UNIT 12: PROCESSOR STRUCTURE AND FUNCTION

UNIT STRUCTURE

12.1 Learning Objectives

12.2 Introduction

12.3 Processor Organization

12.4 Registrar Organization

12.4.1 User Visible Registers

12.4.2 Control and Status Registrar

12.5 Steps for Executing a Program

12.6 Instruction Cycle

12.7 Instruction Pipelining

12.8 Let Us Sum Up

12.9 Further Reading

12.10 Answers to Check Your Progress

12.11 Model Question

12.1 LEARNING OBJECTIVES

After going through this unit, you will be able to:

- learn about processor organization and functioning
- learn about types of user visible registers and their roles
- describe role of control and status registers in processor operations
- describe various steps of instruction cycle
- learn about basics of instruction pipeline.

12.2 INTRODUCTION

This unit discusses internal structure of processor organization. Register which forms the internal memory of a processor are analyzed in this unit. Various functions of registers including user visible register and control and status register are discussed in detail. Instruction cycle and the phases of instruction cycle are also described. The concept of instruction

pipeline which increases the performance of a system are also discussed at the end of this unit.

12.3 PROCESSOR ORGANIZATION

A processor consists of three major components namely Arithmetic Logic Unit (ALU), Control Unit (CU) and Registers. The figure 12.1 gives a simplified view of a processor. All the components within the processor are interconnected through buses. There are two types of buses: internal bus and external bus. The internal bus is responsible for transferring information within the components in the processor. On the other hand the external bus is responsible for transferring data to and from the processor to the memory unit and the input-output devices. The processors internal data bus is connected to external data bus by logic called bi-directional bus or transreceiver.

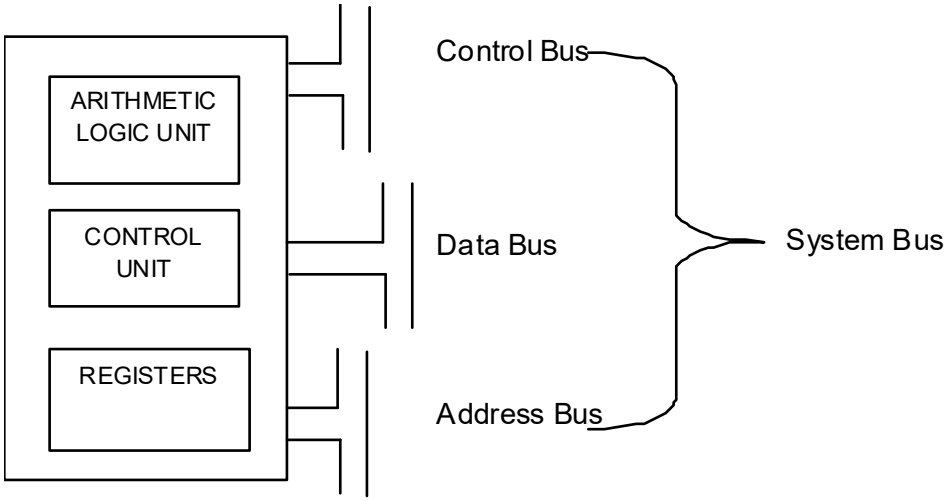


Figure 12.1: Simplified View of Processor Organization

The detailed diagram of a general processor organization is given in figure 12.2. The processor needs a small temporary memory to store data and instruction. It reads the instruction from memory, decodes it, fetches data, executes instruction and finally writes the result back to memory. For doing so it utilizes a set of registers. The ALU does the main processing of the data which are stored in the registers. The Control unit's responsibility is to control the operations of the ALU. The Control unit also checks the movement of data in and out of the ALU.

