9 Computer Organization and Computer Architecture

9.1 : Introduction

Q.1 Distinguish between computer architecture and computer organization with the help of appropriate examples.

Ans. : • **Computer architecture** refers to those attributes of a system visible to a programmer. In other words, we can also say that the computer architecture refers to the attributes that have a direct impact on the logical execution of the program.

- **Computer organisation** refers to the operational units and their interconnections that realize the architectural specifications.
- The **architectural attributes** include the instruction set, data types, number of bits used to represent data types, I/O mechanism, and techniques for addressing memory.
- The **organisational attributes** include those hardware details transparent to programmer, such as control signals, interfaces between the computer, memory and I/O peripherals.
- **For example,** it is an **architectural issue** whether a computer will have a multiply and division instructions. It is an **organisational issue** whether to implement multiplication and division by special unit or by a mechanism that makes repeated use of the add and subtract unit to perform multiplication and division, respectively.

9.2 : Functions and Types of Computer Units

Q.2 Explain functional units of computer.

☞ [SPPU : May-12, Dec.-12, Marks 6]

Ans. : • The computer consists of five functionally independent units :

- Input
- Memory
- Arithmetic and logic
- Output and
- Control units.

• Fig Q.2.1 shows the block diagram of a digital computer.

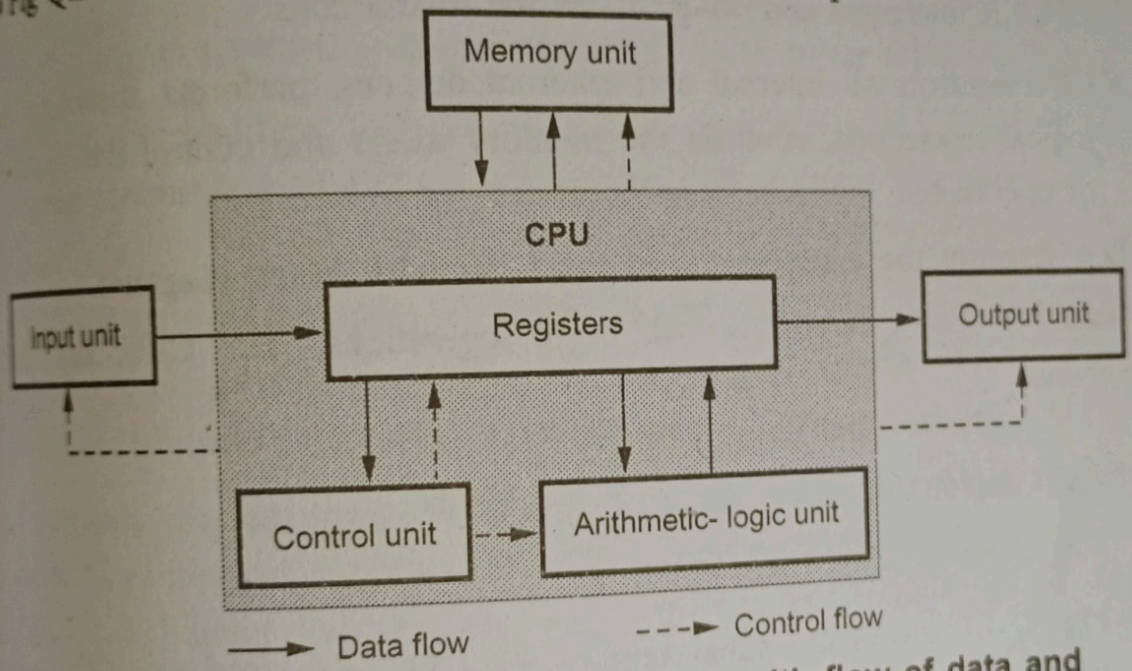


Fig. Q.2.1 Block diagram of a computer with flow of data and instructions

• It consists of input unit, control unit, Arithmetic-Logic Unit (ALU), registers, memory unit and output unit.

1. **Input unit :** It accepts instructions and data from the user and applies them as input to the computer.
2. **Central Processing Unit (CPU) :** As shown in Fig. Q.2.1, it consists of arithmetic logic unit. It processes and controls instructions and data inside the computer.

3. **Output unit :** After completing processing, this unit communicates results to the user.
4. **Memory unit :** The memory unit stores the data read from input device and provides it for processing when required. It also stores the intermediate and final results of processing.

9.3 : Central Processing Unit (CPU)

Q.3 What is CPU ?

Ans. : • The CPU is the brain of the computer system. It works as an administrator of a system.

- All the operations within the system are supervised and controlled by CPU. It interprets and co-ordinates the instructions.
- CPU controls all internal and external devices, performs arithmetic and logical operations, controls the memory usage and control the sequence of operations.

Q.4 Explain the components of the CPU with block diagram.

Ans. : • For performing all these operations, the CPU has three subunits :

- Arithmetic and Logic Unit (ALU)
- Control Unit
- Memory (CPU registers) Unit

- Fig. Q.4.1 shows the subsystem in the CPU and CPU interaction with other units.

ALU :

- It performs arithmetic operations like addition, subtraction and logic operations like OR, AND, invert, exclusive-OR on binary words. The data stored in memory unit is transferred to ALU. The ALU performs the operation, that is, the data is processed and the result is stored in internal memory unit of CPU.
- Arithmetic and logic operations performed by ALU sets flags to represent certain conditions such as equal to condition, zero condition,

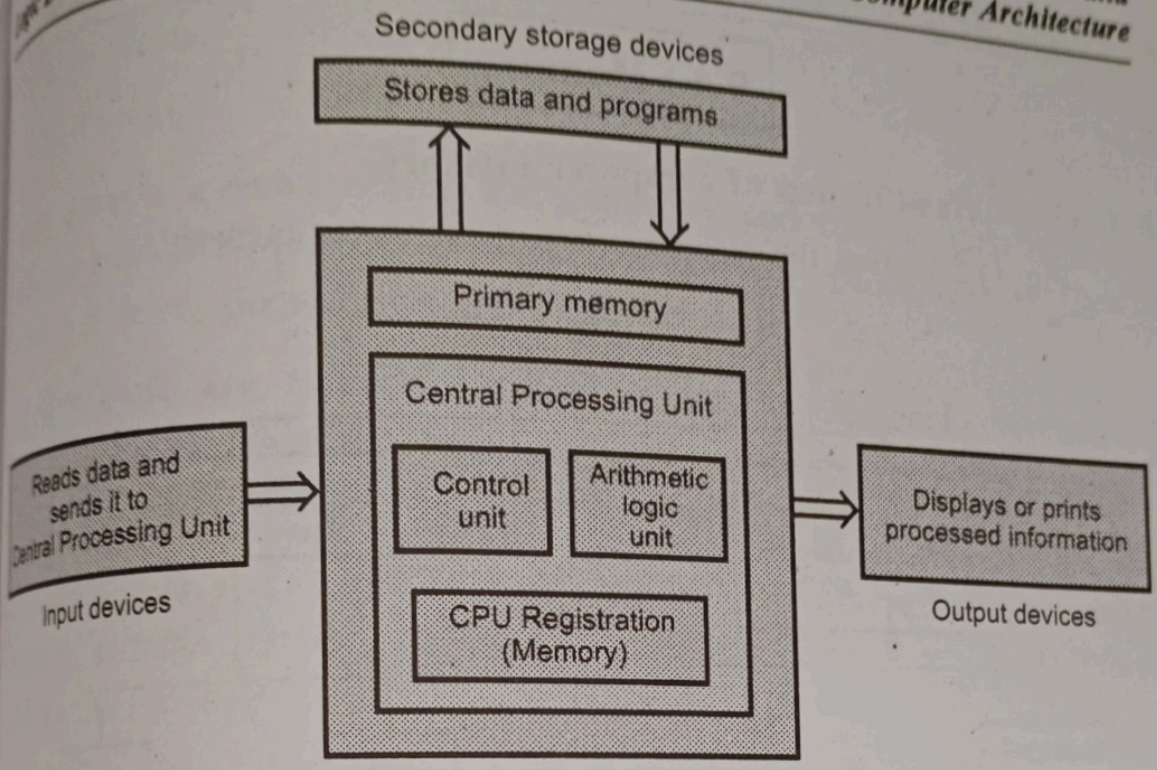


Fig. Q.4.1 CPU and its interaction with other units

greater than condition and so on. These conditions are checked by program instructions to change the sequence of program execution.

Control Unit

- It controls all the operations which internally take place within the CPU and also the operations of CPU related to input/output devices. The control unit directs the overall functioning of a computer system.
- It interprets program instructions and generates control signals to ensure correct execution of the program. The control signals generated by the control unit direct the overall functioning of the other units of the computer.

Memory unit

- It consists of two sections :
  - internal memory section
  - external memory section
- An internal memory section for storage of active data and instructions and an external memory section for long term storage.

9.4 : Memory

Q.5 Give the classification of computer memory.

Ans. : Fig. Q.5.1 gives the classification of computer memory.

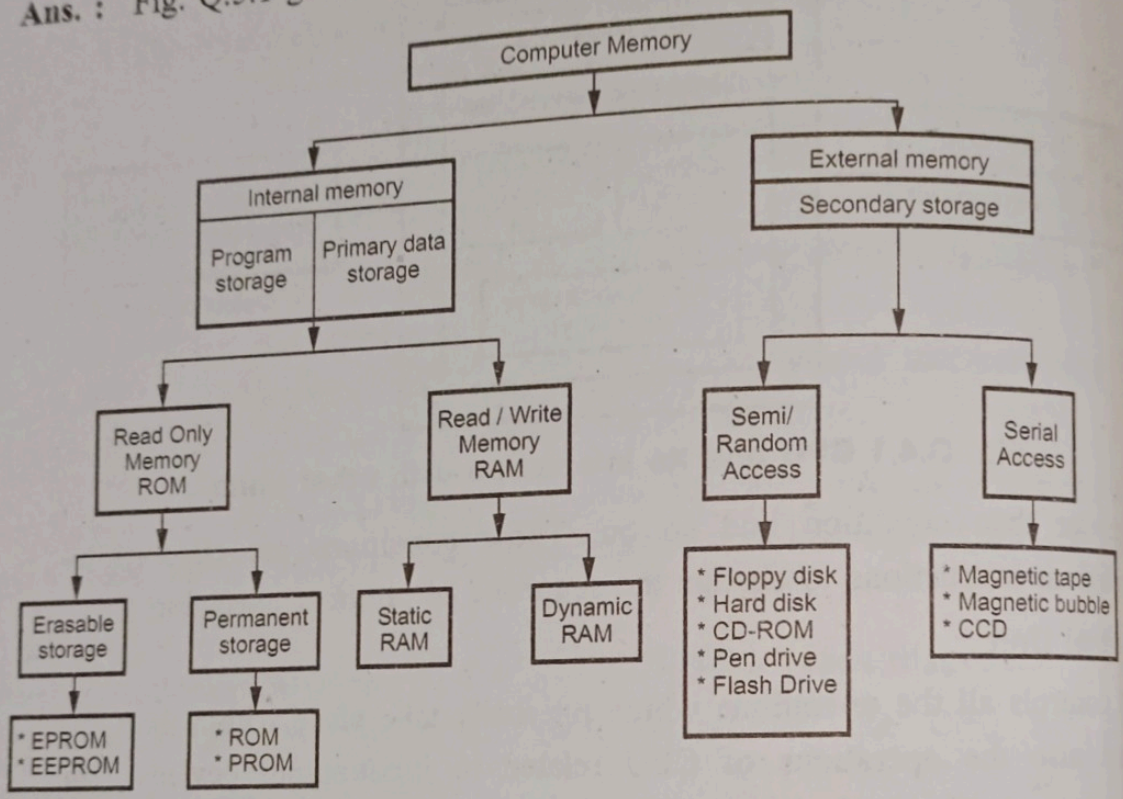


Fig. Q.5.1 Types of memory

- The internal memory is also called **primary memory** as it can be directly accessed by processor with a random access. Due to this such memories are able to respond fast enough to synchronize with the execution speed of the processor.
- On the other hand external memory devices cannot be directly accessed by the processor. These devices are also called **secondary devices.**

External Memory

- External memory consists of floppy drive, hard disk, compact disks, DVDs, magnetic tapes, flash drives, memory cards, charged coupled devices (CCD) etc.
- The primary feature of these devices are :
  - High capacity

- Slow access
- Low cost.

- Flash drive is a small device that can store and transfer data to and from various other devices. It has plenty of other names such as USB drive, pen drive, thumb drive, and more.
- Memory cards are smaller and thinner external memory that users usually store in other devices. The most common types of memory cards are CompactFlash, a Memory Stick, and SD card.
- Nowadays, you don't have to have physical devices to have external memory storage. Online, or "cloud storage" is another option that utilizes the Internet.

Internal Memory

- Internal memory is further subdivided into program storage memory and data storage memory. Typically, internal memory is implemented with both, ROM and RAM ICs.
- Data, whether they are to be interpreted as numbers, characters, or instructions, can be stored in either ROM or RAM. RAM and ROM memories consist of an array of registers, in which each register has unique address.

Q.6 Give the difference between primary and secondary memory.

Ans. :

Parameter	Primary Storage Memory	Secondary Storage Memory
Types	It includes static RAM and dynamic RAM.	It includes Floppy disk, Hard disk, CDROM, Magnetic tape, Magnetic bubble memory etc.
Storage capacity	It has less storage capacity ranging from 1 kbytes to 512 kbytes.	It has a high storage capacity and it is practically unlimited because when one disk or tape is full, the next one can be used.

Access	Microprocessor can access and process data/program directly from these memories.	Microprocessor cannot directly access or process data/program stored in these memories. Data/program need to be copied into primary storage memory for microprocessor access.
Speed	It can be accessed with a greater speed.	Its access speed is slow.
Cost	Its cost in high for per unit storage capacity.	Its cost is low for per unit storage capacity.
Physical size	Physical size is large for per unit storage capacity.	Physical size is small for per unit storage capacity.

9.5 : Input / Output

Q.7 What are the functions of input/output devices ?

Ans. : We can interact with the personal computer using input-output (I/O) devices. Using input devices, computer can accept data and instructions from the user or another computer system (For example, computer on the Internet). Using output devices, computer can send the processed data to the user or to another computer system.

Q.8 List and explain the functions of various input devices.

Ans. : The function of an input device is to apply data to the computer for processing.

- **Keyboard :** It is the most commonly used input device which accepts text and numbers.
- **Mouse :** It is again the commonly used input device to position the screen cursor.
- **Trackball :** It allows to produce screen cursor movement. It is two-dimensional positioning device.

- **Spaceball** : It is usually used in three-dimensional positioning and selecting operations in virtual-reality systems.
- **Joystick** : It has a small, vertical lever (called the stick) mounted on the base and used to steer the screen cursor around. Both x and y co-ordinate positions can be simultaneously altered by the motion of a single lever in a joystick.
- **Scanner** : The scanner is a device, which is used to store drawings, graphs, photos or text available in printed form for computer processing.
- **Digital Camera** : The still images can be recorded with it. These images can be viewed and edited any time on the computer.
- **Light Pen** : It is a pencil shaped device used to select positions by detecting the light coming from points on the CRT screen. It consists of photoelectric cell housed in a pencil like case.
- **Microphone** : It enables to input data which is in the form of voice or music.
- **Digitizers** : It is used for applications such as tracing. It consists of flat surface which can detect the position of a movable stylus.

Q.9 List and explain the functions of various output devices.

Ans. : The function of an output device is to present processed data to the user.

- **Monitor** : The computer sends processed data (i.e. output) to the monitor when the user needs to observe the output.
- **Printer/Plotter** : When the user needs hard-copy (i.e. paper-copy) of an output, the computer sends output to the printer/Plotter.
- **Head-phones or Speakers** : When the user needs a sound-output, the computer sends output to the head-phones or speakers.

Some devices act as both, input and output devices.

- **Touch Screen** : This is a monitor having touch sensing mechanism. It displays text or icons you can touch. When you touch the screen, special sensors detect the touch and the computer calculates the screen

co-ordinates of point of contact. According to the location of the touch, the computer displays the information or determines the next action which to be performed.

