UNIT: 4 CPU Design

4.1. Overview of Generic CPU.

A generic CPU refers to a standard or typically central processing unit that is not specific to any particular brand or model. It represents a basic understanding of a CPU's architecture, capabilities and functionalities without considering the specific details of any particular CPU design.

Here are some key characteristics of a generic CPU:

1. Architecture : CPUs can be based on various architectures, such as X86, ARM, MIPS, PowerPc etc. Each architecture has its own instruction set and design principles.

2. Cores : Modern CPUs typically feature multiple cores, which are independent processing units capable of executing instructions simultaneously.

3. Clock Speed : The clock speed of a CPU determines the number of instructions it can execute per second. It is measured in gigahertz (GH_z) and generally correlates with the CPU's processing power.

4. Cache : CPUs incorporate cache memory to store frequently accessed data, reducing the time required to fetch data from the main memory.

5. Instruction set : CPUs support a specific instruction set architecture (ISA) that defines the set of instructions it can execute.

6. Pipelining : CPUs employ instruction pipelining to optimize instructions execution. Pipelining allows multiple instructions to be in various stages of completion simultaneously, improving overall efficiency.

7. Power efficiency : CPUs aim to balance performance with power consumption. Advancements in power management techniques, such as dynamic frequency scaling and low-power idle states help optimize energy efficiency.

8. word size : The word size refers to the number of bits processed in a single instruction or data. A simple CPU might have word size of 8 bits or 16 bits.

9. floating point unit (FPU) : Many CPUs include a dedicated FPU or math coprocessor that accelerates floating point arithmetic operations, crucial for task like scientific calculations, simulations.

4.2. Design and Implementation of very simple CPU.

The part of the computer that does the actual processing operations and computations is called CPU. It's main purpose is to interpret the instructions received from memory and peform arithmetic and logical control operations with data stored in internal registers or from I/O units.

A CPU contains following components:

- → ALU
- → Registers
- → Memory
- → I/O devices
- → control unit
- → clock and
- → Buses.

figure below shows the abstract view of CPU organization along with major components.

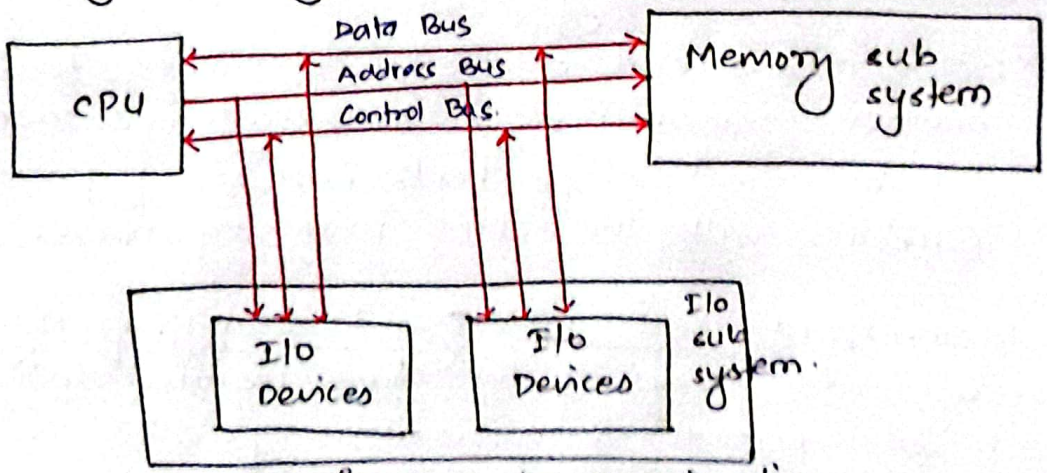


fig: Generic computer organization.

