USN | | | | | | | | | | |

Internal Assessment Test 1 – Dec. 2021

Sub:	Computer Organisation and Architecture					Sub Code:	18EC35	Branch:	ECE
Date:	20-12-2021	Duration:	90 Minutes	Max Marks:	50	Sem / Sec:	5/A,B,C,D		OBE

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1	Briefly Discuss types of Computers. Discuss the Functional units of computer with diagram.	[10]	CO1	L1,L2
2	Explain the basic concepts of computer with neat diagram of connection between memory and processor.	[10]	CO1	L2
3(a)	Discuss the functions of Software.	[05]	CO1	L3
3 (b)	Discuss the basic performance equation.	[05]	CO1	L1,L2
4	In detail discuss the straight line sequencing with an example program of computing [C]= [A]+[B]	[10]	CO1	L1,L2
5	Discuss any 5 addressing modes with examples.	[10]	CO1	L1, L2
6	Write short note on IEEE format of floating representation with example of 32 bit representation of 1259.125	[06]	CO1	L2
7	Explain in detail the basic IO operations for reading and displaying the character.	[10]	CO1	L1
8	Write a short notes on Stack and stack operations.	[10]	CO1	L2

USN | | | | | | | | | | |

Internal Assessment Test 1 – Dec. 2021

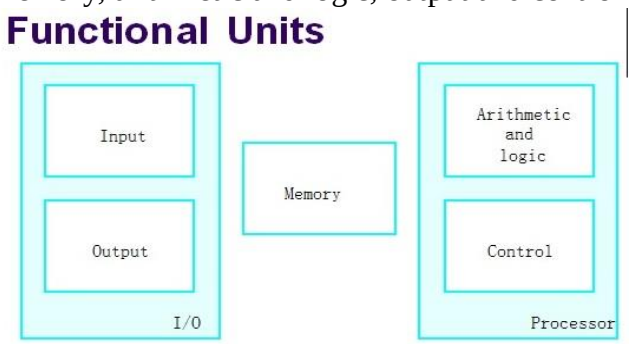
Sub:	Computer Organisation and Architecture					Sub Code:	18EC35	Branch:	ECE
Date:	20-12-2021	Duration:	90 Minutes	Max Marks:	50	Date:	20-12-2021		Duration:

	Answer any FIVE FULL Questions	MARKS	CO	RBT
1	Briefly Discuss types of Computers. Discuss the Functional units of computer with diagram.	[10]	CO1	L1,L2
2	Explain the basic concepts of computer with neat diagram of connection between memory and processor.	[10]	CO1	L2
3(a)	Discuss the functions of Software.	[05]	CO1	L3
3 (b)	Discuss the basic performance equation.	[05]	CO1	L1,L2
4	In detail discuss the straight line sequencing with an example program of computing [C]= [A]+[B]	[10]	CO1	L1,L2
5	Discuss any 5 addressing modes with examples.	[10]	CO1	L1, L2
6	Write short note on IEEE format of floating representation with example of 32 bit representation of 1259.125	[06]	CO1	L2
7	Explain in detail the basic IO operations for reading and displaying the character.	[10]	CO1	L1
8	Write a short notes on Stack and stack operations.	[10]	CO1	L2