- **Communication Devices :** These devices are used to connect two or more computers to each other, that is, for networking. For example, modems. They enable the computers to communicate through telephone lines.
- Another example is network interface cards (NICs) which allow to connect a group of computers to share data and devices.

9.6 : System Bus

Q.10 What is system bus ?

Ans. : The central processing unit, memory unit and I/O unit are the hardware components/modules of the computer. They work together with communicating each other and have paths for connecting the modules together.

- A group of wires, called **bus** is used to provide necessary signals for communication between modules.
- A bus that connects major computer components/modules (CPU, memory, I/O) is called a **system bus**. The system bus is a set of conductors that connects the CPU, memory and I/O. modules.
- Usually, the system bus is separated into three functional groups :
  - Data Bus
  - Address Bus
  - Control Bus

Data Bus : It consists of 8, 16, 32 or more parallel signal lines. These lines are used to send data to memory and output ports, and to receive data from memory and input port. Therefore, data bus lines are **bi-directional.**

Address Bus : It is an **unidirectional** bus.

- The address bus consists of 16, 20, 24 or more parallel signal lines.

- On these lines the CPU sends out the address of the memory location or I/O port that is to be written to or read from.
- Here, the communication is one way, the address is send from CPU to memory and I/O port and hence these lines are **unidirectional**.

Control Bus : • These lines regulate the activity on the bus.

- The CPU sends signals on the control bus to enable the outputs of addressed memory devices or port devices.

Q.11 Explain the single bus structure.

Ans. :

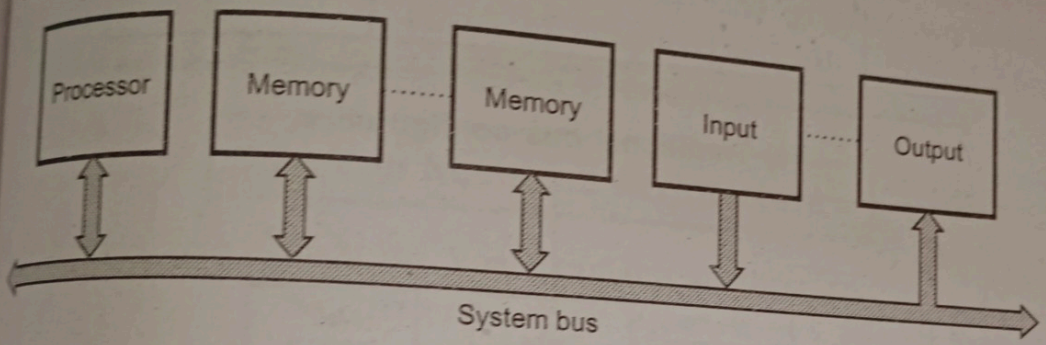
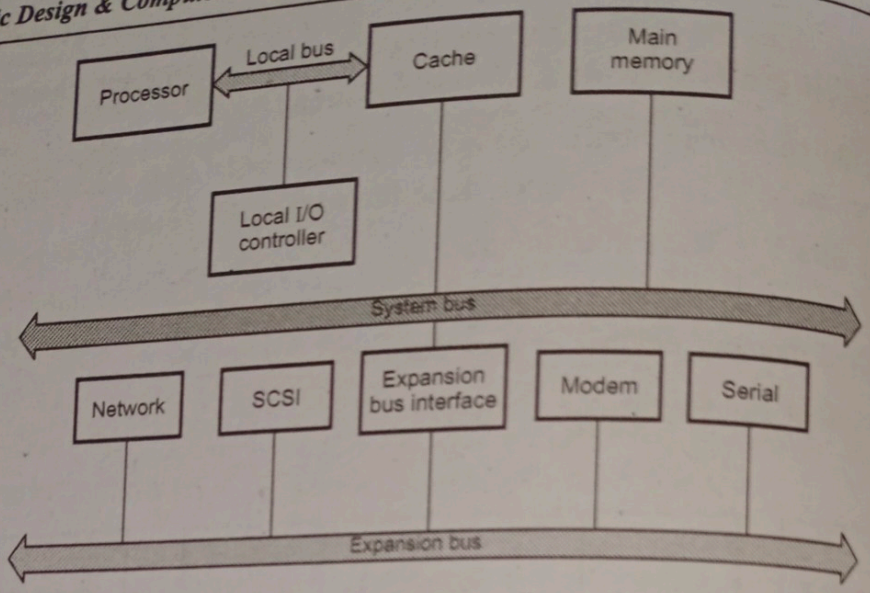


Fig. Q.11.1 Single bus structure

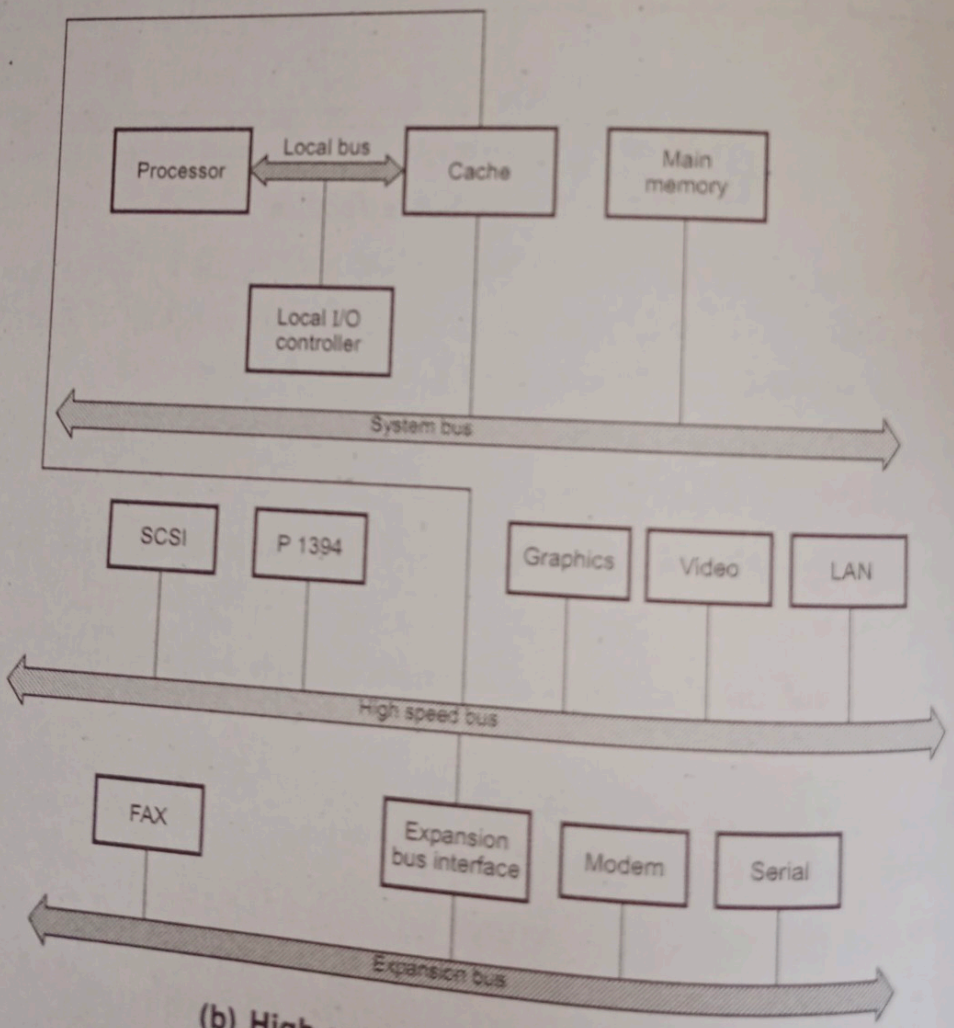
- Here, address bus, data bus and control bus are shown by single bus called **system bus**. Hence such interconnection bus structure is called **single bus structure**. Refer Fig. Q.11.1.
- In a single bus structure all units are connected to **common bus** called **system bus.**
- However, with single bus only two units can communicate with each other at a time.
- The bus control lines are used to arbitrate multiple requests for use of the bus.
- The main advantage of single bus structure is its low cost and its flexibility for attaching peripheral devices.

Q.12 Explain the multibus structure.

Ans. : • The need of high speed shared bus is impractical to satisfy with a single bus. Thus, most computer systems use the multiple buses.



(a) Traditional bus configuration



(b) High-speed bus configuration

Fig. Q.12.1

- These buses have the hierarchical structure.
- Fig. Q.12.1 shows two bus configurations. The traditional bus connection uses three buses : local bus, system bus and expanded bus.
- The high speed bus configuration uses high-speed bus along with the three buses used in the traditional bus connection.
- Here, cache controller is connected to high-speed bus.
- This bus supports connection to high-speed LANs, such as Fiber Distributed Data Interface (FDDI), video and graphics workstation controllers, as well as interface controllers to local peripheral buses including **Small Computer System Interface SCSI** and P1394.

9.7 : Von Neumann Architecture

Q.13 Draw and explain the Von Neumann architecture.

☞ **[SPPU : Dec.-05,08, May-07, Marks 6, June-22, Marks 8]**

Ans. : • Fig. Q.13.1 shows the general structure of a Von Neumann machine (IAS - Institute for Advanced Study, computer).

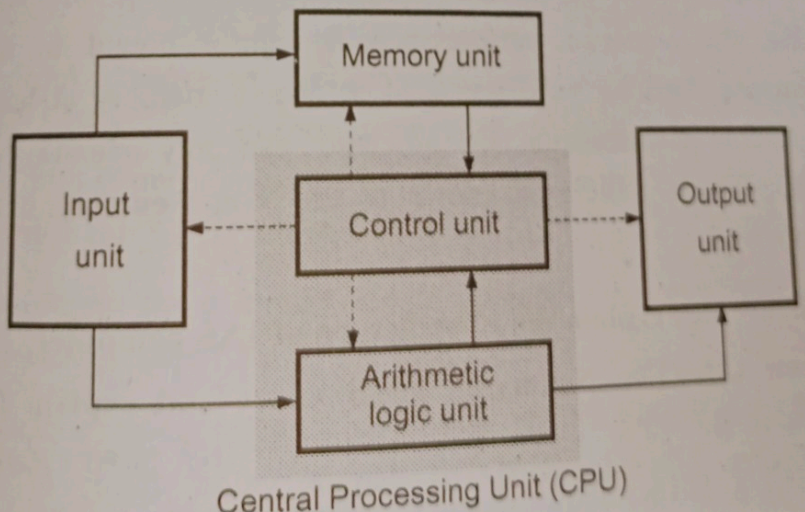


Fig. Q.13.1 Structure of Von Neumann machine (IAS computer)

- It consists of five basic units whose functions can be summarized as follows :
  - The **input unit** transmits data and instructions from the outside world to machine. It is operated by control unit.

- The **memory unit** stores both, data and instructions.
- The **Arithmetic-Logic Unit** (ALU) performs arithmetic and logical operations.
- The **control unit** fetches and interprets the instructions in memory and causes them to be executed.
- The **output unit** transmits final results and messages to the outside world.

Q.14 List the features of Von Neumann architecture.

Ans. : Features of Von Neumann architecture are :

- It uses stored program concept. The program (instructions) and data are stored in a single read-write memory.
- The contents of read-write memory are addressable by location, without regard to the type of data contained there.
- Execution of instructions occurs in a sequential manner (unless explicitly modified) from one instruction to the next.

Q.15 What is Von Neumann bottleneck ?

Ans. : Because of the stored program architecture of Von-Neumann machine, the processor performance is tightly bound to the memory performance. That is, since we need to access memory at least once per cycle to read an instruction, the processor can only operate as fast as the memory. This is sometimes known as the **Von Neumann bottleneck** or **memory wall**.

Q.16 Draw and explain detail structure of IAS computer.

Or Draw IAS (Von Neumann) architecture and explain function of registers in it.

☞ **[SPPU : May-13 Marks 8]**

Ans. : • Fig. Q.16.1 shows detail structure of IAS computer.

• It consists of various processing and control units, along with a set of high speed registers (AC, MQ, DR, IBR, PC, IR and AR). These registers are used to store instructions, memory addresses and data.

• The complete instruction cycle involves three operations : Instruction fetching, opcode decoding and instruction execution.

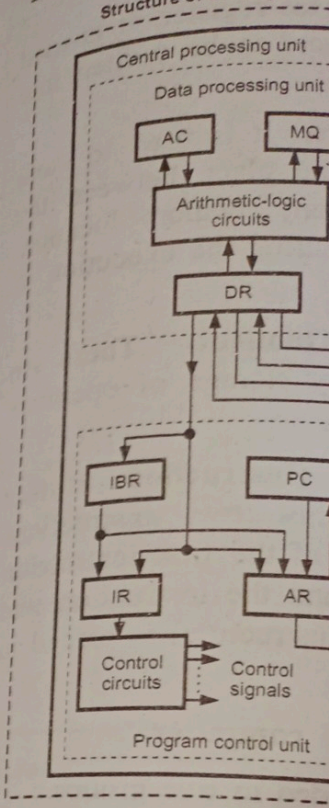


Fig.

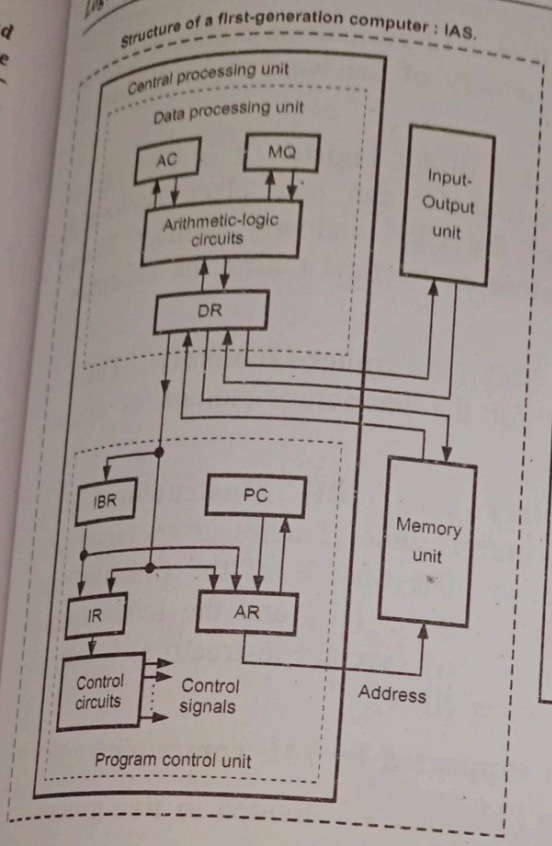
• The control circuits fetching instructions through the system Processing Unit (CP

• After decoding, the perform actions spec

• An electronic clock generate the basic t different parts of the

• The functioning of di

▪ **PC (Program C** store the address also referred to a



Register	Abbreviation
Accumulator	AC
Multiplier Quotient	MQ
Data Register	DR
Program Counter	PC
Instruction Buffer Register	IBR
Address Register	AR
Instruction Register	IR

Fig. Q.16.1 Structure of IAS computer

- The control circuits in the program control unit are responsible for fetching instructions, decoding opcodes, routing information correctly through the system and providing proper control signals for Central Processing Unit (CPU) actions.
- After decoding, the arithmetic logic circuits of the data processing unit perform actions specified by the instruction.
- An electronic clock circuit (not shown in the Fig. Q.16.1) is used to generate the basic timing signals to synchronize the operation of the different parts of the system.
- The functioning of different registers is as given below :
  - **PC (Program Counter)** : It is an address register. It is used to store the address of the next instruction to be executed and hence also referred to as instruction address register.

- **AR (Address Register)** : It is a 12-bit address register. It is used to specify the address in memory of the word to be written into or read from the DR.
- **DR (Data Register)** : It is a 40-bit register. It is used to store any 40-bit word. A word transfer can take place between the 40-bit data register DR of the CPU and any memory location. The DR may be used to store an operand during the execution of an instruction.
- **AC (Accumulator) and MQ (Multiplier-Quotient)** : These are two 40-bit registers used for the temporary storage of operands and results.
- **IR (Instruction Register) and IBR (Instruction Buffer Register)** : Program control unit fetches two instructions simultaneously from memory. The opcode of the first instruction is placed in the Instruction Register (IR) and the instruction that is not to be executed immediately (second instruction) is placed in the Instruction Buffer Register (IBR).

