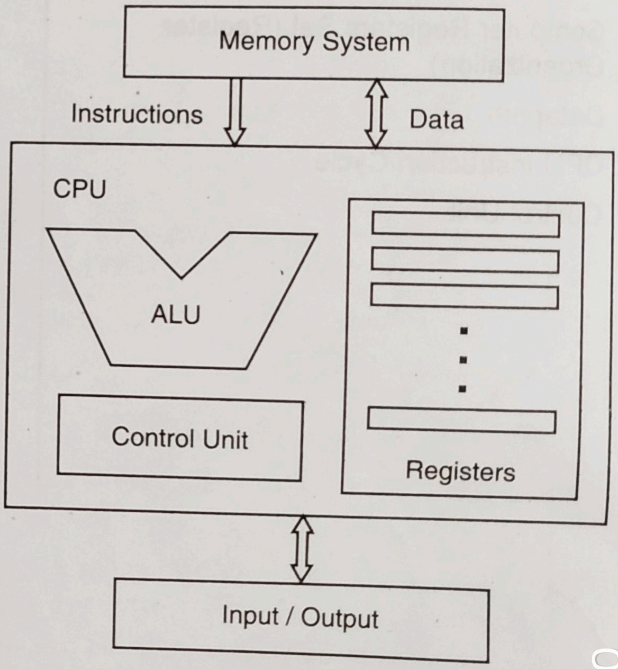

Fig. 6.1. Components of CPU

1. Arithmetic Logic Unit (ALU)

This is an electronic circuit in arithmetic logic unit (discussed in chapter -5), which executes all arithmetic and logical operations. It performs arithmetic calculations like as addition, subtraction, multiplication and division as well as comparisons. The unit can compare numbers, letters, or special characters. There can be more than one Arithmetic logic unit in a CPU, and these ALUs can also be used for the purpose of maintaining timers that help run the computer.

2. Control Unit (CU)

There is a circuit in the control unit which uses electrical signals to instruct the whole computer system for carrying out or executing, already stored program instructions. It controls and co-ordinates computer components. It extracts instructions from memory and decodes and executes them. In fact it regulates the flow of information through the processor. In short, it can be said , this component receives, decodes, stores results and manages execution of data that flows through the CPU. Its communication with both arithmetic unit and memory is inevitable.

3. Registers Set

Registers are temporary storage areas which are responsible for holding the data that is to be processed. They store the instructions and data in a processor. This data is further used by Control Unit. There are some registers that are set aside for specific tasks, these generally include a program counter, stack, and flags.

6.1. COMPUTER REGISTERS SET (REGISTER ORGANIZATION)

Registers are referred as extremely fast memory spaces within the CPU that are used to store the results of execution of the instructions in CPU. Different computers have different set of register. They differ in the number of registers, register types, and the length of each register. They also differ in the usage of each register.

General-purpose registers can be used for multiple purposes and assigned to a variety of functions by the programmer.

Special-purpose registers are restricted to only specific functions. In some cases, some registers are used only to hold data and cannot be used in the calculations of operand addresses. The length of a data register must be long enough to hold values of most data types. Some machines allow two contiguous registers to hold double-length values.

Address registers may be dedicated to a particular addressing mode or may be used as address general purpose. Address registers must be long enough to hold the largest address. The number of registers in a particular architecture affects the instruction set design. A very small number of registers may result in an increase in memory references.

Small Concept

All CPU registers has their unique identity and special role in computer system.

Another type of registers is used to hold processor status bits, or flags. These bits are set by the CPU as the result of the execution of an operation. The status bits can be tested at a later time as part of another operation.

- Memory Access Registers
- Condition Registers
- Instruction Fetching Registers
- Special-Purpose Address Registers

6.1.1. Memory Access Registers

There are two registers referred as Memory Access Registers

1. MDR (Memory Data Register)
2. MAR (Memory Address Register)

Two basic operations of memory are write operation and read operation, the registers memory data register (MDR) and memory address register (MAR) are respectively used for these operations. The MDR and MAR are used exclusively by the CPU. These registers are not directly accessible by developers. In order to perform a write operation into a specified memory location, these are used as following way:

1. The word to be stored into the memory location is first loaded by the CPU into MDR.
2. The address of the location into which the word is to be stored is loaded by the CPU into a MAR.
3. A write signal is issued by the CPU.