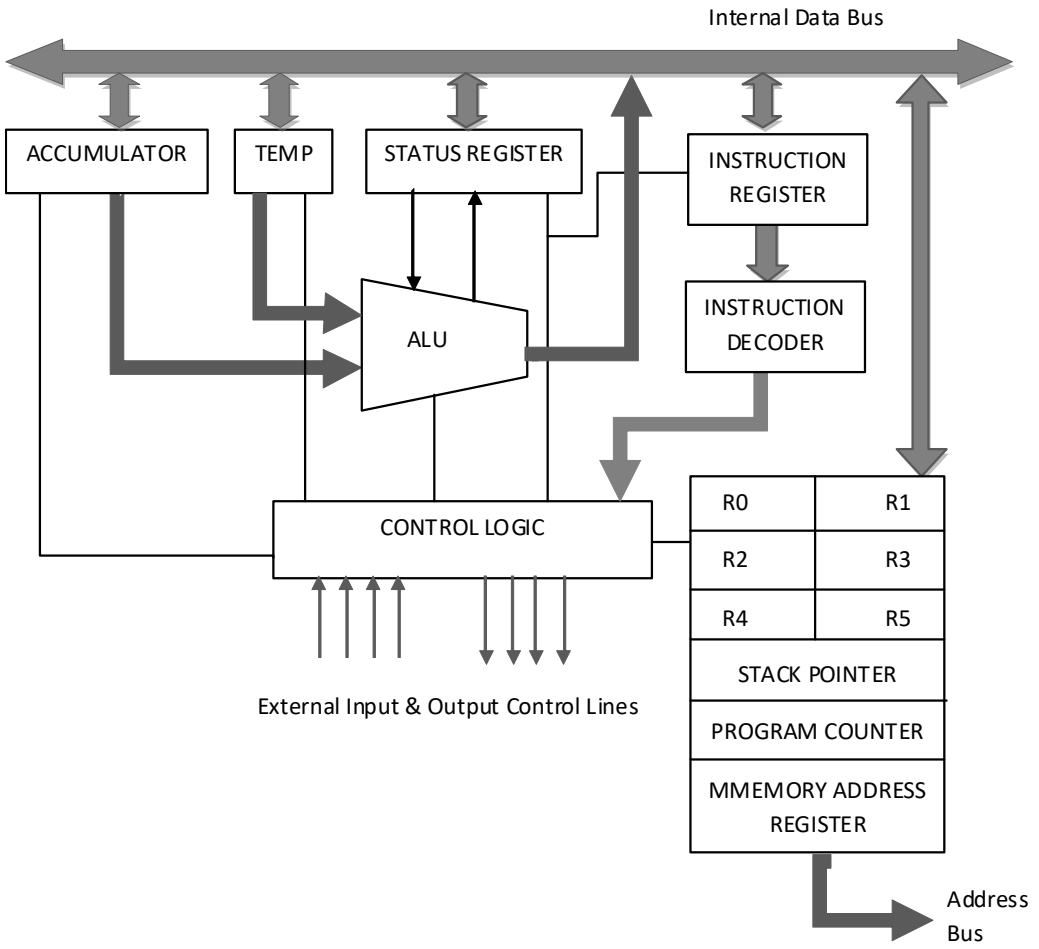


Figure 12.2: General Processor Organization

12.4 REGISTRAR ORGANIZATION

The processor consists of various registers and each register is used for different purposes. Following are the two categories of registers and their functionality.

12.4.1 User Visible Registers

These are the set of registers available to the assembly language programmers so that the main memory references can be minimized. Programmers can make the optimal use of the registers while writing a program. Following are the categories into which user visible register can be divided:

- General Purpose registers
- Accumulator
- Segment Register
- Index Register
- Stack pointer

General Purpose Registers: These are the set of registers ranging from R0 to R(n–1) that are available to the programmers. The programmers can directly make use of these registers in their programs to store the operands and the results. Use of the general purpose registers help minimize memory references therefore making the program execution faster.

Figure 12.3 show general register organization of seven processor registers. It shows how registers are selected by the bit combination of the three select A and select B lines and data is transferred from registers to ALU. Also how the registers are selected using the 3×8 decoders to store the final result. The process involves connecting output of each register to the multiplexer A and multiplexer B. The selection line A & B selects the register outputs and passes them to bus A & B. The two buses form the input to the ALU. The type of ALU operation to be performed on the input is selected by the select operation lines. Finally the result is stored in one of the register. The register selected for storing the result is done using 3×8 decoder.

For example the operation: R1 ← R2 + R3 are executed as

- SELECT A places the content of R2 into bus A
- SELECT B places the content of R3 into bus B
- SELECT OPERATION does the arithmetic addition of the contents of the bus A & B
- SELECT REGISTER is used to select register R1 and transfer the content of the output bus into R1.