Q.17 Explain the instructions supported by IAS computer.

Ans. : The instructions of IAS computer are divided in five groups :

- Data transfer
- Unconditional branch
- Conditional branch
- Arithmetic
- Address modify

Table Q.17.1 shows the instruction set of IAS computer.

Instruction type	Shorthand notation	Description
Data transfer	$AC \leftarrow MQ$	Transfer contents of register MQ to the accumulator AC
	$MQ \leftarrow M(X)$	Transfer contents of memory location X to MQ

	M(X)
	AC ←
	AC ←
	AC
	AC
Unconditional branch	go
	go
Conditional branch	if A
	M(
	if
	M
Arithmetic	A
	A
	A

- **AR (Address Register)** : It is a 12-bit address register. It is used to specify the address in memory of the word to be written into or read from the DR.
- **DR (Data Register)** : It is a 40-bit register. It is used to store any 40-bit word. A word transfer can take place between the 40-bit data register DR of the CPU and any memory location. The DR may be used to store an operand during the execution of an instruction.
- **AC (Accumulator) and MQ (Multiplier-Quotient)** : These are two 40-bit registers used for the temporary storage of operands and results.
- **IR (Instruction Register) and IBR (Instruction Buffer Register)** : Program control unit fetches two instructions simultaneously from memory. The opcode of the first instruction is placed in the Instruction Register (IR) and the instruction that is not to be executed immediately (second instruction) is placed in the Instruction Buffer Register (IBR).

Q.17 Explain the instructions supported by IAS computer.

Ans. : The instructions of IAS computer are divided in five groups :

- Data transfer
- Unconditional branch
- Conditional branch
- Arithmetic
- Address modify

Table Q.17.1 shows the instruction set of IAS computer.

Instruction type	Shorthand notation	Description
Data transfer	$AC \leftarrow MQ$	Transfer contents of register MQ to the accumulator AC
	$MQ \leftarrow M(X)$	Transfer contents of memory location X to MQ

	$M(X) \leftarrow$
	$AC \leftarrow$
	$AC \leftarrow$
	$AC \leftarrow$
	$AC \leftarrow$
Unconditional branch	go to
	go to
Conditional branch	if AC
	M(X,
	if AC
	M(X
Arithmetic	AC
	AC
	AC

	M(X) ← AC	Transfer contents of accumulator to memory location X
	AC ← M(X)	Transfer M(X) to the accumulator
	AC ← – M(X)	Transfer – M(X) to the accumulator
	AC ← | M(X)|	Transfer absolute value of M(X) to the accumulator
	AC ← – |M(X)|	Transfer – |M(X)| to the accumulator
Unconditional branch	go to M(X, 0:19)	Take next instruction from left half of M(X)
	go to M(X, 20:39)	Take next instruction from right half of M(X)
Conditional branch	if AC ≥ 0 then go to M(X, 0:19)	If number in the accumulator is non-negative, take next instruction from left half of M(X)
	if AC ≥ 0 then go to M(X, 20:39)	If number in the accumulator is non-negative, take next instruction from right half of M(X)
Arithmetic	AC ← AC + M(X)	Add M(X) to AC; put the result in AC
	AC ← AC + |M(X)|	Add |M(X)| to AC; put the result in AC
	AC ← AC – M(X)	Subtract M(X) from AC; put the result in AC

	$AC \leftarrow AC - \lvert M(X) \rvert$	Subtract $\lvert M(X) \rvert$ from AC; put the result in AC
	$AC.MQ \leftarrow MQ \times M(X)$	Multiply M(X) by MQ; put most significant bits of result in AC, put least significant bits in MQ
	$MQ.AC \leftarrow AC \times M(X)$	Divide AC by M(X); put the quotient in MQ and the remainder AC
	$AC \leftarrow AC \times 2$	Multiply accumulator by 2, i.e., shift left one bit position
	$AC \leftarrow AC \div 2$	Divide accumulator by 2, i.e., shift right one bit position
Address modify	$M(X, 8:19) \leftarrow AC(28:39)$	Replace left address field at M(X) by 12 rightmost bits of AC
	$M(X, 28:39) \leftarrow AC(28:39)$	Replace right address field at M(X) by 12 rightmost bits of AC

Table Q.17.1 Instruction set of IAS computer

9.8 : Harvard Architecture

Q.18 Draw block diagram for Harvard architecture and explain each block. What are its advantages and disadvantages ?

☞ [SPPU : May-13,14, Marks 6]

Ans. : • Harvard architecture provides separate memory banks for program storage (code memory), the processor stack and variable RAM (data memory), as shown in the Fig. Q.18.1.

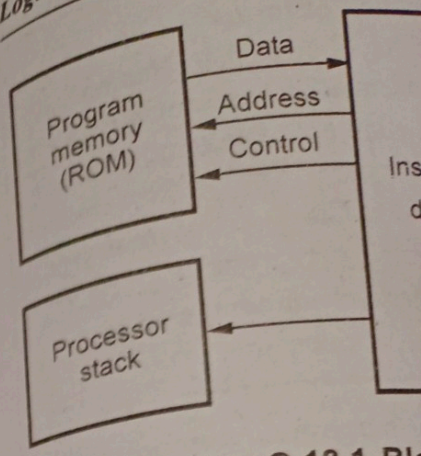


Fig. Q.18.1 Bl

- It has advantage of ex than Princeton (Von Ne
- In Harvard architecture be achieved due to sep

Q.19 Give the compa Architectures.

Ans. :

Sr. No.	Har
1.	It consist of sepa processor, instru e.g.

Program memory — Pro

This architectu code (program

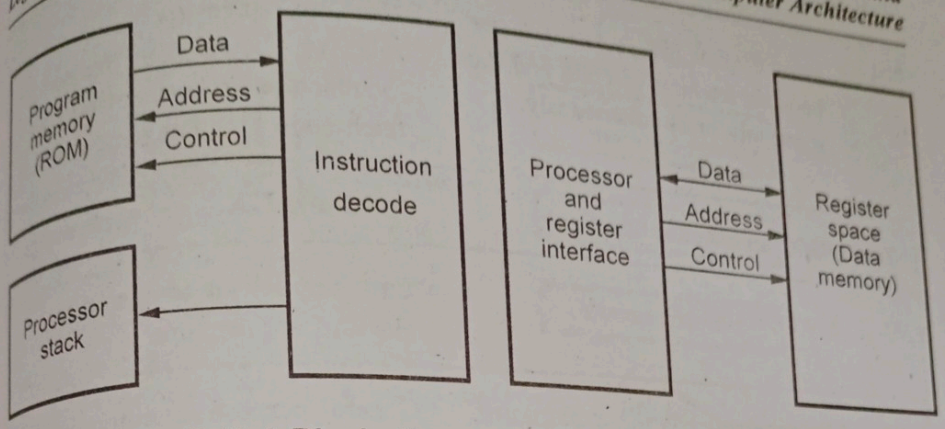
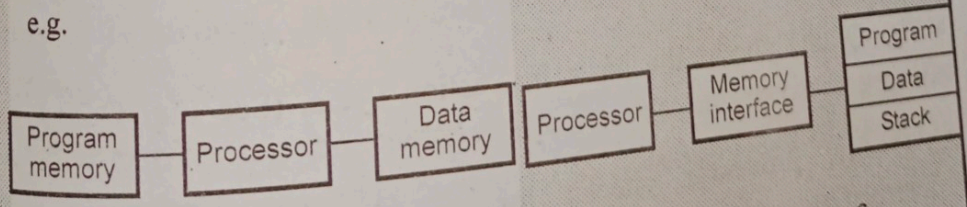


Fig. Q.18.1 Block diagram of Harvard architecture

- It has advantage of executing instructions in **fewer instructions cycles** than Princeton (Von Neumann) architecture.
- In Harvard architecture, **greater amount of instruction parallelism** can be achieved due to separate memory banks.

Q.19 Give the comparison between Harvard and Von-Neumann Architectures. ☞ [SPPU : June-22, Marks 8]

Ans. :

Sr. No.	Harvard	Von-Neumann
1.	It consist of separate memory bank, processor, instruction decode.	It consist of processor, instruction decoder, memory interface unit.
	e.g. 	
	This architecture consist of separate code (program) and data memory.	This architecture consist of memory i.e. program (code) and data used same memory area.

2.	Execution of instruction is faster because fetching of code and data separately or simultaneously.	Execution of instruction is relatively slower than Harvard architecture because not possible to fetch code and data simultaneously.
3.	RISC architecture used only.	RISC and CISC architecture used.
4.	More parallelism.	Less parallelism compared to Harvard architecture.
5.	Microcontroller is the example of Harvard architecture. e.g. Real-time system.	General purpose microprocessor is the example of Von-Neumann architecture.

9.9 : Instruction Cycle

Q.20 Explain the instruction cycle. ☞ [SPPU : May-05, 07, Marks 5]

Or Draw the instruction cycle state diagram.

Ans. : • An instruction cycle involves three subcycles,

- **Fetch :** The fetch phase reads the next instruction from memory into the CPU.
- **Decode :** The decode phase interprets the opcode by decoding it.
- **Execute :** The execute phase performs the indicated operation.

• Fig. Q.20.1 shows the basic instruction cycle. (See Fig. Q.20.1 on next page).

• Actually, processor checks for valid interrupt request after each instruction cycle. If any valid interrupt request is present, Processor saves the current process state and services the interrupt. Servicing the interrupt means executing interrupt service routine. After completing it, processor starts the new instruction cycle from where it has been interrupted. Fig. Q.20.2 shows this instruction cycle with interrupt cycle.

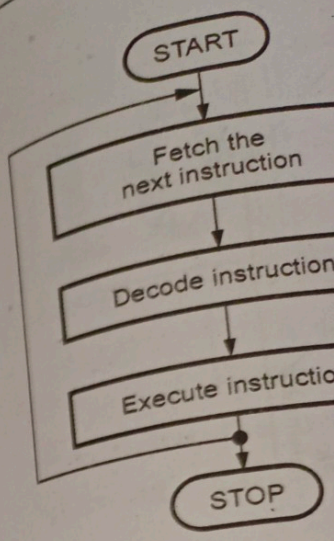


Fig. Q.20.1 Basic

The indirect cy

• If the operand processor-regis execution of a each requires

• For fetching are required.

• After fetchi addressing is addressing.

• Also, after opcode, a s

Q.21 Draw

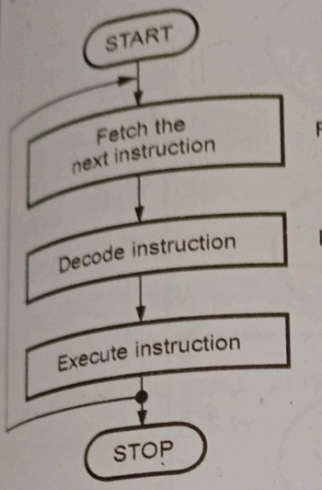


Fig. Q.20.1 Basic instruction cycle

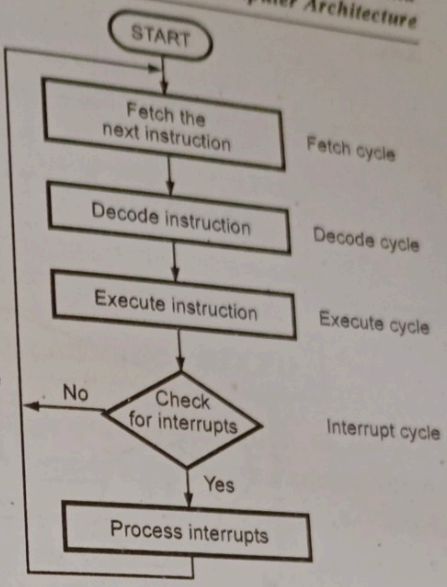


Fig. Q.20.2 Basic instruction cycle with interrupt cycle

The indirect cycle

- If the operands on which the instruction works are present within the processor-registers, a memory access is not required. But if the execution of an instruction involves one or more operands in memory, each requires a memory access.
- For fetching the indirect addresses, one or more instruction sub cycles are required.
- After fetching the instruction, it is decoded and if any indirect addressing is involved, the required operands are fetched using indirect addressing.
- Also, after performing the operation on the operands according to the opcode, a similar process may be needed to store the result in memory.

Q.21 Draw the instruction cycle state diagram.

Ans. :

Fig. Q.21.1 Instruction cycle state diagram

END...

10

10.1 : Sin

Q.1 Draw the neat dia showing ALU, all types

☞ [SPPU : May-

Ans. : • Fig. Q.1.1 show shows how the arithme organized and they

Memory bus

Se

ALU control lines

Fig. Q.1

Unit 4

10 Processor

10.1 : Single Bus Organization of CPU

Q.1 Draw the neat diagram of single bus organization of the CPU showing ALU, all types of registers and the data paths among them.

☞ [SPPU : May-05,07,09,10,12,13,17,19, Dec.-06,08,09,17,19, Marks 8]

Ans. : • Fig. Q.1.1 shows the single bus organization of processor unit. It shows how the arithmetic and logic unit and all processor registers are organized and how they are interconnected.

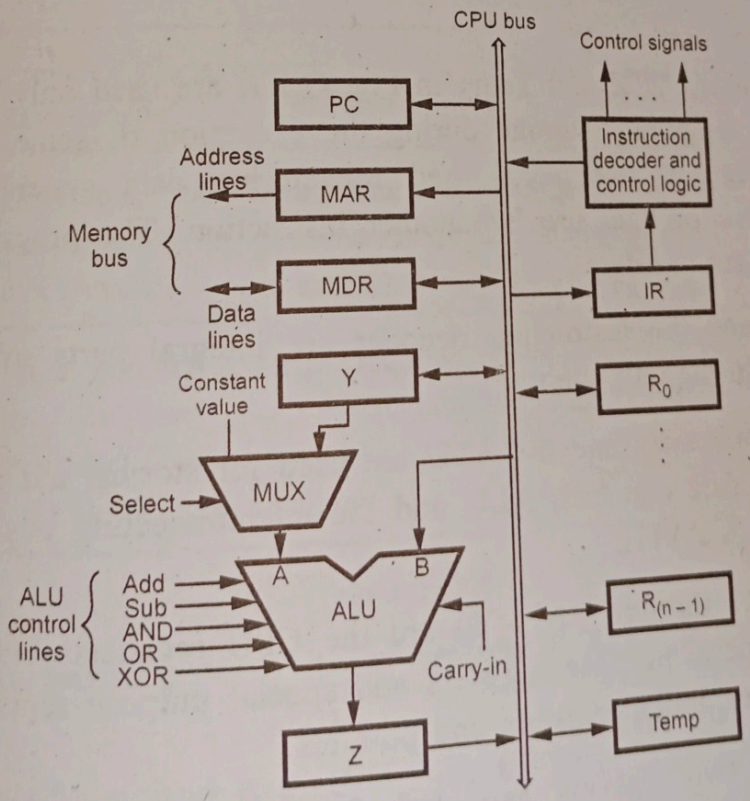


Fig. Q.1.1 Single bus organisation of processor

- The special function registers include Program Counter (PC), instruction register (IR), memory address register (MAR) and memory data register (MDR). It also shows the external memory bus connected to memory address (MAR) and data register (MDR).
- **Program Counter (PC) :** It keeps track of which instruction is being executed and what the next instruction will be.
- **Instruction Register (IR) :** It is used to hold the instruction that is currently being executed. The contents of IR are available to the control logic, which generate the timing signals that control the various processing elements involved in executing the instruction.
- **Memory Address Register (MAR) and Memory Data Register (MDR) :** These registers are used to handle the data transfer between the main memory and the processor. The MAR holds the address of the main memory to or from which data is to be transferred. The MDR sometimes also called MBR (Memory Buffer Register) contains the data to be written into or read from the addressed word of the main memory.
- The registers Y, Z and Temp in Fig. Q.1.1, are used only by the CPU unit for temporary storage during the execution of some instructions. These registers are never used for storing data generated by one instruction for later use by another instruction. The programmer can't access these registers.
- The IR and the instruction decoder are integral parts of the control circuitry in the CPU unit.
- All other registers and the ALU are used for storing and manipulating data. The data registers, ALU and the interconnecting bus are referred to as DATA PATH.
- Register R0 through R(n - 1) are the CPU registers. These registers include general purpose registers and special purpose registers such as stack pointer, index registers and pointers.
- There are two options provided for A input of the ALU. The multiplexer (MUX) is used to select one of the two inputs. It selects either output of Y register or a constant number as an A input for the ALU according to the status of the select input.