Similarly, to perform a memory read operation, the MDR and MAR are used as following way:

1. The address of the location from which the word is to be read is loaded into the MAR.
2. A read signal is issued by the CPU.
3. The required word will be loaded by the memory into the MDR ready for use by the

6.1.2. Instruction Fetching Registers

There are two main registers are involved for fetching an instruction for execution:

1. PC (Program Counter)
2. IR (Instruction Register)

The PC contains the address of the next instruction to be fetched. The fetched instruction is loaded in the IR for execution. After a successful instruction fetch, the PC is updated to point to the next instruction to be executed. In the case of a branch operation, the PC is updated to point to the branch target instruction after the branch is resolved, that is, the target address is known.

6.1.3. Condition Registers

Condition registers, or flags, are used to maintain status information. Some architectures contain a special program status word (PSW) register. The PSW contains bits that are set by the CPU to indicate the current status of an executing program. These indicators are typically for arithmetic operations, interrupts, memory protection information, or processor status.

The condition register indicates the status of the CPU as well as controls its operations. A condition register is a flag register which indicates some conditions produced by the execution of an instruction or controls certain operations of the CPU. A 16-bit flag register in the CPU contains nine active flags.

There are two types of flags:

1. Conditional or Status Flag
2. Machine control flags

Small Concept

The condition register indicates the status of CPU as well as controls its operations.

Conditional or status flags: Six flags are conditional flags. They are set or reset by the EU on the basis of the results of some arithmetic operation.

- **Carry Flag (CF):** indicates a carry after addition or a borrow after subtraction, also indicates error conditions.
- **Parity Flag (PF):** is a logic "0" for odd parity and a logic "1" for even parity.
- **Auxiliary Carry Flag (AF):** important for BCD addition and subtraction; holds a carry (borrow) after addition (subtraction) between bits position 3 and 4. Only used for DAA and DAS instructions to adjust the value of AL after a BCD addition (subtraction).
- **Zero Flag (ZF):** indicates that the result of an arithmetic or logic operation is zero.
- **Sign Flag (SF):** indicates arithmetic sign of the result after an arithmetic operation.
- **Overflow Flag (OF):** a condition that occurs when signed numbers are added or subtracted. An overflow indicates that the result has exceeded the capacity of the machine.

Machine control flags : The three remaining flags in the flags register are used to control certain operations of processor. They are called the control flags.

The control flags are deliberately set or reset with specific instructions you put in your program. The three control flags are:

- **Trap Flag (TF):** used for single stepping through a program;
- **Interrupt Flag (IF):** used to allow or prohibit the interruption of a program;
- **Direction Flag (DF):** used with string instructions.

6.1.4. Special-Purpose Registers

There are three main special purpose registers:

1. Index Register
2. Segment Pointers
3. Stack Pointers

1. Index Register

Index register used in index addressing mode (discussed in Chapter -7), to obtain the address of the operand by adding the content of the register. The index register holds an address displacement. Index addressing is indicated in the instruction by including the name of the index register in parentheses and using the symbol X to indicate the constant to be added.

2. Segment Pointers

In order to support segmentation, the address issued by the processor should consist of a segment number (base) and a displacement (or an offset) within the segment. A segment register holds the address of the base of the segment.

3. Stack Pointer

A stack is a data organization mechanism in which the last data item stored is the first data item retrieved. Two specific operations can be performed on a stack. These are the Push and the Pop operations. A specific register, called the stack pointer (SP), is used to indicate the stack location that can be addressed. In the stack push operation, the SP value is used to indicate the location (called the top of the stack). After storing (pushing) this value, the SP is incremented (in some architectures, e.g. X86, the SP is decremented as the stack grows low in memory).

6.2. DATAPATH

The CPU itself can be also segregated into two sections:

- Data section and
- Control section.

The Data section (Datapath), contains the registers and the ALU. The datapath is capable of performing certain operations on data items.

Small Concept

Data path is used for transfering data contents among the components of computer.

The Control section is basically the control unit, which issues control signals to the datapath. Internal to the CPU, data move from one register to another and between ALU and registers.

Internal data movements are performed via local buses, which may carry data, instructions, and addresses. Externally, data move from registers to memory and I/O devices, often by means of a system bus.

The bus organization may use one-bus, two-bus, or three-bus architecture.

6.2.1. One-Bus Organization

In this organization only one bus is used to move outgoing and incoming data. Since a bus can handle only a single data movement within one clock cycle, two-operand operations will need two cycles to fetch the operands for the ALU. Additional registers may also be needed to buffer data for the ALU.

Small Concept

One-bus organization uses only single bus for all inputs and outputs.