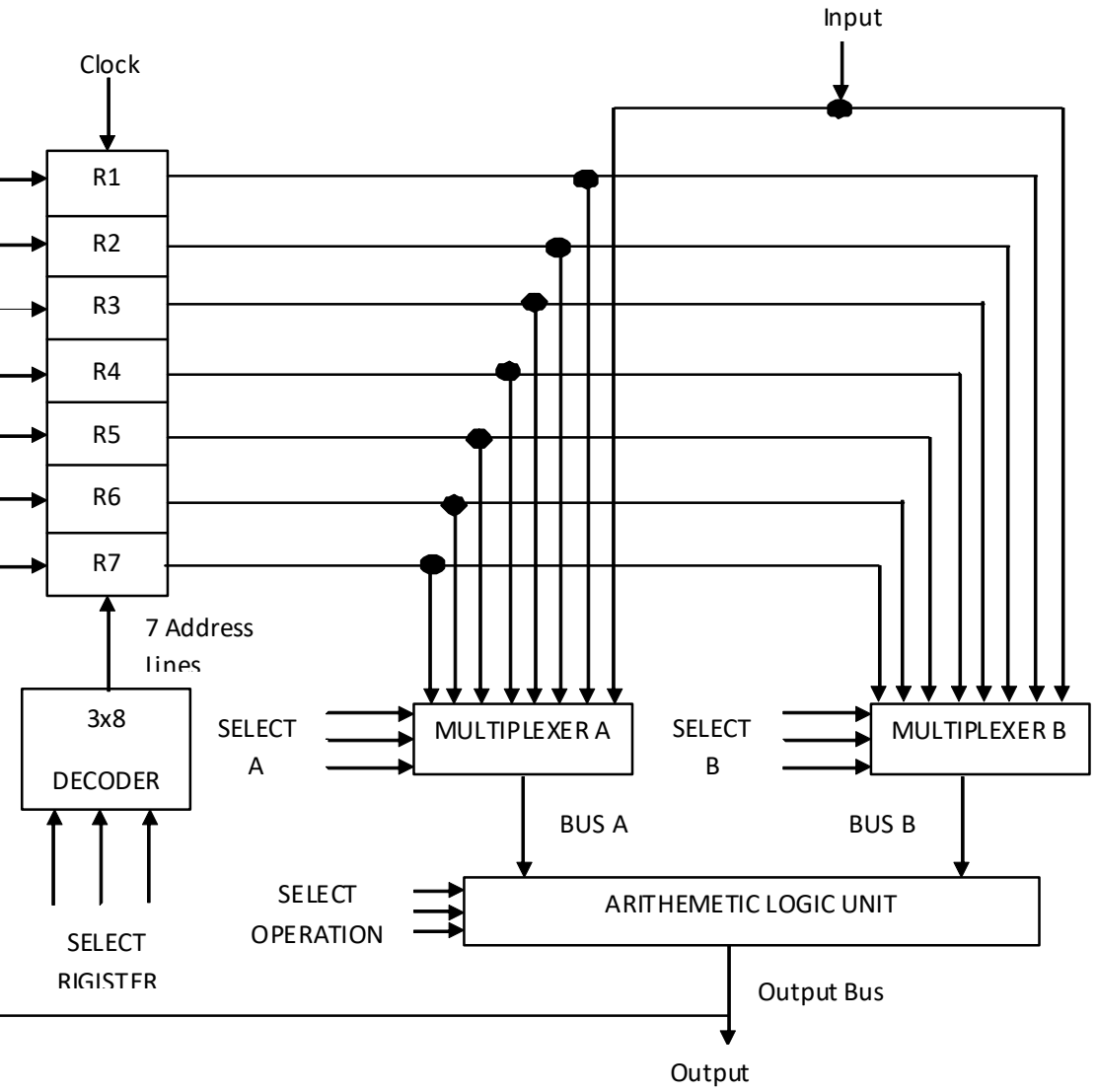


Figure 12.3: General Register Organization

The binary code for selecting register can be understood as:

Binary Code	SELECT A	SELECT B	SELECT REGISTER
000	Input	Input	–
001	R1	R1	R1
010	R2	R2	R2
011	R3	R3	R3
100	R4	R4	R4
101	R5	R5	R5
110	R6	R6	R6
111	R7	R7	R7

Accumulator: It is one of the major working register of a processor. For any operation involving two words in ALU, the accumulator holds one of the operand while the other operand is in another register or in memory. For certain operations like complement, the operand must be stored in the accumulator for complementing. Most of the time the result of operation is stored in accumulator which is later transferred to any other register or memory. Also the transfer of data between processor-memory and processor-input output device involves accumulator. It is user accessible and represent by 'A'.