- It selects output of Y when select input is 1 (select Y) and it selects a constant number when select input is 0 (select C) as an input A for the multiplexer. The constant number is used to increment the contents of program counter.
- In a single bus organization, only one data word can be transferred over the bus in a clock cycle. This increases the steps required to complete the execution of the instruction. To reduce the number of steps needed to execute instructions, most commercial processors provide multiple bus organization, i.e., multiple internal paths that enable several transfers to take place in parallel.

Q.2 Compare single bus organization with multiple bus organization of CPU. ☞ [SPPU : Dec.-09, Marks 2]

Ans. : • Table Q.2.1 gives comparison between single bus organization and multiple bus organization.

Sr. No.	Single bus organization	Multiple bus organization
1.	Provides single internal path for data transfer.	Provides multiple paths of internal data transfer.
2.	Only one data word can be transferred over the bus in a clock cycle.	Multiple words can be transferred over the bus in a clock cycle.
3.	More number of steps required to complete the instruction execution as compared to multiple bus organization.	Less number of steps required to complete the instruction execution as compared to single bus organization.

Table Q.2.1

Q.3 What is flag register / status register ? Explain its use and its structure.

Ans. : The status register is used to store the results of certain condition when certain operations are performed during execution of the program. The **status register** is also referred to as **flag register.** ALU operations and certain register operations may set or reset one or more bits in the status register. Status bits lead to a new set of CPU instructions. These

instructions permit the execution of a program to change flow on the basis of the condition of bits in the status register. So the condition bits in the status register can be used to take logical decisions within the program. Some of the common status register bits are :

1) Carry/Borrow : The carry bit is set when the summation of two 8-bit numbers is greater than 1111 1111 (FFH). A borrow is generated when a large number is subtracted from a smaller number.

2) Zero : The zero bit is set when the contents of register are zero after any operation. This happens not only when you decrement the register, but also when any arithmetic or logical operation causes the contents of register to be zero.

3) Negative or sign : In 2's complement arithmetic, the most significant bit is a sign bit. If this bit is logic 1, the number is negative number, otherwise a positive number. The negative bit or sign bit is set when any arithmetic or logical operation gives a negative result.

4) Auxiliary Carry : The auxiliary carry bit of status register is set when an addition in the first 4 bits causes a carry into the fifth bit. This is often referred as half carry or intermediate carry. This is used in the **BCD arithmetic**.

5) Overflow Flag : In 2's complement arithmetic, most significant bit is used to represent sign and remaining bits are used to represent magnitude of a number (see Fig. Q.3.1). This flag is set if the result of a signed operation is too large to fit in the number of bits available (7-bits for 8-bit number) to represent it.

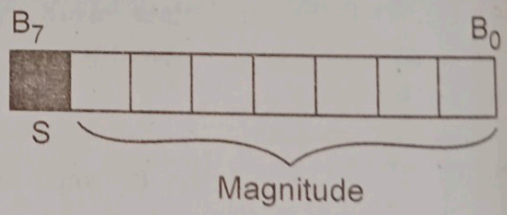


Fig. Q.3.1 2's Complement 8-bit number

For example, if you add the 8-bit signed number 01110110 (+118 decimal) and the 8-bit signed number 00110110 (+ 54 decimal). The result will be 10101100 (+ 172 decimal), which is the correct binary result, but in this case it is too large to fit in the 7-bits allowed for the magnitude in an 8-bit signed number. The overflow flag will be set after this operation to indicate that the result of the addition has overflowed into the sign bit.

6) Parity : When the result of an operation leave the indicated register with an even number of 1s, parity bit is set.

Q.4 What are general purpose registers ?

Ans. : In addition to the special purpose register, most CPUs have other registers called **general purpose registers.** The general purpose registers are used as simple storage area, mainly these are used to store intermediate results of the operation. Getting the operand from the general purpose registers is more faster than from memory so it is better to have sufficient number of general purpose registers.

10.2 : Operations of Control Unit

Q.5 What is instruction sequencing ?

Ans. : Every processor has some basic types of instructions such as data transfer instructions, arithmetic instructions, logical instructions, branch instructions and so on. To perform a particular task on the computer, it is programmer's job to select and write appropriate instructions one after the other, i.e. programmer has to write instructions in a proper sequence. This job of programmer is known as **instruction sequencing**.

Q.6 What is straight - line sequencing ?

Ans. : Processor executes a program with the help of Program counter (PC). PC holds the address of the instruction to be executed next. To begin execution of a program, the address of its first instruction is placed into the PC. Then, the processor control circuits use the information (address of memory) in the PC to fetch and execute instructions, one at a time, in the order of increasing addresses. This is called **straight-line sequencing**.

Q.7 Write a short note on register transfers.

Ans. : • In Fig. Q.7.1, the data transfer between registers and common bus is shown by a line with arrow heads. But in actual practice each register has input and output gating and these gates are controlled by corresponding control signals. This is illustrated in Fig. Q.7.1.

• As shown in Fig. Q.7.1, control signals. $R_{i\,in}$ and $R_{i\,out}$ controls the input and output gating of register R_i. When $R_{i\,in}$ is set to 1, the data

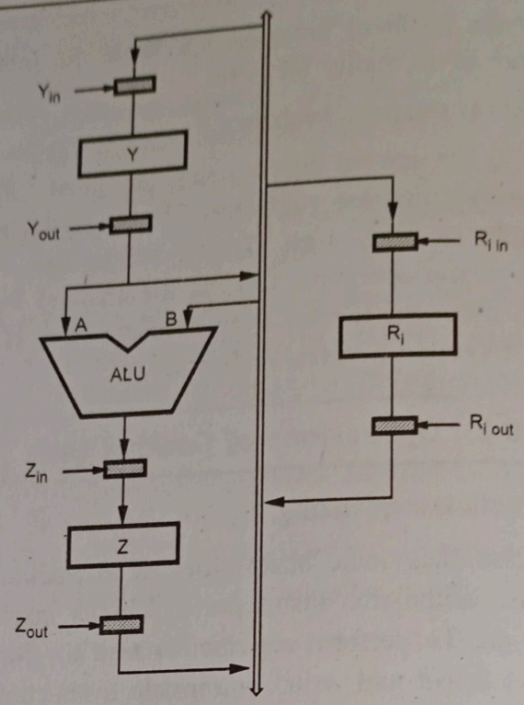


Fig. Q.7.1 Input and output gating for the register

available on the common data bus is loaded into register R_i. Similarly, when $R_{i\,out}$ is set to 1, the contents of register R_i are placed on the common data bus.

- The signals $R_{i\,in}$ and $R_{i\,out}$ are commonly known as input enable and output enable signals of registers, respectively.
- The transfer of data from register R_1 to R_2 can be accomplished as follows :
  - Activate the output enable signal of R_1, $R_{1out} = 1$. This places the contents of R_1 on the common bus.
  - Activate the input enable signal of $R_2 R_{2out} = 1$. This loads data from the common bus into the register R_2.
- All operations and data transfers within the processor take place in synchronisation with the clock signal.

Q.8 What are micro-operatio

Ans. : • To perform fetch, dec has to perform set of op operations include :

- Transfer a word of data to the ALU.
- Perform the arithmetic CPU registers and store
- Fetch a word of data them into a CPU regist
- Store a word of data f location.

Q.9 Explain the sequence word from memory and s

Ans. : Fetching a word fr

- To fetch a word of data of the memory location activates the read opera
- The processor loads th connected to the addres
- At the same time proc bus to indicate the rea
- When the requested da the MDR, from where

Storing a word in memo

- To write a word of d the address of the de to be written in mem

Q.10 Why is wait-for- reading from or writi

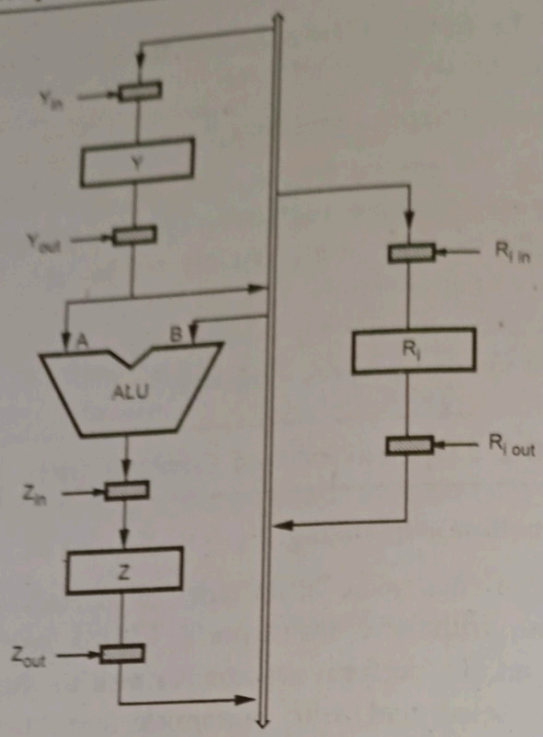


Fig. Q.7.1 Input and output gating for the register

available on the common data bus is loaded into register R_i. Similarly, when $R_{i\,out}$ is set to 1, the contents of register R_i are placed on the common data bus.

- The signals $R_{i\,in}$ and $R_{i\,out}$ are commonly known as input enable and output enable signals of registers, respectively.
- The transfer of data from register R_1 to R_2 can be accomplished as follows :
  - Activate the output enable signal of R_1, $R_{1out} = 1$. This places the contents of R_1 on the common bus.
  - Activate the input enable signal of $R_2 R_{2out} = 1$. This loads data from the common bus into the register R_2.
- All operations and data transfers within the processor take place in synchronisation with the clock signal.

Q.8 What are micro-oper

Ans. : • To perform fetch, has to perform set of operations include :

- Transfer a word of to the ALU.
- Perform the arithme CPU registers and s
- Fetch a word of d them into a CPU re
- Store a word of da location.

Q.9 Explain the sequer word from memory an

Ans. : Fetching a word

- To fetch a word of of the memory locati activates the read op
- The processor loads connected to the add
- At the same time bus to indicate the
- When the requested the MDR, from wh

Storing a word in me

- To write a word o the address of the to be written in me

Q.10 Why is wait-fo reading from or wri

Q.8 What are micro-operations ?

Ans. : • To perform fetch, decode and execute cycles, the processor unit has to perform set of operations called **micro-operations**. These operations include :

- Transfer a word of data from one CPU register to the another or to the ALU.
- Perform the arithmetic or logic operations on the data from the CPU registers and store the result in a CPU register.
- Fetch a word of data from specified memory location and load them into a CPU register.
- Store a word of data from a CPU register into a specified memory location.

Q.9 Explain the sequence of operations needed to perform fetching a word from memory and storing a word in memory.

☞ [SPPU : May-12, June-22, Marks 8]

Ans. : Fetching a word from memory

- To fetch a word of data from memory, the processor gives the address of the memory location where the data is stored on the address bus and activates the read operation.
- The processor loads the required address in MAR, whose output is connected to the address lines of the memory bus.
- At the same time processor sends the read signal of memory control bus to indicate the read operation.
- When the requested data is received from the memory it is stored into the MDR, from where it can be transferred to other processor registers.

Storing a word in memory

- To write a word of data into a memory location processor has to load the address of the desired memory location in the MAR, load the data to be written in memory, in MDR and activate write operation.

Q.10 Why is wait-for-memory-function-completed steps needed when reading from or writing from main memory ?

Ans. : • When access time of memory is larger than the maximum access time allowed in memory read cycle we need to check wait-for-memory-function - complete signal to varify the completion of memory read operation before going to start the next operation.

Q.11 State the actions needed to execute MOVE R_3, (R_2) instruction and also draw the timing diagram.

Ans. : The actions needed to execute MOVE R_3, (R_2) instruction are as follows :

1. MAR $\leftarrow [R_2]$

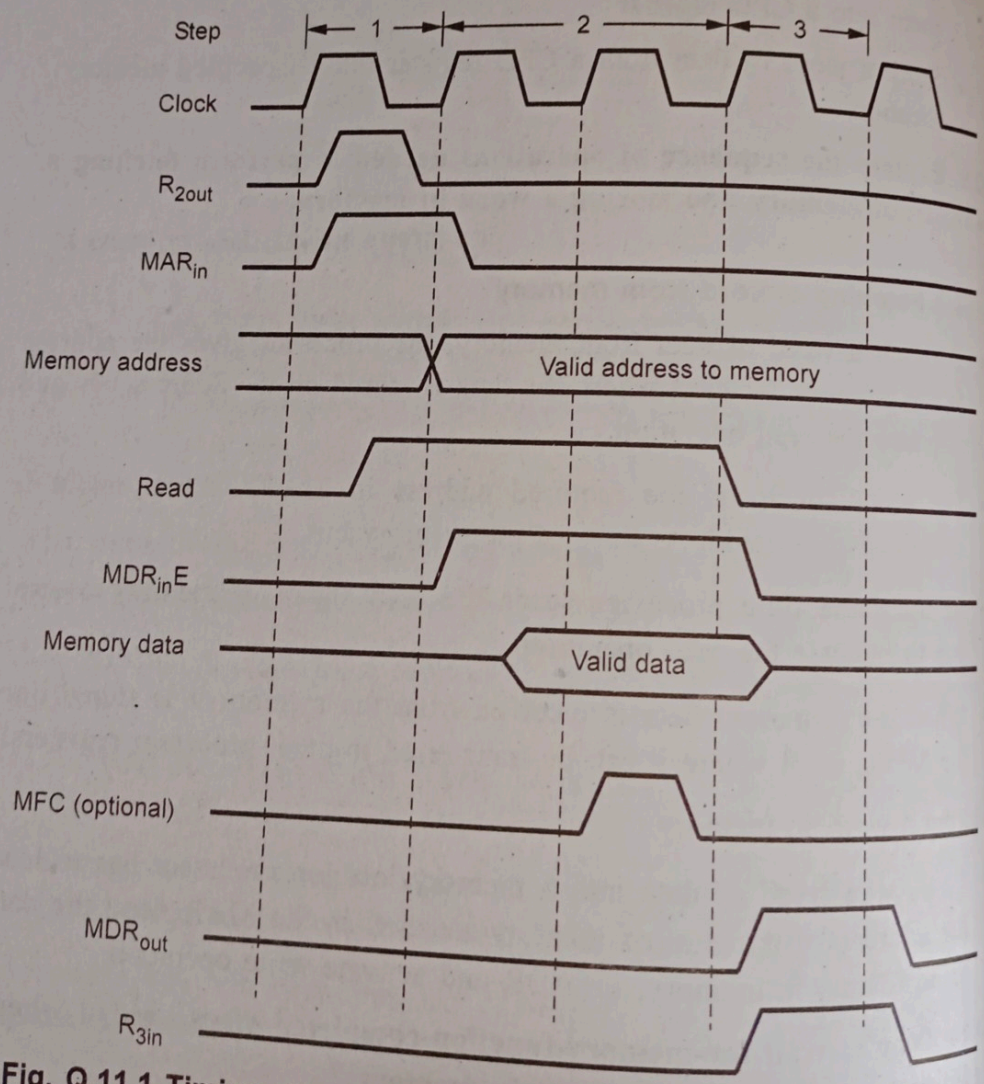


Fig. Q.11.1 Timing diagram for MOVE R_3 (R_2) instruction (memory read operation)

2. Activate the control s
slow, activate wait for M
3. Load MDR from the
4. $R_3 \leftarrow$ [MDR]
The Fig. Q.11.1 shows

Q.12 State the actions and also draw the tim

Ans. :

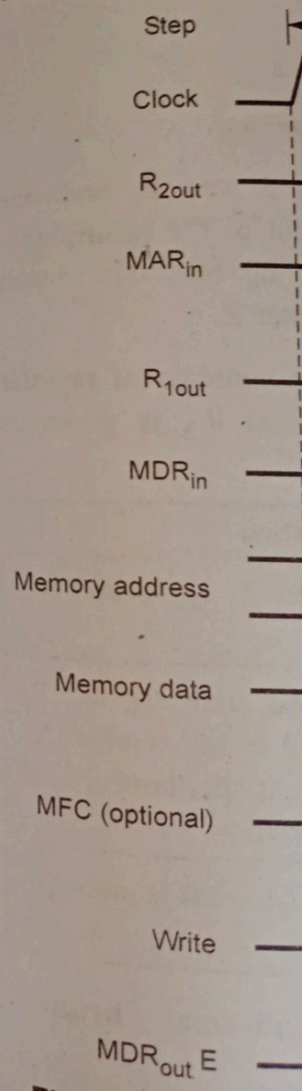


Fig. Q.12.1 Timin

2. Activate the control signal to perform the Read operation. If memory is slow, activate wait for Memory Function Complete (MFC).