This bus organization is the cheapest and simple, but it limits the amount of data transfer that can be done in the same clock cycle. The process of data transfer in same clock will slow down the overall performance of the system.

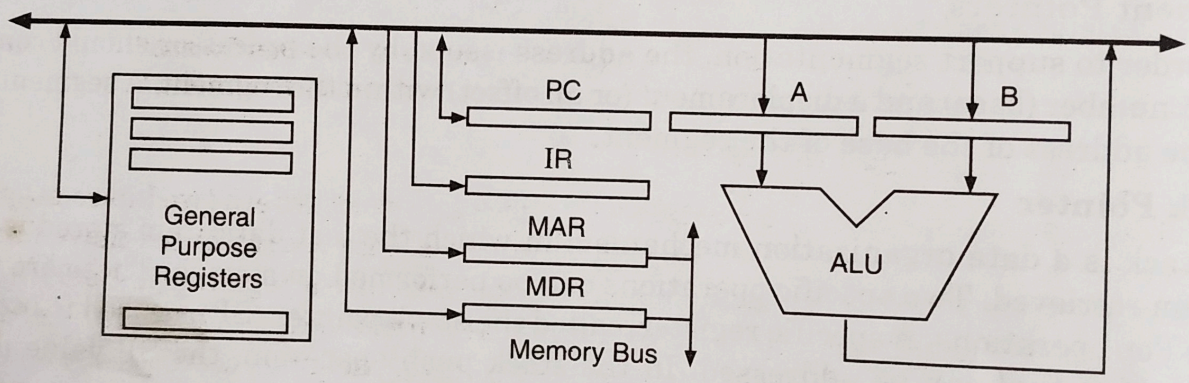

Fig. 6.2. One bus datapath

6.2.2. Two-Bus Organization

The two bus organization has fast performance than the one-bus organization. In this organization, general-purpose registers are directly connected to both buses. The content of two different registers can be transfer into the input point of the ALU at the same time. Therefore, two different operands of an operation can be fetched in the same clock cycle. An additional buffer register may be needed to hold the output of the ALU when the two buses are busy carrying the two operands.

Small Concept

Two-bus organization uses two buses one for input data and one for output data.

Figure 6.3(*a*) shows a two-bus organization. In some cases, one of the buses may be dedicated for moving data into registers (input-bus), while the other is dedicated for transferring data out of the registers (output-bus). In this case, the additional buffer register may be used as one of the ALU inputs, to hold one of the operands. The ALU output can be connected directly to the in-bus, which will transfer the result into one of the registers. Figure 6.3(*b*) shows a two-bus organization with in-bus and out-bus.