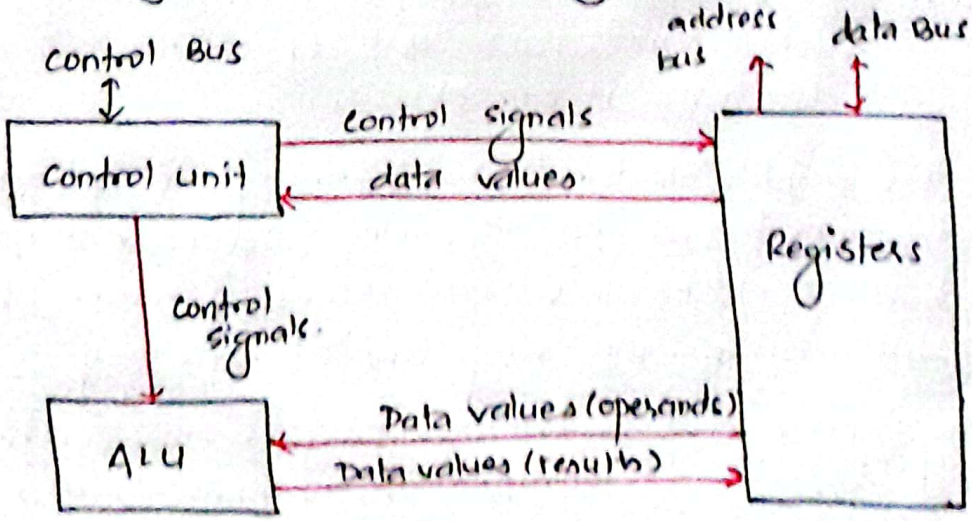


fig: CPU Internal organization.

→ control unit controls the oper of CPU, memory and I/O. It is responsible for generation of control signal by decoding the instruction. It is the core control unit of CPU.

→ The ALU performs arithmetic and logical operations as per the control signals.

→ The registers are small and fast internal memory that are used for temporary storage of data during the execution of an instruction.

→ The Buses provide communication pathways for interaction of different components within the CPU.

4.3. Specification of very simple CPU.

sqn: Explained in 4.1 already.

4.4. fetch, Decode and Execute of Instruction.

fetch - decode - execute cycle is a fundamental process performed by a CPU to execute instructions. Here's a high level overview of each step in this cycle:

1. **fetch:** In this step, the CPU retrives the next instruction from memory. The program counter (PC) holds the memory address of next instruction to be fetched. The CPU fetches the instruction from that address, increments the program counter to point to the next instruction, and stores the fetched instruction in the instruction register (IR).

2. **Decode:** Once the instruction is fetched and stored in the instruction register, the CPU decodes the instruction to determine the operation to be performed and the operands involved. The instruction decoder interprets the binary representation of the instruction and identifies the opcode and any additional fields that specify the operands or addressing modes.

3. **Execute**: After the instruction is decoded, the CPU performs the CPU performs the specified operation. The execution phase varies depending on the type of instruction and the CPU architecture. It may involve arithmetic or logical operations, data transfers between registers or memory, branching to a different part of the program, or interaction with input/output devices.