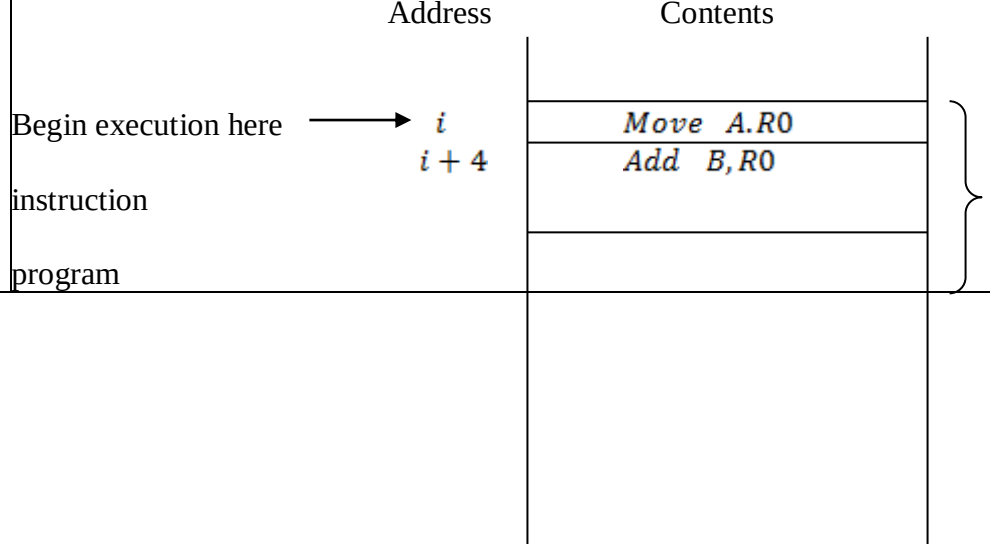
USN | | | | | | | | | | |

Internal Assessment Test 1 – Dec. 2021

Sub:	Computer Organisation and Architecture					Sub Code:	18EC35	Branch:	ECE
Date:	20-12-2021	Duration:	90 Minutes	Max Marks:	50	Sem / Sec:	5/A,B,C,D		OBE

	Scheme and Solution	MARKS	CO	RBT
1	Computers differ widely in size, cost, computational power and intended use. 1. Desktop computers have processing and storage units, visual display and audio output units and keyboard that can be located on a home or office desk. 2. Personal Notebook computers a compact version of personal computers with all components packaged into single unit the size of a thin briefcase. 3. Workstations with high resolution graphics input/output capability with the same dimension as desktop computers have more computational power. 4. A range of large and powerful computer systems exist that are called enterprise systems and servers at low end of the range and servers at high end.	04	CO1	L1,L2
	A computer consists of five functionally independent main parts: input, memory, arithmetic and logic, output and control units as shown in fig  Basic functional units of a computer The input unit accepts coded information from human operators, from electromechanical devices such as keyboards or from other computers over digital communication line. The information received is either stored in the computer memory for later reference or immediately used by the arithmetic and logic circuitry to perform the desired operations. The processing steps are determined by the program stored in the memory. Finally the results are shown on the output unit. All of these actions are co-ordinated by the control unit. We refer to the arithmetic and logic circuits in conjunction to the control circuits as the processor and input and output units are referred to as input-output (I/O) unit. *Instructions* or *machine instructions* are explicit commands that • Govern the transfer of information within a computer as well as between the computers and its I/O devices. • Specify the arithmetic and logic operations to be performed. A list of instructions that perform a task is called *program*. The computer is completely controlled by a stored program except for a possible external interruption by an operator or by I/O devices connected to the machine. Data is used to mean any digital information. Each number, character or instruction is encoded as a string of binary digits known as bits each having one of two	06		

	possible values 0 or 1.			
2	To perform a given task, an appropriate program consisting of a list of instructions is stored in the memory. Individual instructions are brought from the memory into the processor, which executes the specified operations. Data to be used as operands are also stored in the memory. Transfers between memory and processor are started by sending the address of the memory location to be accessed to the memory unit and issuing the appropriate control signals. The data is transferred to or from the memory. The memory and processor connection is shown in Fig. The Instruction register (IR) holds the instruction that is currently being executed. Its output is available to control circuits which generate the timing signals that control various processing elements involved in executing the instruction.The Program Counter (PC) holds the address of the next instruction to be fetched and executed. During the execution of an instruction, the contents of the PC are updated to correspond to the address of the next instruction to be executed. MAR and MDR facilitate communication with the memory.MAR (Memory Address Register) hold the address of the location to be accessed and MDR (Memory Data Register) contains data written into or read out of the addressed location.	04 06	CO1	L2
3(a)	System software is a collection of programs that are executed as needed to perform functions such as: • Receiving and interpreting user commands. • Entering and editing application programs and storing them as files in secondary storage devices. • Managing the storage and retrieval of files in secondary storage devices. • Running standard application programs such as word processors, spreadsheets or games with data supplied by users. • Controlling I/O units to receive input information and produce output results. • Translating programs from source form prepared by user into object form consisting of machine instructions. • Linking and running user written application programs with existing standard library routines such as numerical computational packages.	[05]	CO1	L3
3 (b)	The total time required to execute the program is known as *elapse time.* This		CO1	L1,L2