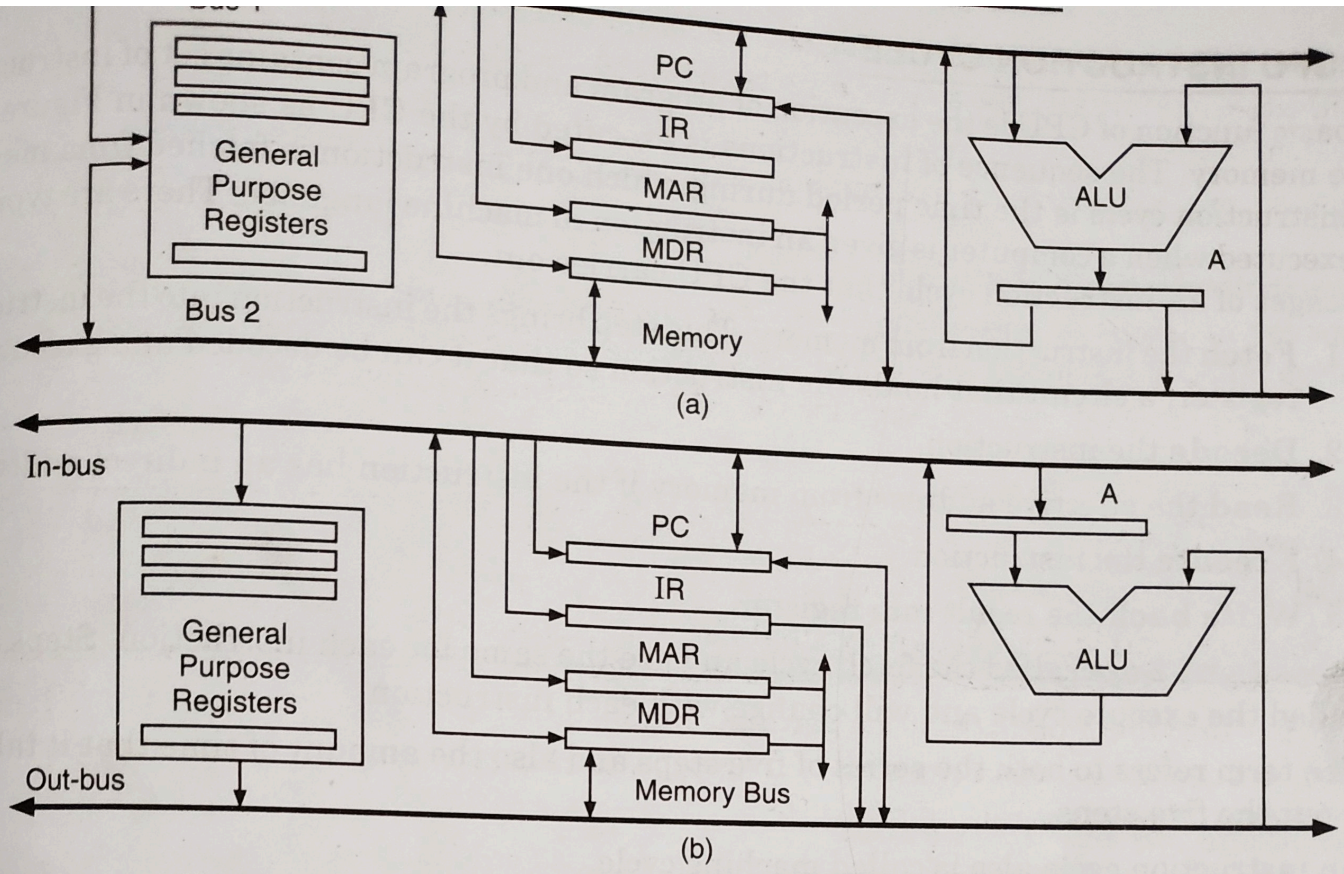

Fig. 6.3. Two-bus organizations. (a) An Example of Two-Bus Datapath
(b) Another Example of Two-Bus Datapath with in-bus and out-bus

6.2.3. Three-Bus Organization

In a three-bus organization, three different buses are used, where two buses may be use as source buses and the third is used as destination bus. The source buses move data out registers (out-bus), and the destination bus may move data into a register (in-bus). Each of th two out-buses is connected to an ALU input point. The output of the ALU is connected directl to the in-bus. As can be expected, the more buses we have, the more data we can move withi a single clock cycle. However, increasing the number of buses will also increase the complexit of the hardware. Figure 6.4 shows an example of a three-bus datapath.

Small Concept

Three-bus organization uses three buses two may use for inputs two may use for output, but one fully dedicated for inputs and one for output.

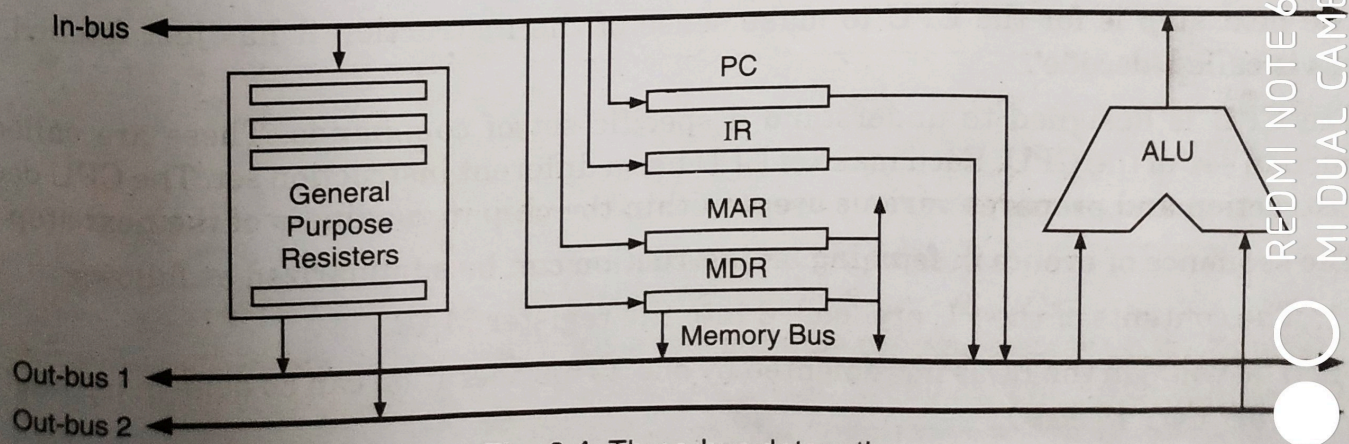

Fig. 6.4. Three-bus datapath

6.3. CPU INSTRUCTION CYCLE

The basic function of CPU is the execution of program and program contains set of instructions in the memory. The sequence of instructions is executed by the CPU as shown in Figure. 6.5. The instruction cycle is the time period during which one instruction is fetched from memory and executed when a computer is given an instruction in machine language. There are typically five stages of an instruction cycle that the CPU carries out:

