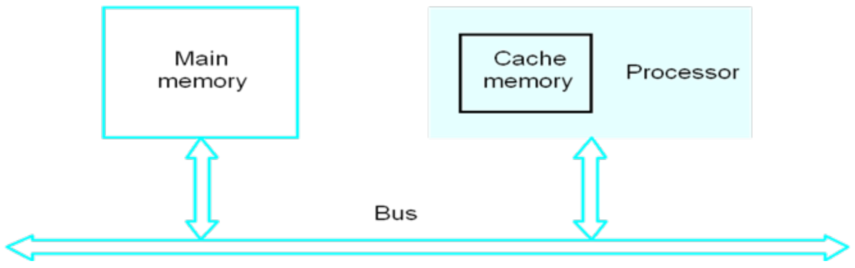
Q2 (a) How do various factors like Hardware design, Instruction set, Compiler related to the performance of a computer?

Answer

The most important measure of a computer is how quickly it can execute programs.

- Three factors affect performance:
- Hardware design
- Instruction set
- Compiler

Processor time to execute a program depends on the hardware involved in the execution of individual machine instructions.



- The processor and a relatively small cache memory can be fabricated on a single integrated circuit chip.
- Speed
- Cost
- Memory management

Processor Clock

- Clock, clock cycle, and clock rate
- The execution of each instruction is divided into several steps, each of which completes in one clock cycle.
- Hertz – cycles per second

Basic Performance Equation

- T – processor time required to execute a program that has been prepared in high-level language
- N – number of actual machine language instructions needed to complete the execution (note: loop)
- S – average number of basic steps needed to execute one machine instruction. Each step completes in one clock cycle
- R – clock rate
- Note: these are not independent to each other

$$T = \frac{N \times S}{R}$$

Pipeline and Superscalar Operation

- Instructions are not necessarily executed one after another.
- The value of S doesn't have to be the number of clock cycles to execute one instruction.
- Pipelining – overlapping the execution of successive instructions.
- Add R1, R2, R3
- Superscalar operation – multiple instruction pipelines are implemented in the processor.
- Goal – reduce S (could become <1!)

Clock Rate

- Increase clock rate
- Improve the integrated-circuit (IC) technology to make the circuits faster
- Reduce the amount of processing done in one basic step (however, this may increase the number of basic steps needed)
    - Increases in R that are entirely caused by improvements in IC technology affect all aspects of the processor's operation equally except the time to access the main memory.

CISC and RISC

- Tradeoff between N and S
- A key consideration is the use of pipelining
- S is close to 1 even though the number of basic steps per instruction may be considerably larger
- It is much easier to implement efficient pipelining in processor with simple instruction sets
- Reduced Instruction Set Computers (RISC)
- Complex Instruction Set Computers (CISC)

Compiler

- A compiler translates a high-level language program into a sequence of machine instructions.
- To reduce N, we need a suitable machine instruction set and a compiler that makes good use of it.
- Goal – reduce N×S
    - A compiler may not be designed for a specific processor; however, a high-quality compiler is usually designed for, and with, a specific processor

Performance Measurement

- T is difficult to compute.
- Measure computer performance using benchmark programs.
- System Performance Evaluation Corporation (SPEC) selects and publishes representative application programs for different application domains, together with test results for many commercially available computers.
- Compile and run (no simulation)

Reference computer

Q2 (b) What are the various Instruction categories?

Answer

Arithmetic Instructions
    Ex.: Add, Sub, Mul etc.
Logical instructions
    Instructions doing comparison operations.
Program Control instructions
    Ex.: Jump to some memory location where the code is place & return etc.
I/O instructions
    Ex.: In, Out
Data Transfer instructions
    Register-Memory / Memory-Register
    Register-Register
    Memory-Memory.

Q3 (a) Consider the symbolic instruction ADD 3000, 5000. Memory locations 3000 and 5000 contain the operands. Suppose the program is stored from the locations 2000. Explain the fetch-decode-execute cycle of this instruction, showing clearly the roles of special purpose registers, Control Unit and ALU. Data is stored in memory locations 3000 and 5000 show how this is fetched and decoded.