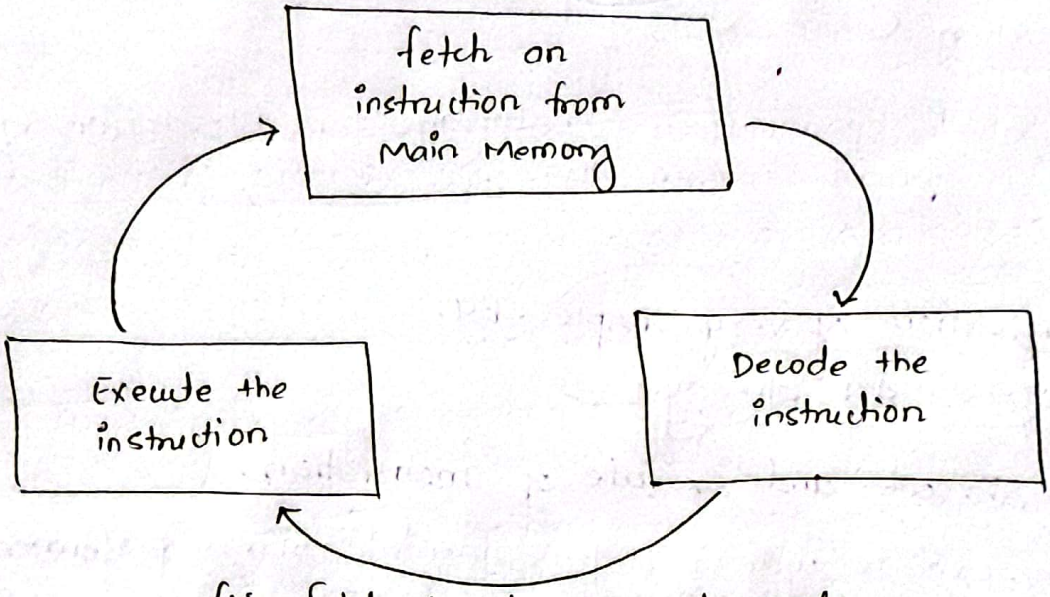
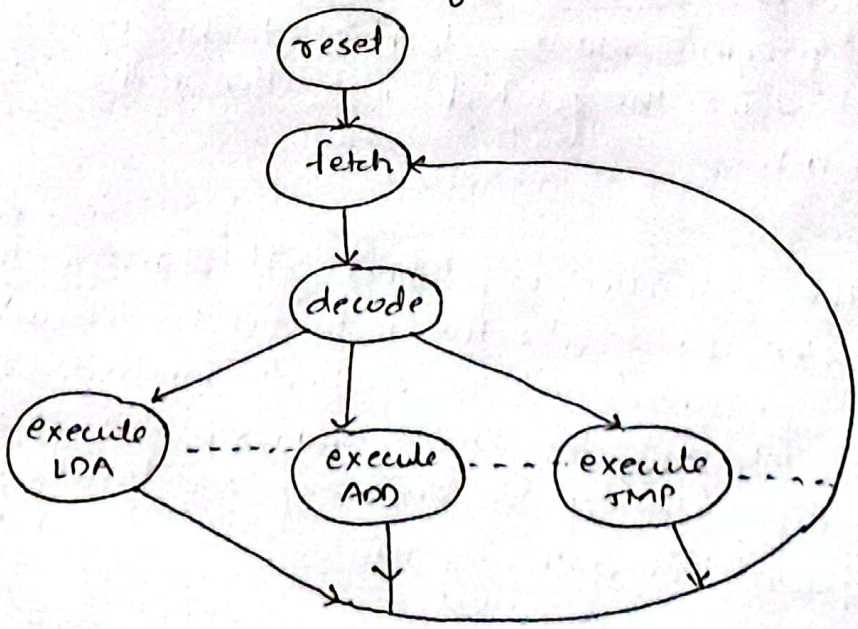


fig: fetch-decode-execute cycle.

4.5 Complete state diagram of very simple CPU.

A state diagram, also known as a state machine diagram, represents the different states that a system or component can be in and the transitions betn those states.

A simple state diagram of CPU is shown below.



The following fig shows the sequence of operations performed by the CPU during execution of an instruction.