1. **Fetch** the instruction from memory. This step brings the instruction into the instruction register, a circuit that holds the instruction so that it can be decoded and executed.
2. **Decode** the instruction.
3. **Read** the effective address from memory if the instruction has an indirect address.
4. **Execute** the instruction.
5. **Write back** the result into register.

Steps 1 and 2 are called the fetch cycle and are the same for each instruction. Steps 3 to 5 are called the execute cycle and will change with each instruction.

The term refers to both the series of five steps and also the amount of time that it takes to carry out the five steps.

An instruction cycle also is called machine cycle.

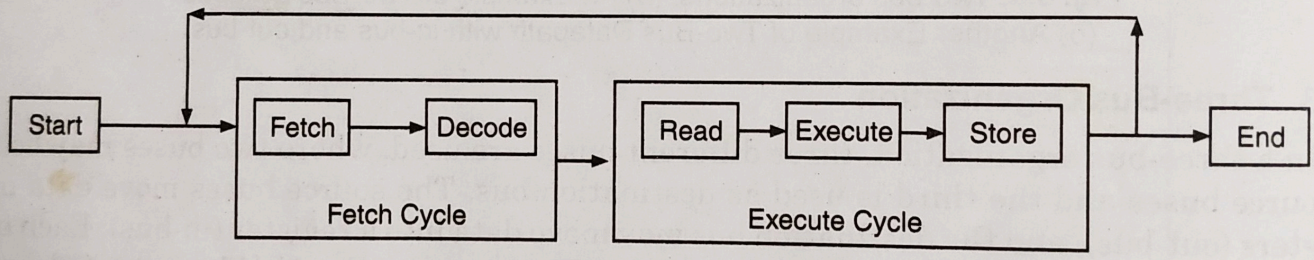

Fig. 6.5. Instruction cycle

6.3.1. Fetch Cycle

The first step the CPU carries out is to fetch some data and instructions (program) from main memory then store them in its own internal temporary memory areas. These memory areas are called 'registers'. This is called the 'fetch' part of the cycle. For this to happen, the CPU makes use of a vital hardware path called the 'address bus'.

The CPU places the address of the next item to be fetched on to the address bus. Data from this address then moves from main memory into the CPU by travelling the data bus.

The next step is for the CPU to make sense of the instruction it has just fetched. This process is called 'decode'.

The CPU is designed to understand a specific set of commands. These are called the 'instruction set' of the CPU. Each make of CPU has a different instruction set. The CPU decodes the instruction and prepares various areas within the chip in readiness of the next step.

The sequence of events in fetching an instruction can be summarized as follows:

1. The contents of the PC are loaded into the register MAR.
2. The value in the PC is incremented by one. (This operation can be done in parallel with a memory access.)

3. As a result of a memory read operation, the instruction is loaded into the register MDR.
4. The contents of the MDR are loaded into the register IR.

6.3.2. Execute Cycle

This is the part of the cycle when data processing actually takes place. The instruction is carried out upon the data (executed). The result of this processing is stored in yet another register. Once the execute stage is complete, the CPU sets itself up to begin another cycle once more.

Small Concept

Execute cycle includes reads contents of register, execute instruction and store results back to register.

Let's take an example of instruction

Add R1, R2, R3

The above instruction adds the contents of source registers R1 and R2, and stores the results in destination register R3. This addition can be executed as follows:

1. The registers R3, R1, R2, are extracted from the IR.
2. The contents of R1 and R2 are passed to the ALU for addition.
3. The output of the ALU is transferred to R3.

6.4. CONTROL UNIT

To execute an instruction, the control unit of the CPU must generate the required control signal in the proper sequence. As for example, during the fetch phase, CPU has to generate PCout signal along with other required signal in the first clock pulse. In the second clock pulse CPU has to generate PCin signal along with other required signals. So, during fetch phase, the proper sequence for generating the signal to retrieve from and store to PC is PC_{out} and PC_{in}.