Segment Register: A segment register is generally found in machine that supports segmented addressing. In such cases, the segment register is used to hold the starting address of the segment. There are three main segments namely code segment that contains all instructions of a program to be executed, data segment contains data and constants to be used by the program in the code segment and lastly the stack segment which stores the return address of procedures or functions. In addition to this there is one more segment called extra segment to store data.

Index Register: The machine that supports indexed addressing mode uses index register. For example SI is a source index register for string operations and DI is a destination index register for string operations.

Stack pointer: It is a user visible stack addressing register used to store the address of the top of the stack. It allows implicit stack addressing as the operations PUSH and POP does not need any explicit stack operand.

12.4.2 Control and Status Registers

These are the set of registers that are not visible to users but are required to control the operations in the processor. Following are the categories of these registers:

i) Program Counter

ii) Instruction Register

iii) Memory Address Register

iv) Memory Buffer Register also called Memory Data Register

v) Program Status Word or Status register or condition code register

i) **Program Counter (PC):** It is an important register used to inform the CPU the address of the next instruction to be executed. A program is a set of instructions which are executed by the CPU in a sequential order. Therefore the program counter helps the CPU by providing the address of the current instruction to be executed and also the address of the instruction to be executed next. The program counter is increment automatically to the next address once the CPU reads the content of program counter. The length of the program counter gives the maximum length of the program in bytes as for example a 16 bit program counter can address to the maximum 2^{16} bytes of memory.

ii) **Instruction Register (IR):** This register is used to store the instruction fetched from memory specified by the program counter, i.e., the instruction is available in the IR after it is fetched from memory.

iii) **Memory Address Register (MAR):** This register is used to store the address of the memory location. In a system the MAR connects directly to the address bus. Data are transferred with the help of MAR and MBR. The memory address from which or to which data are to be transferred is specified by the content of the MAR.

iv) **Memory Buffer Register (MBR):** Contains the data word to be written to memory or read from memory. It works along with MAR. It is also called memory data register. The data is available in the MBR or MDR after reading is completed.

v) **Program Status Word (PSW):** PSW register contains condition code or status information. When some operation is performed in the ALU, some of the status bits/flags in the PSW is either set/reset. Based on the flags the program can take some logical decisions, for example jump to a specified address when the zero flag is set. Following are the common PSW flags /bits:

Carry: Set to 1 if an operation results in a carry (addition) or borrow (subtraction) out of high order bit.

Zero: Set to 1 when result of an operation is zero.

Sign: Set to 1 if result of the last operation is negative.

Auxiliary Carry: Is set to 1 when addition of first four bits results in carry in the 5^{th} bit.

Equal: Set to 1 if logical comparison operation results in equality.

Overflow: Set to 1 if the result of addition of two signed numbers overflows into sign bit.

Parity: Set to 1 if the operation results in byte that contains even number of one's.

12.5 STEPS FOR EXECUTING PROGRAM

To execute a program/instruction stored in memory the processor goes through the following steps:

- **Fetch Instruction:** The task of fetching the instruction one at a time is achieved with the help of two registers namely Program counter and Instruction Register. The program counter keeps track of which instruction is being executed and what instruction will be executed next. The instruction register is used to store the instruction currently being executed. In addition to this two more registers MAR and MDR are used to transfer data between memory and processor.
- **Interpret Instruction:** After fetching the instruction, the next step is to decode or interpret the instruction so as to know what action needs to be taken. This step also identifies how many bytes the instruction is? This helps to identify how many numbers of bytes of data to read from memory.
- **Fetch Data:** At this step the bytes of data is read from memory or I/O device depending upon the type of instruction; one byte, two byte or three byte. If the instruction is of one byte then data is assumed to be in special register called accumulator. If instruction is of two bytes, first byte represent instruction and second byte represent data. If instruction is of three bytes, first byte represents instruction and second and third

byte represents data. So data of one or two bytes is fetched from memory and stored in processor registers at this step.