	is a measure of performance of entire computer system. The periods during which processor is active is used to measure the performance of processor. The sum of these periods is referred to as *processor time*. The processor time depends on the hardware involved in the execution of individual machine instructions. Processor circuits are controlled by a timing signal called *clock*. The clock defines regular time intervals called *clock cycles*. To execute a machine instruction, the processor divides the action to be performed into a sequence of basic steps, such that each can be completed in one clock cycle. The length P of one clock cycle is an important parameter that affects the processor performance. Its inverse is the clock rate, $R = 1/P$ which is measured in cycles per second. Let T be the processor time required to execute a program that has been prepared by some high level language. The compiler generates machine level object program that corresponds to source program. Assume that complete execution of the program requires the execution of N machine language instructions. Suppose that the average number of basic steps needed to execute one machine instruction is S, where each basic step is completed in one clock cycle. If the clock rate is R cycles per second, the program execution time is given by *basic performance equation.* $$T = \frac{N \times S}{R}$$ To achieve high performance, the value of T must be reduced which can be done by reducing N and S, and increasing R. The value of N is reduced if the source program is compiled in fewer machine instructions. The value of S is reduced if instructions have a smaller number of basic steps to perform or if the execution of instructions are overlapped. Using a higher-frequency clock increases the value of R which means the time required to complete a basic execution step is reduced	[05]		
4	a program segment in the memory of a computer. The word length is 32 bits and the memory is byte addressable. Each instruction is 4 bytes long, the second and third instructions start at addresses $i + 4$ and $i + 8$. The Program Counter (PC) contains the address of the instruction to be executed next. To begin executing a program, the address of its fist instruction must be placed in to PC. Then the processor control circuits use the information in the PC to fetch and execute the instructions, one at a time, in the order of increasing addresses. This is called *straight-line sequencing.* Executing a given instruction is a two phase-procedure. In the first phase called *instruction fetch*, the instruction is fetched from the memory location whose address is in the PC. This instruction is placed in the *instruction register* (IR) in the processor. At the start of second phase called *instruction execute*, the instruction in the IR is examined to determine which operation is to be determined.  	[07] [03]	CO1	L1,L2