To generate the control signal in proper sequence, a wide variety of techniques exist. Most of these techniques, however, fall into one of the two categories,

1. Hardwired Control Unit
2. Microprogrammed Control Unit.

6.4.1. Hardwired Control Unit

In this hardwired control techniques, the control signals are generated by means of hardwired circuit. The main objective of control unit is to generate the control signal in proper sequence.

Small Concept

Hardwired control unit is designed by hardware components such as logic gates, flip-flops, decoders or other digital circuits.

Consider the sequence of control signal required to execute the ADD instruction that is explained in previous lecture. It is obvious that eight non-overlapping time slots are required for proper execution of the instruction represented by this sequence.

Each time slot must be at least long enough for the function specified in the corresponding step to be completed. Since, the control unit is implemented by hardwire device and every device is having a propagation delay, due to which it requires some time to get the stable output signal at the output port after giving the input signal. So, to find out the time slot is a complicated design task.

For the moment, for simplicity, let us assume that all slots are equal in time duration. Therefore the required controller may be implemented based upon the use of a counter driven by a clock.

Each state, or count, of this counter corresponds to one of the steps of the control sequence of the instructions of the CPU.

Basic Features of Control Unit :

- The control logic is implemented with gates, Flip-Flops, Decoders, and other Digital Circuits
- Its provides fast operation due to direct connection.
- Big disadvantage is the change of wiring if the design has to be modified.

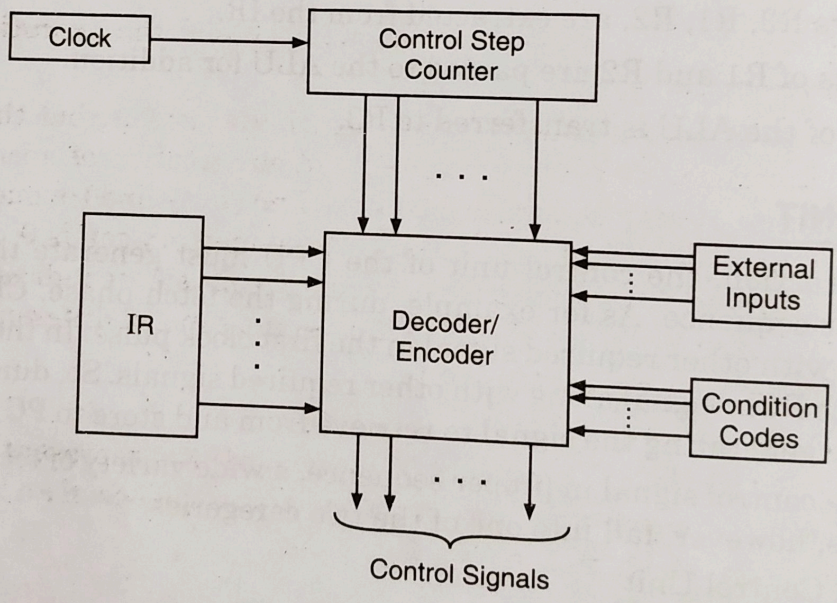

Fig. 6.6. Organization of control unit

The structure of control unit can be represented in a simplified view by putting it in block diagram. The detailed hardware involved may be explored step by step. The simplified view of the control unit is given in the Fig. 6.6.

The decoder/encoder block is simply a combinational circuit that generates the required control outputs depending on the state of all its input.

The decoder part of decoder/encoder part provide a separate signal line for each control step, or time slot in the control sequence. Similarly, the output of the instructor decoder consists of a separate line for each machine instruction loaded in the IR, one of the output line INSm is set to 1 and all other lines are set to 0.

6.4.2. Microprogrammed Control Unit

In hardwired control, we saw how all the control signals required inside the CPU. There is an alternative approach by which the control signals required inside the CPU can be generated. This alternative approach is known as microprogrammed control unit.

In microprogrammed control unit, the logic of the control unit is specified by a microprogram. A microprogram consists of a sequence of instructions in a microprogramming language. These are instructions that specify microoperations.

Small Concept

Micro-programmed control unit designed by software programs instead of hardware components using micro-instruction.

A microprogrammed control unit is a relatively simple logic circuit that is capable of

1. sequencing through microinstructions and
2. generating control signals to execute each microinstruction.

The concept of microprogram is similar to computer program. In computer program the complete instructions of the program is stored in main memory and during execution it fetches the instructions from main memory one after another. The sequence of instruction fetch is controlled by program counter (PC) .