Answer

Fetch Phase

The CU is responsible for the execution of this phase.
1.The contents of PC are transferred to Memory MAR
2.The content of the Main Memory as pointed by the MAR is accessed through the Address Bus
3.The current instruction is fetched into MBR. These contents flow through the DataBus
4.The instruction is then transferred from MBR to IR

Decode Phase

The Instruction decoder unit in the CU is responsible for the execution of this phase
5.The opcode part of the instruction is transferred to the Instruction Decoder
6.The Instruction Decoder then decodes the opcode part (ADD), interprets and
understands what to do (In this case the Adder in ALU is invoked)
7.The PC is then incremented to point to the next instruction

Q3 (b) A program begins from memory location *4000H*. An instruction *MOV 600AH, R1* is stored in the memory starting from location *4020H.* Assume that the whole program is shifted to a locations starting from *6000H*. The memory location *600AH*, which contained the data to be moved to register *R1* is over written by the content of the program. Suggest solutions to overcome this situation.

Answer
Problem of direct addressing: the change in the location of the program is associated with the change in all absolute memory references.

Solution is to represent the address of the data indirectly. There are two ways to do it:

1) **Register Indirect Addressing:** the address of the data is stored in a Register.

2) **Memory Indirect Addressing:** the address of the data is stored in another memory location

Q4 (a) What do you mean by bus arbitration? Differentiate between centralized and distributed arbitration.

Answer Page Number 237-239 of text book

Q4 (b) How the interrupt-service routine is different from that of a subroutine? Summarize the sequence of events involved in handling an interrupt request from a single device.

Answer Page Number 212 of text book

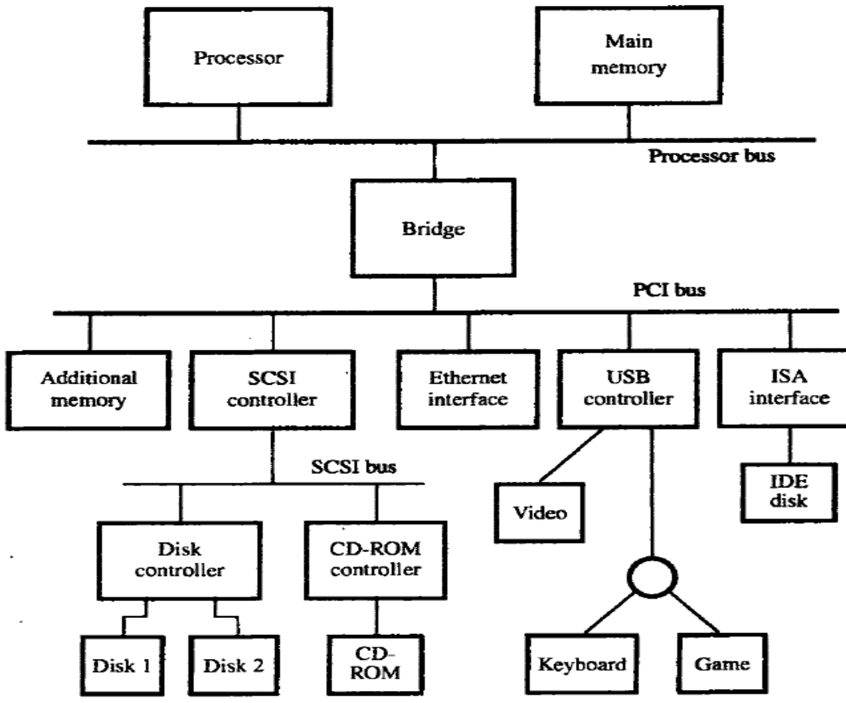
Q5 (a) What are the various functions of Input/output interface?

Answer

1. Provides a storage buffer for at least one word of data (or one byte, in the case of byte-oriented devices)
2. Contains status flags that can be accessed by the processor to determine whether the buffer is full (for input) or empty (for output)
3. Contains address-decoding circuitry to determine when it is being addressed by the processor
4. Generates the appropriate timing signals required by the bus control scheme
5. Performs any format conversion that may be necessary to transfer data between the bus and the I/O device, such as parallel-serial conversion in the case of a serial port