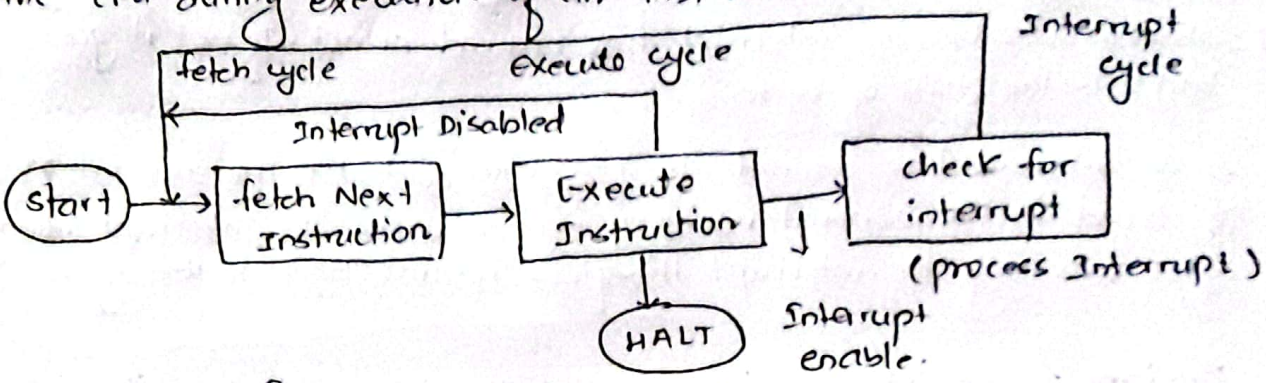


fig: Instruction cycle with interrupt.

The fig below shows a detailed instruction cycle in form of states that the cycle has through during execution of an instruction. It is generally termed as instruction cycle state diagram.