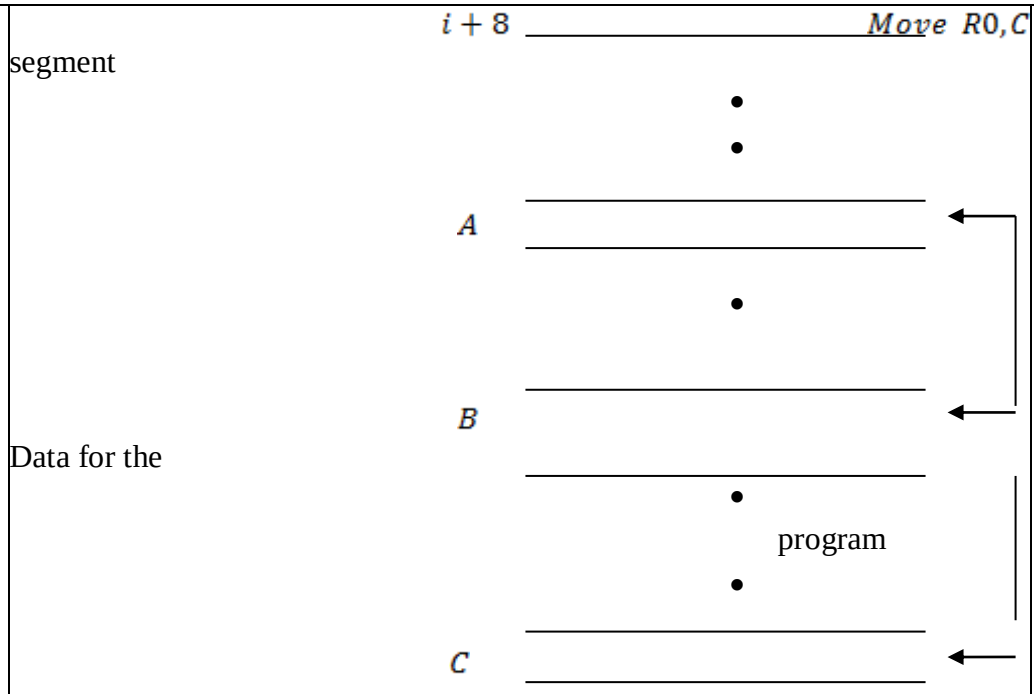


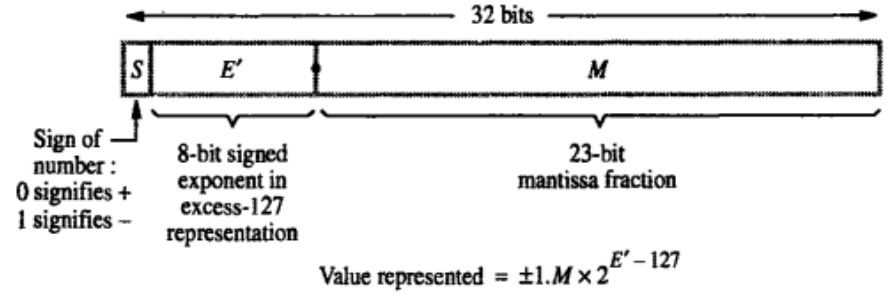
Fig 1.12: A program for $C \leftarrow [A] + [B]$

5 The different ways in which the location of an operand is specified in an instruction is known as *addressing modes*. [10] CO1 L1, L2

Name	Assembler syntax	Addressing function
Immediate	#Value	Operand = Value
Register	R*i*	EA = R*i*
Absolute(Direct)	LOC	EA = LOC
Indirect	(R*i*) (LOC)	EA = [R*i*] EA = [LOC]
Index	X(R*i*)	EA = [R*i*] + X
Base with index	(R*i*,R*j*)	EA = [R*i*] + [R*j*]
Base with index and offset	X(R*i*,R*j*)	EA = [R*i*] + [R*j*] + X
Relative	X(PC)	EA = [PC] + X
Autoincrement	(R*i*)+	EA = [R*i*]; Increment R*i*
Autodecrement	−(R*i*)	Decrement R*i*; EA = [R*i*]

6 A computer must be able to represent numbers and operate on them in such a way that the position of the binary point is variable and automatically adjusted as the computation proceeds. In such a case binary point is said to float and the numbers are called *floating-point numbers*. For example, in the familiar decimal scientific notation, the numbers can be written as $6.0247 \times 10^{23}, 6.6254 \times 10^{-27}$. The numbers are said to be given to five *significant digits*. When the decimal point is placed to the right of the first (nonzero) significant digit, the number is said to be *normalized*. The floating-point number is represented by its sign, string of significant digits, commonly called *mantissa*, and an exponent to an implied base for the scale factor. [05] CO1 L2

The standard for representing floating-point numbers in 32 bits is specified by IEEE is given in Fig 1.14. A 24 bit mantissa can represent a 7-digit decimal number and an 8 bit exponent. The sign of the number is given in the first bit, followed by the representation for the exponent (to the base 2) of the scale factor. The signed exponent component E is stored in the form of unsigned integer $E' = E + 127$. This is called excess-127 format.