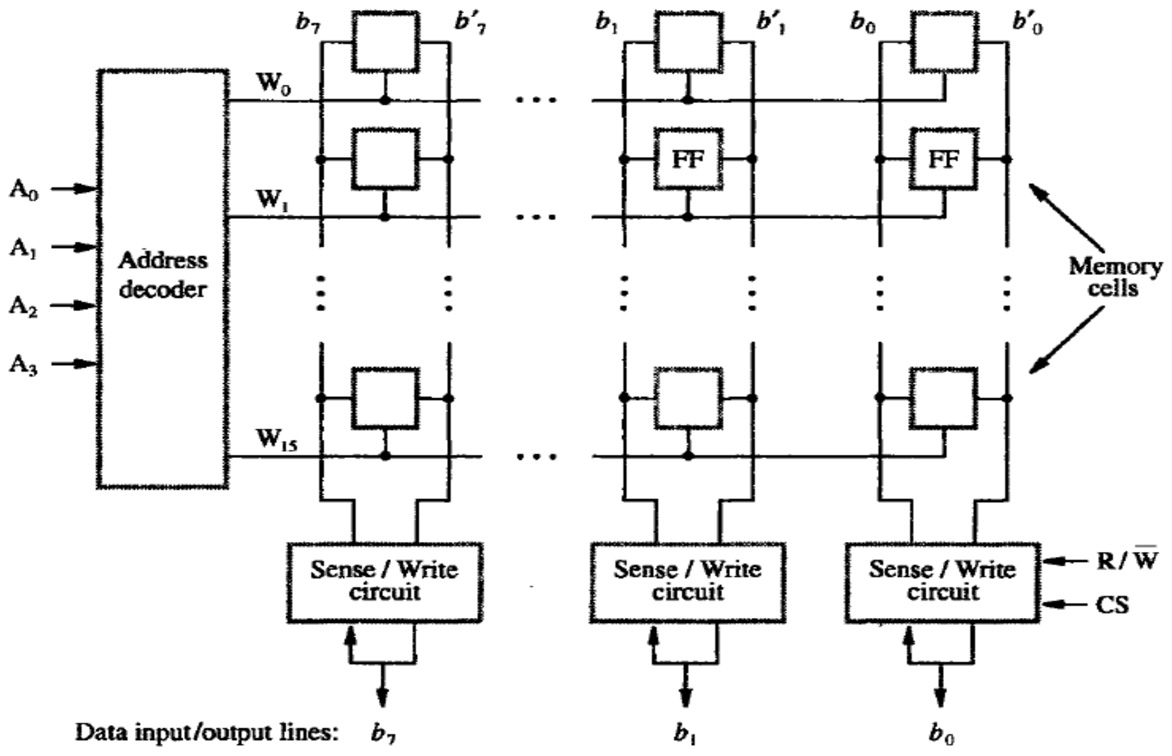
Q5 (b) With the help of a block diagram show a typical arrangement in which different interfaces are used to connect various peripheral devices.

Answer



Q6 (a) Explain with a neat sketch the internal organization of a memory chip consisting of 16 words of 8 bits each (16 x8 organization.