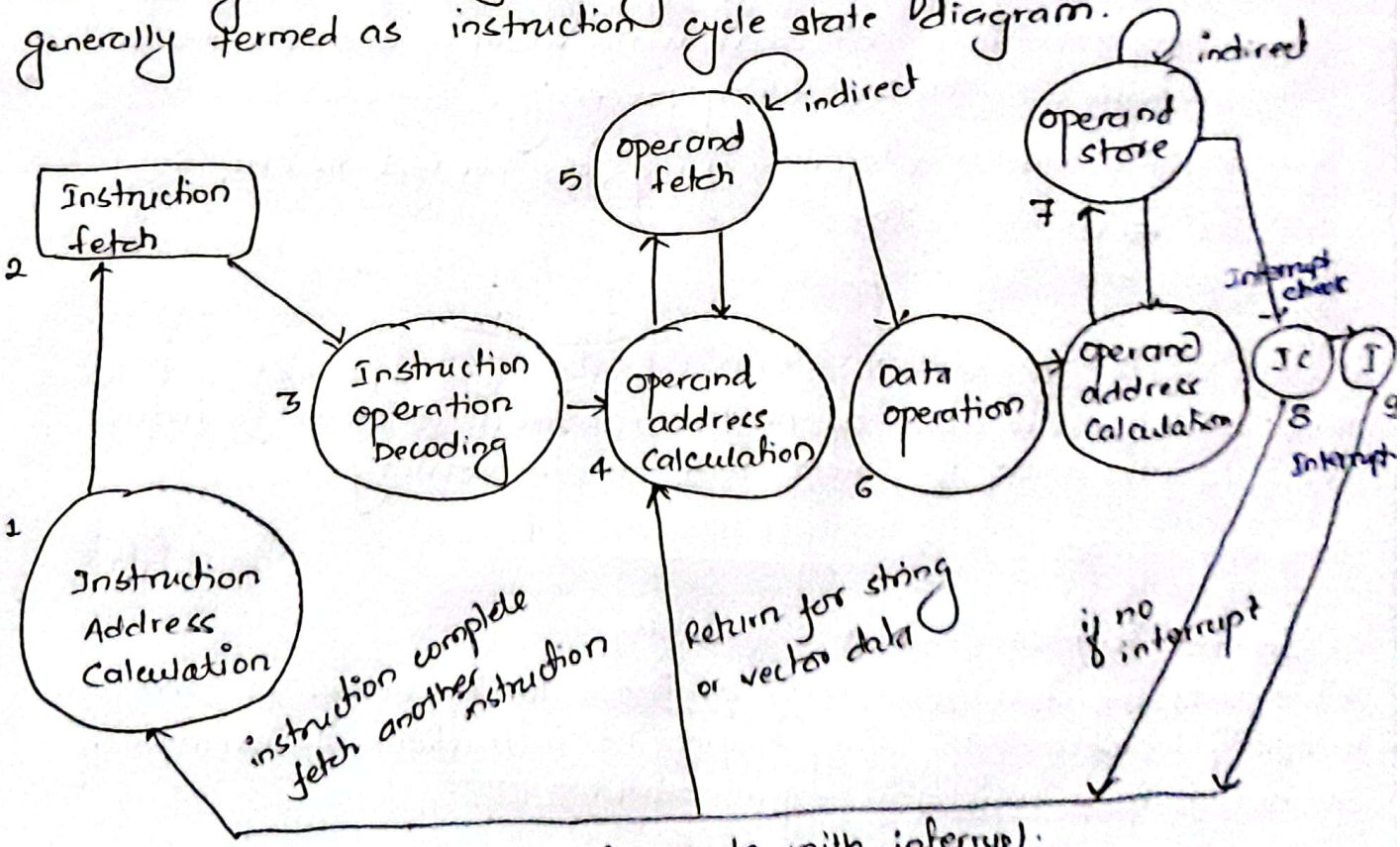


fig: State diagram of instruction cycle with interrupt.

1. IAC : Determines address of next instruction to be executed.
2. IF : Reads the instruction from memory of CPU.
3. IOD : Analyzes the instruction to determines the type of operation to be performed.
4. OAC : It determines whether from I/O or memory to fetch the operands required for the execution.
5. OF : fetches or reads the operand from I/O or memory.
6. DO : Performs operations on fetched operands as per the instruction

7. OS : stores the result into particular memory location specified in the instruction.

8. IC: checks whether an interrupt has occured or not. If not it goes backs to the flow of cycle.

9. I: If an interrupt has occured, the program control to the ISR which is stored in the central memory and services the interrupt. finally after servicing the interrupt the flow returns back to the cycle.

4.6. Design of Register Section and Arithmetic Logical Unit.

Register Organization:

Registers are the fastest form of memory available in a computer system. Registers are placed above main memory and resides within the processor and are aid during processing.

Generally there are 2 categories of registers involved in CPU:

i) User visible registers
ii) control and status registers.

i) User Visible Registers:

These registers are used by the programmers in order to reduce the interaction of CPU with main memory.

They are categorized as:

a) General Purpose Registers: ~~These by~~.

This type of registers are freely available for programmer and can be easily accessed by using specified instructions. An example of general purpose register is Accumulator.

b) Data Registers: They cannot be used as part of instruction in a program. They are only meant for storage of data and cannot be used for address calculations.

c) Address Registers: They are used to hold the address of specific memory depending upon the addressing mode.

- segment pointers (points the base of the segment)
- Index register
- stack pointer. (Holds the address of the top of the stack)

d) Condition codes: These registers are similar to flags. They are set or reset as per the result of arithmetic or logical operations performed by the ALU.

ii) Control and status Registers:

Control registers are employed to the control the operation of the processor. These registers are not accessible for the users.

some control registers are:

a) Program counter (PC) : It holds the address of next instruction to be executed.

b) Instruction Register (IR) : It holds the instruction being currently executed.

c) Memory Address Registers (MAR) : Holds the address of memory location data is to be read or written to.

d) Memory Buffer Register (MBR) : Holds the actual data to be written to memory or sent to I/O.

status registers are used to reflect the status of the arithmetic operations. They are also called flags.

- sign
- carry
- zero
- Equal
- overflow
- interrupt enable | disable.

The register organization of 8086 µp is shown below.