- **Process Data:** Once the data and the action to be taken on the data are available, the processor executes the instruction requiring some arithmetic or logical operations.
- **Write Data:** The result obtained from previous step will require the data to be written or stored in memory or I/O device.

12.6 INSTRUCTION CYCLE

It is a process by which a computer goes through a cycle of steps to read instruction and data from memory, execute the instruction and store the result. The figure 12.4 shows an instruction cycle. The instruction cycle is consists of the following four cycles:

- Fetch Cycle
- Indirect Cycle
- Execute Cycle
- Interrupt Cycle

The process starts with fetch cycle. The Interrupt Cycle is always followed by the Fetch Cycle Then Indirect Cycle is followed by the Execute Cycle. In case of fetch and execute cycles, the next cycle is dependent on the current state of the system.

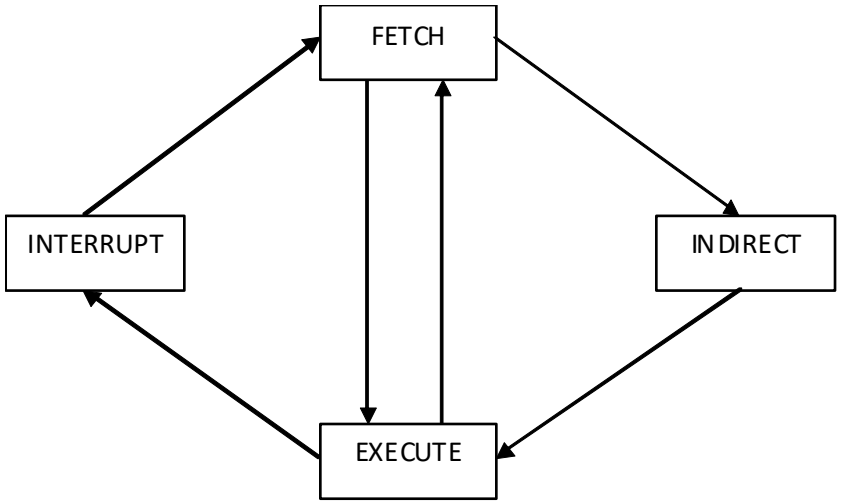


Figure 12.4: Instruction Cycle

The instruction cycle involves the following registers in the process of instruction execution.

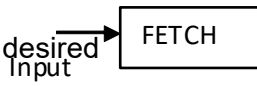
- Memory Address Register (MAR)
- Memory Buffer Register (MBR)
- Program Counter (PC)
- Instruction Register (IR)

Fetch cycle: In the beginning of any cycle, the instruction to be executed next is available in the Program Counter (PC). And following steps are taken:

- The content (address) of program counter is placed in the memory address register (MAR). The MAR is connected to the address bus of the system bus.
- The control unit then issues a READ command and the data from the memory specified by MAR is copied to data bus.
- Next the data bus content is copied to memory buffer register (MBR) and program counter (PC) is incremented by one, so that address of the next instruction is ready.
- Finally the MBR content is moved to instruction register (IR).

Indirect cycle: After fetching the instruction and storing it in instruction register, the next step is to fetch the operands by indirect addressing. If the operands are in register then no need to fetch the operands. Otherwise following steps are taken:

- Rightmost N bits of the Memory buffer register (MBR) containing the reference operand address is transferred to the MAR.
- Control unit issues a memory read instruction, and the desired operand address is moved to the MBR.



Execute Cycle: This cycle depends on the type of instruction in the IR. The instruction may be a data transfer instruction, a branch instruction, a memory read or write instruction, and an I/O read or write instruction or some arithmetic or logical operation in the ALU.

Interrupt Cycle: On completion of the execute cycle the processor checks if the any interrupt has occurred or not. If the interrupt is enabled and any interrupt occurs then the interrupt cycle takes place.