Answer

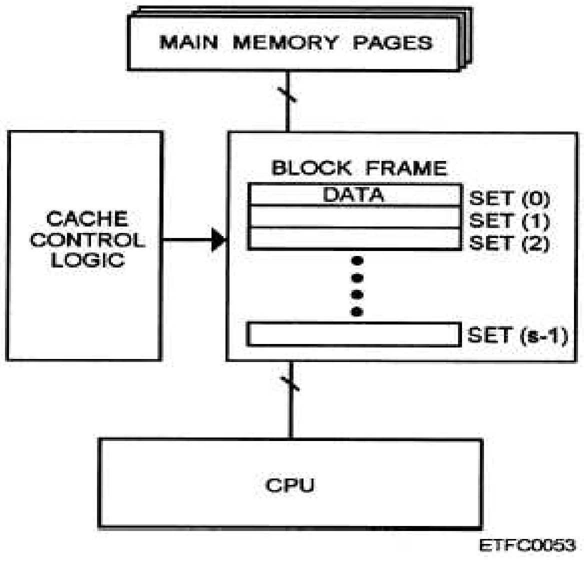


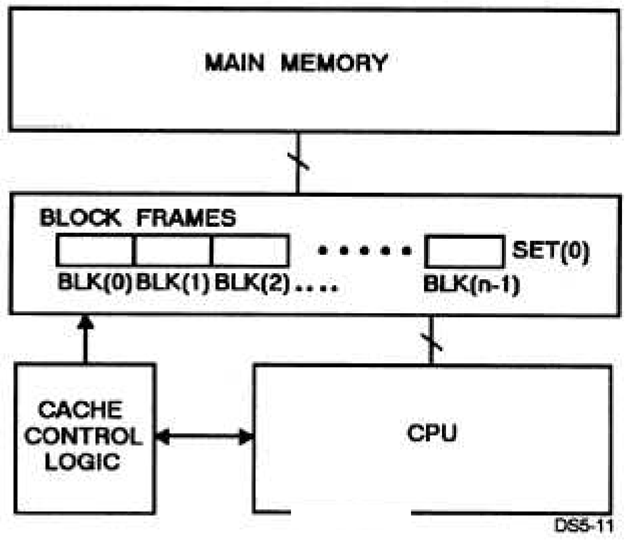
Memory cells are usually organized in the form of an array, in which each cell is capable of storing one bit of information. A possible organization is illustrated in Figure 5.2. Each row of cells constitutes a memory word, and all cells of a row are connected to a common line referred to as the *word line*, which is driven by the address decoder on the chip. The cells in each column are connected to a Sense/Write circuit by two *bit lines*. The Sense/Write circuits are connected to the data input/output lines of the chip. During a Read operation, these circuits sense, or read, the information stored in the cells selected by a word line and transmit this information to the output data lines. During a Write operation, the Sense/Write circuits receive input information and store it in the cells of the selected word.

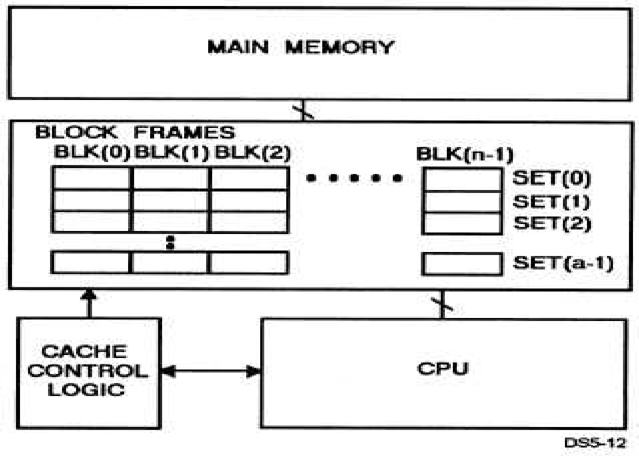
Figure 5.2 is an example of a very small memory chip consisting of 16 words of 8 bits each. This is referred to as a 16×8 organization. The data input and the data output of each Sense/Write circuit are connected to a single bidirectional data line that can be connected to the data bus of a computer. Two control lines, $R/\overline{W}$ and CS, are provided in addition to address and data lines. The $R/\overline{W}$ (Read/$\overline{\text{Write}}$) input specifies the required operation, and the CS (Chip Select) input selects a given chip in a multichip memory system. This will be discussed in Section 5.2.4.

Q6 (b) Outline the three cache mapping policies: direct mapping, associative mapping, and set associative mapping. Briefly discuss the advantages and disadvantages of each.

Answer
If the particular address is found in the cache, the block of data is sent to the CPU, and the CPU goes about its operation until it requires something else from memory. When the CPU finds what it needs in the cache, a hit has occurred. When the address requested by the CPU is not in the cache, a miss has occurred and the required address along with its block of data is brought into the cache according to how it is mapped. Cache processing in some computers is divided into two sections: **main** cache and **eavesdrop** cache. Main cache is initiated by the CPU within. Eavesdrop is done when a write to memory is performed by another requestor (other CPU or IOC). Eavesdrop searches have no impact on CPU performances. **CACHE**
MAPPING TECHNIQUES — Cache mapping is the method by which the contents of main memory are brought into the cache and referenced by the CPU. The mapping method used directly affects the performance of the entire computer system.
Direct mapping —Main memory locations can only be copied into one location in the cache. This is accomplished by dividing main memory into pages that correspond in size with the cache
Fully associative mapping —Fully associative cache mapping is the most complex, but it is most flexible with regards to where data can reside. A newly read block of main memory can be placed anywhere in a fully associative cache. If the cache is full, a replacement algorithm is used to determine which block in the cache gets replaced by the new data
Set associative mapping —Set associative cache mapping combines the best of direct and associative cache mapping techniques. As with a direct mapped cache, blocks of main memory data will still map into as specific set, but they can now be in any N-cache block frames within each set







Q7 (a) What is full adder and n-bit ripple-carry adder? Explain logic specification for a stage of binary addition.