General Registers

AX	Accumulator
BX	Base
CX	Count
DX	Data

pointers and index

SP	Stack pointer
BP	Base pointer
SI	Source index
DI	Destination index

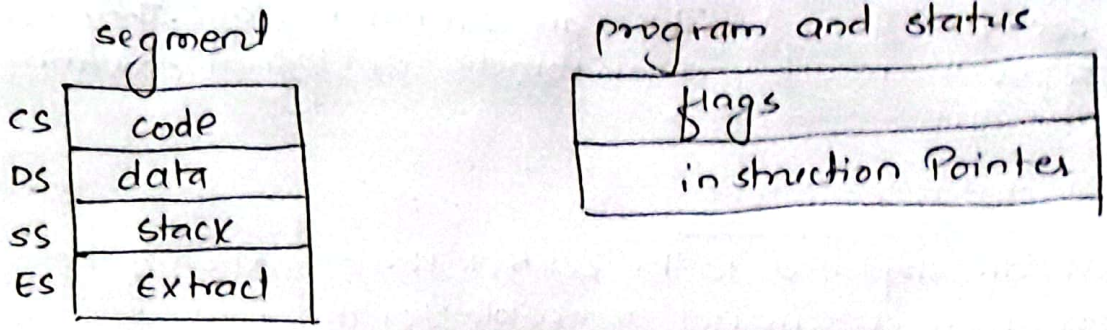


fig: register organization of 8086 µp.

imp 4.7. Design of Hardwired control unit.

In this approach control signals are defined by control functions, expressed in terms of control input line status of flags, control of IR, control signal from Bus etc. The control signal/function is defined over control input and is implemented using logic gates.

This approach is fully focused in hardware. The control unit is viewed as interconnection of sequential logic circuits implemented using decoders and counters. They are very fast in operation and useful in high speed computers system design. This implementation is favoured in RISC architecture.

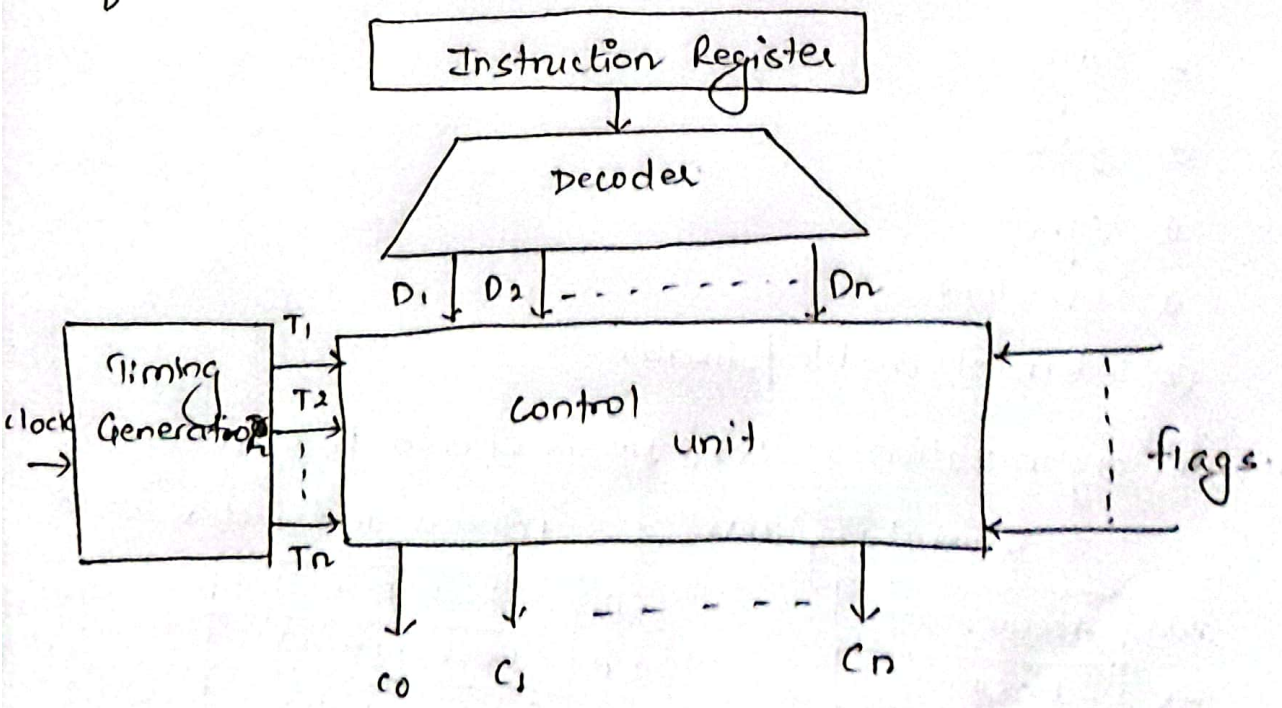


fig: Hardware implementation of cu

4.8. Control signal Generation:

In computer architecture, control signal generation involves generating signals that control the operation of various components within a computer system, such as the CPU, memory, input/output devices and other peripherals.