3. Load MDR from the memory bus

4. $R_3 \leftarrow [MDR]$

The Fig. Q.11.1 shows the timing diagram of a memory read operation.

Q.12 State the actions needed to execute MOVE (R_2), R_1 instruction and also draw the timing diagram.

Ans. :

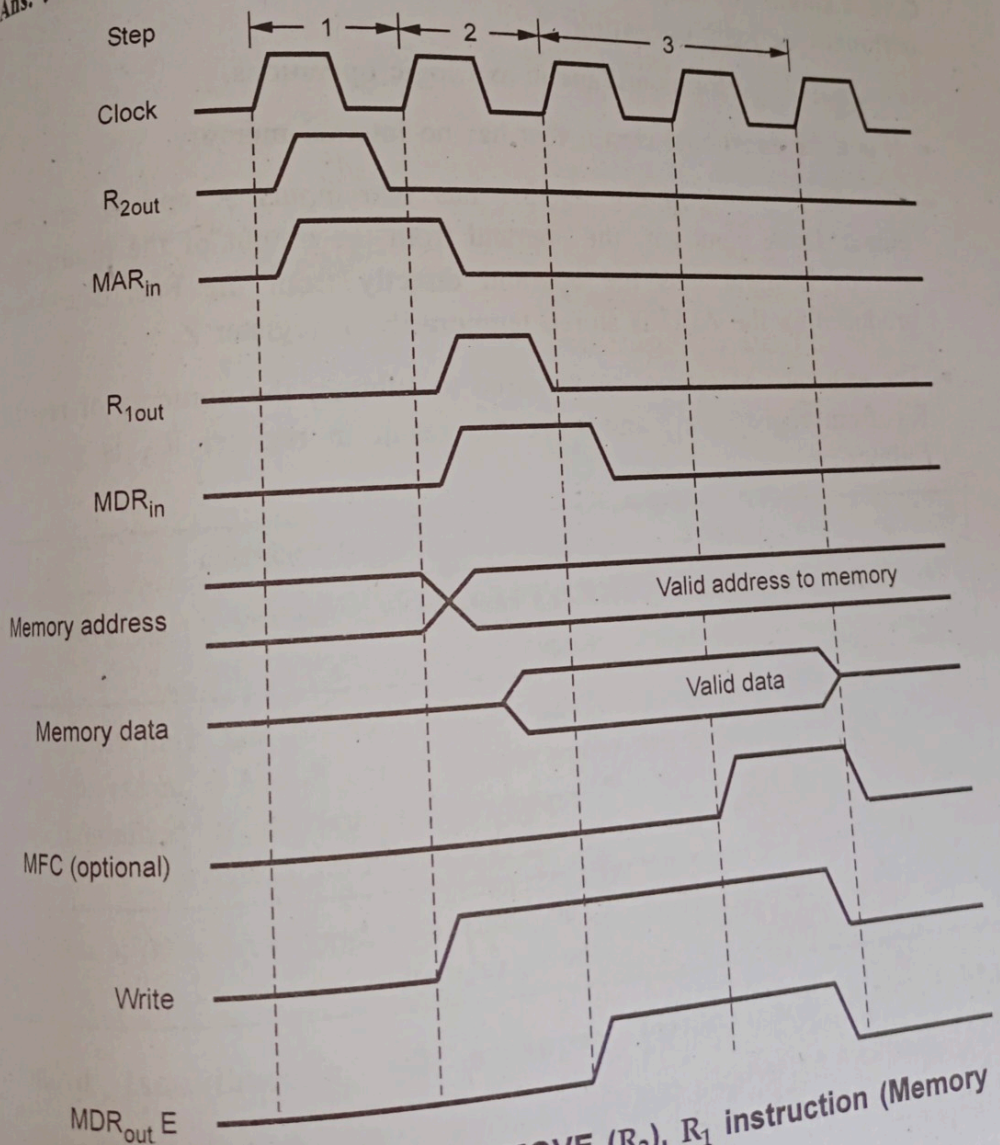


Fig. Q.12.1 Timing diagram for MOVE (R_2), R_1 instruction (Memory write operation)

- The actions needed to execute MOVE $(R_2), R_1$ instruction are as follow.

1. MAR $\leftarrow [R_2]$
2. MDR $\leftarrow [R_1]$
3. Activate the control signal to perform the write operation. If memory is slow, wait for Memory Function Complete (MFC).

The Fig. Q.12.1 (see on previous page) shows the timing diagram of a memory write operation.

Q.13 Explain the sequence of operation required for performing an arithmetic or logic operation.

Ans. : • ALU performs arithmetic and logic operations.

- It is a combinational circuit that has no internal memory.
- The ALU shown in Fig. Q.13.1 has two inputs A and B, and one output. Its A input gets the operand from the output of the multiplexer and its B input gets the operand directly from the bus. The result produced by the ALU is stored temporarily in register Z.
- The sequence of operations required to subtract the contents of register R_2 from register R_1, and store the result in register R_3 is given as follows :

Sr. No.	Action	Description
1.	R_{1out}, Y_{in}	The contents from register R_1 are loaded into register Y.
2.	R_{2out}, Select Y, Sub, Z_{in}	The contents from Y and from register R_2 are applied to the A and B inputs of ALU, respectively, subtraction is performed, and result is stored in the Z register.
3.	Z_{out}, R_{3in}	The contents of Z-register (result) is stored in the R_3 register.

Q.14 Explain the control sequence for unconditional branch instruction.

Ans. : The branch instru
that PC will fetch the ne
branch target address is
contents of PC. The offs
sequence for condition

Sr. No.	Ac
1.	PC_{out}, MAR_{in}, Add, Z_{in}
2.	Z_{out}, PC_{in}, Yi
3.	MDR_{out}, IR_{in}

Ans. : The branch instruction loads the branch target address in PC so that PC will fetch the next instruction from the branch target address. The branch target address is usually obtained by adding the offset in the contents of PC. The offset is specified within the instruction. The control sequence for unconditional branch instruction is as follows :

Sr. No.	Action	Description
1.	PC_{out}, MAR_{in}, Read, SelectC, Add, Z_{in}	The instruction fetch operation is initiated by loading the contents of the PC into the MAR and activating Read signal. By activating select C input of multiplexer, a constant value is added to the operand at input B, which is the contents of the PC. By activating Z_{in} signal result is stored in the register Z.
2.	Z_{out}, PC_{in}, Yin WMFC	The contents of register Z are transferred to PC register by activating Z_{out} and PC_{in} signal. This completes the PC increment operation and PC will now point to next instruction. After receiving WMFC signal, the contents of specified location are available in MDR register.
3.	MDR_{out}, IR_{in}	The contents of MDR register are transferred to the Instruction Register (IR) of the processor. The step 1 through 3 constitute the instruction fetch phase. At the beginning of step 4, the instruction decoder interprets the contents of the IR. This enables the control circuitry to activate the control signals for steps 4 through 7, which constitute the execution phase.

4.	Offset_field_Of_IR_{out}, SelectY, Add, Z_{in}	The contents of PC and the offset field of IR register are added and result is saved in register Z by activating corresponding signals.
5.	Z_{out}, PC_{in}, End	The contents of register Z are transferred to PC by activating Z_{out} and PC in signals.

Q.15 Write a control sequence for branch on negative.

Ans. : • The Fig. Q.15.1 shows the implementation of instruction Branch on negative. (Branch < 0). When this instruction is loaded into IR, a branch microinstruction transfers control to the corresponding microroutine, which is assumed to start at location 45 in the control memory.

- This address is the output of the starting address generator block in Fig. Q.15.1.
- The microinstruction at location 45 tests the N bit of the condition codes. If this bit is equal to 0, a branch takes place to location 0 to fetch a new machine instruction. Otherwise, the microinstruction at location 46 is executed to put the branch target address into register Z. The microinstruction in location 47 loads this address into the PC.

Address	Microinstruction
0	PC_{out}, MAR_{in}, Read, Y_{in}, SelectC, Add, Z_{in}
1	Z_{out}, PC_{in}, WMFC
2	MDR_{out}, IR_{in}
3	Branch to starting address of appropriate microroutine
45	If N=0, then branch to microinstruction 0
46	offset_field_of_IR_{out}, SelectY, Add, Z_{in}
47	Z_{out}, PC_{in}, End

Fig. Q.15.1 Microroutine for the instruction branch < 0

Q.16 Explain the ne

Ans. : • With the s
transferred over the b
to complete the exe
steps needed to exec
multiple internal pa
parallel.

Q.17 Draw and ex

Ans. : • Fig. Q.17
three buses are used
the figure all gener
register file. The r
input port and two
register in one clo
bus C and data fr
respectively. Buse
the A and B inp
operation result i
(Refer Fig. Q.17.1

- To increment t
fetch the next
as **incremente**
to the length o
the sequence.
connected at t

Q.18 Explain t
multi-bus organ

Ans. : • Add R
and the contents
organisation co
R_2, R_3 are as fo
1. PC_{out}, R = B
3. MDR_{out}, IR_{in}

Q.16 Explain the need of multibus organization.

Ans. : • With the single bus organization only one data word can be transferred over the bus in a clock cycle. This increases the steps required to complete the execution of the instruction. To reduce the number of steps needed to execute instructions, most commercial processors provide multiple internal paths that enable several transfer to take place in parallel.

Q.17 Draw and explain the multi-bus organization.

Ans. : • Fig. Q.17.1 shows a three-bus structure of the processor. Here, three buses are used to connect registers and the ALU of the processor. In the figure all general purpose registers are shown by a single block called **register file**. The register file shown in Fig. Q.17.1 has three ports : one input port and two output ports. So it is possible to access data of three register in one clock cycle; the value can be loaded in one register from bus C and data from two register can be accessed to bus A and bus B, respectively. Buses A and B are used to transfer the source operands to the A and B inputs of the ALU. After performing arithmetic or logic operation result is transferred to the destination operand over bus C. (Refer Fig. Q.17.1 on next page)

• To increment the contents of PC after execution of each instruction to fetch the next instruction, separate unit is provided. This unit is known as **incrementer**. Incrementer increments the contents of PC accordingly to the length of the instruction so that it can point to next instruction in the sequence. The incrementer eliminates the need of multiplexer connected at the A input of ALU.

Q.18 Explain the execution of Add R_1, R_2, R_3 instruction with multi-bus organization.

Ans. : • Add R_1, R_2, R_3. This instruction adds the contents of registers R_2 and the contents of register R_3 and stores the result in R_1. With three-bus organisation control steps required for execution of instruction Add R_1, R_2, R_3 are as follows :

1. PC_{out}, R = B, MAR_{in}, Read, IncPC
2. WMFC
3. MDR_{out}, IR_{in}, R = B
4. R_{2out}, R_{3out}, Add, R_{1in}, End

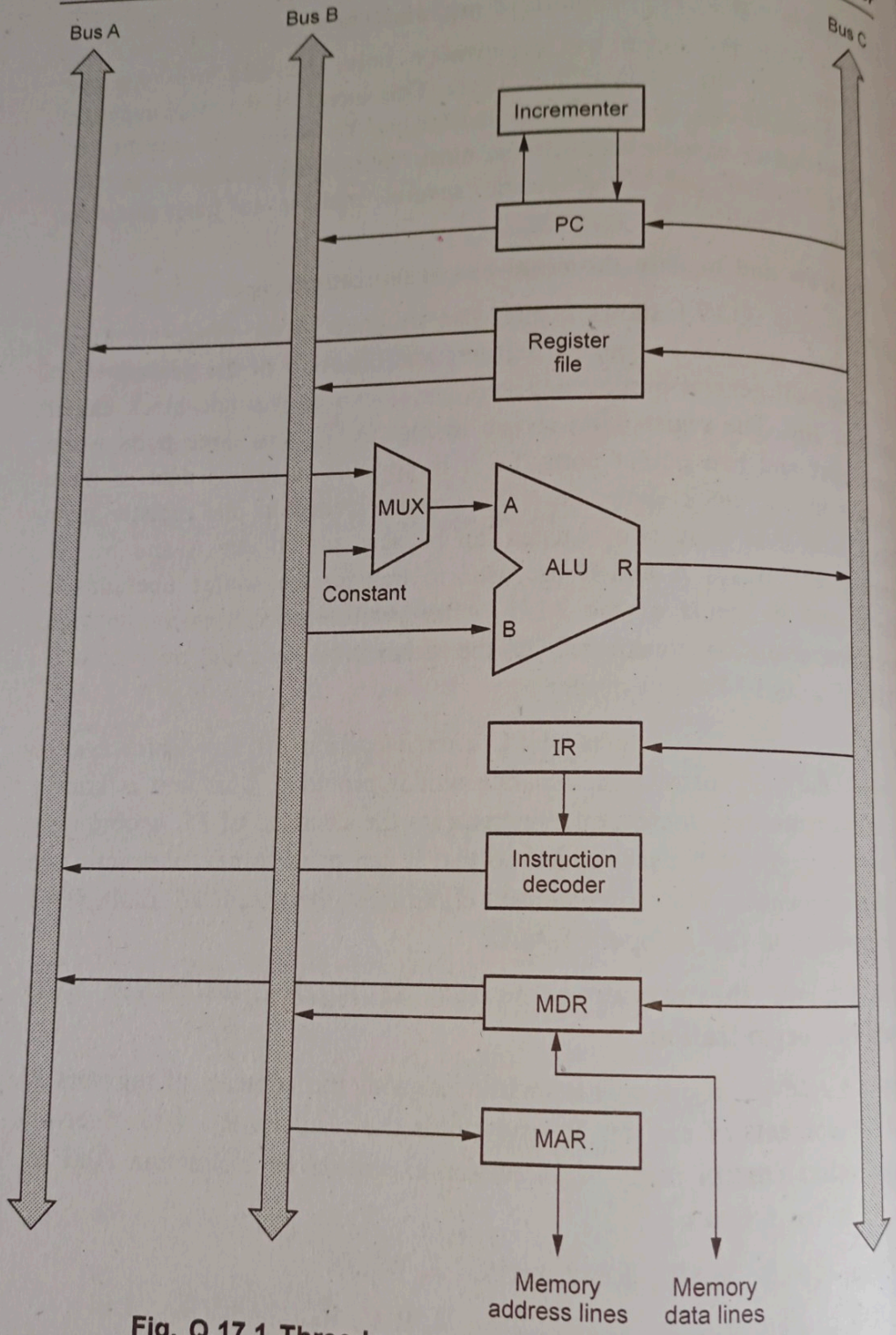


Fig. Q.17.1 Three-bus organisation of processor

- In step 1, the contents start the Read operati the next instruction . the instruction code beginning of step 4, the IR. This enables for step 4, which co
- In step 4, two ope available at A and These two inputs a stored in R_1 through

Q.19 Explain how t system in detail.

Ans. : • Let us cons three-bus organizatio for execution of instr

1. PC_{out}, R = B, M
2. WMFC
3. MDR_{out}, IR_{in},
4. B_{out}, C_{out}, Sub,

- In step 1, the co start the Read op the next instructi the instruction c beginning of ste the IR. This ena for step 4, whic
- In step 4, two available at A a operand C is s and result is sto

- In step 1, the contents of PC are transferred to MAR through Bus B to start the Read operation and simultaneously PC is incremented to point the next instruction . In step 2, the processor waits for MFC. In step 3, the instruction code is transferred from MDR to IR register. At the beginning of step 4, the instruction decoder interprets the contents of the IR. This enables the control circuitry to activate the control signals for step 4, which constitute the execution phase.
- In step 4, two operands from register R_2 and register R_3 are made available at A and B inputs of ALU through BUS A and BUS B. These two inputs are added by activation of **Add** signal and result is stored in R_1 through Bus C.