Answer Page Number 368-369

Q7 (b) Explain virtual-memory address translation scheme with the help of a diagram.

Answer Page Number 339-340

Q8 (a) Explain Booths algorithm for multiplying two signed binary numbers.

Answer

Booth's multiplication algorithm **is a multiplication algorithm that multiplies two signed binary numbers in two's complement notation.**

If *x* is the count of bits of the multiplicand, and *y* is the count of bits of the multiplier :

- Draw a grid of three rows, each with columns for *x* + *y* + 1 bits. Label the lines respectively A (add), S (subtract), and P (product).
- In two's complement notation, fill the first *x* bits of each line with :
    - A: the multiplicand
    - S: the negative of the multiplicand (in 2's complement format)
    - P: zeroes
- Fill the next *y* bits of each line with :
    - A: zeroes
    - S: zeroes
    - P: the multiplier
- Fill the last bit of each line with a zero.

- Do both of these steps *y* times :
    1. If the last two bits in the product are...
        - 00 or 11: do nothing.
        - 01: P = P + A. Ignore any overflow.
        - 10: P = P + S. Ignore any overflow.
    2. Arithmetically shift the product right one position.

- Drop the first (we count from right to left when dealing with bits) bit from the product for the final result.

How it works

Consider a positive multiplier consisting of a block of 1s surrounded by 0s. For example, 00111110. The product is given by :

$$M \times {''}0\ 0\ 1\ 1\ 1\ 1\ 1\ 0{''} = M \times (2^5 + 2^4 + 2^3 + 2^2 + 2^1) = M \times 62$$

where M is the multiplicand. The number of operations can be reduced to two by rewriting the same as

$$M \times {''}0\ 1\ 0\ 0\ 0\ 0\text{-}1\ 0{''} = M \times (2^6 - 2^1) = M \times 62$$

In fact, it can be shown that any sequence of 1's in a binary number can be broken into the difference of two binary numbers:

$$(\ldots 0\overbrace{1\ldots1}^{n}0\ldots)_2 \equiv (\ldots 1\overbrace{0\ldots0}^{n}0\ldots)_2 - (\ldots 0\overbrace{0\ldots1}^{n}0\ldots)_2.$$

Hence, we can actually replace the multiplication by the string of ones in the original number by simpler operations, adding the multiplier, shifting the partial product thus formed by appropriate places, and then finally subtracting the multiplier. It is making use of the fact that we do not have to do anything but shift while we are dealing with 0s in a binary multiplier, and is similar to using the mathematical property that 99 = 100 - 1 while multiplying by 99.

This scheme can be extended to any number of blocks of 1s in a multiplier (including the case of single 1 in a block). Thus,

$$M \times {''}0\ 0\ 1\ 1\ 1\ 0\ 1\ 0{''} = M \times (2^5 + 2^4 + 2^3 + 2^1) = M \times 58$$
$$M \times {''}0\ 1\ 0\ 0\text{-}1\ 1\text{-}1\ 0{''} = M \times (2^6 - 2^3 + 2^2 - 2^1) = M \times 58$$

Booth's algorithm follows this scheme by performing an addition when it encounters the first digit of a block of ones (0 1) and a subtraction when it encounters the end of the block (1 0). This works for a negative multiplier as well. When the ones in a multiplier are grouped into long blocks, Booth's algorithm performs fewer additions and subtractions than the normal multiplication algorithm.

Q8 (b) Give algorithm for restoring division. Also draw a logic circuit that implements restoring division.