Microprogram are stored in microprogram memory and the execution is controlled by microprogram counter (μPC).

Microprogram consists of microinstructions which are nothing but the strings of 0's and 1's. In a particular instance, we read the contents of one location of microprogram memory, which is nothing but a microinstruction. Each output line (data line) of microprogram memory corresponds to one control signal. If the contents of the memory cell is 0, it indicates that the signal is not generated and if the contents of memory cell is 1, it indicates to generate that control signal at that instant of time. First let discuss the some of different terminologies that are related to microprogrammed control unit.

Control Word (CW)

Control word is defined as a word whose individual bits represent the various control signal. Therefore each of the control steps in the control sequence of an instruction defines a unique combination of 0s and 1s in the CW.

Microinstructions & Microprogram

A sequence of control words (CWs) corresponding to the control sequence of a machine instruction constitutes the microprogram for that instruction. The individual control words in this microprogram are referred to as microinstructions.

Microprogram Memory

The microprograms corresponding to the instruction set of a computer are stored in a special memory which will be referred to as the microprogram memory. The control words related to an instructions are stored in microprogram memory also referred as control memory.

Microprogram Counter (μPC)

The control unit can generate the control signals for any instruction by sequentially readi the CWs of the corresponding microprogram from the microprogram memory. To read t

control word sequentially from the microprogram memory a microprogram counter (μPC) is needed. The basic organization of a microprogrammed control unit is shown in the Fig. 6.7.

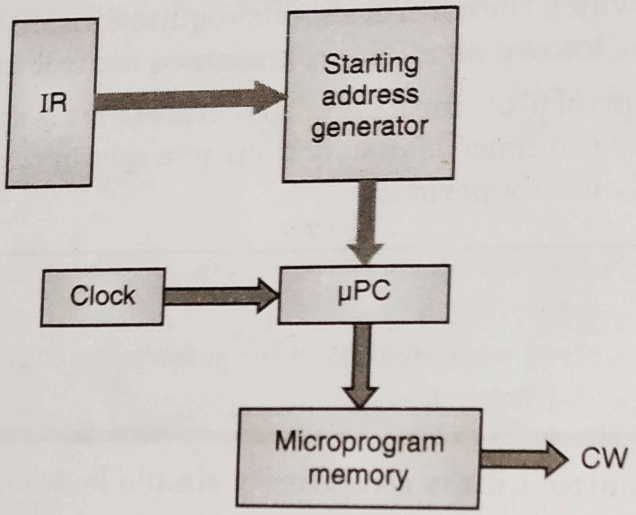

Fig. 6.7. Basic organization of a microprogrammed control Unit

The "starting address generator" block is responsible for loading the starting address of the microprogram into the PC every time a new instruction is loaded in the IR. The μPC is then automatically incremented by the clock, and it reads the successive microinstruction from memory.

Each microinstruction basically provides the required control signal at that time step. The microprogram counter ensures that the control signal will be delivered to the various parts of the CPU in correct sequence.

We have some instructions whose execution depends on the status of condition codes and status flag, as for example, the branch instruction. During branch instruction execution, it is required to take the decision between the alternative action. To handle such type of instructions with microprogrammed control, the design of control unit is based on the concept of conditional branching in the microprogram. For that it is required to include some conditional branch microinstructions.

Small Concept

- **Micro-operations** are the arithmetic logic operations on the registers.
- **Micro instruction** is the set of micro-operations.
- **Micro program** is the set of micro-instructions.

In conditional microinstructions, it is required to specify the address of the microprogram memory to which the control must direct. It is known as branch address. Apart from branch address, these microinstructions can specify which of the states flags, condition codes, or possibly, bits of the instruction register should be checked as a condition for branching to take place.

To support microprogram branching, the organization of control unit should be modified to accommodate the branching decision. To generate the branch address, it is required to know the status of the condition codes and status flag. To generate the starting address, we need the instruction which is present in IR. But for branch address generation we have to check the content of condition codes and status flag.

The organization of control unit to enable conditional branching in the microprogram is shown in the Fig. 6.8.