Q.19 Explain how the instruction A = B – C gets executed in a system in detail.

Ans. : • Let us consider that registers A, B and C are available in the three-bus organization. With three-bus organization control steps required for execution of instruction A = B – C are as follows :

1. PC_{out}, R = B, MAR_{in}, Read, IncPC
2. WMFC
3. MDR_{out}, IR_{in}, R = B
4. B_{out}, C_{out}, Sub, A_{in}, End

- In step 1, the contents of PC are transferred to MAR through Bus B to start the Read operation and simultaneously PC is incremented to point the next instruction. In step 2, the processor waits for MFC. In step 3, the instruction code is transferred from MDR to IR register. At the beginning of step 4, the instruction decoder interprets the contents of the IR. This enables the control circuitry to activate the control signals for step 4, which constitute the execution phase.
- In step 4, two operands from register B and register C are made available at A and B inputs of ALU through BUS A and BUS B. The operand C is subtracted from operand B by activation of **sub** signal and result is stored in register A through bus C.

10.3 : Basic Concepts of Functional Organization of Hardwired Control Unit

Q.20 Draw and explain the block diagram of hardwired control unit.
[SPPU : Dec.-06,16,17,18,19, May-14,18, Marks 6]

Ans. : • In the hardwired control, the control units use fixed logic circuits to interpret instructions and generate control signals from them.

- The fixed logic circuits use contents of the control step counter, contents of the instruction register, contents of the condition code flag and the external input signals such as MFC and interrupt requests to generate control signals.
- Fig. Q.20.1 shows the typical hardwired control unit. Here, the fixed logic circuit block includes combinational circuit (decoder and encoder) that generates the required control outputs, depending on the state of all its inputs.

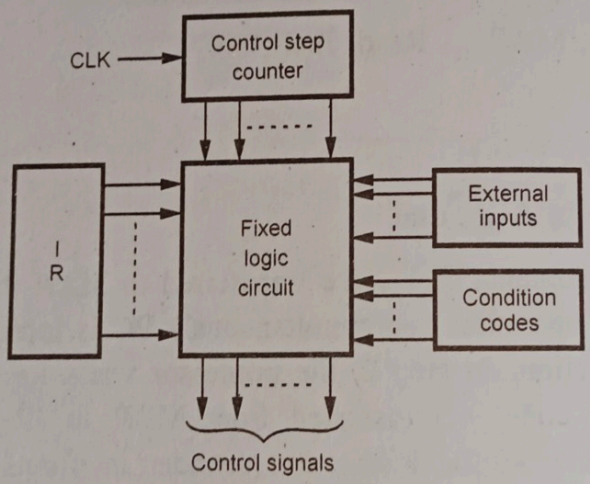


Fig. Q.20.1 Typical hardwired control unit

- By separating the decoding and encoding functions, we can draw more detail block diagram for hardwired control unit as shown in the Fig. Q.20.2.
- The instruction decoder decodes the instruction loaded in the IR. If IR is an 8-bit register then instruction decoder generates 2^8, i.e. 256 lines; one for each instruction.

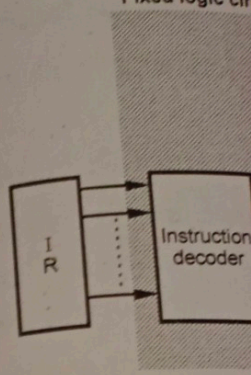


Fig. Q.20.2 Detail

- According to code in decoder goes high i.e.
- The step decoder pro slot, in a control sequ
- The encoder gets in external inputs and c the individual contro
- After execution of e control step counter for next instruction.

Q.21 With the help generated.

Ans. : For the single circuit implements the

$$Z_{in} = T$$

- Fig. Q.21.1 shows bus organization time slot T_1 for during T_4 for an

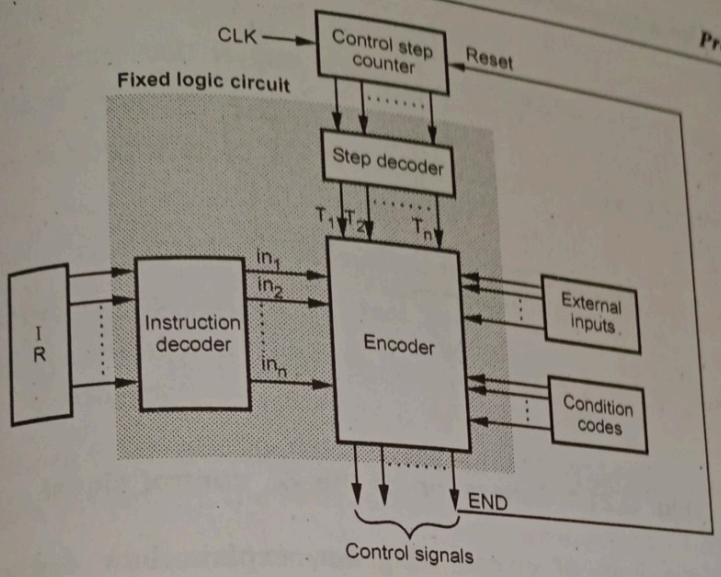


Fig. Q.20.2 Detail block diagram for hardwired control unit

- According to code in the IR, only one line amongst all output lines of decoder goes high i.e., set to 1 and all other lines are set to 0.
- The step decoder provides a separate signal line for each step or time slot, in a control sequence.
- The encoder gets in the input from instruction decoder, step decoder, external inputs and condition codes. It uses all these inputs to generate the individual control signals.
- After execution of each instruction end signal is generated which resets control step counter and make it ready for generation of control step for next instruction.

Q.21 With the help of circuit diagram, explain how Z_{in} signal is generated. ☞ [SPPU : Dec.-07, May-09, Marks 4]

Ans. : For the single bus organization shown in Fig. Q.1.1, the encoder circuit implements the following logic function to generate Z_{in}.

$$Z_{in} = T_1 + T_6 \cdot ADD + T_4 \cdot BR + \ldots\ldots$$

- Fig. Q.21.1 shows the generation of the Z_{in} control signal for single bus organization shown in Fig. Q.1.1. This signal is asserted during time slot T_1 for all instructions, during T_6 of an Add instruction, during T_4 for an unconditional branch instruction and so on.

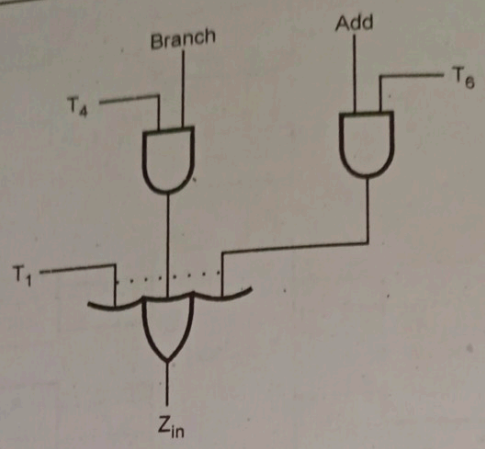


Fig. Q.21.1 Generation of the Z_{in} control signal

Q.22 With the help of circuit diagram, explain how end signal is generated. [SPPU : Dec.-07, May-09, Marks 4]

Ans. : The logic function, to generate end signal is

$$End = T_7 \cdot ADD + T_5 \cdot BR + (T_5 \cdot N + T_4 \cdot \overline{N}) \cdot BRN + ...$$

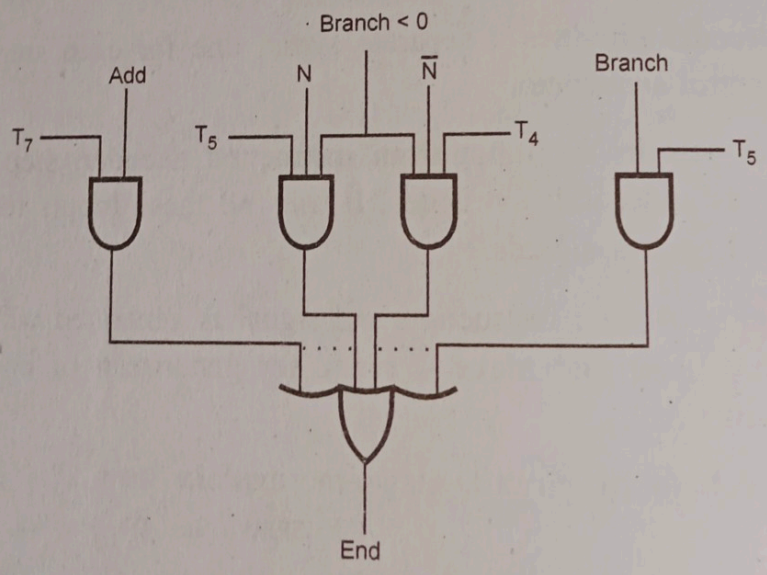


Fig. Q.22.1 Generation of the end control signal

Q.23 State the advantages and disadvantages of hardwired control unit.

Ans. : Advantages

- Hardwired control combinational circ
- The delay in gen gates.
- It has greater ch

Disadvantages of

- More the contro design of contro
- Modifications i rearranging of
- It is difficult feature in exist

10.4 : Ba

Q.24 Define the
i) Microoperati
iii) Microcode

Ans. :

i) **Micro-ope** processor **micro-ope**

ii) **Microinst** microinstr perform

iii) **Microco** produces

iv) **Micropr** designed multiplic

Ans. : Advantages of Hardwired Control Unit

- Hardwired control unit is fast because control signals are generated by combinational circuits.
- The delay in generation of control signals depends upon the number of gates.
- It has greater chip area efficiency since its uses less area on-chip.

Disadvantages of Hardwired Control Unit

- More the control signals required by CPU; more complex will be the design of control unit.
- Modifications in control signal are very difficult. That means it requires rearranging of wires in the hardware circuit.
- It is difficult to correct mistake in original design or adding new feature in existing design of control unit.

10.4 : Basic Concepts of Functional Organization of Micro-programmed Control Unit

Q.24 Define the following terms :
i) Microoperation ii) Microinstruction
iii) Microcode iv) Microprogram.

Ans. :

i) **Micro-operation :** To perform fetch, decode and execute cycles, the processor unit has to perform set of operations called **micro-operations**.

ii) **Microinstruction :** Each word in the control memory is a microinstruction which specifies the control signals to be activated to perform one or more micro-operations.

iii) **Microcode :** The translation of symbolic microprogram to binary produces a binary microprogram called **microcode**.

iv) **Microprogram :** A sequence of one or more micro-operations designed to perform specific operation, such as addition, multiplication is called a microprogram.

Q.25 Define control memory.

Ans. : Microprogramming is a method of control unit design in which the control signal selection and sequencing information is stored in a ROM or RAM called a **control memory** CM.

Q.26 Discuss the basic structure of microprogrammed control unit.

☞ **[SPPU : May-08,11,17,19, Dec.-08,10, Marks 8]**

Ans. : Fig. Q.26.1 shows the microprogrammed control unit. It consists of control memory, control address register, micro instruction register and microprogram sequencer.

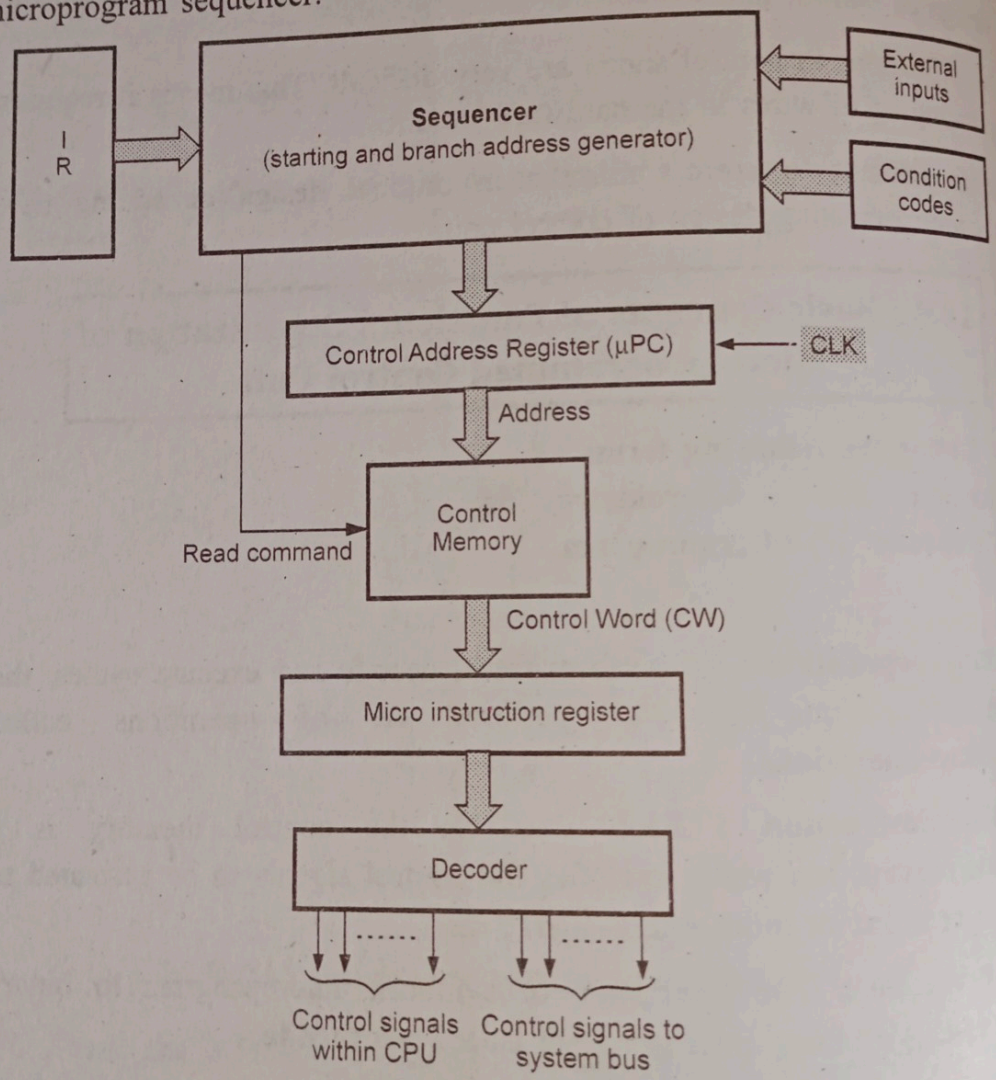


Fig. Q.26.1 Microprogrammed control unit

The components of control unit work together as follows :

- The control address register (μpc) holds the address of the next microinstruction to be read. Every time a new instruction is

loaded into the IR, the output of the block labeled "starting address generator" is loaded into the μpc.

- When address is available in control address register, the sequencer issues READ command to the control memory.
- After issue of READ command, the word from the addressed location is read into the microinstruction register.
- The μpc is then automatically incremented by the clock, causing successive microinstructions to be read from the control memory.
- The content of the micro instruction register generates control signals which are delivered to various parts of the processor in the correct sequence.

- Number of times the control unit is required to check the status of the condition codes or external inputs to choose between alternative courses of action. In such situation, microprogrammed control use conditional branch microinstructions. In additions to the branch address, these instructions specify which of the external inputs, condition codes, or, possibly bits of the instruction register should be checked as a condition for branching to take place.

Q.27 State the advantages and disadvantages of microprogrammed control unit. ☞ [SPPU : May-06, 13, Dec.-12, Marks 8]

Ans. : Advantages :

- It simplifies the design of control unit. Thus it is both, cheaper and less error prone to implement.
- Control functions are implemented in software rather than hardware.
- The design process is orderly and systematic.
- More flexible, can be changed to accommodate new system specifications or to correct the design errors quickly and cheaply.
- Complex function such as floating point arithmetic can be realised efficiently.
- The new or modified instruction set of CPU can be easily implemented by simply rewriting or modifying the contents of control memory.

- The fault can be easily diagnosed in the micro-program control unit using diagnostics tools by maintaining the contents of flags, registers and counters.