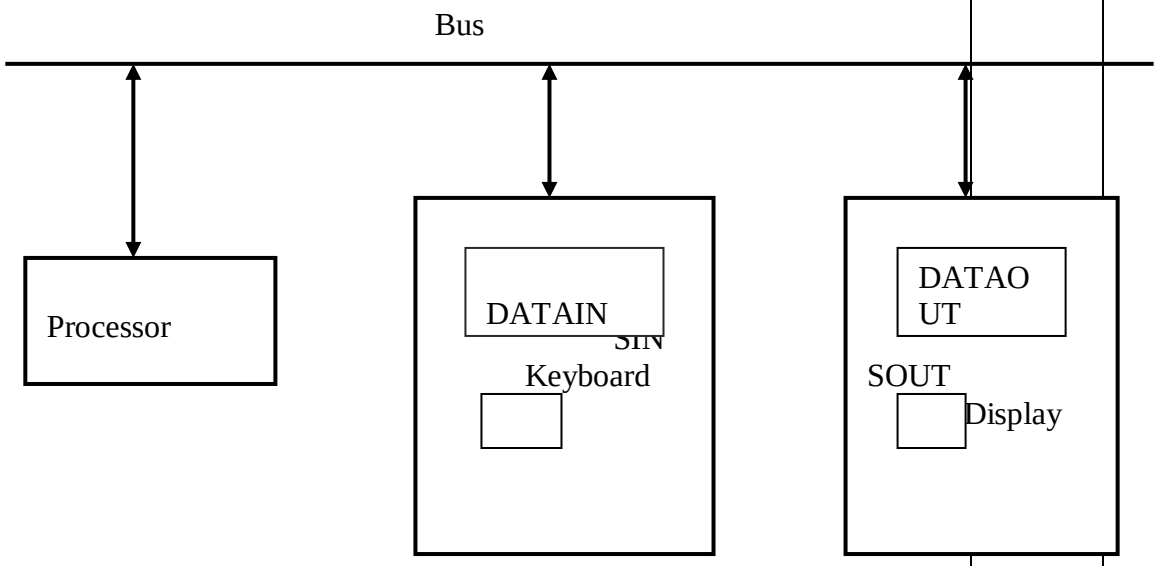


[05]

1259.125=0(S) 10001001(E) 00111010110010000000000(M)

7	problem of moving a character code from the keyboard to the processor. Striking a key store the corresponding character code in an 8-bit buffer register associated with the keyboard. Let us call this register DATAIN as shown in Fig 2.3. To inform the processor that a valid character is in DATAIN, a status control flag, SIN, is set to 1. A program monitor SIN, and when SIN is set to 1, the processor reads the contents of DATAIN. When the character is transferred to processor, SIN is automatically cleared to 0. If the second character is entered at the keyboard, SIN is again set to 1 and the process repeats.	[07]+[03]	CO1	L1

An analogous process takes place when the characters are transferred from the processor to display. A buffer register, DATAOUT and a status control register SOUT are used for this transfer. When SOUT equals 1, the display is ready to receive a character. Under program control, the processor monitors SOUT and when SOUT is set to 1, the processor transfers a character code to DATAOUT. The transfer of character to DATAOUT clears SOUT to 0, when the display is ready to receive a second character; SOUT is set again to 1. The buffer registers DATAIN and DATAOUT and the status flags SIN and SOUT are part of circuitry known as *device interface.*



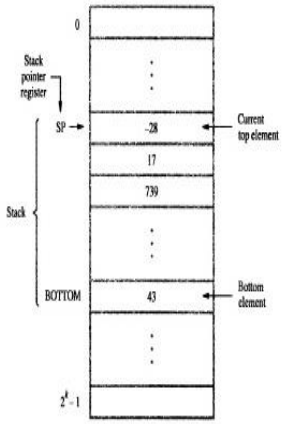
Bus connection for processor, keyboard and display

The processor can monitor the keyboard status flag SIN and transfer a character from DATAIN to register R! By the following sequence of operations:

READWAIT Branch to READWAIT if SIN=0
Input from DATAIN to R1

The first instruction tests the status flag and the second performs the

	branch. The processor monitors the status flag by executing a short *wait loop* and proceeds to transfer the input data when SIN is set to 1 as a result of key being struck. The input operation resets SIN to 0. The sequence of operations are used for transferring the output to display are **WRITEWAIT Branch to WRITEWAIT if SOUT=0** **Output from R1 to DATAOUT**			

8	Data operated on by a program can be organized in a variety of ways. A *stack* is a list of data elements usually words or bytes with the accessing restriction that elements can be added or removed at one end of the list only. This end is called the top of the stack and the other end is called the bottom. The structure is called *pushdown* stack. *Last-in-first-out* (LIFO) stack is a type of storage mechanism where the last data item placed on the stack is the first one removed when retrieval begins. The terms *push* and *pop* are used to describe placing a new item on the stack and removing top item from the stack. The stack grows in the direction of decreasing memory address. Fig shows a stack of word data items in the memory of a computer. It contains the numerical values, with 43 at the bottom and -28 at the top. A processor register is used to keep track of the address of the element of the stack that is at the top at any given time. This register is called the *stack-pointer* (SP).
	 
	
	Figure 2.21 A stack of words in the memory.
	
	byte-addressable memory with a 32-bit word length, the push operation can be implemented as
	Subtract #4,SP
	Move NEWITEM,(SP)
	where the Subtract instruction subtracts the source operand 4 from the destination operand contained in SP and places the result in SP. These two instructions move the word from location NEWITEM onto the top of the stack, decrementing the stack pointer by 4 before the move. The pop operation can be implemented as
	Move (SP),ITEM
	Add #4,SP
		[10]	CO1	L2