Answer Page Number 391of Text Book

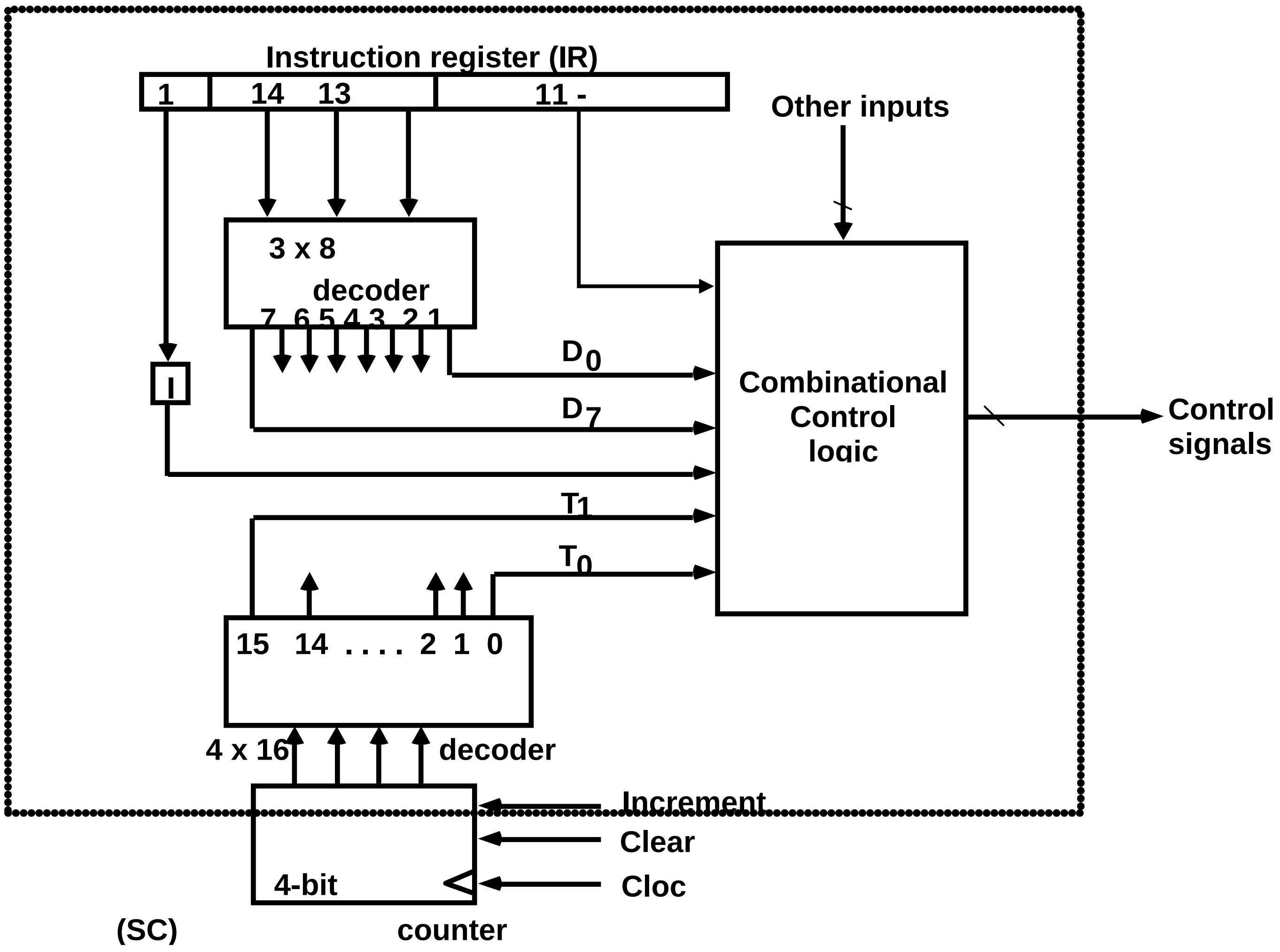
Q9 (a) Using suitable example, Explain how a word is fetched from memory and how it is stored in memory.

Answer Page Number 418-420 of Text Book.

Q.9.b. Draw the block diagram of a basic hardwired control organization with two decoders, a sequence counter and a number of control logic gates?

Answer
•Control unit (CU) of a processor translates from machine instructions to the control signals for the micro operations that implement them
•Control units are implemented in one of two ways
•***Hardwired*** **Control**
–CU is made up of sequential and combinational circuits to generate the control signals
•***Micro programmed*** **Control**
–A control memory on the processor contains micro programs that activate the necessary control signals

Hardwired Control



<u>**Textbook**</u>

Computer Organization,Carl Hamacher, Zvonko Vranesic, Safwat Zaky, 5th Edition, TMH, 2002.