Disadvantages :

- A microprogrammed control unit is somewhat slower than the hardwired control unit, because time is required to access the microinstructions from CM.
- The flexibility is achieved at some extra hardware cost due to the control memory and its access circuitry.
- The design duration of micro-program control unit is more than hardwired control unit for smaller CPU.

Q.28 Define vertical and horizontal organization.

Ans. : • Highly encoded scheme that use compact codes to specify only a small number of control functions in each microinstruction are referred to as a **vertical organisation**.

- The minimally encoded scheme, in which resources can be controlled with a single instruction, is called a **horizontal organisation**.

Q.29 Compare horizontal and vertical organisation.

☞ [SPPU : May-10,11,13, Dec.-07,08,09,10,12]

Ans. : Table Q.29.1 shows the comparison between horizontal and vertical organisation.

Sr. No.	Horizontal	Vertical
1.	Long formats.	Short formats.
2.	Ability to express a high degree of parallelism.	Limited ability to express parallel microoperations.
3.	Little encoding of the control information.	Considerable encoding of the control information.
4.	Useful when higher operating speed is desired.	Slower operating speeds.

Table Q.29.1

Q.30 Draw and explain the microinstruction sequencing organization.
☞ [SPPU : May-14, Marks 6]

Ans. : • Fig. Q.30.1 shows microprogram sequencer.

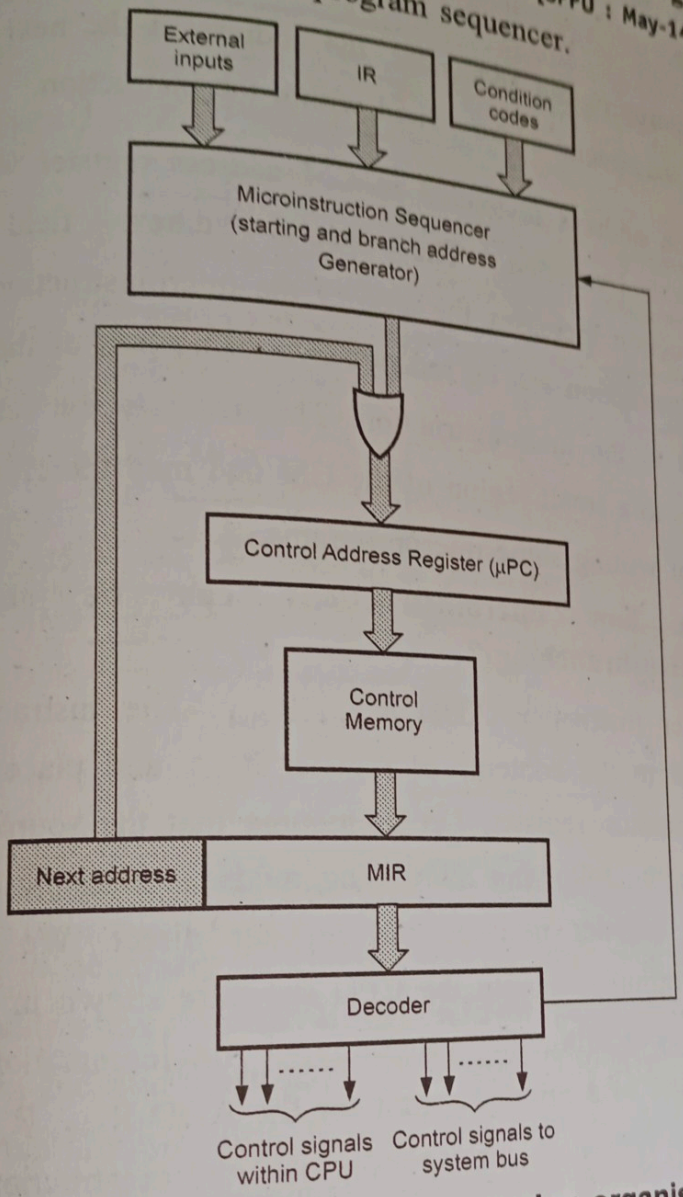


Fig. Q.30.1 Microinstruction sequencing organisation

• The two basic tasks performed by a microprogrammed control unit are :

- **Microprogram sequencing :** Get the next microinstruction from the control memory.
- **Microinstruction execution :** Generate the necessary control signals to execute the microinstruction.

- The task of microprogram sequencing is done by microprogram sequencer.
- In this microprogram sequencer, the address of the next instruction is stored in the special address field within the instruction.
- The branch address is loaded in CM address register when a branch condition is satisfied. The special address field within the microinstruction increases the size of the microinstruction. The size of the microinstruction can be reduced by storing part of the address (low order bits) in the microinstruction. This restricts the range of branch instructions to a small region of the CM and may therefore increase the difficulty of writing some microprograms.

Q.31 Explain how microinstructions can be shared using microinstruction branching ?

Ans. : Consider instruction ADD R_{src}, R_{dst}. The instruction adds the source operand to the contents of register R_{dst} and places the sum in R_{dst}, the destination register. Let us assume that the source operand can be specified in the following addressing modes : Indexed, autoincrement, autodecrement, register indirect and register direct. We now use this instruction in conjunction with the CPU structure shown in Fig. Q.31.1 to demonstrate a possible microprogrammed implementation. Fig. Q.31.2 shows a flowchart of a microprogram for the ADD R_{src}, R_{dst} instruction. Each box in the flowchart corresponds to a microinstruction that controls the transfers and operations indicated within the box. The microinstruction is located at the address indicated by the number above the upper right-hand corner of the box. During the execution of the microinstruction, the branching takes place at point A. The branching address is determined by the addressing mode used in the instruction. (Refer Fig. Q.31.2 on page no. 10 - 26).

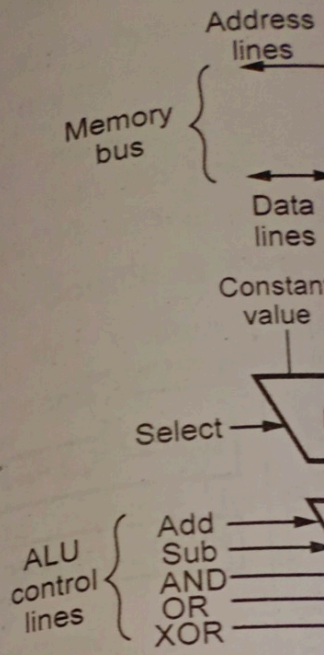


Fig. Q.31.1

At point B, it is necess and indirect addressing instruction, then the mi the operand from the r must be bypassed by remaining microoperati are same and hence different addressing mo

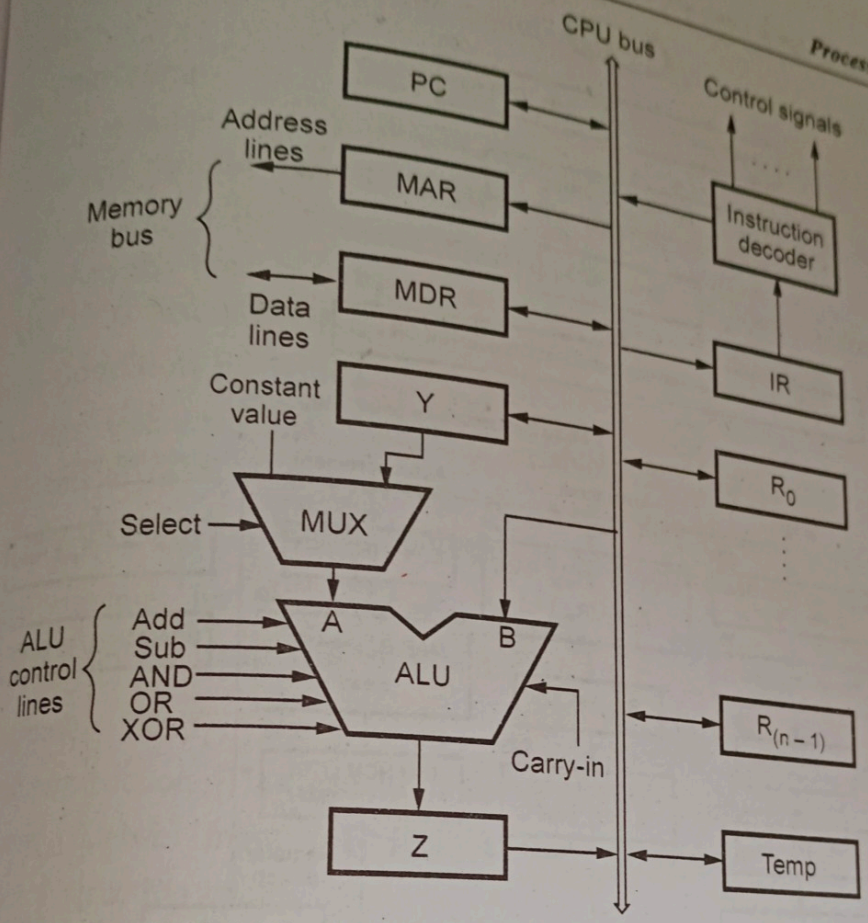


Fig. Q.31.1 Single bus organisation of processor

At point B, it is necessary to choose between actions required by direct and indirect addressing modes. If the indirect mode is specified in the instruction, then the microinstruction in location 170 is performed to fetch the operand from the memory. If the direct mode is specified, this fetch must be bypassed by branching immediately to location 171. The remaining microoperations required to complete execution of instruction are same and hence shared by the instructions having operand with different addressing modes.

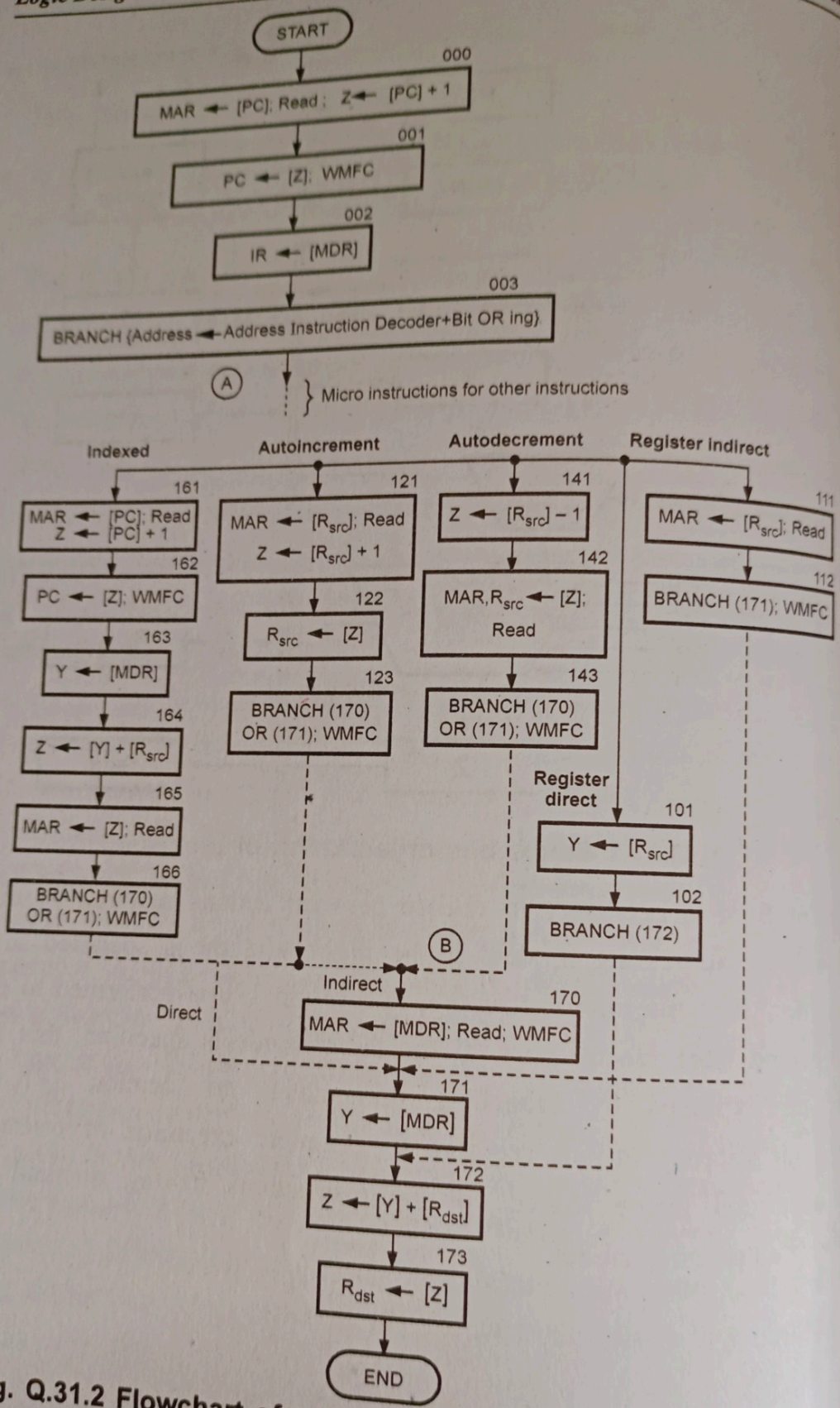


Fig. Q.31.2 Flowchart of a microprogram for the ADD (R_{src}), R_{dst} instruction

Q.32 Explain the techniques for modification or generation of branch addresses.

Ans. : Bit-ORing

- In this technique, the branch address is determined by ORing particular bit or bits with the current address of the microinstruction.
- For example, if the current address is 170 and the branch address is 172 then the branch address can be generated by ORing 02 (bit 1), with the current address.

Using Condition Variables

- In this technique the condition variables are used to modify the contents of the CM address register directly, thus eliminating whole or in part the need for branch addresses in microinstructions.
- For example, let the condition variable CY indicate occurrence of CY = 1 and no carry when CY = 0.
- Suppose that we want to execute a SKIP_ON_CARRY microinstruction. This can be done by logically connecting CY to the count enable input of μpc at on appropriate point in the microinstruction cycle.
- This allows the overflow condition to increment μpc an extra time, thus performing the desired skip operation.

Wide-Branch Addressing

- Generating branch addresses becomes more difficult as the number of branches increases. In such situations Programmable Logic Array can be used to generate the required branch addresses.
- This simple and inexpensive way of generating branch addresses is known as **wide-branch addressing.**
- Here, the opcode of a machine instruction is translated into the starting address of the corresponding micro-routine.

- This is achieved by connecting the opcode bits of the instruction register as inputs to the PLA, which acts as a decoder. The output of the PLA is the address of the desired microroutine.

Q.33 For a single bus organisation of CPU, write a microprogram for instruction. Add $(R_{src})+$, R_{dst}. ☞ [SPPU : Dec.-06, May-07, Marks 8]

Ans. : Let us examine the path needed for the flowchart in Fig. Q.31.2 to execute the instruction Add$(R_{src})+$, R_{dst}.

In this instruction the source operand is accessed in the autoincrement mode and the R_{src} and R_{dst} are general purpose registers in the processor. We assume that the processor has 16 registers that can be used for addressing purpose, each specified using a 4-bit code. We also assume that the instruction has a 3-bit field, (bits 8-10) used to specify the addressing mode for the source operand, as shown in the Fig. Q.33.1. Bit patterns 11,10,01 and 00 located in bits 10 and 9 denote the indexed, autodecrement, autoincrement and register modes respectively. For each of these modes bit-8 is used to specify the indirect version. For example, 100 in the mode field specifies the direct version of the autodecrement mode, whereas 101 specifies the indirect version.

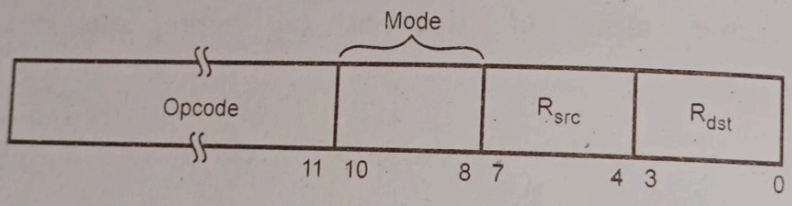


Fig. Q.33.1 Contents of IR

As a part of execution, first the opcode and mode fields are decoded to determine that an R_{src} or R_{dst} register is involved. The decoded output is then used to gate the contents of the R_{src} or R_{dst} fields in the IR into a second decoder, which produces the gating signals for the actual registers R_0 to R_{15}.