- Contents of program counter are transferred to memory buffer register and the contents are saved so that on completion of the interrupt cycle it can return to the point where it transferred the control to interrupt.
- Contents of the PC are saved to the address specified by MAR.
- Next PC is loaded with the starting address of the interrupt service routine.

On completion of the interrupt cycle, previously saved return address is used to fetch the next instruction.

12.7 INSTRUCTION PIPELINING

A pipeline is similar to an assembly line in a production factory. A product has to go through multiple stages in the assembly line before the final product is manufactured. At a time all the stages work simultaneously but on different stages of the product. This process is referred to as pipelining. In a computer system the technique of executing multiple instructions in an overlapped fashion is known as ***pipelining***. A pipeline consists of many stages and these stages are connected to one another in a pipe like structure. An instruction enters one end of the pipeline, goes through several stages before exiting from another end. Pipelining improves the overall throughput of the system.

In a pipeline system, each stage uses register to hold the output of that stage. Output of one stage is applied as input to the next stage.

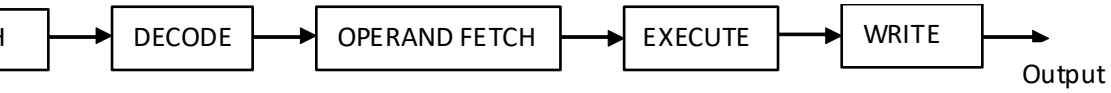


Figure 12.5: Five Stage Instruction Pipeline

Figure 12.5 shows an example of five stage instruction pipeline consisting of fetch, decode, operand fetch, execute and write stages. Here streams of instructions are executed in overlapped fashion thereby increasing the throughput of the computer system.

Figure 12.6 shows the timing diagram of an instruction pipeline. While the instruction pipeline reads one instruction from the memory, previous

instructions is executed in other stage of the pipeline. Thus multiple instructions are executed simultaneously. From the figure 12.6 it can be observed that while the first instruction started at time period one, the second instruction started at time period two and so on. Up to time period four not all stages were working simultaneously, but from time period five onwards all the five stages are working simultaneously. Therefore from instruction number five onwards each stage is working on a different instruction as:

Instruction 1: Write

Instruction 2: Execute

Instruction 3: Operand Fetch

Instruction 4: Decode

Instruction 5: Fetch

Time →

	1	2	3	4	5	6	7	8	9	10	11
Instruction 1	F	D	OF	E	W						
Instruction 2		F	D	OF	E	W					
Instruction 3			F	D	OF	E	W				
Instruction 4				F	D	OF	E	W			
Instruction 5					F	D	OF	E	W		
Instruction 6						F	D	OF	E	W	
Instruction 7							F	D	OF	E	W

Figure 12.6: Timing Diagram for Instruction Pipeline Operation

If there are ***k*** number of stages and ***n*** number of instructions, then total time ***T*** taken to execute n instructions can be given as:

$$T = k + (n–1)$$

The pipeline performs to its maximum if all the stages take same amount of time to process and there is no branch instruction in the program. In case of a branch instruction, figure 12.7 shows the timing diagram of the branch instruction pipeline assuming instruction four is a branch instruction. Here branch is not taken until the end of time period 7. At this point the pipeline clears all those instruction which are not useful i.e. instruction 5, 6 and 7. Also no instruction complete in time period 9, 10 and 11. This is known

as branch penalty as the processor could not anticipate the branch. So the instruction 17 assumed to be the instruction to be executed on the branch starts at time period eight.

Time → |←Branch Penalty→|

	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Instruction 1	F	D	OF	E	W									
Instruction 2		F	D	OF	E	W								
Instruction 3			F	D	OF	E	W							
Instruction 4				F	D	OF	E	W						
Instruction 5					F	D	OF							
Instruction 6						F	D							
Instruction 7							F							
Instruction 17								F	D	OF	E	W		
Instruction 18									F	D	OF	E	W	