These control signals co-ordinate the flow of data and instructions, synchronize operations and enable proper functioning of the system.

The block diagram ~~shows~~ below shows the control signal generation.

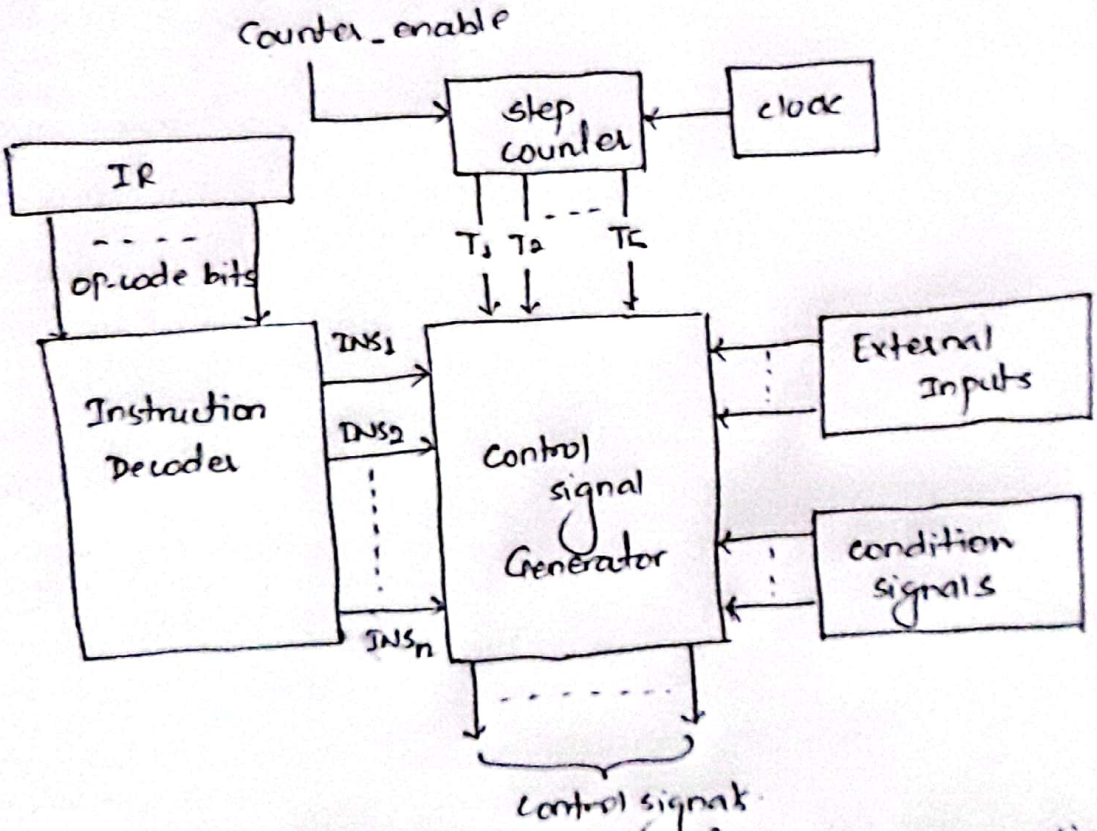


fig: Block diagram that shows the generation of control signals.

→ The above block diagram shows the hardware generation of control signals.

→ step counter keeps track of control steps. When step counter gets the clock pulses, it generates one control step.