The flowchart of a microprogram for the ADD R_{src}, R_{dst} instruction in Fig. Q.31.2 is drawn by combining the microroutines for all possible values of the mode field, resulting in a structure that requires many branch points. The instruction Add $(R_{src})+$, R_{dst} requires two branch microinstructions. In each branch microinstruction, the expression in brackets indicates the branch address that is to be loaded into the µpc and how this address is modified using the bit-ORing scheme. For example,

the branch instruction at location 123 modifies the branch address 170 to 171 by ORing the bit-8 in the IR with bit μpc0 to change the addressing mode from indirect to direct, as shown in Fig. Q.33.2.

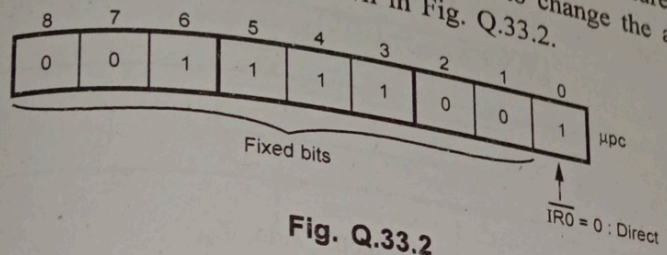


Fig. Q.33.2

The address for branch microinstruction at location 003 is generated as shown in Fig. Q.33.3.

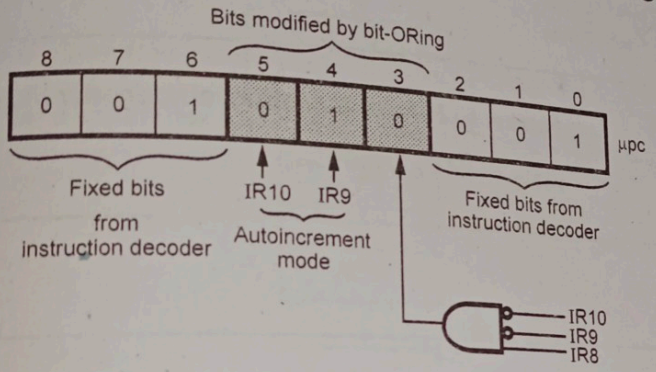


Fig. Q.33.3 Bit ORing for branch microinstruction at location 003

Table Q.33.1 gives the microinstruction sequence for the execution of Add $(R_{src})+$, R_{dst} instruction.

Address (Octal)	Microinstruction
000	PC_{out}, MAR_{in}, Read, SelectC, Add, Z_{in}
001	Z_{out}, PC_{in}, Y_{in}, MWFC
002	MDR_{out}, IR_{in}

0 0 3	μBranch $\begin{cases}\mu pc \leftarrow (101)_8 \text{ from instruction decoder} \\ \mu pc_{5,4} \leftarrow [IR_{10,9}],\ \mu pc_3 \leftarrow [\overline{IR}_{10}] \cdot [\overline{IR}_9] \cdot [IR_8]\end{cases}$ i.e. $\mu pc_{5,4,3} \leftarrow (010)_2$
⋮	
1 2 1	R_{srcout}, MAR_{in}, Read, SelectC, Add, Z_{in}
1 2 2	Z_{out}, R_{srcin}
1 2 3	μBranch $\{\mu pc \leftarrow (170)_8$ from instruction decoder $\mu pc_0 \leftarrow \overline{IR}_8\}$, WMFC
⋮	
1 7 1	MDR_{out}, Y_{in}
1 7 2	R_{dstout}, SelectY, Add, Z_{in}
1 7 3	Z_{out}, R_{dstout}, End

Table Q.33.1 Microinstruction for Add $(R_{src})+$, R_{dst}

Q.34 For a single bus organisation of CPU draw flowchart of a microprogram for instruction Add $(R_{src})+$, R_{dst}.

☞ [SPPU : Dec.-06, May-07, Marks 8]

Ans. : Flowchart

Q.35 For a sin write a microp microprogram R_{dst} uses direct

Ans. : Flowchart

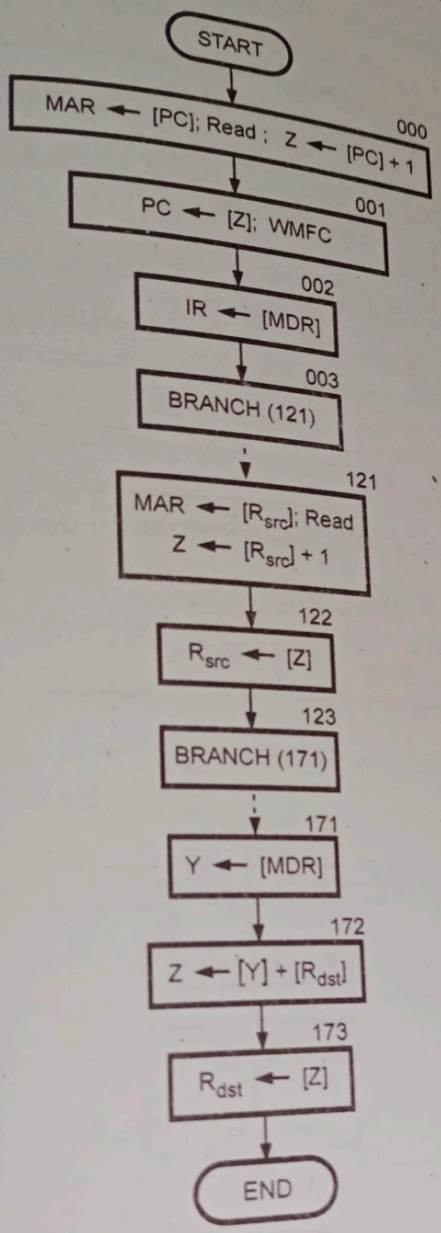


Fig. Q.34.1

Q.35 For a single bus organisation of data paths inside the CPU, write a microprogram of micro-instructions and draw chart of a microprogram for the following instruction. $MOV\ (R_{src}), R_{dst}$. R_{dst} uses direct and R_{src} autoincrement addressing.

☞ [SPPU : Dec.-10, Marks 8]

Ans. :

Address (Octal)	Microinstruction
0 0 0	PC_{out}, MAR_{in}, Read, SelectC, Add, Z_{in}
0 0 1	Z_{out}, PC_{in}, Y_{in}, MWFC
0 0 2	MDR_{out}, IR_{in}
0 0 3	μBranch $\begin{cases} \mu pc \leftarrow (101)_8 \text{ from instruction decoder} \\ \mu pc_{5,4} \leftarrow [IR_{10,9}],\ \mu pc_3 \leftarrow [\overline{IR}_{10}] \cdot [\overline{IR}_9] \cdot [IR_8] \end{cases}$
⋮	
1 1 1	R_{srcout}, MAR_{in}, Read
1 1 2	μBranch{$\mu pc \leftarrow (270)_8$ from instruction decoder $\mu pc_0 \leftarrow \overline{IR}_8$}, WMFC
2 7 1	MDR_{out}, R_{dstin}, End

Flowchart

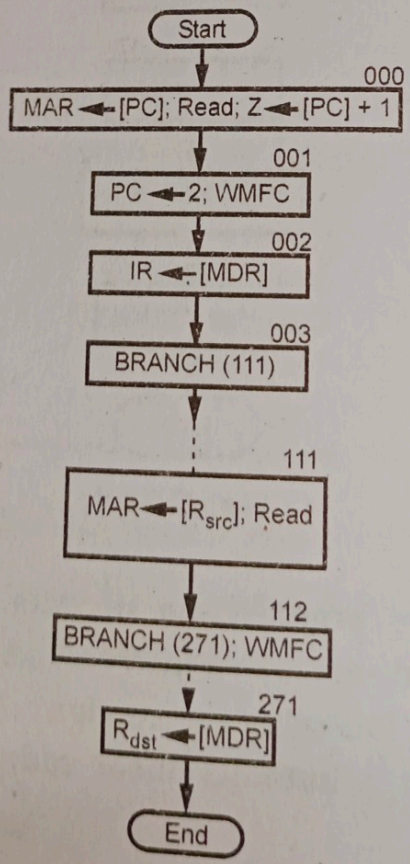


Fig. Q.35.1

Q.36 For a single bus organisation of CPU draw a flowchart and write microinstructions for instruction ADD (R_{src}), R_{dst}.

☞ [SPPU : Dec.-07,10,18, May-09,10,18, Marks 8]

Ans. : Microinstructions

Address (Octal)	Microinstruction
000	PC_{out}, MAR_{in}, Read, SelectC, Add, Z_{in}
001	Z_{out}, PC_{in}, Y_{in}, MWFC
002	MDR_{out}, IR_{in}
003	$\mu Branch \begin{cases} \mu pc \leftarrow (101)_8 \text{ from instruction decoder} \\ \mu pc_{5,4} \leftarrow [IR_{10,9}],\ \mu pc_3 \leftarrow [\overline{IR}_{10}] \cdot [\overline{IR}_9] \cdot [IR_8] \end{cases}$ i.e. $\mu pc_{5,4,3} \leftarrow (001)_2$
111	R_{srcout}, MAR_{in}, Read
	$\mu Branch \{\mu pc \leftarrow (170)_8$ from instruction decoder $\mu pc_0 \leftarrow \overline{IR}_8\}$, WMFC
112	$\mu Branch \{\mu pc \leftarrow (170)_8$ from instruction decoder $\mu pc_0 \leftarrow \overline{IR}_8\}$, WMFC
⋮	
171	MDR_{out}, Y_{in}
172	R_{dstout}, SelectY, Add, Z_{in}
173	Z_{out}, R_{dstin}, End

Table Q.36.1 Microinstruction for Add (R_{src}), R_{dst}

Flowchart

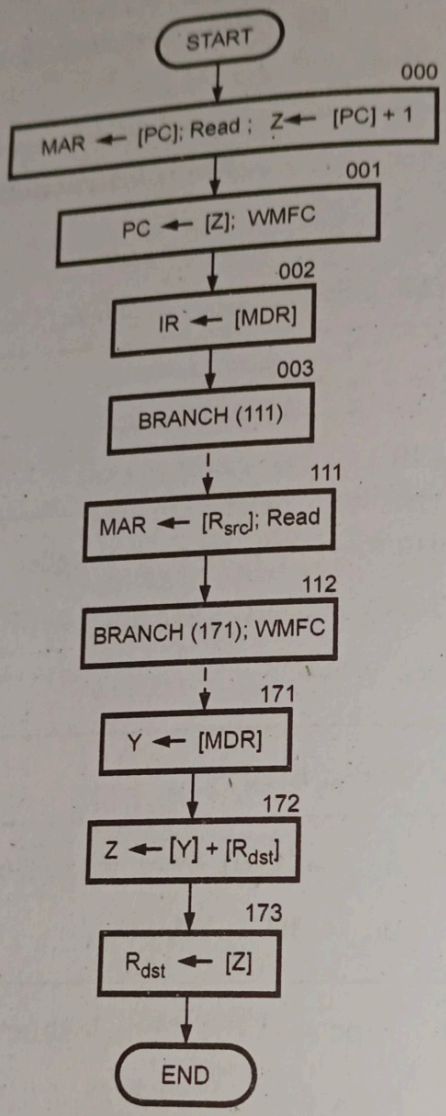


Fig. Q.36.1

Q.37 For a single bus organisation of CPU draw a flowchart and write microinstructions for instruction SUB (R_{src}), R_{dst}.

☞ **[SPPU : May-06,08, Dec.-05,12, Marks 8]**

Ans. : Microinstruction

Address (Octal)	
0 0 0	PC_{out}, MAR_{in},
0 0 1	Z_{out}, PC_{in}, Y_{in},
0 0 2	MDR_{out}, IR_{in}
0 0 3	µBranch {µpc µpc i.e. $\mu pc_{5,4,3}$
1 1 1	R_{srcout}, MA µBranch {µ µp
1 1 2	µBranch
⋮	
1 7 1	MDR_{out},
1 7 2	R_{dstout},
1 7 3	Z_{out}, R_{ds}

Table

Ans. : Microinstructions

Address (Octal)	Microinstruction
0 0 0	PC_{out}, MAR_{in}, Read, SelectC, Add, Z_{in}
0 0 1	Z_{out}, PC_{in}, Y_{in}, MWFC
0 0 2	MDR_{out}, IR_{in}
0 0 3	$\mu Branch \begin{cases} \mu pc \leftarrow (101)_8 \text{ from instruction decoder} \\ \mu pc_{5,4} \leftarrow [IR_{10,9}],\ \mu pc_3 \leftarrow [\overline{IR}_{10}] \cdot [\overline{IR}_9 \cdot [IR_8] \end{cases}$ i.e. $\mu pc_{5,4,3} \leftarrow (001)_2$
1 1 1	R_{srcout}, MAR_{in}, Read
	$\mu Branch \{\mu pc \leftarrow (170)_8$ from instruction decoder $\mu pc_0 \leftarrow \overline{IR}_8\}$, WMFC
1 1 2	$\mu Branch \{\mu pc \leftarrow (170)_8$ from instruction decoder $\mu pc_0 \leftarrow \overline{IR}_8\}$, WMFC
⋮	
1 7 1	MDR_{out}, Y_{in}
1 7 2	R_{dstout}, SelectY, Sub, Z_{in}
1 7 3	Z_{out}, R_{dstin}, End

Table Q.37.1 Microinstruction for Sub (R_{src}), R_{dst}

Flowchart

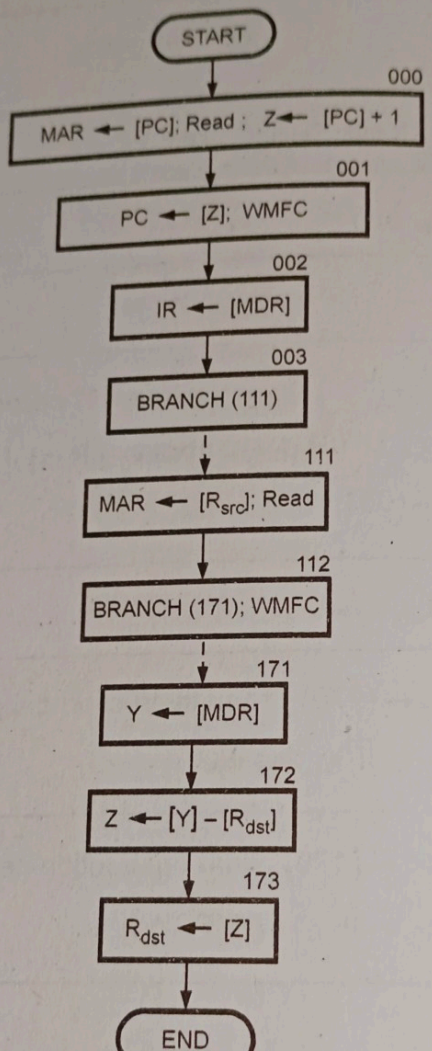


Fig. Q.37.1

Q.38 Give comparison between hardwired control and micro - programmed control.

Ans. :

Attribute	Hardwi
Speed	
Control functions	Implement
Flexibility	Not flexil new syst new inst
Ability to handle large/complex instruction sets	Somewh
Ability to support operating systems and diagnostic features	Very anticip
Design process	Som
Applications	Mos micr
Instruction set size	Usu inst
ROM size	
Chip area efficiency	Us

Ans. :

Attribute	Hardwired control	Microprogrammed control
Speed	Fast	Slow
Control functions	Implemented in hardware	Implemented in software
Flexibility	Not flexible, to accommodate new system specifications or new instructions.	More flexible, to accommodate new system specification or new instructions redesign is required.
Ability to handle large/complex instruction sets	Somewhat difficult	Easier
Ability to support operating systems and diagnostic features	Very difficult (unless anticipated during design)	Easy
Design process	Somewhat complicated	Orderly and systematic
Applications	Mostly RISC microprocessors	Mainframes, some microprocessors
Instruction set size	Usually under 100 instructions	Usually over 100 instructions
ROM size	—	2 K to 10 K by 20-400 bit microinstructions
Chip area efficiency	Uses least area	Uses more area

END...