In the previous discussion, to design a micro programmed control unit, we have to do the following:

- For each instruction of the CPU, we have to write a microprogram to generate the control signal. The microprograms are stored in microprogram memory (control store). The starting address of each microprogram are known to the designer
- Each microprogram is the sequence of microintructions. And these microinstructions are executed in sequence. The execution sequence is maintained by microprogram counter.
- Each microinstructions are nothing but the combination of 0's and 1's which is known as control word. Each position of control word specifies a particular control signal. 0 on the control word means that a low signal value is generated for that control signal at that particular instant of time, similarly 1 indicates a high signal.
- Since each machine instruction is executed by a corresponding micro routine, it follows that a starting address for the micro routine must be specified as a function of the contents of the instruction register (IR).
- To incorporate the branching instruction, i.e., the branching within the microprogram, a branch address generator unit must be included. Both unconditional and conditional branching can be achieved with the help of microprogram. To incorporate the conditional branching instruction, it is required to check the contents of condition code and status flag.

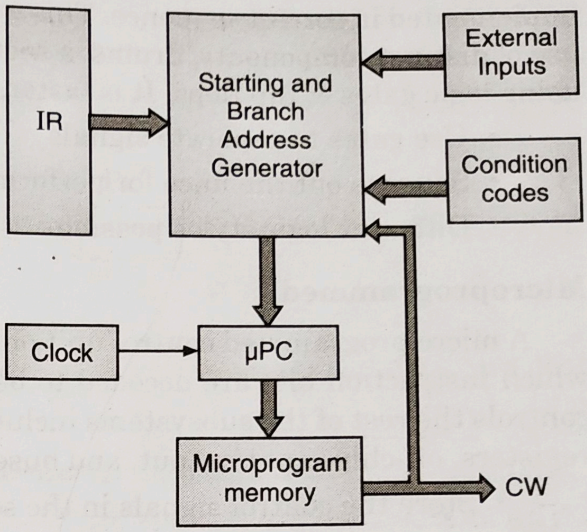

Fig. 6.8. Organization of microprogrammed control with conditional branching

Advantages and Disadvantages

The principal advantage of the use of microprogramming to implement a control unit is that it simplifies the design of the control unit. Thus, it is both cheaper and less error prone to implement. A hardwired control unit must contain complex logic for sequencing through the many micro-operations of the instruction cycle. On the other hand, the decoders and sequencing logic unit of a microprogrammed control unit are very simple pieces of logic.

Small Concept

Micro program reduce the hardware complexity as well as size of hardware requirements.

The principal disadvantage of a microprogrammed unit is that it will be somewhat slower than a hardwired unit of comparable technology. Despite this, microprogramming is the dominant technique for implementing control units in pure CISC architectures, due to its ease of implementation. RISC processors, with their simpler instruction format, typically use hardwired control units.

6.4.3. Hardwired Vs Microprogrammed

Hardwired

A hardwired control unit has a processor that generates signals or instructions to be implemented in correct sequence. This was the older method of control that works through the use of distinct components, drums, a sequential circuit design, or flip chips. It is implemented using logic gates & flip flops. It is faster, less flexible & limited in complexity

- Use gates to generate signals
- Squeeze out the juice for performance
- Different logic styles possible

Microprogrammed

A micro programmed control unit on the other hand makes use of a micro sequencer from which instruction bits are decoded to be implemented. It acts as the device supervisor that controls the rest of the subsystems including arithmetic and logic units, registers, instruction registers, off-chip input/output, and buses.

- Store the control signals in the sequence
- Just read from the memory every clock cycle
- It is slower, more flexible & greater complexity

POINTS TO REMEMBER

- This chapter introduced the basic functions of processor. The CPU (Central Processing Unit) or Processor is responsible for fetching, decoding and executing an instruction.
- A CPU fetches program instructions from RAM (input), interprets and processes it (execution) and then sends back the computed results so that the relevant components can carry out the instructions.
- There are four important functions of CPU,
  - (i) **Fetch:** the process to read the instruction from memory
  - (ii) **Decode:** the process to interpret the instruction
  - (iii) **Execute:** the process to perform the required arithmetic and logical operation.
  - (iv) **Write back:** the process to transfer the result of an execution of instruction.
- There are three basic components of the CPU:
  - (i) Arithmetic Logic Unit
  - (ii) Control Unit
  - (iii) Register Set (Memory Unit)
- ALU executes all arithmetic and logical operations calculations such as addition, subtraction, multiplication and division as well as comparisons.
- The Control Unit uses electrical signals to instruct the whole computer system for carrying out or executing, already stored program instructions.
- Registers are temporary storage areas which are responsible for holding the data that is to be processed. They store the instructions and data in a processor.
- General-purpose registers can be used for multiple purposes and assigned to a variety functions by the programmer.
- Special-purpose registers are restricted to only specific functions.