Figure 12.7: Timing Diagram for Instruction Pipeline Operation



CHECK YOUR PROGRESS

Q.1: Choose the correct option:

i) A processor consists of–

a) Registers b) ALU, Control Unit

c) ALU, Control Unit, Registers

d) ALU, Control Unit, Registers, Memory

ii) Which of the following is not a user visible register–

a) Accumulator b) Segment Register

c) Index Register d) Program counter

iii) A program consists of the following segment registers–

a) Code, Data b) Code, Data, Stack

c) Code. Data, Stack, Extra

d) Code. Data, Stack, Segment

iv) The register that points to the top of the stack is called–

a) Stack register b) Stack pointer

c) Push Pop pointer d) All of the above

v) The register that is used to keep track of next instruction to be executed is–

a) Program Counter b) Instruction Register

c) Program Status Word d) Stack Pointer

vi) Consider subtraction of two floating point numbers 10101010 and 10101011. Which bit will be set?

a) Sign b) Carry

c) Overflow d) Both (a) and (b)

vii) What will be the auxiliary carry and parity bit if the number 10101011 is complemented

a) 1, 0 b) 0, 1 c) 1, 1 d) 0, 0

viii) Which of the following is not a valid flag?

a) Borrow b) Direction

c) Index d) Overflow

ix) The order in which phases of instruction cycle are executed is

a) Fetch cycle, Indirect cycle, Execute cycle, Interrupt cycle

b) Fetch cycle, Execute cycle, Indirect cycle, Interrupt cycle

c) Fetch cycle, Interrupt cycle, Execute cycle, Indirect cycle

d) Interrupt cycle, Fetch cycle, Execute cycle, Indirect cycle

x) Time taken to execute 10 instruction in a five stage pipeline is–

a) 12 b) 13 c) 14 d) 15



12.8 LET US SUM UP

- A processor consists of three major components namely Arithmetic Logic Unit (ALU), Control Unit (CU) and Registers.

- User Visible Registers are the set of registers available to the assembly language programmers so that the main memory references can be minimized.
- It is one of the major working register in the processor. For any operation in ALU involving two words, the accumulator holds one of the operand while the other operand is in another register or in memory.
- A segment register is generally found in machine that supports segmented addressing. In such cases, the segment register is used to hold the starting address of the segment.
- It is a user visible stack addressing register used to store the address of the top of the stack.
- It is an important register used to inform the CPU the address of the next instruction to be executed.
- Instruction Register is used to store the instruction fetched from memory specified by the program counter.
- Memory Address Register is used to store the address of the memory location.
- Memory Buffer Register Contains the data word to be written to memory or read from memory. It works along with MAR.
- Program Status Word register contains condition code or status information. When some operation is performed in the ALU, some of the status bits/flags in the PSW is either set/reset.
- To execute a program/instruction stored in memory the processor goes through following: fetch instruction, interpret instruction, fetch data, process data, and write data.
- It is a process by which a computer goes through a cycle of steps to read instruction and data from memory, execute the instruction and store the result.
- In a computer system the technique of executing multiple instructions in an overlapped fashion is known pipelining.



12.9 FURTHER READING

1) Hamacher, Carl. (5th Edition). *Computer Organization*. McGraw Hill.
2) Mano, M. Morris; Mall, Rajib. (3rd Edition). *Computer System Architecture*. Pearson.
3) Stallings, William. (10th Edition). *Computer Organization and Architecture: Designing for Performance*. Pearson.



12.10 ANSWERS TO CHECK YOUR PROGRESS

Ans. to Q. No. 1: i) (c), ii) (d), iii) (c), iv) (b), v) (a), vi) (d), vii) (d), viii) (c), ix) (a), x) (c)



12.11 MODEL QUESTION

Q.1: Explain the roles of user visible processor registers?

Q.2: Discuss segment registers role in a user program.

Q.3: What is a program status word? How different flag bits contribute in an ALU operation?

Q.4: What are the various phases of instruction cycle?

Q.5: Explain instruction pipeline using a suitable example.

Q.6: Briefly explain how instruction pipeline can contribute to conditional branch instruction.

Q.7: Explain how memory address register and memory buffer register help in data transfer operation.

Q.8: Using an example show the difference between Carry flag and Overflow flag.

Q.9: Consider addition of two 8-bit numbers 00110111 and 11100010. What will be the value of following flags: (a) Carry, (b) Auxiliary Carry, (c) Sign, (d) Zero, (e) Parity

*** ***** ***